UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 6 BIOINFORMÁTICA

Trabajo de Diploma

Módulo de predicción de Actividad Biológica de compuestos orgánicos empleando Programación Genética.

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

Autor(es): Yuleidys Mejias Cesar

Yania Molina Souto

Tutor(es): M.C. Aurelio Antelo Collado

Dr. Ramón Carrasco Velar

Julio, 2007

Nunca consideres el estudio como una obligación,
sino como una oportunidad para penetrar en el bello
y maravilloso mundo del saber.
Albert Einstein

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 ____ días del mes de _____ del año _____.

_____	_____
Yania Molina Souto	Yuleidys Mejias Cesar
Autor	Autor
_____	_____
M.C. Aurelio Antelo Collado	Dr. Ramón Carrasco Velar
Tutor	Tutor

Datos de Contacto

Tutores:
M.C. Aurelio Antelo Collado
Universidad de las Ciencias Informáticas, Habana, Cuba.
Email: aantelo@uci.cu

Dr. Ramón Carrasco Velar
Centro de Química Farmacéutica, Habana, Cuba.
Email: ramón.carrasco@cqf.sld.cu

Agradecimientos

Queremos agradecer primeramente a nuestros tutores, por brindar de forma desinteresada sus conocimientos y ayudarnos a estar hoy aquí. Agradecer a todos aquellos que tuvieron que ver de una forma u otra con nuestra formación profesional, a nuestros padres, a nuestras amistades por los buenos momentos que pasamos y los recuerdos que hoy llevamos con nosotras. A Edel por su ayuda desinteresada, su paciencia y su amistad. A la profe Lesly, a Liesner, a Maypher y al profe Febles por sus consejos y por los momentos que de forma desinteresada nos regalaron para que pudiéramos graduarnos. A nuestro Comandante en Jefe Fidel por darnos la oportunidad de estudiar en una universidad de excelencia como esta.

Dedicatoria

A mi mamá y a mi papá, sabiendo que no habrá una forma en está vida de agradecer todo lo que han hecho por mí.

A mi mami además por ser mi mejor amiga, quien me toma de la mano cuando veo que los caminos se cierran y por ser la personita más dulce y tierna que he conocido jamás y a mi papi por sus consejos, su apoyo y su esfuerzo para que hoy pueda estar aquí, a ambos por el simple hecho de confiar en mí.

A mis abuelitos, que lo que más desean es estar aquí hoy conmigo, viendo mi sueño de ser una profesional a punto de realizarse.

A mis hermanos Yanet, Erier, Yady por el simple hecho de existir y hacerme feliz.

A mis dos bellas princesas Elizabeth y Helo, que llenan mi vida de luz.

A mis tíos y mis primos por ser conmigo como mis padres y hermanos y ser de esas personas que me han apoyado siempre sin esperar nada a cambio.

A una personita que se puso triste porque no lo puse en el lugar que merece estar.

A Yunet, a Suanly, a Nina, a Yuly, a Gretel, al Rafa por ser mis amigos más sinceros y a todas las demás amistades que hicieron que mis 5 años fueran felices en esta universidad.

A Edel por ser mi más fiel amigo y el que más confianza tenía en que hoy estaría aquí.

Gracias a todos los que están, y los que no, pero que no dejan de ser importantes para mí, por ayudarme a lograr mis expectativas, cumplir mis metas y ser hoy una profesional… Gracias!!!

Yania Molina Souto.

Dedicatoria

A mi madre, que además de ser preciosa, es el ser más dulce que existe sobre la faz de la Tierra (mamita,ojalá, fuera como tú). Siempre me ha apoyado y ha compartido todos mis estudios y mis sueños. Me brindó todo el amor maternal que se pueda recibir y me enseñó que la perseverancia y el esfuerzo son el camino para lograr los objetivos.

A mi papito lindo, que a pesar de estar muy lejos siempre me ha brindado todo su cariño y me ha dado fuerzas para enfrentar los problemas.

A mi hermanita, por su apoyo, amor y por la confianza que depositó en mí. Ella me ha dado el impulso para convertirme en la hermana mayor de la que siempre habla con tanto orgullo.

A Manza, por guiarme por el camino correcto, por brindarme su amor de padre y por sus sabios consejos, gracias a los cuales he podido alcanzar mi meta.

A Nina, Yania, Milton, Yanitza, Indira, Yanet, Lia, Yadira, Suanly y Yolaisy porque gracias a su cariño y apoyo constante he podido llegar a donde estoy.

A mi novio por su cariño, paciencia y constante estímulo.

A Edel por su permanente disposición y ayuda desinteresada. Sin su ayuda no hubiera podido llegar hasta aquí.

A mis compañeros por su continuo y afectuoso aliento.

Y a todos los que de una forma u otra contribuyeron a que yo pudiera realizar mi máxima aspiración, convertirme en una profesional.

Muchas gracias a todos !!!

Yuleidys Mejias Cesar.

Resumen

El presente trabajo forma parte del proyecto de investigación conjunta entre el Centro de Química Farmacéutica y la Facultad de Bioinformática de la Universidad de las Ciencias Informáticas titulado: Plataforma Inteligente para la Predicción de Actividad Biológica de Compuestos Orgánicos. La misma cuenta con varios módulos independientes, entre ellos los módulo de técnicas de Inteligencia Artificial (IA). Estos módulos son los encargados de generar los diferentes modelos de predicción y predecir la actividad biológica en compuestos orgánicos, utilizando dos modos de descripción de la estructura química (fragmentos y descriptores). Se analizó, diseñó e implementó un módulo de programación genética para el desarrollo de modelos predictivos, y de predicción de actividad biológica de compuestos orgánicos basado en fragmentos ponderados por el Índice del Estado Refractotopológico Total. Se trabajó con varias muestras de tamaño variable con actividad anticancerígena reportada, tomada de la base de datos del National Cancer Institute.

INDICE

Introducción

El auge de la industria médico-farmacéutica en los siglos XX y XXI es consecuencia de la necesidad de desarrollar y aplicar métodos que ayuden a obtener productos más potentes, más específicos, con menos efectos colaterales y sobre todo más seguros, con un gasto mínimo de recursos.

En el campo de la Química Medicinal, las empresas farmacéuticas invierten aproximadamente entre 10 y 12 años para introducir en el mercado un nuevo fármaco. Además debe tenerse en cuenta, el tiempo que demora el nuevo medicamento en ser aprobado por las entidades competentes, para lo cual también se invierte una gran cantidad de dinero y recursos.

La mayoría de las nuevas moléculas candidatas a nuevos fármacos surgen de la investigación minuciosa de científicos con el propósito de alcanzar un determinado efecto en el cuerpo humano. En algunos casos los científicos basan sus estudios en sustancias que se encuentran en la naturaleza, en otros solo son el resultado de la concepción genial del investigador. Lo cierto es que se necesitan largos períodos de tiempo, y las inversiones sobrepasan los millones de dólares. (1)

Johnson & Johnson es una de las primeras empresas del mundo en el sector del Cuidado de la Salud, y posee varias compañías farmacéuticas de gran prestigio internacional como Janssen-Cilag. Esta compañía posee un importante Centro de Investigación Química en Toledo y estudios realizados en el mismo revelan que cada semana son sintetizadas 45 nuevas moléculas, diseñadas minuciosamente para interferir en uno u otro proceso biológico. De ellas, solo una de cada cinco mil, después de diez años de investigación multidisciplinaria, resultará un fármaco. (1)

En los últimos años, la industria farmacéutica ha reorientado sus investigaciones y prestado más atención a aquellos métodos que permitan una selección racional o diseño de nuevos compuestos con propiedades deseadas. Muchos de esos enfoques están basados en la interrelación estructura química-actividad biológica de las moléculas.

La actividad biológica de un fármaco está condicionada, por su estructura química, la cual determina su actividad intrínseca o su potencia, causa fundamental de que muchas de las

investigaciones farmacológicas estén dirigidas a estudios de correlación estructura química-actividad biológica. Estos estudios permiten identificar aquellos rasgos y propiedades estructurales que son responsables de la actividad biológica de un compuesto químico y del diseño de nuevas moléculas con interés biológico.

Los primeros pasos en esa dirección se dieron con el surgimiento de las técnicas QSAR (Quantitative Structure-Activity Relationships) en el año 1962, que aparejado al acelerado desarrollo de las tecnologías de la computación y de la programación dieron paso a su vez a la posibilidad de enfrentar nuevos y más complejos problemas dentro de esta nueva disciplina, el Diseño Racional de Fármacos. (2)

En la actualidad, una parte de las investigaciones se han centrado en establecer estas relaciones con el empleo de técnicas de inteligencia artificial (IA), estadística, técnicas de reducción de datos, árboles de decisión, entre otras, con las que se obtienen resultados positivos. Es por eso que resulta interesante disponer de una herramienta, que haciendo uso de estas técnicas, sea capaz de predecir cuáles serán las características físicas, químicas, farmacológicas, etc., de los posibles candidatos a fármacos, a fin de restringir el campo de búsqueda, privilegiando así las moléculas más prometedoras, con lo cual se acortan los plazos de ejecución del proceso de investigación-desarrollo de nuevos fármacos.

Actualmente existen diferentes herramientas concebidas para el diseño de fármacos como el ADAPT, APEX, Accelerys, Insigth II, BioViz/ChemFinder 9.0, etc., que se basan en diferentes técnicas de procesamiento de la información, en métodos químico-cuánticos y en técnicas de inteligencia artificial.

La gran mayoría de estas herramientas, sobre todo las de mayor prestigio, son muy costosas e inaccesibles para Cuba o inadecuadas para resolver el problema que se presenta en el proyecto, debido a que ninguna se basa en el Índice del Estado Refractotopológico Total para describir los fragmentos - descriptor desarrollado en Cuba recientemente -, siendo de interés para el proyecto predecir la actividad anticancerígena a partir de este índice, lo que podría inducir alguna mejora en los nuevos modelos encontrados con respecto a los ya existentes.

El presente trabajo tiene su origen dentro del proyecto de investigación conjunta del Centro de Química Farmacéutica (CQF) y la Facultad 6 de la Universidad de las Ciencias Informáticas (UCI)

denominado “Plataforma Inteligente para la Predicción de Actividad Biológica de Compuestos Orgánicos”. La plataforma cuenta con varios módulos implementados de forma independiente que se conectarán a la aplicación central como plug-ins. Dentro de los módulos se encuentran el Módulo de Inteligencia Artificial (I) y (II), con el objetivo de predecir actividad biológica anticancerígena utilizando diferentes técnicas de IA, a partir de la descripción de la molécula por diferentes vías (fragmentos o descriptores) siendo una necesidad del proyecto conocer los resultados de la predicción a través de estas dos vías, para luego ser analizados. Las técnicas de IA que se decidió analizar fueron lógica difusa, máquina de soporte vectorial y programación genética. En esta última se centrará nuestro trabajo.

Una vez mostrada la situación a que nos enfrentamos estamos en condiciones de plantearnos nuestro problema científico:

¿Qué herramienta rápida y eficiente, es posible desarrollar para acortar la etapa inicial de búsqueda de nuevos fármacos utilizando técnicas de inteligencia artificial?

Es por ello que se evaluó la necesidad de desarrollar una herramienta capaz de predecir la actividad anticancerígena de una molécula a partir de sus fragmentos. El enfoque estructural utilizado consistió en la fragmentación de las moléculas en fragmentos ponderados por el Índice del Estado Refractotopológico Total, utilizando Programación Genética (PG). Teniendo como objeto de estudio la Inteligencia Artificial aplicada a la predicción de actividad biológica y como campo de acción la PG aplicada al estudio de la relación estructura-actividad anticancerígena de compuestos orgánicos utilizando fragmentos ponderados por el Índice del Estado Refractotopológico Total.

Por lo que se trazó como objetivo general: Desarrollar un módulo para la Plataforma capaz de predecir actividad anticancerígena en compuestos orgánicos a partir de fragmentos ponderados por el Índice del Estado Refractotopológico Total, utilizando Programación Genética como técnica de inteligencia artificial.

A partir de un análisis del objetivo general se derivan los siguientes objetivos específicos:

- Analizar un módulo basado en programación genética que permita predecir
- estructura-actividad.
- Diseñar el módulo analizado.

- Implementar el módulo diseñado.
- Comprobar los modelos encontrados.
- Para alcanzar estos objetivos se llevaron a cabo las siguientes tareas:
- Revisión del estado del arte acerca de sistemas de predicción existentes en el mundo.
- Análisis del estado del arte que permita dejar definida la posición del investigador respecto al uso de la Programación Genética.
- Análisis y diseño de la solución propuesta.
- Implementación del módulo de predicción de actividad anticancerígena en compuestos orgánicos.
- Realización de pruebas para comprobar los modelos generados, comparando los valores experimentales obtenidos con valores reales.

El presente documento se estructura en:

Capítulo 1: Fundamentación teórica

En este capítulo se brinda una breve reseña histórica sobre las técnicas QSAR que establecen las relaciones entre la estructura molecular y las propiedades de las sustancias. Se muestra una breve explicación del Índice del Estado Refractotopológico Total y se presentan los resultados del estudio bibliográfico realizado sobre la Programación Genética. Se describen las tendencias actuales para desarrollar la herramienta, así como algunas de las tecnologías y se justifica su uso en el desarrollo de la aplicación.

Capítulo 2: Descripción del sistema actual.

Se explican las características específicas del algoritmo de Programación Genética que se utilizó. Mediante los componentes del modelo de dominio de la metodología RUP se describe la solución propuesta. Se aborda lo referente a las reglas del negocio, requisitos funcionales y no funcionales y por último se muestra el Diagrama de Casos de Uso del Sistema para la aplicación que se desarrolló, así como la descripción de cada uno de sus Casos de Uso.

Capítulo 3: Características del sistema

En este capítulo se entra en el flujo de trabajo de Análisis y Diseño mostrándose los diagramas de clases del análisis por cada Caso de Uso. En el Diseño se muestran los diagramas de interacción y de clases del diseño separados por Casos de Uso, además de la descripción de

cada una de las clases del diseño. Se muestra por paquetes el diagrama de componentes y se describen los patrones de diseño que se usaron en el desarrollo del sistema, así como el estilo arquitectónico que utiliza la aplicación.

Capítulo 4: Análisis de los resultados

Se realizan pruebas experimentales con datos reales reportados en la base de datos del National Cancer Institute y se analizan los resultados obtenidos en cada uno de los experimentos de forma individual. Son analizados los porcientos de predicción, los errores en los modelos y por último la influencia de los parámetros y las operaciones, en la calidad de los modelos matemáticos generados.

Capítulo 1: Fundamentación Teórica.

1.1 Introducción.

Durante muchos años los químicos han tratado de encontrar una relación entre la estructura molecular y determinadas propiedades de las sustancias. A finales del siglo XIX, Richet, formula la idea de que es posible relacionar las variaciones estructurales, en una serie de ligandos, con variaciones en la actividad de forma cuantitativa, a través de la ecuación (3):

$$\Delta\Phi = f(\Delta C)$$

Siendo ***C*** la estructura química y Φ la medida de actividad biológica, comenzando así la era de los estudios QSAR (Quantitative Structure-Activity Relationships). Pero no fue hasta principio de los años 60 que se realizan las primeras aproximaciones QSAR realizadas con éxito en el diseño de nuevas moléculas. Desarrollándose 2 métodos, desde puntos de vistas teóricos diferentes:

La aproximación de Hansch [4, 5]**:** que supone que la energía libre de unión ligando-receptor se puede aproximar mediante una combinación lineal de contribuciones lipofílica, electrónica y estérica. Relacionando así actividad biológica con parámetros físico-químicos relativos a la lipofilia y la electronegatividad a través de ecuaciones del tipo:

$$log\ 1/C = k_1\ logP + k_2\ \sigma + k_3$$

La aproximación de Free-Wilson [6]: Trataba de considerar las aportaciones que hacen a la actividad de un compuesto cada uno de sus sustituyentes químicos χ_i localizados en posiciones ***j*** de la estructura μ: $log\ 1/C = \Sigma\chi_i + \mu$

El método aditivo de Free-Wilson fue pensado para construir moléculas nuevas a partir de la unión de sus fragmentos pero solo pudo ser utilizado en estudios de relación estructura-actividad de series congenéricas.

La necesidad de representar la estructura molecular por un simple número, se origina a partir del hecho de que las propiedades moleculares o actividades de un compuesto son representadas como números, y por lo tanto, al lograr representar de la misma forma la estructura y la actividad en un compuesto, los modelos QSAR se reducen a una correlación entre dos conjuntos de números, relacionados por una expresión algebraica (un conjunto de números representan las propiedades y el otro representa las estructuras moleculares de las moléculas que se estudian).(4)

En sus inicios las técnicas QSAR fueron empleadas en áreas como la agroquímica y la farmacología, en la actualidad resultan ser de amplia aplicación en química medicinal, en el diseño de fármacos y en estudios toxicológicos, asistidos por computadora.

1.2 Descriptores.

Dentro del modelado molecular, el diseño de nuevos compuestos resulta un tema interesante. Los métodos QSAR han demostrado que las relaciones entre la estructura molecular y las propiedades físico-químicas de los compuestos se pueden cuantificar matemáticamente a partir de parámetros estructurales simples, conocidos como descriptores.

En los últimos años, la industria farmacéutica ha dirigido una parte de sus esfuerzos al diseño racional de fármacos a través de métodos computarizados. La validez de estos métodos depende en gran medida de los descriptores utilizados para caracterizar la estructura química de un compuesto y de la calidad de los modelos matemáticos que describen los fenómenos biológicos.

Dentro de los descriptores más utilizados se encuentran los índices topológicos que se basan únicamente en la estructura 2D o topología de la molécula, los cuales se derivan generalmente de las matrices de conectividad o de distancia del grafo molecular. (5)

Se distinguen también los índices topográficos, los cuales incluyen además, otras propiedades estructurales de los átomos implicados y los índices basados en la teoría de la información. En general, estos índices contienen información relacionada con la forma molecular, el grado de ramificación, tamaño molecular y la flexibilidad estructural. Entre los más conocidos se destacan los índices de conectividad molecular, propuestos por Randic y desarrollados en profundidad por

Hall y Kier. Son rápidos de calcular y se ha comprobado que correlacionan con diferentes propiedades químico-físicas y biológicas. (5)

Recientemente se desarrollaron, en Cuba, dos nuevos índices, el de Partición de la Refractividad Molecular (molecular) y el Índice del Estado Refractotopológico (atómico), que parten de la matriz de conectividad del grafo químico completo, ponderado por la refractividad atómica tal y como la definieron Ghose y Crippen. Se emplearon para su definición los algoritmos para el cálculo de los índices de Randic y el del Estado Electrotopológico, respectivamente. Por su naturaleza no se consideran índices topológicos puros, sino híbridos pues poseen información químico-física adicional que los modifican. (6)

1.2.1 Índice del Estado Refractotopológico Total.

Se define como la suma de los valores del Índice del Estado Refractotopológico de cada átomo del fragmento considerado en una molécula dada. El Índice del Estado Refractotopológico se desarrolla a partir de la teoría del grafo químico y de la partición de la refractividad atómica definida por Ghose y Crippen. El índice se basa en la influencia de las fuerzas de dispersión de cada átomo sobre cada uno de los restantes en la molécula, modificado por la topología molecular.

Definición de R_t

El ***R-state*** o R_t , para un átomo ***i*** se define por la ecuación: $R_t = AR_t + \Delta AR_t$

Donde AR_t es el valor de refractividad intrínseco del átomo ***i*** y ΔAR_t es un término de perturbación definida por la ecuación: $\Delta AR_i = (Ar_i - AR_j)/ r_{2ij}$

Donde se suman todos los vértices ***j*** adyacentes en el grafo, AR_t y AR_j son los valores intrínsecos de la refractividad de los átomos ***i*** y ***j***, respectivamente, y r_{2ij} es el número de átomos del camino más corto entre los átomos ***i*** y ***j***, incluyendo tanto a ***i*** como a ***j***. Al igual que en el ***Estate*** y en el ***S-state***, la distancia topológica cuadrática indica que debe haber una disminución de la interacción, con el aumento de la distancia de separación entre los átomos.

Características de R_t

A diferencia de otros índices topológicos, los cuales no consideran la influencia de los átomos de hidrógeno, este índice sí incluye sus contribuciones al valor intrínseco de los átomos pesados. Esta inclusión refleja la capacidad potencial de interacción total del grupo con una supuesta

proteína ligando y esto es importante no solamente para el valor intrínseco del átomo al cual esta directamente enlazado sino para los otros grupos presentes en la molécula.

En el caso del Índice de Estado Refractotopológico Total, se analiza el papel de las fuerzas de dispersión de London a escala de cada átomo en la molécula, considerando la influencia de su entorno molecular.

En el presente trabajo se utilizará el Índice del Estado Refractotopológico Total para la construcción de un modelo matemático de predicción de la actividad anticancerígena de compuestos orgánicos en fragmentos. (6)

1.3 Modelos Matemáticos.

Un modelo matemático se define como una descripción desde el punto de vista de las matemáticas de un hecho o fenómeno del mundo real, con el objetivo de entender ampliamente el fenómeno y poder predecir su comportamiento en el futuro.

En la actualidad, para entender los problemas biológicos, médicos, para hacer estudios epidemiológicos y observacionales, se utiliza ampliamente la modelación matemática, debido a que permite establecer relaciones formales entre las variables presentes en el problema. (7)

Se puede concluir entonces que un modelo matemático es la función matemática que permite establecer una relación entre un conjunto de variables X_1 , X_2 ... X_n y una variable dependiente ***Y***. [7,10] En el caso específico que se está estudiando, el modelo está expresado en términos de los Índices del Estado Refractotopológico Total de cada uno de los fragmentos que se utilizan y que constituyen los términos independientes o ***"Xs"*** del problema y la actividad biológica como variable dependiente.

Para ejemplificar cómo encontrar el modelo matemático para el problema específico que se quiere resolver, se pueden señalar las siguientes etapas:

- Obtener datos experimentales, o sea, todos los fragmentos con sus respectivos índices calculados y su actividad biológica de la base de datos.

- Correlacionar los datos utilizando una técnica de IA (Programación Genética) y obtener el modelo QSAR.
- Realizar pruebas de predicción para comprobar el modelo.
- Finalmente se tiene el modelo buscado para predecir la actividad biológica en nuevos fragmentos que se necesite analizar.

1.4 Técnicas utilizadas para la construcción de modelos matemáticos para la predicción.

Existen varias técnicas para establecer relaciones estructura química–actividad biológica con el fin de encontrar modelos predictivos, unas con resultados más alentadores que otras, pero todas con resultados positivos. Dentro de las técnicas utilizadas para establecer relaciones QSAR se encuentran (8) (9):

- Los métodos estadísticos:
  - ✓ Regresión.
  - ✓ Técnicas de reducción de datos.
- Modelos basados en árboles de decisión.
- Modelos QSAR – 3D.
- Redes neuronales.
- Lógica Difusa.
- Máquinas de soporte vectorial.
- Computación Evolutiva.
  - ✓ Programación Evolutiva.
  - ✓ Estrategias Evolutivas.
  - ✓ Algoritmos Genéticos.
  - ✓ Programación Genética.

1.5 Computación Evolutiva

Dentro de las ramas de la Inteligencia Artificial, la Computación Evolutiva ha obtenido un gran prestigio en problemas de clasificación y optimización. Partiendo de la Teoría de la Evolución de Darwin, vista como un proceso adaptativo de optimización, sugiere un modelo en el que poblaciones de estructuras computacionales evolucionan mediante la aplicación de operadores

análogos a los utilizados en la naturaleza con el fin de mejorar el desempeño de la población. Este modelo se asocia a la función objetivo que se quiere optimizar, y se busca mejorar dicho desempeño en la medida en que el valor de la función se aproxima al mejor valor posible (óptimo global).

Los métodos más conocidos en la actualidad, englobados por la Computación Evolutiva son los Algoritmos Genéticos, la Programación Evolutiva, la Programación Genética (PG), y las Estrategias Evolutivas, los cuales presentan características que los vuelven muy atractivos en la resolución de problemas de gran porte (10). Entre tales características pueden citarse:

- **Transportabilidad:** los algoritmos que simulan procesos naturales pueden ser implementados en cualquier arquitectura de computadoras.
- **Robustez:** el algoritmo es capaz de hallar muy buenos resultados para una amplia gama de problemas.
- **Implementación relativamente fácil.**

Si se desea solucionar un problema aplicando cualquiera de los métodos englobados dentro de la Computación Evolutiva (algoritmos genéticos, estrategias evolutivas, programación genética, etc.) se utiliza un paradigma de Algoritmo Evolutivo (AE) que posee especificaciones diferentes para cada método pero que de forma general utiliza técnicas de búsqueda estocásticas (probabilísticas) en el espacio de posibles soluciones a un problema dado, basado en la abstracción de varios procesos de la Teoría de la Evolución darwiniana. Los AEs aplican dos principios básicos de la evolución natural:

1. Si por medio de un procesamiento genético se crea un individuo cuyo ajuste está por encima de la media, su período de supervivencia será mayor a la media y por tanto tendrá más oportunidades que un individuo medio de producir descendencia que cuente con algunos de sus rasgos destacados. (11)

2. Un individuo cuyo ajuste a la media es inferior, su probabilidad de supervivencia es menor a la media por lo que terminará siendo eliminado de la población. (11)

Los AEs poseen una serie de características propias de su funcionamiento, tales como:

- Trabajan con varios individuos a la vez.
- Poseen una medida de calidad que determina hasta qué punto es buena una solución.
- Se generan descendientes a través de operadores estocásticos (los más usados son la recombinación y la mutación) aplicados a individuos de la generación actual.

Como ya se explicó anteriormente existen varias técnicas que se agrupan bajo la definición de AE. Partiendo de que un individuo constituye un candidato a solución del problema y que se representa en una estructura de datos mediante una codificación, todos los AEs se ajustan al "modelo" general que se muestra a continuación:

procedimiento AE

1. Generar aleatoriamente una población inicial P de **n** individuos

2. Evaluar los individuos de la población. Calcular la función de adaptación o ajuste. asociada a cada individuo.

3. Mientras no se alcance la condición de parada, tiene lugar el ciclo evolutivo, este es:

3.1 Seleccionar individuos para tener descendencia
3.2 Aplicar operadores genéticos a individuos seleccionados
3.3 Evaluar descendencia
Se ha obtenido una nueva generación..
fin del mientras
4. Devolver mejor individuo (solución óptima o cuasi-óptima al problema) o mejor conjunto de individuos de la población final según sea el caso.

1.5.1 Programación Genética.

La Programación Genética (PG) es un algoritmo evolutivo concebido en un inicio para lograr la evolución automática de programas de computadoras usando las ideas basadas en la selección natural de Darwin.
Esta técnica fue creada por John Koza a finales de los 80's, culminando con la publicación de su libro titulado "Genetic Programming: On the Programming of Computers by Means of Natural Selection" (1992). Koza propuso, por medio de esta extensión de lo que originalmente es un algoritmo genético, un método para la evolución de estructuras más complejas como pueden ser estructuras de programas de computadoras o funciones matemáticas.(11)
El hecho de que muchos problemas prácticos de diferentes dominios de aplicaciones puedan ser formulados como un problema de determinación de un "individuo solución" que produzca una salida deseada cuando se tienen presentes ciertas entradas particulares, hace a la Programación Genética una novedosa línea de investigación.

Esta técnica ha demostrado su capacidad para resolver problemas donde la interrelación entre las variables es desconocida; donde una solución aproximada es aceptable; pero sobre todo donde son muy valoradas las posibles pequeñas mejoras obtenidas y a pesar de su corta edad ya se han reportado resultados de gran envergadura (10).

1.5.1.1 Elementos básicos de la Programación Genética.

Los elementos básicos a definir al aplicar un algoritmo de PG, para dar solución a un problema, son:

1. El conjunto de símbolos terminales.
2. El conjunto de funciones permitidas.
3. La función de aptitud o adaptación (Método para evaluar el desempeño de los individuos).
4. Los parámetros para controlar el desarrollo del proceso evolutivo.
5. El método para designar un resultado y criterio de parada del algoritmo.

El conjunto de símbolos terminales.

En Programación Genética, los individuos (que pueden ser ecuaciones matemáticas), para su codificación, se representan con estructuras no lineales, como árboles. Por ejemplo para la ecuación siguiente la representación sería la que se muestra:

$$x = \frac{\sqrt{2b - c} + b}{c - a}$$

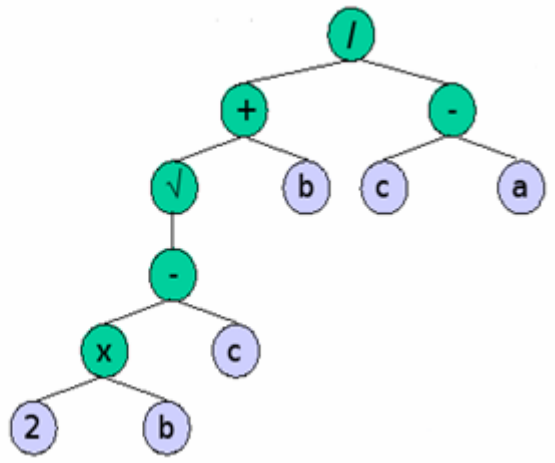


Fig 1. Representación en árbol de un individuo.

Los terminales son las hojas de los árboles que corresponden a variables o a valores constantes (se pueden ver como las entradas al programa).

Para el ejemplo mostrado anteriormente el conjunto de terminales serían los siguientes: **a, b, c, 2.**

En el caso que se le está dando solución los terminales lo constituyen los índices de refractividad atómica que describen cada uno de los fragmentos con los que se generará el modelo.

El conjunto de funciones permitidas.

En el conjunto de funciones permitidas pueden incluirse:

· operaciones aritméticas (+, -, *, etc.).
· funciones matemáticas (seno, coseno, log, etc.).
· operadores boléanos (and, or, not, etc.).
· operadores condicionales (if-then-else).
· funciones que causen iteración (do-until).
· funciones que causen recursión.
· cualquier otro tipo de función específica del dominio que sea definida.

Las funciones se usan junto con los terminales para generar la expresión matemática que trata de satisfacer la muestra finita de datos dados. Un modelo matemático (árbol de análisis gramatical) es una composición de funciones del conjunto F de funciones y terminales del conjunto T de terminales.

Cada una de las funciones del conjunto F debe ser capaz de aceptar, como sus argumentos, cualquier valor y tipo de dato que pueda ser retornado por cualquier función del conjunto de funciones, y cualquier valor y tipo de dato que pueda tomar por cualquier terminal del conjunto T. Esto es, que el conjunto de funciones y el conjunto de terminales deben tener la propiedad de clausura.

El conjunto de terminales (junto con el conjunto de funciones) son los ingredientes a partir de los cuales la PG, trata de construir un individuo, o lo que es lo mismo, un modelo para solucionar total, o parcialmente, un problema.

La medida de la aptitud.

Cada individuo en la población, se mide en términos de qué tan bien se comporta en el ambiente del problema particular. Esta medida se llama medida de aptitud y varía con el problema.

Los parámetros para controlar la ejecución.

Normalmente, los parámetros que controlan la ejecución de PG son: el tamaño de la población; el tamaño de los árboles; el número de generaciones; y las probabilidades de aplicación de los operadores genéticos de cruce y mutación entre otros.

El método para designar un resultado y el criterio para terminar la ejecución del programa.

Un método posible de designación de resultado consiste en seleccionar al mejor individuo que haya aparecido en cualquier generación, para lo cual es necesario colocarlo en memoria durante la ejecución.

Otro método alternativo es el de designar como resultado al mejor individuo de la población al momento de terminar la ejecución. Obviamente, no es necesario colocar al individuo en memoria en este último caso.

Por lo general, el mejor individuo hasta el momento se encontrará en la población al momento de la terminación, en cuyo caso ambos métodos designarían el mismo resultado.

El criterio de terminación del algoritmo de PG puede estar dado cuando se alcance un cierto número máximo G de generaciones en una ejecución, o cuando algún predicado de éxito específico del problema sea alcanzado (como por ejemplo, encontrar una solución 100% correcta).

1.5.2 Algoritmo general de la Programación Genética.

El algoritmo general de la PG funciona creando una población inicial al azar de P programas compuestos por los símbolos terminales y no terminales (funciones).

Mientras no se cumple el criterio de parada:

- ✓ Ejecuta cada programa de la población y obtiene su aptitud.
- ✓ Selecciona los individuos de la población de forma proporcional a su aptitud.
- ✓ Crea nuevos individuos a partir de la aplicación de operadores genéticos con probabilidades específicas. (Reproducción, Cruce, Mutación)

Finalmente devuelve el mejor individuo o mejor conjunto de individuos de la población final según sea el caso.

La PG incuba individuos ejecutando los siguientes pasos:

1. Generar una población inicial de composiciones aleatorias de funciones y terminales del problema.

2. Ejecutar iterativamente los siguientes sub-pasos hasta que el criterio de terminación sea satisfecho:

a. Evaluar cada función en la población y asignarle un valor de aptitud de acuerdo a cuan bien solucione el problema.

b. Crear una nueva población de individuos aplicando las siguientes dos operaciones primarias. Las operaciones son aplicadas a los individuos en base a cierto criterio de selección.

i. Copiar programas de computadora existentes en la nueva población.

ii. Crear nuevos programas de computadora recombinando genéticamente partes aleatoriamente seleccionadas de los indivduos ya existentes.

3. El mejor individuo que haya aparecido en cualquier generación es designado como el resultado de la programación genética. Este resultado puede ser una solución perfecta o aproximada al problema.

1.5.3 Generación de la población inicial.

La población inicial en la programación genética está compuesta por individuos representados por expresiones. Cada expresión es realizada mediante la generación al azar de un árbol rotulado en cada uno de sus puntos.

Se comienza seleccionando una de las funciones en el conjunto F al azar (utilizando una distribución uniforme de probabilidad).

La siguiente figura muestra el comienzo de la creación de un árbol aleatorio. Se seleccionó la función suma del conjunto de funciones, la cual lleva dos argumentos.

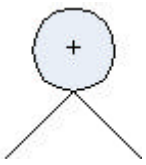


Figura 2. Nodo raíz de un individuo.

Cuando un punto del árbol es rotulado con la función ***f*** del conjunto ***F***, entonces se dibujan ***z(f)*** líneas irradiándose desde ese punto, donde ***z(f)*** es la aridad de ***f***. Por cada línea, un elemento del conjunto de funciones y de terminales es seleccionado al azar para ser el punto final de esa línea.

Si se selecciona una función como rótulo para cualquiera de estos puntos finales, el proceso de generación entonces sigue recursivamente como se describió más arriba. Si se selecciona un

terminal como rótulo para cualquiera de estos puntos finales, ese punto se convierte en una hoja del árbol y el proceso de generación es terminado para ese punto.
Un ejemplo de estructura ya terminada es el siguiente árbol:

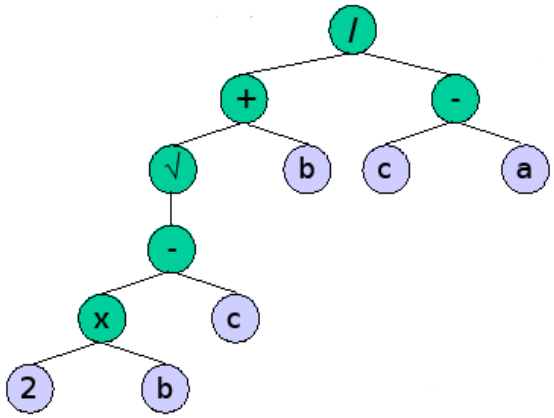


Figura 3. Ejemplo de un individuo ya generado.

El tamaño de los árboles va a ser menor que un tope permitido, este tope puede ser tanto en profundidad como en el número de nodos máximo por cada árbol, discriminando todos aquellos que lo excedan.
Para introducir diversidad entre la población de estructuras (árboles) se puede generar un porcentaje de árboles de cada tamaño válido o generar estructuras puramente aleatorias que hagan uso de todas las entradas.

1.5.4 Operadores genéticos.

Aquí se describirán los operadores genéticos reproducción, cruce y mutación utilizados en la Programación Genética.

Reproducción

Esta operación, la cual es asexual porque actúa sobre un solo individuo a la vez, consiste en dos pasos muy simples. Primero, un único individuo es seleccionado de la población utilizando algún método de selección, luego es copiado sin alteración, desde la población actual hacia la nueva población (la nueva generación).

Cruzamiento

El operador de cruzamiento comienza con dos individuos padres y produce dos descendientes. Es una operación sexual, los padres son seleccionados según algún método de selección. La operación comienza seleccionando (utilizando una distribución uniforme de probabilidad) un punto de cruce para cada padre por separado. El fragmento de cruce para un padre en particular es el árbol que posee su raíz en el punto de cruce de ese padre y que consiste en el subárbol entero que se encuentra por debajo del punto de cruce. Dicho subárbol puede consistir en un solo terminal (si el punto de cruce es un terminal) o incluso en el árbol completo que representa a la expresión (si el punto de cruce es la raíz de dicho árbol). El primer descendiente se producirá borrando el fragmento de cruce del primer padre e ingresando el fragmento de cruce del segundo padre en el punto de cruce del primer padre. El segundo descendiente se obtiene con igual procedimiento, borrando el fragmento de cruce del segundo padre e ingresando el fragmento de cruce del primer padre en el punto de cruce del segundo padre.

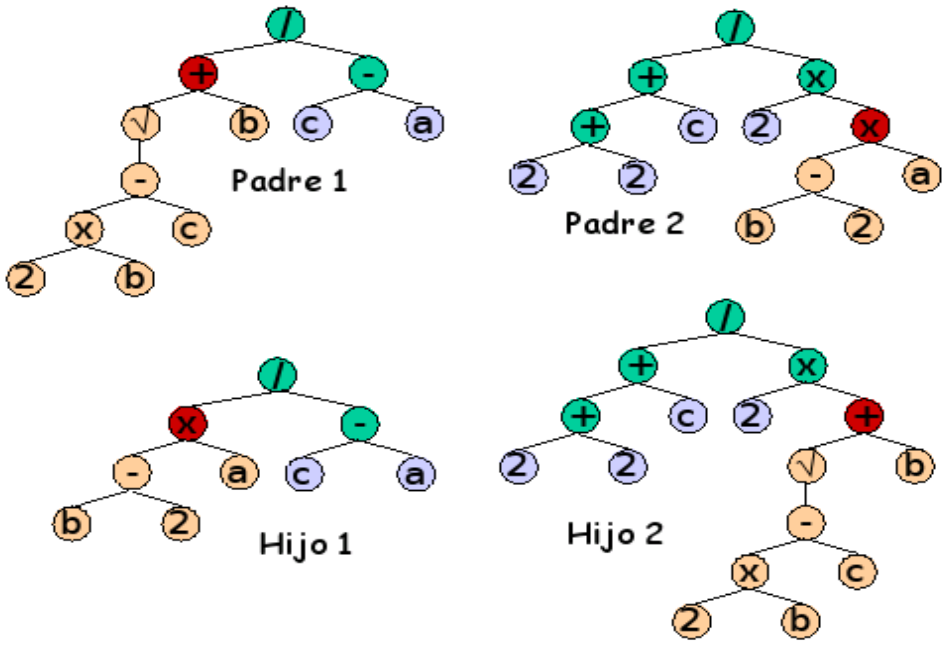


Fig 4. Ejemplo de cruce.

Mutación

La mutación es asexual y opera sobre una única expresión, la cual es seleccionada en base a una probabilidad proporcional a su aptitud. La operación de mutación comienza seleccionando un punto aleatorio dentro del árbol. Dicho punto puede ser interno o externo. La operación de

mutación remueve lo que esté seleccionado en el punto elegido como así también todo aquello que esté por debajo de ese punto, y luego inserta un árbol creado aleatoriamente en ese punto. Esta operación es controlada por un parámetro que especifica el máximo tamaño (medido por profundidad) para el nuevo subárbol creado que será insertado.

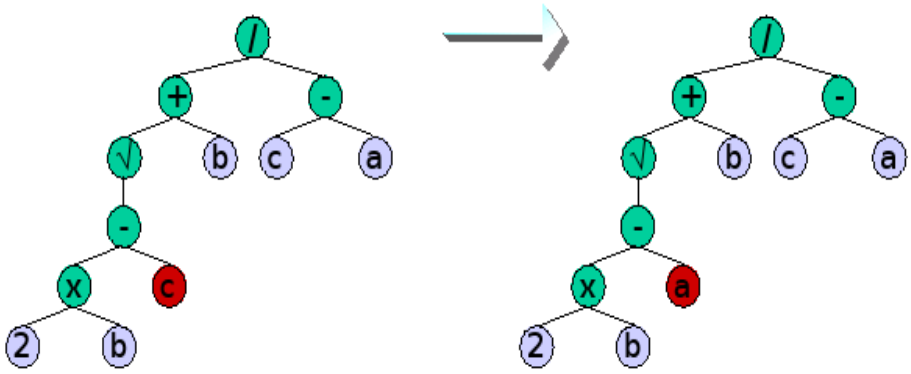


Fig 5. Ejemplo de mutación.

1.5.4 Métodos de selección

Existen diversos métodos de selección que pueden ser utilizados durante la selección de progenitores y de sobrevivientes. Algunos de ellos son:

Selección elitista: se garantiza la selección de los ***k*** miembros más aptos de cada generación.

Selección proporcional a la aptitud: los individuos más aptos tienen más probabilidad de ser seleccionados, pero no la certeza.

Selección por ruleta por bondad: una forma de selección proporcional a la aptitud en la que la probabilidad de que un individuo sea seleccionado es proporcional a la diferencia entre su aptitud y la de sus competidores. Conceptualmente, esto puede representarse como un juego de ruleta donde el individuo obtiene una sección de la ruleta, pero los más aptos obtienen secciones mayores que las de los menos aptos. Luego la ruleta se hace girar, y en cada ocasión se elige al individuo que «posea» la sección en la que se asiente la bola que viaja por la ruleta.

Selección por ruleta por orden: los individuos se ordenan según su aptitud, y los números de orden resultantes se utilizan como probabilidades de selección para formar la muestra. La selección se basa en este ranking, en lugar de las diferencias absolutas en aptitud (como lo es

en el caso de la ruleta por bondad). La ventaja de este método es que puede evitar que individuos muy aptos ganen dominio al principio a expensas de los menos aptos, lo que reduciría la diversidad genética de la población y podría obstaculizar la búsqueda de una solución aceptable.

Selección por torneo: se eligen subgrupos de individuos de la población, y los miembros de cada subgrupo compiten entre ellos. Sólo se elige a un individuo de cada subgrupo para la reproducción.

Selección por torneo binario: un caso particular del método de selección anterior, en el cual se eligen aleatoriamente ***k*** pares de individuos de la población base, y se constituye la muestra seleccionando el mejor de cada par.

1.6 Algunos softwares de predicción existente.

Se han desarrollado varias herramientas para predecir actividad biológica en compuestos, unas con más prestaciones y exactitud que otras, pero todas, sin dudas con resultados prometedores. A continuación se analizarán algunas de estas herramientas:

OncoLogic

Fue desarrollado según los términos de un acuerdo cooperativo entre la oficina de EPA (Environmental Protection Agency) de la prevención de la contaminación, Toxics (OPPT) y LogiChem Inc. OncoLogic analiza las estructuras químicas para determinar la probabilidad de estos de provocar cáncer mediante la aplicación de reglas de análisis SAR (Structure-Activity Relationships) y la incorporación de conocimientos sobre cómo las sustancias químicas causan cáncer en animales y humanos. Es el único sistema experto para la predicción de cáncer que evalúa no sólo la estructura química sino también factores no estructurales y propiedades físicas.

Aunque hace uso de ciertas características físicas tales como forma, tamaño, arreglo químico, y distribución de grupos funcionales y después estudia la contribución de cada uno de estos factores a las actividades biológicas, no hace uso de los modelos QSAR en sus evaluaciones, y no puede calcular características fisicoquímicas para apoyar tales modelos. (12)

Hazardexpert y Metabolexpert

Son programas desarrollados por CompuDrug Chemistry Ltd (Budapest, Hungary). El Hazardexpert predice una gama de compuestos químicos peligrosos para la salud mientras el Metabolexpert predice los metabolitos probables.
Aunque fueron desarrollados como paquetes separados una corta versión del Metabolexpert ha sido incorporada dentro del Hazardexpert. Este programa trabaja mediante la búsqueda de toxicóforos que se encuentran en la base de conocimientos de fragmentos tóxicos que se han obtenido de la literatura QSAR o mediante reportes de la EPA.
La identificación de un toxicóforo conduce a las estimaciones de la toxicidad mediante reglas en las bases de conocimiento. Las reglas describen segmentos tóxicos y sus efectos sobre varios sistemas biológicos, y se basan en el uso combinado del conocimiento toxicológico, del juicio experto, de los modelos de QSAR, y de la lógica difusa (que simula los efectos de diversas condiciones de la exposición).Sin embargo en importantes fragmentos este software no podía hacer predicciones, ejemplos de estos fragmentos son (13):

1. cloruto de vinilo;
2. organofosfatos;
3. compuestos organometálicos;

Apex-3D

Es un sistema experto desarrollado para representar, elucidar y utilizar conocimiento sobre relación estructura - actividad. Es utilizado para crear modelos 3D SAR y QSAR que luego son usados para la predicción y clasificación de la actividad biológica. Este sistema experto está insertado dentro del paquete de programas Insight II y tuvo su predecesor en el sistema experto OREX implementado sobre IBM PC 80286. Este último se basa en la descomposición topológica de las moléculas en fragmentos estructurales y su asociación a las actividades biológicas reportadas en una base de datos interna de alrededor de 15 000 compuestos. Emplea también la teoría lógico-combinatoria para el establecimiento de reglas de inferencia. (14)

BioVis ultra

ChemFinder 9.0 es uno de los software más respetados por los químicos a nivel internacional, por su eficiencia y su gran número de prestaciones. BioViz es un módulo de visualización de biomoléculas que se le adicionó a ChemFinder 9.0 y que permite al usuario correlacionar

actividad biológica con estructuras químicas, transformando números de su base de datos en gráficos fáciles de entender permitiendo percibir relaciones estructura actividad en una ventana interactiva, mostrando diagramas de dispersión, histogramas y otros gráficos útiles, ayudando a los químicos a ganar tiempo y dinero en las investigaciones. (15)

ADAPT

Es un sistema de programas que le permite al usuario el desarrollo de relaciones estructura-actividad y estructura-propiedad. Brinda la facilidad de entrada gráfica y almacenamiento de estructuras moleculares y sus datos asociados, generación de estructuras 3D, cálculo de descriptores moleculares y análisis de estos empleando estadística multivariada, reconocimiento de patrones o redes de neuronas para construir modelos predictivos. Los enfoques estadísticos incluyen regresión lineal múltiple, análisis clúster, discriminante y redes neuronales. Se ejecuta sobre estaciones de trabajo Sun con sistema operativo UNIX. (16)

Otros software reconocidos internacionalmente que establecen relaciones QSAR y que es importante al menos mencionarlos por ser muy aceptados por los científicos debido a su eficiencia son Codessa, Accelerys, HyperChem, ChewSW y Tripos entre otros.

1.7 Tendencias y tecnologías actuales.

Para el desarrollo de la aplicación se utilizó metodología RUP, manteniendo compatibilidad con la plataforma a la cual se integrará posteriormente la aplicación en forma de plug-in, esta metodología es iterativa incremental, o sea, va eliminando los errores cometidos en las iteraciones previas, obteniéndose a medida que avanza el proyecto un producto de mayor calidad. Define los roles a jugar por cada miembro del equipo de desarrollo en cada una de las etapas por las que transcurre el proyecto y facilita la comunicación entre los diferentes miembros del equipo de desarrollo.

Como lenguaje de modelado se utilizó Unified Modeling Languaje (UML) notación con la cual se puede especificar, construir, visualizar y documentar los artefactos de un sistema de software orientado a objetos (OO). Esta preescribe un conjunto de notaciones y diagramas estándares, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Entre las características más importantes de UML se pueden citar su viabilidad para corregir errores,

desarrollo iterativo e incremental, tecnología orientada a objeto y una de las más importantes, brinda la posibilidad al cliente de participar en todas la etapas del proyecto.(17)

La herramienta CASE utilizada para modelar el programa es Visual Paradigm, un producto de Visual Paradigm UML Community que, a su vez, es una de las principales compañías de herramientas CASE. Tiene disponible distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community la cual es gratuita y la compañía facilita licencias especiales para fines académicos. Esta herramienta sirve para realizar modelado UML y tiene características gráficas muy cómodas que facilitan la realización de los diagramas de modelado que sigue el estándar de UML como los diagramas de clase, casos de uso, comunicación, secuencia, estado, actividad, componentes, etc. Otras características importantes de Visual Paradigm es su integración con algunos IDE de desarrollo como Eclipse desarrollado por IBM, Netbeans de Sun o JBuilder de Borland que permite la generación automática de sus clases y las relaciones entre ellas.(18)

En cuanto a plataforma escogida se optó por JDK versión 1.5.0_10 y como lenguaje de programación Java debido a que es un lenguaje de propósito general y al hecho de ser independiente de la plataforma, buscando la portabilidad en los diferentes sistemas operativos y plataformas de hardware.(19)

El entorno de desarrollo empleado es Eclipse, siendo esta una plataforma de desarrollo extensible, basada en Java y de tipo open-source (CPL). Este entorno de desarrollo integrado ofrece, el control del editor de código, del compilador y del depurador desde una única interfaz de usuario. Su misión consiste en evitar tareas repetitivas, facilitar la escritura de código correcto, disminuir el tiempo de depuración e incrementar la productividad del desarrollador.

Finalmente como sistema gestor de bases de datos se ha utilizado MySQL en su versión 5.x debido que al ser software libre, es posible obtenerlo gratis o a bajo costo. Es uno de los gestores más rápidos que se encuentran en el mercado, almacena los datos en tablas separadas en lugar de ponerlos todos en un solo lugar, lo que provee velocidad y flexibilidad a la aplicación. Las tablas son enlazadas al definir relaciones que hacen posible combinar datos de varias tablas que necesitan ser consultados.

El servidor MySQL se desarrolló para manejar grandes bases de datos mucho más rápido que las soluciones existentes y ha sido usado exitosamente en ambientes de producción sumamente exigentes por varios años. Aunque se encuentra en desarrollo constante, el servidor MySQL ofrece hoy un conjunto rico y útil de funciones. Su conectividad, velocidad, y seguridad hacen de MySQL un servidor bastante apropiado para acceder a bases de datos en Internet. Presenta versiones en varios sistemas operativos y compatibilidad entre ellas.(20)

1.8 Conclusiones.

Luego de haber realizado el estudio teórico de esta investigación se vio la necesidad de desarrollar este módulo, ya que puede aportar importantes resultados a los estudios QSAR al ser la primera aplicación que prediga la actividad biológica asociada a un compuesto químico utilizando el Índice del Estado Refractotopológico Total para describir los fragmentos a través de modelos predictivos, obtenidos por Programación Genética.

En este capítulo se define además conceptos importantes para la comprensión de la investigación; se profundizó en el Índice de Estado Refratotopológico Total desarrollado en Cuba recientemente; se explicaron los métodos y mecanismos utilizados para predecir y se explica la Programación Genética como técnica a utilizar. Por último se analizan algunos sistemas automatizados utilizados en la actualidad para la predicción de actividad biológica y las tendencias y tecnologías que se usaron para el desarrollo de la aplicación.

Capítulo 2: Características del Sistema.

2.1 Introducción

En este capítulo se describen las particularidades del algoritmo de PG que se utilizó y se describe la solución propuesta utilizando los componentes del modelo de dominio de la metodología RUP. De este modelo se tendrá en cuenta la definición de las entidades y los conceptos principales, así como su representación gráfica. Además se describen las reglas del negocio, los requisitos funcionales y no funcionales del sistema, los actores que intervienen en el mismo, así como el diagrama de casos de uso del sistema su descripción correspondiente.

2.2 Algoritmo de Programación Genética utilizado para el desarrollo del sistema

En este epígrafe se explica de forma detallada el algoritmo de PG utilizado en el sistema con la intensión de entender a fondo su funcionamiento.

Antes de entrar a explicar cómo se desarrolla la secuencia de pasos del algoritmo es preciso explicar el problema a enfrentar y definir una serie de elementos que son fundamentales para poder resolverlo.

En la sección 1.1 del capítulo anterior se explicó como se logró representar, a través de números, la estructura química y la actividad biológica de un compuesto, por lo que si se desea encontrar un modelo que describa cierta actividad biológica, el problema que se intenta resolver se centra en encontrar una función matemática que relacione estos dos conjuntos de números, de tal forma que se pueda describir la propiedad biológica que se tiene. En el caso particular que se está analizando, la descripción de la estructura química de los fragmentos está dada por el índice de refractividad atómica, y la actividad biológica por la concentración del compuesto a través de la relación ***Actividad Biológica = log (1/C)*** donde ***C*** es la concentración efectiva de cada fragmento.

Teniendo bien definido el problema se deben definir también algunos elementos necesarios para darle solución a través de PG:

El conjunto de símbolos terminales.

En el caso que particular que se está analizando los terminales lo constituyen los índices de refractividad atómica que describen cada uno de los fragmentos con los que se generará el modelo.

El conjunto de funciones permitidas.

En este caso se trabajará con funciones aritméticas y algunas funciones matemáticas, debido a que el problema en cuestión se basa en encontrar el modelo de mejor ajuste que describa la actividad biológica que se está analizando o, lo que es lo mismo, la función matemática de mejor ajuste, por lo que no tiene sentido usar funciones de tipo booleanas, iterativas, recursivas, etc.

La medida de aptitud

La medida de aptitud para encontrar el modelo que describe la actividad biológica consiste en evaluar cada uno de los individuos encontrados (un individuo es un modelo) a través de los valores de los índices de refractividad atómica, el resultado obtenido es la concentración que posee ese fragmento, que luego es comparado con el valor real registrado en la base de datos, obteniendo un valor de ***"fitness"*** o, lo que es lo mismo, un valor de aptitud, lo que permite analizar qué tan apto es un individuo en una población.

Los parámetros para controlar la ejecución.

La aplicación que se desarrolló permite al usuario controlar los parámetros que controlan la ejecución, explicados en el capitulo anterior, definiéndolos en la interfaz visual antes de comenzar la corrida del programa.

El método para designar un resultado y el criterio para terminar la ejecución del programa.

En el problema que se está analizando se procede de la siguiente forma:

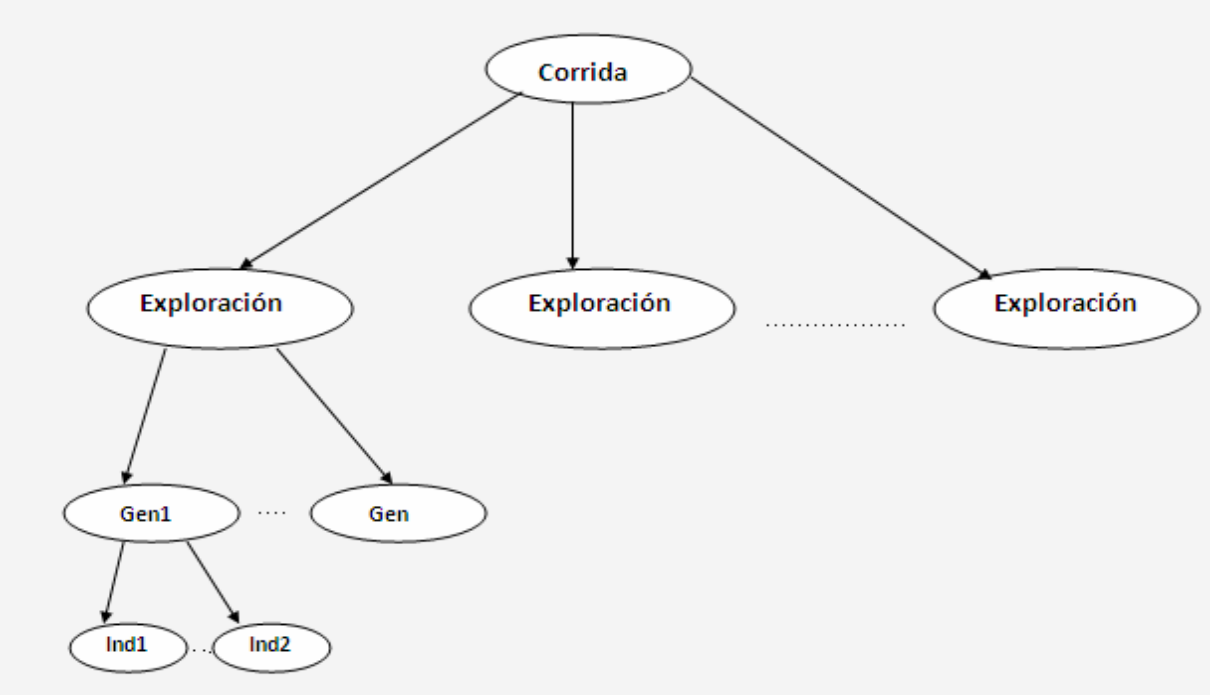


Fig 6. Ejecución del algoritmo por corridas, exploraciones y generaciones.

El usuario define el número de corridas, de exploraciones, de generaciones y la cantidad de individuos. Cada corrida posee un número de exploraciones que a su vez posee un número de generaciones, y esta última un número de individuos que se cumplirá con seguridad cada 5 generaciones de cada exploración.

Una vez generada aleatoriamente la primera generación, de la primera exploración, los individuos se cruzan entre ellos y se mutan, dando lugar a nuevos individuos. De la nueva población se eliminan todos aquellos que hayan sido repetidos y todos los que sobrepasen el tamaño definido por el usuario, el resto pasa a ser la próxima generación dentro de la misma exploración. Cada 5 generaciones se escoge el mejor individuo, utilizando selección elitista, este será copiado sin alteración en la generación siguiente (por ejemplo, si el usuario define 10 individuos por generación y estamos en la quinta generación copiamos a la sexta el mejor individuo encontrado hasta el momento que será el de mayor aptitud, generando nueve individuos aleatoriamente hasta completar nuevamente la cantidad especificada por el usuario, desechando el resto de la población que se tenía hasta ese momento y volviendo a cruzar, mutar y eliminar individuos formando una nueva generación, repitiéndose el ciclo hasta cumplirse el número de generaciones definido por el usuario). Los mejores individuos de cada 5 generaciones se van comparando y quedándose en memoria, el mejor encontrado en la última generación de la exploración resultará ser el mejor de esa exploración, luego se comparan los

individuos por exploraciones y de este modo se obtiene el mejor individuo de la corrida en general que a su vez será comparado con los mejores individuos del resto de las corridas hallándose la mejor solución, siendo este el modelo de mejor ajuste para la actividad biológica analizada.

Generación de la población inicial.

La población inicial se genera de forma aleatoria haciendo uso del conjunto de funciones, terminales y constantes del problema, hasta llegar al número de individuos especificados por el administrador del sistema a través de la interfaz visual.

Operadores genéticos.

Reproducción

En el algoritmo que se utilizó la reproducción se realiza cada 5 generaciones, donde por selección de tipo elitista se garantiza la selección del miembro más apto, y es copiado sin alteración a la generación siguiente.

Cruzamiento

El cruzamiento se realiza escogiendo pares de individuos en la población, donde los individuos más aptos tienen más probabilidad que el resto de ser seleccionados.

Mutación

La mutación se realiza en individuos de forma independiente escogidos por selección proporcional a la aptitud, la probabilidad de mutación es entrada por el usuario en la interfaz visual.

2.3 Definición de Modelo de Dominio

Un modelo de dominio captura los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema. Los objetos del dominio representan las "cosas" que existen o los eventos que suceden en el entorno en el que trabaja el sistema y tiene como objetivo fundamental la comprensión y descripción de las clases más importantes del sistema. Los conceptos definidos para dicho modelo son:

Especialista: Cliente que utilizará el sistema para predecir.

Administrador: Persona que se encargará de generar los modelos matemáticos que serán utilizados para predecir la actividad biológica.

Modelo: Es la función matemática que describe la actividad biológica.

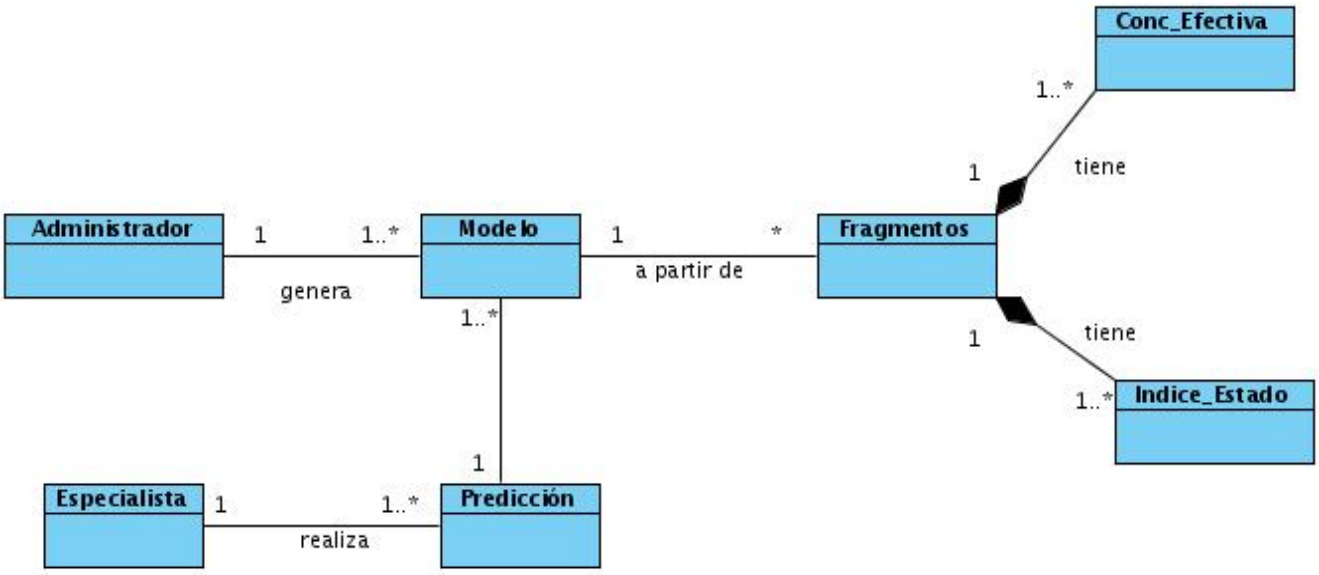
Predicción: Acción que el sistema realiza para emitir un resultado solicitado por el especialista.

Fragmento: Porción de una molécula que tiene asociada una actividad biológica.

Concentración Efectiva: Actividad biológica (**AB**) asociada a un fragmento molecular, está dada por la expresión ***AB = log 1/C*** donde ***C*** es la concentración del fragmento.

Índice del Estado Refractotopológico Total: Descriptor que aporta información sobre la estructura molecular de un fragmento.

2.3.1 Modelo de dominio



2.4 Reglas del negocio.

- ✓ Para poder generar un modelo y predecir actividad biológica es necesario que el usuario se loguee correctamente y pueda acceder a la base de datos.
- ✓ Cuando se va a generar un modelo es preciso cargar antes los fragmentos de la base de datos.
- ✓ Para predecir la actividad biológica es necesario cargar antes los fragmentos de la base de datos.

- ✓ Cuando se va a predecir la actividad biológica de los fragmentos se debe cargar el modelo con el que se desea predecir.
- ✓ Para conservar los modelos predictivos generados deben ser guardados.

2.5 Especificación de Requerimientos del sistema.

Todas las ideas que los clientes, usuarios y miembros del equipo de proyecto tengan acerca de lo que debe hacer el sistema, deben ser analizadas como candidatas a requisitos. Los requisitos se pueden clasificar en: funcionales y no funcionales.

2.5.1 Requisitos funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir.

- ➢ R1: Crear el Modelo.
- ➢ R2: Cargar los ficheros de los fragmentos.
- ➢ R3: Cargar el modelo matemático.
- ➢ R4: Comprobar el modelo.
- ➢ R5: Guardar el Modelo.
- ➢ R6: Predecir Actividad Biológica.
- ➢ R7: Guardar los resultados de la predicción.

2.5.2 Requisitos no funcionales

Los requerimientos no funcionales responden a cualidades que el producto debe tener y las características para que este sea atractivo, confiable, usable y seguro.

Apariencia o interfaz externa

- ✓ El sistema cuenta con una interfaz agradable a la vista del usuario, fácil de usar y entender, evitando que el usuario se pierda dentro del sistema.

Usabilidad

La aplicación está concebida para ser utilizada por cualquier persona con los conocimientos mínimos de computación ya que la misma está encaminada a ser usada esencialmente por especialistas en química, los cuales no deben necesitar más que los conocimientos básicos para el buen uso de la aplicación. Se necesita contar con conocimientos especializados en química para entender los resultados dados por el sistema.

Rendimiento

La eficiencia del producto está determinada en gran medida por el aprovechamiento de los recursos que se disponen en el modelo Cliente/Servidor, y la velocidad de las consultas en la Base de Datos. La herramienta propuesta debe ser rápida y el tiempo de respuesta debe ser el mínimo posible, adecuado a la rapidez con que el cliente requiere la respuesta a su acción.

Soporte

El sistema debe permitir la interacción con los demás módulos que componen la plataforma. Una vez terminado el software se realizarán distintas pruebas para comprobar su funcionalidad y se prestarán servicios de instalación, configuración y mantenimiento de la aplicación.

Requerimientos de Portabilidad

La herramienta propuesta podrá ser usada bajo cualquier sistema operativo, para su implementación se usaron Herramientas de Programación y Gestión de Bases de Datos que son multiplataforma.

Software

Se debe disponer de sistemas operativos Linux, Windows 95 o superior para la instalación de la aplicación. Debe tenerse instalado el Java Runtime Environment (JRE) versión 1.5 o superior.

Hardware

Para el desarrollo y puesta en práctica del proyecto se requieren máquinas con los siguientes requisitos:

• Procesador Pentium 3 o superior.

• 256 Mb de RAM.

• 50 Mb de espacio en disco duro.

2.6 Actores del sistema.

Los actores de un sistema pueden ser las personas, sistemas o hardware externo que se relacionan o interactúan con dicho sistema. Cada actor juega un rol determinado al interactuar con el sistema y diferentes usuarios pueden asumir el mismo rol de un actor. Luego de definir qué es un actor se definirán los actores de la aplicación en cuestión.

Actores	Justificación
Administrador	Es la persona encargada de utilizar el sistema para generar los modelos y comprobarlos definiendo también los parámetros de entrada. Todo esto con el fin de obtener el modelo matemático que describe la actividad biológica.
Especialista	Es la persona encargada de utilizar el sistema para predecir la actividad biológica.

2.7 Diagrama de casos de uso del sistema.

Un diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores.

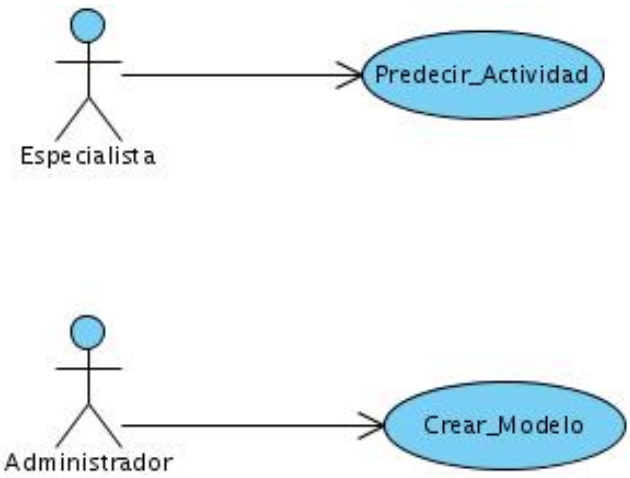


Fig 7. Diagrama de casos de uso del sistema.

2.8 Descripción de los casos de uso del sistema

En esta sección se describen cada uno de los Casos de Uso del sistema.

2.8.1 Descripción del Caso de Uso Crear Modelo.

Caso de Uso	Crear Modelo
Actores	Administrador
Propósito	El propósito es crear el modelo matemático necesario para predecir la actividad biológica de los compuestos.
Resumen	El Caso de Uso se inicia cuando el administrador selecciona la opción de Crear Modelo en el menú principal del sistema, carga la familia de compuestos a analizar y genera el modelo matemático.
Referencia	**R1, R2, R3, R4, R5.**
CU asociados	-
Precondiciones	• El administrador debe estar autentificado en la plataforma central. • Los fragmentos estén cargados.
Poscondiciones	Se devolverá el modelo.
Descripción	
Interfaz	Interfaz asociada a este caso de uso es: CI_Crear Modelo, CI_Cargar_Frag.
Flujo Normal de Eventos	

Acción del Actor	Respuesta del Sistema
1. El Administrador selecciona la opción Crear Modelo del menú inicial.	1.1 El sistema muestra la clase interfaz CI_ Crear_Modelo.
2. El especialista selecciona la opción Cargar Fragmentos.	2.1 El sistema muestra la clase interfaz CI_Cargar Fragmentos.
3. El Administrador predefine los valores por los que consultará en la base de datos y acepta.	3.1 El sistema carga los fragmentos en la interfaz CI_Crear_Modelo.
4. El Administrador predefine los valores de entrada (parámetros y operaciones).	4.1 Se activa la opción Generar Modelo.
5. Selecciona la opción Generar Modelo.	5.1 Se crea el modelo matemático. 5.2 Se activan las opciones Salvar Modelo y Comprobar Modelo.
6. El Administrador selecciona la opción Salvar Modelo.	6.1 El sistema guarda el modelo matemático en la base de conocimientos.
7. El Administrador selecciona la opción Salir.	7.1 Termina el Caso de Uso.
Curso alterno de eventos	
Acción del Actor	**Acción del Actor**
6. Selecciona la opción Comprobar Modelo.	6.1 Se comprueba la validez del modelo. 6.2 Muestra el error del modelo matemático. 6.3 Muestra el número de certezas encontradas.
7. Administrador selecciona la opción Salvar Modelo.	7.1 Se guarda el modelo matemático en la base de conocimientos.
8. Administrador selecciona la opción Salir	8.1 Termina el caso de uso.
Prioridad: Crítico.	

2.8.2 Descripción del Caso de Uso Predecir Actividad.

Caso de Uso	Predecir Actividad Biológica
Actores	Especialista
Propósito	El propósito es predecir la actividad biológica de una muestra de fragmentos.
Resumen	El Caso de Uso se inicia cuando el especialista selecciona la opción Predecir Actividad, se cargan los fragmentos y el modelo matemático por el que se va a predecir y se realiza la predicción.
Referencia	**R2, R6, R7.**
CU asociados	-
Precondiciones	• El especialista debe estar autentificado en la plataforma central. • Los fragmentos estén cargados.
Poscondiciones	Actividad predicha para cada fragmento individualmente.
Descripción	
Interfaz	La interfaz asociada a este caso de uso es: CI_Predecir

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El especialista selecciona la opción Predecir Actividad del menú principal del sistema.	1.1 El sistema muestra la clase interfaz CI_Predecir_Actividad.
2. El especialista selecciona la opción Cargar Modelo.	2.1 El sistema carga el modelo matemático.
3. El especialista selecciona la opción Cargar Fragmentos.	3.1 El sistema muestra la clase interfaz CI_Cargar Fragmentos.
3. El especialista selecciona los fragmentos que desea consultar en la base de datos y acepta.	3.1 El sistema carga los fragmentos en la interfaz CI_Predecir_Actividad.
4. El especialista selecciona la opción Predecir.	4.1 El sistema muestra la predicción por fragmentos. 4.1 Se activa la opción Guardar Predicción.
5. El especialista selecciona la opción Salir.	5.1 Termina el caso de uso.

Curso alterno de eventos	
Acción del Actor	**Respuesta del Sistema**
5. El especialista selecciona la opción Guardar Predicción.	5.1 El sistema pide al usuario la dirección donde será guardada la predicción.
6. El especialista especifica la dirección.	6.1 El sistema guarda la predicción.
7. El especialista selecciona la opción Salir.	7.1 Termina el caso de uso.
Prioridad: Crítico	

2.9 Conclusiones.

En este capítulo se analiza la solución propuesta basada en el modelo del dominio, se definen las reglas del negocio a tener en cuenta en la aplicación y se describen cada uno de los casos de usos del sistema. Se definen los requisitos funcionales y no funcionales que debe cumplir el sistema desarrollado, ganando claridad en la concepción del sistema a construir, y sienta las bases para las restantes fases del proceso de diseño e implementación del sistema. Además se muestra un glosario de términos en el cual se explican aquellas palabras que puedan tener alguna ambigüedad en la comprensión del modelo del dominio.

Capítulo 3: Análisis y Diseño.

3.1 Introducción.

Como resultado del trabajo realizado en el capítulo anterior se obtuvo una vista externa del sistema. En este capítulo, a través del análisis y diseño del módulo que se está realizando, se profundizará en los casos de usos detallándolos de manera que permitan reflejar una vista interna del sistema descrita con el lenguaje de los desarrolladores. En esta vista interna se especificarán mejor los casos de uso y se determinarán las clases necesarias para llevar a cabo las funcionalidades en ellos contenidas.

En la construcción del modelo de análisis se identificarán las clases que describen la realización de los casos de uso, los atributos y las relaciones entre ellas. Con esta información se construirá el Diagrama de clases del análisis, lo que facilitará la transición al diseño e implementación de la solución.

A pesar de que en el modelo del análisis hay un refinamiento de los requisitos, no se tomarán en cuenta el lenguaje de programación a usar en la construcción, la plataforma en la que se ejecutará la aplicación, los componentes prefabricados o reusables de otras aplicaciones, entre otras características que afectan al sistema, esto se tendrá en cuenta en el diseño a través de la descripción de los CU del diseño, los Diagramas de Clases de Diseño, los Diagramas de Colaboración y Secuencia, así como el Diagramas de Componentes.

3.2 Análisis.

El flujo de trabajo de Análisis se desarrolla fundamentalmente dentro de la fase de elaboración y se corresponde principalmente con el Flujo de Trabajo de Análisis y Diseño según RUP. En Análisis hay un refinamiento de los requisitos, sin tener en cuenta aún el lenguaje de programación que se utilizará para la desarrollar el sistema, la plataforma en la que se ejecutará la aplicación, entre otras características que afectan al sistema. El análisis se centra en comprender los requisitos funcionales del software y no en precisar cómo se implementará la solución.

3.2.1. Diagramas de clases del análisis.

Muestra las relaciones entre los actores y las clases del sistema sin entrar en especificaciones de lenguaje en el que será desarrollada la aplicación.

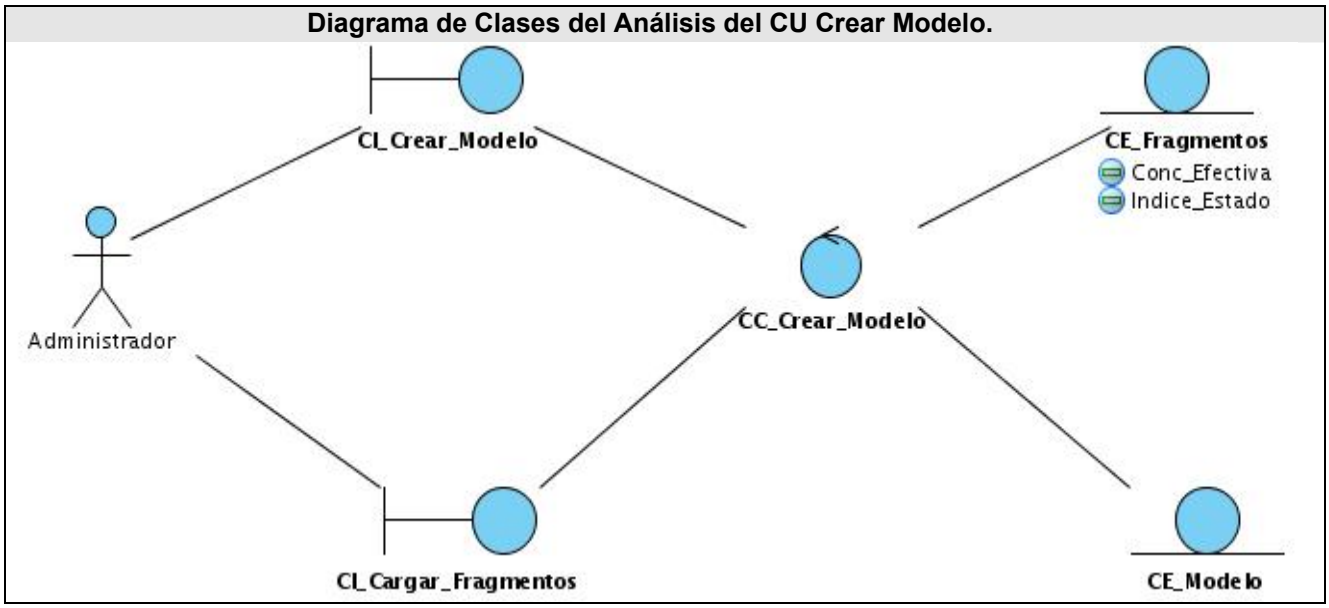


Fig 8. Diagrama de clases del análisis para el CU Crear Modelo.

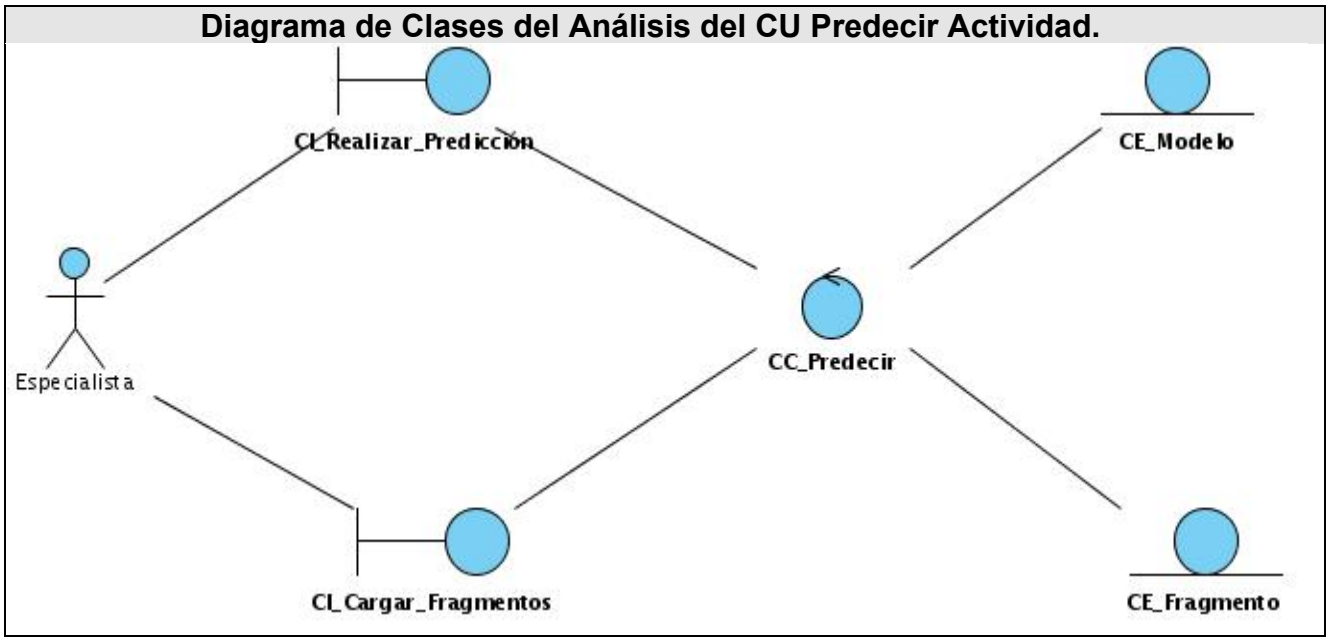


Fig 9. Diagrama de clases del análisis para el CU Predecir Actividad.

3.3 Diseño

El flujo de trabajo de diseño tiene el propósito de formular los modelos que se centran en los requisitos no funcionales, en el dominio de la solución y que prepara para la implementación y prueba del sistema creando un plano del modelo de implementación.

Durante esta fase se analiza si es posible dar una solución que satisfaga a los requerimientos significativos de la arquitectura describiendo los subsistemas y los componentes de un sistema informático y las relaciones entre ellos.

3.3.1 Diagramas de secuencias del diseño.

Estos diagramas, sin explicar el cómo, muestran la interacción entre el sistema y los actores. Para cada uno de los Casos de uso se construyó un diagrama de secuencia, donde intervienen los actores del caso de uso, los objetos que representan al sistema, y los eventos que envía cada actor al sistema.

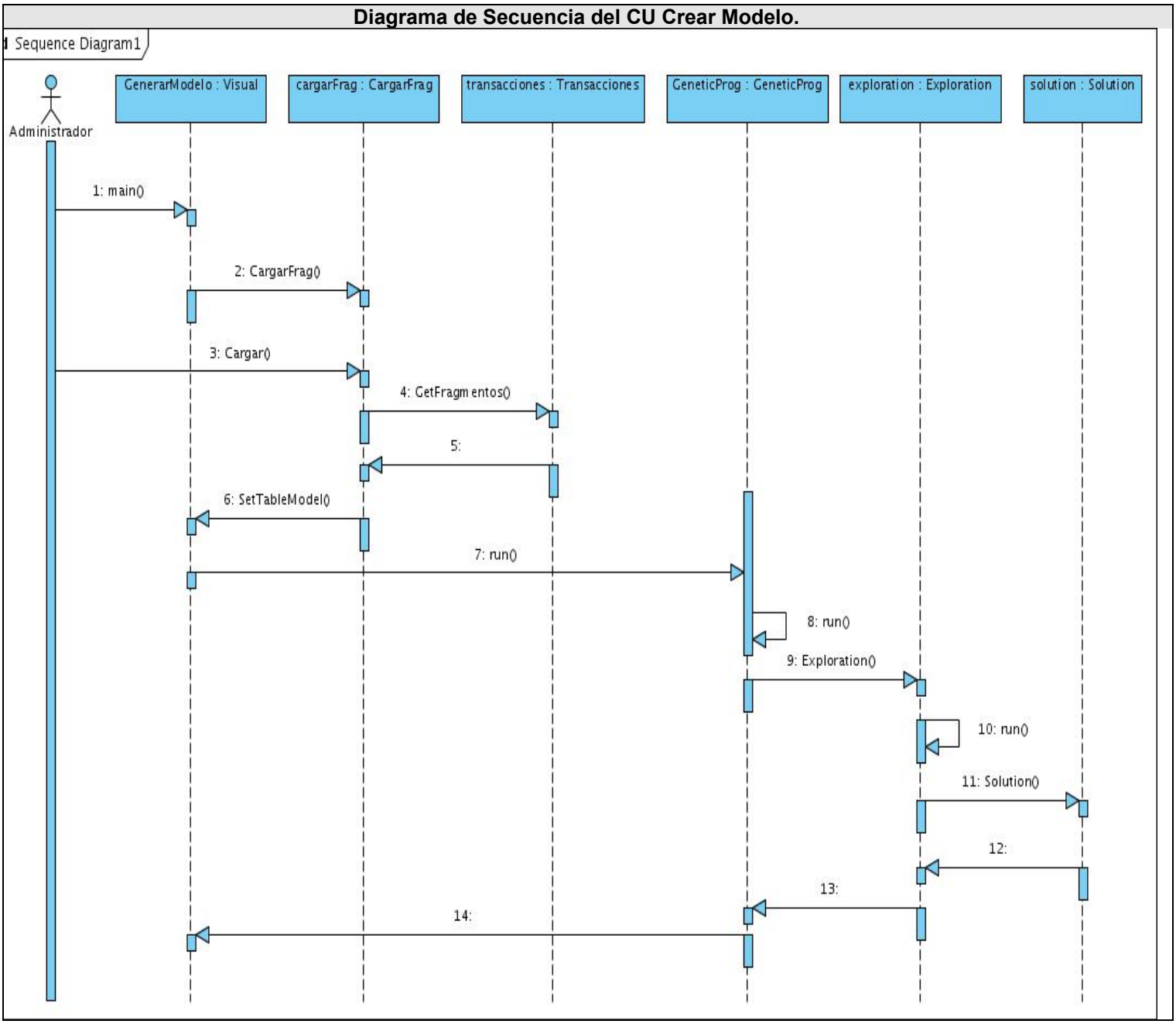


Fig 10. Diagrama de Secuencia del CU Crear Modelo.

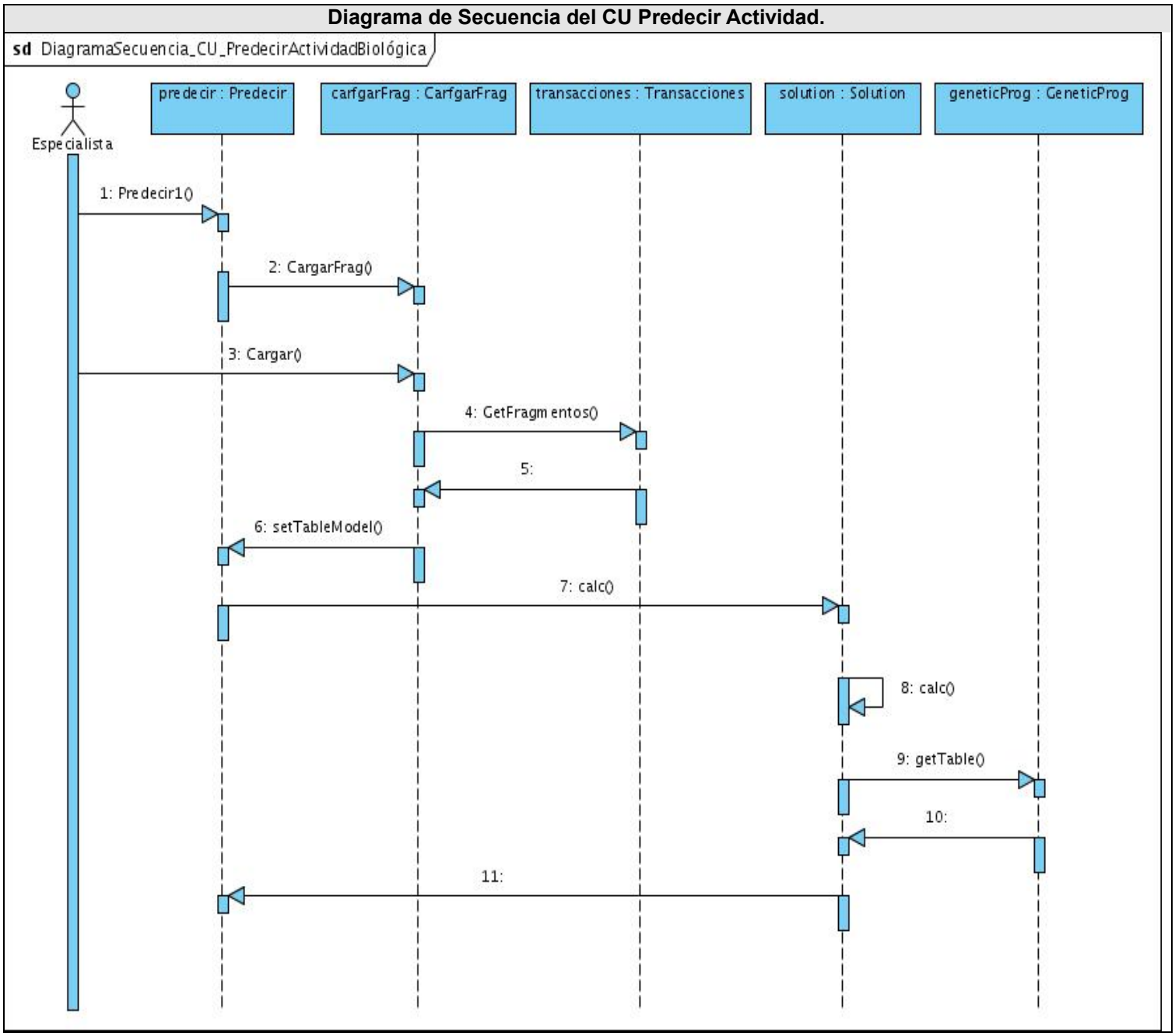


Fig 11. Diagrama de Secuencia del CU Predecir Actividad.

3.3.2 Diagramas de clases del diseño.

A continuación se muestran los diagramas de clases del diseño que muestra la relación entre las clases del diseño.

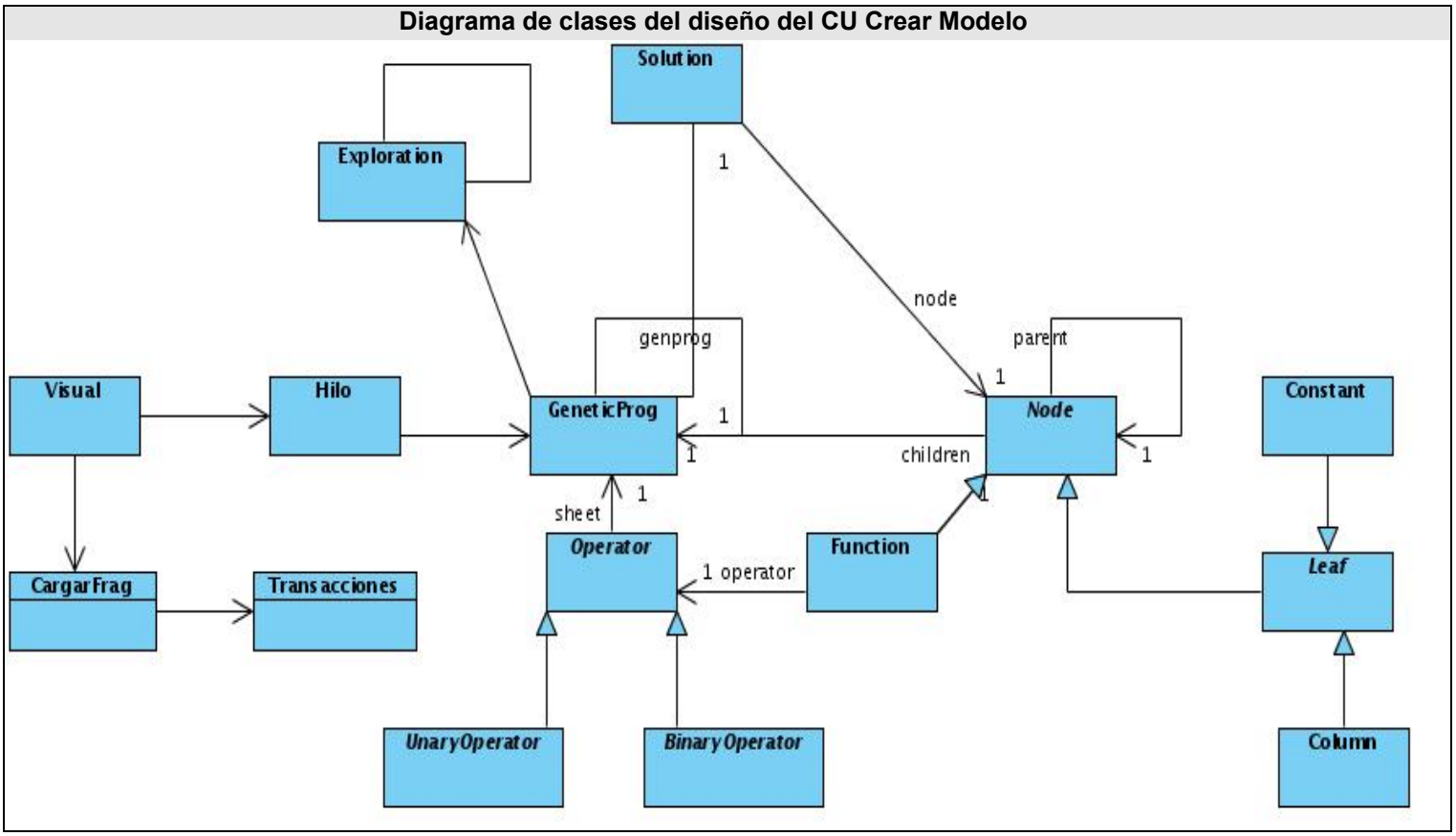


Fig 12. Diagrama de clases del diseño del CU Crear Modelo

Diagrama de clases del diseño del CU Predecir Actividad.

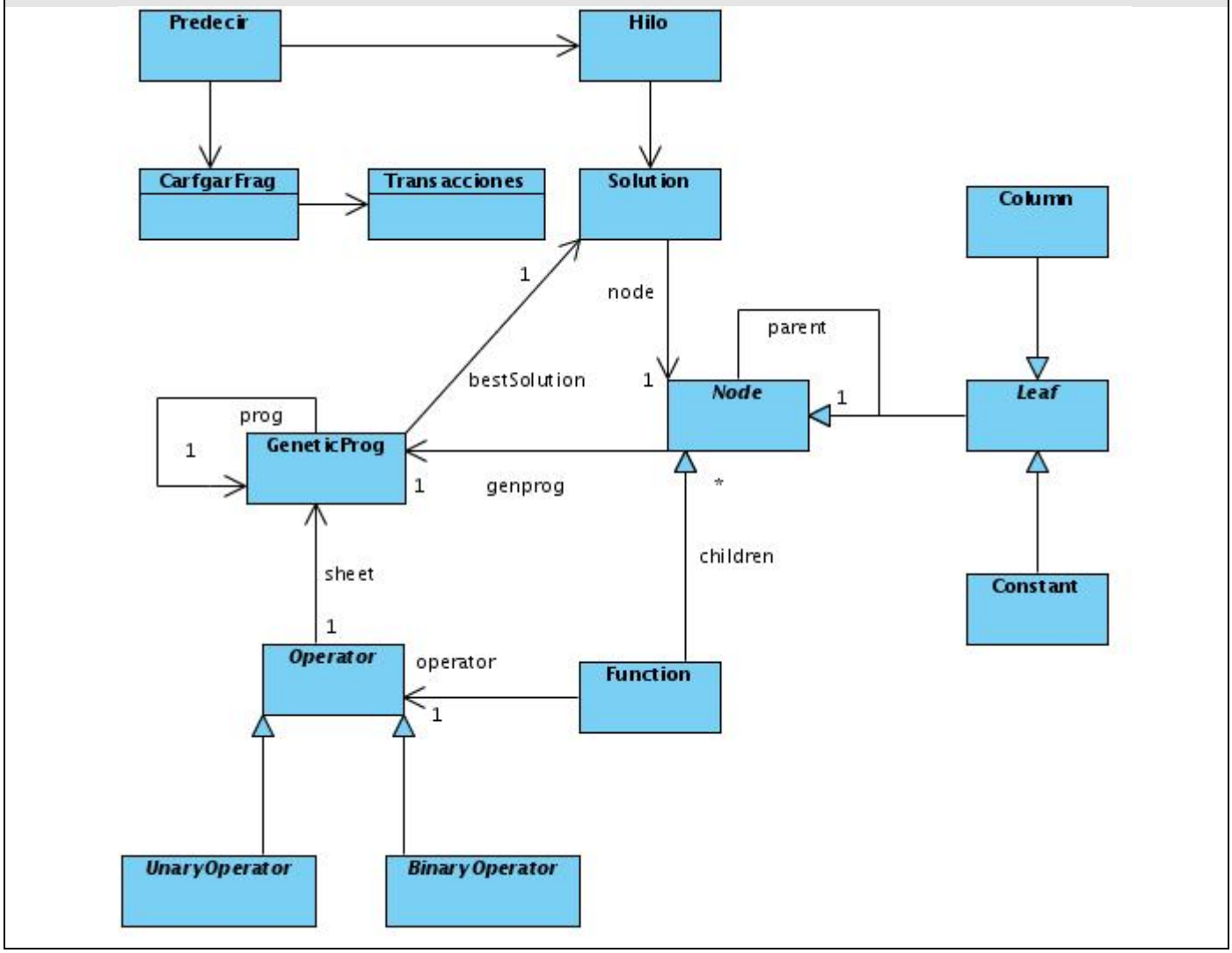


Fig 13. Diagramas de clases del diseño del CU Predecir Actividad.

3.4 Descripción de las clases del diseño.

Nombre: BinaryOperator	
Tipo de clase: Entidad	
Modificador: abstract	
Padre: Operator	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
argc()	int
calc(int rowIndex, double v1, double v2)	abstract Number

calc(int rowIndex, Node[] childs)	Number
getName()	String
Descripción:	
La clase BinaryOperator representa a los operadores matemáticos de tipo binario. En ella se encuentran los métodos que identifican a los operadores y los métodos para definir las operaciones.	

Nombre: Column	
Tipo de clase: Entidad	
Modificador: -	
Padre: Leaf	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
Column(GeneticProg genprog)	-
calc(int row)	Number
clone(Node parent)	Node
equals(Object o)	boolean
getNodeType()	Node.NodeType
muteIt()	void
print(PrintWriter out)	void
Descripción:	
La clase Column representa a las Variables del modelo matemático. En ellas están los métodos que identifican y devuelven las variables, además de los métodos para definir las operaciones principales como evaluar, mutar e imprimir.	

Nombre: Constant	
Tipo de clase: Entidad	
Modificador: -	
Padre: Leaf	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
Constant(GeneticProg genprog)	-
calc(int row)	Number
clone(Node parent)	Node
equals(Object o)	boolean
getNodeType()	Node.NodeType
muteIt()	void
print(PrintWriter out)	void
Descripción:	
La clase Constant representa a las Constantes Numéricas del modelo matemático. En ella se encuentran los métodos que identifican y devuelven a las constantes y los métodos para definir las operaciones principales como evaluar, mutar e imprimir.	

Nombre: Leaf	
Tipo de clase: Entidad	
Modificador: abstract	
Padre: Node	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
getChildAt(int index)	Node
getChildCount()	int
Descripción:	
La clase Leaf representa a las hojas del árbol que se emplea para representar el delo matemático.	

Nombre: Node	
Tipo de clase: Entidad	
Modificador: abstract	
Padre: -	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
calc(int row)	abstract Number
clone(Node parent)	abstract Number
countDescendant()	int
equals(Object o)	abstract boolean
getAllNodes()	Vector<Node>
getChildAt(int index)	abstract Node
getChildCount()	abstract int
getGeneticProg()	GeneticProg
getNodeType()	abstract Node.NodeType
getParent()	Node
getRoot()	Node
muteIt()	abstract void
print(PrintWriter out)	abstract void
toString()	String
Descripción:	
La clase Node representa a los nodos del árbol que se emplea para representar el modelo matemático. Los nodos pueden ser de tres tipos (COLUMN, CONSTANT, FUNCTION)	

Nombre: Operator
Tipo de clase: Entidad
Modificador: abstract
Padre: -
Implementa: Serializable
Principales Responsabilidad

Nombre	Tipo
argc()	abstract int
calc(int rowIndex, Node[] childs)	abstract Number
equals(Object o)	boolean
getName()	abstract String
getSpreadSheet()	GeneticProg
hashCode()	int
toString()	String
Descripción:	
La clase Operator representa a los operadores matemáticos.	

Nombre: Solution	
Tipo de clase: Entidad	
Modificador: -	
Padre: -	
Implementa: Serializable, Cloneable, Comparable<Solution>	
Principales Responsabilidad	
Nombre	**Tipo**
Solution(Node node, int generation)	-
calc()	double[]
Calculo()	double[]
clone()	Object
compareTo(Solution src)	int
equals(Object obj)	boolean
getGeneration()	int
getGeneticProg()	GeneticProg
getNode()	Node
getScore()	Double
mute()	void
toString()	String
Descripción:	
La clase Solution representa a las soluciones o modelos. En ella se encuentran los métodos que identifican al modelo y los métodos para definir las operaciones principales como evaluar, comparar e imprimir.	

Nombre: UnaryOperator	
Tipo de clase: Entidad	
Modificador: abstract	
Padre: Operator	
Implementa: Serializable	
Principal Responsabilidad	
Nombre	**Tipo**
argc()	int
calc(int rowIndex, double value)	abstract Number
calc(int rowIndex, Node[] childs)	Number

getName()	String
Descripción:	
La clase UnaryOperator representa a los operadores matemáticos de tipo unitario. En ella se encuentran los métodos que identifican al operador y los métodos para definir las operaciones.	

Nombre: Visual	
Tipo de clase: vista	
Modificador: -	
Padre: -	
Implementa: Runnable	
Principales Responsabilidad	
Nombre	**Tipo**
GenerarModeloPG()	-
main(String[] args)	static void
run()	void
Descripción:	
La clase Visual es la vista principal en la generación de modelo. Tiene como objetivo controlar las acciones solicitadas por el usuario y mostrarles los resultados de las mismas.	

Nombre: Predecir	
Tipo de clase: vista	
Modificador: -	
Padre: -	
Implementa: Runnable	
Principales Responsabilidad	
Nombre	**Tipo**
Predecir()	-
main(String[] args)	static void
read(BufferedReader in)	void
run()	void
Descripción:	
La clase Predecir es la vista principal en la predicción de la actividad. Tiene como objetivo controlar las acciones solicitadas por el usuario y mostrarles los resultados de las mismas.	

Nombre: GeneticProg	
Tipo de clase: Controladora	
Modificador: -	
Padre: -	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
GeneticProg()	-
getColumnCount()	int
getGenerationPerExplorer()	int
getMaxNANPercent()	double

getMinmax()	double[]
getNormalizedResultAt(int row)	Double
getNumberOfExplorer()	int
getNumberOfRun()	double
getRandom()	Random
getResultAt(int row)	Double
getRowCount()	int
getTable()	JTable
getValueAt(int row, int col)	Double
max_nodes_in_a_tree()	int
num_extra_parents()	int
num_parents()	int
proba_create_leaf()	double
proba_mutation()	double
read(BufferedReader in)	void
rnd()	double
run()	void
SalvarModeloSeriado()	void
SalvarModeloTexto()	void
setGenerationLabel(.JLabel generationLabel)	void
setTable(JTable table)	void
setTableModel(table.DefaultTableModel model)	void
setTableModelDefault(table.DefaultTableModel model)	void
setTextArea(JTextArea textarea)	void
setTextSVGPane(SVGPane svgPanes)	void
UpdateEXTRA_COLUMNS()	void
UpdateParam(int maxNodeInATree, int numberOfParent, double probaMutation, double probaLeaf, double maxNANPercent, int numberOfExplorer, int numberOfGeneration, double numberOfRun, int numSum, int numResta, int numMult, int numDiv, int numSQRT, int numLog, int numExp)	void
Descripción:	
La clase GeneticProg representa a la Programación Genética (PG). Su principal objetivo consiste en desarrollar y controlar el algoritmo de solución.	

Nombre: Transacciones	
Tipo de clase: acceso a datos	
Modificador: -	
Padre: -	
Implementa: -	
Principales Responsabilidad	
Nombre	**Tipo**
Conect()	void
close()	void
Transacciones(string usser, string password, string server, string database)	-

GetFragmentos(string Tipo, int longitud)	ResultSet
GetCantFragmentos(string Tipo, int longitud)	ResultSet
Descripción:	
La clase Transacciones es la clase de acceso a datos de la aplicación. Tiene como objetivo realizar las consultas a la base de datos para cargar los fragmentos especificados por el usuario a través la interfaz.	

Nombre: Function	
Tipo de clase: Entidad	
Modificador: -	
Padre: Node	
Implementa: Serializable	
Principales Responsabilidad	
Nombre	**Tipo**
Function(GeneticProg genprog, src.GeneticProg.Shuttle n)	-
calc(int row)	Number
clone(Node parent)	Node
equals(Object o)	boolean
getNodeType()	Node.NodeType
muteIt()	void
getChildAt(int index)	Node
getChildCount()	int
toString()	String
print(PrintWriter out)	void
Descripción:	
La clase Function representa a las funciones formadas por un Operador y sus parámetros correspondientes, estas funciones son las que conforman el modelo matemático. En ella se encuentran los métodos que identifican las funciones, y los métodos para definir las operaciones principales como evaluar, mutar e imprimir.	

Nombre: CargarFrag	
Tipo de clase: vista	
Modificador: -	
Padre: JFrame	
Implementa: -	
Principales Responsabilidad	
Nombre	**Tipo**
CargarFrag(Transacciones t, JPanel v)	-
main(String[] args)	static void
run()	void
Descripción:	
La clase CargarFrag es la vista que se le muestra al usuario para que entre la información necesaria para realizar las consultas a la base de datos.	

3.5 Estilo arquitectónico. Justificación.

Arquitectura en capas: Se utilizó la arquitectura en capas debido a que organiza el modelo de diseño a través de capas que pueden estar físicamente distribuidas, lo cual quiere decir que los componentes de una capa sólo pueden hacer referencia a componentes en capas inmediatamente inferiores. Este patrón es importante porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. La cantidad de niveles que se utilizarán serán tres (es decir tres capas). Las mismas se mencionan a continuación:

- Capa de presentación: representa las interfaces de la aplicación.
- Capa de acceso a datos: representa las clases controladoras y entidades del negocio.
- Capa de datos: representa la base de datos donde esté almacenada la información del sistema.

3.6 Definición de Diagrama de componentes.

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software. Los componentes pueden ser de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo. Los elementos de modelado dentro de un diagrama de componentes serán componentes y paquetes.

3.6.1 Diagrama de componentes por paquetes.

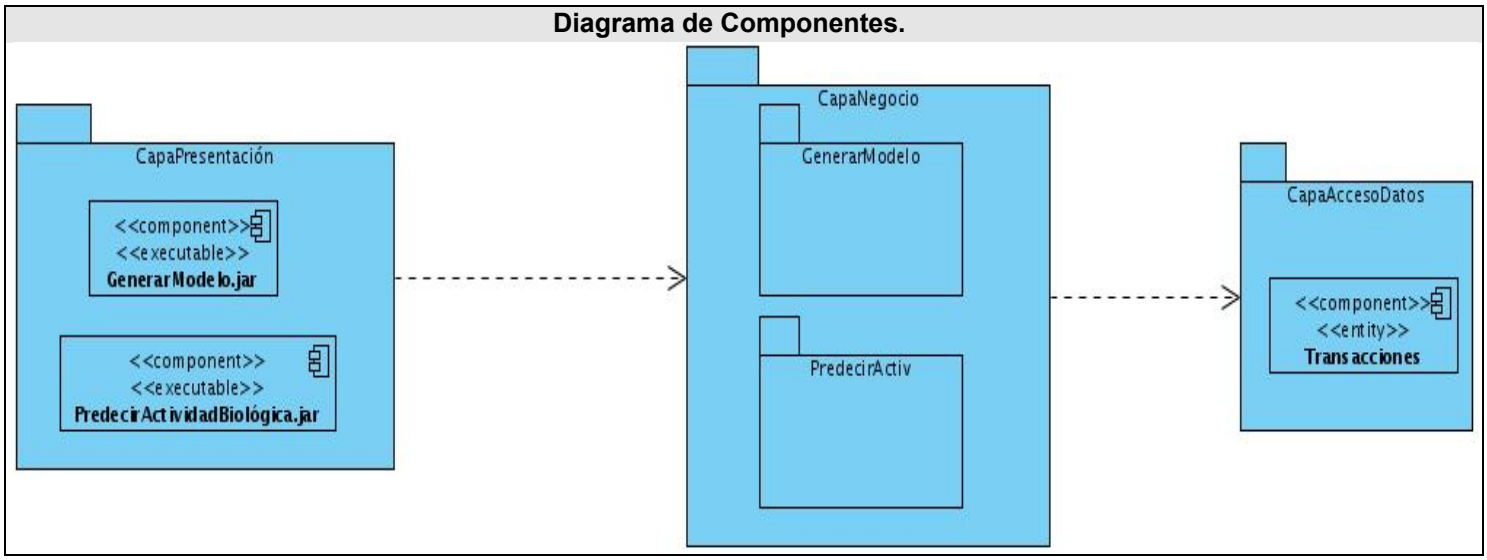


Fig 14. Diagrama de componentes.

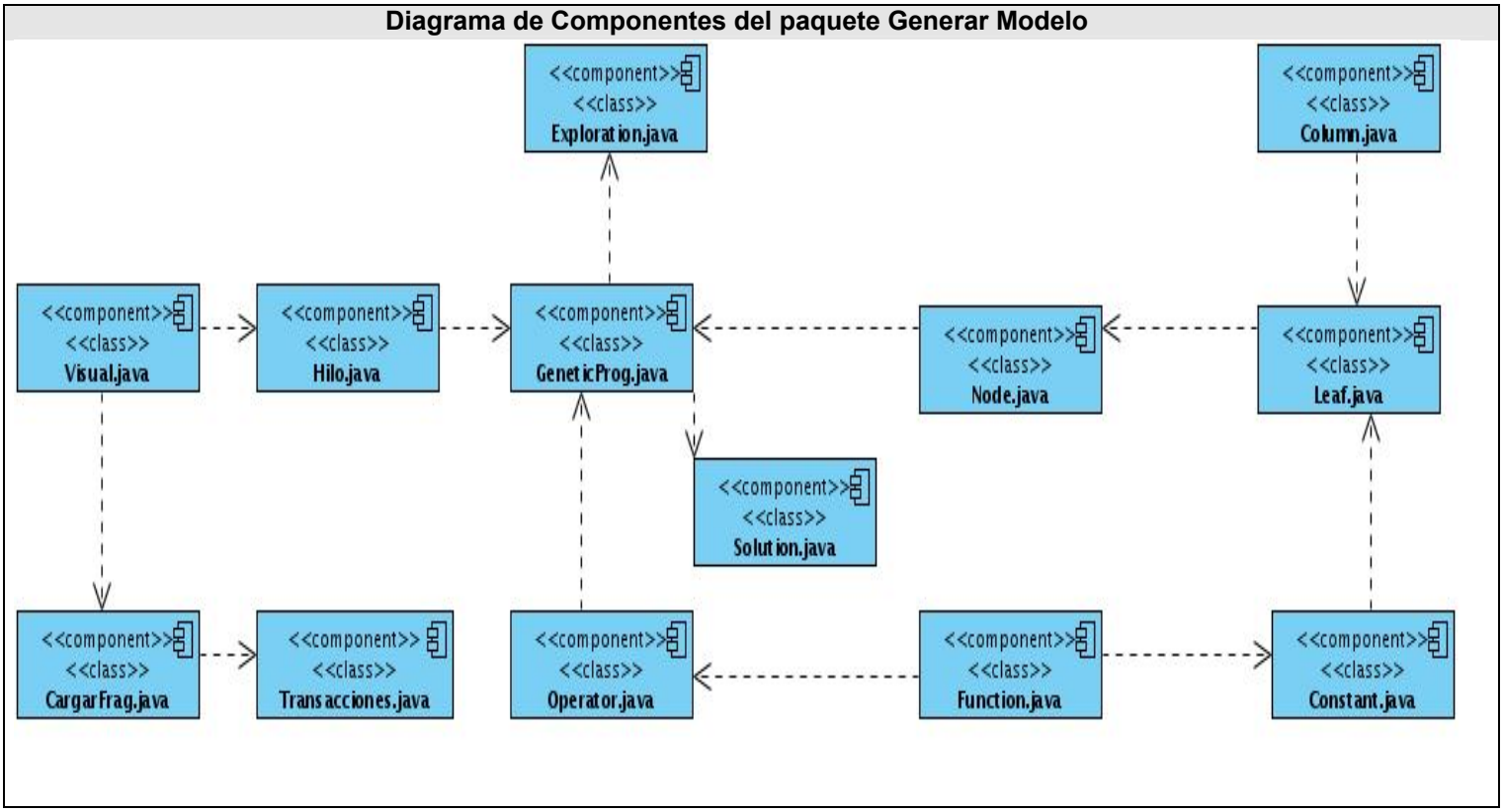


Fig 15. Diagrama de Componentes del paquete Generar Modelo.

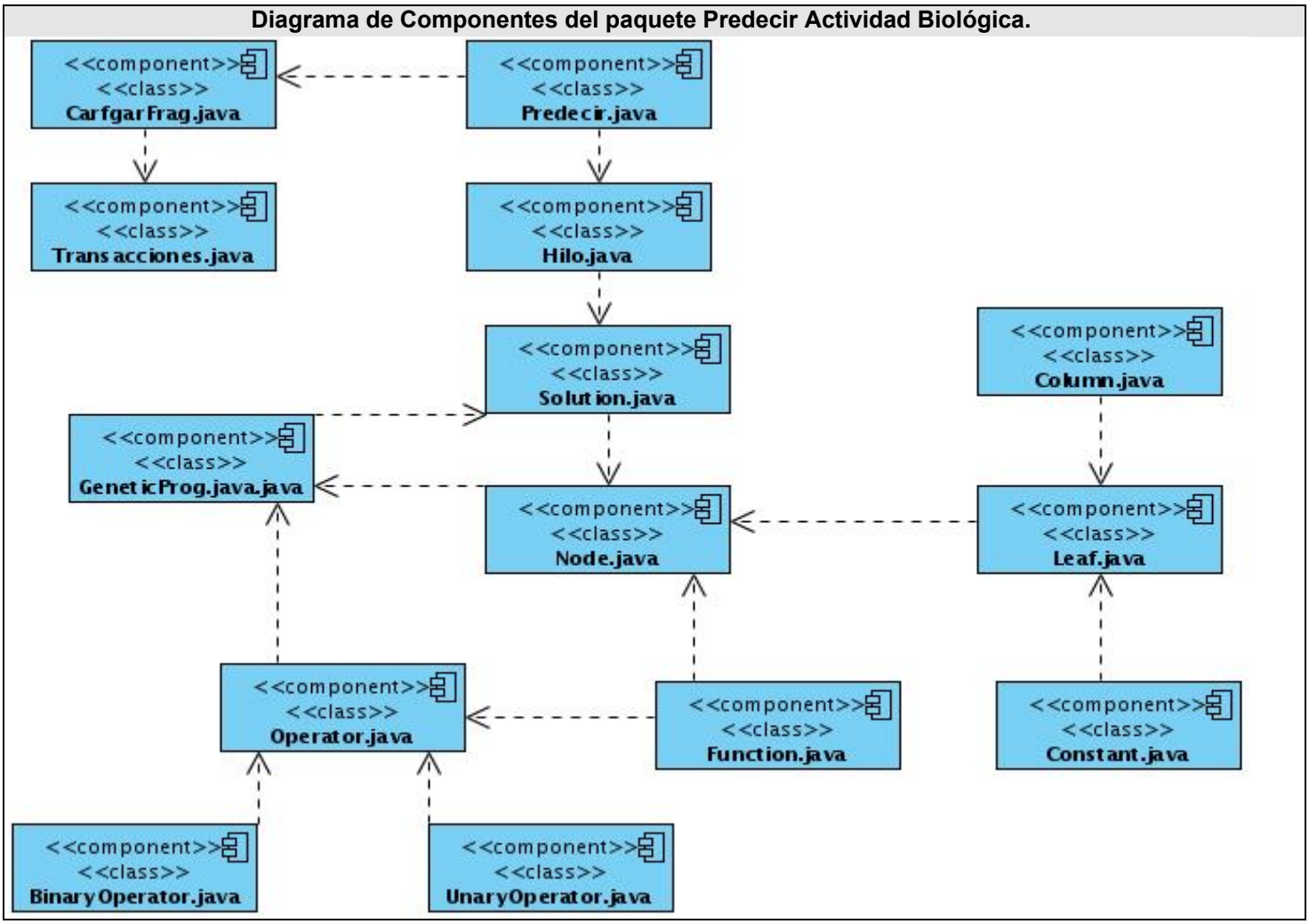


Fig 16. Diagrama de Componentes del paquete Predecir Actividad Biológica.

3.7 Conclusiones

En este capítulo se muestran los diagramas de clases del análisis, y los de interacción del diseño. Se describen las clases utilizadas en el diseño y se muestran los diagramas de clases del diseño por cada caso de uso. Se justifica el estilo arquitectónico usado y se muestra el diagrama de componentes por paquetes.

Capítulo 4: Análisis de los resultados.

4.1 Introducción.

En este capítulo se realizarán pruebas para analizar el funcionamiento de la aplicación desarrollada variando variables de entrada como el conjunto de funciones y los parámetros que controlan la corrida (por ejemplo, el número de individuos por generación, el tamaño de los individuos, etc.). En todas las pruebas realizadas se trabajó con una muestra de compuestos con actividad anticancerígena reportada, obtenidos de la base de datos del National Cancer Institute. Las muestras fueron seleccionadas por tipo de fragmentos y dividida cada una de ellas en 2 partes, la primera parte con el 80 por ciento del tamaño total, para generar el modelo, y el 20 por ciento restante para comprobar el mismo.

4.2 Resultado Experimental número 1.

Se utilizó una muestra de fragmentos de tipo Cluster de longitud 3, con 305 compuestos, de ellos 145, que representan el 80 % se utilizaron para generar el modelo y el resto (20%) para comprobarlo. La interfaz visual de la aplicación se cargó con los siguientes parámetros de entrada:

Máxima cantidad de nodos en un árbol: 23

Número de individuos por generación: 10

Probabilidad de Mutación: 0.95

Probabilidad de que un nodo sea hoja: 0.4

Número de Exploraciones: 10

Número de Generaciones por Exploración: 50

Número de Corridas: 3

Operadores que se utilizaron: +, -, /, *, √, log, exp.

Se obtuvo un modelo para Cluster de longitud 3 con un error de 0.109205, y al predecir con el resto de la muestra, de 61 compuestos se predijo que 28 eran activos frente al cáncer, para un porciento de predicción del 46%.

4.3 Resultado Experimental número 2.

Se utilizó una muestra de fragmentos de tipo Cluster de longitud 3, con 305 fragmentos, de ellos 6433, que representan el 80 % se utilizaron para generar el modelo y el resto (20%) para comprobar el modelo. La interfaz visual de la aplicación se cargó con los siguientes parámetros de entrada:

Máxima cantidad de nodos en un árbol: 23

Número de individuos por generación: 10

Probabilidad de Mutación: 0.95

Probabilidad de que un nodo sea hoja: 0.4

Número de Exploraciones: 10

Número de Generaciones por Exploración: 50

Número de Corridas: 3

Operadores que se utilizaron: +, -, /, *, √.

Se obtuvo un modelo para Cluster de longitud 3 con un error de 0.10, y al predecir con el resto de la muestra, de 61 compuestos se predijo que 34 eran activos frente al cáncer, para un porciento de predicción del 55.7%.

Al analizar el resultado experimental anterior se puede observar como mejora el porciento de predicción considerablemente, disminuyendo también el error cometido al calcular el modelo matemático. Por lo que se puede concluir que al variar las operaciones con las que se genera el modelo este puede ser beneficiado o afectado.

4.3 Resultado Experimental número 3.

Se utilizó una muestra de fragmentos de tipo Cluster_3_Camino de longitud 4, con 8042 fragmentos, de ellos 6433, que representan el 80 % se utilizaron para generar el modelo y el resto (20%) para comprobar el modelo. La interfaz visual de la aplicación se cargó con los siguientes parámetros de entrada:

Máxima cantidad de nodos en un árbol: 23

Número de individuos por generación: 10

Probabilidad de Mutación: 0.95

Probabilidad de que un nodo sea hoja: 0.4

Número de Exploraciones: 10

Número de Generaciones por Exploración: 50

Número de Corridas: 3

Operadores que se utilizaron: +, -, /, *, √, log, exp.

Se obtuvo un modelo para Cluster de longitud 3 con un error de 0.098, y al predecir con el resto de la muestra, de 1608 compuestos se predijo que 1019 eran activos frente al cáncer, para un porciento de predicción del 63.3%. En este experimento se puede observar como a medida que aumentan los tamaños de las muestras los resultados en las predicciones son mucho mejores.

4.4 Resultado Experimental número 4.

Se utilizó una muestra de fragmentos de tipo Cluster_3_Camino de longitud 5, con 11140 compuestos, de ellos el 80 % se utilizaron para generar el modelo y el resto (20%) para comprobar el modelo. La interfaz visual de la aplicación se cargó con los siguientes parámetros de entrada:

Máxima cantidad de nodos en un árbol: 23

Número de individuos por generación: 10

Probabilidad de Mutación: 0.95

Probabilidad de que un nodo sea hoja: 0.4

Número de Exploraciones: 5

Número de Generaciones por Exploración: 30

Número de Corridas: 3

Operadores que se utilizaron: +, -, /, *, √, log, exp.

Se obtuvo un modelo para Cluster_3_Camino de longitud 5 con un error de 0.10002, y al predecir con el resto de la muestra, de 2228 compuestos se predijo que 1415 eran activos frente al cáncer, para un porciento de predicción del 63.5%.

4.5 Resultado Experimental número 5.

Se utilizó una muestra de fragmentos de tipo Cluster_3_Camino de longitud 6, con 14613 compuestos, de ellos 11690 que representan el 80 % se utilizaron para generar el modelo y el resto (20%) para comprobar el mismo. La interfaz visual de la aplicación se cargó con los siguientes parámetros de entrada:

Máxima cantidad de nodos en un árbol: 23

Número de individuos por generación: 10

Probabilidad de Mutación: 0.95

Probabilidad de que un nodo sea hoja: 0.4

Número de Exploraciones: 5

Número de Generaciones por Exploración: 20

Número de Corridas: 3

Operadores que se utilizaron: +, -, /, *, √, log, exp.

Se obtuvo un modelo para Cluster_3_Camino de longitud 6 con un error de 0.10218, y al predecir con el resto de la muestra, de 2922 compuestos se predijo que 1906 eran activos frente al cáncer, para un porciento de predicción del 65%. Se debe resaltar además que fueron variados los parámetros de entrada, disminuyendo el número de corridas, generaciones e individuos por generación, y los resultados de la predicción son mejores que en experimentos anteriores, además de que la muestra utilizada es la mayor analizada hasta el momento, por lo que se concluye una vez más que las muestras grandes mejoran los modelos matemáticos, y en el caso de los parámetros se puede decir que su variación influye en el modelo.

4.6 Conclusiones.

Se realizaron 6 experimentos con diferentes muestras, obteniéndose un modelo por experimento. En cada uno de los casos se analizó los factores que influían en los modelos matemáticos. Entre los análisis que se hicieron se pudo observar que a medida que aumenta el tamaño de las muestras mejoran la calidad de los modelos que describen la actividad biológica. También se analizó que los parámetros de entrada que controlan el algoritmo influyen en los resultados, beneficiándolos o afectándolos. Cuando se variaban las operaciones matemáticas con las que se genera el modelo, estos últimos también se beneficiaban en algunos casos y se afectaban en otros. Las mejores predicciones en la mayoría de los experimentos se lograban con funciones matemáticas básicas (+,- , *, /, √).

Conclusiones Generales

✓ Se analizó, diseño e implementó una aplicación que permite generar modelos matemáticos que describen la actividad biológica anticancerígena a través de Programación Genética, basada en fragmentos ponderados por el Índice del Estado Refractotopológico Total y guardar los modelos generados.

✓ Se analizó, diseño e implementó una aplicación que permite predecir actividad biológica anticancerígena en fragmentos ponderados por el Índice del Estado Refractotopológico Total y guardar las predicciones.

✓ Se generaron 5 modelos predictivos de la actividad biológica anticancerígena modificando el tamaño de la muestra, los parámetros de entrada y las operaciones. La comprobación en las muestras de prueba correspondientes mostró la dependencia de los resultados con el tamaño de la muestra, los parámetros y las operaciones utilizadas para su generación

✓ Se realizaron pruebas para comprobar la validez de los modelos matemáticos encontrados.

✓ Los valores de los errores relativos en los modelos se encuentran entre el 8-10% para las muestras de entrenamiento lo cual resulta aceptable para esta primera versión de los resultados.

✓ Los valores en las predicciones también son aceptables, siendo los fragmentos predichos con condiciones muy rigurosas lo que provoca que los porcientos de predicción sean más bajos y se encuentren entre 46% y 67 %.

Recomendaciones

✓ Lograr incorporar más información sobre los fragmentos en la base de datos para poder mejorar las predicciones.

✓ Generar los modelos con muestras grandes para lograr que estos puedan describir mejor la actividad biológica.

✓ Utilizar la GRID para hacer cálculos distribuidos y mejorar el tiempo de generación de los modelos.

✓ Trabajar con otros tipos de errores que sean más bondadosos a la hora de predecir para incluir aquellos fragmentos que poseen menos actividad pero que son considerados activos.

Referencias Bibliográficas.

1. JANSSEN-CILAG. *Investigación de Janssen-Cilag en Toledo* Última actualización: 30 December [Consultado el: 19 de enero de 2007]. Disponible en: http://www.janssen-cilag.es/bgdisplay.jhtml?itemname=research.

2. ABRAHAM, D. J. *History of Quantitative Structure-Activity Relationships* California: [Consultado el: 22 de noviembre de 2006]. Disponible en: http://media.wiley.com/product_data/excerpt/03/04712709/0471270903.pdf. ISBN 0-471-27090-3.

3. KUBINYI, H. QSAR: Hansh analysis and approaches 1993, nº

4. M DIUDEA. QSPR/QSAR Studies by Molecular Descriptors. *Ed. Nova Science Publishes Inc.*, 2001, nº

5. VELAR, R. C. *Introducción al diseño de Fármacos. [Folleto para la docencia de la asignatura de Farmacia].* Universidad de Oriente:

6. ---. *Nuevos índices híbridos para el estudio de estructura actividad. Tesis para optar por el título de Doctor en Ciencias Químicas.* Habana: 2003.

7. CAMPOLLO, R. *Modelos matemáticos en medicina y biología* [Consultado el: 20 de noviembre de 2006]. Disponible en: http://www.imbiomed.com/Innsz/Nnv46n4/espanol/Wnn44-07.html.

8. MARTINS, F. *Utilización de QSAR en el diseño de moléculas farmacológicamente activas* Última actualización: mayo [Consultado el: febrero de 2007]. Disponible en: http://www.dqb.fc.ul.pt/cadeiras/qfina/documentos/Aula%20Qu%C3%ADmica%20Fina%202005.pdf.

9. BUERA, M. D. P. *Relaciones estructura-actividad cuantitativas (QSAR)* Buenos Aires: Última actualización: 6 de junio 2005. Disponible en: http://www.qo.fcen.uba.ar/Cursos/quimed_files/QM7.pdf.

10. PETTIS, S. L. *Aplicación para el control de un motor genérico.* Tesis de doctorado, Universidad de Belgrano, 2005.

11. KOZA, J. R. *Sitios oficiales de* California: [Consultado el: 18 de junio]. Disponible en: http://www.genetic-programming.com/, http://www.genetic-programming.org

12. WOO, D. Y.-T. *OnCologic* [Consultado el: enero de 2007]. Disponible en: http://www.epa.gov/oppt/cahp/actlocal/can.htm.

13. COMPUDRUG INTERNATIONAL, I. *Hazardexpert y Metabolexpert* [Consultado el: 12 de febrero de 2007]. Disponible en: http://www.compudrug.com.

14. VALERY GOLENDER y VESTERMAN, B. *Apex 3D* San Diego: [Consultado el: febrero de 2007]. Disponible en: http://www.netsci.org/Science/Compchem/feature09.html

15. CORPORATION, C. *About CambridgeSoft Desktop Software* [Consultado el: 27 de marzo de 2007]. Disponible en: http://www.cambridgesoft.com/about/profile/DesktopSoftware/.

16. MAGAZINE, C. P. *Adapt or perish.* 2004, Disponible en: http://www.chemicalprocessing.com/articles/2004/303.html

17. LARMAN, C. *UML y Patrones.* 1999.

18. *Visual Paradigm for UML* Programación en castellano, Última actualización: 5 de julio de 2005. [Consultado el: 5 de marzo de 2007]. Disponible en: http://www.programacion.com/noticia/1363/. .

19. EXEC. *Características del lenguaje Java* [Consultado el: 8 de noviembre de 2006]. Disponible en: http://www.mailxmail.com/curso/informatica/java/capitulo2.htm

20. J TRAMULLAS y KRONOS, E. *Los sistemas de gestión de bases de datos* Última actualización: 1997-2000. [Consultado el: 5 de marzo de 2007]. Disponible en: http://tramullas.com/documatica/2-4.html. .

Bibliografía

1. *Visual Paradigm for UML* Programación en castellano, Última actualización: 5 de julio de 2005. [Consultado el: 5 de marzo de 2007]. Disponible en: http://www.programacion.com/noticia/1363/. .

2. ABRAHAM, D. J. *History of Quantitative Structure-Activity Relationships* California: [Consultado el: 22 de noviembre de 2006]. Disponible en: http://media.wiley.com/product_data/excerpt/03/04712709/0471270903.pdf. ISBN 0-471-27090-3.

3. BUERA, M. D. P. *Relaciones estructura-actividad cuantitativas (QSAR)* Buenos Aires: Última actualización: 6 de junio 2005. Disponible en: http://www.qo.fcen.uba.ar/Cursos/quimed_files/QM7.pdf.

4. C HANSCH y FUJITA, T. Method for Correlation of Biological Activity and Chemical Structure. . *Journal of American Chemical Society*, 1964, nº

5. CAMPOLLO, R. *Modelos matemáticos en medicina y biología* [Consultado el: 20 de noviembre de 2006]. Disponible en: http://www.imbiomed.com/Innsz/Nnv46n4/espanol/Wnn44-07.html.

6. COMPUDRUG INTERNATIONAL, I. *Hazardexpert y Metabolexpert* [Consultado el: 12 de febrero de 2007]. Disponible en: http://www.compudrug.com.

7. EXEC. *Características del lenguaje Java* [Consultado el: 8 de noviembre de 2006]. Disponible en: http://www.mailxmail.com/curso/informatica/java/capitulo2.htm

8. FONT, M. *El diseño racional de fármacos asistido por computadora*

9. HERNÁNDEZ, S. Á. *Metodología para el desarrollo de aplicaciones con tecnología Orientada a Objetos utilizando notación UML.* La Habana: 2000.

10. J TRAMULLAS y KRONOS, E. *Los sistemas de gestión de bases de datos* Última actualización: 1997-2000. [Consultado el: 5 de marzo de 2007]. Disponible en: http://tramullas.com/documatica/2-4.html. .

11. JANSSEN-CILAG. *Investigación de Janssen-Cilag en Toledo* Última actualización: 30 December [Consultado el: 19 de enero de 2007]. Disponible en: http://www.janssen-cilag.es/bgdisplay.jhtml?itemname=research.

12. JORGE MARTÍN MARTÍN y MORATE, D. G. *Seminario: algoritmos genéticos* Valladolid: Disponible en:

http://www.infor.uva.es/~calonso/IAI/TrabajoAlumnos/memoriaAG.pdf.

13. KOZA, J. R. *Sitios oficiales de* California: [Consultado el: 18 de junio]. Disponible en: http://www.genetic-programming.com/, http://www.genetic-programming.org

14. KOZA., J. R. *Genetic programming: on the programming of computers by means of natural selection.* 1992.

15. KUBINYI, H. QSAR: Hansh analysis and approaches 1993, nº

16. LARMAN, C. *UML y Patrones.* 1999.

17. LEAD, P. P. *Introducción a la plataforma Eclipse* [Consultado el: 25 de marzo

18. M DIUDEA. QSPR/QSAR Studies by Molecular Descriptors. *Ed. Nova Science Publishes Inc.*, 2001, nº

19. MAGAZINE, C. P. *Adapt or perish.* 2004, Disponible en: http://www.chemicalprocessing.com/articles/2004/303.html

20. MARCZYK, A. *Algoritmos genéticos y computación evolutiva.* 2004, Disponible en: http://the-geek.org/docs/algen/.

21. MARTINS, F. *Utilización de QSAR en el diseño de moléculas farmacológicamente activas* Última actualización: mayo [Consultado el: febrero de 2007]. Disponible en: http://www.dqb.fc.ul.pt/cadeiras/qfina/documentos/Aula%20Qu%C3%ADmica%20Fina%202005.pdf.

22. MOLINERO, L. M. *Construcción de modelos de regresión multivariantes* Última actualización: Abril [Consultado el: febrero de 2007]. Disponible en: http://www.seh-lelha.org/regresion1.htm.

23. NATURALES, D. D. C. E. Y. *Relaciones estructura-actividad cuantitativas (QSAR)* Buenos Aires: Última actualización: 6 de junio de 2005. [Consultado el: 22 de mayo de 2007]. Disponible en: http://www.qo.fcen.uba.ar/Cursos/quimed_files/QM7.pdf.

24. PETTIS, S. L. *Aplicación para el control de un motor genérico.* Tesis de doctorado, Universidad de Belgrano, 2005.

25. ROMERO, A. V. G. *SISTEMA PARA PREDICCIÓN ACTIVIDAD BIOLÓGICA DE COMPUESTOS ORGÁNICOS.* Tesis de Diploma, Instituto Superior Politécnico “José Antonio Echeverría”, 2006.

26. SANCHEZ, J. F. *2010: el Nuevo listón de la información* [Consultado el: mayo de 2007]. Disponible en: http://www-5.ibm.com/es/press/notas/2002/diciembre/farma.html

27. SM FREE, J. W. *Mathematical Contribution to Structure-Activity studies Med Chem* [Consultado el: abril de 2007].

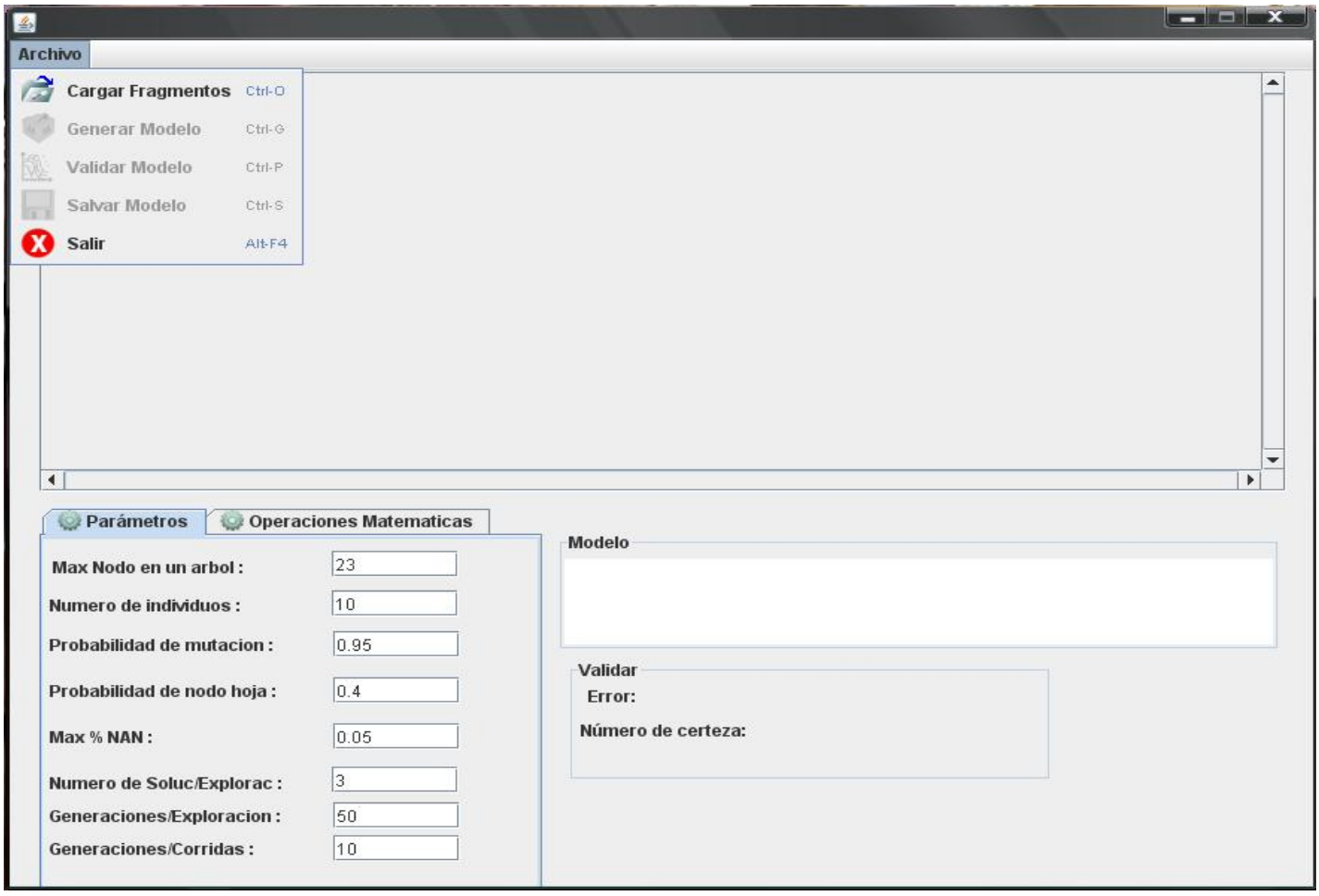
28. VALERY GOLENDER y VESTERMAN, B. *Apex 3D* San Diego: [Consultado el: febrero de 2007]. Disponible en: http://www.netsci.org/Science/Compchem/feature09.html

29. VELAR, R. C. *Introducción al diseño de Fármacos. [Folleto para la docencia de la asignatura de Farmacia].* Universidad de Oriente:

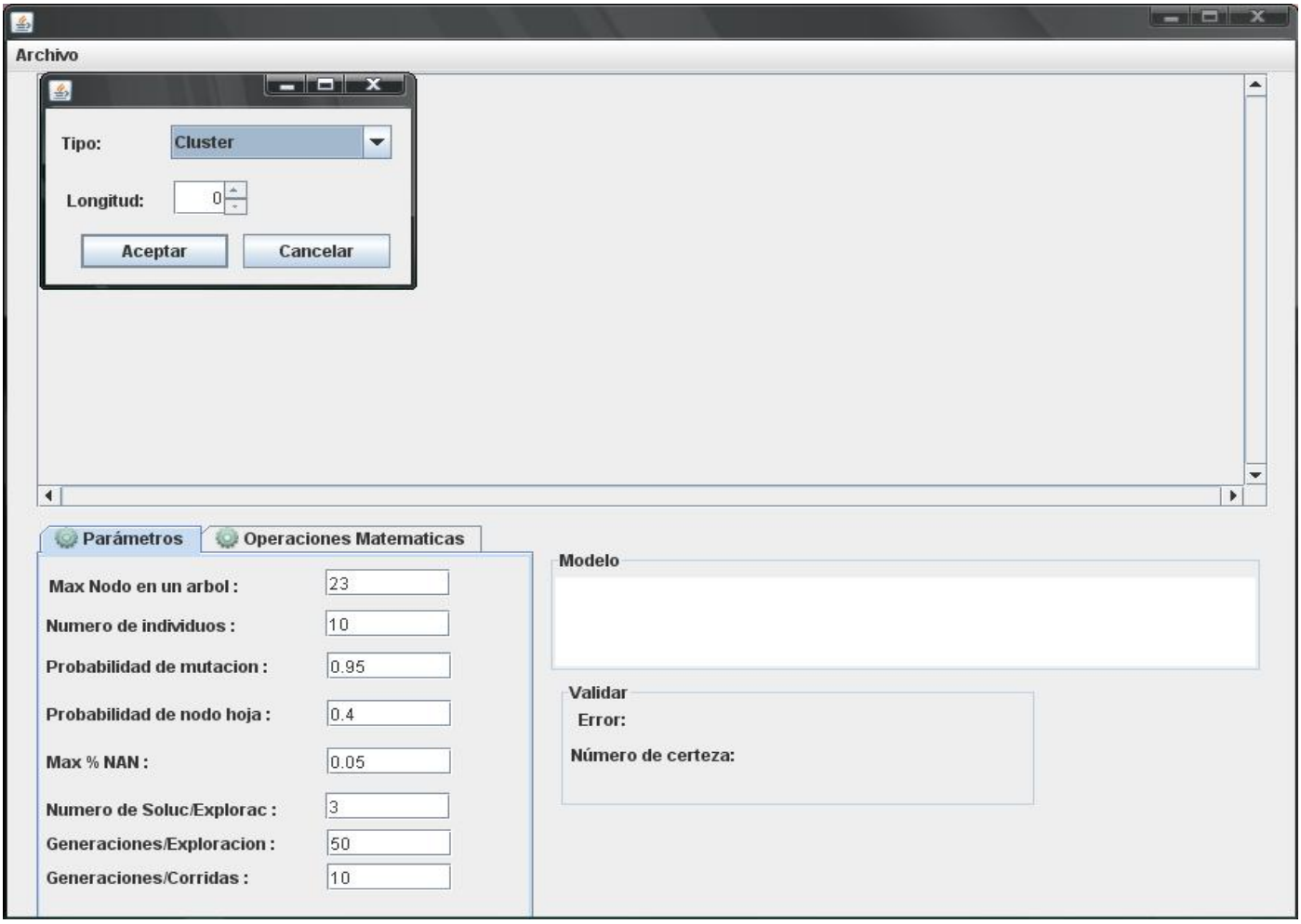
30. ---. *Nuevos índices híbridos para el estudio de estructura actividad. Tesis para optar por el título de Doctor en Ciencias Químicas.* Habana: 2003.

31. WOO, D. Y.-T. *OnCologic* [Consultado el: enero de 2007]. Disponible en: http://www.epa.gov/oppt/cahp/actlocal/can.htm.

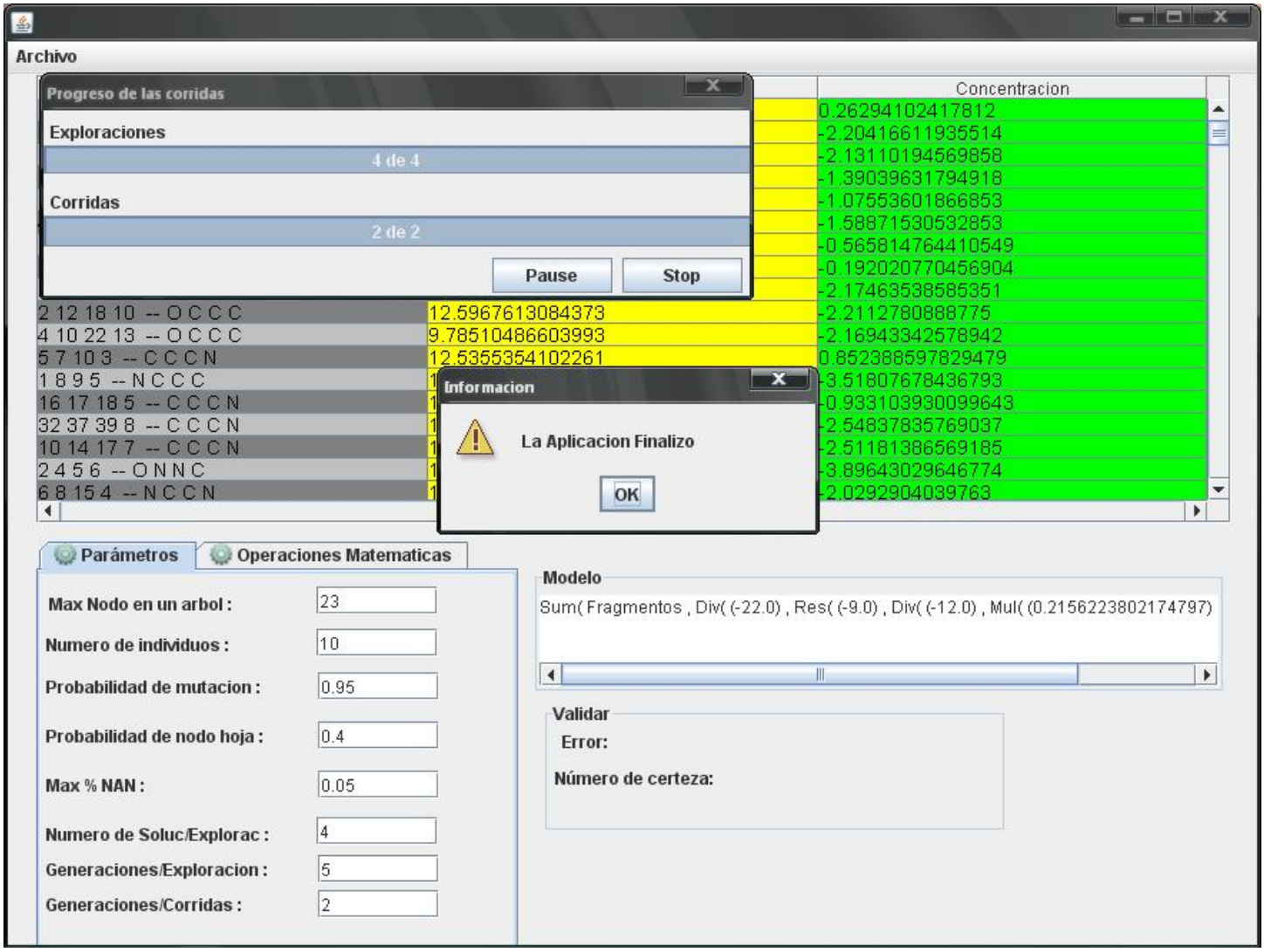
Anexos



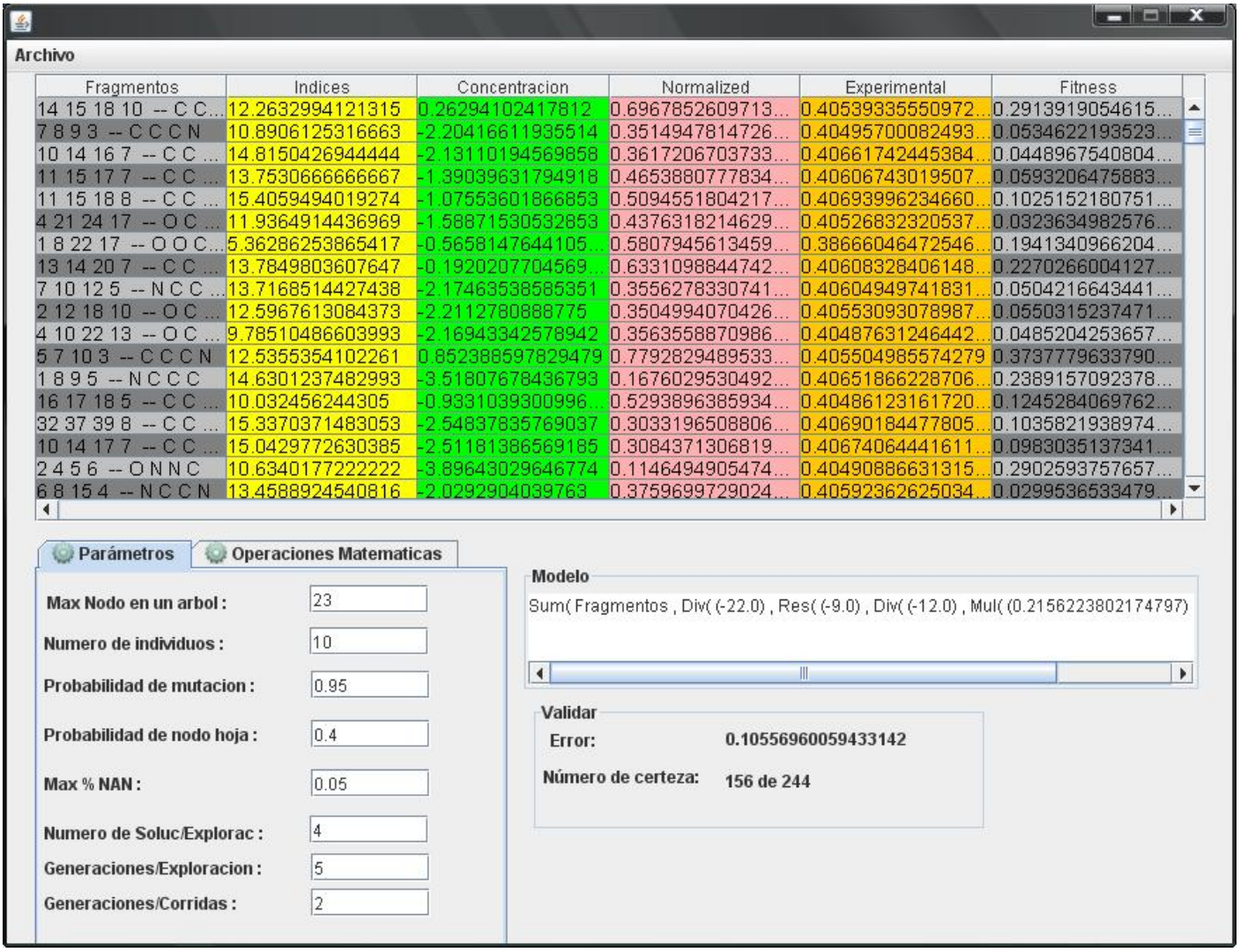
Anexo 1. Interfaz de la aplicación para crear modelos.



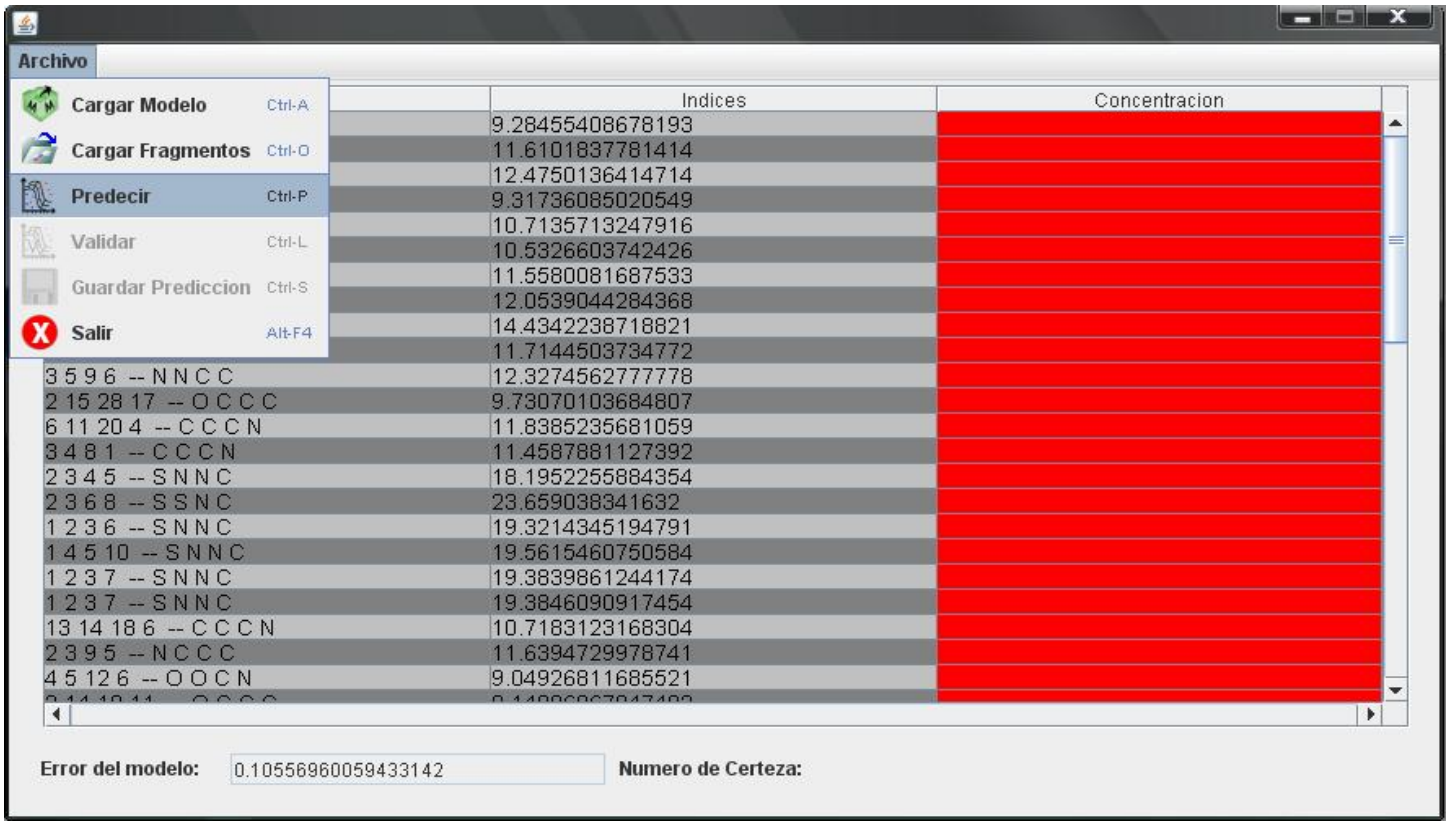
Anexo 2. Interfaz de la aplicación para crear modelo consultando los fragmentos de la base de datos del proyecto para cargarlos.



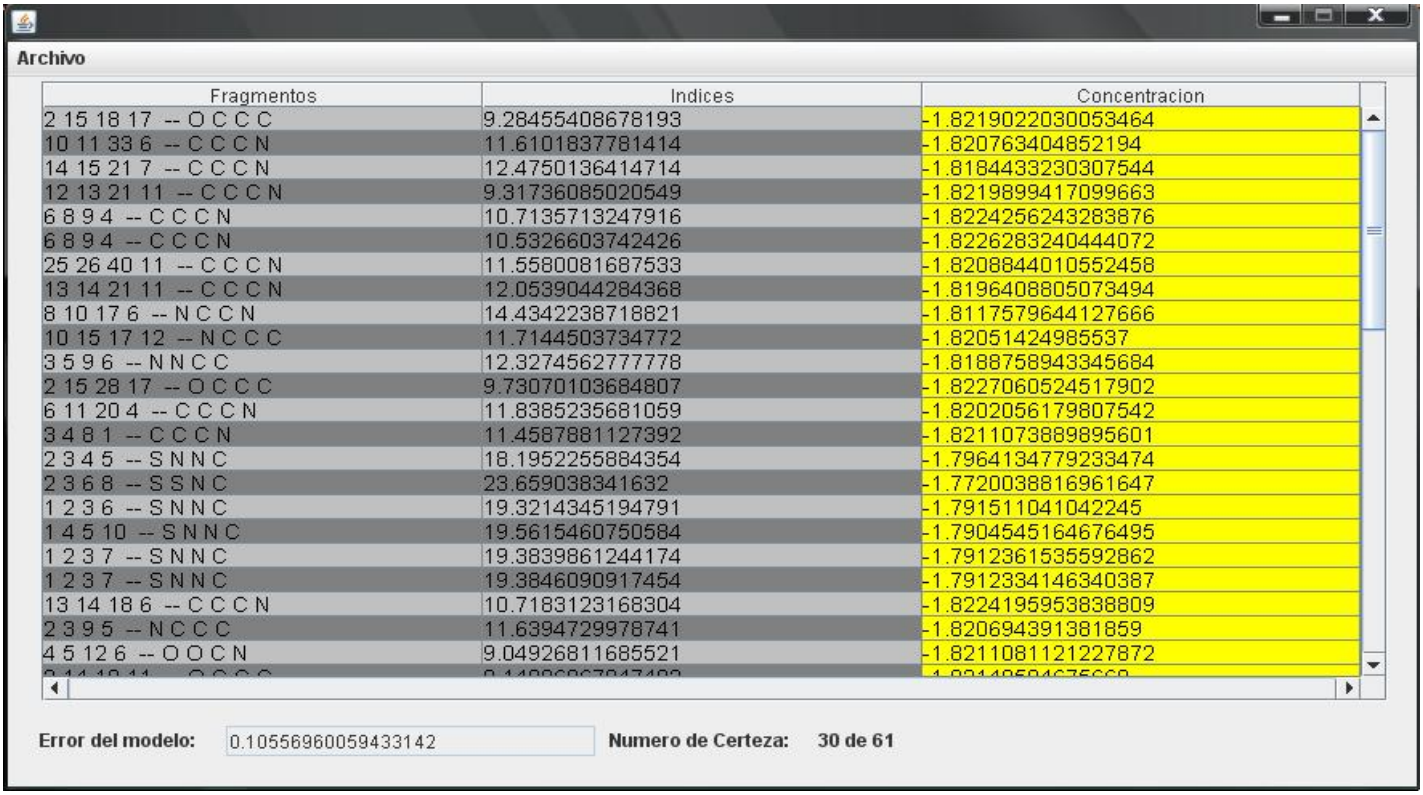
Anexo 3. Interfaz de la aplicación para crear modelo luego de generar el modelo.



Anexo 4. Interfaz de la aplicación para crear modelo comprobando el modelo.



Anexo 5. Interfaz de la aplicación para predecir actividad biológica en fragmentos.



Anexo 5. Interfaz de la aplicación para predecir actividad luego predecir la concentración de los fragmentos y comprobarlos con los valores reales tomados de la base de datos.

Glosario de Términos

Sintetizar: Proceso de obtención de un compuesto a partir de sustancias más sencillas.
Moléculas: Una molécula es una partícula formada por un conjunto de átomos ligados por enlaces covalentes.
Átomos: Es la entidad química más pequeña, el mismo está compuesto de protones y neutrones.
Diagramas de dispersión: Es el diagrama que se utiliza principalmente en el estudio conjunto de dos variables, para saber si existe algún tipo de relación entre ellas.
Histograma: Representación gráfica de una distribución de frecuencias por medio de rectángulos, cuyas anchuras representan intervalos de la clasificación y cuyas alturas representan las correspondientes frecuencias.
Descriptor: Número que describe la estructura química o una propiedad de la molécula o fragmento de esta.
Índices: Contienen información relacionada con la forma molecular, el grado de ramificación, tamaño molecular y la flexibilidad estructural.
Índice del Estado Refractotopológico Total: Se define como la suma de los valores del Índice del Estado Refractotopológico de cada átomo del fragmento considerado en una molécula dada.
Índice Topológico: Número que se calcula generalmente a partir de la matriz de adyacencia o de distancia de los elementos de un grafo molecular.
Plug-in: Es una aplicación informática que interactúa con otra aplicación para aportarle una función o utilidad específica, generalmente muy específica, como por ejemplo servir como driver en una aplicación, para hacer así funcionar un dispositivo en otro programa.
Ponderar: Determinar el peso de algo. Atribuir un peso a un elemento de un conjunto con el fin de obtener la media ponderada.
Ligando: Iones o molécula que rodean a un metal en un complejo. Un ligando enlazado a un ion central se dice que está coordinado al ion.
CASE: Acrónimo inglés de Computer Aided Software Engineering, que significa Ingeniería de **Software Asistida por Ordenador.**
Lipofílica: Propiedad de las moléculas de disolverse en grasas.
Estérica: Término utilizado para describir el tamaño y el volumen de fragmentos, moléculas.
Congenéricas: Término que se utiliza para nombrar a las series del mismo género.
Toxicológicos: Perteneciente o relativo a la toxicología. Estudio de las sustancias tóxicas y sus efectos.

Compuestos orgánicos: Los compuestos o moléculas orgánicas son los compuestos químicos basados en Carbono, Hidrógeno y Oxígeno, y muchas veces con Nitrógeno, Azufre, Fósforo, Boro, Halógenos.