Universidad Michoacana de San Nicolás de Hidalgo

Facultad de Ciencias Físico-Matemáticas
"Mat. Luis Manuel Rivera Gutiérrez"

Desarrollo de software de apoyo para la generación de ambientes tecnológicos interactivos para el razonamiento matemático

Tesis
Para obtener el grado de
Licenciado en Ciencias Físico Matemáticas

Presenta
Josué Daniel González Parra

Asesores:
Dr. José Carlos Cortés Zavala

Dra. Graciela Eréndira Núñez Palenius

Morelia Michoacán, abril 2013

"[. . .] Pasaste por varias transformaciones, fuiste tan poderoso que todos te odiamos. Nos obligaste a dar lo mejor de nosotros mismos para acabar contigo."
Son Gokú

Dedicatoria

A Dios, *por darme la oportunidad de vivir y por estar conmigo en cada paso que doy, por fortalecer mi corazón, iluminar mi mente y por haber puesto en mi camino a aquellas personas que han sido mi soporte y compañía durante toda mi vida.*

A mi familia, *por haberme apoyado en todo momento, por sus consejos, sus valores, la motivación constante, por los ejemplos de perseverancia y constancia que los caracterizan y que me han infundado siempre, pero más que nada, por su amor.*

A mis maestros, *por su gran apoyo y motivación para la culminación de mis estudios y para la elaboración de esta tesis; por su tiempo compartido y por impulsar el desarrollo de mi formación profesional y personal.*

A mis amigos, *que nos hemos apoyado mutuamente en nuestra formación profesional y que hasta ahora, seguimos estando juntos.*

Agradecimientos

Al finalizar un trabajo tan arduo y lleno de dificultades como el desarrollo de una tesis, es inevitable que nos invada un, muy humano, egocentrismo que nos lleva a concentrar la mayor parte del mérito en el aporte que hemos hecho. Sin embargo, haciendo un análisis objetivo nos muestra inmediatamente que la magnitud de ese aporte hubiese sido imposible sin la participación de personas e instituciones que faciliten las cosas para que este trabajo llegue a un feliz término. Por ello, es para mí un verdadero placer utilizar este espacio para ser justo y consecuente con ellas, expresándoles mis agradecimientos.

Quiero agradecer a mi novia Erika Ruiz, mil gracias por acompañarme en este proceso, pero sobre todo por tu cariño, tu comprensión, paciencia y fortaleza que permite que pueda realizar y acabar todos mis proyectos. Como en todo lo que hago, estás presente en mi mente y, por supuesto, en el alma de estas líneas. Contigo aprendo constantemente.

Mi más profundo agradecimiento a la Doctora G. Eréndira Núñez, por ser una de las mejores profesoras que he tenido, por su esfuerzo en hacer relucir lo mejor de nosotros, por su crítica certera, por las conversaciones, porque a veces simplemente escuchó, por su preocupación que iba desde lo académico a lo cotidiano, y por ende de lo intangible a lo concreto. Son tantas las cosas que pienso en este momento por agradecer, pero las quiero sintetizar en su cariño. Sus ideas, siempre caracterizadas por su formalidad y calidez, han sido la clave del buen trabajo que hemos realizado juntos, el cual no se puede concebir sin su siempre oportuna participación. Más allá de que hoy esté a un paso más cerca de poder ser su colega, Usted seguirá siendo mi maestra.

Quiero expresar también mi más sincero agradecimiento al Dr. J. Carlos Cortés quien me apoyó durante el desarrollo de este trabajo, cultivando en mí grandes conocimientos tanto en aspectos didácticos como en la lógica de programación. Gracias por darme un amplio margen de libertad en el proceso de investigación y redacción.

De forma particular, agradezco al M.C. Cuauhtémoc Rivera quien, además de brindarme sus conocimientos y experiencias profesionales desde el primer momento, me ha permitido considerarlo un amigo con el que siempre se puede contar. Con Usted hemos aprendido, no sólo lo escrito en los libros, sino el modo en que con una sonrisa todo es más fácil. Su apoyo y confianza en mi trabajo, y su capacidad para guiar mis ideas ha sido un aporte invaluable no solamente en mi formación profesional, sino también en lo personal. Siempre tendremos en mente lo que nos ha enseñado. Muchas gracias Profesor, espero que en el futuro la vida nos dé otras oportunidades de compartir experiencias y trabajos.

De igual manera agradecer al M.C. Christian Morales por su visión crítica de muchos aspectos cotidianos de la vida, por su rectitud en su profesión como docente, por sus consejos, que ayudaron a formarme como persona y profesional. Sus comentarios, apreciaciones y críticas hicieron que nuestros trabajos resultaran ser mil veces mejor que lo que se proyectaban originalmente.

Para todos mis amigos que han compartido conmigo el "ir y venir" en el plano personal durante este largo camino, tengo sólo palabras de agradecimiento, especialmente por aquellos momentos en los que pude ser inferior a sus expectativas. Ha sido un camino largo y duro en el que, algunas veces, la fijación por lograr nuestros objetivos nos hizo olvidar la importancia del contacto humano. Sin embargo, como en todas las actividades de la vida, siempre al final hay algunos criterios que te permiten priorizar y es por ello que debo resaltar mis agradecimientos para todos ustedes.

Y, por supuesto, el agradecimiento más profundo y sentido va para mi familia. Sin su apoyo, colaboración e inspiración habría sido imposible llevar a cabo este viaje. Por su ejemplo de lucha, honestidad, paciencia, inteligencia, constancia y generosidad, muchas gracias.

Un agradecimiento especial al Programa de Mejoramiento del Profesorado (PROMEP) por el apoyo otorgado durante la realización de este trabajo.

Índice general

Índice de figuras

Índice de códigos

Prefacio

Es indudable que la problemática en la enseñanza y el aprendizaje de las matemáticas, en especial del cálculo diferencial e integral, ha interesado a la mayoría de los investigadores a lo largo del tiempo, debido entre otras cosas, a la importancia que dicha problemática tiene para el desarrollo científico y tecnológico de los diferentes países y, paradójicamente, a los dramáticos resultados que regularmente en la formación integral de la educación ha sido reconocido el papel de las matemáticas en la formación integral de los individuos, ya que desarrolla competencias intelectuales útiles en los más diversos ambientes de la vida cotidiana, profesional y social.

En la presente tesis se expone un acercamiento a los conceptos del cálculo a través del diseño y desarrollo de un software denominado "Fun-Der", sobre el cual se plantean ejercicios didácticos de distintos niveles de complejidad para introducir el concepto de derivada e integral, mediante el manejo de tablas y gráficas, tomando como base teórica las ideas de R. Duval sobre registros semióticos de representaciones. Se pretende crear un espacio de reflexión y estudio sobre los conceptos del cálculo en cuanto objeto de enseñanza y aprendizaje. Por tanto, este trabajo de investigación es un punto de partida, que deja abierta la posibilidad de seguir estudiando los fenómenos educacionales, en cuanto se refiere a formas de enseñanza que ayuden al usuario a ser cada vez más independiente, crítico y analítico.

Capítulo 1

Introducción

1.1. Antecedentes

El presente trabajo forma parte de un proyecto de investigación que se está desarrollando en la Facultad de Ciencias Físico Matemáticas de la Universidad Michoacana de San Nicolás de Hidalgo. El proyecto global consiste en el diseño, desarrollo y evaluación, tanto técnico como educativo, de software de apoyo para el aprendizaje de las matemáticas. El proyecto incorpora el uso de las Tecnologías de la Información y la Comunicación (TIC) a la enseñanza de las matemáticas en las escuelas, por lo que una parte muy importante del proyecto global es incorporar al aula los prototipos informáticos desarrollados. El trabajo de tesis que se propone, consiste en realizar la programación del software llamado *"Fun-Der"* que forma parte de uno de los sistemas informáticos desarrollados en el proyecto global y que tiene que ver con los temas de cálculo diferencial e integral.

El uso de las TIC en las escuelas ya tiene una historia de más de una década, considerando los intentos y los experimentos llevados a cabo en países pioneros en este campo. Los resultados más relevantes reportados, coinciden en que los alumnos experimentan un aprendizaje significativo a través de un uso apropiado de las TIC, (Dunham y Dick, 1994; Borres-van Oosterum, 1990; Rojano, 1996); así como los profesores con poca experiencia en el uso de las TIC, tienen gran dificultad en apreciar su poder como herramienta de aprendizaje, y como consecuencia de lo anterior, que de no atenderse la carencia de conocimiento tecnológico de los docentes, las TIC no

tendrán una influencia importante en la cultura del aula (McFarlane, 2001).

En la actualidad se reconocen internacionalmente tres concepciones diferentes de las TIC:

1. Las TIC como un conjunto de habilidades o competencias.

2. Las TIC como un conjunto de herramientas, o de medios, de hacer la misma tarea, pero de un modo más eficiente.

3. Las TIC como un agente de cambio de un impacto revolucionario (McFarlane *et al.*, 2000).

La primer concepción propone a las TIC como materia de enseñanza, lo cual conduce a logros en el nivel de las competencias informáticas mismas, garantizando que dichos logros se reflejen automáticamente en áreas curriculares como las matemáticas.

Por otra parte, la segunda concepción se enfatiza en la relación que existe entre las TIC y el currículo, la cual consiste en agregar elmentos de tecnología informática a las tareas de aprendizaje para mejorar los objetos que se plantean en el currículo vigente. Dicho en otras palabras, estos modelos anticipan el efecto de las TIC en el logro de objetivos, tal y como lo proveen los sistemas de evaluación estandarizados. Esto último ha sido muy cuestionado por los especialistas en el aprendizaje mediado por las TIC, los cuales se basan en el aprendizaje situado (Lave, 1988; Rogoff y Lave, 1984; Wertsch, 1991); y cuyas consideraciones conducen a concluir que el aprendizaje que se lleva a cabo bajo entornos de tecnología no siempre se transfiere de manera espontánea a otros tipos de entornos (por ejemplo, lápiz y papel), de modo que aunque existan coincidencias en una variedad de estudios en lo que el uso de la TIC promueve el aprendizaje colectivo y mejora la capacidad de los alumnos para plantear preguntas y tomar decisiones apropiadas, sus logros no se ven reflejados en las calificaciones finales de los estudiantes. De ahí que los intentos de balance de las TIC sobre los objetivos educativos ha sido, en términos generales y en el mejor de los casos, más o menos favorable. Destacando que ésta concepción ha recibido severas críticas por el hecho de centrarse en el estudiante como usuario de la tecnología, sin dar la debida importancia al papel del maestro.

Finalmente, la tercera concepción considera a las TIC como agentes de cambio y con una gran potencialidad de revolucionar las prácticas en el aula, ésta hoy en día es muy difundida en los medios académicos (comunidad de especialistas e investigadores del uso de las TIC en educación; Crook, 1994); sin embargo, es difícil encontrar ejemplos de su implementación en el sistema educativo.

Esta iniciativa intenta mostrar que es factible aprovechar las nuevas tecnologías (apoyadas en un método pedagógico que permita construir ambientes de aprendizaje apropiados) para enriquecer y mejorar la enseñanza actual de las matemáticas, en este caso particular sobre los temas de cálculo diferencial e integral.

Entre las características principales del modelo que se propone el proyecto general, se encuentra la selección de herramientas, para lo cual se hicieron las siguientes consideraciones:

1. La utilización de software educativo como una herramienta que hace posible dar un tratamiento fenomenológico a los conceptos matemáticos; es decir, con dichas herramientas se puede concretar la idea de que los conceptos son organizadores de fenómenos. Así, la contextualización de las actividades matemáticas no es una mera ambientación, sino que las situaciones planteadas por la actividad corresponden a comportamientos de fenómenos, que en cierto modo, forman parte de la esencia del concepto que se busca enseñar.

2. La utilización de software y herramientas que impliquen representaciones ejecutables, es decir, que contemplen la manipulación directa de objetos o de representaciones de objetos matemáticos.

3. La utilización de software y herramientas cuyo uso está relacionado con un área específica de la matemática escolar (aritmética, álgebra, geometría, geometría analítica, cálculo, probabilidad, modelación, etc.).

4. La puesta en marcha de un modelo de colaboración para el aprendizaje: los estudiantes trabajarán en parejas frente a la computadora en una misma actividad, lo que promoverá la discusión y el intercambio de ideas.

5. La práctica de un modelo pedagógico en el que el profesor promueve el intercambio de ideas y la discusión en grupo, y al mismo tiempo actúa como

mediador entre el estudiante y la herramienta, es decir, el ambiente computacional, asistiendo a los estudiantes en su trabajo con las actividades de clase y compartiendo con ellos el mismo medio de expresión.

A continuación se dará una breve introducción a los capítulos que integrarán el trabajo de tesis que se propone.

En el Capítulo 1 se hará referencia a los antecedentes del proyecto, la justificación, los objetivos del trabajo que aquí se propone y se plantean las preguntas de investigación en las que se basó este trabajo.

En el Capítulo 2 se establecerán las bases teóricas que sustentan esta tesis, como son el uso de nuevas tecnologías en la enseñanza de las matemáticas en ambientes de aprendizaje, la teoría de registros de representación de Duval (1993 y 1996) y el aprendizaje colaborativo.

En el Capítulo 3 se hablará acerca de las actividades planteadas en el software educativo, además se analizará el tiempo requerido para la elaboración de las actividades, el material didáctico que se requiere, el papel del maestro en este tipo de trabajo, así como algunos comentarios y observaciones del mismo.

Finalmente, en el Capítulo 4 se escribirán las conclusiones, observaciones, comentarios y sugerencias que se tengan como resultado de realizar el presente trabajo.

Por lo que se refiere a la educación matemática, ésta debe reflexionar sobre la influencia de la tecnología computacional, tanto en las matemáticas como en la sociedad. Esta tecnología computacional requiere una nueva interpretación del proceso de enseñanza-aprendizaje, por lo que la extensa naturaleza de los recursos tecnológicos ahora disponibles pone de manifiesto los cambios significativos en el ámbito escolar y pone al alcance de los estudiantes una extensa disponibilidad tanto de software como de hardware.

Una apropiada tecnología computacional lleva al usuario a que tome el control de su propio aprendizaje. Lo cual fomenta tanto el aprendizaje colaborativo como el independiente, mientras extiende y soporta el proceso de aprendizaje.

1.2. Planteamiento del Problema

Como bien es sabido, el cálculo es una herramienta muy poderosa en la resolución de problemas matemáticos complicados, por eso el dominio de sus conceptos básicos se hace esencial para que los estudiantes tengan éxito a la hora de aplicarlo, entre los conceptos que los estudiantes deben dominar y uno de los más difíciles, es el de derivada.

Por tanto, el software educativo de apoyo para los cursos de cálculo diferencial pretende mostrar una representación tanto gráfica como numérica de la razón de cambio, ya que muchos autores remarcan como gran importancia introducir el concepto de derivada mediante el uso de razones de cambio, para con ello comenzar a abordar los conceptos de derivada.

Se pretende que el alumno desarrolle su capacidad creativa y analítica mediante los procesos de visualización sobre esta temática, ya que la visualización en matemáticas es el proceso de formación de imágenes y el uso de tales imágenes en forma efectiva, ayuda para el descubrimiento matemático y su entendimiento. En consecuencia, favorece la comprensión de los mismos.

Este proyecto intenta mostrar que es factible aprovechar las nuevas tecnologías apoyadas en un método pedagógico que permita construir ambientes de aprendizaje apropiados para enriquecer y mejorar la enseñanza actual de las matemáticas, en este caso particular sobre el tema de funciones y derivadas.

Traduciéndose en modelos específicos para la enseñanza de las matemáticas, bajo los siguientes principios:

Didáctico: mediante el cual se diseñan actividades para el aula, siguiendo un tratamiento fenomenológico de los conceptos que se enseñan.

De especialización: por el cual se seleccionan herramientas y piezas de software de contenido. Los criterios de selección se derivan de didácticas específicas acordes con las matemáticas.

Cognoscitivo: por cuyo conducto se seleccionan herramientas que permiten la manipulación directa de objetos matemáticos y de modelos de fenómenos mediante representaciones ejecutables.

Empírico: bajo el cual se seleccionan herramientas que han sido probadas en algún sistema educativo.

Pedagógico: por cuyo intermedio se diseñan actividades de uso de las TIC, para que promuevan el aprendizaje colaborativo y de interacción entre los alumnos, así como entre profesores y alumnos.

De equidad: con el que se seleccionan herramientas que permiten a los alumnos de bachillerato el acceso temprano a ideas importantes en matemáticas.

Con todo lo anterior este tipo de trabajos siempre beneficiará en gran medida tanto a profesores, estudiantes, así como al grupo de diseñadores y desarrolladores de software educativo.

1.3. Hipótesis

"Los conceptos involucrados en el aprendizaje del cálculo diferencial e integral serán mejor entendidos por el estudiante si se le presentan actividades de aprendizaje en forma interactiva utilizando software especializado, como lo es *Fun-Der*"

Ya que es un hecho que con las nuevas tecnologías el alumno se motiva más, si se le presenta un método diferente y divertido, que se salga de lo que es la enseñanza tradicional, lo lleve poco a poco y a su ritmo, al entendimiento de los diferentes conceptos vistos en un curso de cálculo.

Ayudándonos con el diseño de actividades con las que se pretende que el alumno desarrolle su capacidad creativa y analítica, mismas que se pueden llevar a cabo en tiempo extra clases. Ahora bien, el software "FunDer" intenta apoyar los procesos de visualización sobre esta temática ya que como señalan Zimmerman y Cunningham (1990, p.3)

> "Visualizar un diagrama significa simplemente formar una imagen mental del mismo, pero visualizar un problema significa entenderlo en términos de un diagrama o de una imagen visual. La visualización en matemáticas es el proceso de formación de imágenes (mentales, con lápiz y papel, o con ayuda de la tecnología) y el uso de tales imágenes en forma efectiva ayuda para el descubrimiento matemático y el entendimiento".

Por lo que creemos que la utilización de la computadora ayuda a la visualización de conceptos matemáticos y éstos a su vez, favorecen la compresión. David Tall (1991) asegura que: *"La visualización matemática involucra la intuición que es el primer acercamiento a un nuevo conocimiento"*, menciona también que *"negar los métodos visuales es negar las raíces de una gran cantidad de ideas matemáticas profundas"*.

1.4. Justificación

La computadora como una de las tecnologías más avanzadas e integral, misma que ha venido revolucionando los procesos de aprendizaje casi en todos sus sentidos, ofrece hoy por hoy nuevas herramientas para poder explicar conceptos, en especial en la enseñanza y el aprendizaje del cálculo, por su capacidad de utilizar distintas representaciones tales como fórmulas algebraicas, gráficos, tablas de valores o geometría dinámica; permitirá facilitar la interacción del estudiante con el tema por aprender, familiarizando más al alumno con el proceso educativo.

Especialmente, en el campo de las matemáticas, los software representan un gran apoyo en la enseñanza, ayudando no solamente en agilizar el tiempo empleado en la resolución de algún determinado problema, sino también a potenciar el desarrollo cognitivo del estudiante, es por ello que en los últimos años se le ha dado gran importancia a los Sistemas Computacionales de Álgebra (CAS), con programas como Derive, Maple o Mathematica, éstos combinan las fórmulas algebraicas, gráficos y tablas, permitiendo además el paso inmediato de una representación a otra. (Cortés, 2005)

Comprobando que el cálculo es uno de los cursos más problemáticos para los estudiantes, muchos lo reprueban o abandonan. Una de las razones por las cuales se puede explicar este hecho, es que a los estudiantes les cuesta mucho entender los conceptos básicos que ahí se enseñan, ya que todas las definiciones utilizan procesos al infinito, que son difíciles de comprender en un inicio, por esto se ha visualizado que la función del software "Fun-Der", fortalecerá el desarrollo del pensamiento matemático a partir de un acercamiento a la razón de cambio generando una función, la función derivada y graficarlas mostrando sus diferencias. Por sus características será así mismo de gran utilidad para los estudiantes de áreas afines, debido a que

el manejo gráfico y numérico del concepto de razón de cambio logra un importante paso al entendimiento de la derivada.

Por otra parte, como redacta Cortés (2005) en su libro de reflexiones sobre el aprendizaje del cálculo y su enseñanza:

> *[. . .] argumentaba Mejía* "en cuanto al manejo de tablas, hay muy pocos trabajos respecto a su tratamiento"*, por lo que consideramos importante promover este tipo de acercamiento, ya que de acuerdo a Confrey* "la representación algebraica de funciones obscurece la conexión funcional numérica en un grado importante. Las tablas pueden ayudar a iluminar esta conexión"*.*

Muchas veces surgen interrogantes en el proceso enseñanza-aprendizaje de las matemáticas, por ejemplo: "una mayoría de profesores de matemáticas enfatizaban demasiado las actividades en un sólo sistema de representación: el algebraico. Por ello, son relevantes las representaciones propuestas por Duval, donde las tareas de tratamiento en una misma representación y de conversión entre representaciones de conceptos matemáticos sean la parte esencial" (Cortés, 2005).

Otro aspecto importante es desarrollar en el estudiante la habilidad de visualización matemática, que puede ayudarlo en la resolución de problemas, esta tiene que ver con el entendimiento de un enunciado y la puesta en marcha de una actividad que si bien no llevará a la respuesta correcta, sí puede llevar al estudiante a la profundización de la situación que esté tratando.

Siendo este el motivo por el cual, cada vez se recurre más frecuentemente a las herramientas de la computación, consecuencia de que el software educativo se ha convertido en un importante apoyo a los métodos de aprendizaje. Finalmente, el software educativo, por su ahorro de tiempo en la resolución de problemas, ofrece grandes facilidades y ventajas en el campo de la educación, ya que al tener acceso a este tipo de tecnologías, también se tiene la oportunidad de adquirir una rápida formación intelectual.

1.5. Objetivos

La tesis consistirá en la creación de un software en lenguaje de programación Java que se pretende usar como apoyo educativo para temas de cálculo diferencial e integral en el bachillerato.

Partiremos del diseño desarrollado por Cortés (2005) en el software Fun-Der realizado en Visual Basic; se diseñarán en Java tantos los aspectos de navegación en el software, así como los aspectos didácticos de cada una de las actividades que compondrán el software.

Desarrollar actividades de aprendizaje para los temas de derivación e integración.

Diseñar y desarrollar todos los "métodos" (programación de los archivos `.class`) necesarios para que el software cumpla con las actividades necesarias.

Realizar pruebas técnicas del software programado.

En concreto, se pretende que Java pueda derivar e integrar funciones polinomiales y trigonométricas; se implementarán métodos numéricos, técnicas de interpolación numérica, polinomial y trigonométrica, y nuevas funciones para graficar funciones polinomiales y trigonométricas.

Capítulo 2

Marco Teórico

2.1. Desarrollo de Software

En esta era tecnológica, muchos aspectos de nuestras vidas están regidos por las computadoras y el software que las controla. Las consecuencias generadas por el uso de software es algo más que la programación; hay etapas que la precede y otras que la siguen. (Weitzenfeld, 2005)

La ingeniería de software permite al diseñador de programas, realizar su tarea de construcción de software como un problema de ingeniería haciendo uso de guías, principios y normas que le permitirán el correcto desarrollo de su labor. Adicionalmente, dispondrá de un conjunto de herramientas que le permitirán la evaluación, validación, depuración y corrección del software desarrollado. (Braude, 2007)

La técnica utilizada por los desarrolladores profesionales de software, es comprender lo mejor posible el problema que se está tratando de resolver, y crear una solución de software apropiada y eficiente que se denomina proceso de desarrollo de software.

El desarrollo de un buen sistema de software se realiza durante el ciclo de vida, que es el período de tiempo que se extiende desde la concepción inicial del sistema, hasta su eventual retirada a la comercialización del mismo. Las actividades humanas relacionadas con el ciclo de vida del software, implican procesos tales como análisis de requisitos, diseño, implementación, codificación, pruebas, verificación, documentación, mantenimiento y evolución del sistema y obsolescencia. (Aguilar, 2003)

Los métodos y técnicas de la ingeniería del software se inscriben dentro del marco delimitado por el ciclo de vida del software y, más concretamente, por las diferentes etapas que se distinguen. La misma existencia de distintos modelos del ciclo de vida del software hace comprender que no hay ninguno que sea ideal, o que no tenga grandes limitaciones.

Sin embargo, es indispensable que todo proyecto se desarrolle dentro del marco de un ciclo de vida claramente definido, si se quiere tener una mínima garantía de cumplimiento de los plazos, y respetar los límites de los recursos asignados. Además la garantía de calidad y certificaciones de calidad, también presuponen que el proceso de producción del software se desarrolle según un ciclo de vida con etapas bien definidas (Braude, 2007).

2.1.1. Análisis

Esta etapa también se denomina "análisis de sistemas" o "ingeniería de sistema", en esta etapa se definen los grandes rasgos del sistema de software que tendrá que dar soporte informático a unas actividades determinadas de unos ciertos usuarios dentro del marco más general a la actividad de la empresa u organización. (Sommerville, 2005)

La parte más difícil en la tarea de crear un software, es definir cuál es el problema y después especificar lo que se necesita para resolverlo. Normalmente la definición del problema comienza analizando los requisitos de usuario, pero estos requisitos con frecuencia, suelen ser imprecisos y difíciles de describir. Se deben especificar todos los aspectos del problema, pero en ocasiones las personas que describen el problema no son programadores y eso hace imprecisa la definición. La fase de especificación, requiere normalmente la comunicación entre los programadores y los futuros usuarios del sistema e iterar cualquier detalle hasta que el programador y los usuarios estén satisfechos y hayan resuelto el problema. (Alonso, 2005)

El usuario final, normalmente no conoce con exactitud lo que desea que haga el sistema, por consiguiente el analista de software o programador, en su caso, debe interactuar con el usuario para encontrar lo que él quiere que haga el sistema. El resultado final de la fase de análisis, es un documento de especificación de requisitos del software que describe explícitamente la funcionalidad, es decir explica con

precisión lo que hace el producto y numera cualquier restricción que deba cumplir el producto. La fase de especificaciones tiene dos salidas principales: la primera es el documento de especificaciones y la segunda es un plan de gestión del proyecto de software. (Aguilar, 2003)

2.1.2. Diseño e implementación

Arrancando de las especificaciones, el equipo de diseño determina la estructura interna del producto. Los diseñadores descomponen el producto en módulos, piezas independientes de código con interfaces bien definidas en el resto del producto. Una vez que el equipo ha completado la descomposición en módulos (diseño arquitectónico) se realiza el diseño detallado, para cada módulo se seleccionan los algoritmos y estructuras de datos elegidas. Es preciso determinar si se pueden utilizar programas o subprogramas que ya existen, o si es preciso construirlos totalmente. (Aguilar, 2003)

Del software hay que diseñar varios aspectos diferenciados; su arquitectura general, las estructuras de datos, la especificación de cada programa, se construye el algoritmo que se utilizará para resolver el problema y las interfaces con el usuario; y se tiene que llevar a cabo de manera que a partir de todo esto, se pueda codificar el software, de forma parecida a la construcción de un edificio o de una máquina a partir de unos planos. (Braude, 2007)

En este punto es importante especificar claramente no sólo el propósito de cada módulo, sino también el flujo de los datos entre módulos. El diseño es una actividad de sólo ingeniería, su entrada principal en la especificación que se traducirá en un documento de diseño, escritos por los ingenieros de software. El documento resultante es la especificación del diseño, la etapa de diseño es el mejor momento para elaborar la especificación de la prueba que describe con qué datos se tiene que probar cada programa, o grupo de programas, y cuáles son los resultados esperados en cada caso. (Aguilar, 2003)

En la etapa de codificación, partiendo del análisis y diseño de la solución, se procede a desarrollar el correspondiente programa que solucione el problema mediante el uso de una herramienta computacional determinada, es decir, esta etapa traduce los algoritmos del diseño en un programa escrito con un lenguaje de programación. (Sommerville, 2005)

2.1.3. Pruebas

El desarrollo de sistemas de software, implica una serie de actividades de producción en las que las posibilidades de que aparezca el fallo humano son enormes. Los errores pueden empezar a darse desde el inicio del proceso, en el que los objetivos pueden estar especificados de forma errónea o imperfecta, así como dentro de etapas de diseño y desarrollo posteriores.

Debido a la imposibilidad humana de trabajar y comunicarse de forma perfecta, el desarrollo de software debe de ir acompañado de una actividad que garantice la calidad. De este modo, la prueba de software se posiciona como un elemento crítico para la garantía de calidad del software y representa una revisión final de la codificación.

Los objetivos principales de una prueba de software son:

- Detectar un error.
- Tener un buen caso de prueba, es decir, que tenga una probabilidad mayor de mostrar un error no descubierto antes.
- Descubrir un error no descubierto antes (lo que llevará al éxito de la prueba).

Estos objetivos sistemáticamente nos guiarán a descubrir diferentes tipos de errores con menor tiempo y esfuerzo, sumando que todas las pruebas se realizan a nivel de unidad y no al sistema en forma global.

Tipos de pruebas:

Pruebas de regresión. Su objetivo es verificar al sistema después de un cambio introducido, en la especificación original del sistema.

Prueba de operación. Su objetivo es verificar el sistema de operación en un periodo de tiempo largo bajo condiciones normales de uso, nos permite medir la confiabilidad del sistema.

Prueba de escala completa (stressing). Trata de Verificar el sistema en su carga máxima mediante la asignación de parámetros en su valor límite.

Prueba de rendimiento o capacidad. Tiene como propósito medir la cantidad de procesamiento bajo diferentes cargas, incluyendo el espacio de almacenamiento y la utilización del CPU, la cantidad de memoria y verificar que los valores sean los requeridos.

Prueba de sobrecarga. Se emplea esta prueba para saber en qué punto colapsa el sistema. (Braude, 2007)

2.1.4. Manteniento

Una vez instalado un programa y puesto en marcha para realizar la solución del problema previamente planteado o satisfacer una determinada necesidad, es importante mantener una estructura de actualización, verificación y validación que permitan a dicho programa ser útil y mantenerse actualizado según las necesidades o requerimientos planteados durante su vida útil. (Sommerville, 2005)

2.1.5. Principios de diseño de software

El diseño de sistemas de software de calidad requiere el cumplimiento de una serie de características y objetivos. En un sentido general, los objetivos a conseguir que se consideren útiles en el diseño de sistemas incluyen al menos los siguientes principios: (Aguilar, 2003)

a) Modularidad.

b) Abstracción y ocultamiento de la información.

c) Modificabilidad.

d) Comprensibilidad y fiabilidad.

e) Interfaces de usuario.

f) Facilidad de uso.

g) Eficiencia.

h) Estilo de programación.

i) Depuración.

j) Documentación.

2.1.6. Estilos de programación

Una de las características más importante en la construcción de programas, sobre todo de gran tamaño, es el estilo de programación. La buena calidad en la producción de programas tiene relación directa con la escritura de un programa, su legibilidad y comprensibilidad. Un buen estilo de programación suele venir con la práctica, pero el requerimiento de unas reglas de escritura del programa, al igual que sucede con la sintaxis y reglas de escritura de un lenguaje natural humano, debe de buscar que no sólo sea legible y modificable por la persona que lo ha construido, sino también y esencialmente puedan ser leídos y modificados por otras personas distintas.

Normalmente las reglas de estilos para construir estilos claros, legibles y fácilmente modificables dependerán del tipo de programación y lenguaje elegido. Entre las más generales tenemos: modularizar un programa en partes, usar nombres significativos para identificadores, definir constantes con nombre al principio del programa, presentación (comentarios adecuados), manejo de erores (`try-catch` para el caso de lenguaje java), legibilidad y documentación. (Aguilar, 2003)

2.1.7. Documentación

Un programa (paquete de software), necesita siempre una documentación que permita a sus usuarios aprender a utilizarlo y mantenerlo. La documentación es una parte importante de cualquier paquete de software y a su vez, su desarrollo es una pieza clave en la ingeniería de software. (Aguilar, 2003)

La importancia de la documentación, radica en que a menudo un programa escrito por una persona, sea modificado por otra. Por ello la documentación sirve para ayudar a comprender o usar un programa, o para facilitar futuras modificaciones (mantenimiento).

La documentación se compone de tres partes:

Documentación Interna. Son los comentarios o mensajes que se añaden al código

fuente para hacer más claro el entendimiento de los procesos que lo conforma, incluyendo las precondiciones y las poscondiciones de cada función.

Documentación Externa. Se define en un documento escrito con los siguientes puntos: Descripción del problema, Algoritmo (diagrama de flujo o pseudocódigo), Diccionario de Datos, y Código Fuente (programa).

Manual de Usuario. Describe paso a paso la manera de cómo funciona el programa, con el fin de que el usuario lo pueda manejar para que obtenga el resultado deseado.

Una buena documentación de usuario hará al programa más accesible y asequible (Braude, 2007).

2.2. Diseño y Arquitectura de Software

2.2.1. Implementación de arquitectura de software

Como bien es sabido, los software educativos no pueden tener una plantilla de arquitectura que marque cómo estructurar las clases, por ser de carácter educativo poseen sus propios modelos arquitectónicos con base en sus necesidades, sin embargo siguiendo las normas de ingeniería de software se tomó como guía la arquitectura "modelo-vista-controlador", sobre la cual se construyó la aplicación, favoreciendo a reunir los criterios de modelado y facilitar la comunicación entre los diseños, la cual permita su manejo así como cambios futuros (figura 2.1).

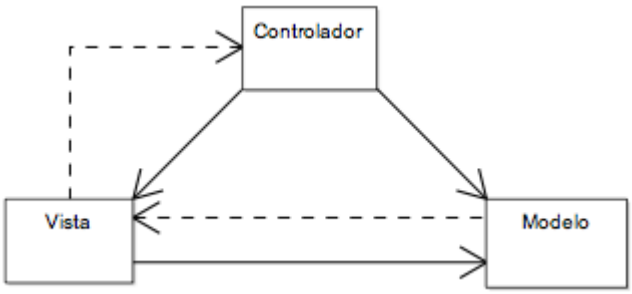

Figura 2.1: Diagrama que muestra la relación entre el modelo, la vista y el controlador. Las líneas sólidas indican una asociación directa, y las punteadas una indirecta.

2.2.2. Diagramas en la construcción del software

En el desarrollo de los esquemas a emplear, en los cuales se plasma la estructura, secuencia, estados y lógica de las funciones del software, entre ellos, diagramas de casos de uso, diagramas de secuencia, diagramas de estado, diagramas de clases y diagramas de flujo, lo que lleva al software a una sólida documentación de diseño.

2.3. Razón de Cambio

La razón de cambio (de una variable respecto de otra) es la magnitud del cambio de una variable por unidad de cambio de la otra. Si las variables no tienen ninguna dependencia la tasa de cambio es cero.

En general, es una relación funcional $y = f(x)$, la razón de cambio de la variable dependiente y respecto de la independiente x, se calcula mediante un proceso de límite de la razón $\frac{f(x+t)-f(x)}{t}$, denominada cociente diferencial.

En sentido estricto entonces, la razón de cambio es el límite del cociente diferencial cuando t tiende a cero. De esta manera, la razón de cambio es la interpretación fundamental de la derivada de una función.

Por ejemplo:

En la función lineal $f(x) = mx + b$, no es necesario tomar el límite pues:

$$f(x + t) - f(x) = mx + mt + b - mx - b = mt$$

y la t se cancela en la razón $\frac{f(x+t)-f(x)}{t}$ sin necesidad de pasar al límite. Resaltando que m es la pendiente de la recta $f(x) = mx + b$. Y es la razón de cambio de la altura y (variable dependiente) respecto a la x (variable independiente). Viéndose gráficamente, es el cambio de la altura y por unidad de cambio (aumento) en x.

Es importante, no sólo porque, al tomar el límite cuando t tiende a cero resulta la derivada de la función, sino también porque admite la siguiente interpretación fundamental para la comprensión de la derivada y para sus aplicaciones: El cociente diferencial es la pendiente de la secante (recta que corta a la gráfica de f) que pasa por los puntos $(x, f(x))$ y $(x + t, f(x + t))$.

Es debido a esta interpretación del cociente diferencial que se dice que: La de-

rivada es la pendiente de la tangente a la curva (la gráfica de $f(x)$) en el punto $(x, f(x))$.

Así mismo, t se interpreta como una variación de x y debe considerarse pequeña (puesto que al final "se irá a cero"). (Matemat, 2009)

2.4. Diferenciación Numérica

La derivada de una función f en x_0 es:

$$\lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}$$

Esta fórmula indica una manera obvia de generar una aproximación de $f'(x)$; basta calcular

$$\frac{f(x_0) + h) - f(x_0)}{h}$$

para valores pequeños de h. Aunque esto parezca evidente, no es muy útil, debido a nuestro error por redondeo. Sin embargo, ciertamente es un punto de partida. (Burden, 2002)

Por lo anterior se ha originado el desarrollo de diversas fórmulas de diferenciación numérica, como es:

La diferencia dividida central, que no es sino una de tantas que surgen en el desarrollo de la serie de Taylor para la aproximación de las derivadas numéricas:

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$$

Donde a h se le llama "tamaño de paso", esto es, la longitud del intervalo sobre el cual se hace la aproximación. El análisis de la serie de Taylor ha llevado a la información práctica de que la diferencia central es la representación más exacta de la derivada. (Suárez, 2003)

2.5. Recta Tangente y Recta Secante

La recta secante es una recta que corta a una circunferencia en dos puntos. Conforme estos puntos se acercan y su distancia se reduce a cero, la recta adquiere el nombre de recta tangente.

Esencialmente, el problema de hallar la recta tangente en un punto P se reduce al de hallar su pendiente en ese punto. Se puede aproximar la pendiente de la recta tangente usando la recta secante que pasa por P y otro punto cercano a la curva, como se indica en la figura (2.2)

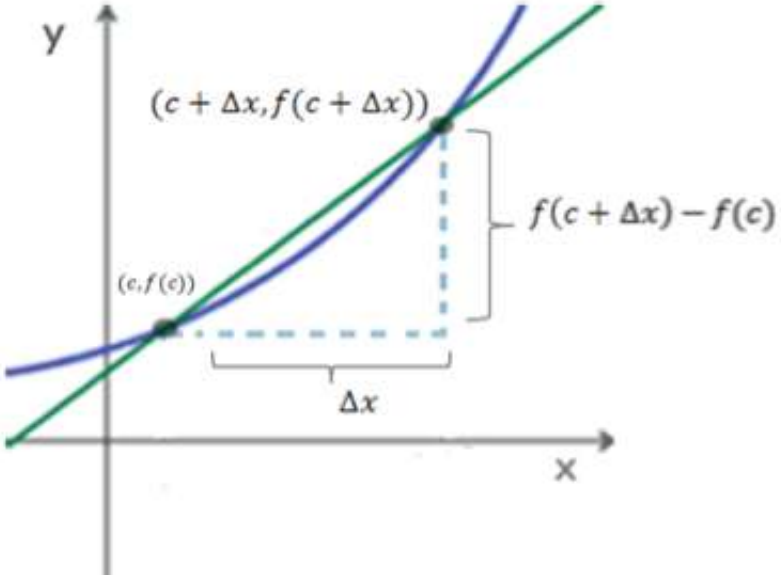

Figura 2.2: Recta secante

Si $(c,\ f(c))$ es el punto de tangencia y $(c+\Delta x, f(c+\Delta x))$ es el otro punto de la gráfica de f, la pendiente de la recta secante que pasa por esos dos puntos viene dada por:

$$\begin{aligned} m &= \frac{y_2 - y_1}{x_2 - x_1} \\ m_{sec} &= \frac{f(x+\Delta x) - f(c)}{(c+\Delta x) - c} \\ &= \frac{f(c+\Delta x) - f(c)}{\Delta x} \end{aligned}$$

El elemento de la derecha en esta ecuación es un cociente incremental (o de incrementos). El denominador Δx es el cambio (o incremento) en x y el numerador $\Delta y = f(c + \Delta x) - f(c)$ es el cambio o incremento en y.

Con este proceso, se pueden obtener aproximaciones más, y más precisas de la pendiente de la recta tangente tomando puntos de la gráfica cada vez más próximos al punto P de tangencia. Es decir, si f está definida en un intervalo abierto (a, c) y además existe el límite:

$$\lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(c + \Delta x) - f(c)}{\Delta x} = m$$

Entonces, la recta que pasa por $(c, f(c))$ con pendiente m se llama recta tangente a la gráfica de f en el punto $(c, f(c))$. (Larson, 1999)

2.6. Interpolación

En ciertos casos el usuario conoce el valor de una función $f(x)$ en una serie de puntos $x_1, x_2, \cdots, x_N$ pero no se conoce una expresión analítica de $f(x)$ que permita calcular el valor de una función para un punto arbitrario. Un ejemplo claro son las mediciones de laboratorio, donde se mide cada minuto un valor, pero se requiere el valor en otro punto que no ha sido medido. Otro ejemplo son mediciones de temperatura en la superficie de la Tierra, que se realizan en equipos o estaciones meteorológicas y se necesita calcular la temperatura en un punto cercano, pero distinto al punto de medida.

La idea de la interpolación es poder estimar $f(x)$ para un x arbitrario, a partir de la construcción de una *curva* o *superficie* que une los puntos donde se han realizado las mediciones y cuyo valor sí se conoce. Se asume que el punto arbitrario x se encuentra dentro de los límites de los puntos de medición, en caso contrario se llamaría *extrapolación*.

Existe un sinnúmero de métodos de interpolación, incluyendo la interpolación *lineal, polinomial, lagrange, splines* y la de *mínimios cuadrados*.

En muchos casos el usuario se enfrenta a funciones para las cuales la interpolación no funciona. Por ejemplo, la función:

$$f(x) = \frac{1}{3}x^2 + \frac{1}{\pi}\ln[(\pi - x)^2] + 1 \tag{2.1}$$

tiene una singularidad en $x = \pi$ (ver figura 2.3). Cualquier método de interpolación que se base en valores de $x = 3.14, 3.15, 3.16$ muy probablemente va a generar un resultado erroneo para $x = 3.1415\ldots$ (figura 2.3).

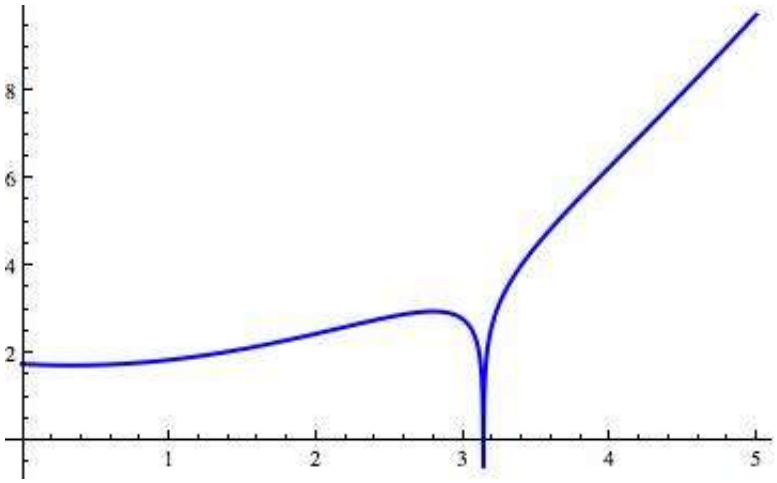

Figura 2.3: Los métodos de interpolación pueden tener problemas interpolando una función con singularidades.

En concreto el problema que se plantea es el siguiente: Interpolar una función $f : I \subseteq \mathbf{R} \to \mathbf{R}$ en un conjunto abierto D y en un conjunto de $n + 1$ puntos $\{x_0, x_1 \cdots, x_n\} \subset I$ es encontrar otra función Φ de manera que sobre estos puntos, la nueva función tome los mismos valores que la función original. Es decir, verificando

$$\Phi(x_i) = f(x_i) = f_i, \qquad i = 1, \cdots, n$$

Algunas técnicas de interpolación son:

- Interpolación polinómica: Φ es una función polinómica de x, es decir:

$$\Phi(x; \quad a_0, \ldots, a_n) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

- Interpolación racional: Φ es una función racional (cociente de polinomio) de x,

es decir:

$$\Phi(x; a_0, \ldots, a_n, b_0, \ldots, b_m) = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n}{b_0 + b_1 x + b_2 x^2 + \cdots + b_m x^m}$$

- Interpolación exponencial: Φ es una combinación lineal de exponenciales reales, es decir:

$$\Phi(x; a_0, \ldots, a_n, b_0, \ldots, b_n) = a_0 e^{b_0 x} + a_1 e^{b_1 x} + a_2 e^{b_2 x} + \cdots + a_n e^{b_n x}$$

- Interpolación trigonométrica: Φ es una combinación lineal de exponenciales imaginarias, es decir:

$$\Phi(x; a_0, \ldots, a_n) = a_0 + a_1 e^{ix} + a_2 e^{2ix} + \cdots + a_n e^{nix}$$

con $i = \sqrt{-1}$. Recordemos que por la fórmula de Euler se tiene que $e^{ix} = \cos(x) + i \sin(x)$ con $x \in \mathbf{R}$

2.6.1. Interpolación Polinomial

Con frecuencia se encontrará con que tiene que estimar valores intermedios entre datos definidos por puntos. El método más común que se usa para este propósito es la interpolación polinomial. La fórmula general para un polinomio de $n-$ésimo grado es:

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \tag{2.2}$$

Dados $n + 1$ puntos, hay uno y sólo un polinomio de grado n que pasa a través de todos los puntos[1]. Por ejemplo, hay sólo una línea recta (es decir, un polinomio de primer grado) que une dos puntos; de manera similar, únicamente una parábola une un conjunto de tres puntos: (figura 2.4).

[1] De hecho se puede probar que dados $n + 1$ puntos, con abscisas distintas entre sí, existe uno y sólo un polinomio de grado a lo más n que pasa por estos puntos

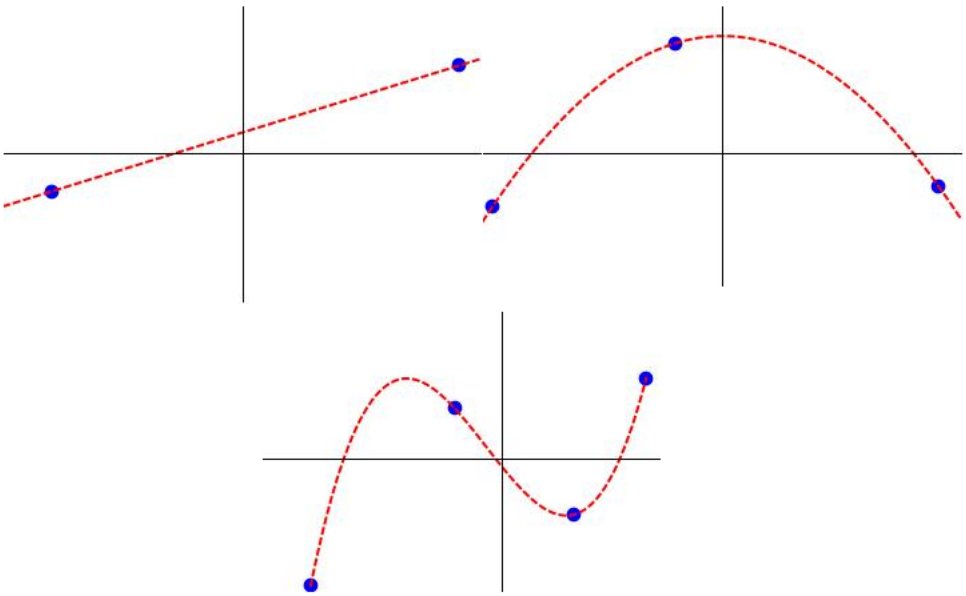

Figura 2.4: Ejemplos de interpolación polinomial. De primer grado (lineal) que une dos puntos; de segundo grado (cuadrática o parábola) que une tres puntos; y de tercer grado (cúbica) que une cuatro puntos.

La *interpolación polinomial* consiste en determinar el polinomio único de $n-$ésimo grado que se ajusta a $n+1$ puntos. Este polinomio, entonces, proporciona una fórmula para calcular valores intermedios.

Aunque hay uno y sólo un polinomio de $n-$ésimo grado que se ajusta a $n+1$ puntos, existe una gran variedad de formas matemáticas en las cuales puede expresarse este polinomio.

2.6.2. Aproximación por Mínimos Cuadrados

En el caso de la interpolación, la condición impuesta al polinomio de interpolación de ser igual a la función en los puntos del soporte ($P_n(x_k) = f(x_k)$), requiere implícitamente que el valor $f(x_k)$ sea exacto. Sin embargo, en situaciones reales de valores $f(x_k)$ obtenidos experimentalmente, esta situación nunca es exacta debido a múltiples factores. En tales condiciones, no tiene mucho sentido exigir que $P_n(x_k) = f(x_k)$. Se hace pues necesario introducir una técnica de aproximación que no requiera de estas restricciones.

Definamos V un espacio funcional en el cual se puede definir el producto escalar (sólo caso discreto para fines de esta tesis):

$$\langle f, g\rangle = \sum_s f(x_k)g(x_k)dx, \quad x_k \in S$$

Entonces una norma sobre V se construye a partir de:

$$||f||_v = \sqrt{\sum_s f^2(x)}dx$$

Con el producto escalar y la norma así definida, se dice que V constituye un espacio de Hilbert.

Consideremos ahora una base formada por las $m+1$ funciones $b_0(x), b_1(x), \ldots, b_m(x)$, con las cuales se quiere construir el polinomio de aproximación:

$$P_m(x) = c_0b_0(x) + c_1b_1(x) + \cdots + c_mb_m(x)$$

Entonces el error cometido al aproximar f por P_m es:

$$E(c_k) = f(x) - P_m(x)$$

Es obvio que para cada polinomio P_m, el error dependerá de los coeficientes c_k que son quienes controlan la calidad de la aproximación y la cual se trata de que sea óptima. Esta optimización equivale a pedir que el error sea mínimo y esto, en un espacio de Hilbert, significa que el tamaño de $E(c_k)$ sea lo más pequeño posible.

$f(x) \approx P_m(x)$ es óptima si $||E||_v$ es mínimo. Pero $||E||_v = \langle E(c_k), E(c_k)\rangle^{\frac{1}{2}}$

Para valores inferiores a 1, el cuadrado es más pequeño que el número:

$$\langle E(c_k), E(c_k)\rangle < \langle E(c_k), E(c_k)\rangle^{\frac{1}{2}}$$

es mejor entonces minimizar $||E||_v^2$, para lo cual es suficiente que:

$$\frac{\partial}{\partial c_k}||E||_v^2 = 0 \tag{2.3}$$

Por otra parte $||E||_v^2 = ||f - P_m||_v^2 = \langle f(x) - P_m(x), f(x) - P_m(x)\rangle$ y entonces:

$$\frac{\partial}{\partial C_k}||E||_v^2 = \frac{\partial}{\partial c_k}\int_v [f(x) - P_m(x)]^2\, dx = -2\int [f(x) - P_m(x)] \cdot b_x(x) dx$$

La condición (2.3) es equivalente a:

$$-\frac{1}{2}\frac{\partial}{\partial c_k}||E||_v^2 = \langle f(x) - P_m(x), b_k(x)\rangle = 0 \tag{2.4}$$

Pero como $P_m(x) = \sum_{k=0}^{m} c_k b_k(x)$, entonces (2.4) se escribe:

$$\begin{aligned}\left\langle f(x) - \sum_{k=0}^{m}(c_k b_k(x), b_k(x)\right\rangle &= \langle f(x) - c_0 b_0(x), b_k(x)\rangle + \cdots + \langle f(x) - c_m b_m(x), b_k(x)\rangle \\ &= \langle f(x), b_k(x)\rangle - c_0\langle b_0(x), b_k(x)\rangle - \cdots - c_m\langle b_m(x), b_k(x)\rangle \\ &= 0\end{aligned}$$

Finalmente:

$$c_0\langle b_k, b_0\rangle + c_1\langle b_k, b_1\rangle + K + c_m\langle b_k, b_m\rangle = \langle f, b\rangle$$

Como esto debe cumplirse para toda $b_k(x)$ para $k = 0, \cdots, m$; entonces lo anterior equivale al sistema lineal:

$$\begin{pmatrix} b_{00} & b_{01} & b_{02} & \cdots & b_{0m} \\ b_{10} & b_{11} & b_{12} & \cdots & b_{1m} \\ b_{20} & b_{21} & b_{22} & \cdots & b_{2m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ b_{m0} & b_{m1} & b_{m2} & \cdots & b_{mm} \end{pmatrix} \bullet \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_m \end{pmatrix} = \begin{pmatrix} \langle f, b_0\rangle \\ \langle f, b_1\rangle \\ \langle f, b_2\rangle \\ \vdots \\ \langle f, b_m\rangle \end{pmatrix} \tag{2.5}$$

En donde $b_{ij} = \langle b_i, b_j\rangle$ es el producto escalar definido anteriormente. Para el caso discreto, la ecuación (2.5) constituye el método de *mínimos cuadrados*. En ambos casos, el polinomio de aproximación es:

$$P_m(x) = (b_0(x), b_1(x), b_2(x), \cdots, b_m(x)) \tag{2.6}$$

2.7. Ambientes Tecnológicos

Un ambiente de aprendizaje es un espacio propicio para la adquisición de nuevos conocimientos y experiencias, en donde se presenta la interacción entre los distintos componentes que lo comprenden. Como se ha podido ejemplificar en clase, éstos no sólo se presentan mediante la interacción de profesor-alumno, sino también se pueden dar entre alumno-alumno y en todos aquellos espacios en los que éstos se relacionan e interactúan.

Así mismo, se puede definir como un ambiente de aprendizaje a:

> "Un espacio donde se lleva a cabo un proceso de aprendizaje. Se genera desde una situación inicial, con unos recursos particulares y unos participantes específicos, en una institución educativa con características dadas por su contexto" (Jaramillo, Castellano, Castañeda, Ordoñez, 2004).

Lo que denota que los ambientes de aprendizaje son propicios en la interacción entre alumno-profesor o alumno-alumno, así como hace referencia a la experiencia que se ha generado entre ellos de acuerdo a las necesidades de cada persona.

Según Nassif (1984):

> El ambiente es el medio en el que viven los individuos y a los grupos en el cual están inmersos. El ambiente puede referirse al contexto cercano en el que el individuo vive su cotidianeidad y que es percibido de manera directa por sus sentidos, como el contexto global y universal. Habiendo diferentes tipos de ambientes, el ambiente natural, el ambiente social y el ambiente cultural.

El diseño de los Ambientes Tecnológicos Interactivos propicios para el Aprendizaje de las Matemáticas (ATIAM) debe incorporar propuestas teóricas y además actividades que faciliten y estimulen la construcción de aprendizajes. La potencialidad de motivación de los ATIAM, permite que el estudiante desarrolle las habilidades necesarias para el aprendizaje de conceptos matemáticos (discusión, análisis, reflexión, etc.) y los procesos de construcción involucrados (Núñez, 2008).

Asimismo, se deben de tomar en cuenta algunos alcances que tienen los ambientes tecnológicos en el aprendizaje de las matemáticas:

1. La interactividad entre los diferentes actores, la computadora y el objeto de conocimiento se fortalecieron.

2. La motivación y el dinamismo en los estudiantes con respecto a los tradicionales.

3. Fortalecer de sobremanera el aprendizaje colaborativo cuando se trabaja en estos tipos de ambientes.

4. La experiencia en un medio interactivo deberá ser más productiva si la tecnología se combina, además con una estrategia metodológica adecuada.

5. Los estudiantes participantes adquieren un aprendizaje mayor, con un soporte técnico adecuado que les permita manejar las herramientas del ambiente de aprendizaje.

6. Se fortalece la capacidad para identificar, acceder y manejar fuentes de información.

7. Se presupone que a la mayoría de los alumnos les gustará este ambiente, aunque esto pueda ocasionar una mayor responsabilidad en su actividad y ansiedad que en un curso tradicional.

8. El ambiente de interactividad tiende a ser exitoso (según las teorías constructivistas) y con efectos positivos sobre el aprendizaje de los alumnos.

Por otro lado, hay que tomar en cuenta que, al cambiar de escenario para la enseñanza también se deberán cambiar las actitudes de todos los involucrados:

1. Si el profesor no se desprende de los hábitos que se adquieren en la enseñanza tradicional, será imposible lograr un aprendizaje significativo utilizando estos ambientes.

2. Es necesario que el alumno trabaje de tal forma que la interacción con compañeros, genere comentar sus dudas y al mismo tiempo encuentren sugerencias para resolverlas. Si ésta parte se descuida, el ambiente de aprendizaje corre el riesgo de caer en una clase tradicional.

3. En ocasiones, muchas escuelas tienen laboratorios de computación improvisados que no cubren los requerimientos mínimos. El espacio físico debe ser adecuado a las necesidades de trabajo, es decir, amplio, con mobiliario cómodo y con espacio suficiente en los escritorios para que los estudiantes pongan sus libretas cuando están resolviendo en forma escrita cada una de sus actividades.

4. En algunas ocasiones los centros educativos no cuentan con el tiempo necesario en el aula de cómputo, para materias diferentes a las de informática que puedan accesar por lo menos dos veces por semana al mismo.

2.8. Aprendizaje Colaborativo

Muchos de los desarrollos recientes que integran las nuevas tecnologías a la enseñanza de las ciencias están basados en modelos de aprendizaje colaborativo, que hacen uso intensivo del potencial comunicativo e interactivo de las nuevas tecnologías (Núñez, 2008).

Las corrientes de pensamiento socio-constructivista han mostrado que las nuevas tecnologías permiten poner en práctica principios pedagógicos, en los que el estudiante es el principal actor en la construcción de sus conocimientos. Con el apoyo de situaciones diseñadas y desarrolladas por el docente, se puede ayudar al alumno a adquirir aprendizajes significativos, dentro de un marco de acción concreta y al mismo tiempo colectiva. Las tecnologías actúan como catalizadores de cambio cuando son usadas para la enseñanza de las ciencias. El estudiante que aprende cuando se encuentra en entornos de aprendizaje tecnológicamente enriquecidos puede construir una comprensión del mundo a través de los objetos que manipula y sobre los cuales reflexiona (Núñez, 2012)

Waldegg (2002) menciona que "colaboración se refiere a cualquier actividad que dos o más individuos realizan juntos"; de manera más precisa, señala que "lo común de las diferentes definiciones de colaboración es que se enfatiza la idea de corresponsabilidad en la construcción del conocimiento y el compromiso compartido de los participantes".

Ferreiro y Calderón (2007) hacen mención, de que los antecedentes del aprendizaje colaborativo se remontan a la historia de la humanidad, ya que el aprendizaje, aunque

es un fenómeno individual, se da en un marco de relaciones sociales y de ayuda.

Hacen mención de cuatro elementos básicos que pueden ser parte de un modelo de un grupo pequeño en el aprendizaje colaborativo:

- La interacción cara a cara.
- Responsabilidad individual.
- Interdependencia positiva.
- Desarrollo de estrategias sociales.

Reynolds (1995) señala, "el aprendizaje colaborativo es en esencia, el proceso de aprender en grupo. Pero el ser capaz de introducir y dirigir el aprendizaje en equipo implica vivenciar el desarrollo de habilidades".

Según Dubinsky (1996) en el aprendizaje colaborativo, los estudiantes realizan el trabajo en grupos o equipos, esto proporciona un ambiente de interacción social que puede mejorar la maduración de su entendimiento. En este tipo de aprendizaje:

> El papel del profesor se mueve de ser la figura central de toda actividad a ser un componente del ambiente total de aprendizaje, con un papel de guía, facilitador, creador de situaciones de aprendizaje y asesor, tomando el estudiante la responsabilidad de aprender y hacer las construcciones mentales que requiera.

En adición, la tecnología, el trabajo colaborativo y la evaluación cotidiana son mediadores del aprendizaje en los alumnos. Al trabajar en equipo con la tecnología y las actividades, las interacciones que se suscitan y las negociaciones que se dan entre los estudiantes, al analizar e interpretar los resultados, ayudan a lograr un aprendizaje sólido y significativo de los conceptos que trabajan (Núñez, 2011).

Para Slavin (1999), "La esencia del aprendizaje colaborativo, es la motivación para ayudarse y alentarse para aprender, y, quizás más importante, aplaudir el éxito de otro y no su fracaso".

El aprendizaje colaborativo, menciona Ferreiro (2001), es el medio para:

- La construcción social del conocimiento.

- Lograr la calidad de la educación.
- Desarrollar las potencialidades individuales y de los equipos.

Las dos principales perspectivas teóricas de la *Teoría de la Colaboración*, que explican los mecanismos para lograr el aprendizaje colaborativo basado en computadora, son el pensamiento de Piaget y el pensamiento de de Vigotsky.

El primer mecanismo que promueve el aprendizaje en este contexto es el *conflicto sociocognitivo* de origen piagetiano, donde los niños de diferentes niveles de desarrollo cognitivo o con el mismo pero con perspectivas diferentes, pueden comprometerse en una interacción social en el pensamiento de los participantes, que dé como resultado la construcción de nuevas estructuras conceptuales y una nueva comprensión

El segundo mecanismo se formula sobre la base de las ideas de Vigotsky, en donde hay dos interpretaciones básicas. La primera supone que, a causa del compromiso en actividades colaborativas, el individuo puede realizar algo que no podía antes de la colaboración, gana en conocimiento y desarrolla nuevas competencias como resultado de la *internalización* que ocurre en un contexto de aprendizaje colaborativo. La otra interpretación, enfatiza el rol del compromiso mutuo y la construcción compartida del conocimiento, en donde el aprendizaje es más un asunto de participación en un proceso social de construcción de conocimiento que un esfuerzo individual.

Hoy en día las actividades escolares requieren que tanto profesores como alumnos, sean capaces de organizar, atender y coordinar acciones en equipo para que desarrollen en los estudiantes la motivación de dicho trabajo y sea útil en la vida laboral futura de éstos.

2.9. Registros de Representación

No es posible estudiar los fenómenos relativos al conocimiento, sin recurrir a la noción de representación. Luego de Descartes y de Kant, la representación ha sido el centro de toda reflexión que se preocupe por las cuestiones que tiene que ver con la posibilidad y la construcción de un conocimiento cierto. Y esto, porque no hay conocimiento que un sujeto pueda movilizar sin una actividad de representación. No es sorprendente pues, que igualmente se haya impuesto la noción de representación

en los estudios psicológicos sobre la adquisición de los conocimientos o sobre sus tratamientos. Ahora bien, esta noción de representación se ha presentado en tres ocasiones distintas, cada una con una determinación totalmente diferente al fenómeno designado.

La primera vez que se presentó fue en los años 1924 – 1926, como representación mental en el estudio de Piaget sobre *La representación del mundo en el niño*, relativo a las creencias y las explicaciones de los niños pequeños sobre los fenómenos naturales y físicos. El método empleado para el estudio de las representaciones mentales fue esencialmente la entrevista, en lo que puede parecer un error, es considerado como una visión distinta de las cosas o de otra lógica. En *"El nacimiento de la inteligencia en el niño"* (1937), Piaget recurre a la noción de representación como "evocación de los objetos ausentes", para caracterizar la novedad del último de los estadios de la inteligencia sensomotríz (p. 305-306)

La segunda vez, a partir de 1955 – 1960, como representación interna o computacional con las teorías que privilegian el tratamiento, por un sistema de las representaciones recibidas de modo tal que produzcan una respuesta adaptada.

Y la tercera vez, desde hace unos diez años, como una representación semiótica en el marco de los trabajos sobre la adquisición de los conocimientos matemáticos y sobre los considerables problemas que su aprendizaje suscita. La especificidad de las representaciones semiótica, consiste en que son relativas a un sistema particular de signos: el lenguaje, la escritura algebraica o los gráficos cartesianos y en que pueden ser convertidas en representaciones "equivalentes" en otro sistema semiótico, pero pudiendo tomar significaciones diferentes para el sujeto que la utiliza. La noción de representación semiótica presupone, la consideración de sistemas semióticos diferentes y una operación cognitiva de conversión de las representaciones de un sistema semiótico a otro. Esta operación ha de ser descrita en primer lugar como "un cambio de forma". Ligar una representación gráfica con una fracción numérica, o pasar del enunciado de una relación a la escritura literal de esta relación, habrá de considerarse como el "cambio de la forma en que un conocimiento está representado".

Se ha probado que cambiar la forma de una representación, es para muchos alumnos en los diferentes niveles de enseñanza, una operación difícil e incluso en ocasiones resulta casi imposible. Todo sucede como si la comprensión de un contenido

lograda por la mayoría de los alumnos quedara limitada a la forma de representación utilizada.

El análisis del desarrollo de los conocimientos y de los obstáculos encontrados en los aprendizajes fundamentales relativos al razonamiento, a la comprensión de textos y a la adquisición de tratamientos lógicos y matemáticos, enfrenta tres fenómenos que están estrechamente ligados.

Diversificación de los registros de representación semiótica. El lenguaje natural y las lenguas simbólicas no pueden considerarse como un único y mismo registro. Así como tampoco los esquemas, las figuras geométricas o los números fraccionarios.

Diferenciación entre representación y representado. O al menos entre forma y contenido de una representación semiótica. Esta diferenciación generalmente está asociada a la comprensión de lo que una representación representa y, por tanto, a la posibilidad de asociar estas representaciones y de integrarlas en los procedimientos de tratamiento.

Coordinación entre registros de representación semiótica disponibles. El conocimiento de las reglas de correspondencia entre dos sistemas semióticos diferentes no es suficiente para que puedan ser movilizados y utilizados conjuntamente.

Existen dos aspectos importantes a considerar cuando se trabaja con representaciones, éstas son: la semiosis y la noesis.

La semiosis es cualquier forma de actividad, conducta o proceso que involucre signos. Incluyendo la creación de un significado. Es un proceso que se desarrolla en la mente del intérprete; se inicia con la percepción del signo y finaliza con la presencia en su mente del objeto del signo.

La noesis es la actividad del pensamiento por la que éste accede a un conocimiento directo e inmediato del objeto. Se opone, pues, tanto a la percepción sensible, que requiere la mediación de los sentidos, como al pensamiento discursivo que recurre a la mediación del razonamiento y/o del cálculo. Para Platón la *noesis*

representa el grado más elevado de conocimiento, al ser la única capacidad del alma que permite la captación directa de las Ideas, de la verdadera realidad.

El fenómeno que nos permite comprender el papel de la semiosis en el funcionamiento del pensamiento y en el desarrollo de los conocimientos no es el empleo de tal o cual tipo de signos, sino la variedad de los signos que pueden ser utilizados. La semiosis es inseparable de una diversidad inicial de tipos de signos disponibles.

Los sistemas semióticos deben permitir cumplir las tres actividades cognitivas inherentes a toda representación:

Formación. Construir una marca o un conjunto de marcas perceptibles que sean identificables como una representación de alguna cosa en un sistema determinado.

Tratamiento. Transformar las representaciones de acuerdo con las únicas reglas propias al sistema, de modo que se obtengan otras representaciones que puedan construir una ganancia de conocimiento en comparación con las representaciones iniciales.

Conversión. Convertir las representaciones producidas en un sistema de representaciones en otro sistema, de manera tal que éstas últimas permitan explicitar otras significaciones relativas a aquello que es representado.

No todos los sistemas semióticos permiten estas tres actividades cognitivas fundamentales, por ejemplo, el lenguaje morse o la codificación de tránsito. Pero el lenguaje natural, las lenguas simbólicas, los gráficos, las figuras geométricas, etc. sí las permiten.

Los registros de representación constituyen los grados de libertad de los que puede disponer un sujeto para objetivarse él mismo de una idea aún confusa, un sentimiento latente, para explorar las informaciones o, simplemente, para comunicarlas a un interlocutor. La cuestión de la relación entre semiosis y noesis concierne únicamente a los sistemas que permiten las tres actividades de representación y no a todos los sistemas semióticos.

Generalmente no se distinguen las actividades de tratamiento y de conversión de las representaciones, no obstante ser tan diferentes. En la descripción de los procedimientos, de los recorridos o de las estrategias de respuesta, esas actividades son

reducidas a su tratamiento común, el de transformación de las representaciones dadas. Sin embargo es esencial separarlas muy bien.

Un tratamiento es una transformación que se efectúa al interior de un mismo registro, aquél en que son utilizadas las reglas de funcionamiento: un tratamiento, pues, no moviliza más que un sólo registro de representación.

La conversión es, al contrario, una transformación que hace pasar de un registro a otro; requiere pues su coordinación por parte del sujeto que la efectúa. El estudio de esta actividad de conversión debe entonces permitir la naturaleza del estrecho lazo entre semiosis y noesis.

2.9.1. Representaciones semióticas y aprendizaje

Duval (1993) señala que cada una de las diferentes formas en que puede representarse un objeto matemático proporciona diferente información acerca del mismo y, por tanto, el tener presentes varias de ellas de manera simultánea, permite tener un mejor entendimiento de él. Así mismo, las diferentes acciones que se pueden realizar sobre una representación influyen en nuestro entendimiento de los conceptos matemáticos.

Cuando se realiza un cambio de una representación a otra; es decir, cuando se realiza una conversión entre representaciones (Duval, 1993), se conecta y expande la cantidad de información que tenemos del objeto. Así mismo, cuando realizamos tratamientos sobre una representación, modificando algunas condiciones del objeto representado, se puede observar los invariantes y variaciones en los elementos característicos los objetos, lo que puede llevar al planteamiento y verificación de conjeturas, y a trabajar en una matemática más experimental.

Una comprensión de las relaciones existentes entre representaciones mentales, representaciones computacionales y representaciones semióticas, jugando un papel en la cognición, pasa por la posibilidad de una clasificación de estos diferentes tipos de representación. Para caracterizarlas, los autores Ny (1985), Palvo (1986), Larkin & Simon (1987), recurren generalmente a una de las dos oposiciones clásicas siguientes:

1. La oposición **consciente/no-consciente** es la oposición entre lo que aparece a un sujeto y él observa, de una parte; y lo que a él se le escapa y no puede

observar, de otra. En este sentido, a conciencia se caracteriza por la mirada de "alguna cosa" que toma *ipso facto* el status de objeto para el sujeto que efectúa esta mirada. El pasaje de la no conciencia a la conciencia, corresponde a un proceso de objetivación para el sujeto que toma conciencia. La objetivación corresponde al descubrimiento por el sujeto mismo de aquello que hasta entonces no sospecha, incluso si otros se lo hubieran explicado. Las representaciones conscientes son aquellas que presentan este carácter intencional y que cumplen una función de objetivación. La significación es la condición necesaria de la objetivación para el sujeto, es decir, de la posibilidad de tomar conciencia.

2. La oposición **externo/interno** es la oposición entre lo que de un individuo, de un organismo o de un sistema es directamente visible y observable, y lo que al contrario, no lo es. Esta oposición permite dividir el dominio de las representaciones mediante dos precisiones suplementarias. La primera es que todas las representaciones llamadas "externas" son representaciones producidas como tales por un sistema: no son síntomas. Como por ejemplo, la expresión de las emociones que se percibe en la cara; la segunda es que la producción de una representación externa sólo puede efectuarse a través de la aplicación de un sistema semiótico. Las representaciones externas son, por naturaleza, representaciones semióticas. Estas representaciones están por tanto estrechamente ligadas a un estado de desarrollo y dominio de un sistema semiótico. Son accesibles a todos los sujetos que han aprendido el sistema semiótico utilizado. Las representaciones internas son las representaciones que pertenecen a un sujeto y que son comunicadas a otro por la producción de una representación externa.

Una representación interna puede ser consciente ó no-consciente, mientras que una representación consciente puede ser exteriorizada ó no. El cruce de estas dos oposiciones permite distinguir tres tipos de representaciones.

1. Representaciones semióticas, son a la vez conscientes y externas.

2. Representaciones mentales, son todas aquellas que permiten una mirada del objeto en ausencia total de significado perceptible. Generalmente son identificadas con las "imágenes mentales" en tanto que entidades psicológicas que han tenido una relación con la percepción.

3. Representaciones computacionales, son todas aquellas cuyos significantes, de naturaleza homogénea, no requieren de la mirada al objeto y permiten una transformación algorítmica de una serie de significantes en otra. Estas representaciones expresan la información externa en un sistema de manera tal que la hacen direccionable, recuperable y combinable al interior de ese sistema.

2.9.2. Representaciones semióticas, tratamientos intencionales y aprendizaje

A pesar del parentesco aparente, las representaciones computacionales y las representaciones semióticas no tienen la misma naturaleza. Unas son representaciones internas a un sistema e independientes de toda visión del objeto. Otras son representaciones conscientes, inseparables de la visión de alguna cosa que toma *ipso facto* el status del objeto. Esta diferencia de naturaleza se expresa en la existencia de dos tipos de tratamiento, cuya complementariedad es indispensable para dar cuenta del funcionamiento y del desarrollo cognitivo del pensamiento humano: los tratamientos cuasi-instantáneos y los tratamientos intencionales.

Los tratamientos cuasi-instantáneos. Son aquellos que son efectuados antes de haber sido observados y que producen las informaciones y las significaciones de las cuales un sujeto tiene inmediata conciencia. Intuitivamente, los tratamientos cuasi-instantáneos corresponden a la familiaridad o a la experiencia que resulta de una larga práctica o de una competencia adquirida en un dominio.

Los tratamientos intencionales. Son aquellos que para ser efectuados toman al menos el tiempo de un control consciente y que se dirigen exclusivamente a los datos previamente observados, en una visión incluso furtiva del objeto.

Toda actividad cognitiva humana se basa en la complementariedad de estos dos tipos de tratamientos.

2.9.3. Las actividades cognitivas fundamentales de la representación ligadas a la semiosis

Hay tres actividades cognitivas de representación inherentes a la semiosis. La primera es evidentemente, la formación de representaciones en un registro semiótico particular, otra para "expresar" una representación mental, otra para "evocar" un objeto real. Esta formación implica siempre una selección en el conjunto de los caracteres y de las determinaciones que constituyen lo que se "quiere" representar. Las otras dos actividades están directamente ligadas a la propiedad fundamental de las representaciones semióticas: su transformabilidad en otras representaciones que conservan ya sea todo el contenido de la representación inicial o bien sólo una parte de ese contenido. Hablaremos de "tratamiento" cuando la transformación produce otra representación en el mismo registro. Y hablaremos de "conversión" cuando la transformación produce una representación en un registro distinto al de la representación inicial. Formación, tratamiento y conversión son las actividades cognitivas fundamentales de la semiosis.

Estas diferentes actividades están reagrupadas en lo que generalmente se llaman "tareas de producción" y "tareas de comprensión".

2.9.4. Formación de representaciones semióticas y conformidad a las restricciones de un sistema semiótico

La formación de una representación semiótica es el recurso a un signo para actualizar la mirada de un objeto. Excepto en los casos de idiosincrasia, los signos utilizados pertenecen a un sistema semiótico ya constituido y ya utilizado por otros: la lengua materna, un código icónico de representación gráfica o artística, una lengua forma, etc. Los actos más elementales de formación son, según los registros, la designación nominal de objetos, la reproducción de su contorno percibido, la codificación de relaciones o de algunas propiedades de un movimiento. Naturalmente, estos actos elementales son interesantes sólo en la medida en que las representaciones así formadas, están implícita o explícitamente, articuladas en representaciones de orden superior: frase, imagen, esquema, tabla, etc. Esta articulación en representaciones de orden superior depende de las posibilidades de estructuración propia a cada sistema

semiótico. Por tanto, es importante que la formación de representaciones semióticas respete las reglas propias al sistema empleado no sólo por razones de comunicabilidad, sino también para hacer posible la utilización de los medios de tratamiento que ofrece ese sistema semiótico empleado. Así, más que reglas de producción, preferimos hablar de reglas de conformidad.

Las reglas de conformidad son aquellas que definen un sistema de representación y, en consecuencia, los tipos de unidades constitutivas de todas las representaciones posibles en un registro. Estas se refieren esencialmente a:

- La determinación (estrictamente limitada, o al contrario, abierta) de unidades elementales (funcionalmente homogéneas o heterogéneas): símbolos, vocabulario, etc.
- Las combinaciones admisibles de unidades elementales para formar unidades de nivel superior: reglas de formación de un sistema formal, gramática de las lenguas naturales, etc.
- Las condiciones para que una representación de orden superior sea una producción pertinente y completa: reglas canónicas propias a un género literario o a un tipo de producción en un registro.

Las reglas de conformidad permiten identificar un conjunto de elementos físicos o de trazos (sonidos, posiciones opuestas de un circuito, caracteres, rayas, etc.) como una representación de alguna cosa en un sistema semiótico: es un enunciado en alemán, es un cálculo, es una fórmula física, es una figura geométrica, es una caricatura, es un esquema de un circuito, etc. Las reglas, pues, permiten el reconocimiento de las representaciones en un registro determinado.

2.9.5. Tratamiento de las representaciones semióticas y expansión informacional

Un tratamiento es la transformación de una representación (inicial) en otra representación (terminal), respecto a una cuestión, a un problema o a una necesidad, que proporcionan el criterio de interrupción en la serie de las transformaciones efectuadas. También es la transformación de una representación interna a un registro de

representación o a un sistema. El cálculo es un tratamiento interno al registro de una escritura simbólica de cifras o de letras; sustituye en el mismo registro de escritura de los números, expresiones nuevas por expresiones dadas.

Las reglas de expansión de una representación, son por definición, reglas cuya aplicación da una representación del mismo registro que la representación de partida, estas reglas son totalmente distintas a las reglas de conformidad. Las primeras reglas de expansión informacional explícitamente realzadas, lo han sido en el marco de la lógica, bajo la apelación de reglas de derivación (regla de apelación y regla de sustitución). Las reglas de producción, reglas de naturaleza inferencial definidas en el marco de la Inteligencia Artificial, reglas de coherencia temática y las reglas asociativas de contigüidad y de similitud.

2.9.6. Conversión de las representaciones y cambio de registro

La conversión es la transformación de la representación de un objeto, de una situación o de una información dada en un registro, en una representación de este mismo objeto, esta misma situación o de la misma información en otro registro. Las operaciones que habitualmente se han designado con los términos “traducción”, “ilustración”, “transposición”, “interpretación”, “codificación”, etc., son operaciones que hacen corresponder una representación dada en un registro con otra representación en otro registro. La conversión es pues, una transformación externa relativa al registro de la representación de partida. Por ejemplo: el planteamiento en ecuación de los datos del enunciado de un problema, es la conversión de las diferentes expresiones lingüisticas de las relaciones en otras expresiones de esas relaciones en el registro de una escritura simbólica.

2.10. Uso de Tecnologías en la Enseñanza de las Matemáticas

Estudios realizados en los últimos años han demostrado que el uso de nuevas tecnologías abre perspectivas interesantes para la enseñanza de las matemáticas y de otras ciencias (López, 2008). Entre los beneficios que brindan se pueden mencionar los siguientes:

- Ofrece al estudiante ambientes de trabajo que estimulan la reflexión y lo convierten en un ser activo y responsable de su propio aprendizaje.
- Provee un espacio problemático común al maestro y al estudiante, para construir significados.
- Elimina la carga de los algoritmos rutinarios para concentrarse en la conceptualización y la resolución de problemas.
- Da un soporte basado en la retroalimentación.
- Reduce el miedo del estudiante a expresar algo erróneo y, por lo tanto, se aventura más a explorar sus ideas.

En el área de las Matemáticas es posible destacar un sinnúmero de herramientas físicas e intelectuales que se han venido manejando a través de los años, hasta llegar hoy en día al uso de la computadora, la cual es clasificada como una herramienta general, limitada en su aplicación sólo por la imaginación del usuario.

Por lo que respecta a su uso en la enseñanza de las matemáticas, debemos de reflexionar la gran influencia que ha tenido tanto en la educación matemática como en la sociedad, ya que esta nueva tecnología requiere una nueva interpretación del proceso que se lleva a cabo en la enseñanza y en todo lo concerniente con el proceso de educar a las nuevas generaciones.

Tomando en cuenta la extensa variedad de los recursos tecnológicos que se manejan actualmente, incluyendo el desarrollo de nuevos programas para computadora, ponen de manifiesto la necesidad de que el alumno tenga una nueva visión sobre las

distintas alternativas que se le pueden presentar para su enseñanza de las matemáticas. Actualmente existe un gran acceso al manejo de una computadora, tanto en los hogares como en las escuelas, esto se debe a la demanda que existe por parte de la sociedad al uso de nuevas tecnologías.

Es de manejo cotidiano el escuchar el uso de calculadoras en las escuelas, por lo que es importante destacar la diferencia existente entre las calculadoras y la tecnología computacional; las calculadoras por un lado podrían distinguirse por su naturaleza manual, ya que fueron pensadas con propósitos específicos de herramientas de cálculo para matemáticas, cuyo rango de modelos va desde las más simples con operaciones básicas, hasta las que incluyen graficación, resolución de ecuaciones, capacidades de programación, manipulación simbólica, etc. Mientras que por el otro lado, las computadoras son de naturaleza más general, funcionalmente definidas por su software, mismas que incluyen modelos manuales, los cuales han sido diseñados con las necesidades de las matemáticas de hoy en día.

La computadora como herramienta de aprendizaje puede favorecer diferentes aspectos de la enseñanza; por ejemplo, cuando se requieren hacer cálculos repetitivos o graficar conjuntos de datos, la potencia de cálculo de una computadora nos permite acelerar este proceso. Para la computadora esta etapa es trivial, en el sentido de que solo ayuda a obtener resultados más rápidamente. La computadora actúa como un amplificador (Dreyfus, 1994).

Como herramienta, la computadora nos permite ver diferentes ángulos que no se ven en la forma tradicional. En este caso, los alumnos pueden desarrollar y ver dinámicamente el proceso de incorporación, repetir la función, observar secuencias numéricas, interactuar con diferentes representaciones a la vez, analizar los cambios que se producen en unas cuando se modifican otras. Muchos de estos aspectos dinámicos pueden ser descritos en términos cualitativos y no en términos cuantitativos. También puede cambiar la calidad de los objetos matemáticos y los procesos de las experiencias de aprendizaje; así la computadora puede convertirse en una herramienta cognitiva para el aprendizaje de las matemáticas.

Las actividades matemáticas que incluyen la tecnología computacional a todos los niveles son:

- Comunicación.

- Modelado matemático.
- Buscar, recuperar, analizar y presentar información.
- Manipulación de números, símbolos y figuras.
- Representación de ideas matemáticas y procesos usando una variedad de formas.
- Estimación de la racionalidad de resultados.
- Investigación de patrones de problemas.

El uso de la tecnología apropiada lleva al usuario a que tome el control de su propio aprendizaje, lo cual fomenta tanto el aprendizaje colaborativo como el independiente, mientras se extiende y soporta el proceso de aprendizaje.

Las herramientas de software apropiadas para el aprendizaje matemático en todos los niveles, incluyen procesadores de texto, bases de datos, paquetes para dibujar, para pintar, para comunicarse, etc. Más específicamente, existe software matemático, tal como LOGO, hojas de cálculo, graficadoras, manipuladores simbólicos, geometría dinámica, paquetes estadísticos, etc.

En el presente trabajo, se hará uso de un software para el aprendizaje de funciones, derivadas e integrales llamado "Fun-Der". Su uso se hará con la finalidad de que el alumno explore, analice, manipule y en general haga uso de su imaginación y creatividad para la completa comprensión de los conceptos involucrados en las operaciones con fracciones numéricas.

En tanto que el uso de las calculadoras y las computadoras se ha hecho más accesible, los profesores de matemáticas en todos los niveles han estado experimentando las diferentes formas de incrementar el aprendizaje de los conceptos matemáticos por parte de los estudiantes; una de ellas es el trabajo de los estudiantes en grupos o equipos de trabajo, lo que ha llevado a los profesores a observar que a medida que aumenta la interacción entre los estudiantes, aparentemente se incrementa su profundización en los conceptos que se están aprendiendo, así como su interés en la solución de problemas.

Algo que es importante destacar, es que ni la calculadora ni la computadora llegarán a sustituir al maestro, sino que hay que tomarlos como instrumentos de apoyo,

tal como lo son el gis y el pizarrón (o sus equivalentes), aunque sus características sean esencialmente diferentes.

Siendo el objetivo principal del empleo de la tecnología en el aula, no el reducir la práctica de algoritmos, sino el de ayudar al estudiante a descubrir y construir conceptos y técnicas mediante el ejercicio de la reflexión. De esta manera, la matemática pasa a ser mucho más que una simple mecanización de procedimientos.

Capítulo 3

"Fun-Der"

3.1. Introducción

En este capítulo se presenta el acercamiento a los conceptos de derivadas e integrales por medio del manejo de tablas y gráficas, las cuales permiten realizar un desarrollo intuitivo a las funciones derivada e integral. Para tal efecto se desarrolló un software educativo, que toma como base teórica las ideas propuestas por Duval sobre Registros Semióticos de Representación, para con ellas a su vez, proponer una serie de actividades didácticas.

Se expondrá una alternativa para la enseñanza de los conceptos de derivada e integral a partir de analizar los puntos de una función dada. Para ello, se construyó un software educativo en el que se presentan diferentes registros semióticos de representación. Como punto de arranque se inicia con el registro de representación numérico, mediante la introducción de ideas relacionadas con progresiones aritméticas, posteriormente se intercala el registro semiótico de representación gráfico.

Cabe señalar también, que un tratamiento numérico y gráfico del concepto de razón de cambio, es un pilar importante para abordar posteriormente el concepto de función derivada e integral. Por otro lado, como argumenta Mejía (1997, p. 316), "en cuanto al manejo de tablas, hay muy pocos trabajos respecto a su tratamiento" por lo que consideramos importante promover este tipo de acercamiento, ya que de acuerdo con Confrey (Scher, 1993) "la presentación algebraica de funciones obscurece la conexión funcional numérica en un grado importante. Las tablas pueden ayudar

a iluminar a esta conexión". Parte de este tipo de acercamiento queda plasmado en los ejemplos que se presentan en el libro Funciones en Contexto (Hitt, 2002) y que se ha extendido de tal manera, que se permita introducir explícitamente, las nociones de incremento y de razón de cambio.

Es importante señalar que en diversos estudios se ha comprobado que la conversión de un registro semiótico de representación a otro, causa un conflicto que no es trivial; por ejemplo Hitt (1995, p. 63) detectó errores en "no contextualizar (analíticamente) la variable independiente que aparece en una gráfica", Duval (1988) afirma que es de mayor dificultad el paso de una representación gráfica a una algebraica, no siendo así en forma inversa.

3.2. Interfaz Principal

El software Fun-Der tiene tres módulos (Tratamiento numérico, Derivadas, e Integrales) estructurados a través de un menú, los cuáles muestran el proceso de enseñanza que se propone con la expectativa numérico-gráfico y conceptual; cada módulo presenta a su vez secciones de trabajos específicos. El diseño de la interfaz principal está ilustrada en la figura (3.1).

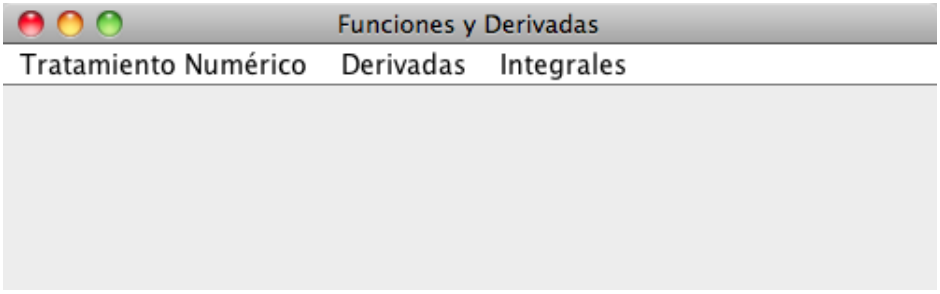

Figura 3.1: Interfaz principal.

El primer módulo "Tratamiento Numérico" comprende tres secciones (`JMenuItem`), estructurados con base en la secuencia de aprendizaje que se propone, comprendiendo tanto un trabajo meramente numérico y el trabajo numérico-gráfico como se presenta en la tabla 3.1.

Tratamiento Numérico			
Trabajo Numérico	Trabajo Numérico y Gráfico	Trabajo Numérico y Gráfico	Trabajo Numérico y Gráfico
Progresiones Aritméticas	Incrementos (de una variable)	Razones de cambio (Construcción de la función razón de cambio)	Diferencias de funciones

Tabla 3.1: Estructura del módulo "Tratamiento Numérico".

El segundo y tercer módulo, "Derivadas" e "Integrales" respectivamente, comprenden dos secciones (`JMenuItem`), organizadas para presentar en primera , un trabajo numérico (con diferencias e incrementos, respectivamente), y por otra parte el gráfico de derivadas de funciones (gráficas de una función y su derivada o integral), estructurado como se muestra en la tabla 3.2

Derivadas e Integrales	
Trabajo Gráfico	Trabajo gráfico
Grafico de funciones	Grafico de derivadas

Tabla 3.2: Estructura del módulo "Derivadas" e "Integrales".

3.3. Tratamiento Numérico

Como se ha mencionado antes, este módulo consta de tres secciones `JMenuItem`, siendo: "Progresiones Ariméticas", "Incrementos" y "Razones de Cambio". Dichas secciones se describirán más adelante (figura 3.2).

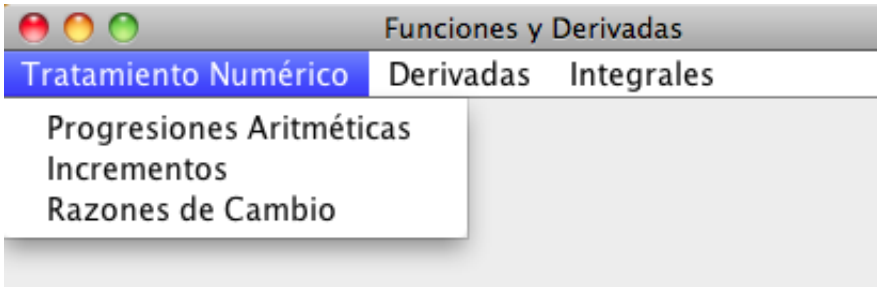

Figura 3.2: Menú "Tratamiento Numérico".

3.3.1. Progresiones ariméticas

En esta primer sección se propuso realizar un trabajo numérico, con la idea de que el estudiante recordara progresiones aritméticas, generando en el usuario una motivación al momento de retarlo a solucionar el problema planteado.

Por otra parte, al iniciar con una actividad sencilla produce que el usuario entienda perfectamente la tarea que se plantea, y en consecuencia, adquiera confianza. (Figura 3.3)

Figura 3.3: Interfaz de "Progresiones Aritméticas".

Se programan las tablas utilizando algunos `JTextField[]`; un grupo de botones (`ButtonGroup`) integrados por cuatro `JButton` que representan los niveles de dificultad; un `JButton` que permite borrar los datos de las tablas; un `JButton` que verifica los resultados; y un `JButton` para regresar a la pantalla principal.

Para que el estudiante pueda comprender la definición de lo que es una progresión aritmética, se abordan ejercicios, como el obtener la diferencia entre dos elementos consecutivos; también se introduce la terminología que se usará en el software. *¿Qué es "posición"?*, y también *"¿Qué es "valor"?*.

En los ejercicios planteados se relacionan dos progresiones aritméticas, a la primera se le denomina "posición" y a la segunda "valor".

Nivel 1

La diferencia entre los elementos de "Posición", siempre es 1. Las diferencias entre los elementos de "Valor" es un entero entre 0 y 10 (figura 3.4). Donde la relación posición-valor se presenta en forma de tabla; inicialmente sólo se agregan los tres primeros valores y es el objetivo para el usuario agregar el resto de ellos.

De este modo, el usuario comienza a trabajar con la relación entre progresiones (código A.12).

Figura 3.4: Tabla de Progresiones, nivel 1.

Nivel 2

Dentro de este nivel, la diferencia entre los elementos de "Posición" es distinto de 1, pero permanece constante y los elementos de "Valores" siguen con una diferencia constante entre 0 y 10 (figura 3.5), la estrategia en cuanto a trabajar en este nivel es similar a la del nivel 1 (código A.13).

Figura 3.5: Tabla de Progresiones, nivel 2.

Nivel 3

En este nivel la "Posición" se presenta con diferencias distintas de 1 y no constantes, el "Valor" tendrá una diferencia entre 0 y 10 (figura 3.6). En cuanto al trabajo del usuario, en este nivel es similar al nivel anterior, salvo posiciones grandes con la finalidad de que el usuario deduzca una fórmula para calcular dichos valores. Parte del psuedo-código se muestra en el código A.14.

Figura 3.6: Tabla de Progresiones, nivel 3.

Nivel 4

En este nivel la posición se presenta con diferencias distintas de 1 y no constantes, los datos correspondientes al valor aparecen en distintas posiciones, lo que aumenta la complejidad para el usuario (figura 3.7). En cuanto al trabajo del usuario en este nivel, se debe explicitar una fórmula para calcular los valores (código A.16).

Figura 3.7: Tabla de Progresiones, nivel 4.

3.3.2. Incrementos

Esta clase contiene tablas creadas con `JTextField`; un grupo de botones de opción con cuatro `JRadioButton` para los niveles del 1 al 4; un objeto `Canvas` conteniendo un gráfico procedente de otra clase; dos `JSlider` que manejan la escala del gráfico; tres `JButton`, un botón para limpiar los datos, otro para verificar los datos introducidos y un tercer botón para regresar a la pantalla principal. (figura 3.8)

Con los elementos antes mencionados el desarrollo de la actividad didáctica es: una función lineal aleatoria para presentar en las tablas los tres primeros valores de (x, y) y de sus respectivos incrementos Δx y Δy, de igual forma su representación gráfica. Esto se origina cuando es pulsado uno de los `JRadioButton` de niveles ya que cada uno de ellos es asignado a una clase interna anónima, a su vez, los datos manejados en estas clases internas son enviadas a la clase gráfica `Canvas` para su trabajo gráfico. Cuando el usuario comienza la actividad, debe llenar los `JTextField` con los datos correctos, evaluados dentro de la clase "incrementos" en el método

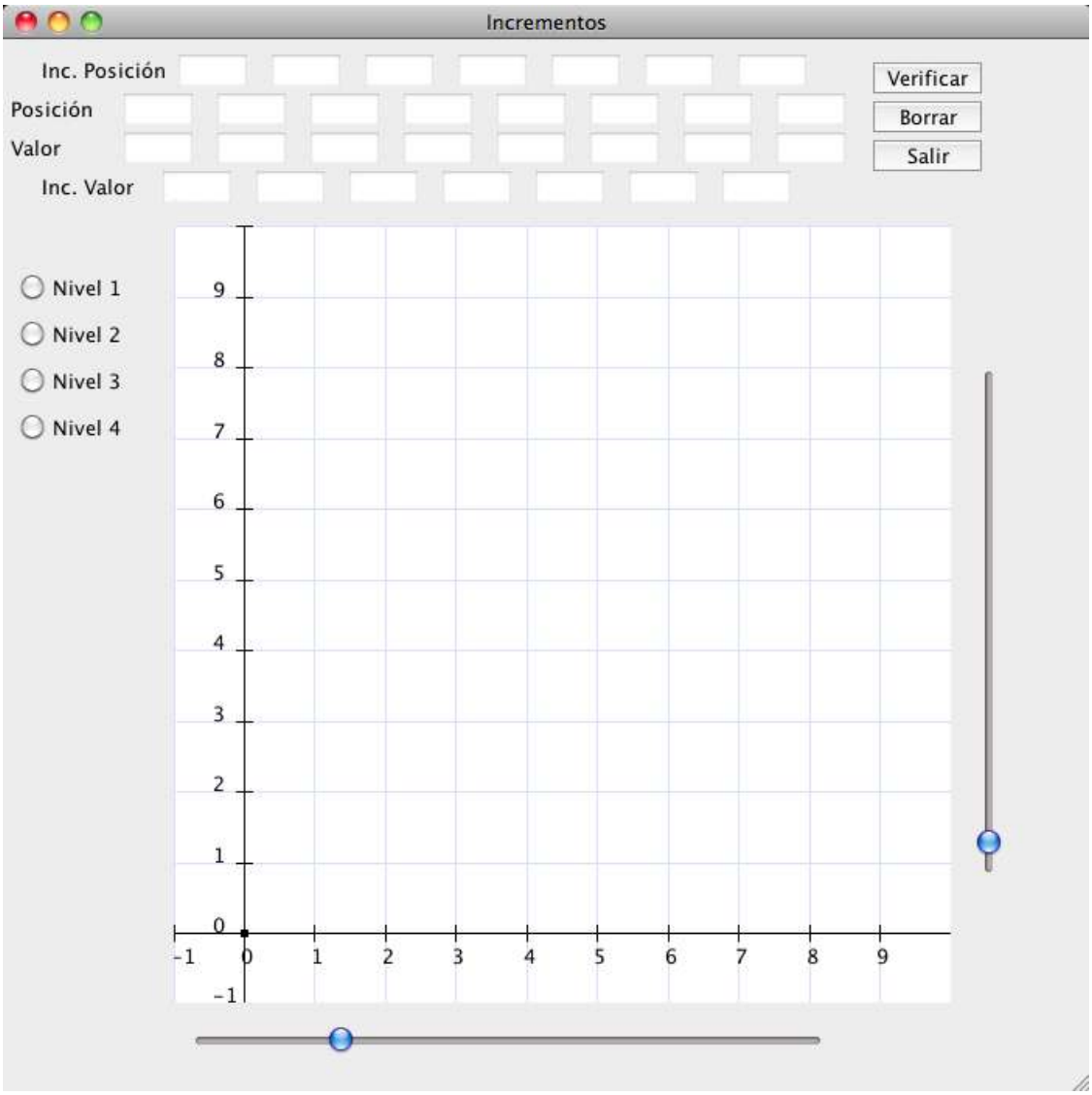

Figura 3.8: Interfaz de "Incrementos".

`actionPerformed` y una vez que sean correctos, se hace la llamada a la clase `Canvas`, de ese modo se presentará al usuario la información gráfica y numérica.

A los `JSlider`, de igual forma se les tiene programados con clases internas anónimas realizando una llamada a `addChangeListener`, que permita capturar los eventos del slider, mandar los valores a la clase `Canvas` donde esta última realizará los respectivos cálculos permitiendo el redimensión del gráfico.

En esta sección se proponen ejercicios con un enfoque numérico-gráfico para que se pueda producir una aproximación al registro gráfico. El objetivo didáctico de los ejercicios planteados es introducir al estudiante en el uso de los incrementos de una variable y explicar gráficamente lo que es una razón de cambio vista como pendiente de una línea recta.

Nivel 1

Los valores de la posición comienzan en 1, y su diferencia siempre es de 1; los valores tienen una diferencia constante entre -10 y 10.

Se muestran los tres primeros valores de "Incremento en la posición", de "Valor" y de "Incremento en Valor", y se muestra toda la tabulación de "Posición" (figura 3.9). Siendo tarea del usuario llenar el resto de la tabla correctamente (código A.17).

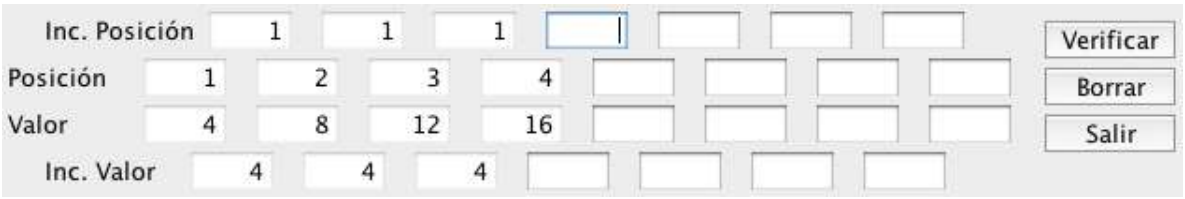

Figura 3.9: Tabla de Incrementos, nivel 1.

En la ventana gráfica se muestran los puntos en azul, y los incrementos en rojo (figura 3.10).

Con ello se puede explicar la relación de lo que se está obteniendo (razón de cambio) y lo que es la pendiente de la recta que une a dos puntos.

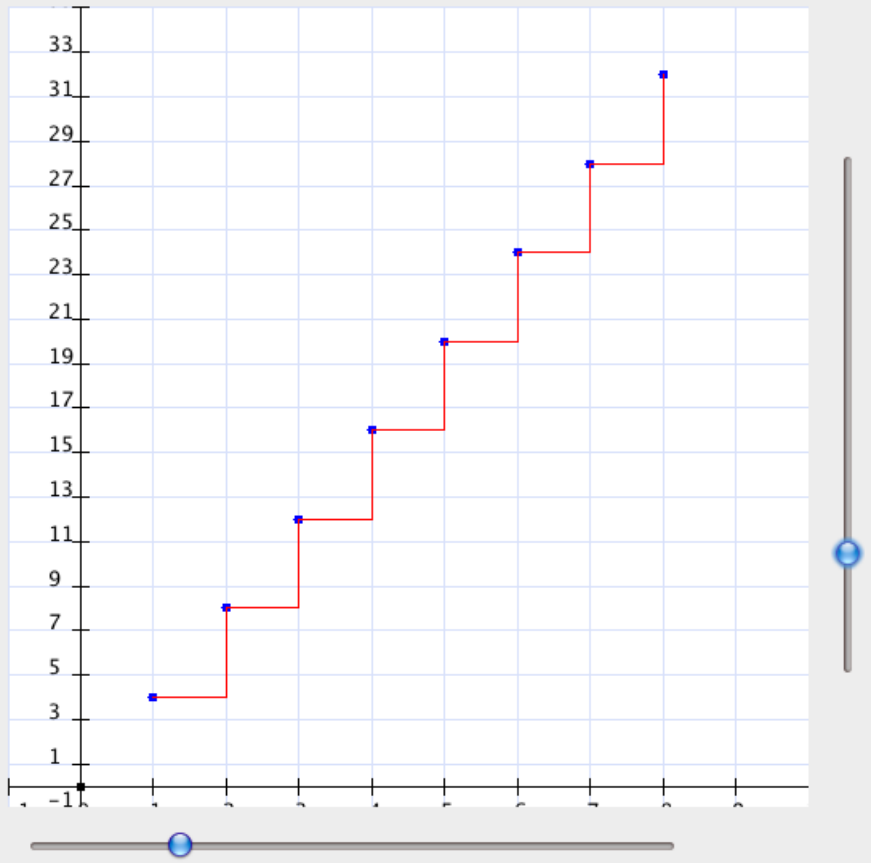

Figura 3.10: Gráfica de Incrementos.

Nivel 2

Dentro de este nivel, el "Inc. Posición" será una constante distinta de 1; y los "Valores" siguen con una diferencia entre -10 y 10 (figura 3.11), la estrategia para trabajar en este nivel es similar a la del nivel 1. (Código A.17)

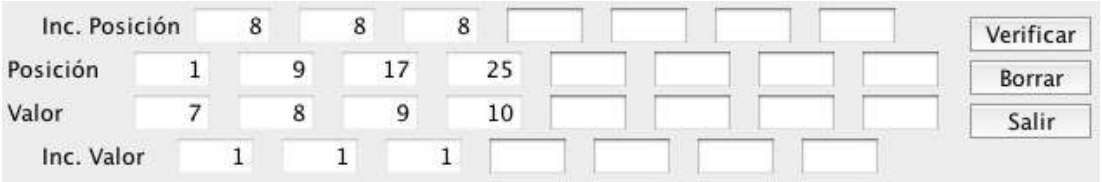

Figura 3.11: Tabla de Incrementos, nivel 2.

Nivel 3

En este nivel, el "Incremento en la Posición" se presenta con diferencias no constantes, el "Valor" tendrá una diferencia entre 1 y 10 (figura 3.12). En cuanto al trabajo del usuario en este nivel, deberá ser similar al nivel anterior, salvo en las posiciones "grandes" con la finalidad de que el estudiante deduzca una fórmula para calcular dichos valores. (Código A.18)

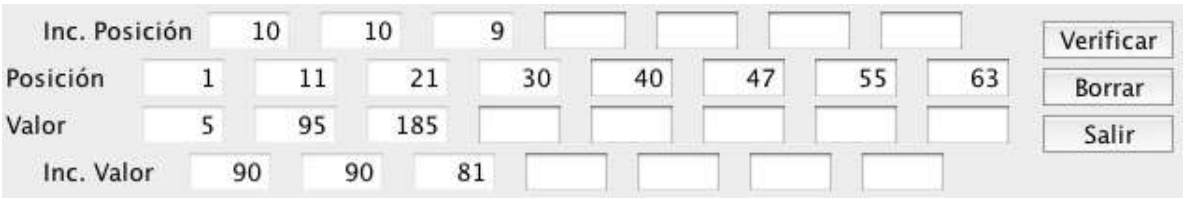

Figura 3.12: Tabla de Incrementos, nivel 3.

Nivel 4

En este nivel, la "Posición" se presenta con diferencias distintas de 1 y no constantes; los datos de "Valor" aparecen en distintas posiciones, lo que aumenta la complejidad para el usuario (figura 3.13). Para resolver este nivel, el usuario deberá deducir la fórmula para calcular los valores. (Código A.18)

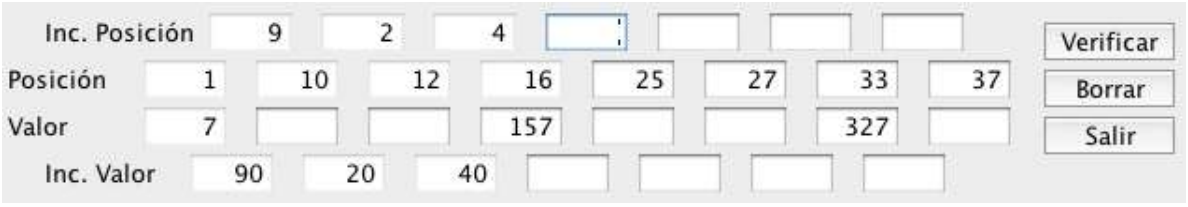

Figura 3.13: Tabla de Incrementos, nivel 4.

En cuanto al manejo gráfico, se mostraran los puntos en color azul, mientras que los incrementos Δx y Δy se mostrarán en verde y amarillo, respectivamente (figura 3.10).

3.3.3. Razón de Cambio

La interfaz de "Razones de Cambio" (figura 3.14) se compone de un menú (Tipo de Función), el cual engloba funciones polinomiales (lineal, cuadrática y cúbica) y funciones trigonométricas (seno, coseno y tangente). Muestra también una tabulación para los pares (x, y) y sus respectivos incrementos Δx y Δy. De igual forma, muestra otra tabla para la función "Razón de Cambio", y gráficos para la función seleccionada, como para la función razón de cambio. Se tienen cuatro botones, "Verificar", "Cambiar Función", "Borrar" y "Salir".

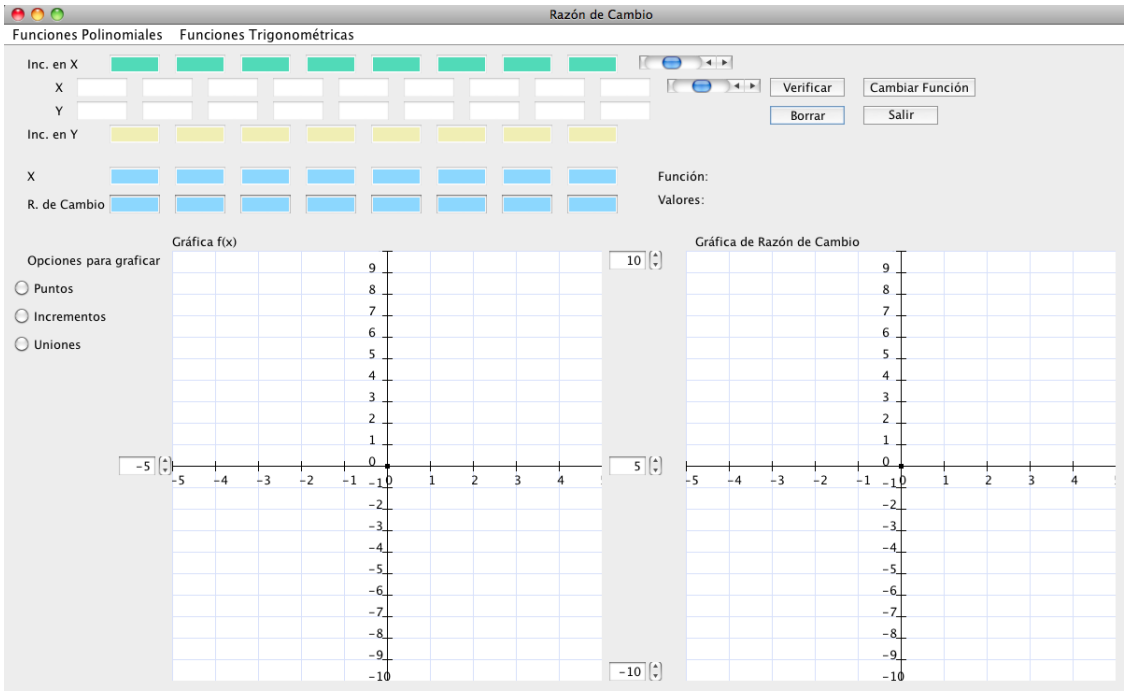

Figura 3.14: Interfaz de "Razón de Cambio"

Esta clase contiene un `JMenu` que permite seleccionar el tipo de función; la tabulación con `JTextField` tanto de la función (x, y) como la función de la razón de cambio; sus respectivos gráficos (`Canvas` que contienen los métodos para graficar); `JSlider` con los cuales poder modificar las escalas del eje x y del eje y de los gráficos; un grupo de botones `JCheckBox` de selección para dar opción sobre qué es lo que se desea graficar; un par de etiquetas `JLabel` para mostrar la función seleccionada con sus respectivos valores y parámetros; por último la programación de cuatro `JButton`,

para verificar, borrar, salir y cambiar los parámetros de la función elegida.

La estrategia de esta clase es de la siguiente forma: Cuando se selecciona un tipo de función (con alguno de los `JMenuItem`), los cuales son clases internas anónimas, se generan en las tablas los datos (excepto la tabla de razón de cambio) y se envían los valores a las clases gráficas asociadas, imprimiendo así la representación gráfica de la función que se ha seleccionado, y la función razón de cambio. Cuando el usuario completa la tabla de razón de cambio, son evaluadas con el método `actionPerformed` en la clase "Razón de Cambio", una vez que los datos son válidos se envían los valores para que el segundo `Canvas` los grafique.

Para la manipulación de los gráficos se tienen programados cuatro `JSpinner` con clases internas anónimas, que realizan llamadas a `addChangeListener`, que permiten capturar los nuevos valores de los `JSpinner` y mandar a redimensionar los `Canvas`.

Para los gráficos de las funciones $f(x)$ y la función razón de cambio, se dan las opciones de graficar los puntos, los incrementos y/o las uniones; las acciones de selección son recibidas por los `actionListener` de los JCheckBox y pasan la información recibida a los Canvas, para que realicen un `repaint` y pinten lo necesario.

Función Polinomial

Al seleccionar una función, por ejemplo una polinomial cuadrática, se muestran una tabla con la información de los puntos (x, y) y sus incrementos $(\Delta x, \Delta y)$ (figura 3.15), se le pide al usuario que llene la tabla correspondiente a la función "razón de cambio" (figura 3.16) que resulta de dividir Δy entre Δx (en los códigos A.20 y A.21 se muestran cómo se guardan los arreglos de puntos).

Inc en X	1	1	1	1	1	1	1
X	0	1	2	3	4	5	6	7
Y	-9	-11	-7	3	19	41	69	103
Inc en Y	-2	4	10	16	22	28	34	

Figura 3.15: Tabla de valores $(x,\ f(x))$, sección "Razón de Cambio".

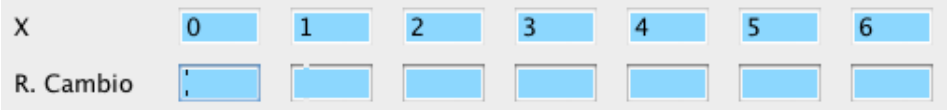

Figura 3.16: Tabla de valores $\frac{\Delta y}{\Delta x}$, sección "Razón de Cambio".

Se presenta también la información en formato gráfico; se muestra en el panel izquierdo la función $y = f(x)$, y en el panel derecho la función razón de cambio (figura 3.17), así como la posibilidad de conocer la función generada (figura 3.18).

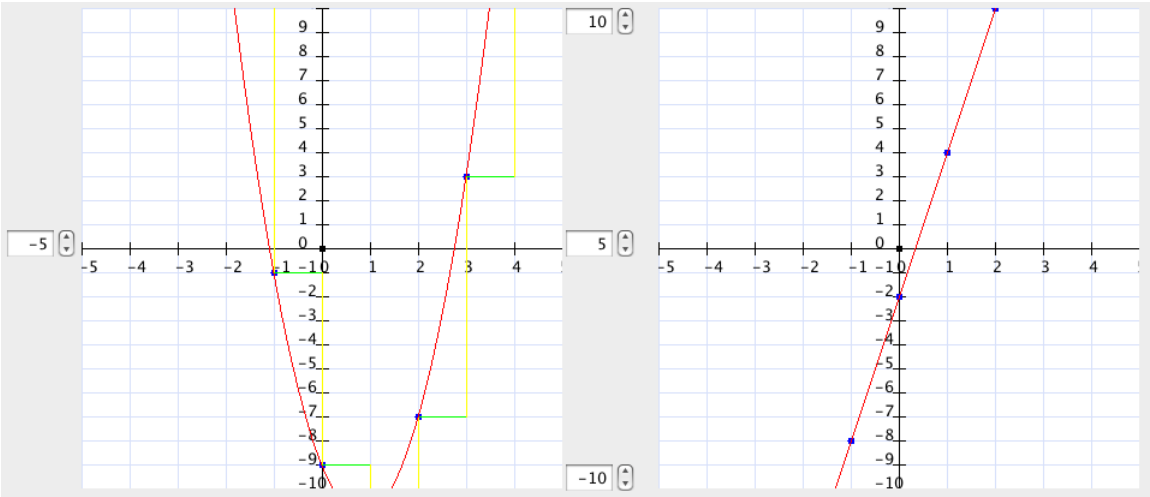

Figura 3.17: Gráficos de funciones, sección Razón de Cambio.

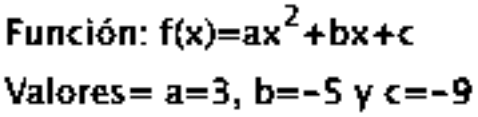

Figura 3.18: Recuadro con los valores de función, sección Razón de Cambio.

Para poder graficar la curva $f(x)$ usamos el método descrito en el código A.8, mientras que para graficar la segunda curva necesitaremos interpolar la función que aproxima a $f'(x)$, para ello usaremos el método del código A.10 y después lo mandamos graficar (código A.8).

Función Trigonométrica

En caso de elegir una función trigonométrica, se obtiene la figura 3.19.

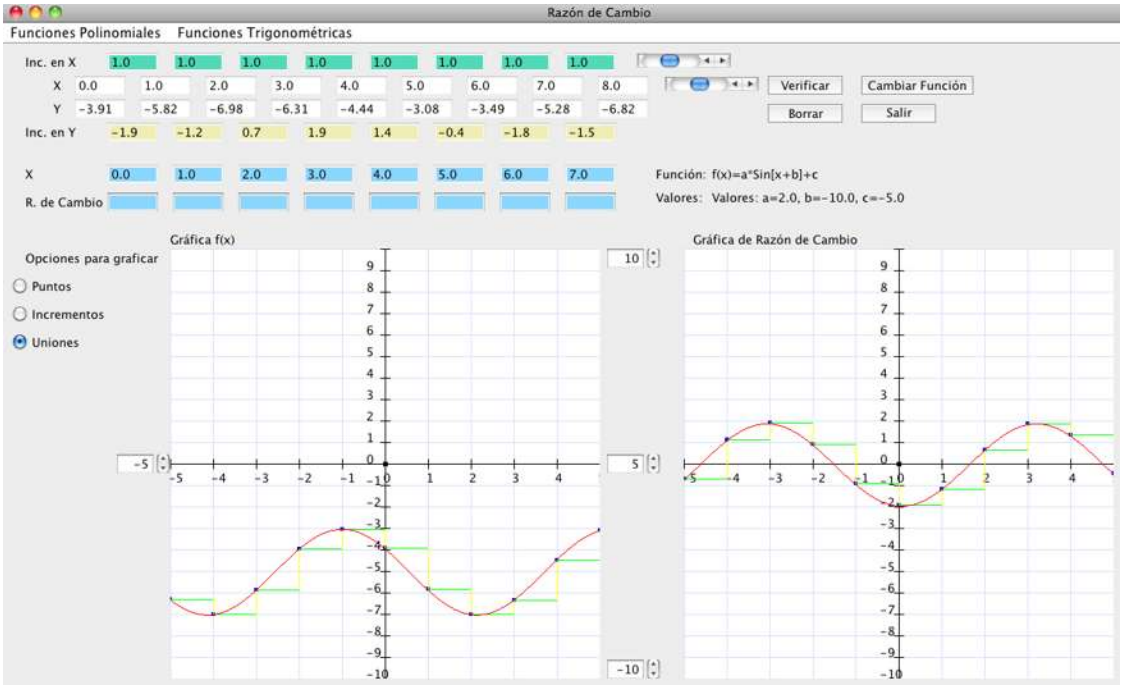

Figura 3.19: Interfaz de Razón de cambio, función trigonométrica.

De manera análoga al caso en donde la función es polinomial, obtenemos la tabulación (figura 3.20), y las gráficas de la función original y razón de cambio (figura 3.21).

Inc. en X	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
X	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0
Y	-3.91	-5.82	-6.98	-6.31	-4.44	-3.08	-3.49	-5.28	-6.82
Inc. en Y	-1.9	-1.2	0.7	1.9	1.4	-0.4	-1.8	-1.5

Figura 3.20: Tabulación de una función trigonométrica.

El llenado de datos y la gráfica de la función original se hace de forma análoga; la diferencia está en la forma de interpolar los puntos para generar un polinomio trigonométrico de aproximación.

Para el panel de la izquierda: se evalúa la función elegida (código A.22) para generar la tabulación $(x, f(x))$; ahora creamos la tabulación de las razones de cambio (código A.21); el último arreglo de puntos se convierte a pixel (código A.7) y se mandan a graficar.

Para el segundo panel interpolamos los puntos obtenidos resolviendo la matriz generadas con el código A.24 y código A.25, y se manda graficar el nuevo polinomio de aproximación.

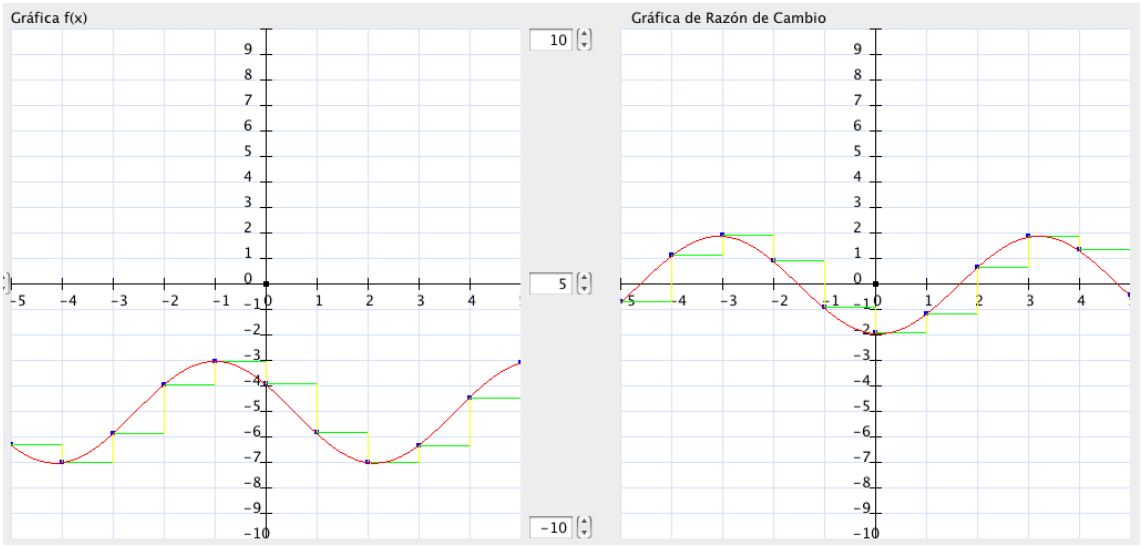

Figura 3.21: Gráfica de una función trigonométrica y su razón de cambio.

3.4. Derivadas

Este módulo consta de dos secciones (`JMenuItem`), siendo: "Aspecto Numérico y Gráfico" y "Gráfica de Derivadas". Dichas secciones se describirán a continuación (figura 3.22).

En la presente sección se propone un análisis de las funciones, en especial de las polinomiales y trigonométricas, por medio de la observación de las diferencias que se implementan, es decir, primera, segunda y tercer diferencia.

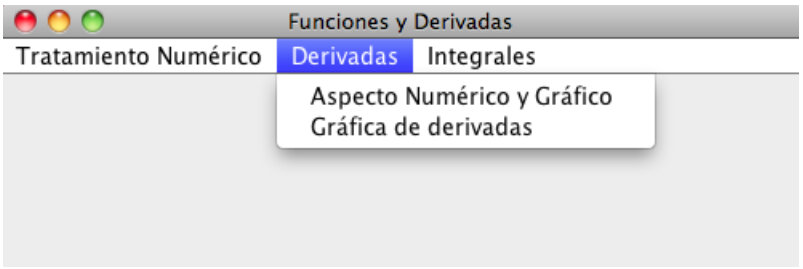

Figura 3.22: Menú "Derivadas".

En la interfaz de la sección "Derivadas" (figura 3.23), se aprecia la organización tanto de tablas y gráficos (de la función seleccionada como de sus respectivas diferencias); también se dispone de elementos como botones, opciones para los gráficos y cajas de texto. Al igual que en la sección "Razón de Cambio", también hay un menú "Tipo de Funciones" para poder seleccionar entre funciones polinomiales (lineales, cuadráticas y cúbicas) y trigonométricas (seno, coseno y tangente).

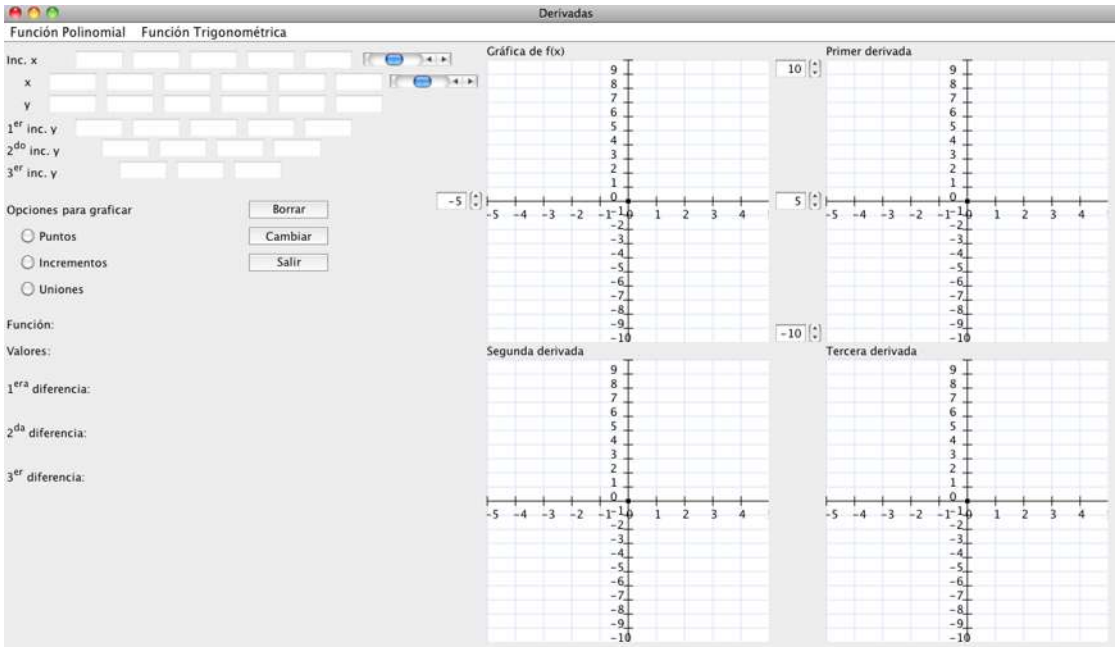

Figura 3.23: Interfaz de la sección "Derivadas".

Se programaron elementos del tipo: `JMenuBar`, `JMenu`, `JMenuItem`, `JTextField`, `JCheckBox`, `JButton`, `JPanel`, `JLabel`, `JSpinner`, `JSlider`, `JScrollBar`, y cuatro

objetos `Canvas`.

El funcionamiento de esta clase es similar a las clases anteriores, donde se calculan valores iniciales según el tipo de función que el usuario haya seleccionado, se puede modificar el incremento Δx que se asocia con un `actionListener`, después se envían los valores a las clases para graficar.

En este espacio se desarrollan ejercicios de diferencias con base en las funciones, para que el usuario desarrolle un sistema de ecuaciones lineales que le permitan solucionar los problemas propuestos.

Para la manipulación de los gráficos, de igual forma, se tienen programados cuatro JSpinner's con clases internas anónimas realizando una llamada a `addChangeListener`, que permita capturar los eventos de los spinner's, mandar los nuevos valores a las clases Canvas donde se realizan los respectivos cálculos, permitiendo así la redimensión del gráfico.

3.4.1. Función Polinomial

Al seleccionar una función cúbica: $y = 4x^3 + 4x^2 + 4x + 3$, se asignan los datos en las tablas: función (x, y), incremento en x (Δx), incremento en y (Δy) o bien primera diferencia, segunda diferencia y tercera diferencia (figura 3.24)

Siendo evidente que:

- El primer incremento de y es una función cuadrática:

$$f(x) = 12x^2 + 20x + 12$$

- El segundo incremento de y es una función lineal:

$$f(x) = 24x + 32$$

- El tercer incremento de y es una función constante:

$$f(x) = 24$$

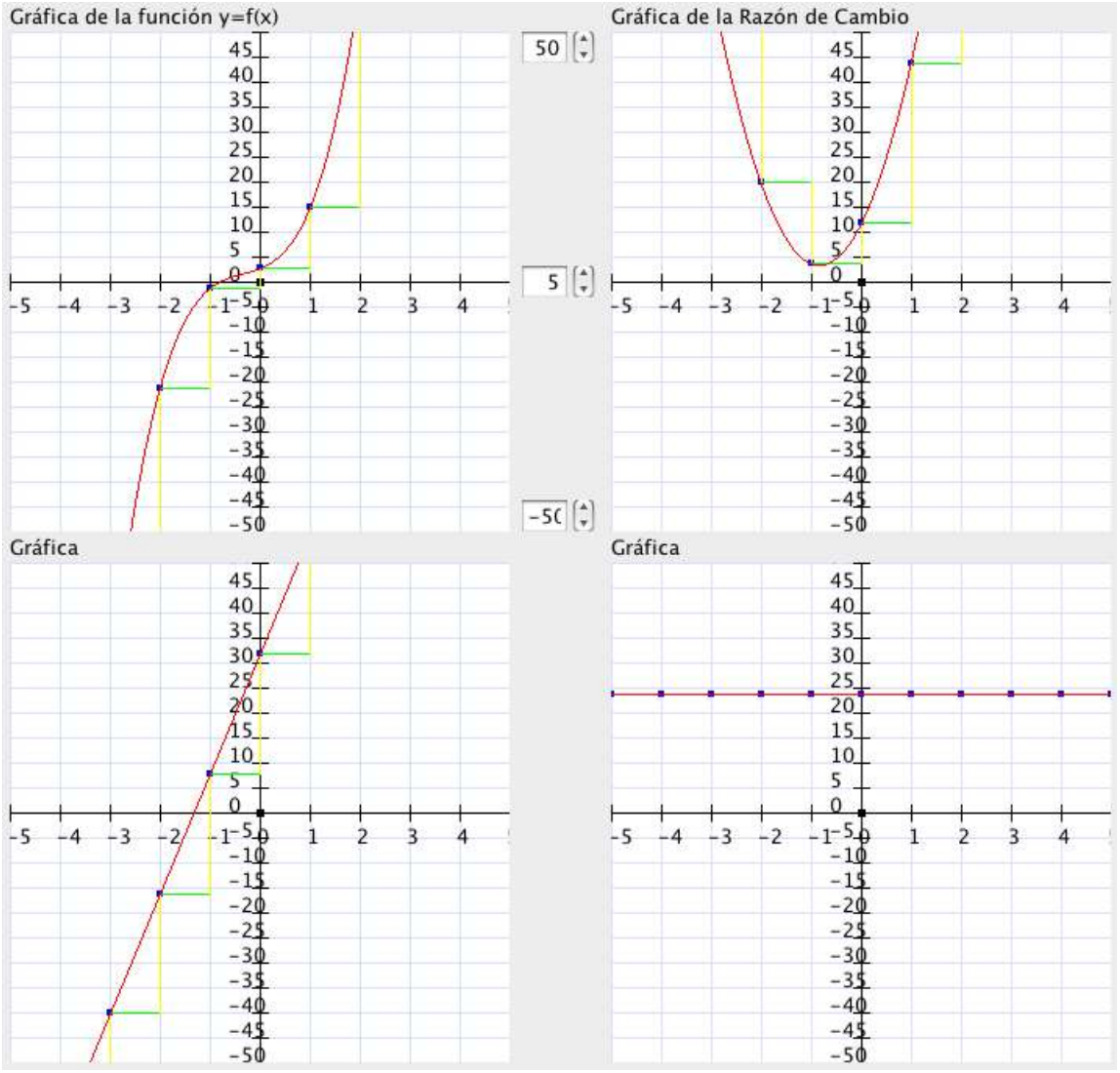

Figura 3.24: Gráficos de funciones, sección "Derivadas".

Dichas expresiones algebraicas de las diferencias se muestran en cajas de texto dentro de la interfaz, así mismo, se puede apreciar la función que ha sido seleccionada y sus respectivos parámetros (figura 3.25).

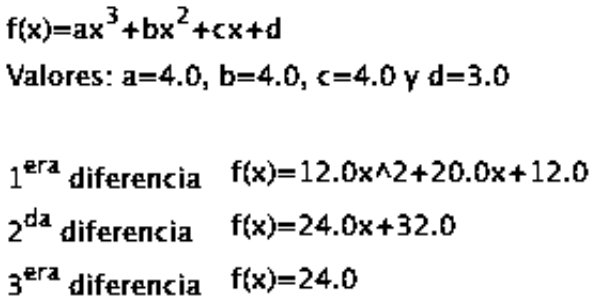

Figura 3.25: Expresión algebraica de las funciones, sección "Derivadas".

Estas funciones se construyen considerando las siguientes relaciones tabulares. La expresión $f(x) = 12x^2 + 20x + 12$, se obtiene de la tabla mostrada en la figura 3.26:

x	0.0	1.0	2.0	3.0	4.0
1er Inc. y	12.0	44.0	100.0	180.0	284.0

Figura 3.26: Tabla de valores de la primera diferencia.

La expresión $f(x) = 24x + 32$, se obtiene de la tabla ilustrada en la figura 3.27:

x	0.0	1.0	2.0	3.0
2do Inc. y	32.0	56.0	80.0	104.0

Figura 3.27: Tabala de valores de la segunda diferencia.

La expresión $f(x) = 24$, se obtiene de la tabla ilustrada en la figura 3.28:

x	0.0	1.0	2.0	3.0
3er Inc. y	24.0	24.0	24.0	104.0

Figura 3.28: Tabla de valores de la tercer diferencia.

Además de todo lo descrito, se puede variar el incremento en x, teniendo un rango de valores para Δx desde 0.1 hasta 2. En este ejemplo tenemos $\Delta x = 1$, si se modifica dicho incremento tomando un valor más pequeño, por ejemplo, $\Delta x = 0.1$, los valores cambian al igual que las expresiones algebraicas (figura 3.29).

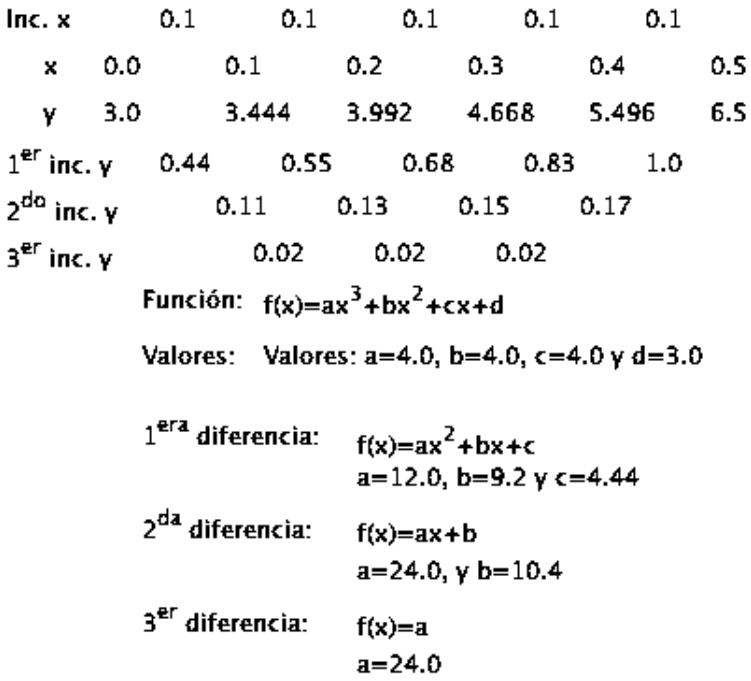

Figura 3.29: Tabulaciones con $\Delta x = 0.1$

Para ello, se pretende que se realice un análisis para concluir que las diferencias son un acercamiento a las derivadas, después de realizar los cocientes correspondientes.

El propósito de este acercamiento, es el de visualizar cómo cada una de las diferencias representa una nueva función, cada diferencia va reduciendo el grado del polinomio, además analizar que mientras menor sea el incremento de x en la razón de cambio, se obtiene un mejor acercamiento a la función derivada.

En esta sección se usan los mismos códigos que se han usado hasta ahora (códigos A.21, A.7, A.10 y A.8)

3.4.2. Función Trigonométrica

En caso de seleccionar una función trigonométrica (figura 3.30) los cálculos se hacen de la misma forma que en la sección *Razón de Cambio, Función Trigonométrica* de la página 57.

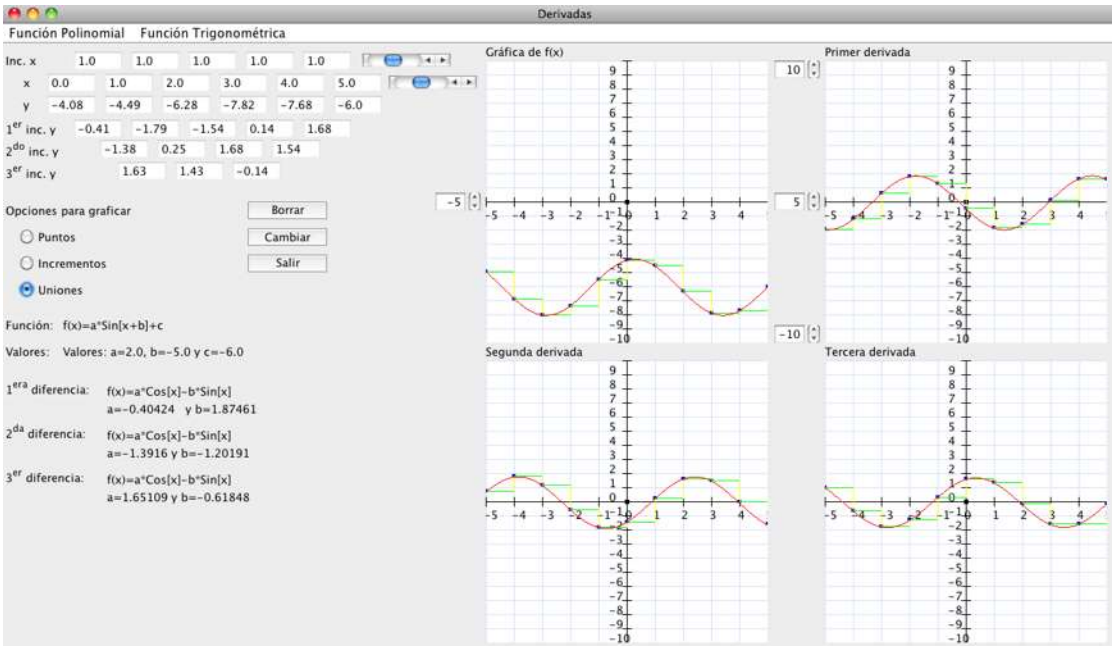

Figura 3.30: Derivadas, función trigonométrica

3.5. Integrales

Este módulo consta de dos secciones (`JMenuItem`), siendo: "Tratamiento Numérico" y "Gráfica de Integrales" (figura 3.31). Dichas secciones se describirán a continuación.

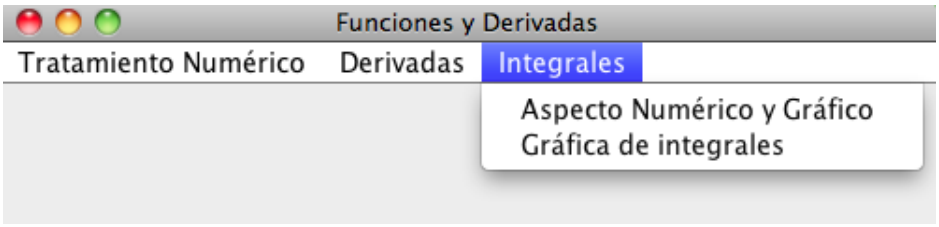

Figura 3.31: Menú "Integrales"

Al igual que en el panel de "Diferencias", tiene los siguientes elementos: JMenuBar, JMenu, JMenuItem, JTextField, JCheckBox, JButton, JPanel, JLabel, JSpinner, JSlider, JScrollBar y Canvas (figura 3.32)

Esta clase funciona igual que la clase "Derivadas", donde el usuario puede elegir funciones polinomiales o trigonométricas y tabular sus valores.

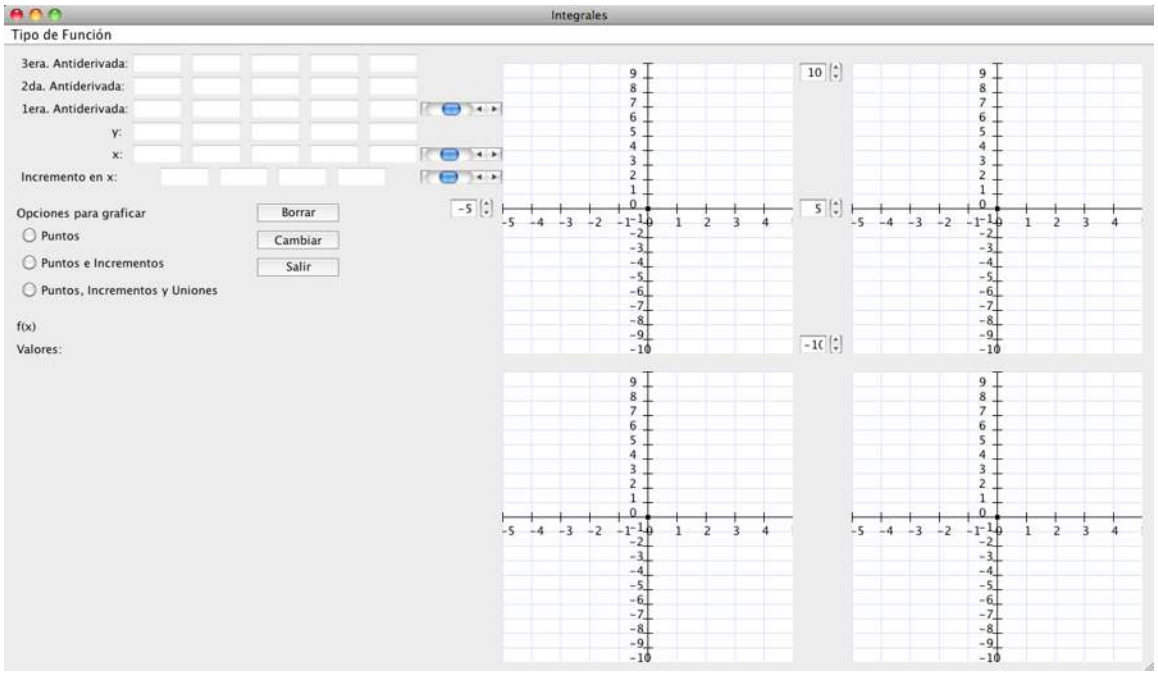

Figura 3.32: Interfaz de "Integrales"

Así mismo, se pueden modificar los valores de las tablas variando el incremento en x, o las condiciones iniciales de las antiderivadas; para que finalmente, se manden a graficar en los `Canvas`.

3.5.1. Función Polinomial

En esta sección se propone un análisis de funciones, similar a las funciones del menú "Derivadas", al mismo tiempo que se muestran las gráficas de sus antiderivadas.

En la interfaz principal se observa la organización de las tabulaciones y las gráficas; existen botones, opciones para graficar, y un menú para cambiar entre funciones polinomiales (constantes, lineales y cuadráticas) y trigonométricas (seno, coseno y tangente), la interfaz se muestra en la figura 3.33.

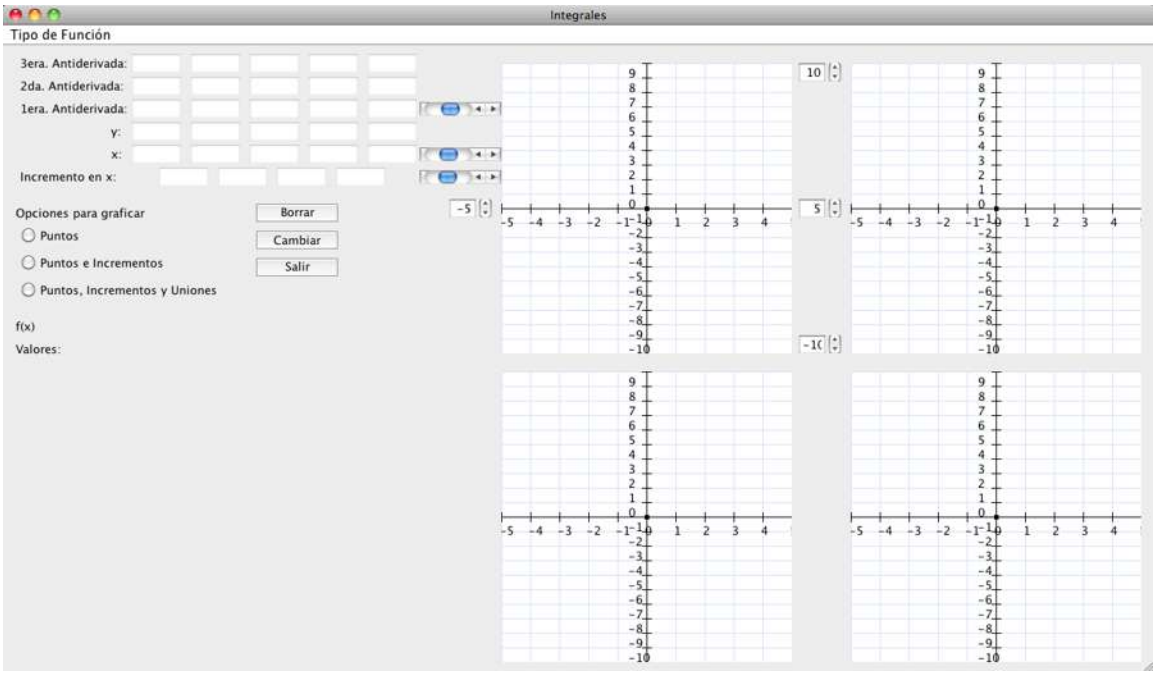

Figura 3.33: Menú "Integrales".

Por ejemplo, en una función constante $y = -2$, se muestran los valores de la tabla (x, y), los incrementos en x (Δx) y las tablas con sus antiderivadas (figura 3.34).

3era. Antiderivada:	1	2.0	4.0	5.0	3.0
2da. Antiderivada:	1	2.0	1.0	-2.0	-7.0
1era. Antiderivada:	1	-1.0	-3.0	-5.0	-7.0
y:	-2.0	-2.0	-2.0	-2.0	-2.0
x:	0.0	1.0	2.0	3.0	4.0
Incremento en x:		1.0	1.0	1.0	1.0

Figura 3.34: Tabla de datos, menú "Integrales"

Se pueden ver las gráficas de los puntos que se tabularon, así mismo se muestran los incrementos y las uniones (figura 3.35).

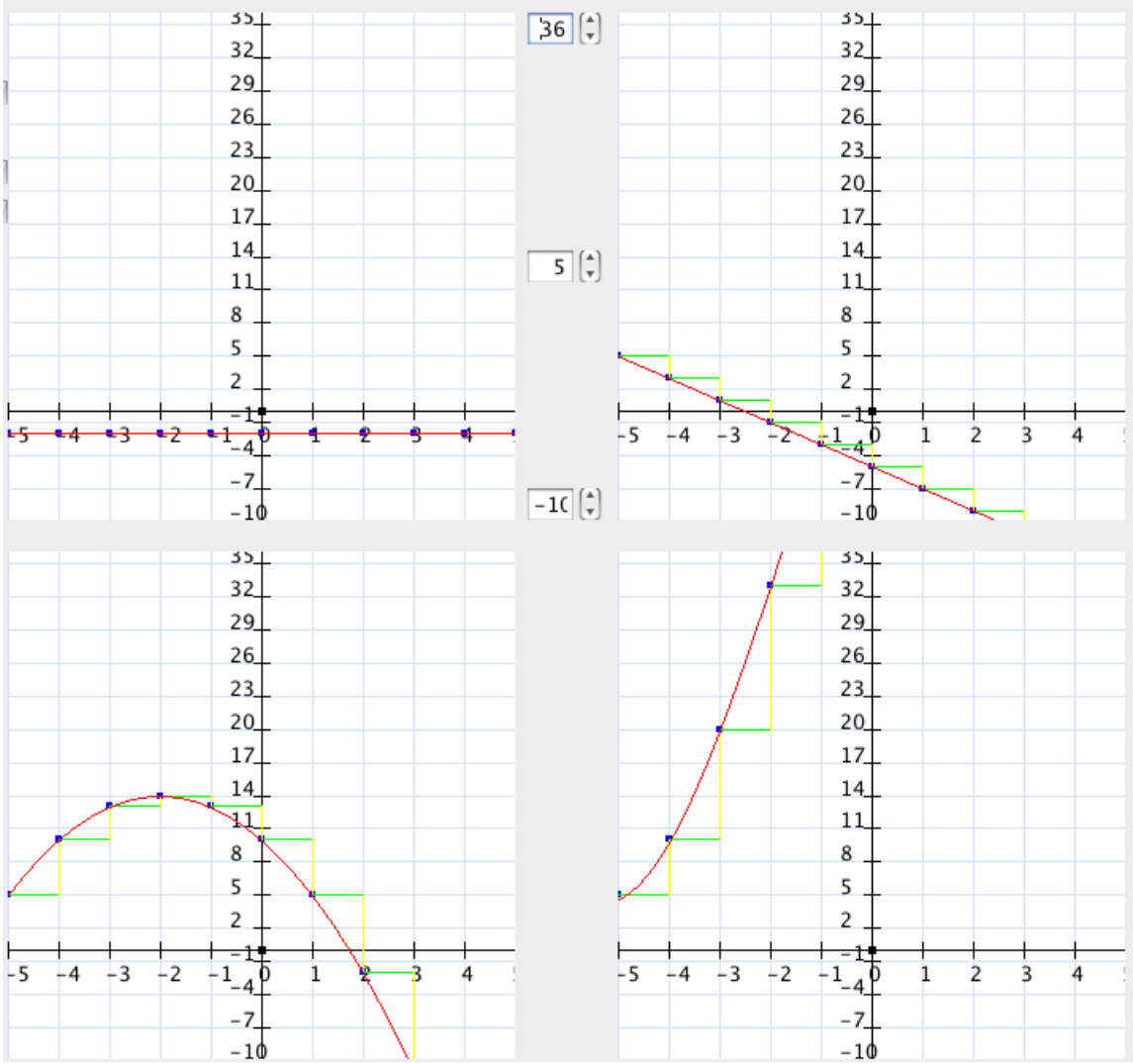

Figura 3.35: Gráficos de "Integrales" con $\Delta x = 1$

Al igual que en la sección de Derivadas, el incremento en x puede variar desde 0.1 hasta 2. En la figura 3.35 el incremento es de 1, si variamos el incremento a 0.5, las gráficas cambian (figura 3.36).

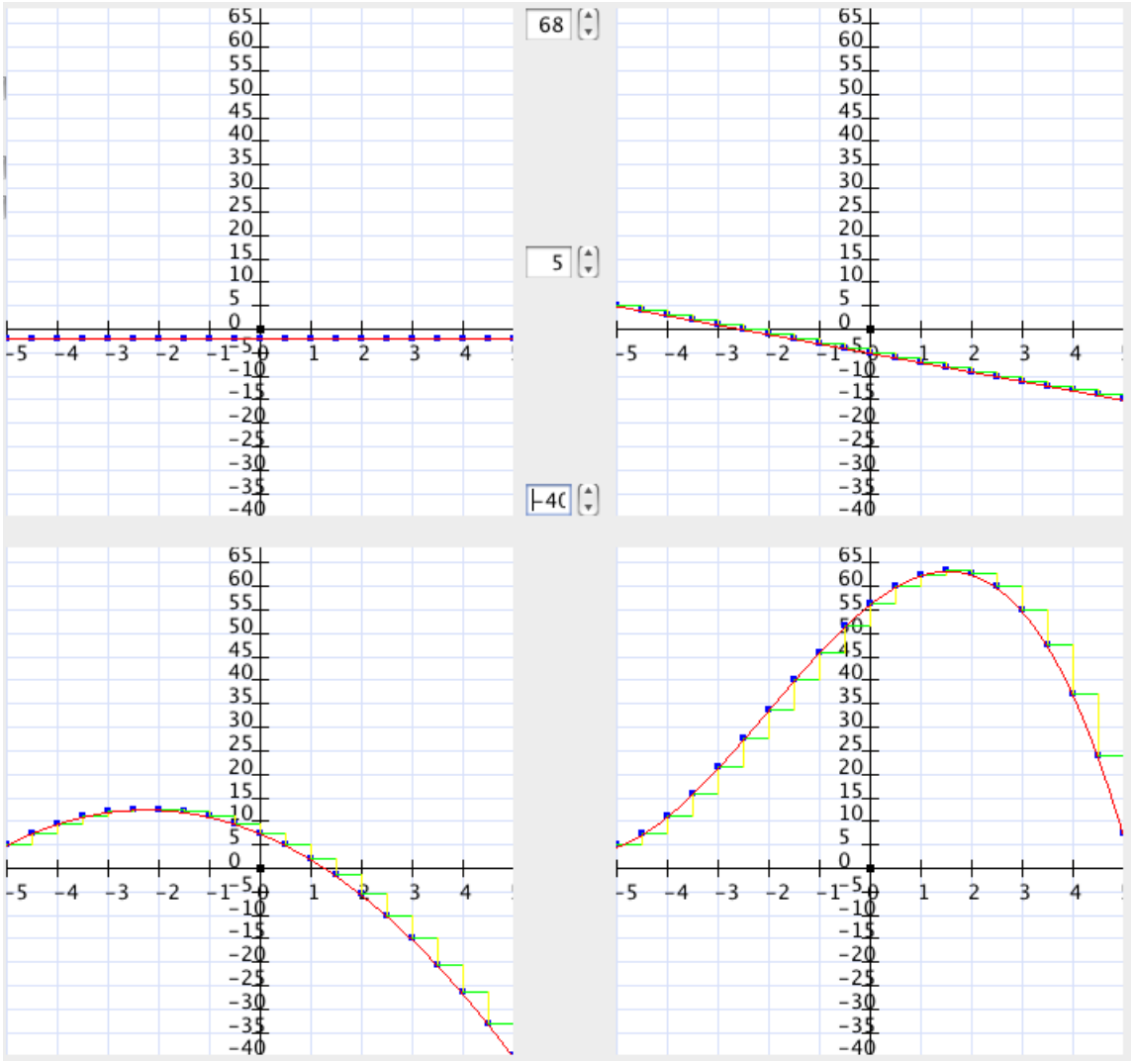

Figura 3.36: Gráfica de "Integrales" con $\Delta x = 0.5$

Nuevamente, el propósito de esta actividad es visualizar cómo se generan las antiderivadas de funciones polinomiales, y cómo se acerca cada vez más a la integral definida conforme el incremento en x se acerca a 0.

3.5.2. Función Trigonométrica

Al momento de seleccionar una función trigonométrica (figura 3.37) se hacen los cálculos correspondientes (descrito en la página 57).

1. Llenar las tabulaciones de antiderivadas (código A.26).
2. Crear la matriz y el vector de la ecuación (2.5) y resolverlas (código A.10).
3. Graficar el polinomio.
4. Convertirlas a coordenadas pixel (código A.7).

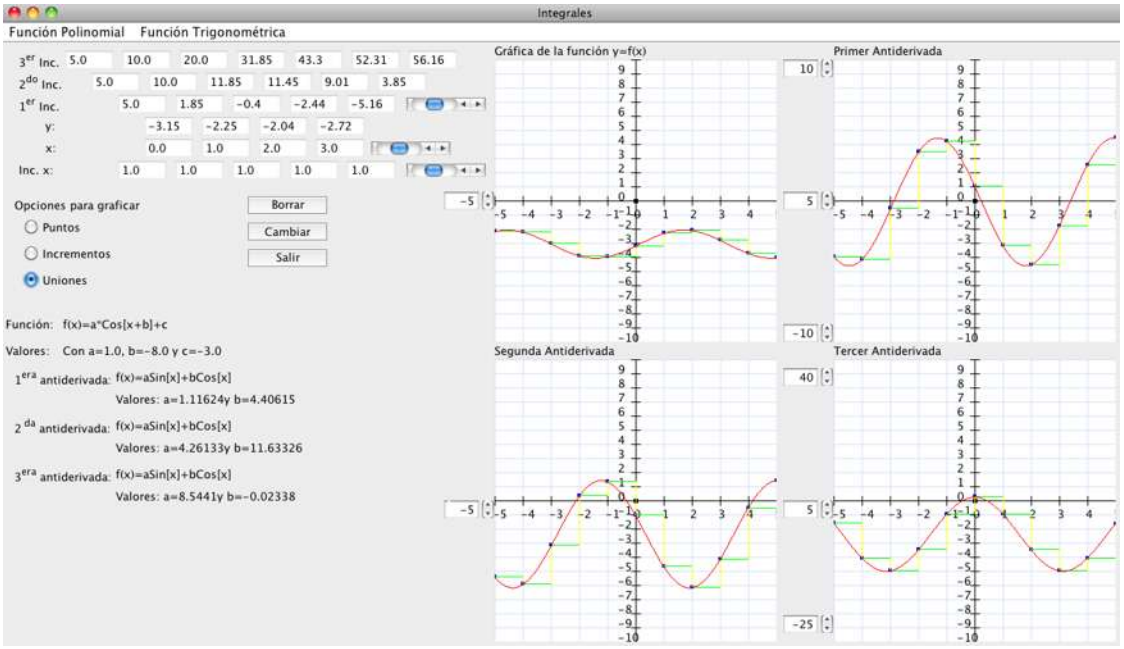

Figura 3.37: Integrales, función trigonométrica.

Capítulo 4

Conclusiones

A medida que las nuevas tecnologías abren un paradigma educacional gracias a la interacción del software, a las actividades de aprendizaje, el objeto de conocimiento, el alumno como el principal actor en la construcción de sus conocimientos y el profesor como un mediador y facilitador del aprendizaje, se han logrado crear *ambientes de aprendizaje interactivos* que abren nuevas posibilidades al proceso de enseñanza-aprendizaje. Especialmente en la enseñanza de áreas que por su naturaleza requieren de una gran interactividad, como es el caso de las matemáticas.

Para llevar a cabo la implementación de estos ambientes, es necesario evaluar varios factores, entre los cuales se encuentran los aportes de la tecnología, las estrategias metodológicas apropiadas para el uso de las mismas y la participación del docente en el proceso.

Dentro de este contexto, el software "FunDer" tiene como propósito observar los alcances y las limitaciones que tiene este programa informático, para que a través de un ambiente interactivo los estudiantes aprendan a derivar e integrar.

Podemos concluir que la implementación de ambientes interactivos de aprendizaje asistidos por la computadora, permitirá de acuerdo a las observaciones recopiladas, modificar el modelo tradicional de la enseñanza, llevándolo a un modelo más activo basado en la construcción del conocimiento, haciendo que tanto el educador como el aprendiz ejerzan un rol protagónico. En el caso del *aprendiz*, implica que éste sea un individuo creativo, analítico, capaz de pensar, razonar y abstraer, resolver problemas, diseñar, desarrollar y evaluar trabajos colaborativamente. El rol del educador

también se transforma, dejando de ser un emisor de información, para convertirse en un facilitador del conocimiento, un mediador que organice y diseñe actividades de aprendizaje significativas para el aprendiz.

Desde esta perspectiva, el proceso de enseñanza-aprendizaje es un proceso activo, producto de la interacción del aprendiz con el medio que lo rodea.

La secuencia de actividades que se presentan en el software "FunDer" permite al estudiante entender lo que es la función derivada y ejercitarse en las operaciones básicas del cálculo diferencial. Este software se ha probado de manera informal, con algunos estudiantes de bachillerato, y según lo dicho por ellos se les hace una manera agradable de aprender.

4.1. Recomendaciones

El desarrollo de los ambientes tecnológicos interactivos para el aprendizaje de las matemáticas, no se realiza con el afán de resaltar que son "mejores" métodos que los métodos tradicionales que se han venido ejecutando, sino por el contrario, lo importante es que el alumno desarrolle un aprendizaje significativo, por lo que se deben de tomar los beneficios que ofrecen tanto los métodos que implementan tecnología como de los que no, realizar una combinación de ambos y poder así generar un crecimiento en el razonamiento matemático de los estudiantes.

Como maestros, la meta debe de ir más allá de lograr que los alumnos sepan los procedimientos para calcular una derivada o una integral, es deber esforzarse para desarrollar una comprensión más avanzada que permita a los estudiantes entender a fondo los conceptos de cálculo e incluso aplicarlos.

Por ello se les sugiere:

- No enfaticen mucho en un solo tipo de representaciones, es decir la algebraica, pues se limita o inclusive se imposibilita la capacidad de razonar y el desarrollo de la memoria visual del alumno sobre lo que está haciendo, y en consecuencia, no puede dar conclusiones a lo que está realizando.

- Que implementen actividades donde se realice una exploración inicial con la tecnología, teniendo cuidado que los problemas no se vuelvan triviales. Esto

puede permitir que los estudiantes comprendan mejor el problema, entiendan exactamente lo que se pide e incluso puedan encontrar una conjetura inicial.

- Por el contrario a lo anterior, se pueden plantear actividades a los estudiantes donde al inicio se realicen los cálculos a través de papel y lápiz, y una vez resueltas, se les permita verificar los resultados con la tecnología. Esto contribuye al igual que la metodología anterior, a que los estudiantes se acostumbren a que los problemas siempre se deben revisar, validar y tratar de resolverlos de distintas formas.

El profesor debe de jugar el papel de guía y los estudiantes deben trabajar activamente en los problemas.

Apéndice A

Códigos Auxiliares

En este capítulo se presentarán los pseudo-códigos auxiliares para cada sección.

A.1. Generales

Con el siguiente método puedo crear un arreglo de coeficientes para simular un polinomio entero.

```
1  int[] crearPolinomios(int grado){
2        double[] poli=new double[grado+1];
3        while(poli[0]==0) //El primer coeficiente no sea cero.
4              poli[0]=signoAleatorio(5);
5
6        for(int i=1; i<poli.length; i++)
7              poli[i]=signoAleatorio(15);
8
9        return poli;
10 }
```

Código A.1: Método para crear polinomios

Con el siguiente método genero números aleatorios entre un rango de $[-L, L]$, con $L \in \mathbf{N}$

```
1  int signoAleatorio(int limite){
2        //Obtener un número en el rango [-limite, limite].
```

```
3       int numero=numeroAleatorio(limite);
4
5       int volado=(int)(Math.random()*10);
6       volado=volado%2;
7       int signo=(int)(Math.pow(-1, volado));
8       return (signo*numero);
9   }
```

Código A.2: Método para generar números aleatorios con signo

Con este método genero un número entero aleatorio en el rango $[0, L]$.

```
1   int numeroAleatorio(int limite){
2       //Este código regresa un número aleatorio entre [0 y Limite].
3       int cifras=cuentaDigitos(limite);
4       int potencia=(int)(Math.pow(10, cifras));
5
6       int aleatorio=(int) (Math.random()*potencia);
7       aleatorio=aleatorio%(limite+1);
8
9       return aleatorio;
10  }
```

Código A.3: Método para generar números aleatorios sin signo

Este método sirve para contar las cifras del número dado.

```
1   int cuentaDigitos(int numero){
2       //Regresa la cantidad de dígitos de 'numero'.
3       int contador=0;
4       while(numero>0){
5           numero=numero/10;
6           contador++;
7       }
8
9       return contador;
10  }
```

Código A.4: Método para contar las cifras de un número

Método para redondear un número dado a los decimales que sean necesarios.

```
1 double Redondear(double numero, int cuantos){
2       double p=Math.pow(10, cuantos);
3       numero=numero*p;
4       double tmp=Math.round(numero);
5       return tmp/p;
6 }
```

Código A.5: Método para redondear los decimales

Con el siguiente método puedo "evaluar" el arreglo de coeficientes (polinomio) en un valor dado.

```
1  double evaluarPolinomio(double[] poliRecibido, double x){
2       // Este método recibe el polinomio de la forma an+a(n-1)+...+a1+a0
3       int longitud=poliRecibido.length;
4
5       //Esta parte voltea el polinomio, para que sea más fácil evaluarlo
        .
6       double[] Polinomio=new double[longitud];
7       for(int i=0; i<longitud; i++)
8            Polinomio[longitud-1-i]=poliRecibido[i];
9
10      double resultado=0, auxiliar=1;
11      double coeficiente, temporal;
12
13      //Aquí se hace la evaluación.
14      for(int contador=0; contador<Polinomio.length; contador++){
15           coeficiente=Polinomio[contador];
16           temporal=coeficiente*auxiliar;
17           resultado=resultado+temporal;
18           auxiliar=auxiliar*x;
19      }
20
21      return Redondear(resultado, 3);
22 }
```

Código A.6: Método para evaluar polinomios

El siguiente método sirve para poder graficar de forma adecuada una función continua. Recibe un arreglo de puntos `double[][] original` con las coordenadas cartesianas, y las convierte en coordenadas pixel.

```
1  int[][] cambiandoaPixel(double[][] original, int tamano){
2      //Obtener los valores del plano.
3      int xinf, xsup, yinf, ysup;
4
5      int longitud=original.length;
6      int[][] pixel=new double[longitud][2];
7      for(int i=0; i<longitud; i++){
8          pixel[i][0]=((original[i][0]-xinf)/(xsup-xinf)*(tamano));
9          pixel[i][1]=((original[i][1]-yinf)/(ysup-yinf)*(tamano));
10     }
11
12     return pixel;
13 }
```

Código A.7: Método para cambiar a coordenadas pixel

El siguiente método recibe un arreglo de coeficientes (polinomio) para poder crear la gráfica de esa función.

```
1  int[] creacionContinua(double[] polinomio, int tamano){
2      int[] continuos=new int[tamano];
3      int ys;
4      for(int i=0; i<continuos.length; i++){
5          ys=pixelaCartesiana(polinomio, i),
6          continuos[i]=ys;
7      }
8
9      return continuos;
10 }
```

Código A.8: Método para crear la gráfica de una función

Con el siguiente método convierto coordenadas cartesianas a coordenadas pixel.

```
1  int pixelaCartesiana(double[] polinomio, int xpixel){
2       int ancho, alto;
3       double xreal, yreal;
4       int ypixel;
5       double ymin, ymax, xmin, xmax;
6
7       // Cartesianas equivalentes al punto de la pantalla
8       xreal = (xpixel * (xmax-xmin) / (ancho-1)) + xmin;
9       // Calculamos el valor de ese punto
10      yreal = Auxiliares.evaluarPolinomio(polinomio, xreal);
11      // Escalamos la coordenada 'ypantalla' dentro de los limites de la
        ventana
12      ypixel = (int)( (yreal-ymin) * (alto-1) / (ymax-ymin) );
13      // Reconvertinos el xpixel cartesiano a punto de pantalla
14      ypixel = alto - ypixel;
15
16      return ypixel;
17 }
```

Código A.9: Método para convertir coordenadas.

Este método recibe un arreglo de puntos, y crea un "polinomio" que pase por esos puntos.

```
1  double[] gaussJordan(double[] polinomio, double paso){
2       double aux1, aux2, auxr, d, c;
3
4       int cuantasIncognitas=1;
5       if(polinomio.length!=1)
6            cuantasIncognitas=polinomio.length-1;
7
8       double[][] m=new double[cuantasIncognitas][cuantasIncognitas];
9       double[] r=new double[cuantasIncognitas];
10
11      //Aquí es el llenado de datos.
12      for(int i=0; i<cuantasIncognitas; i++){
13           aux2=evaluarPolinomio(polinomio, (i+1)*paso);
14           aux1=evaluarPolinomio(polinomio, i*paso);
```

```
15            auxr=(aux2-aux1)/paso;
16            r[i]=auxr;
17            for(int j=0; j<cuantasIncognitas; j++){
18                  m[i][j]=Math.pow(i*paso, j);
19            }
20      }
21
22      //Empezamos la reducción de la matriz.
23      for(int i=0; i<cuantasIncognitas; i++){
24            d=m[i][j];
25            for(int s=0; s<cuantasIncognitas; s++){
26                  m[i][s]=((m[i][s])/d);
27            }
28            r[i]=(r[i])/d;
29            for(int x=0; x<cuantasIncognitas; x++){
30                  if(i!=x){
31                        c=m[x][i];
32                        for(int y=0; y<cuantasIncognitas; y++){
33                              m[x][y]=m[x][y]-c*m[i][y];
34                        }
35                        r[x]=r[x]-c*r[i];
36                  }
37            }
38      }
39
40      return r;
41  }
```

Código A.10: Método para interpolar un conjunto de puntos

A.2. Progresiones Aritméticas

Con los siguientes códigos se hace el llenado de los datos en la sección de "Progresiones Aritméticas".

Durante toda la sección, los elementos de "Posición" y de "Valor" son arreglos de enteros:

```
1 int[] posicion=new int[8];
2 int[] valor=new int[8];
```

Código A.11: Variables Globales para Progresiones Ariméticas

Nivel 1.

La diferencia entre los elementos de "Posición" será 1, y las diferencias entre los elementos de "Valor" será un entero entre 0 y 10.

```
1  //Llenar las casillas de Posición.
2  for(int i=0; i<8; i++){
3        posicion[i]=i+1;
4  }
5
6  //Valor
7  int diferencia=numeroAleatorio(10);
8  valor[0]=numeroAleatorio(10);
9  for(int i=1; i<8; i++){
10       valor[i]=valor[i-1]+diferencia;
11 }
```

Código A.12: Progresiones Aritméticas, Nivel 1.

Nivel 2.

La diferencia entre los elementos de "Posición" es una constante mayor a 1, y los elementos de "Valor" siguen con una diferencia constante entre 0 y 10.

```
1  //Posiciones.
2  posicion[0]=1;
3  int incrementoPosicion=numeroAleatorio(10);
4  for(int i=1; i<8; i++){
5        posicion[i]=posicion[i-1]+incrementoPosicion;
6  }
7
8  //Valores.
9  valor[0]=numeroAleatorio(10);
10 int diferenciaValor=numeroAleatorio(10);
11 for(int i=1; i<8; i++){
12       valor[i]=incrementoPos*incrementoValor+valor[i-1];
13 }
```

Código A.13: Progresiones Aritméticas, Nivel 2.

Nivel 3.

La diferencia entre los elementos de "Posición" y de "Valor" no son constantes.

```
1  int k=numeroAleatorio(10);
2
3  //Posiciones.
4  posicion[0]=1;
5  for(int i=1; i<8; i++){
6        posicion[i]=posicion[i-1]+numeroAleatorio(10);
7  }
8
9  //Valores.
10 valor[0]=numeroAleatorio(10);
11 for(int i=1; i<8; i++){
12       valor[i]=k*(posicion[i]-posicion[0])+valor[0];
13 }
```

Código A.14: Progresiones Aritméticas, Nivel 3.

Nivel 4.

En este nivel la posición se presenta con diferencias distintas de 1 y no constantes, los datos correspondientes al valor aparecen en distintas posiciones.

```
1  int k=numeroAleatorio(10);
2
3  //Iniciales.
4  posicion[0]=1;
5  valor[0]=numeroAleatorio(10);
6
7  //Los demás.
8  for(int i=1; i<8; i++){
9        posicion[i]=posicion[i-1]+numeroAleatorio(10);
10       valor[i]=k*(posicion[i]-posicion[0])+valor[0];
11 }
12
13 //Volados.
14 int volado=(int)(Math.random());
15 if(volado==0)
16       //Muestro en pantalla valor[0]
17 else
18       //Muestro en pantalla valor[1]
19
20 //Repetir para los pares {valor[2], valor[3]}, {valor[4], valor[5]} y {
       valor[6], valor[7]}.
```

Código A.15: Progresiones Aritméticas, Nivel 4.

A.3. Incrementos

Códigos de apoyo para la sección (3.3.2) de Incrementos.

Los arreglos son:

```
deltaPosicion=new JTextField[7];
posicion=new JTextField[8];
valor=new JTextField[8];
deltaValor=new JTextField[7];

//Requeriremos un arreglo de enteros para simular un polinomio:
elPolinomio=crearPolinomio(1);
```

Código A.16: Variables globales de la sección Incrementos

Niveles 1 y 2

```
void niveles12(int nivel){
    int diferencia;

    if(nivel==1)
        diferencia=1;
    else
        diferencia=numeroAleatorio(10);

    //Para llenar los Incrementos Posición.
    for(int i=0; i<7; i++){
        deltaPosicion[i]=diferencia;
    }

    //Para llenar la Posición y el Valor.
    posicion[0]=1;
    valor[0]=evaluarPolinomio(elPolinomio, 1);
    for(int i=1; i<8; i++){
        posicion[i]=posicion[i-1]+diferencia;
        valor[i]=evaluarPolinomio(elPolinomio, posicion[i]);
    }

    //Para llenar Incremento Valor.
```

```
23        for(int i=0; i<7; i++){
24              deltaValor[i]=valor[i+1]-valor[i];
25        }
26  }
```

Código A.17: Incrementos, Niveles 1 y 2

Niveles 3 y 4

```
1   void niveles34(int nivel){
2         int diferencia;
3
4         //Posiciones.
5         posicion[0]=1;
6         for(int i=1; i<8; i++){
7               diferencia=numeroAleatorio(10);
8               posicion[i]=posicion[i-1]+diferencia;
9         }
10
11        //Incremento Posición.
12        for(int i=0; i<7; i++)
13              deltaPosicion[i]=posicion[i+1]-posicion[i-1];
14
15        //Incremento Valor.
16        for(int i=0; i<7; i++){
17              int cte=elPolinomio[0];
18              int delta=deltaPosicion[i];
19              deltaValor[i]=cte*delta;
20        }
21
22        if(nivel==3){ //Para el nivel 3.
23              for(int i=0; i<8; i++)
24                    valores[i]=evaluarPolinomio(elPolinomio, posicion[i]);
25        }
26        else //Para el nivel 4.
27              //Se repiten los volados como en el código (A.16)
28  }
```

Código A.18: Incrementos, Niveles 3 y 4

A.4. Razón de Cambio

Métodos de apoyo para la sección 3.3.3.

Necesitaremos varios arreglos de `JTextField` para guardar los datos:

```
1 JTextField incrementosX[], incrementosY[];
2 JTextField cajaX[], cajaY[];
3 JTextField xRazon[], yRazon[];
4 int elPolinomio[];
```

Código A.19: Variables globales, Razón de Cambio

Método para crear la tabulación $(x, f(x))$.

```
1  double[][] ys(){
2      //Obtener los límites de los Spinners
3      int xinf, xsup;
4
5      int longitud=(xsup-xinf)/paso;
6      double[][] puntosReales=new double[longitud][2];
7
8      double x=xinf, y;
9      for(int i=0; i<longitud; i++){
10         y=evaluarPolinomio(elPolinomio, x);
11         puntosReales[i][0]=x;
12         puntosReales[i][1]=y;
13         x+=paso;
14     }
15
16     return puntosReales;
17 }
```

Código A.20: Método para crear $(x, f(x))$

Crear tabulación con las razones de cambio.

```
1 double[][] razonesReales(double[][] originales){
2     int longitud=originales.length-1;
3     double[][] lasRazones=new double[longitud][2];
4
5     double dy, dx;
```

```
6       for(int i=0; i<longitud; i++){
7           dy=originales[i+1][1]-originales[i][1];
8           dx=originales[i+1][0]-originales[i][0];
9
10          lasRazones[i][0]=originales[i][0];
11          lasRazones=Redondear(dy/dx, 1);
12      }
13
14      return lasRazones;
15  }
```

Código A.21: Método para obtener las razones de cambio

A.5. Apoyo Trigonométrico

En esta sección se escribirán los pseudo-códigos para los métodos necesarios para evaluar, graficar e interpolar funciones trigonométricas.

```
1   double evaluarTrigo(int opcion, double[] unPolinomio, double x0){
2       double valor;
3       double a=unPolinomio[0];
4       double b=unPolinomio[1];
5       double c=unPolinomio[2];
6
7       if(opcion==1)
8           valor=a*Math.sin(x0+b)+c;
9       else if(opcion==2)
10          valor=a*Math.cos(x0+b)+c;
11      else
12          valor=a*Math.tan(x0+b)+c;
13
14      return valor;
15  }
```

Código A.22: Método para evaluar un polinomio trigonométrico

A.6. Interpolación por Mínimos Cuadrados

Con este código guardo, y evalúo, las bases trigonométricas para la interpolación.

```
1 double baseTrigo(int opcion, double valor){
2       if(opcion==0)
3             return Math.cos(valor);
4       else
5             return -1*Math.sin(valor);
6 }
```

Código A.23: Bases trigonométricas

Creamos una matriz según la ecuación (2.5).

```
1  double[][] crearMatriz(double[][] puntos){
2       double tmp, suma;
3       double matriz[][]=new double[2][2];
4
5       for(int i=0; i<2; i++){
6             for(int j=0; j<2; j++){
7                   suma=0;
8                   for(int k=0; k<puntos.length; k++){
9                   tmp=baseTrigo(i,puntos[k][0])*baseTrigo(j,puntos[k][0]);
10                  suma+=tmp;
11                  }
12            matriz[i][j]=suma;
13            }
14      }
15      return matriz;
16 }
```

Código A.24: Método para crear matriz

Creamos un vector según la ecuación (2.5).

```
1 double[] crearVector(double[][] puntos){
2       double tmp, suma;
3       double[] vector=new double[2];
4       for(int i=0; i<2; i++){
5             suma=0;
6             for(int k=0; k<puntos.length; k++){
```

```
7                   tmp=puntos[k][1]*basesTrigonometricas(i, puntos[k][0]);
8                   suma+=tmp;
9               }
10              vector[i]=suma;
11          }
12
13          return vector;
14      }
```

Código A.25: Método para crear un vector

A.7. Integrales

Método para crear una tabla de antiderivadas.

```
1   double[][] crearAntiderivadas(int condicionInicial, double[][] viejo){
2       int xinf=0;
3       int xsup=10;
4
5       int longitud=(xsup-xinf)/paso;
6       double[][] nuevo=new double[longitud][2];
7
8       nuevo[0][0]=0;
9       nuevo[0][1]=condicionInicial;
10
11      for(int i=1; i<nuevo.length; i++){
12          nuevo[i][0]=viejo[i][0];
13          nuevo[i][1]=paso*viejo[i-1][1]+nuevo[i-1][1];
14      }
15
16      return nuevo;
17  }
```

Código A.26: Método para crear antiderivadas

Referencias

[1] Agarwal, R. (2005). *Software Development vs Software Maintenance.* Consultado el 12 de febrero, 2013 en

`http://www.irahul.com/2005/10/software-development-vs-software.html`

[2] Ariza, M. (2009); *Noesis, semiosis y matemáticas (2009).* Consultado el lunes 14 de enero, 2013

`http://www.academia.edu/413771/Noesis_semiosis_y_matematicas`

[3] Braude, E. J. (2007); *Ingeniería del Software, una persepectiva orientada a objetos.* México: Alfaomega.

[4] Calderon, A. (2010). *Procesos de valicación en geometría de estudiantes de bachillerato: favoreciendo la estructura en la demostración geométrica.* Tesis de licenciatura.

[5] Ceballos, F. (2010) *Java 2. Interfaces Gráficas y Aplicaciones para Internet.* 978-84-7897-859-5. Ra-Ma Editorial.

[6] Cortés & Hitt (2005); *Reflexiones Sobre el Aprendizaje del Cálculo y su Enseñanza.*

[7] Cortés, J. (2010); *Uso de Tecnología en Educación Matemática. Investigaciones y Propuestas.*

[8] Deitel. P, (2008), *Java, cómo programar.* 978-01-3222-220-4. Prentice Hall.

[9] Duval, R. (1993). *Registros de representación semiótica y funcionamiento cognitivo del pensamiento*. En Hitt, F(Ed), Investigaciones en Matemática Educativa II (p.113-201). Grupo editorial Iberoamericana, México.

[10] Duval, R. (1999). *Los problemas fundamentales en el aprendizaje de las matemáticas y las formas superiores del desarrollo cognitivo*. Santiago de Cali, Colombia: Universidad del Valle.

[11] Duvinsky, E. (1996). *Aplicación de la perspectiva piagetiana a la educación matemática universitaria*. Educación Matemática. 8(3), 24-40.

[12] Ferreiro G., Calderon E. (2001); *El ABC del aprendizaje cooperativo. Trabajo en equipo para enseñar y aprender*. Ed. Trillas, México.

[13] García, L. (2005). *Métodos Numéricos con Mathematica*. 978-97-0150-977-7. Alfaomega.

[14] Guerrero, L. (2009); *Desarrollo de Software para Promover el Aprendizaje de la Demostración en Geometría.* Investigaciones y Propuestas sobre el uso de la Tecnología en la Educación Matemática, Vol III.

[15] Guerrero, L., Morales C. (2010); *Diseño de Software Educativo: Construcción de Algoritmos para el Manejo de Gráficas y Construcciones Geométricas Interactivas..*

[16] Guillén, B. (2012). *Diseño y Desarrollo de Software Educativo*. Tesis de licenciatura.

[17] Hitt, F., Cortés, C. (2012). *Formation a la recherche en didactique des mathématiques.* 978-2-923565-54-5.

[18] Jaramillo, Castellanos, Castañeda, Ordoñez (2005); *Informática todo un reto. Ambientes de aprendizaje en el aula. ¿Fomentan el manejo de la información?.* Consultado el martes 12 de febrero, 2013 en

`http://redalyc.uaemex.mx/pdf/834/83412219011.pdf`

[19] López A. (2007). *Introducción al Desarrollo de Programas con Java*. 978-97-0324-317-4. UNAM.

[20] López A. (2008); *Programación de un ambiente interactivo tecnológico de aprendizaje para el tema de fracciones.* Tesis de licenciatura.

[21] Manber, U. (1990). *Introduction to Algorithms. A Creative Approach*. 978-02-0112-037-0. Addison-Wesley.

[22] Nieves, A., Dominguez, F. (2002). *Métodos Numéricos Aplicados a la Ingeniería*. 978-60-7438-317-1. Patria.

[23] Núñez, G. (2008). *Ambientes tecnológicos interactivos para el aprendizaje de las matemáticas*. Tesis doctoral.

[24] Núñez, G., Cortés, C. (2001). *Desarrollo de Ambientes Tecnológicos Interactivos para el Aprendizaje de las Matemáticas: Una Experiencia con la Línea Recta*. AMIUTEM, colección Uso de Tecnología en Educación Matemática.

[25] Suárez, M. *Introducción a la Teoría de la Aproximación*. Notas de curso de "Métodos Numéricos" 2010.