UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERIA

DESARROLLO DE UN SIMULADOR PARA FISICA I DE LA UNIVERSIDAD DON BOSCO

PROYECTO DE GRADUACIÓN PARA OPTAR AL GRADO DE INGENIERO EN CIENCIAS DE LA COMPUTACIÓN

PRESENTADO POR:

CASTRO PADILLA, CARLOS JOSÉ

MAZARIEGO QUINTEROS, GLADIS MARGARITA

OCTUBRE 2005

SAN SALVADOR, EL SALVADOR C.A.

UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERIA

Ing. Federico Miguel Huguet Rivera

Rector

Presbítero Víctor Bermúdez Yanes

Vicerrector Académico

Lic. Mario Rafael Olmos

Secretario General

Ing. Ernesto Godofredo Girón

Decano Facultad de Ingeniería

UNIVERSIDAD DON BOSCO

FACULTAD DE INGENIERIA

DESARROLLO DE UN SIMULADOR PARA FISICA I
DE LA UNIVERSIDAD DON BOSCO

Ing. Guillermo Valencia	**Ing. Alfredo Méndez**	**Ing. Rudy Torres**
Jurado 1	**Jurado 2**	**Jurado 3**

Ing. Jaime Anaya	**Ing. Raúl Martínez**
Asesor	**Tutor**

INDICE

INTRODUCCIÓN i

CAPÍTULO I
PLANTEAMIENTO DEL PROBLEMA

1.1 ANTECEDENTES 1

1.2 IMPORTANCIA DE LA INVESTIGACIÓN 3

1.2.1 PLANTEAMIENTO DEL PROBLEMA 3
1.2.2 DEFINICIÓN DEL TEMA 4
1.2.3 JUSTIFICACIÓN 4

1.3 OBJETIVOS 5

1.3.1 GENERAL 5
1.3.2 ESPECIFICOS 5

1.4 ALCANCES 6

1.5 LIMITACIONES 7

1.6 PROYECCIÓN SOCIAL 7

1.7 PLAN DE SOLUCIÓN 8

CAPÍTULO II
FACTIBILIDADES DEL PROYECTO

2.1 FACTIBILIDADES DEL PROYECTO 10

2.1.1 FACTIBILIDAD TÉCNICA 10
2.1.2 FACTIBILIDAD ECONÓMICA 10
2.1.2.1 PRESUPUESTO DE DESARROLLO 10
2.1.2.2 PRESUPUESTO DE IMPLEMENTACIÓN 11

CAPÍTULO III
METODOLOGÍAS

3.1 METODOLOGÍAS DE LA INVESTIGACIÓN 12

3.1.1 OBJETIVO DE LA INVESTIGACIÓN 12
3.1.2 TÉCNICAS DE LA INVESTIGACIÓN 13
3.1.2.1 FUENTES 13
3.1.2.2 TÉCNICAS 13
3.1.2.3 DEFINICIÓN DE LA POBLACIÓN 14

3.2 METODOLOGÍA DE DESARROLLO 14

3.2.1 HERRAMIENTAS DEL ANÁLISIS Y DISEÑO 14
3.2.2 HERRAMIENTAS DE DESARROLLO 15
3.2.2.1 MODELO DE ENSAMBLAJE DE COMPONENTES 15
3.2.2.2 PROGRAMACIÓN ORIENTADA A OBJETOS 15
3.2.2.2.1 VISUAL BASIC .NET 16
3.2.2.2.2 BASE DE DATOS ACCESS 16

CAPÍTULO IV
MARCO TEÓRICO

4.1 REFERENCIA HISTÓRICA 18

4.2 MARCO CONCEPTUAL 20

4.2.1 EL PROCESO DEL SOFTWARE 20
4.2.1.1 MODELOS DE PROCESOS EVOLUTIVOS DE SOFTWARE 20
4.2.1.1.1 EL MODELO EN ESPIRAL 21
4.2.1.1.2 EL MODELO DE ENSAMBLAJE DE COMPONENTES 23
4.2.2 MODELADO DEL ANÁLISIS 24
4.2.2.1 MODELADO FUNCIONAL Y FLUJO DE INFORMACIÓN 24
4.2.2.1.1 DIAGRAMA DE FLUJO DE DATOS 24
4.2.2.1.1.1 CREACIÓN DE UN MODELO DE FLUJO DE DATOS 26
4.2.3 MODELADO DEL DISEÑO 26
4.2.3.1 MODELADO DE DATOS 26
4.2.3.1.1 OBJETO DE DATOS, ATRIBUTOS, RELACIONES 27
4.2.3.1.2 CARDINALIDAD Y MODALIDAD 28
4.2.3.1.3 DIAGRAMA ENTIDAD – RELACIÓN 29
4.2.3.1.3.1 CREACIÓN DE UN DIAGRAMA ENTIDAD – RELACIÓN 30

4.2.4 TECNOLOGÍA .NET 30
4.2.4.1 COMMON LANGUAGE RUNTIME 32
4.2.4.2 BIBLIOTECA DE CLASES 32
4.2.4.3 ENSAMBLADOS 34
4.2.4.4 VENTAJAS DE .NET 34
4.2.4.5 ¿QUÉ ES EL .NET FRAMEWORK? 36

CAPÍTULO V
DESARROLLO DEL SISTEMA

5.1 CICLO DE VIDA 37

5.2 ANÁLISIS Y DISEÑO 38

5.2.1 ANÁLISIS 38
5.2.1.1 IDENTIFICACIÓN DE REQUISITOS 43
5.2.1.1.1 DIAGRAMAS JERÁRQUICOS 44
5.2.1.1.1.1 DIAGRAMA JERÁRQUICO GENERAL 44
5.2.1.1.1.2 DIAGRAMA JERÁRQUICO DE CONTENIDO TEÓRICO 44
5.2.1.1.1.3 DIAGRAMA JERÁRQUICO DE SIMULACIÓN 45
5.2.1.1.1.4 DIAGRAMA JEPARQUICO DE EVALUADOR 45
5.2.1.1.1.5 DIAGRAMA JERÁRQUICO DE ÍNDICE CONCEPTUAL 46
5.2.1.1.2 DIAGRAMAS DE FLUJO 46
5.2.1.1.2.1 DFD A NIVEL DE CONTEXTO 46
5.2.1.1.2.2 DFD DE NIVEL UNO 47
5.2.1.1.2.3 DFD A NIVEL 2, PROCESO SELECCIÓN DE MÓDULO 47
5.2.1.1.2.4 DFD A NIVEL 3, PROCESO REALIZA PRUEBA 48
5.2.2 DISEÑO 48
5.2.2.1 DISEÑO DE DATOS 48
5.2.2.1.1 DIAGRAMA ENTIDAD – RELACIÓN 50
5.2.2.2 COMPORTAMIENTO DE MÓDULOS DE LA APLICACIÓN 50
5.2.2.2.1 CONTENIDO TEÓRICO 50
5.2.2.2.2 SIMULACIÓN 51
5.2.2.2.3 EVALUACIÓN TEÓRICA 57
5.2.2.2.4 INDICE CONCEPTUAL 58
5.2.2.3 FLUJOGRAMAS 58
5.2.2.3.1 FLUJOGRAMA PARA SIMULACIÓN DE VECTORES 59

5.2.2.3.2 FLUJOGRAMA PARA SIMULACIONES DE MOVIMIENTO HORIZONTAL, VERTICAL, PARABÓLICO, CIRCULAR Y ROTACIÓN. 60
5.2.2.3.3 FLUJOGRAMA PARA SIMULACIÓN DE FUERZA Y MOVIMIENTO 61
5.2.2.3.4 FLUJOGRAMA PARA SIMULACIÓN DE TRABAJO Y ENERGÍA 62
5.2.2.3.5 FLUJOGRAMA PARA SIMULACIÓN DE COLISIONES 63
5.2.2.3.6 FLUJOGRAMA PARA PROGRAMA EVALUADOR 64
5.2.2.4 DISEÑO DE INTERFAZ 65
5.2.2.4.1 DISEÑO DE MENU 65
5.2.2.4.2 INTERFACES DEL SISTEMA 66

CONCLUSIONES ..... i.i
FUENTES DE INFORMACIÓN iii
GLOSARIO iv
ANEXOS

INTRODUCCIÓN

La simulación digital es aquella que está basada en la ejecución de un programa en una computadora logrando una reproducción artificial de un fenómeno; generalmente esto ocurre cuando la operación de un sistema o la experimentación en él son imposibles, costosas, peligrosas o poco prácticas.

La simulación digital es sin duda alguna un avance sorprendente de la tecnología y uno de los campos beneficiados con este tipo de avances es la pedagogía, la cual ha sido enriquecida con simuladores que proporcionan una enseñanza práctica y de bajo costo, compuesto por mundos ideales creados a partir de la Realidad Virtual.

Este proyecto presenta la creación de una herramienta innovadora de estudio por medio del Análisis y Desarrollo de simulador para la cátedra de Física I, basado en la planificación docente de la Universidad Don Bosco.

De esta forma, el proyecto se ha estructurado por capítulos, los cuales se constituyeron en base a los objetivos trazados para el mismo.

CAPITULO I: PLANTEAMIENTO DEL PROBLEMA

Presenta los antecedentes y planteamiento del problema. Incluye además la justificación del trabajo, mencionando a su vez los objetivos que se desean lograr; se detallan los alcances y limitaciones. Además, contiene un plan de solución para dicho problema.

CAPITULO II: FACTIBILIDADES DEL PROYECTO

Este capítulo detalla la factibilidad técnica para el desarrollo del proyecto, muestra además un presupuesto de desarrollo e implementación del mismo.

CAPITULO III: METODOLOGÍAS

Describe la metodología utilizada en el proyecto, técnicas y recursos que proporcionan información que fundamentan la parte operativa del mismo, así como también la metodología de desarrollo, describiendo las herramientas empleadas para el diseño y desarrollo de éste.

CAPITULO IV: MARCO TEÓRICO

En este capítulo se ha plasmado la información teórica en que se basa el proyecto para conseguir su desarrollo, contemplando dentro de ésta, el análisis y diseño de proyectos similares.

CAPITULO V: DESARROLLO DEL SISTEMA

Contiene el análisis e identificación de requisitos, objetos de datos y funciones. Utilizando las metodologías descritas en el Capítulo III.

Luego se presenta las conclusiones y recomendaciones del proyecto, también las fuentes de información que fueron utilizadas como apoyo para el desarrollo del proyecto. Se finaliza con los anexos, que brindan información adicional.

CAPÍTULO I
PLANTEAMIENTO DEL PROBLEMA

1.1 ANTECEDENTES

La física puede definirse como una ciencia que investiga conceptos fundamentales de la materia, energía, espacio y las relaciones entre ellos. Ésta a su vez es una de las cátedras más importantes en el comienzo de toda carrera de ingeniería, no sólo para la Universidad Don Bosco, sino para el resto de centros educativos de El Salvador. La Física I trata sobre los fundamentos de la mecánica de la partícula y del cuerpo rígido y sus aplicaciones prácticas.

El estudio de la materia comprende la asimilación de conceptos del tema, seguido por la resolución de ejemplos y discusión de los mismos. Las gráficas, dibujos y tablas con frecuencia son incluidas en el material de apoyo ya que son de gran utilidad para la comprensión de los fenómenos físicos. Dicho estudio se complementa con la aplicación de los conceptos en el laboratorio práctico que el alumno visita regularmente (de 8 a 12 veces por ciclo) en el que estará guiado por un instructor designado para la materia, utilizando para esto, material especializado, que en la mayoría de ocasiones, es de poca o nula accesibilidad al alumno fuera de la Universidad. Esta última actividad resulta de gran utilidad para el entendimiento de la materia.

En la actualidad, existen muchos libros de textos que ayudan al alumno a simplificar el aprendizaje de dicha materia, sin embargo, se sabe que esto no es suficiente para su total comprensión, por lo que queda a criterio del alumno traspasar las fronteras de la Universidad y bibliotecas, para descubrir de manera particular, nuevas y diferentes metodologías de aprendizaje.

Es en esto último, en lo que se enfoca el máximo esfuerzo para desarrollar una tecnología que ofrezca una herramienta de apoyo tanto para docentes como para alumnos y que sirva de base para un nuevo concepto de enseñanza y que a su vez sea eficiente y atractivo.

Esta tecnología nos ofrecerá, en las ocasiones que lo requiera, un espacio ideal que reforzará la enseñanza de la materia concreta que estamos estudiando, utilizando medios visuales y dinámicos ayudando a la capacidad de retener lo que vemos y hacemos.

Actualmente, dicha tecnología ya ha sido implementada en algunas universidades de El Salvador, pero aún no se ha explotado todas las virtudes que puede ofrecer, tal es el caso de la Universidad Francisco Gavidia que cuenta con laboratorios virtuales en algunas de sus cátedras, lo mismo en la Universidad Don Bosco, donde algunas de sus prácticas de laboratorio de materias como Física y Electrónica, se desarrollan apoyadas por simuladores.

A continuación se mencionan algunas instituciones que han desarrollado tecnología de simulación para la enseñanza de Física y temas relacionados:

- Física por Ordenador. Curso completo de Física interactivo de la Escuela Universitaria de Ingeniería Técnica Industrial de Eibar (España)
- Laboratorio virtual de la Universidad de Oregon (USA): http://jersey.uoregon.edu/vlab/
- Davidson College Department of Physics (USA): http://webphysics.davidson.edu/faculty/jkk/
- Página del profesor de Física Jesús Ruiz Felipe del Instituto Julio Rey Pastor (Albacete,España): http://acacia.pntic.mec.es/~jruiz27/contenidos.htm
- Paul Falstad's Home Page software developer in Santa Barbara, CA: http://www.falstad.com/index.html
- Demostración de principios básicos de Física, por Wolfgang Bauer, WalterBenenson, Gerd Kortemeyer, y Gary Westfall: http://www.msu.edu/user/brechtjo/physics/
- The Department of Physics and Astronomy at Northwestern University: http://www.physics.nwu.edu/ugrad/vpl/

1.2 IMPORTANCIA DE LA INVESTIGACIÓN

1.2.1 PLANTEAMIENTO DEL PROBLEMA

En la actualidad todas las carreras técnicas y de ingeniería cuentan con un buen número de materias que requieren, para su completo desarrollo y entendimiento, la aplicación de los conceptos teóricos en laboratorios prácticos que acerquen al estudiante a la comprensión del tema. Esto es, sin duda, una gran ayuda para éste, y a su vez, indispensable para el desarrollo del curso.

La materia de Física I, como otras de su rama, posee una dificultad especial que la vuelven un obstáculo grande a los alumnos que la cursan. Tal es el caso, que uno de los problemas que la Universidad enfrenta hoy en día es la deserción de estudiantes de sus respectivas carreras a tempranas horas de la misma, esto debido a que en la mayoría de los casos el alumno pierde el incentivo a causa de bajas calificaciones en sus materias. Gracias a fuentes administrativas de la Universidad, se ha podido confirmar este hecho, al encontrar un buen porcentaje de estudiantes reprobados por ciclo para Física I (Ver anexo I).

Muchas veces la metodología empleada para dar a conocer y entender los conceptos, no es la más adecuada o efectiva, debido a que estos conceptos se exponen únicamente basados en la teoría de libros de texto en lugar de introducir el concepto de manera tal, que éste se entienda sin necesidad de memorizarlo textualmente, sino más bien, basado en su comprensión. Esta tarea la cumplen los laboratorios prácticos, pero lastimosamente, éstos no están al alcance de cualquier estudiante, debido a que el equipo utilizado para esto no puede ser adquirido con facilidad y resultaría dificultoso tener que viajar a la Universidad cada vez que una duda surja para el entendimiento de un tema, agregando a esto la poca disponibilidad del equipo ya mencionado. Además podemos agregar, que la metodología actual muchas veces resulta poco atractiva para el estudiante, y no estaría de más experimentar un cambio que involucre la tecnología actual.

SÍNTESIS DE LA PROBLEMÁTICA

El problema lo podemos resumir en la alta tasa de reprobación que existe para la materia de Física I como lo muestran las estadísticas, esto produce una considerable tasa de deserción de estudiantes a ese nivel de la carrera, lo que obliga a desarrollar diferentes metodologías que se traduzcan en un mejor entendimiento del contenido de la materia por parte de los estudiantes y mejore el nivel de aprobación de la misma.

1.2.2 DEFINICIÓN DEL TEMA

El actual documento presenta el desarrollo de un simulador que será una herramienta de gran utilidad a alumnos y docentes involucrados en la materia de Física I, la cual se desarrolla basándose en la aplicación de tecnología de punta en programación orientada a objetos.

Dicha aplicación tiene la capacidad de:

- Proveer al usuario el contenido teórico de cada tema, definición de conceptos y fórmulas que se aplicarán en la práctica.
- Resolver problemas de carácter teórico en un ambiente gráfico bidimensional, que simula la situación planteada por el usuario para llegar a un resultado cuantitativo.
- Evaluar al usuario en cuanto al conocimiento del tema en estudio.

Se pretende que dicha solución proporcione una mayor atención, comprensión y facilidad al estudiante en el desarrollo del curso.

1.2.3 JUSTIFICACIÓN

El problema planteado en el apartado 1.2.1 define la necesidad del desarrollo de una herramienta que aporte nuevos métodos de estudio a alumnos que cursan materias que poseen un considerable grado de dificultad. Las expectativas que genera un proyecto de este tipo son muchas, debido a los beneficios que se pretende obtener del mismo, tanto a nivel pedagógico como en el desarrollo tecnológico actual.

Describiendo de manera más clara lo anterior, podemos afirmar que es necesario lo siguiente:

- Desarrollar una herramienta que sirva de apoyo a alumnos en la comprensión de una materia que generalmente resulta dificultosa en su estudio, pretendiendo disminuir así, el número de estudiantes reprobados o que abandonen la carrera por no poder superar estos obstáculos.
- Impulsar la creación de métodos de estudio que cambien el concepto de aprendizaje, debido a que mantendrá al alumno más apegado a la realidad de lo que la situación de un problema dado plantea, y no lo envolverá en una serie de números y fórmulas que tienden a volver el estudio bastante tedioso.
- Debido al cambio tecnológico que día a día sucede en todos los ámbitos, es necesario aportar una herramienta que sobrepase la metodología de estudio clásica de la Universidad, para aumentar las fortalezas de dicha institución.

1.3 OBJETIVOS

1.3.1 GENERAL

- Desarrollar un simulador para la enseñanza de la asignatura Física I para la Universidad Don Bosco en un ambiente gráfico bidimensional, que sea una herramienta útil, fácil y accesible para alumnos y docentes.

1.3.2 ESPECIFICOS

- Analizar la problemática que causa el alto índice de reprobación con que cuenta la materia de Física I.
- Determinar las factibilidades de desarrollo e implementación del proyecto.
- Aplicar diferentes técnicas de investigación y análisis para diseño y desarrollo de una herramienta capaz de apoyar a docentes y estudiantes en sus respectivos roles.
- Obtener la información adecuada para el diseño de la aplicación utilizando diferentes métodos de investigación y desarrollo.
- Desarrollar una interfaz gráfica, que sirva de apoyo para la comprensión y entendimiento de la Física I.

1.4 ALCANCES

A continuación se detallan los aspectos que han sido tomados en cuenta para la realización del proyecto, los cuales sirvieron de referencia al momento de evaluar los detalles que éste abarca.

- El desarrollo de una aplicación multimedia que pueda ser utilizada como herramienta de enseñanza y estudio, que sea amigable y de fácil entendimiento para el usuario.
- El desarrollo de manuales de usuario y programador.
- La visualización en pantalla de los últimos cinco resultados obtenidos en las simulaciones, los cuales están disponibles únicamente si la aplicación está en ejecución.
- El desarrollo de un índice de conceptos con su respectiva definición relacionados a una unidad en estudio específica.
- El desarrollo de un módulo de evaluación al usuario, que devuelve los resultados obtenidos de la misma.
- Desarrollo de problemas planteados por el usuario mediante determinados parámetros.
- La simulación del desarrollo de un problema en un ambiente gráfico bidimensional, devolviendo él o los resultados cuantitativos según el problema lo amerite.
- Acceso a información que sirva de referencia al usuario de cómo utilizar la aplicación.
- Los temas contemplados en la simulación son los siguientes:

  UNIDAD I: *VECTORES*

  1. Suma y resta de vectores.
  2. Multiplicación de un vector por un escalar.

  UNIDAD II: *MOVIMIENTO EN UNA DIMENSION*

  1. Velocidad y aceleración instantánea.
  2. Movimiento con aceleración constante
  3. Movimiento vertical

  UNIDAD III: *MOVIMIENTO EN DOS DIMENSIONES*

  1. Velocidad y aceleración instantánea
  2. Movimiento de proyectiles
  3. Movimiento circular uniforme

  UNIDAD IV: *FUERZA Y MOVIMIENTO*

  1. Primera ley de Newton
  2. Segunda ley de Newton
  3. Tercera ley de Newton
  4. Fuerzas de fricción

UNIDAD V: *TRABAJO Y ENERGIA*

1. Trabajo efectuado por una fuerza variable
2. Conservación de la energía mecánica

UNIDAD VI: *COLISIONES*

1. Colisiones elásticas en una dimensión

UNIDAD VII: *ROTACIÓN*

1. Movimiento de rotación con aceleración constante

1.5 LIMITACIONES

Se ha tomado en cuenta también, aspectos que dificultan el desarrollo del proyecto o que en su defecto, no son tomados en cuenta para la realización del mismo, tales aspectos se detallan a continuación.

- No se contempla el desarrollo de interfaces gráficas tridimensionales.
- Se interactúa con la aplicación mediante el establecimiento de parámetros sin manipular las interfaces para modificar su entorno.
- No se contempla el uso de sonidos dentro de la aplicación.

1.6 PROYECCIÓN SOCIAL

Es uno de los principales objetivos de todo estudiante universitario al finalizar su carrera, hacer uso de su aprendizaje para devolver en cierta manera la ayuda prestada por la institución a su desarrollo, colaborando con proyectos que sirvan verdaderamente al desarrollo de las generaciones siguientes y que deje prueba física de que el aprendizaje recibido a lo largo de la carrera es totalmente aplicable al servicio de la sociedad.

Teniendo en cuenta lo anterior, el proyecto descrito en este documento, no tiene una mayor finalidad que el dejar un legado del aprendizaje que cada uno de sus desarrolladores obtuvo a lo largo de su vida universitaria.

Enumerando de manera más clara los principales aspectos sociales que el proyecto pretende abarcar, podemos mencionar los siguientes:

- Contribuir, sin lugar a dudas, a facilitar el aprendizaje en un área de las carreras técnicas y de ingeniería en las que seguramente, muchos alumnos desearían haber contado con una mayor asistencia tanto en el aspecto teórico como práctico, apoyando el trabajo de docentes y libros de textos.
- Ayudar a reforzar el aspecto tecnológico de la enseñanza, enlazando los métodos pedagógicos con el mundo informático, mediante una herramienta multimedia, la cual se espera influya en el desarrollo de una Universidad virtual que traspase las fronteras de la pedagogía tradicional.
- Impulsar el desarrollo de métodos innovadores que coloquen a la Universidad Don Bosco en una aún mayor categoría entre las instituciones nacionales.
- No se puede dejar a un lado el aspecto pedagógico en el que el proyecto contribuirá cuando ya implementado sirva de ejemplo a proyectos nuevos de aspecto similar, los cuales en la actualidad son muy escasos en nuestro medio.

1.7 PLAN DE SOLUCIÓN

Definido el problema principal y las justificaciones del proyecto, no queda más que hablar un poco del diseño de la estrategia de solución más adecuada para que éste sea todo un éxito y se pueda sacar el mayor provecho a sus ventajas.

La herramienta consiste en una aplicación multimedia para la simulación de problemas correspondientes a la materia de Física I, dicha aplicación está compuesta por las siguientes partes:

- Descripción de conceptos y fórmulas de cada tema relacionado a la práctica con el objetivo de proveer al usuario la teoría complementaria para apoyarlo en la comprensión del tema.
- Simulación gráfica de diferentes problemas planteados de acuerdo a la necesidad del usuario mediante parámetros ya establecidos, resolviéndolo dentro de un ambiente ideal, que sirve para una mejor comprensión de la resolución misma del problema.
- Índice conceptual, el cual proporciona una lista de conceptos relevantes con su respectiva definición.

- Evaluación de conocimientos conceptuales mediante una prueba que devuelve el resultado de la misma.
- Instalador automático que provee la facilidad de uso en cualquier ordenador con sistema operativo Windows y a su vez no necesita conexión a internet para su funcionamiento.

En cuanto a la implementación, la propuesta de solución es la de proporcionar los instaladores de la aplicación en un dispositivo de almacenamiento láser para proveer la mayor accesibilidad posible a alumnos y docentes, y con el consentimiento de las autoridades de la Universidad, hacerlo disponible en el sitio web de la misma. La idea básica de ésta solución es que el proyecto llegue con la mayor facilidad posible a cualquier estudiante que necesite hacer uso de sus ventajas, que lo pueda utilizar desde su casa o universidad.

2.1 FACTIBILIDADES DEL PROYECTO

De acuerdo a la información obtenida en la investigación de campo realizada en la Universidad , se presenta a continuación los siguientes resultados.

2.1.1 FACTIBILIDAD TÉCNICA

Para el desarrollo del proyecto se necesita contar con un equipo con las siguientes características:

- Disco duro de 40 Gigabytes o más
- Memoria RAM de 256 Megabytes o más
- Microprocesador de 700 MHz o más
- Sistema Operativo Windows XP Professional.
- Microsoft Visual Studio.NET
- Microsoft Office 2000 o mayor

2.1.2 FACTIBILIDAD ECONÓMICA

2.1.2.1 PRESUPUESTO DE DESARROLLO

GASTO	CANTIDAD ($)
Electricidad	235.00
Internet	162.30
Papelería	114.5
Tinta para impresoras	108.23
Otros	90.00
Total:	**710.73**

Tabla 2.1 Presupuesto de desarrollo

2.1.2.2 PRESUPUESTO DE IMPLEMENTACIÓN

Con la idea básica de favorecer a los estudiantes de Física I a lo largo del curso, se ha considerado la posibilidad de hacer llegar ésta herramienta a cada uno de ellos, para que esté disponible en cualquier momento que deseen utilizarla.

Basado en lo anterior, se debe considerar los costos de implementación respecto al número de estudiantes que cursan la materia por año, incluyendo interciclos (si los hubiere). Física I cuenta con un promedio anual aproximado de 200 alumnos inscritos.[1]

La reproducción del sistema "Física Interactiva" en disco láser no deberá exceder el costo del dispositivo e impresión de carátula (si se desea), lo que se calcula en un promedio de $ 0.85 dependiendo de la presentación que se desee darle al producto. Al trasladar este datos a costo por año, nos da un resultado de $170.00 anuales, seguramente un costo insignificante, tomando en cuenta los beneficios que tanto alumnos como institución obtendrán.

[1] Según Administración Académica de la Universidad Don Bosco.

CAPÍTULO III
METODOLOGÍAS

3.1 METODOLOGÍAS DE LA INVESTIGACIÓN

Para la etapa de investigación de un proyecto, existen diferentes tipos de técnicas; las cuales son utilizadas según los requerimientos, condiciones y características del objeto de estudio.

Con la finalidad de obtener información pertinente que determine las necesidades de desarrollo de un Simulador para la materia de Física I, se considera conveniente aplicar una metodología de investigación que conlleve las técnicas mediante las cuales se haga mayor referencia al problema detectado.

Se toma como sujeto de información a alumnos y docentes de la materia de Física I impartida en la Universidad Don Bosco; contando de esta forma con elementos de juicio necesarios en la selección de técnicas más apropiadas.

3.1.1 OBJETIVO DE LA INVESTIGACIÓN

- Conocer los diversos aspectos que involucra el desarrollo de un simulador para la materia en mención.
- Sondear preferencias, dentro de alumnos y docentes, en cuanto a presentación y simulación de temas.
- Buscar la solución más adecuada posible para satisfacer las necesidades que presentan los alumnos y docentes.

3.1.2 TÉCNICAS DE LA INVESTIGACIÓN

3.1.2.1 FUENTES

Con el objeto de recopilar información para el desarrollo de la aplicación, se hace necesario el uso de:

- *Fuentes Primarias:* éstas se refieren a la investigación que se realiza a través de la observación y entrevistas con la que se puede recolectar información de una forma directa permitiendo analizar la situación actual y poder dar alternativas de solución.

- *Fuentes Secundarias*, para la obtención de la información ya escrita y que ha sido recopilada para su utilización, como libros de texto, metodologías de la investigación, análisis y diseño de sistemas, publicaciones de diferentes manuales, boletines e información que se adecue al enriquecimiento del objeto en estudio que se encuentran en sitios por Internet.

3.1.2.2 TÉCNICAS

a) Observación Directa.

Permitió conocer información que no se puede obtener por otras técnicas, pues es de primera mano, concediendo la oportunidad de estar en contacto con la forma en que se efectúan las actividades realmente, visualizando aspectos que ayudan a hacer la información más objetiva. En este caso se observó el desarrollo de algunos temas y prácticas de laboratorio de la materia.

b) Encuestas

Permitió conocer y formarse un criterio para posteriormente cuantificar diferentes tipos de datos e información, orientados a una muestra representativa de personas involucradas en los diferentes roles educativos. Para poder satisfacer las necesidades de los mismos. (Ver anexo III),

c) Cuestionarios

El cuestionario fue la técnica o instrumento mediante el cual se recopiló la información que está íntimamente relacionada con los objetivos del proyecto.

El fundamento de éste instrumento fueron las preguntas contenidas en él. Las respuestas proporcionadas fueron de mucha importancia para obtener criterios y opiniones de los sujetos de información. (Ver anexo V).

3.1.2.3 DEFINICIÓN DE LA POBLACIÓN

La población seleccionada para la investigación está dirigida a tres direcciones que son:

- Alumnos de la materia: por ser éstos los principales usuarios y beneficiarios de la aplicación.
- Docentes: por ser las personas encargadas de transmitir a los alumnos sus conocimientos acerca de la materia y que podrán apoyarse en la aplicación para que se les facilite su labor.
- Instructores: ya que son ellos quienes están encargados de explicar los sucesos de la materia de una manera práctica y visual.

3.2 METODOLOGÍA DE DESARROLLO

3.2.1 HERRAMIENTAS DEL ANÁLISIS Y DISEÑO

- *Diagrama de Flujos de Datos:* indica como se comporta el sistema a consecuencia de sucesos externos. El DFD es también conocido como grafo de flujo de datos o como diagrama de burbujas.

- *Diagrama Jerárquico:* Muestra la estructura de niveles de la aplicación a nivel de procedimientos.

- *Diagrama Entidad – Relación*: Representa las relaciones entre los objetos de datos. El diagrama de entidad – relación permite que un ingeniero del software especifique los objetos de datos que entran y salen de un sistema, los atributos que definen las propiedades de estos objetos, y las relaciones entre los objetos.

- *Flujogramas:* representa de forma gráfica la estructura de la codificación para cada uno de las diferentes funciones con que cuenta la aplicación.

3.2.2 HERRAMIENTAS DE DESARROLLO

3.2.2.1 MODELO DE ENSAMBLAJE DE COMPONENTES

Para el desarrollo del sistema se utiliza la programación avanzada orientada a objetos, por lo cual se ha decidido adoptar el Modelo de Ensamblaje de Componentes para su respectivo desarrollo. Dicho modelo consiste en una secuencia de desarrollo en espiral, que involucra varias fases o actividades estructurales, las cuales son constantemente llevadas a cabo a medida se avanza en el proceso.

Además el modelo involucra el desarrollo de clases o componentes los cuales son almacenados en una biblioteca de clases pasa su posterior reutilización o ensamblaje.

3.2.2.2 PROGRAMACIÓN ORIENTADA A OBJETOS

La programación orientada a objetos es una evolución de la programación procedural basada en funciones. La POO nos permite agrupar secciones de código con funcionalidades comunes.

La organización de una aplicación en POO se realiza mediante estructuras de código, también llamados objetos. Estos objetos contienen una serie de procedimientos e información destinados a resolver un grupo de tareas con un denominador común. Un procedimiento que este situado en un objeto no podrá ser usado por otro procedimiento perteneciente a otro objeto, si no es bajo una serie de reglas. Los datos que mantenga el objeto, permanecerán aislados del exterior y sólo se podrá acceder a ellos siguiendo ciertas normas.

El objetivo de POO es catalogar y diferenciar el código, en base a estructuras jerárquicas dependientes, al estilo de un árbol genealógico.

3.2.2.2.1 VISUAL BASIC .NET

Visual Basic.NET (VB.NET) es una versión de Visual Basic enfocada al desarrollo de aplicaciones .NET. El lenguaje de programación es Visual Basic, que apareció el año 1991 como una evolución del QuickBasic que fabricaba Microsoft.

Es un lenguaje de programación orientado a objetos, y como novedades más importantes en la versión .Net podemos citar la posibilidad de definir ámbitos de tipo, clases que pueden derivarse de otras mediante herencia, sobrecarga de métodos, nuevo control estructurado de excepciones o la creación de aplicaciones con múltiples hilos de ejecución.

Otras de sus características más importantes son:

- Diseño de controles de usuario para aplicaciones Windows y Web.
- Programación de bibliotecas de clase.
- Envío de datos vía documentos XML.
- Generación de reportes basados en Crystal Reports a partir de información obtenida de orígenes de datos (archivos de texto, bases, etc.)

3.2.2.2.2 BASE DE DATOS ACCESS

Microsoft Access es un sistema de gestión de bases de datos (DBMS) para uso personal o de pequeñas organizaciones. Para bases de datos de gran calibre (en cuanto a volumen de datos o de usuarios) es recomendable usar otros sistemas como Microsoft SQL Server, MySQL u Oracle.

Su funcionamiento se basa en un motor llamado Microsoft Jet, y permite el desarrollo de pequeñas aplicaciones formadas por formularios Windows y código VBA (Visual Basic para Aplicaciones). Entre las principales funcionalidades de Access se encuentran:

Crear tablas de datos indexadas. Modificar tablas de datos. Relaciones entre tablas (creación de bases de datos relacionales). Creación de consultas y vistas. Consultas referencias cruzadas. Consultas de acción (INSERT, DELETE, UPDATE). Formularios. Informes. Llamadas a la API de Windows. Iteración con otras aplicaciones que usen VBA (resto de aplicaciones de Microsoft Office, Autocad, etc.). Macros.

Además, permite crear frontends de bases de datos más potentes ya que es un sistema capaz de acceder a tablas externas a través de ODBC como si fueran tablas Access.

Es un software de gran difusión entre pequeñas empresas (PYMES) cuyas bases de datos no requieren de excesiva potencia, ya que se integra perfectamente con el resto de aplicaciones de Microsoft y permite crear pequeñas aplicaciones con unos pocos conocimientos de programación.

Entre sus mayores inconvenientes figuran que no es multiplataforma, pues sólo está disponible para sistemas operativos de Microsoft, y que no permite transacciones. Su uso es inadecuado para grandes proyectos de software que requieren tiempos de respuesta críticos o muchos accesos simultáneos a la base de datos.

4.1 REFERENCIA HISTÓRICA

Los laboratorios virtuales comenzaron a desarrollarse en 1997 en el Centro de Investigación Académica de la Universidad Estatal a Distancia de Costa Rica. Si se juzga con base en la información disponible en Internet, fueron de los primeros laboratorios virtuales para enseñanza a distancia a nivel mundial. Cuatro años después, había un proyecto comercial similar, el Virtual Frog Dissection Kit 1.0[2] y tres académicos: Diffusion Proceses Virtual Laboratory (Johns Hopkins University,[3] The Virtual Microscope[4]. Había también dos proyectos con nivel de realidad virtual, nivel que requiere cascos tipo VR, en Estados Unidos y Canadá[5]. No sabemos de proyectos similares en América Latina, fuera del de la UNED. En la UNED, el primer bloque se desarrolló durante tres años y correspondió al curso de biología. El objetivo básico no ha cambiado desde entonces: lograr un producto tan bueno como los de los países más avanzados en docencia electrónica, a un costo muy inferior al de ellos, que funcionara casi en cualquier computadora y que solo requiriera programas que son gratuitos en todo el mundo.

Transcurridos seis años, ¿cuál es el estado de los laboratorios virtuales en el mundo? Una búsqueda en Internet (junio 2002) indicó que ha aumentado mucho el número de proyectos semejantes y que la mayoría se refieren al área de la física, aunque también los hay de química y biología.

La mayoría de los laboratorios virtuales de física son pequeñas simulaciones escritas en JAVA, un lenguaje de programación interactivo para multimedios. Ejemplo de los tipos de microprácticas en física son los de la Universidad Nacional de Colombia[6], la Universidad de Oregón[7] y un material privado brasileño llamado "Sala de Física"[8]. También se ha desarrollado simulaciones mediante las cuales se modifica los datos en una tabla y se ve la modificación resultante en un esquema, en el Centro Nacional de Información y Comunicación Educativa de Madrid[9]. Un nivel algo más avanzado existe en dos laboratorios en los que se usa sonido e imágenes realistas, en lugar de esquemas: el sintetizador de

[2] http://www.cs.ubc.ca/nest/magic/projects/hands/home, febrero 2000
[3] http://www.jhu.edu/~virtlab/virtlab.html
[4] University of Winnipeg, http://www.uwinnipeg.ca/~simmons/index.htm
[5] NASA Virtual reality Virtual Object Manipulation, www.nasa.gov y Virtual Hand Laboratory, University of British Columbia, http://www.cs.ubc.ca/nest/magic/projects/hands/home
[6] http://www.unalmed.edu.co/~daristiz/LABFIS/Principal/Labfis.htm
[7] http://jersey.uoregon.edu/vlab/
[8] http://geocities.yahoo.com.br/saladefisica3/laboratorio.htm
[9] http://enebro.pntic.mec.es/~fmag0006/Prism200.html

sonido Tempes de la compañía Virtual Laboratories de Singapur[10] y el "Filtro de polarización" de la red Physicsweb[11].

Los laboratorios virtuales de química parecen ser escasos. La Universidad de Oxford presenta, de manera gratuita vía Internet, laboratorios virtuales de experimentos químicos que usan animaciones, videos y moléculas que pueden hacerse girar en la pantalla, manipulables en tres dimensiones[12]. El estudiante debe responder a una serie de preguntas, y si lo hace correctamente, tiene acceso a una fotografía de la mesa de trabajo de la cual puede seleccionar compuestos y experimentos para ver videos sobre su uso. En algunas escenas aparecen rótulos de apoyo que explican el procedimiento. Hay además un texto sobre química en formato HTML con problemas y cuestionario.

La disponibilidad de laboratorios virtuales de biología es intermedio entre los de física y química.

El nivel más sencillo es el que tiene básicamente un texto y dibujos sin movimiento. Ejemplos de este nivel son el Digital Frog de 1995[13] y el Laboratorio Virtual de Nutrición de la UNED de 1997 (Monge-Nájera 1998). Digital Frog permite hacer una disección simulada de una rana, evitando los problemas éticos y psicológicos de hacerlo con un animal real. Nuestro laboratorio de nutrición permite "alimentar" una mascota digital o tamaguchi y ver los efectos de la dieta sobre su salud.

En un segundo nivel de complejidad, existen laboratorios que usan animaciones usando el formato GIF, compatible con Internet. Un ejemplo es el Laboratorio Virtual de Reproducción de la UNED de 1997 (Monge-Nájera 1998), en el cual se puede seleccionar organismos para ver una animación que muestra su secuencia reproductiva, incluyendo imágenes de microscopio electrónico. El Laboratorio Virtual de Depredadores y Presas (UNED 2002) permite variar la proporción de organismos en un ambiente y ver el efecto sobre la población.

El tercer nivel corresponde a los laboratorios que usan videos para mostrar prácticas verdaderas. Ejemplos de este nivel son el Laboratorio Virtual de Digestión desarrollado por la UNED en 1997 (Monge-Nájera 1998) y el Digital Frog 2, versión mejorada del ya mencionado, en que además de las imágenes fijas hay videos[14].

---

[10] http://www.vlabs-online.com/
[11] http://physicsweb.org/vlab/
[12] http://www.chem.ox.ac.uk/vrchemistry/
[13] http://www.digitalfrog.com/products/frog.html
[14] http://www.digitalfrog.com/products/frog.html

En el cuarto nivel de complejidad están aquellos laboratorios en los cuales se ven pantalla objetos o escenas que pueden ser manipulados por el estudiante. La UNED desarrolló entre 1997 y el 2002 una serie de laboratorios virtuales de este nivel. En el Laboratorio Virtual de Tejidos Humanos, se separa la capa de tejidos de un cuerpo humano y se les lleva a un microscopio para verlos ampliados. El Laboratorio Virtual de Ecología permite ubicar organismos en una pirámide ecológica y la computadora indica si se les ha ubicado correctamente. En el Laboratorio Virtual sobre Lepidópteros, se puede manipular un panorama que simula el efecto de girar la cabeza mientras se camina por un bosque. Incluso se puede buscar organismos ocultos en el bosque, solicitar ayuda a la computadora para encontrarlos y obtener información adicional sobre cada uno que se descubra. Laboratorios similares en su nivel son el Virtual Drosophila Project japonés (Kioda y Kitano 1999) y Mouse genetics[15].

4.2 MARCO CONCEPTUAL

4.2.1 EL PROCESO DEL SOFTWARE

El proceso del software es un marco de trabajo de las tareas que se requieren para construir software de alta calidad. Define el enfoque que se toma cuando el software es tratado por la ingeniería, pero la tecnología del software también acompaña a las tecnologías que pueblan el proceso (métodos técnicos y herramientas automatizadas).

4.2.1.1 MODELOS DE PROCESOS EVOLUTIVOS DE SOFTWARE

Se reconoce que el software al igual que todos los sistemas complejos, evoluciona con el tiempo. Los ingenieros del software necesitan un modelo de proceso que se haya diseñado explícitamente para acomodarse a un producto que evolucione con el tiempo.

Los modelos evolutivos son iterativos. Se caracterizan por la forma en que permiten a los ingenieros de software desarrollar versiones cada vez más completas del software.

[15] http://www.explorescience.com/activities/Activity_page.cfm?ActivityID=39

4.2.1.1.1 EL MODELO EN ESPIRAL

El modelo en espiral, es un modelo de proceso de software evolutivo que acompaña la naturaleza interactiva de construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. Se proporciona el potencial para el desarrollo rápido de versiones incrementales del software. En el modelo espiral, el software se desarrolla en una serie de versiones incrementales. Durante las primeras iteraciones, la versión incremental podría ser un modelo en papel o un prototipo. Durante las últimas iteraciones, se producen versiones cada vez más completas de ingeniería del sistema.

El modelo en espiral se divide en un número de actividades estructurales, también llamadas regiones de tareas[16]. Generalmente existen entre tres y seis regiones de tareas. La figura 4.1 representa un modelo en espiral que contiene seis regiones de tareas:

- **Comunicación con el cliente:** las tareas requeridas para establecer comunicación entre el desarrollador y el cliente.
- **Planificación:** las tareas requeridas para definir recursos, el tiempo y otras informaciones relacionadas al proyecto.
- **Análisis de riesgos:** las tareas requeridas para evaluar riesgos técnicos y de gestión.
- **Ingeniería:** las tareas requeridas para construir una o más representaciones de la aplicación.
- **Construcción y adaptación:** las tareas requeridas para construir, probar, instalar y proporcionar soporte al usuario (p. Ej. Documentación y práctica).
- **Evaluación del cliente:** las tareas requeridas para obtener la reacción del cliente según la evaluación de las representaciones del software creadas durante la etapa de ingeniería e implementada durante la etapa de instalación.

Cada una de las regiones están pobladas por una serie de tareas que se adaptan a las características del proyecto que va a emprenderse. Para proyectos pequeños, el número de tareas y su formalidad es bajo. Para proyectos mayores y más críticos, cada región contiene tareas que se definen para lograr un nivel más alto de formalidad.

[16] El modelo en espiral de esta sección es una variante sobre el modelo propuesto por Boston.

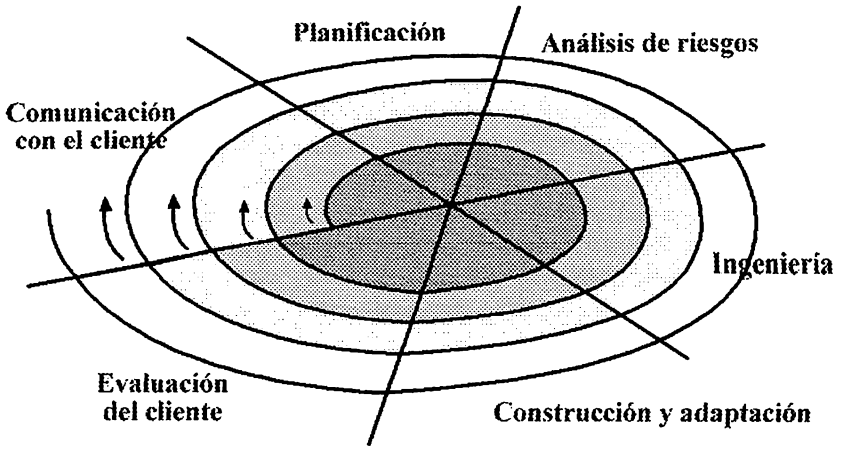


Figura 4.1 Un modelo en espiral típico

Cuando empieza este proceso evolutivo, el equipo de ingeniería del software gira alrededor de la espiral en dirección de las agujas del reloj, comenzando por el centro. El primer circuito de la espiral produce un desarrollo de una especificación de productos; los pasos siguientes en la espiral se podrían utilizar para desarrollar un prototipo y progresivamente versiones más sofisticadas del software. Cada paso de la región de planificación produce ajustes en el plan de proyecto. El coste y la planificación se ajustan según la reacción ante la evaluación del cliente. Además, el gestor del proyecto ajusta el número planificado de iteraciones requeridas para completar el software.

El modelo en espiral es un enfoque realista del desarrollo de sistemas y de software a gran escala. Como el software evoluciona, a medida que progresa el proceso, el desarrollador y el cliente comprenden y reaccionan mejor ante riesgos en cada uno de los niveles evolutivos. El modelo en espiral utiliza la construcción de prototipos como mecanismo de reducción de riesgos, pero lo que es más importante, permite a quien lo desarrolla aplicar el enfoque de construcción de prototipos en cualquier etapa de evolución del producto. Mantiene el enfoque sistemático de los pasos sugeridos por el ciclo de vida clásico, pero lo incorpora al marco de trabajo interactivo que refleja de forma más realista el mundo real. El modelo en espiral demanda una consideración directa de los riesgos técnicos en todas las etapas del proyecto, y si se aplica adecuadamente, debe reducir los riesgos antes de que se conviertan en problemáticos.

4.2.1.1.2 EL MODELO DE ENSAMBLAJE DE COMPONENTES

El modelo de ensamblaje de componentes (Figura 4.2) incorpora muchas de las características del modelo en espiral. Es evolutivo por naturaleza, y exige un enfoque interactivo para la creación del software. Sin embargo, el modelo ensamblador de componentes configura aplicaciones desde componentes preparados de software algunas veces llamados "clases".

La actividad de la ingeniería comienza con la identificación de clases candidatas. Esto se lleva a cabo examinando los datos que se van a manejar por parte de la aplicación y el algoritmo que se va a aplicar para conseguir el tratamiento[17]. Los datos y los algoritmos correspondientes se empaquetan en una clase.

Las clases llamadas componentes en la Figura 4.2 creadas en los proyectos de ingeniería del software anteriores se almacenan en una biblioteca de clases o depósito. Una vez identificadas las clases candidatas, la biblioteca de clases se examina para determinar si estas clases ya existen. En caso de que así fuera, se extraen de la biblioteca, se aplican a los métodos orientados a objetos. Se compone así la primera interacción de la aplicación a construirse, mediante las clases extraídas de la biblioteca y las clases nuevas construidas para cumplir las necesidades únicas de la aplicación. El flujo del proceso vuelve a la espiral y volverá a introducir por último la iteración ensambladora de componentes a través de la actividad de ingeniería.

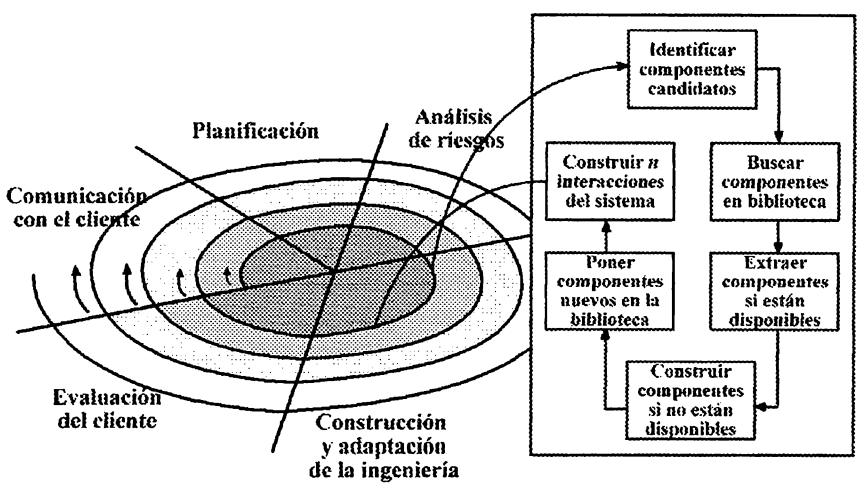


Figura 4.2 El modelo de ensamblaje de componentes

[17] Esta es una descripción simplificada de clase.

El modelo ensamblador de componentes lleva a la reutilización del software, y la reutilización proporciona beneficios a los ingenieros del software.

4.2.2 MODELADO DEL ANÁLISIS

4.2.2.1 MODELADO FUNCIONAL Y FLUJO DE INFORMACIÓN

La información se trasforma a medida que fluye por un sistema basado en computadora. El sistema acepta entradas en una gran variedad de formas, aplica elementos de hardware, software y humanos para transformar la entrada en salida, y produce salida en una gran variedad de formas.

Podemos crear un modelo de flujo para cualquier sistema de computadora, independientemente del tamaño y complejidad.

4.2.2.1.1 DIAGRAMA DE FLUJO DE DATOS

El diagrama de flujo de datos (DFD) es una técnica que representa el flujo de la información y las transformaciones que se aplican a los datos al moverse desde la entrada hasta la salida. En la figura 4.3 se muestra la forma básica de un diagrama de flujo de datos. El DFD es también conocido como grafo de flujo de datos o como diagrama de burbujas.

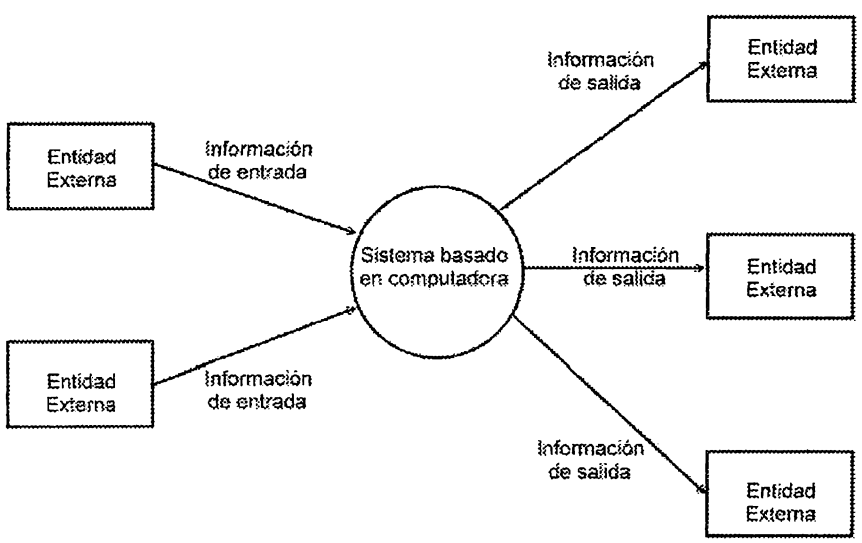


Figura 4.3 Modelo del flujo de información

Se puede usar el diagrama de flujo de datos para representar un sistema o software a cualquier nivel de abstracción. De hecho, los DFD pueden ser divididos en niveles que representen un mayor flujo de información y un mayor detalle funcional. Por consiguiente, el DFD proporciona un mecanismo para el modelado funcional así como para el modelado del flujo de información.

En la figura 4.4 se ilustra la notación básica que se usa para crear un DFD. El rectángulo se usa para representar una entidad externa, es decir un elemento del sistema , u otro sistema que produzca información a ser transformada por el software o que reciba información producida por el software. Un círculo representa un proceso o transformación que se aplica a los datos y cambia de alguna forma. Todas las flechas de un diagrama de flujo de datos deben estar etiquetadas.

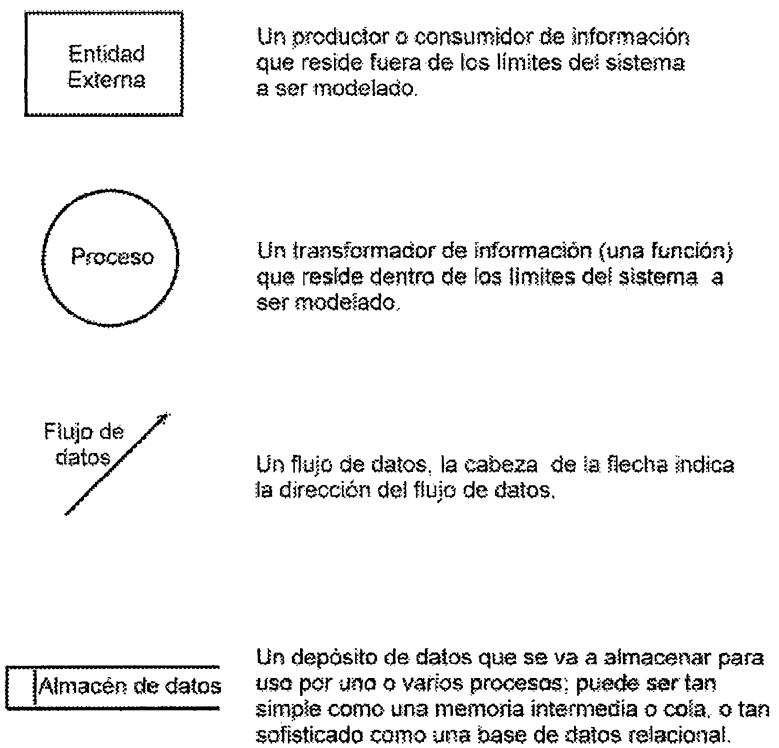


Figura 4.4 Notación DFD básica

4.2.2.1.1.1 CREACIÓN DE UN MODELO DE FLUJO DE DATOS

El diagrama de flujo de datos (DFD) permite al ingeniero del software desarrollar los modelos del ámbito de información y del ámbito funcional al mismo tiempo.

Unas pocas directrices sencillas pueden ayudar de forma considerable durante la elaboración de un diagrama de flujo de datos:

1. El diagrama de flujo de datos de nivel 0 debe representar el software / sistema como una sola burbuja.
2. Se deben anotar cuidadosamente las entradas y salidas principales
3. El refinamiento debe comenzar aislando los procesos, los objetos de datos y los almacenes de datos que sean candidatos a ser representados en el siguiente nivel.
4. Todas las flechas y las burbujas deben ser rotuladas con nombres significativos.
5. Entre sucesivos niveles se debe mantener la continuidad del flujo de información.
6. Se deben refinar las burbujas de una en una.

4.2.3 MODELADO DEL DISEÑO

4.2.3.1 MODELADO DE DATOS

El modelado de datos responde a una serie de preguntas específicas importantes para cualquier aplicación de procesamiento de datos. ¿Cuáles son los objetos de datos primarios que va a procesar el sistema?, ¿Cuál es la composición de cada objeto de datos y que atributos describe el objeto?, ¿Dónde residen actualmente los objetos?, ¿Cuál es la relación entre los objetos y los procesos que los transforman?.

Para responder estas preguntas, los métodos de modelado de datos hacen uso del diagrama entidad – relación, que permite que un ingeniero del software identifique objetos de datos y sus relaciones mediante una notación gráfica.

4.2.3.1.1 OBJETO DE DATOS, ATRIBUTOS, RELACIONES

Un **objeto de datos** es una representación de cualquier composición de información compuesta que deba comprender el software. Entendiéndose por composición de información, todo aquello que tiene un número de propiedades o atributos diferentes.

Los objetos de datos se relacionan con otros. Las relaciones siempre se definen por el contexto del problema que se está utilizando.

Un objeto de datos encapsula datos solamente por consiguiente, el objeto de datos se puede representar como una tabla. Los encabezamientos de las tablas reflejan atributos del objeto. El cuerpo de la tabla representa ocurrencias del objeto de datos.

Los **atributos** definen las propiedades de un objeto de datos y toman una de las tres características diferentes, se pueden usar para:

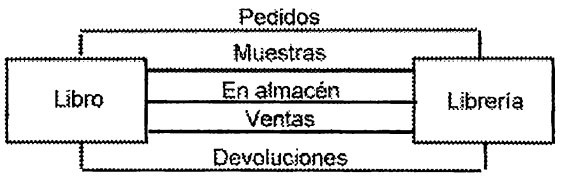
1. Nombrar una ocurrencia del objeto datos
2. Describir la ocurrencia
3. Hacer referencias a otra ocurrencia en otra tabla.

Además uno o varios atributos se definen como un identificador, es decir, el atributo identificador supone una "clave" cuando queramos encontrar una instancia del objeto de dato. En algunos casos, los valores para los identificadores son únicos, aunque esto no es requisito.

Relaciones. Los objetos de datos se conectan entre sí de muchas formas diferentes. Considere dos objetos de datos, *libro* y *librería*. Estos objetos se pueden representar mediante la notación simple señalada en la figura 4.5a. Se establece una conexión entre *libro y librería* porque los dos objetos se relacionan. Pero ¿Qué son relaciones? Para determinar la respuesta, debemos comprender el papel del *libro y librería* dentro del contexto del software que se va a construir. Podemos definir un conjunto de parejas objeto – relación que definen las relaciones relevantes. Por ejemplo:

- una librería pide libros
- una librería muestra libros
- una librería almacena libros.
- una librería vende libros.
- una librería devuelve libros.

a. Una conexión básica entre objetos



b. Relaciones entre objetos

Figura 4.5 Relaciones

Las relaciones *pide*, *muestra*, *almacena*, *vende* y *devuelve* define las conexiones relevantes entre *libro y librería*. La figura 4.5b ilustra estas parejas objeto – relación gráficamente.

Es importante destacar que las parejas objeto – relación tienen dos direcciones; esto es, se pueden leer en cualquier dirección. Una librería pide libros o los libros son pedidos en una librería[18].

4.2.3.1.2 CARDINALIDAD Y MODALIDAD

La **Cardinalidad** representa el número de ocurrencias de objetos que se dan en una relación de la forma siguiente:

- *Uno a uno (1:1):* Una ocurrencia de un objeto A se puede relacionar a una y sólo una ocurrencia del objeto B, y una ocurrencia del objeto B se puede relacionar sólo con una ocurrencia de A.
- *Uno a muchos (1:N):):* Una ocurrencia de un objeto A se puede relacionar a una o muchas ocurrencias del objeto B, pero una de B se puede relacionar sólo a una ocurrencia de A.
- *Mucho a muchos (M:N):):* Una ocurrencia de un objeto A se puede relacionar a una o muchas ocurrencias del objeto B, mientras que una de B se puede relacionar con una o más ocurrencias de A.

La **modalidad** de una relación es cero si no hay una necesidad explícita de que ocurra una relación o de que sea opcional. La modalidad es 1 si una ocurrencia de la relación es obligatoria.

[18] Para evitar cualquier ambigüedad, se debe considerar la manera en que se etiqueta la relación. Por ejemplo, si se considera el contexto para una relación bidireccional, la figura 4.5b podría interpretarse erróneamente como libros piden librerías. En tales casos, se necesita volver a escribir la frase.

4.2.3.1.3 DIAGRAMA ENTIDAD – RELACIÓN

La pareja objeto – relación es la piedra angular del modelo de datos. Estas parejas se pueden representar gráficamente mediante el diagrama entidad – relación (DER)[19]. Se identifica un conjunto de componentes primarios para el DER: objetos de datos, atributos, relaciones y varios indicadores tipo. El propósito primario del DER es representar objetos de datos y sus relaciones.

Los objetos de datos son representados por un rectángulo etiquetado. Las relaciones se indican mediante una línea etiquetada conectando objetos. En algunas variaciones del DER, la línea de conexión contiene un rombo que se etiqueta con la relación. Las conexiones entre objetos de datos y relaciones se establecen mediante una variedad de símbolos especiales que indican cardinalidad y modalidad.

La relación entre los objetos de datos coche y fabricante se representaría como se muestra en la figura 4.6 Un *fabricante* construye uno o muchos *coches*.

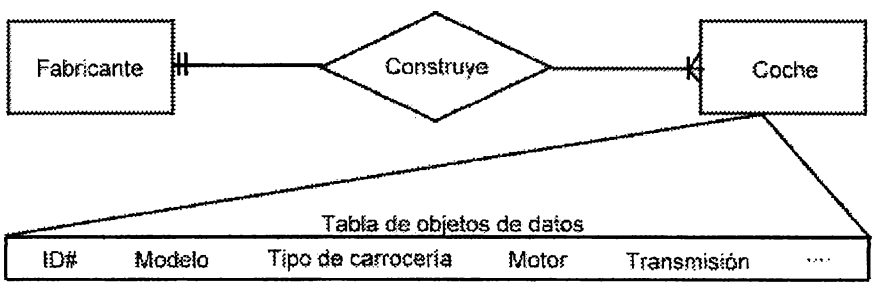


Figura 4.6 Un DER y una tabla de objetos de datos simples

Además de la notación básica de DER básica, el analista puede representar jerarquías de tipos de objetos de datos. En muchas ocurrencias, un objeto de datos realmente puede representar una clase o categoría de información.

El modelado de datos y el diagrama entidad - relación proporcionan al analista una notación concisa para examinar datos dentro del contexto de una aplicación de procesamiento de datos. En la mayoría de los casos, el enfoque del modelado de datos se utiliza para crear una parte del modelo de análisis, pero también se puede utilizar para el diseño de bases de datos y para soportar cualquier otro método de análisis de requisitos.

---

[19] Hoy en día el término "entidad de dato" o "entidad" se ha sustituido por "objeto de datos". Sin embargo, el término "entidad" sigue siendo parte del nombre de la notación gráfica para las parejas objeto – entidad.

4.2.3.1.3.1 CREACIÓN DE UN DIAGRAMA ENTIDAD – RELACIÓN

El diagrama de entidad – relación permite que un ingeniero del software especifique los objetos de datos que entran y salen de un sistema, los atributos que definen las propiedades de estos objetos, y las relaciones entre los objetos. Al igual que la mayoría de los elementos del modelo de análisis y las relaciones entre objetos, el DER se construye de una forma interactiva. Se toma el enfoque siguiente:

1. Durante la recopilación de requisitos, se pide que los clientes listen las "cosas" que afronta el proceso de la aplicación y/o del negocio. Estas "cosas" evolucionan en una lista de objetos de datos de entrada y de salida así como entidades externas que producen o consumen información.
2. Tomando un objeto a la vez, el analista y el cliente definen si existe una conexión (sin especificarlo en ese momento) o no entre ese objeto de datos y otros objetos.
3. Siempre que existe una conexión, el analista y el cliente crean una o varias parejas de objeto - relación.
4. Para cada pareja objeto – relación se explora la cardinalidad y la modalidad.
5. Interactivamente se continúan los pasos del 2 al 4 hasta que se hayan definido todas las parejas objeto – relación. Es normal descubrir omisiones a medida que el proceso continúa. Objetos y relaciones nuevos se añadirán invariablemente a medida que crezca el número de interacciones.
6. Se definen los atributos de cada entidad.
7. Se formaliza y revisa el diagrama entidad – relación.
8. Se repiten los pasos del 1 al 7 hasta que se termina el modelado de datos.

4.2.4 TECNOLOGÍA .NET

La nueva tecnología de Microsoft ofrece soluciones a los problemas de programación actuales, como son la administración de código o la programación para internet. Para aprovechar al máximo las características de .Net es necesario entender la arquitectura básica en la que está implementada ésta tecnología y así beneficiarse de todas las características que ofrece ésta nueva plataforma.

El Framework de .Net es una infraestructura sobre la que se reúne todo un conjunto de lenguajes y servicios que simplifican enormemente el desarrollo de aplicaciones. Mediante ésta herramienta se ofrece un entorno de ejecución altamente distribuido, que permite crear aplicaciones robustas y escalables. Los principales componentes de éste entorno son:

- Lenguajes de compilación
- Biblioteca de clases de .Net
- CLR (Common Language Runtime)

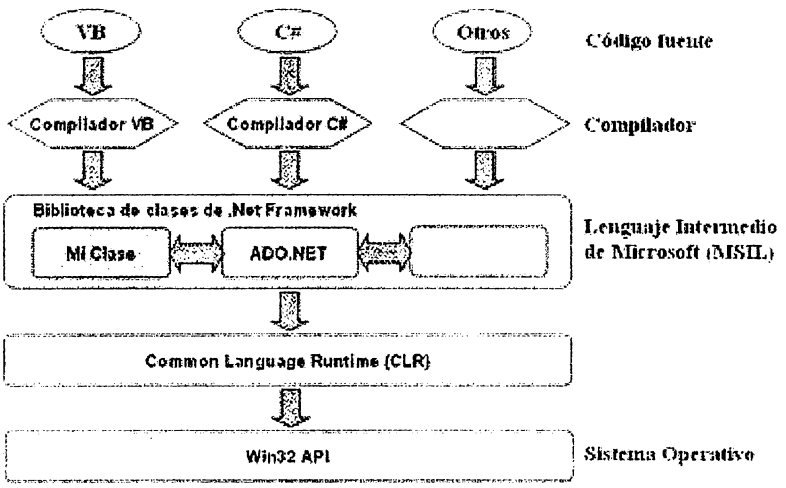


Figura 4.7 Arquitectura de .NET Framework

Actualmente, el Framework de .Net es una plataforma no incluida en los diferentes sistemas operativos distribuidos por Microsoft, por lo que es necesaria su instalación previa a la ejecución de programas creados mediante .Net. El Framework se puede descargar gratuitamente desde la web oficial de Microsoft.

.Net Framework soporta múltiples lenguajes de programación y aunque cada lenguaje tiene sus características propias, es posible desarrollar cualquier tipo de aplicación con cualquiera de estos lenguajes. Existen más de 30 lenguajes adaptados a .Net desde los más conocidos como C# (C Sharp), Visual Basic o C++ hasta otros lenguajes menos conocidos como Perl o Cobol.

4.2.4.1 COMMON LANGUAGE RUNTIME

El CLR es el verdadero núcleo del Framework de .Net ya que es el entorno de ejecución en el que se cargan las aplicaciones desarrolladas en los distintos lenguajes, ampliando el conjunto de servicios que ofrece el sistema operativo estándar Win32.

La herramienta de desarrollo compila el código fuente de cualquiera de los lenguajes soportados por .Net en un mismo código, denominado código intermedio (MSIL, Microsoft Intermediate Language). Para generar dicho código el compilador se basa en el Common Language Specification (CLS) que determina las reglas necesarias para crear código MSIL compatible con el CLR.

De esta forma, indistintamente de la herramienta de desarrollo utilizada y del lenguaje elegido, el código generado es siempre el mismo, ya que el MSIL es el único lenguaje que entiende directamente el CLR. Este código es transparente al desarrollo de la aplicación ya que lo genera automáticamente el compilador.

Sin embargo, el código generado en MSIL no es código máquina y por tanto no puede ejecutarse directamente. Se necesita un segundo paso en el que una herramienta denominada compilador JIT (Just-In-Time) genera el código máquina real que se ejecuta en la plataforma que tenga la computadora.

De esta forma se consigue con .Net cierta independencia de la plataforma, ya que cada plataforma puede tener su compilador JIT y crear su propio código máquina a partir del código MSIL.

La compilación JIT la realiza el CLR a medida que se invocan los métodos en programa y, el código ejecutable obtenido, se almacena en la memoria caché de la computadora, siendo recompilado sólo cuando se produce algún cambio en el código fuente.

4.2.4.2 BIBLIOTECA DE CLASES

Cuando se está programando una aplicación muchas veces se necesita realizar acciones como manipulación de archivos, acceso a datos, conocer el estado del sistema, implementar seguridad, etc. El Framework organiza toda la funcionalidad del sistema operativo en un espacio de nombres jerárquico de forma que a la hora de programar resulta bastante sencillo encontrar lo que se necesita.

Para ello, el Framework posee un sistema de tipos universal, denominado Common Type System (CTS). Este sistema permite que el programador pueda interactuar los tipos que se incluyen en el propio Framework (biblioteca de clases de .Net) con los creados por él mismo (clases). De esta forma se aprovechan las ventajas propias de la programación orientada a objetos, como la herencia de clases predefinidas para crear nuevas clases, o el polimorfismo de clases para modificar o ampliar funcionalidades de clases ya existentes.

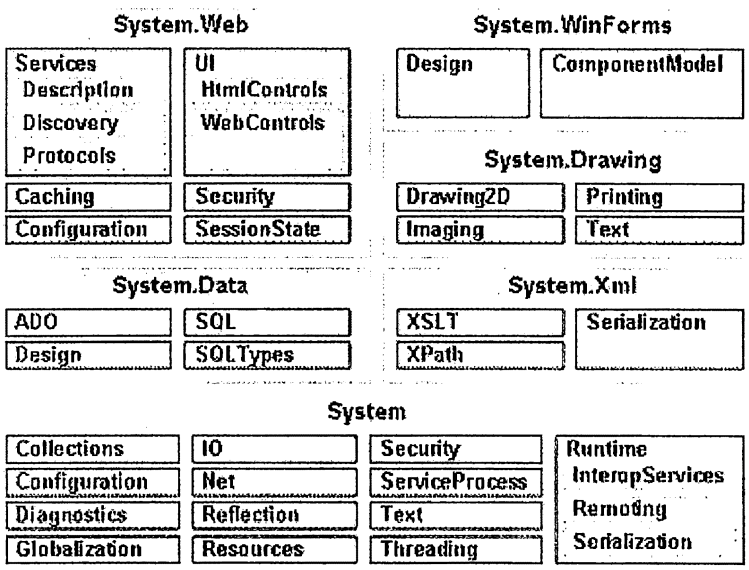


Figura 4.8 Biblioteca de clases de .NET Framework

La biblioteca de clases de .Net Framework incluye, entre otros, tres componentes clave:

- ASP.NET para construir aplicaciones y servicios Web
- Windows Forms para desarrollar interfaces de usuario
- ADO.NET para conectar las aplicaciones a base de datos.

La forma de organizar la biblioteca de clases de .Net dentro del código es a través de los espacios de nombres (namespaces), donde cada clase está organizada en los espacios de nombres según su funcionalidad. Por ejemplo, para manejar ficheros se utiliza el espacio de nombres System.IO y si lo que se quiere es obtener información de una fuente de datos se utilizará el espacio de nombres System.Data.

La principal ventaja de los espacios de nombres de .Net es que de esta forma se tiene toda la biblioteca de clases de .Net centralizada bajo el mismo espacio de nombres (System). Además, desde cualquier lenguaje se usa la misma sintaxis de invocación, ya que a todos los lenguajes se aplica la misma biblioteca de clases.

4.2.4.3 ENSAMBLADOS

Uno de los mayores problemas de las aplicaciones actuales es que en muchos casos tienen que tratar con diferentes archivos binarios (DLL's), elementos de registro, conectividad abierta a base de datos (ODBC), etc.

Para solucionarlo el Framework de .Net maneja un nuevo concepto denominado ensamblado. Los ensamblados son ficheros con forma EXE o DLL que contienen toda la funcionalidad de la aplicación de forma encapsulada. Por tanto la solución al problema puede ser tan fácil como copiar todos los ensamblados en el directorio de la aplicación.

Con los ensamblados ya no es necesario registrar los componentes de la aplicación. Esto se debe a que los ensamblados almacenan dentro de sí mismos toda la información necesaria en lo que se denomina el manifiesto del ensamblado. El manifiesto recoge todos los métodos y propiedades en forma de meta – datos junto con otra información descriptiva, como permisos, dependencias, etc.

Para gestionar el uso que hacen las aplicaciones de los ensamblados .Net utiliza la llamada caché global de ensamblados (GAC, Global Assembly Cache). Así, .Net Framework puede albergar en el GAC los ensamblados que puedan ser usados por varias versiones de un mismo ensamblado, algo que no era posible con el anterior modelo COM.

4.2.4.4 VENTAJAS DE .NET

A continuación se resumen las ventajas más importantes que proporciona .Net Framework:

- **Código administrado**: El CLR realiza un control automático del código para que éste sea seguro, es decir, controla los recursos del sistema para que la aplicación se ejecute correctamente.

- **Interoperabilidad multilenguaje**: El código puede ser escrito en cualquier lenguaje compatible con .Net ya que siempre se compila en código intermedio (MSIL).
- **Compilación just-in-time**: El compilador JIT incluido en el Framework compila el código intermedio (MSIL) generando el código máquina propio de la plataforma. Se aumenta así el rendimiento de la aplicación al ser específico para cada plataforma.
- **Garbage collector**: El CLR proporciona un sistema automático de administración de memoria denominado recolector de basura (garbage collector). El CLR detecta cuándo el programa deja de utilizar la memoria y la libera automáticamente. De ésta forma el programador no tiene por qué liberar la memoria de forma explícita aunque también sea posible hacerlo manualmente (mediante el método disponse() liberamos el objeto para que el recolector de basura lo elimine de memoria).
- **Seguridad de acceso al código**: Se puede especificar que una pieza de código tenga permisos de lectura de archivos pero no de escritura. Es posible aplicar distintos niveles de seguridad al código, de forma que se puede ejecutar código procedente del Web sin tener que preocuparse si esto va a estropear el sistema.
- **Despliegue**: Por medio de los ensamblados resulta mucho más fácil el desarrollo de aplicaciones distribuidas y el mantenimiento de las mismas. El Framework realiza esta tarea de forma automática mejorando el rendimiento y asegurando el funcionamiento correcto de todas las aplicaciones.

¿TODO SON VENTAJAS?

Procesos como la recolección de basura de .Net o la administración de código introducen factores de sobrecarga que repercuten en la demanda de más requisitos del sistema.

El código administrado proporciona una mayor velocidad de desarrollo y mayor seguridad de que el código sea bueno. En contrapartida el consumo de recursos durante la ejecución es mucho mayor, aunque con los procesadores actuales esto cada vez es menos inconveniente.

El nivel de administración del código dependerá en gran medida del lenguaje que utilicemos para programar. Por ejemplo, mientras que Visual Basic .Net es un lenguaje totalmente administrado, C Sharp permite la administración de código de forma manual, siendo por defecto también un lenguaje administrado. Mientras que C++ es un lenguaje no administrado en el que se tiene un control mucho mayor del uso de la memoria que hace la aplicación.

4.2.4.5 ¿QUÉ ES EL .NET FRAMEWORK?

Existen varias definiciones para lo que es .NET Framework, a continuación se listan algunas de ellas:

- ".NET Framework es un entorno para construir, instalar y ejecutar servicios Web y otras aplicaciones. Se compone de tres partes principales: El Common Language Runtime, las clases Framework y ASP.NET"
- "El .NET Framework es un entorno multi-lenguaje para la construcción, distribución y ejecución de Servicios Webs y aplicaciones."
- "El .NET Framework es una nueva plataforma diseñada para simplificar el desarrollo de aplicaciones en el entorno distribuido de internet"
- "El .NET Framework consta de dos componentes principales: el Common Language Runtime y la librería de clases .NET Framework."

5.1 CICLO DE VIDA

El desarrollo de la aplicación abarca diferentes aspectos técnicos los cuales se repiten a medida se avanza a lo largo de dicho proceso, gracias al modelo de desarrollo aplicado.

El modelo de ensamblaje de componentes, consiste en la repetición continua del ciclo de vida del sistema, debido a que el desarrollo de un determinado componente facilita el trabajo de las demás etapas, las cuales tienen idéntica estructura a sus antecesoras y permiten la reutilización. A medida esto ocurre el sistema va tomando su forma y tamaño deseado acercándose al objetivo planificado.

La figura 5.1 representa el modelo de construcción del sistema, compuesto por 4 etapas principales, conocidas en conjunto como ciclo de vida del sistema, estas son: Recolección de Información, Análisis y Diseño, Construcción, Prueba y Depuración.

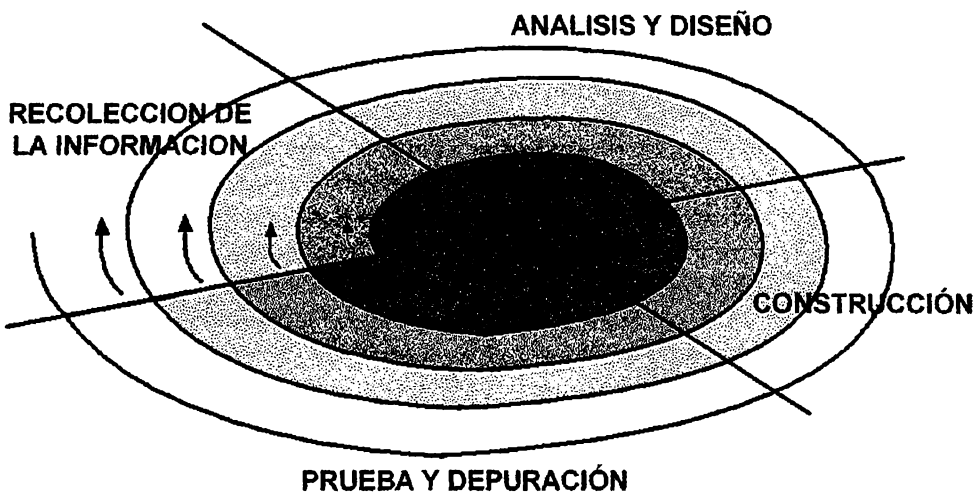


Figura 5.1 Ciclo de Vida del Sistema

5.2 ANÁLISIS Y DISEÑO

5.2.1 ANÁLISIS

Gracias a las diferentes técnicas de investigación utilizadas, se pudo llegar a determinar factores importantes en la construcción del sistema. Dichos factores consisten en las necesidades básicas y que resultan vitales al momento de tomar decisiones en cuanto al diseño del mismo.

a) Observación directa

Esta técnica sirvió para determinar la forma actual en la que la cátedra de Física I es impartida en la Universidad Don Bosco.

La cátedra es impartida de la siguiente manera: el maestro expone un tema según la planificación docente (Ver Anexo II), primero se da a conocer a los alumnos el contenido teórico en el cual se definen conceptos y fórmulas, luego se procede a desarrollar un ejemplo relacionado al mismo en la pizarra con ayuda de dibujos y esquemas que faciliten al alumno el entendimiento y comprensión del tema, además el maestro proporciona una guía de ejercicios que los alumnos deben desarrollar y cada dos semanas se realiza una discusión entre el maestro y los alumnos para resolver un número específico de estos problemas. En cuanto a laboratorios prácticos éstos son programados una vez a la semana y los alumnos deben realizar una guía en la cual deben aplicar los conceptos y fórmulas vistas en la clase.

b) Encuestas (Ver anexo IV)

1. **¿Cómo calificarías a la Física I, en base al grado de dificultad?**
   Un 76% de los encuestados encuentran la materia "Muy difícil", un 20% opinan que es "Difícil" y un 4% que es "Accesible".
   Lo que nos indica que su mayoría, la población opina que la materia si posee un alto grado de dificultad por lo que hay que prestarle atención a la forma de impartirla y estudiarla.

2. **¿Cómo calificarías el nivel académico con que se imparte la cátedra?**
   Un 44% opinan que es "Muy bueno", un 28% "Bueno", un 16% "Regular" y un 12% "Excelente".
   Al profundizar en estas respuesta llegamos a que muchas veces los alumnos califican a sus maestros conforme éstos asimilan la clase y en su mayoría quedan faltos de explicaciones o no entienden a totalidad el tema visto en clase.

3. ¿Cuan importante consideras que son los ejemplos desarrollados en clase por el catedrático para el entendimiento del tema?

El 100% de la muestra opinan que los ejercicios desarrollados en clase son "Muy importantes".

Lo cual nos lleva a la conclusión que para poder entender los fenómenos físicos es sumamente importante y vital el desarrollo de ejercicios que puedan ejemplificar dichos fenómenos.

4. ¿Qué porcentaje aproximadamente de los temas explicados en clase has comprendido sólo con la explicación y el ejemplo?

Un 68% opinan que "Menos del 50%", un 22% que "El 50%" y el 10% opinan que "Más del 50%".

Con lo que se ha podido concluir que no es suficiente la explicación teórico y el desarrollo de un ejemplo en la clase para que los alumnos comprendan los conceptos y fórmulas de un tema en específico.

5. ¿Cuan importante crees que son las guías de ejercicios para la comprensión de los temas?

El 100% de la muestra opina que las guías de ejercicios son "Muy importantes" para la comprensión de los temas de la materia, lo cual nos lleva a la conclusión que es necesario poner en práctica las definiciones de conceptos y fórmulas de manera tal que se comprendan por medio de ejercicios sin necesidad de memorizarlos.

6. ¿Qué porcentaje aproximadamente de las guías de ejercicios has completado en su totalidad?

Un 62% han completado "Menos del 50%" la guía de ejercicios, un 18% solamente un 50%, un 12% "Más del 50%" y un 8% "El 100%".

Con lo cual concluimos que las guías de ejercicios no son desarrolladas por varios motivos, por ejemplo podemos mencionar que en la mayoría de los casos no se entiende el concepto para poder plantear la resolución de un problema.

7. ¿Cuántos de los problemas planteados en las guías has desarrollado llegando a una respuesta pero con la incertidumbre de no saber si la respuesta es correcta o no?

Un 58% respondieron que "La mayoría", 26% "Algunos" y un 16% "La mitad".

Llegamos a la conclusión de que por el simple hecho de memorizar las fórmulas o conceptos no se comprende en realidad el porqué de los fenómenos físicos.

8. ¿Cuántos de los problemas planteados en las guías has logrado desarrollar mediante fórmulas llegando a la respuesta correcta pero sin comprender con claridad el porqué de los datos encontrados ni del fenómeno físico estudiado?

Un 74% respondió "La mayoría", un 20% "Algunos" y un 6% "La mitad".

Llegando a la conclusión de que no es correcto simplemente memorizar las fórmulas para desarrollar los ejercicios y comprender los fenómenos físicos.

9. ¿Cuan importante crees que son los laboratorios prácticos para la comprensión de los temas?

Un 92% de la población respondió "Muy importantes" y un 4% "Poco importantes".

Concluyendo de esta manera que en sí para la mayoría los laboratorios prácticos tienen mucha importancia para la comprensión de los temas de la materia.

10. ¿Qué porcentaje aproximadamente de los laboratorios prácticos desarrollados has completado y comprendido en su totalidad?

Un 34% de la población respondió "El 50%", un 30% "Menos del 50%", un 26% "más del 50%" y un 10% "El 100%".

Concluyendo así que es importante la comprensión de los conceptos para poder aplicarlos en los laboratorios y de ésta manera llegar a la comprensión de los fenómenos físicos.

11. ¿Qué porcentaje aproximadamente de los laboratorios prácticos desarrollados han servido para terminar de comprender el tema en estudio?

Un 38% de la población respondió "El 50%", un 32% "Más del 50%", un 18% "El 100%" y un 12% "Menos del 50%".

Logrando concluir que los laboratorios prácticos sirven en gran manera para la comprensión de los fenómenos físicos.

12. ¿Apoyarías la idea de poder contar con los recursos completos de los laboratorios prácticos en el instante en que tú lo necesites y con toda libertad de uso sin límites de tiempo?

El 100% de la población respondió "Sí", concluyendo de ésta manera que a los alumnos les interesa contar con los recursos necesarios para que sirvan de apoyo en la comprensión de los temas en estudio.

13. ¿Apoyarías la idea de poder contar con un laboratorio virtual que simule de manera gráfica los fenómenos físicos en estudio y que devuelva la resolución de los problemas que tú plantees?

El 100% de la población apoyaría la idea de contar con un laboratorio virtual, lo cual nos indica que el desarrollo de una herramienta de este tipo no sería rechazada por la población.

14. Si la universidad dispusiera la venta de una aplicación para PC como la planteada en la pregunta anterior y a un precio razonable, ¿Estarías dispuesto a adquirirla?

El 100% de la población respondió "Si". Concluyendo de esta manera que los alumnos sí están dispuestos a hacer uso de una herramienta de apoyo para PC.

15. ¿Cómo calificarías una metodología de estudio basada en la simulación de los fenómenos físicos mediante una aplicación en PC para apoyar lo desarrollado en papel?

Un 58% de la población respondió a esta pregunta "Interesante" y un 42% "Atractiva". Dándonos la idea que a los estudiantes le interesaría estudiar con una metodología diferente a la tradicional.

16. ¿Cuan beneficioso crees que sería para ti el poder contar con dicha herramienta de estudio?

Un 33% de la población respondió que sería "Muy beneficioso" y un 17% que "Ayudaría en algo". Concluyendo así que una herramienta de estudio como la mencionada tendría un beneficio para los estudiantes.

17. ¿Consideras que contar con herramientas de este tipo ayudaría a elevar aún más el nivel académico de la Universidad Don Bosco?

Un 84% de la población respondió que "Sí" y un 16% que "No". Lo cual nos indica que la mayoría de los estudiantes opinan que una herramienta de este tipo sí ayudaría a elevar aún más el nivel académico de la universidad.

CONCLUSIÓN GENERAL

Con el resultado de las encuestas realizadas se puede concluir lo siguiente:

- La materia de Física I posee un grado de dificultad elevado para la comprensión y entendimiento de los temas desarrollados en la misma.
- Es necesario ejemplificar de manera gráfica los fenómenos físicos para la comprensión de la materia.
- El desarrollo de laboratorios prácticos es de suma importancia para facilitar la comprensión de los fenómenos físicos.
- Es necesario utilizar una metodología de estudio diferente a la actual que ayude en gran manera a la comprensión del porqué de los fenómenos físicos.
- Es necesario el desarrollo de una herramienta que esté al alcance de los alumnos en cualquier momento que éstos lo necesiten.

c) **Cuestionario**

1. ¿En general, que opinión le merece el funcionamiento del simulador hasta el momento?

Un 58% de la población respondió "Excelente", un 30% "Muy bueno" y un 12% "Bueno".

Nos indica que un buen porcentaje de la población creen que el funcionamiento del simulador es excelente aunque aún existen partes en las cuales hay que mejorar.

2. ¿Qué áreas cree usted que se podrían mejorar?

Un 42% de la población respondió "Teoría", un 34% "Ayuda", un 16% "Ninguna" y un 8% "Simulación".

Nos indica que las partes de Teoría y Ayuda son áreas en las cuales se necesita mejorar.

3. En la parte teórica, ¿que aspectos deben mejorar?

Un 42% de la población respondió "Presentación", un 34% "Contenido" y un 24% "Ninguna".

Para la parte de teoría es necesario mejorar la presentación y el contenido de éstas.

4. En la parte de simulación, ¿qué opinión le merece el aspecto gráfico?

Un 40% de la población respondió "Excelente", otro 40% "Muy bueno" y un 10% "Bueno".

La parte de simulación tuvo bastante aceptación aunque hay aspectos que también se deben mejorar.

5. En cuanto al grado de dificultad operacional, ¿cómo calificaría las simulaciones?

Un 60% de la población respondió "Fácil" y un 40% "Accesible".

La estructura de las simulaciones fueron aceptadas y consideradas como fáciles y accesibles.

6. ¿Qué aspectos mejoraría en las simulaciones?

Un 50% de la población respondió "Ninguna", un 30% "Datos devueltos", un 10% "Captura de datos" y otro 10% "Aspecto gráfico".

Es necesario mejorar en los datos devueltos, captura de datos y en el aspecto gráfico de las simulaciones.

7. En su opinión ¿Cuan útil puede ser la aplicación en la comprensión de los fenómenos físicos estudiados?

El 100% de la población respondió "muy útil".

La población entera considera que la aplicación será de mucha utilidad para el estudio de la materia.

8. ¿Consideraría que contar con herramientas de este tipo ayudaría a elevar aún mas el nivel académico de la Universidad Don Bosco?

Un 80% de la población respondió "Ayudaría mucho" y un 20% "Ayudaría en algo".

La mayoría de la población considera que la herramienta en desarrollo ayudaría mucho para elevar aún más el nivel académico de la universidad.

CONCLUSIÓN GENERAL

En base a las respuestas obtenidas con el cuestionario logramos concluir lo siguiente:

- Era necesario mejorar la presentación para las aplicaciones de teoría y ayuda.
- Se necesitaba profundizar en el desarrollo de los contenidos teóricos.
- Era necesario mejorar la captura de datos, los datos devueltos y el aspecto gráfico para las simulaciones prácticas.
- Las expectativas de la aplicación son muy buenas en caso de llevar a cabo la implementación de la misma.

5.2.1.1 IDENTIFICACIÓN DE REQUISITOS

La figura 5.2 ilustra la descomposición horizontal de software, la cual brinda una mejor visión del comportamiento del mismo. En ésta se observan las cuatro funciones principales que el sistema es capaz de ejecutar y que son básicas para el cumplimiento de sus objetivos.

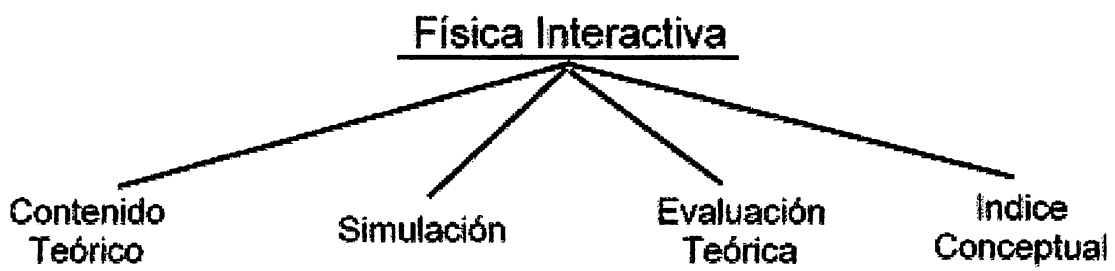


Figura 5.2 Descomposición horizontal

Para la etapa de diseño, se muestra a continuación el diagrama de jerarquías. Este proporciona un mapa que permite localizar un módulo del programa existente dentro del sistema principal.

5.2.1.1.1 DIAGRAMAS JERÁRQUICOS

Los diagramas jerárquicos que a continuación se muestran, representan el nivel de cada proceso ejecutado por los diferentes módulos de la aplicación.

El primer diagrama (Figura 5.3) muestra el primer nivel de sistema, el cual esta dividido en los diferentes módulos que indican las funciones básicas que se ejecutan.

Los siguientes diagramas muestran la descomposición de cada uno de estos módulos en sus diferentes procesos.

5.2.1.1.1.1 DIAGRAMA JERÁRQUICO GENERAL

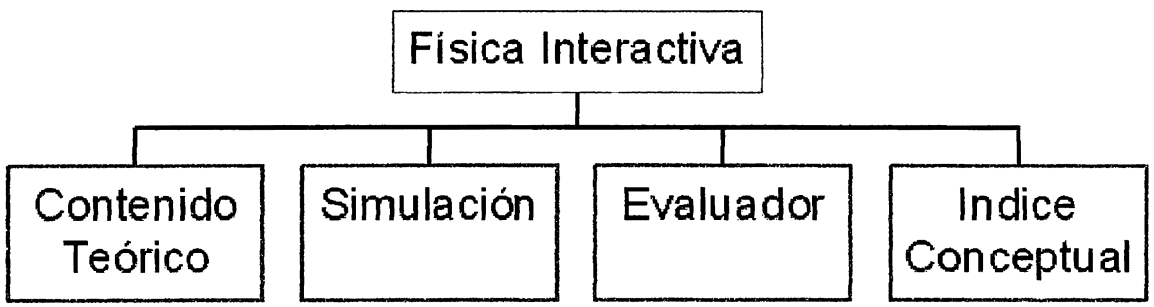


Figura 5.3 Diagrama jerárquico general

5.2.1.1.1.2 DIAGRAMA JERÁRQUICO DE CONTENIDO TEÓRICO

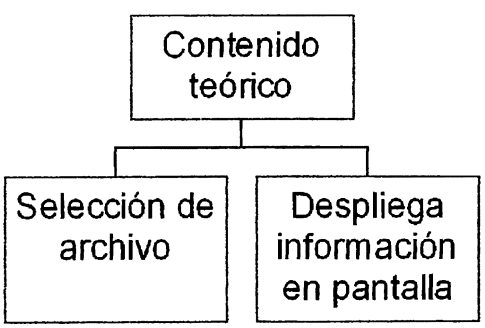


Figura 5.4 Diagrama para Módulo Contenido teórico

5.2.1.1.1.3 DIAGRAMA JERÁRQUICO DE SIMULACIÓN

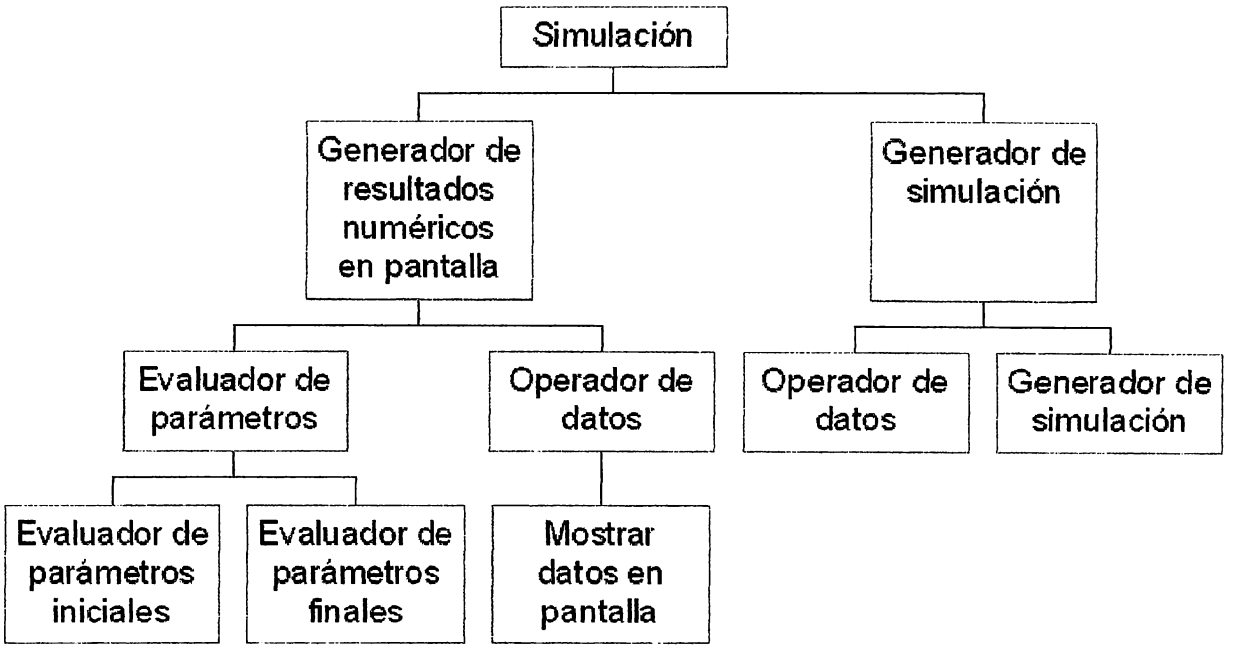


Figura 5.5 Diagrama para Módulo Simulación

5.2.1.1.1.4 DIAGRAMA JEPARQUICO DE EVALUADOR

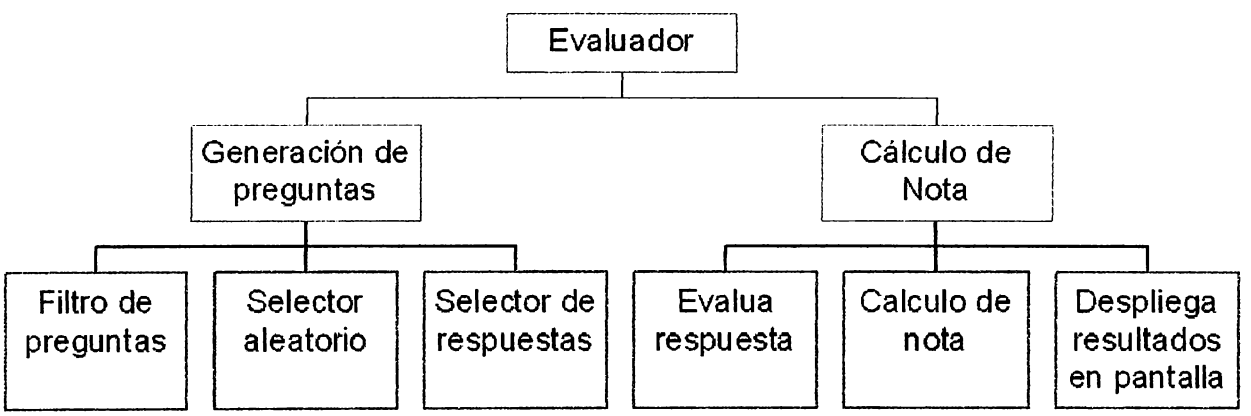


Figura 5.6 Diagrama para Módulo Evaluador

5.2.1.1.1.5 DIAGRAMA JERÁRQUICO DE ÍNDICE CONCEPTUAL

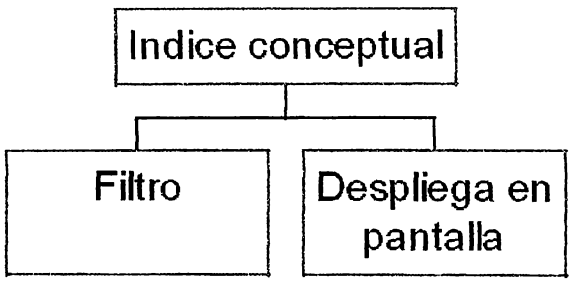


Figura 5.7 Diagrama para Módulo Índice Conceptual

5.2.1.1.2 DIAGRAMAS DE FLUJO

Los Diagramas de Flujos de Datos presentados a continuación representan el flujo de la información y las transformaciones de esta, así como el origen de datos a transformar y las diferentes salidas.

5.2.1.1.2.1 DFD A NIVEL DE CONTEXTO

El DFD de nivel de contexto indica la función básica del sistema, representando una única entrada de datos y cuatro posibles salidas después de someterse a diferentes procesos no detallados aún.

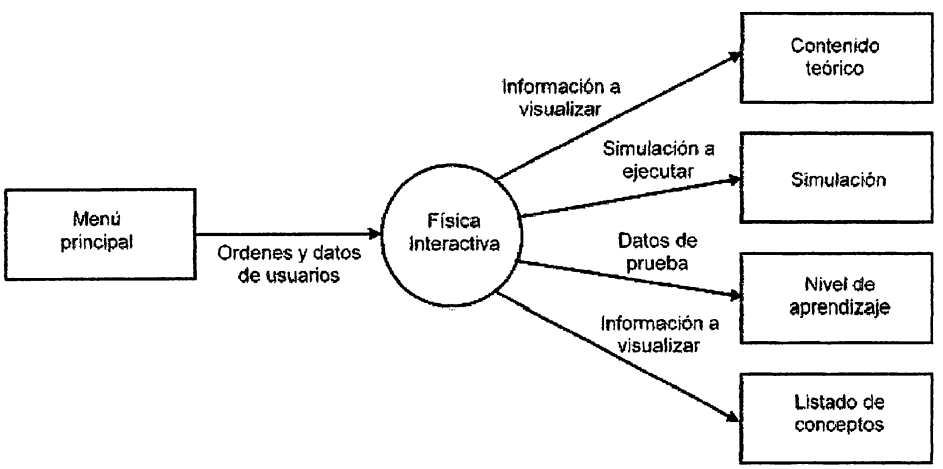


Figura 5.8 DFD a nivel de contexto

5.2.1.1.2.2 DFD DE NIVEL UNO

El DFD de nivel uno divide la transformación de la entrada en dos partes, donde se indica la unidad de estudio la cual genera un proceso de selección de módulo a ejecutar.

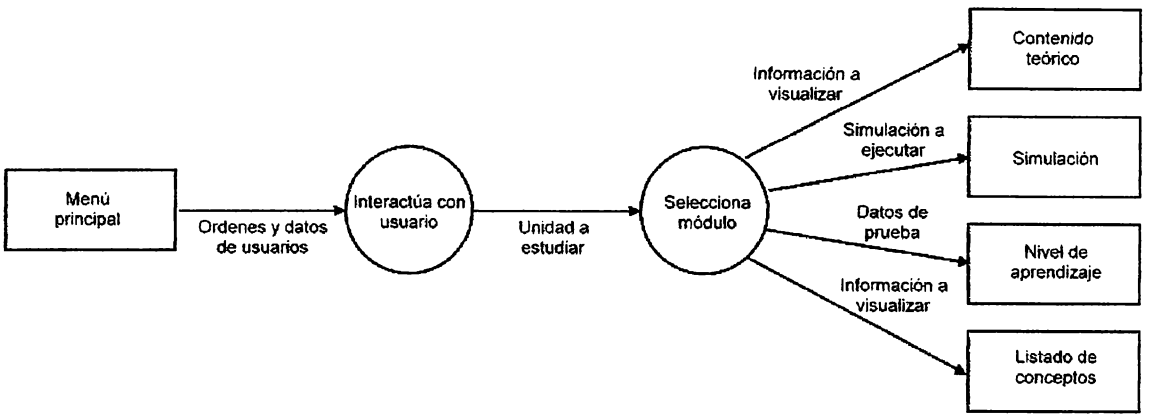


Figura 5.9 DFD a nivel 1

5.2.1.1.2.3 DFD A NIVEL 2, PROCESO SELECCIÓN DE MÓDULO

El DFD de nivel 2 divide el proceso de selección de módulo en dos subprocesos más, los cuales reciben determinados parámetros y devuelven la salida respectiva.

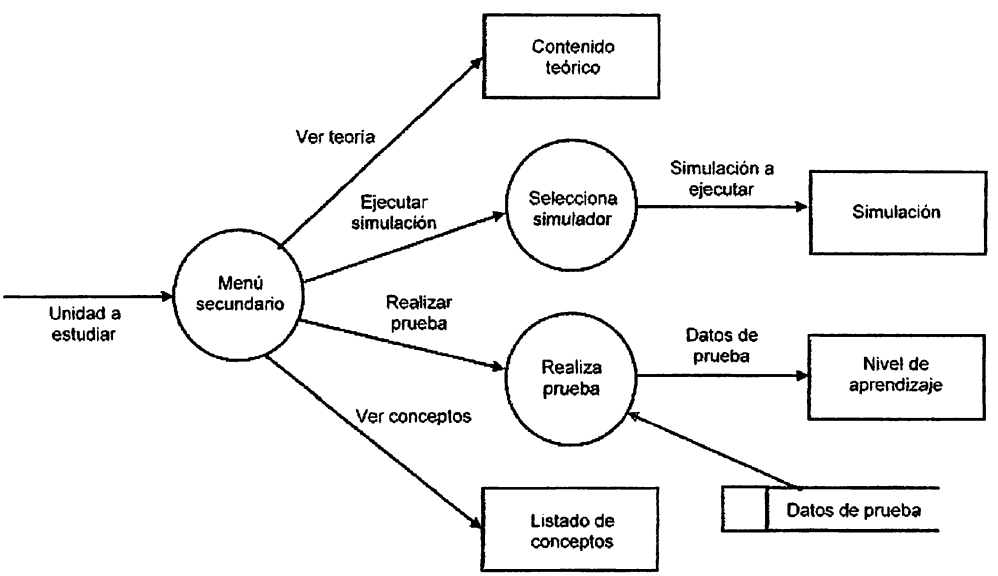


Figura 5.10 DFD a nivel 2

5.2.1.1.2.4 DFD A NIVEL 3, PROCESO REALIZA PRUEBA

Por último el DFD de nivel 3, muestra el proceso de realizar prueba, donde recibe como parámetro el número de preguntas según indique el usuario, estas preguntas las obtiene de la base de datos y las procesa para obtener la respuesta a cada una ellas, luego calcula una nota que convierte en la salida respectiva (resultado o nivel de aprendizaje)

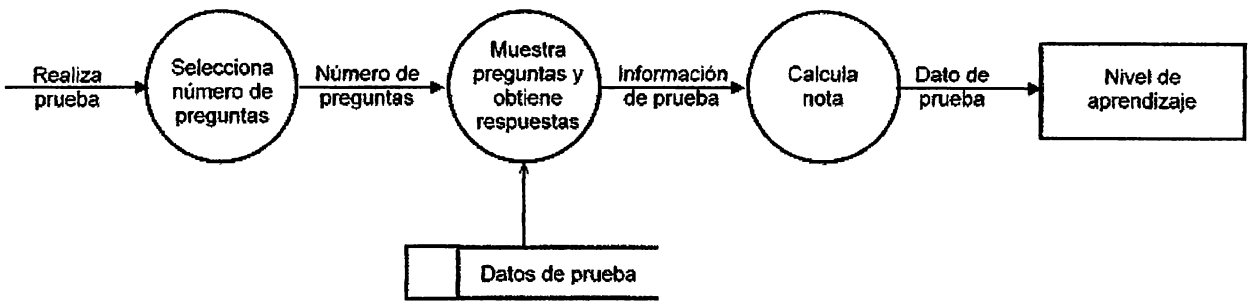


Figura 5.11 DFD a nivel 3

5.2.2 DISEÑO

5.2.2.1 DISEÑO DE DATOS

LISTADO DE ENTIDADES

Nombre de la Tabla	Descripción
F_Unid	Unidades en que está dividido el contenido de la aplicación.
F_Index	Índice de conceptos
F_TsHd	Cuestionario de módulo evaluador
F_TsDt	Listado de posibles respuestas de cuestionario

TABLA: F_Unid

Llave	Columna	Tipo Dato	Largo	Descripción
PK	IdUnidad	ANum	2	Código de unidad
	Descrip	Char	50	Nombre de la unidad

TABLA: F_Index

Llave	Columna	Tipo Dato	Largo	Descripción
PK	IdCpto	ANum	2	Código de concepto
	Cpto	Char	20	Concepto
	Descrip	Char	250	Definición del concepto
	IdUnidad	ANum	2	Unidad a la que pertenece

TABLA: F_TsHd

Llave	Columna	Tipo Dato	Largo	Descripción
PK	IdQst	ANum	4	Código de pregunta
	Descrip	Char	100	Interrogante
	IdUnidad	ANum	2	Unidad a la que pertenece
	Nansw	Num	1	Número de respuesta correcta

TABLA: F_TsDt

Llave	Columna	Tipo Dato	Largo	Descripción
PK	IdQst	Anum	4	Código de pregunta
	Nansw	Num	1	Número de respuesta correcta
	Descrip	Char	200	Descripción de respuesta

5.2.2.1.1 DIAGRAMA ENTIDAD – RELACIÓN

La figura 5.12 nuestra el diagrama entidad relación de las entidades del punto 5.2.2.1

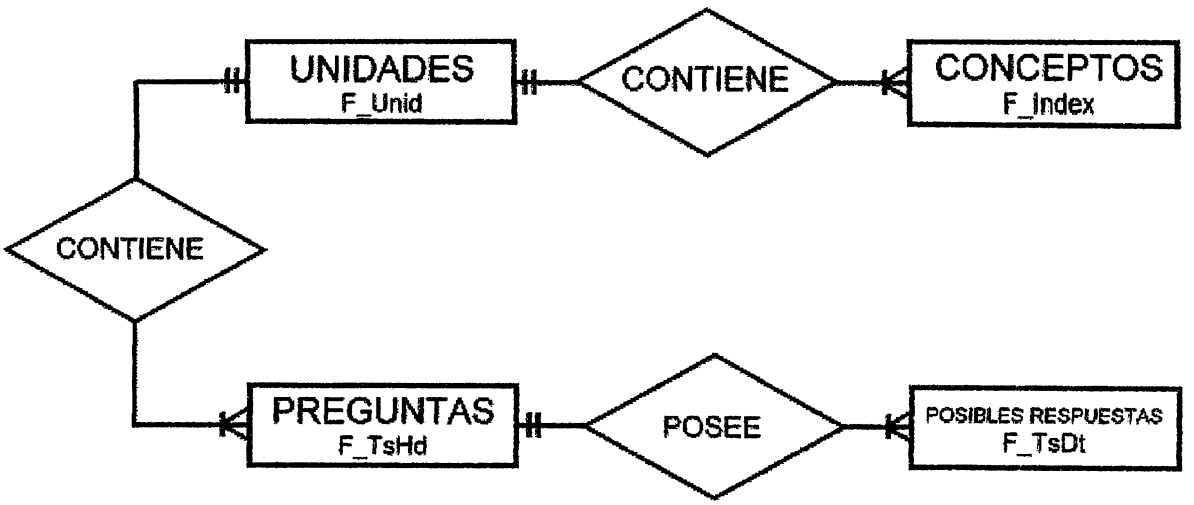


Figura 5.12 Diagrama Entidad – Relación

5.2.2.2 COMPORTAMIENTO DE MÓDULOS DE LA APLICACIÓN

A continuación se presenta la información que cada función del sistema utiliza para generar la respectiva salida y sus diferentes procesos, de acuerdo a la entrada recibida.

5.2.2.2.1 CONTENIDO TEÓRICO

Salidas

Nombre	Descripción	Tipo
Tema	Almacena nombre de archivo .rtf con información teórica	string

Entradas

Nombre	Descripción	Tipo
QueUnid	Indica el número de unidad en estudio	Integer
QuePag	Indica la página de contenido teórico	Integer

Procesos

Nombre	Proceso	Tipo
Tema	BuscaTema(QueUnid,QuePag)	string

5.2.2.2.2 SIMULACIÓN

- **SIMULACIÓN DE VECTORES**

Salidas

Nombre	Descripción	Tipo
X1	Origen de resultante en x.	Double
Y1	Origen de resultante en y.	Double
X2	Fin de resultante en x.	Double
Y2	Fin de resultante en y.	Double

Entradas

Nombre	Descripción	Tipo
Xa	Coordenada en x del vector A	Double
Xb	Coordenada en x del vector B	Double
Ya	Coordenada en y del vector A	Double
Yb	Coordenada en y del vector B	Double
E	Escalar	Double

Procesos

Nombre	Proceso	Tipo
Rsuma	(X1+X2), (Y1+Y2)	Double
Rresta	(X1-X2), (Y1-Y2)	Double
Rmulti	(E*X1,E*Y1)	Double

- **SIMULACIÓN DE MOVIMIENTO HORIZONTAL**

Salidas

Nombre	Descripción	Tipo
Vf	Velocidad final	Double
PosX	Posición final en X	Double

Entradas

Nombre	Descripción	Tipo
Vo	Velocidad inicial	Double
A	Aceleración	Double

Procesos

Nombre	Proceso	Tipo
Posx	(Vo * tiempo) + (½ A * tiempo^2)	Double
Vf	Vo + (A * tiempo)	Double

- **SIMULACIÓN DE MOVIMIENTO VERTICAL**

Salidas

Nombre	Descripción	Tipo
Vf	Velocidad final	Double
PosY	Posición final en Y	Double

Entradas

Nombre	Descripción	Tipo
Vo	Velocidad inicial	Double
A	Aceleración	Double

Procesos

Nombre	Proceso	Tipo
PosY	(Vo * tiempo) + (½ A * tiempo^2)	Double
Vf	Vo + (A * tiempo)	Double

- **SIMULACIÓN DE MOVIMIENTO PARABÓLICO**

Salidas

Nombre	Descripción	Tipo
Vx	Velocidad final en X	Double
Vy	Velocidad final en Y	Double
PosX	Posición final en X	Double
PosY	Posición final en Y	Double

Entradas

Nombre	Descripción	Tipo
Vox	Velocidad inicial en X	Double
Voy	Velocidad inicial en Y	Double
A	Aceleración	Double

Procesos

Nombre	Proceso	Tipo
PosY	(Vo * tiempo) + (½ A * tiempo^2)	Double
PosX	Vx * tiempo	Double
Vy	Vo + (A * tiempo)	Double

- **SIMULACIÓN DE MOVIMIENTO CIRCULAR**

Salidas

Nombre	Descripción	Tipo
Ang	Posición angular	Double
W	Velocidad angular	Double
A	Aceleración centrípeta	Double
T	Período	Double
F	Frecuencia	Double
S	Longitud de arco	Double
Rev	Revoluciones	Double

Entradas

Nombre	Descripción	Tipo
Radio	Radio de la circunferencia	integer
Rapidez	Rapidez angular inicial	Double

Procesos

Nombre	Proceso	Tipo
Ang	((W + Wo) / 2) * Tiempo	Double
W	Wo + (A * Tiempo)	Double
T	T = 2 * 3.1416*r / W	Double
A	V^2 / r	Double
F	1 / T	Double
S	Radio * Ang	Double
Rev	Ang / (3.1416 * 2	Double

SIMULACIÓN DE NEWTON

Salidas

Nombre	Descripción	Tipo
Tiempo	Tiempo	Double
A	Aceleración	Double
Vf	Velocidad	Double
Pos	Desplazamiento	Double

Entradas

Nombre	Descripción	Tipo
M1	Masa1	integer
M2	Masa2	Double
Ur	Coeficiente de Rozamiento	Double

Procesos

Nombre	Proceso	Tipo
Pos	(Vo * tiempo) + + (½ A * $tiempo^2$)	Double
Vf	Vo + (A * tiempo)	Double
A	(F - Fr) / m1	Double

- **SIMULACIÓN DE TRABAJO Y ENERGÍA**

Salidas

Nombre	Descripción	Tipo
Em	Energía mecánica	Double
Ep	Energía potencial	Double
Ek	Energía cinética	Double
X	Desplazamiento	Double
V	Velocidad	Double
W	Trabajo del resorte	Double

Entradas

Nombre	Descripción	Tipo
K	Constante del resorte	Integer
F	Fuerza	Double
M	Masa	Double
Ur	Coeficiente de rozamiento	Double

Procesos

Nombre	Proceso	Tipo
Em	(1 / 2) * k * (Xo ^ 2) * Math.Exp((-Fr * Tiempo) / m)	Double
Ep	0.5 * k * X * X	Double
Ek	Em - Ep	Double
X	Xo * Math.Exp((-Fr * Tiempo) / (2 * m)) * Math.Cos(w * Tiempo)	Double
V	((2 * Ek) / m) ^ (1 / 2)	Double
W	(1 / 2) * k * (X^2)	Double

- **SIMULACIÓN DE COLISIONES**

Salidas

Nombre	Descripción	Tipo
Vf1	Velocidad final masa1	Double
Vf2	Velocidad final masa2	Double
P1	Cantidad de movimiento	Double

Entradas

Nombre	Descripción	Tipo
Vi1	Velocidad inicial masa1	Double
Vi2	Velocidad inicial masa2	Double
M1	Masa 1	Double
M2	Masa 2	Double

Procesos

Nombre	Proceso	Tipo
Vf1	(((m1 - m2) / (m1 + m2)) * Vo1) + (((2 * m2) / (m1 + m2)) * Vo2)	Double
Vf2	(((2 * m1) / (m1 + m2)) * Vo1) + (((m2 - m1) / (m1 + m2)) * Vo2)	Double
P1	mv	Double

- **SIMULACIÓN DE MOVIMIENTO ROTACIÓN**

Salidas

Nombre	Descripción	Tipo
Wf	Velocidad angular final	Double
Af	Aceleración angular final	Double
T	Período	Double
S	Longitud de arco	Double
θ	Ángulo en grados	Double
F	frecuencia	Double
Rev	Revoluciones	Double

Entradas

Nombre	Descripción	Tipo
R	radio	Double
W	Velocidad angular inicial	Double
A	Aceleración angular inicial	Double

Procesos

Nombre	Proceso	Tipo
Wf	Wo + (A * Tiempo)	Double
T	2 * 3.1416 / W	Double
S	Radio * Ang	Double
θ	Ang * 180 / 3.1416	Double
F	1 / T	Double
Rev	Ang / (3.1416 * 2	Double

5.2.2.2.3 EVALUACIÓN TEÓRICA

Salidas

Nombre	Descripción	Tipo
Nota	Devuelve el resultado de la evaluación.	Double

Entradas

Nombre	Descripción	Tipo
Nqst	Indica el Número de Preguntas de la Prueba	Integer
Acum	Acumula el número de preguntas acertadas	Integer

Procesos

Nombre	Proceso	Tipo
Nota	(Acum / Nqst)*10	Double

.2.2.2.4 INDICE CONCEPTUAL

Salidas

Nombre	Descripción	Tipo
Índice	Lista y Definición de conceptos	String

Entradas

Nombre	Descripción	Tipo
QueUnid	Indica el número de unidad en estudio	Integer

Procesos

Nombre	Proceso	Tipo
Índice	Select * from Index order by cpto	String

.2.2.3 FLUJOGRAMAS

Los siguientes esquemas representa las estructura de la codificación utilizada para la construcción e los diferentes programas que componen la aplicación. Algunas estructuras son comunes para iferentes procesos debido a la similitud del comportamiento de los mismos.

5.2.2.3.1 FLUJOGRAMA PARA SIMULACIÓN DE VECTORES

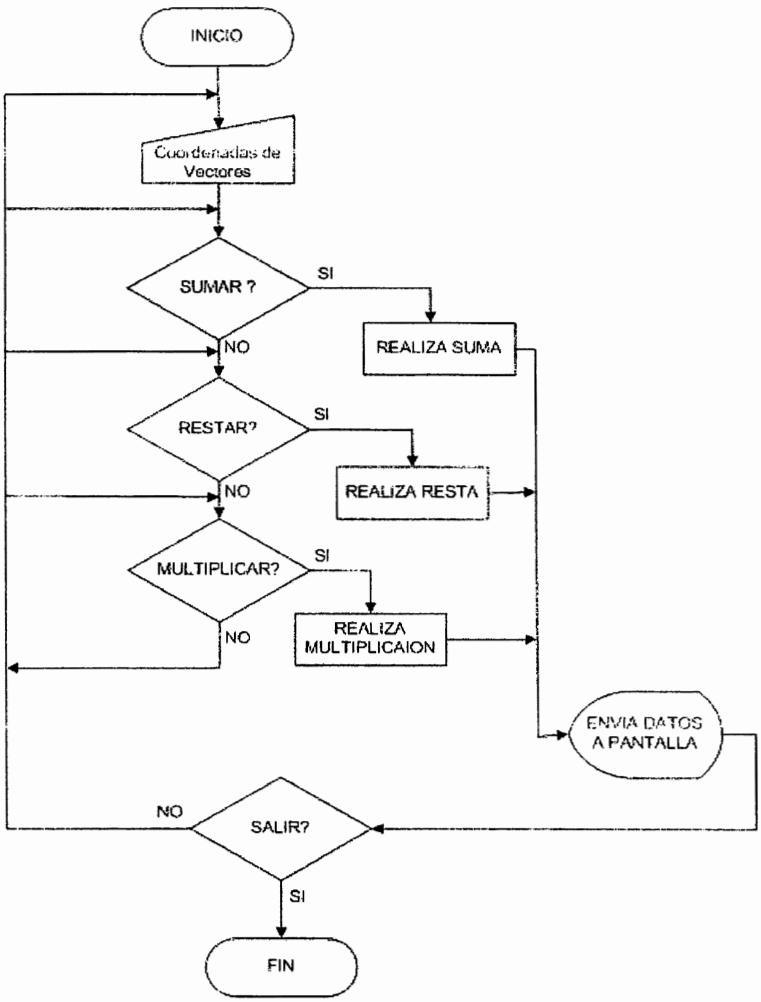


Figura 5.12 Flujograma para vectores

5.2.2.3.2 FLUJOGRAMA PARA SIMULACIONES DE MOVIMIENTO HORIZONTAL, VERTICAL, PARABÓLICO, CIRCULAR Y ROTACIÓN.

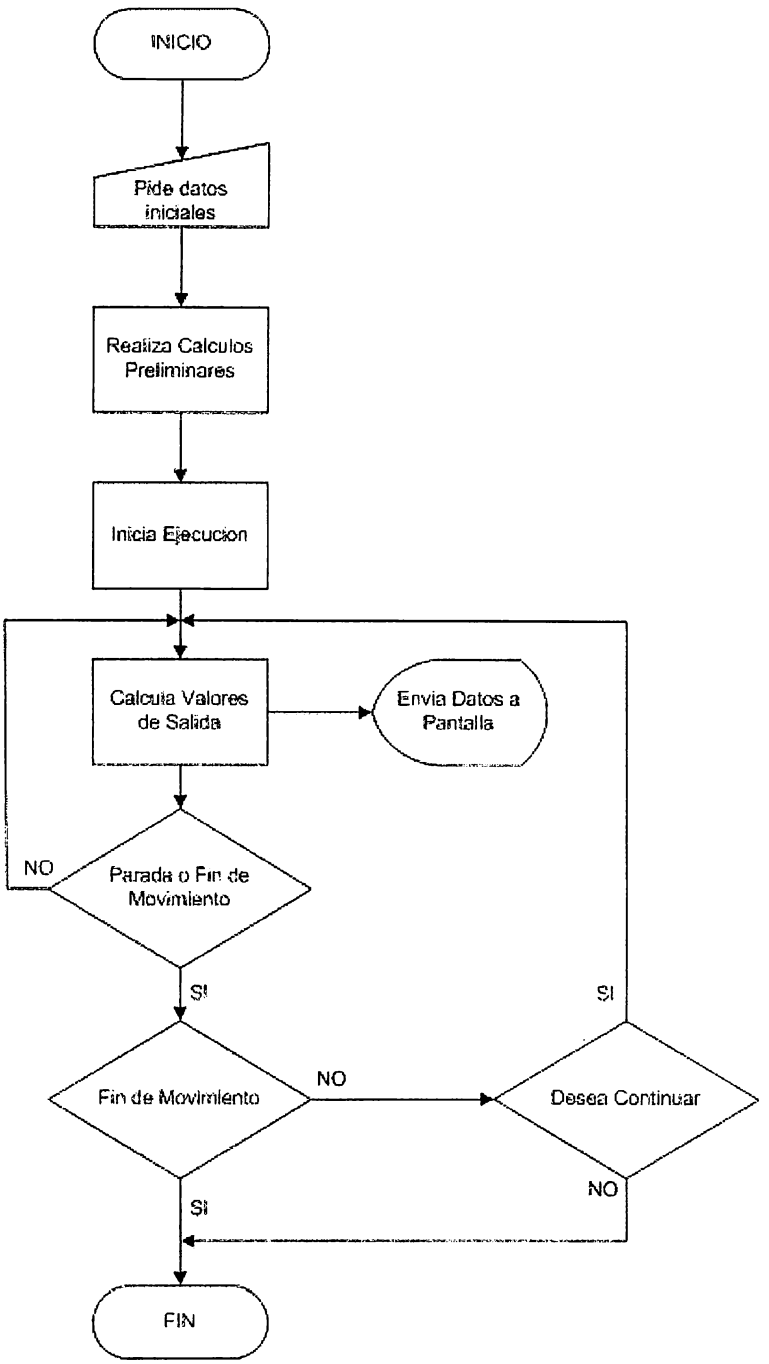


Figura 5.13 Flujograma para Movimiento horizontal, vertical, parabólico, circular y rotación.

5.2.2.3.3 FLUJOGRAMA PARA SIMULACIÓN DE FUERZA Y MOVIMIENTO

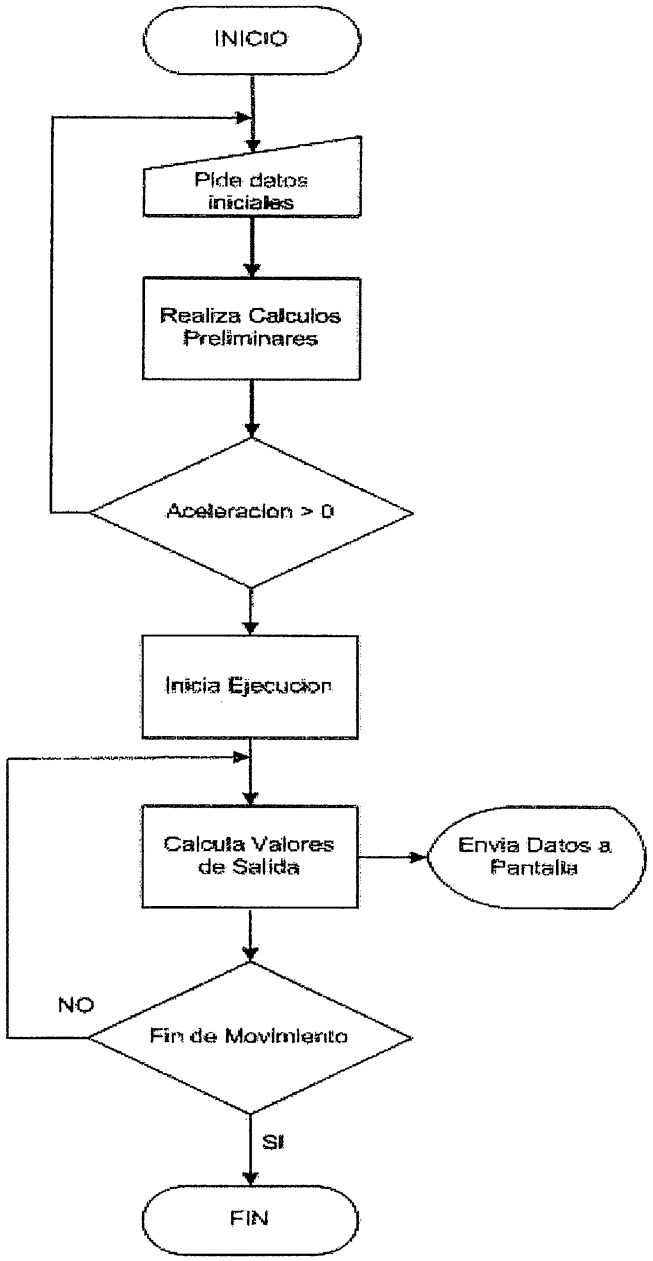


Figura 5.14 Flujograma para Fuerza y Movimiento.

5.2.2.3.4 FLUJOGRAMA PARA SIMULACIÓN DE TRABAJO Y ENERGÍA

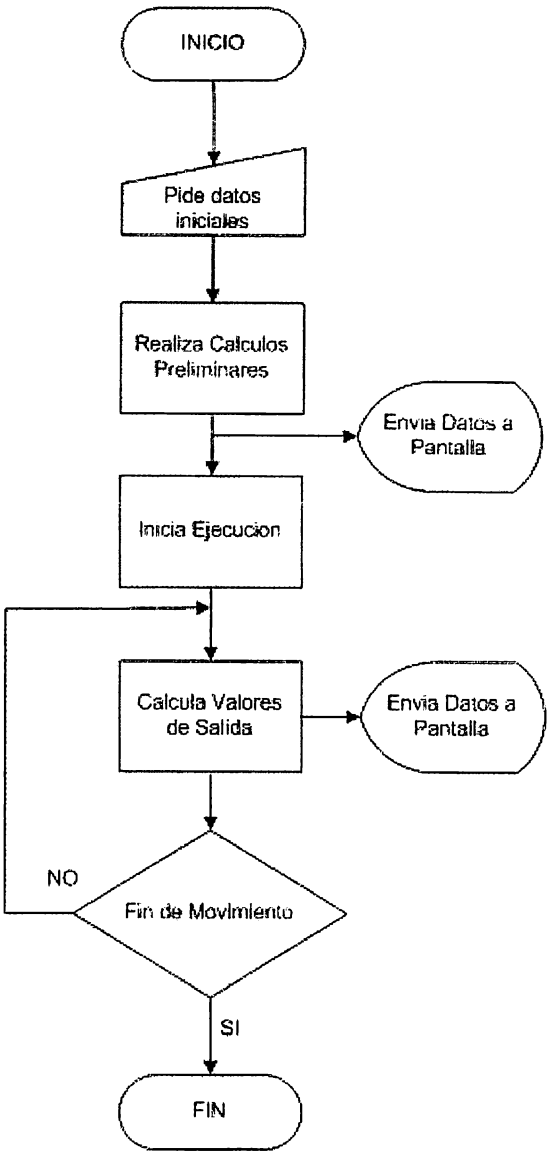


Figura 5.15 Flujograma para Trabajo y Energía

5.2.2.3.5 FLUJOGRAMA PARA SIMULACIÓN DE COLISIONES

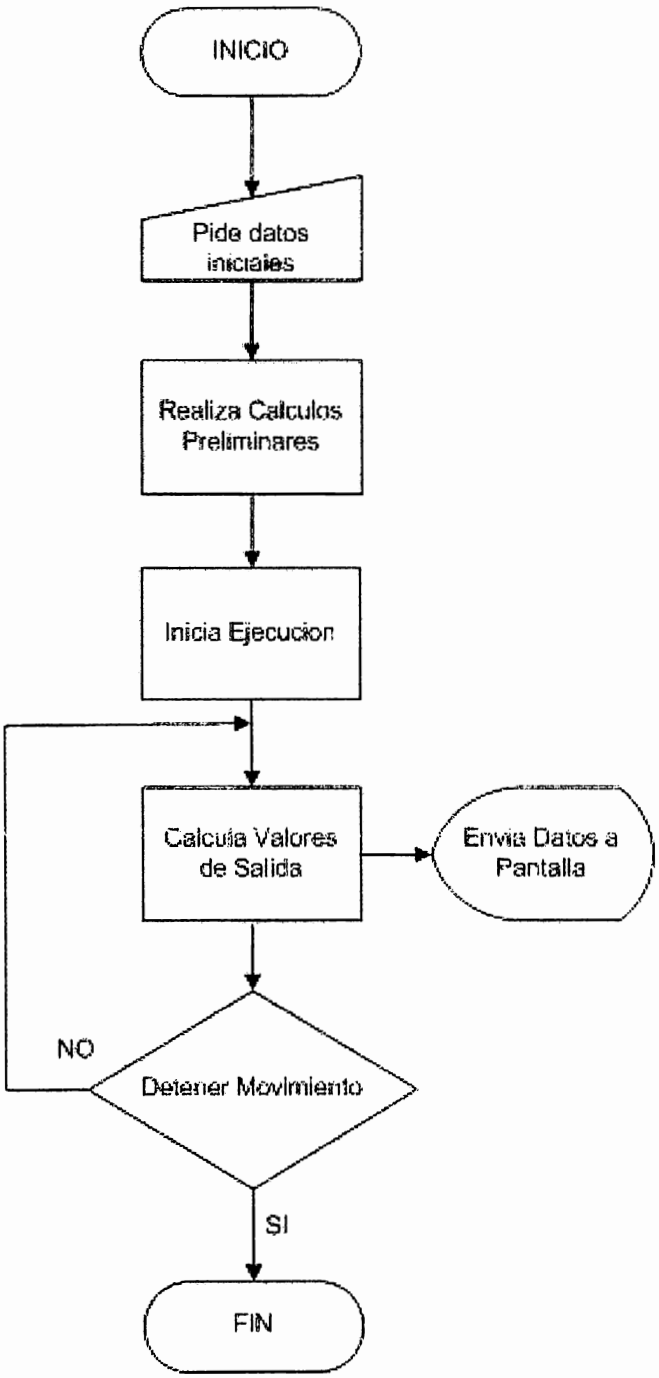


Figura 5.16 Flujograma para Colisiones

5.2.2.3.6 FLUJOGRAMA PARA PROGRAMA EVALUADOR

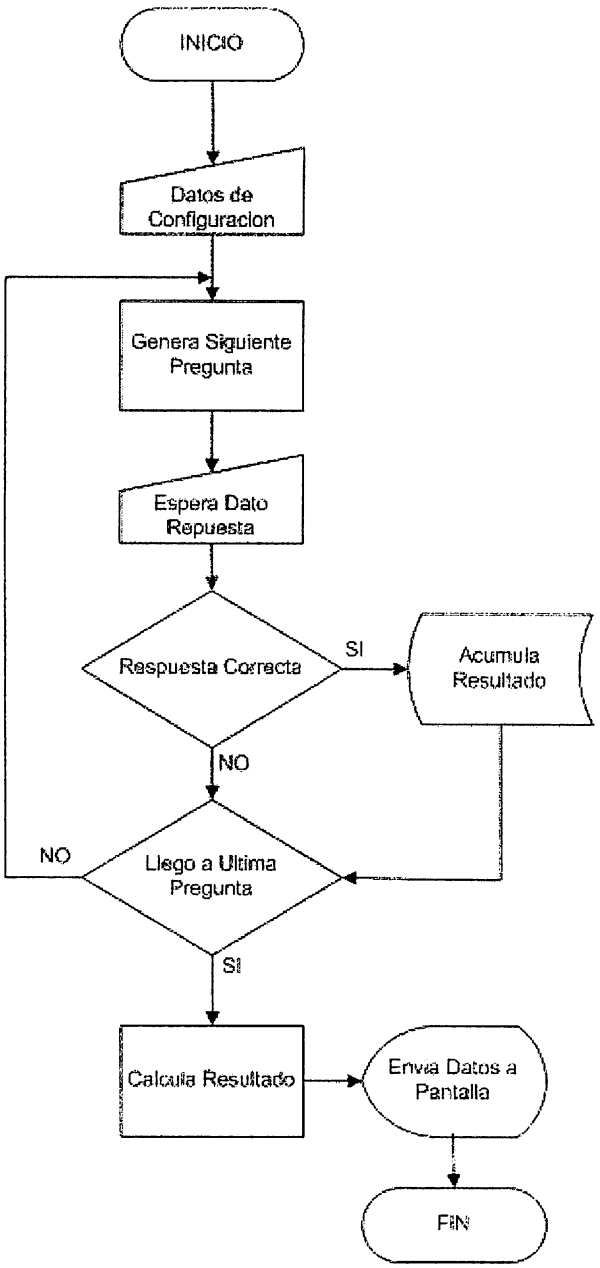


Figura 5.17 Flujograma para Evaluador

5.2.2.4 DISEÑO DE INTERFAZ

5.2.2.4.1 DISEÑO DE MENU

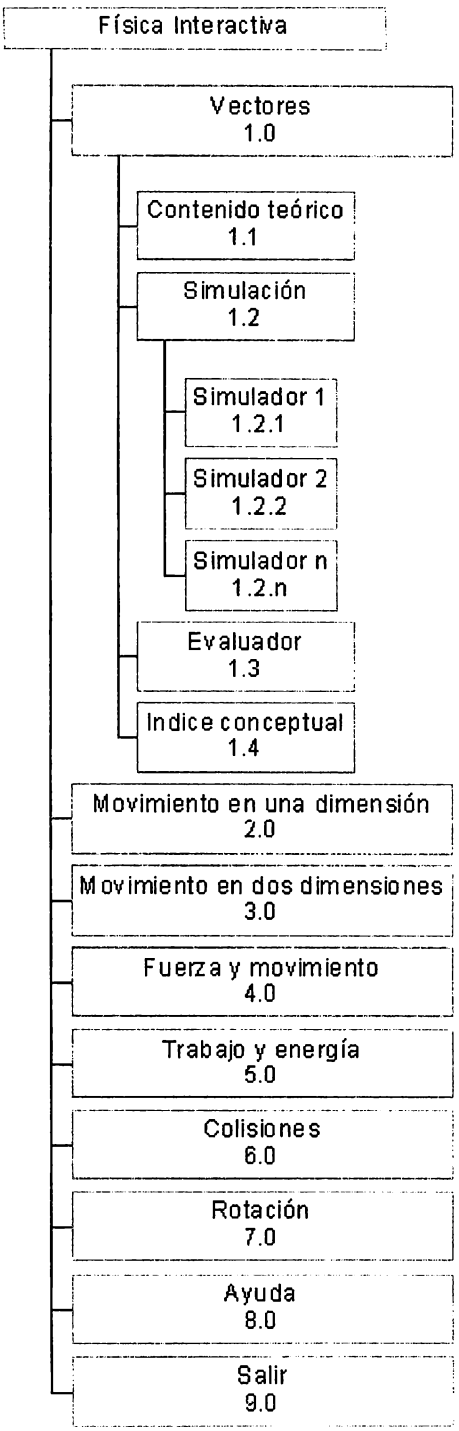


Figura 5.18 Estructura jerárquica del Menú

En la figura 5.17 se presenta la estructura jerárquica del menú contemplado en la aplicación, el cual consiste en tener en el primer nivel del menú las siete unidades de estudio de la aplicación junto con la opción salir.

En la figura 5.17 únicamente se refleja un submenú para la unidad de Vectores que es el punto 1.0, este submenú es para cada unidad del sistema, es decir desde el punto 1.0 al 7.0 llevan el mismo submenú. Dicho submenú consta de: Contenido teórico, Simulación, Evaluación, Índice. Para simulación se tiene otro submenú que consta del número de prácticas que contiene la unidad.

5.2.2.4.2 INTERFACES DEL SISTEMA

A continuación se detallan las principales pantallas de la aplicación, mismas que se explican con más detalle en el manual del usuario. (Ver anexo VII).

PANTALLA PRINCIPAL (main.vb)

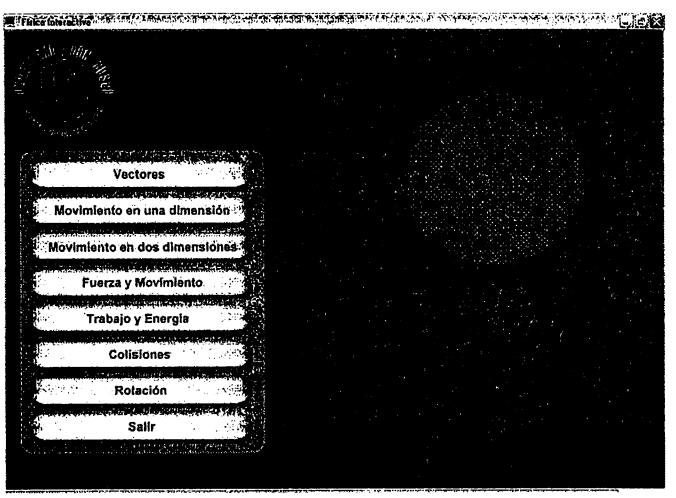


Figura 5.19 Pantalla principal de la aplicación

SUBMENÚ DE UNIDADES (Unidad de Vectores activa)

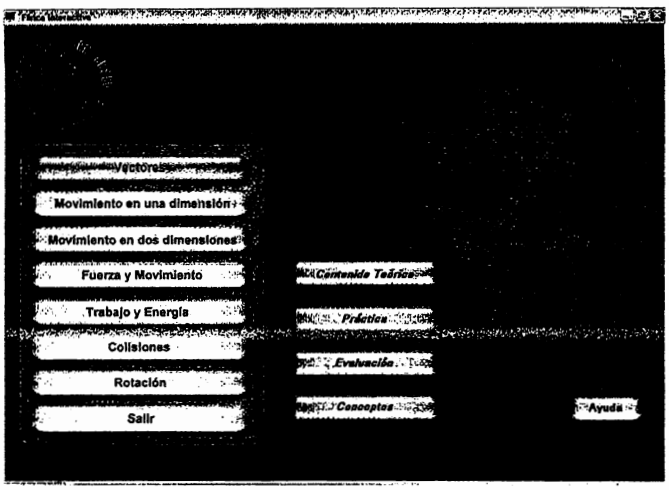


Figura 5.20 Submenú de unidades

MENÚ DE AYUDA (contenidohelp.vb)

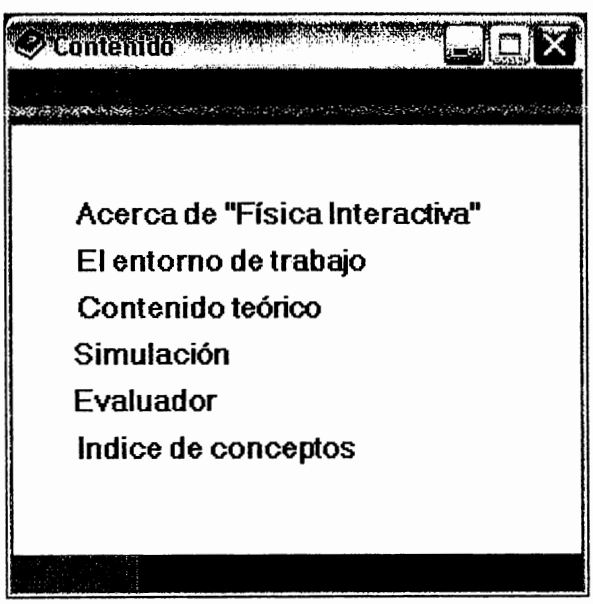


Figura 5.21 Menú de ayuda

DESARROLLO DE CONTENIDO TEÓRICO (teoría.vb)

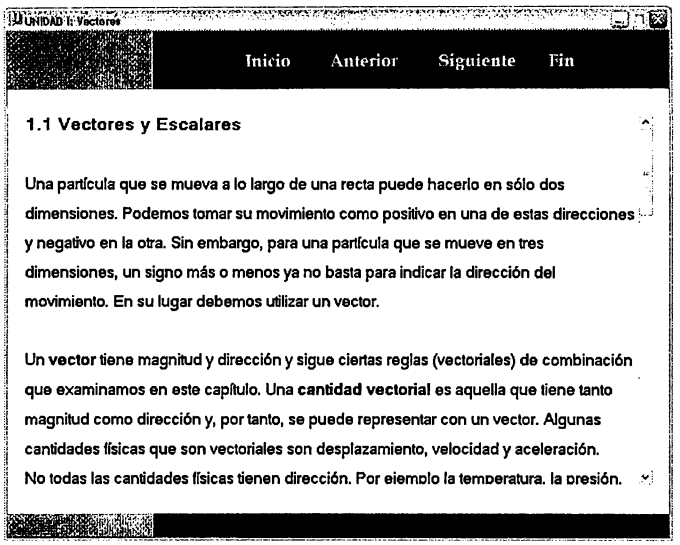


Figura 5.22 Pantalla para contenido teórico

PANTALLA PARA SIMULACIÓN DE VECTORES (vectores.vb)

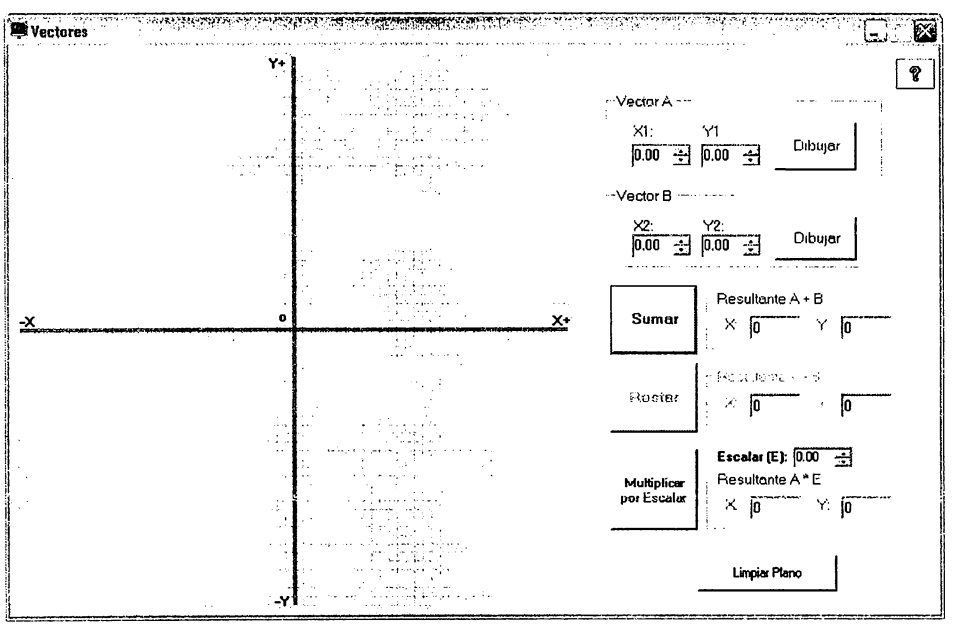


Figura 5.23 Pantalla para contenido teórico

PANTALLA PARA SIMULACIÓN DE MOVIMIENTO HORIZONTAL (acelera.vb)

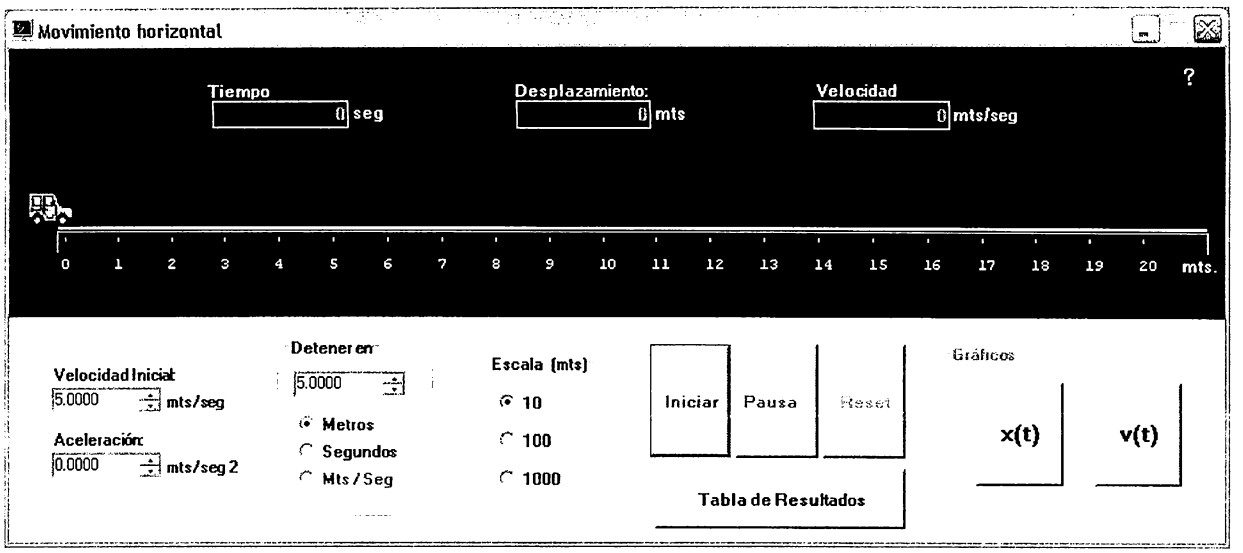


Figura 5.24 Pantalla simulación de movimiento horizontal

PANTALLA PARA SIMULACIÓN DE MOVIMIENTO VERTICAL (caidalibre.vb)

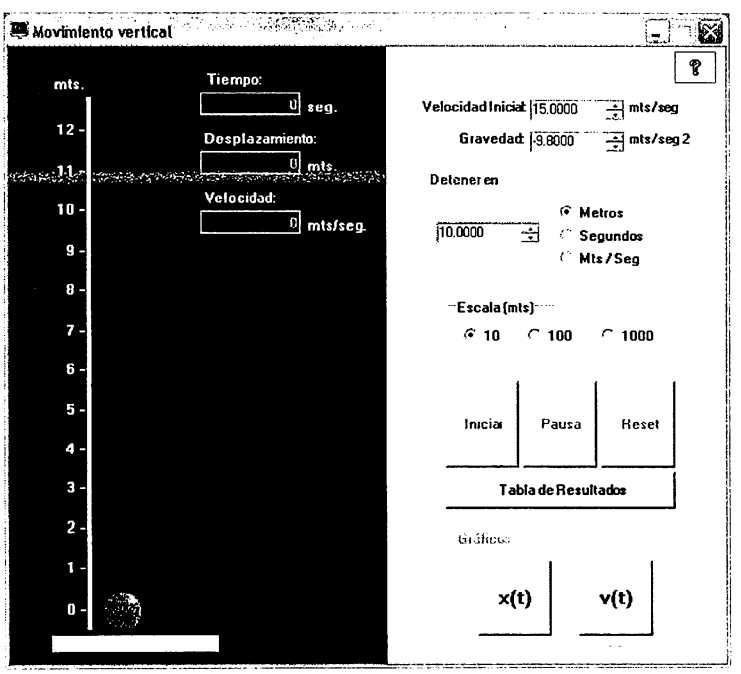


Figura 5.25 Pantalla simulación de movimiento vertical

PANTALLA PARA SIMULACIÓN DE MOVIMIENTO PARABOLICO (tipoparab.vb)

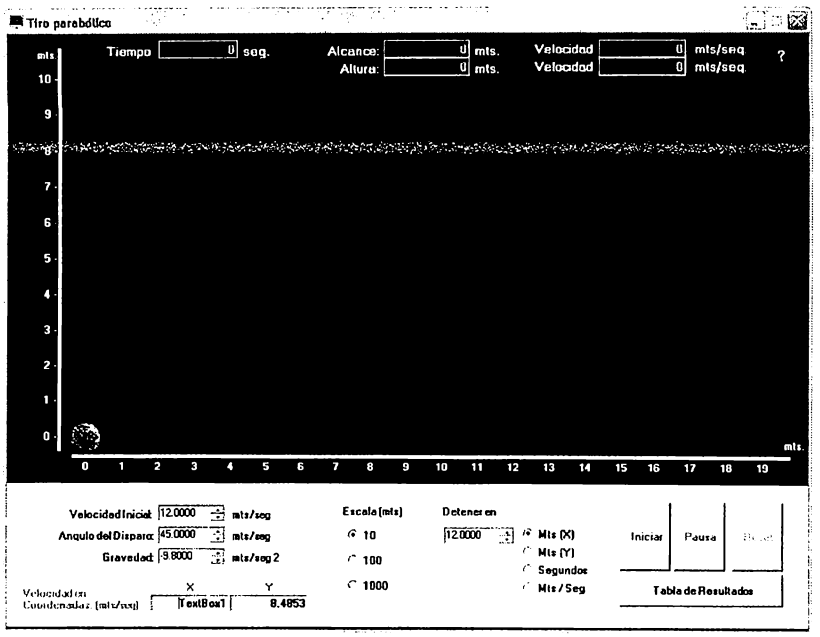


Figura 5.26 Pantalla simulación de movimiento parabólico

PANTALLA PARA SIMULACIÓN DE MOVIMIENTO CIRCULAR (movcirc.vb)

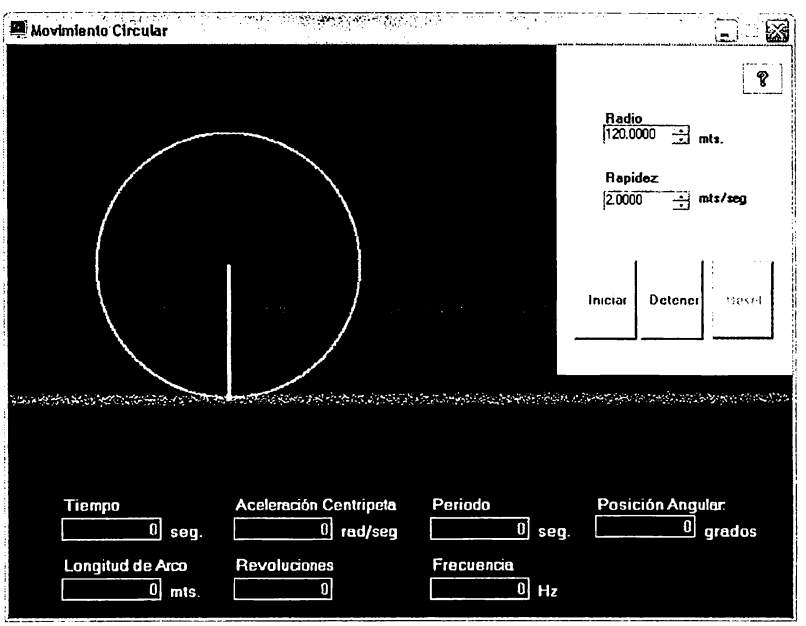


Figura 5.27 Pantalla simulación de movimiento circular

PANTALLA PARA SIMULACIÓN DE LEYES DE NEWTON (newton.vb)

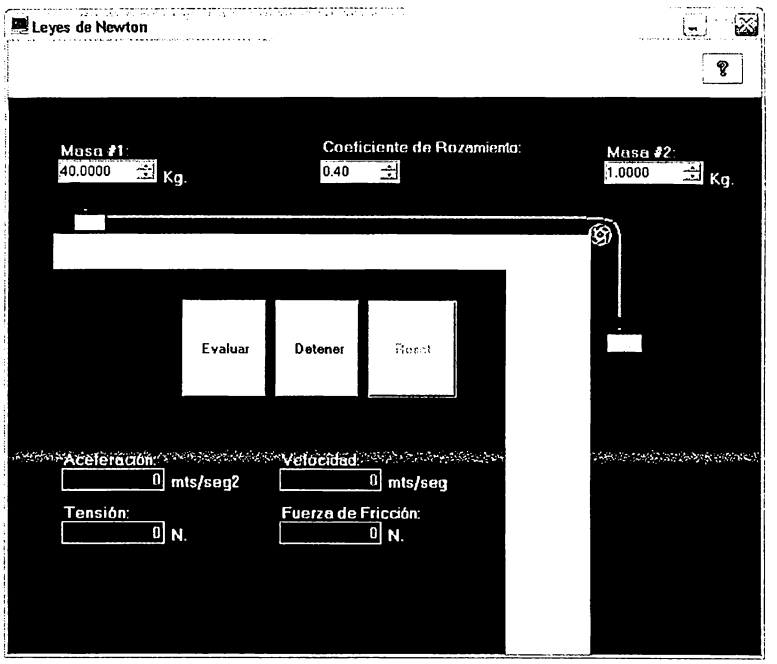


Figura 5.28 Pantalla simulación de leyes de Newton

PANTALLA PARA SIMULACIÓN DE CONSERVACIÓN DE LA ENERGÍA (resorte.vb)

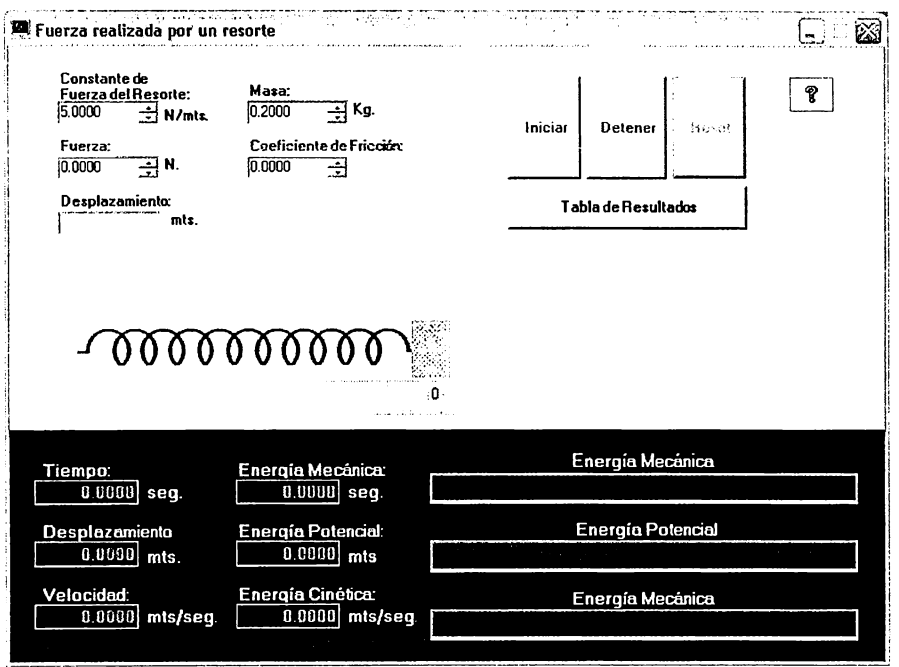


Figura 5.29 Pantalla simulación de conservación de la energía

PANTALLA PARA SIMULACIÓN DE COLISIONES (choque1.vb)

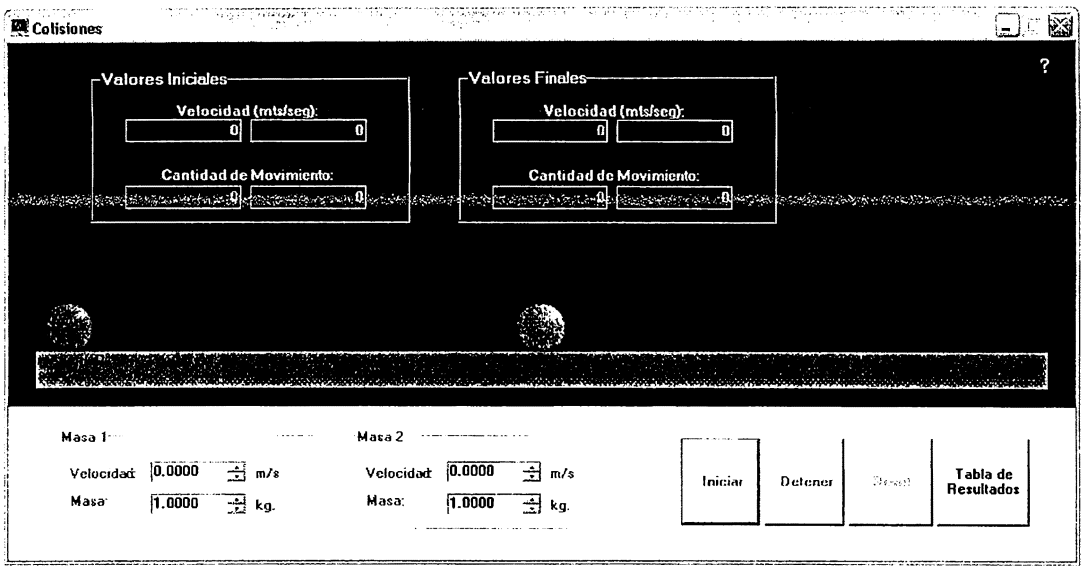


Figura 5.30 Pantalla simulación de colisiones

PANTALLA PARA SIMULACIÓN DE ROTACIÓN (rotacion.vb)

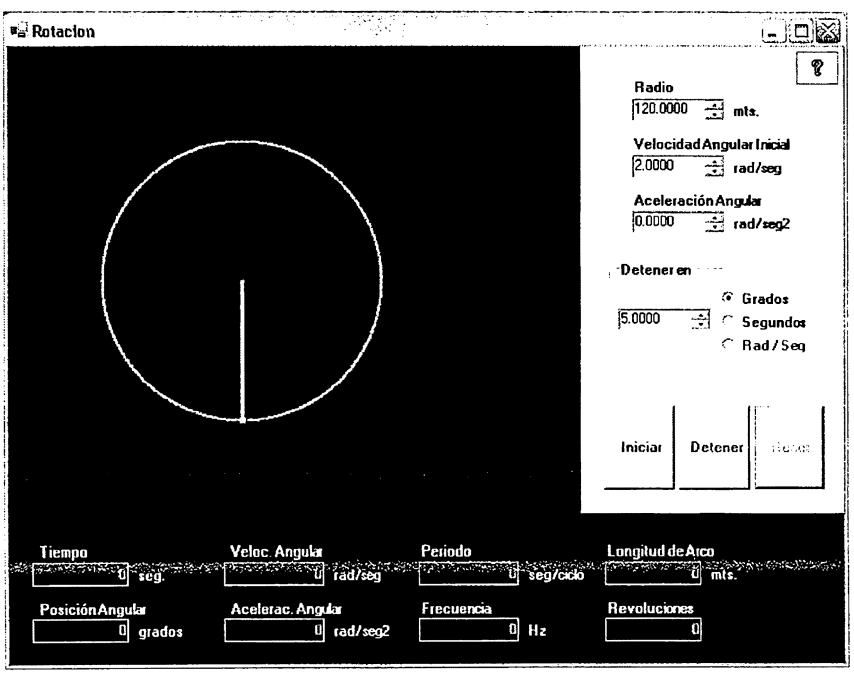


Figura 5.31 Pantalla simulación de rotación

PANTALLA PARA EVALUADOR (evalua.vb)

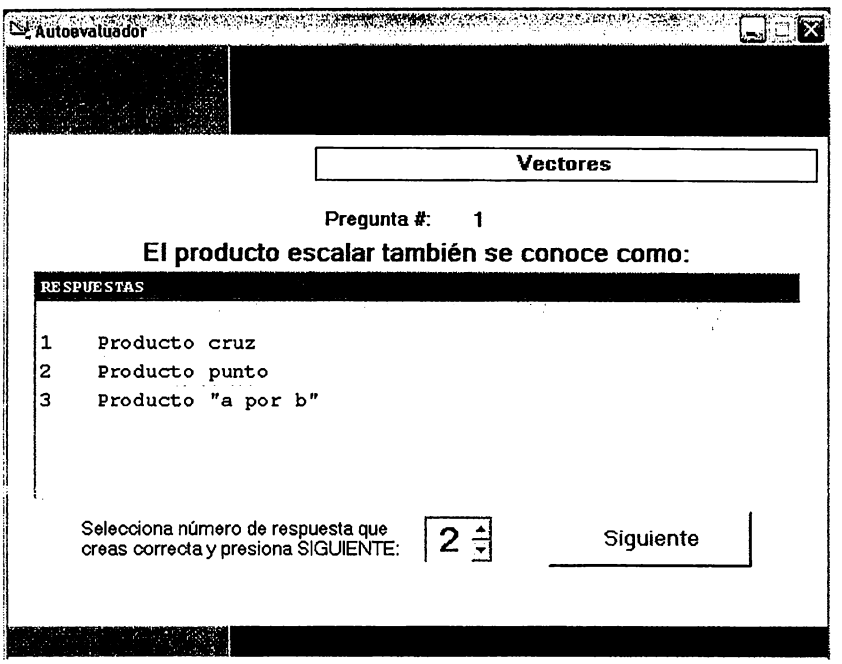


Figura 5.32 Pantalla para evaluador

PANTALLA PARA ÍNDICE CONCEPTUAL (indice.vb)

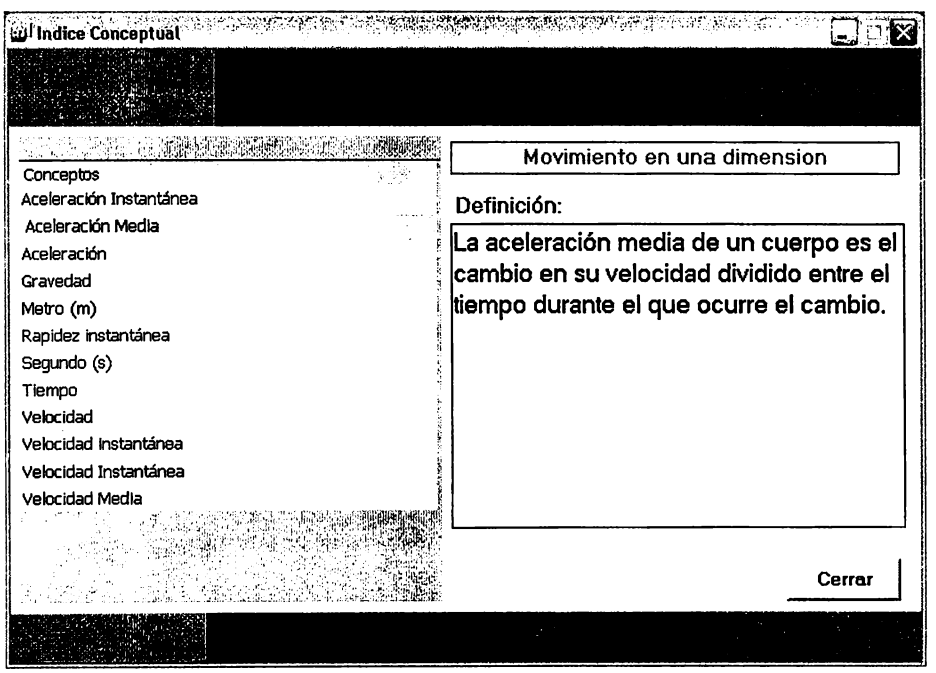


Figura 5.33 Pantalla para índice conceptual

CONCLUSIONES

Actualmente existen a nuestro alrededor diferentes necesidades que dificultan el desarrollo de las muchas actividades diarias, las cuales están a la espera de una solución efectiva y funcional. Pero también existe a su vez, diferentes herramientas que permiten solventar cada una de estas necesidades. Está en nuestras manos la creación de dichas soluciones, dependerá de la capacidad y el esfuerzo dedicado para desarrollar una eficaz herramienta que facilite el trabajo en cada uno de los ámbitos laborales y de estudio.

Existe un problema grande a nuestro alrededor, contra el que tenemos que luchar cada uno de los involucrados en el desarrollo de soluciones tecnológicas, y es la resistencia al cambio. Este es un problema que está presente en cualquier ambiente, y que se debe a diferentes factores. Nuestra misión es dar los primeros pasos, impulsar e innovar hacia el desarrollo de soluciones que demuestren su utilidad y que cambien de una vez por todas la mentalidad estacionaria de la cual somos victimas en más de alguna ocasión.

Algunas necesidades son solventadas con más de una solución y continuamente se extiende el número de éstas a mejoradas versiones que van en beneficio del usuario. Lo que indica que no sólo el desarrollo de soluciones vanguardistas debe ser el centro de atención para los ingenieros, existen también herramientas que pueden ser mejoradas en gran forma y de esta manera lograr que se acerquen cada vez más a la perfección o que se adecuen a necesidades diferentes según el ambiente en que se aplique.

Un factor importante en la creación de soluciones, es vernos a nosotros mismos en alguna etapa de nuestro desarrollo y las dificultades que nos ocasionaba no contar con herramientas tecnológicas adecuadas. No queremos ver a las nuevas generaciones pasar los mismos problemas, queremos ver herramientas innovadoras que les faciliten el trabajo y que les permitan un mejor desarrollo.

a) BIBLIOGRAFÍA

- **Visual Basic.NET**
  Guía de migración y actualización
  Primera Edición
  Mc Graw Hill, 2002

- **Ingeniería del Software**
  Un enfoque práctico
  Quinta Edición
  Mc Graw Hill, 2002

- **Halliday / Resnick / Walker**
  Fundamentos de Física, Volumen 1
  Sexta Edición
  CECSA, 2001

- **Sears Zemansky / Young Freedman**
  Física Universitaria, Volumen 1
  Novena Edición
  Pearson Educación, 1996

b) SITIOS WEB

- http://www.enciga.org/taylor/lv.htm
- http://www-fen.upc.es/wfib/virtualab/victor/web/Espanol/VirtualLab.html
- http://www.unex.es/~optica/laboratorio.html
- http://www.interactivephysics.com/spanish/simulations.html
- http://www.deciencias.net/programas/defisica.htm
- http://www.interactivephysics.com/spanish/description.html
- http://www.interactivephysics.com/spanish/
- http://usuarios.lycos.es/macedoniamagazine/opengl1.htm
- http://www.upcnet.es/~rvm1/e_main.html
- http://www.gamarod.com.ar/comunidad/directorio/directorio.asp?cat_id=7
- http://graficos.conclase.net/curso/index.php
- http://www.programacionfacil.com/java/uno2.htm
- http://pinsa.escomposlinux.org/sromero/articulos/gfx/graf2.html
- http://usuarios.lycos.es/pefeco/temas.html
- http://www.fi.uba.ar/materias/6201/MosqVectoresacr.pdf
- http://www.ugr.es/~aquiran/docencia/apuntes/vectores.pdf
- http://www.virtual.unal.edu.co/cursos/sedes/manizales/4070002/index.html

GLOSARIO

A	
CONCEPTO	**DEFINICIÓN**
ADO.Net	Conjunto de tecnologías de acceso a datos incluidas en las bibliotecas de clases de .NET Framework.
Aplicación Web	Conjunto de clientes y servidores que cooperan para ofrecer la solución a un problema.
Archivo ejecutable	Archivo con el formato ejecutable portable (PE) que el cargador del sistema operativo puede cargar en memoria y ejecutar. Puede ser un archivo .exe o .dll. En el contexto de .NET, Common Language Runtime debe traducir un archivo PE a código nativo para que el sistema operativo pueda ejecutarlo. Vea también: archivo ejecutable portable (PE).
ASP.Net	Plataforma de desarrollo para la creación de aplicaciones Web basadas en servidor. Es una evolución de ASP en el espacio administrado.

B	
CONCEPTO	**DEFINICIÓN**
Biblioteca de clases	Biblioteca compatible con CLS de clases, interfaces y tipos de valor incluidos en .NET Framework SDK. Esta biblioteca brinda acceso a la funcionalidad del sistema y es la base sobre la que se crean las aplicaciones, los componentes y los controles de .NET Framework.

C	
CONCEPTO	**DEFINICIÓN**
Campo	Miembro que representa una variable asociada con un objeto o una clase.
Clase	Tipo de referencia que encapsula datos (constantes y campos) y el comportamiento (métodos, propiedades, indizadores, eventos, operadores, constructores de instancia, constructores estáticos y destructores), y puede contener tipos anidados. Los tipos de clase admiten la herencia, un mecanismo mediante el cual una clase derivada puede extender y especializar una clase base.
Common Language Runtime (CLR)	Motor que es el núcleo de la ejecución de código administrado. El motor de tiempo de ejecución proporciona al código administrado servicios como integración entre varios lenguajes, seguridad de acceso a código, administración de la duración de los objetos, y compatibilidad con la depuración y la generación de perfiles.
Common Language Specification (CLS)	Subconjunto de funciones del lenguaje admitidas por una amplia gama de herramientas compatibles. Se garantiza que las herramientas y los componentes compatibles con CLS pueden interoperar con otras herramientas y componentes compatibles con CLS.
Compilation Just-In-Time (JIT compilation)	Compilación que convierte el Lenguaje intermedio de Microsoft (MSIL) en código máquina cuando se requiere el código en tiempo de ejecución.

D	
CONCEPTO	**DEFINICIÓN**
Disco duro	Disco magnético hecho de metal y cubierto con una superficie de grabación magnética.

E	
CONCEPTO	**DEFINICIÓN**
Ensamblado	Conjunto de funcionalidad que se crea, se controla la versión y se implementa como una única unidad de implementación (uno o varios archivos). Un ensamblado es la unidad de creación principal de una aplicación .NET Framework. Todos los tipos y recursos administrados se marcan como accesibles únicamente dentro de su unidad de implementación o se exportan para su uso por parte de código externo a dicha unidad. En Common Language Runtime, el ensamblado establece el ámbito de nombres para resolver solicitudes y se exigen límites de visibilidad. El motor de tiempo de ejecución puede determinar y encontrar el ensamblado de cualquier objeto que esté en ejecución, ya que cada tipo se carga en el contexto de un ensamblado

Espacio de nombres	Esquema de nombres lógico para agrupar los tipos relacionados. .NET Framework utiliza un esquema de nombres jerárquico para agrupar los tipos en categorías lógicas de funcionalidad relacionada, como la tecnología ASP.NET o la funcionalidad de interacción remota. Las herramientas de diseño pueden utilizar espacios de nombres para que los programadores puedan examinar y hacer referencia más fácilmente a los tipos en el código. El concepto de un espacio de nombres es ortogonal al de un ensamblado: un único ensamblado puede contener tipos cuyos nombres jerárquicos tienen distintas raíces de espacio de nombres y una raíz de espacio de nombres lógico puede abarcar varios ensamblados. En .NET Framework, un espacio de nombres es una comodidad para la nomenclatura lógica en tiempo de diseño, mientras que un ensamblado establece el ámbito de nombres para los tipos en tiempo de ejecución

G	
CONCEPTO	**DEFINICIÓN**
Gif (Graphics Interchange Format)	Formato de intercambio de gráficos. Formato de archivos de rastreo popular desarrollado por CompuServe que maneja color de 8 bits (256 colores) y utiliza el método LZW para alcanzar proporciones de comprensión de aproximadamente 1,5:1 a2:1.
Gigabyte	Un millón de bytes. También GB.

H	
CONCEPTO	**DEFINICIÓN**
Hardware	Son todos los componentes físicos que componen una PC.

I	
CONCEPTO	**DEFINICIÓN**
Icono	Símbolo gráfico que aparece en la pantalla de una PC para representar determinada acción a realizar por el usuario, ejecutar un programa, leer una información, imprimir un texto, etc. Un icono hace referencia a un programa o archivo computacional y por lo tanto le permite el acceso al mismo por parte del usuario.
Internet	Conjunto de redes conectadas entre sí, que utilizan El protocolo TCP/IP para comunicarse.

J	
CONCEPTO	**DEFINICIÓN**
JIT	Acrónimo de "Just-In-Time", una expresión que describe una acción que sólo se realiza cuando es necesario, como la compilación Just-In-Time o la activación de objetos Just-In-Time.

L	
CONCEPTO	**DEFINICIÓN**
Lenguaje de compilación	Un lenguaje de programación de alto nivel que debe ser traducido a lenguaje de máquina antes de ejecutarse el programa.
Lenguaje intermedio de Microsoft (MSIL)	Lenguaje utilizado como salida de una serie de compiladores y como entrada para un compilador Just-In-Time (JIT). Common Language Runtime incluye un compilador Just-In-Time para convertir MSIL a código nativo.

M	
CONCEPTO	**DEFINICIÓN**
Megabytes	1.000.000 ó 1.048.576 bytes o caracteres. También se escribe MB.

N	
CONCEPTO	**DEFINICIÓN**
.NET Framework	Plataforma para crear, implementar y ejecutar aplicaciones y servicios Web XML. Proporciona un entorno de múltiples lenguajes basado en estándares y muy productivo para integrar las inversiones existentes con aplicaciones y servicios de la próxima generación, así como la agilidad necesaria para resolver los desafíos que suponen la implementación y el funcionamiento de las aplicaciones para Internet. .NET Framework consta de tres partes principales: Common Language Runtime, un conjunto jerárquico de bibliotecas de clases unificadas y una versión dividida en componentes de ASP denominada ASP.NET.

R	
CONCEPTO	**DEFINICIÓN**
Recolección de elementos no utilizados (GC) (garbage collection)	Proceso de hacer un seguimiento transitivamente por todos los punteros hasta los objetos utilizados activamente con el fin de encontrar todos los objetos a los que se puede hacer referencia y, después, conseguir la reutilización de toda la memoria de montón que no se haya encontrado durante este seguimiento. El recolector de elementos no utilizados de Common Language Runtime también compacta la memoria en uso para reducir el espacio de trabajo necesario para el montón.

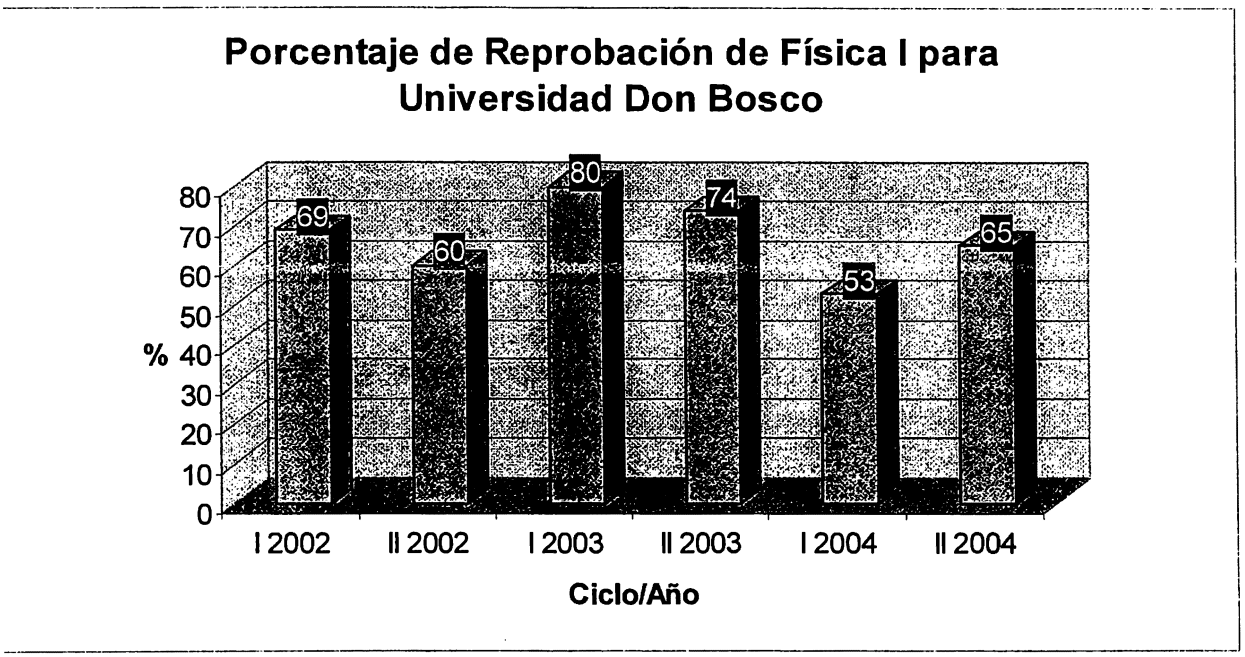
S	
CONCEPTO	**DEFINICIÓN**
Software	Instrucciones para una computadora. Una serie de instrucciones que realizan una tarea en particular se llama programa o programa de software.

W	
CONCEPTO	**DEFINICIÓN**
Web Forms	Marco de trabajo de páginas ASP.NET, que admite controles de servidor que procesan la interfaz de usuario HTML en los exploradores Web.
Windows Forms	Biblioteca de clientes de Windows

X	
CONCEPTO	**DEFINICIÓN**
XML	Subconjunto del Lenguaje de marcado generalizado estándar (SGML) optimizado para su uso a través del Web. XML proporciona un método uniforme para describir e intercambiar datos estructurados que es independiente de las aplicaciones o los proveedores.

ANEXOS

ANEXO I. Porcentaje de reprobación para Física I.



	Física I	Matemática I	Programación I	Estadística	Sistemas Digitales
Porcentaje de Reprobación	66.83 %	53.21 %	14.9 %	59.13 %	31.5 %

* Datos correspondientes a ciclos normales entre 2002 y 2004

ANEXO II. Planificación docente para la materia Física I

UNIDAD I: Vectores

1. Magnitudes vectoriales y Escalares
2. Suma y Resta de Vectores por Método Gráfico
3. Multiplicación de un Vector por un Escalar
4. Vector Unitario
5. Suma y Resta de Vectores por el Método de componentes rectangulares

UNIDAD II: Movimiento en una dimensión

1. Cinemática
2. Concepto de Posición y Desplazamiento
3. Velocidad Media e Instantánea
4. Aceleración Media e Instantánea
5. Movimiento con Aceleración Constante
6. Movimiento de Caída Libre

UNIDAD III: Movimiento de dos dimensiones

1. Representación Gráfica de Trayectoria, Posición y Desplazamiento
2. Velocidad y Aceleración (Media e instantánea)
3. Movimiento de Proyectiles
4. Movimiento Circular Uniforme
5. Movimiento Relativo

UNIDAD IV: Dinámica de Traslación

1. Fuerzas e interacciones
2. Primera Ley de Newton
3. Segunda Ley de Newton
4. Concepto de Peso y Masa
5. Tercera Ley de Newton
6. Aplicaciones de las Leyes de Newton
7. Fuerzas de Fricción
8. Dinámica de movimiento circular uniforme

IIDAD V: Trabajo y Energía

1. Trabajo efectuado por una fuerza constante
2. Teorema de trabajo y energía
3. Trabajo efectuado por una fuerza variable
4. Concepto de potencia
5. Fuerzas conservativas y no conservativas
6. Energía Potencial y Ley de Conservación de la Energía
7. Energía potencial elástica y gravitatoria
8. Ley de la conservación de la energía en presencia de fuerzas no conservativas.

IIDAD VI: Cantidad de movimiento y Choques

1. Cantidad de movimiento lineal y la ley de la conservación de la cantidad de movimiento.
2. Teorema del Impulso y la cantidad de movimiento lineal
3. Choque en una dimensión
4. Choque en dos dimensiones
5. Conceptos de centro de masa

IIDAD VII: Cinemática de Rotación

1. Desplazamiento, velocidad y aceleración angular
2. Movimiento de rotación con aceleración constante
3. Relación entre cantidades lineales y angulares

NIDAD VIII: Dinámica del Movimiento de Rotación

1. Energía cinética en el movimiento de rotación
2. Calculo del momento de inercia para un sistema de partículas
3. Cálculo del momento de inercia para un sistema de masa continua
4. Teorema de los ejes paralelos
5. Momento de torsión
6. Segunda Ley de newton para el movimiento de rotación
7. Trabajo, potencia y teorema del trabajo y energía en el movimiento de rotación.
8. Conservación de la energía en el movimiento de rotación
9. Cantidad de movimiento angular de una partícula
10. Cantidad de movimiento angular de un sistema de partículas
11. Conservación de la cantidad angular.

ANEXO III. Encuesta realizada a alumnos que cursan la materia de Física I

Universidad Don Bosco
Facultad de Ingeniería
Escuela de Computación

DESARROLLO DE SIMULADOR PARA FISICA I

¿Cómo calificarías a la Física I en base al grado de dificultad?
Muy difícil Difícil Accesible Fácil

¿Cómo calificarías el nivel académico con que se imparte la cátedra?
Excelente Muy Bueno Bueno Regular Deficiente

¿Cuan importante consideras que son los ejemplos desarrollados en clase por el catedrático para el entendimiento del tema?
Muy Importantes Poco Importantes Están de más

¿Qué porcentaje aproximadamente de los temas explicados en clase has comprendido solo con la explicación y el ejemplo?
100% más del 50% 50% menos del 50%

¿Cuan importante crees que son las guías de ejercicios para la comprensión de los temas?
Muy Importantes Poco Importantes Están de más

¿Qué porcentaje aproximadamente de las guías de ejercicio has completado en su totalidad?
100% más del 50% 50% menos del 50%

¿Cuántos de los problemas planteados en las guías has desarrollado llegando a una respuesta pero con la incertidumbre de no saber si la respuesta es correcta o no?
La mayoría La mitad Algunos Ninguno

¿Cuantos de los problemas planteados en las guías has logrado desarrollar mediante fórmulas llegando a a respuesta correcta pero sin comprender con claridad el porqué de los datos encontrados ni del enómeno físico estudiado?

La mayoría La mitad Algunos Ninguno

¿Cuan importante crees que son los laboratorios prácticos para la comprensión de los temas?

Muy Importantes Poco Importantes Están de más

¿Qué porcentaje aproximadamente de los laboratorios prácticos desarrollados has completado y comprendido en su totalidad?

100% más del 50% 50% menos del 50%

¿Qué porcentaje aproximadamente de los laboratorios prácticos desarrollados han servido para terminar de comprender el tema en estudio?

100% más del 50% 50% menos del 50%

¿Apoyarías la idea de poder contar con los recursos completos de los laboratorios prácticos en el instante en que tu los necesites y con toda libertad de uso sin límites de tiempo?

Si No

Porqué:_____

_____

¿Apoyarías la idea de poder contar con un laboratorio virtual que simule de manera gráfica los fenómenos físicos en estudio y que devuelva la resolución de los problemas que tu le plantees?

Si No

Porqué:_____

_____

Si la universidad dispusiera la venta de una aplicación para PC como la planteada en la pregunta anterior a un precio razonable, ¿estarías dispuesto a adquirirla?

Si No

Porqué:_____

_____

¿Cómo calificarías una metodología de estudio basada en la simulación de los fenómenos físicos mediante una aplicación en PC para apoyar lo desarrollado en papel?

Atractiva Interesante Problemática Aburrida

an beneficioso crees que sería para ti el poder contar con dicha herramienta de estudios?

Beneficioso Ayudaría en algo No ayudaría

nsideras que contar con herramientas de este tipo ayudaría a elevar aún más el nivel académico de la
ersidad Don Bosco?

No

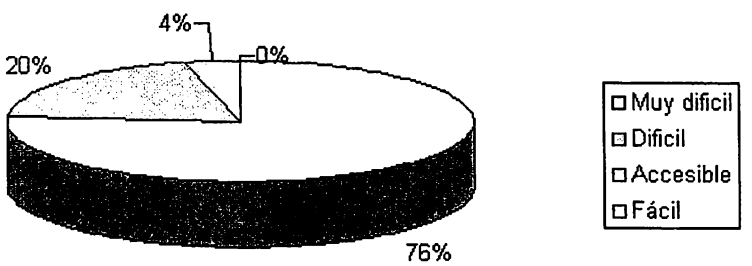
qué:_____

_____

ANEXO IV. Tabulación de datos de la encuesta realizada

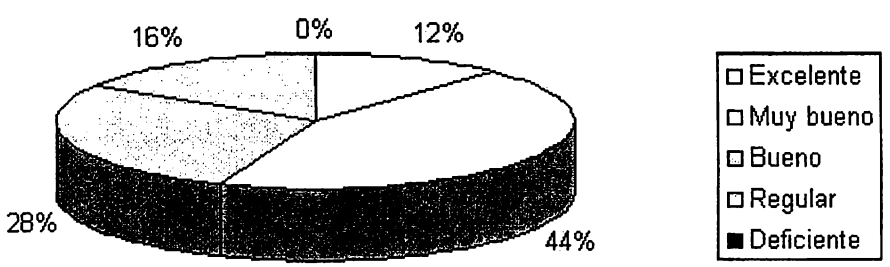
1. ¿Cómo calificarías a la Física I, en base al grado de dificultad?

RESPUESTA	TOTAL
Muy difícil	38
Difícil	10
Accesible	2
Fácil	0
TOTAL DE MUESTRA	**50**



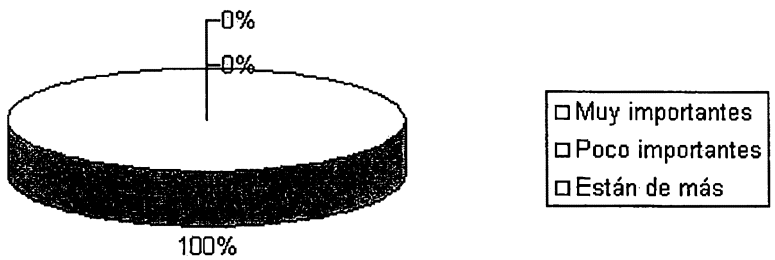
2. ¿Cómo calificarías el nivel académico con que se imparte la cátedra?

RESPUESTA	TOTAL
Excelente	6
Muy bueno	22
Bueno	14
Regular	8
Deficiente	0
TOTAL DE MUESTRA	**50**



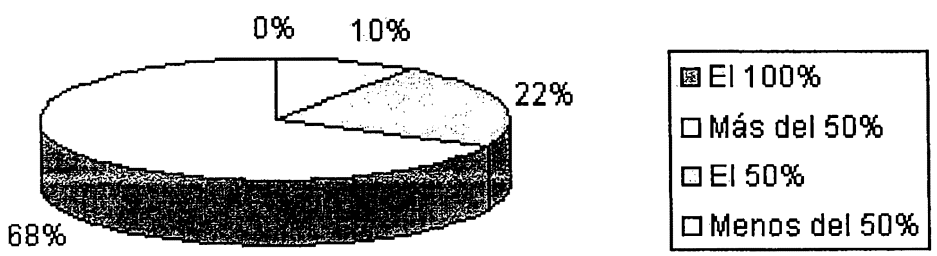
3. **¿Cuan importante consideras que son los ejemplos desarrollados en clase por el catedrático para el entendimiento del tema?**

RESPUESTA	TOTAL
Muy importantes	50
Poco importantes	0
Están de más	0
TOTAL DE MUESTRA	**50**



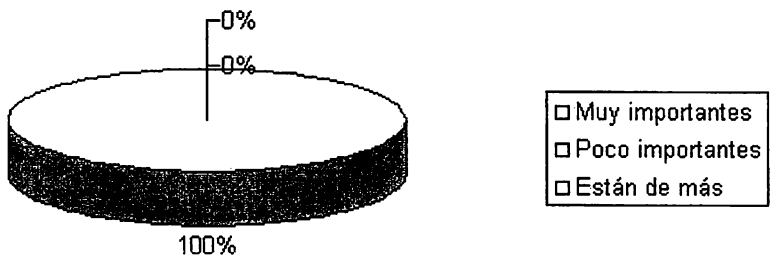
4. **¿Qué porcentaje aproximadamente de los temas explicados en clase has comprendido sólo con la explicación y el ejemplo?**

RESPUESTA	TOTAL
El 100%	0
Más del 50%	5
El 50%	11
Menos del 50%	34
TOTAL DE MUESTRA	**50**



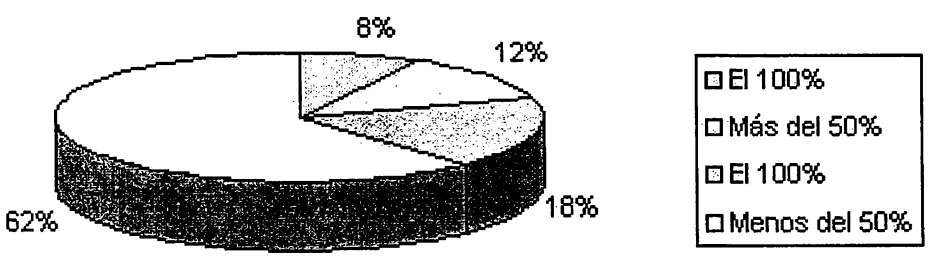
5. **¿Cuan importante crees que son las guías de ejercicios para la comprensión de los temas?**

RESPUESTA	TOTAL
Muy importantes	50
Poco importantes	0
Están de más	0
TOTAL DE MUESTRA	**50**



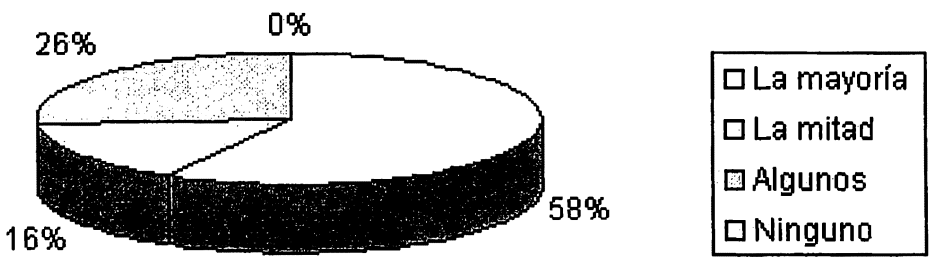
6. **¿Qué porcentaje aproximadamente de las guías de ejercicios has completado en su totalidad?**

RESPUESTA	TOTAL
El 100%	4
Más del 50%	6
El 50%	9
Menos del 50%	31
TOTAL DE MUESTRA	**50**



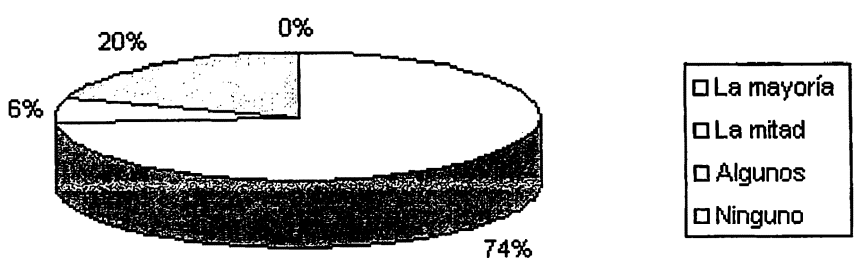
7. **¿Cuántos de los problemas planteados en las guías has desarrollado llegando a una respuesta pero con la incertidumbre de no saber si la respuesta es correcta o no?**

RESPUESTA	TOTAL
La mayoría	29
La mitad	8
Algunos	13
Ninguno	0
TOTAL DE MUESTRA	**50**



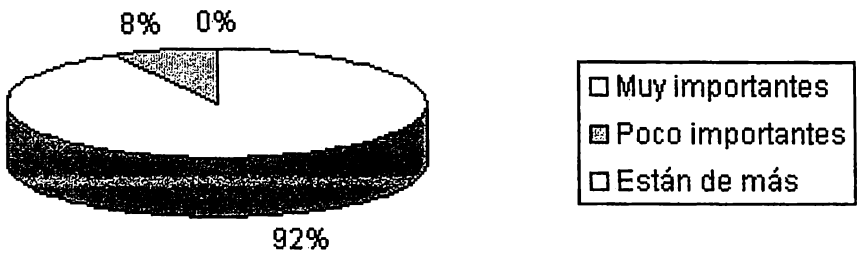
8. **¿Cuántos de los problemas planteados en las guías has logrado desarrollar mediante fórmulas llegando a la respuesta correcta pero sin comprender con claridad el porqué de los datos encontrados ni del fenómeno físico estudiado?**

RESPUESTA	TOTAL
La mayoría	37
La mitad	3
Algunos	10
Ninguno	0
TOTAL DE MUESTRA	**50**



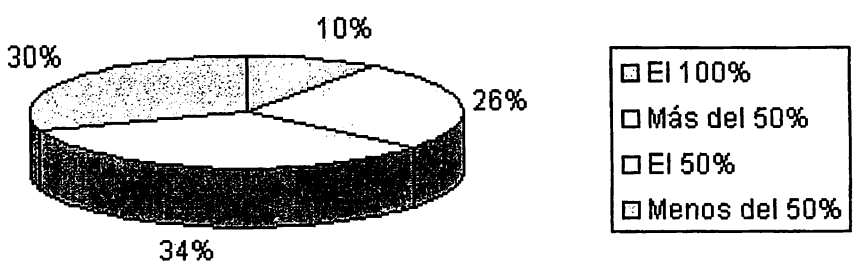
9. **¿Cuan importante crees que son los laboratorios prácticos para la comprensión de los temas?**

RESPUESTA	TOTAL
Muy importantes	46
Poco importantes	4
Están de más	0
TOTAL DE MUESTRA	**50**



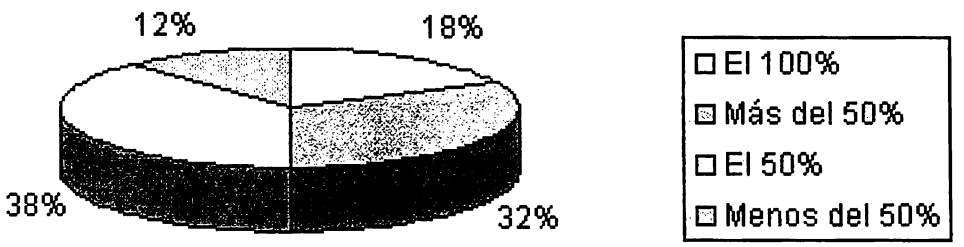
10. **¿Qué porcentaje aproximadamente de los laboratorios prácticos desarrollados has completado y comprendido en su totalidad?**

RESPUESTA	TOTAL
El 100%	5
Más del 50%	13
El 50%	17
Menos del 50%	15
TOTAL DE MUESTRA	**50**



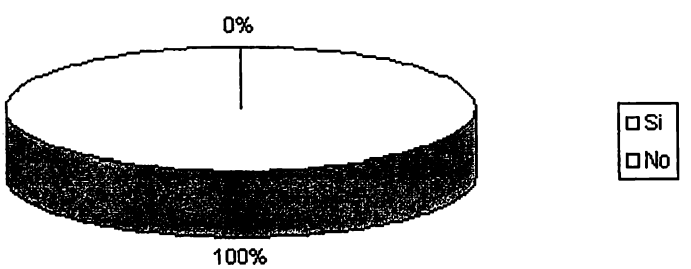
11. **¿Qué porcentaje aproximadamente de los laboratorios prácticos desarrollados han servido para terminar de comprender el tema en estudio?**

RESPUESTA	TOTAL
El 100%	9
Más del 50%	16
El 50%	19
Menos del 50%	6
TOTAL DE MUESTRA	**50**



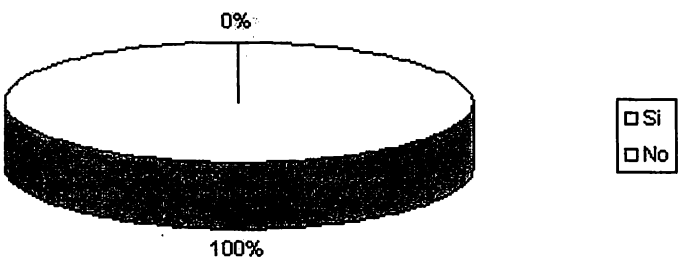
12. **¿Apoyarías la idea de poder contar con los recursos completos de los laboratorios prácticos en el instante en que tú lo necesites y con toda libertad de uso sin límites de tiempo?**

RESPUESTA	TOTAL
Si	50
No	0
TOTAL DE MUESTRA	**50**



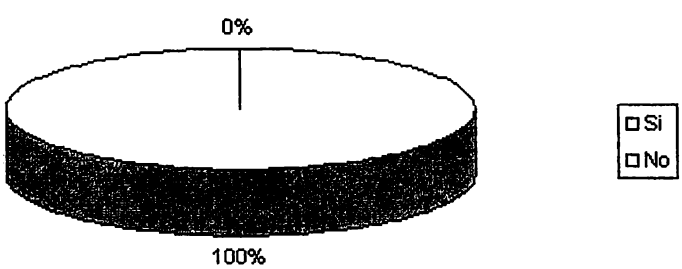
13. **¿Apoyarías la idea de poder contar con un laboratorio virtual que simule de manera gráfica los fenómenos físicos en estudio y que devuelva la resolución de los problemas que tú plantees?**

RESPUESTA	TOTAL
Si	50
No	0
TOTAL DE MUESTRA	**50**



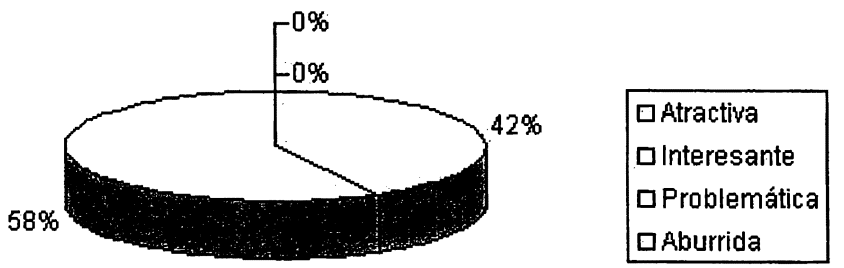
14. **Si la universidad dispusiera la venta de una aplicación para PC como la planteada en la pregunta anterior y a un precio razonable, ¿Estarías dispuesto a adquirirla?**

RESPUESTA	TOTAL
Si	50
No	0
TOTAL DE MUESTRA	**50**



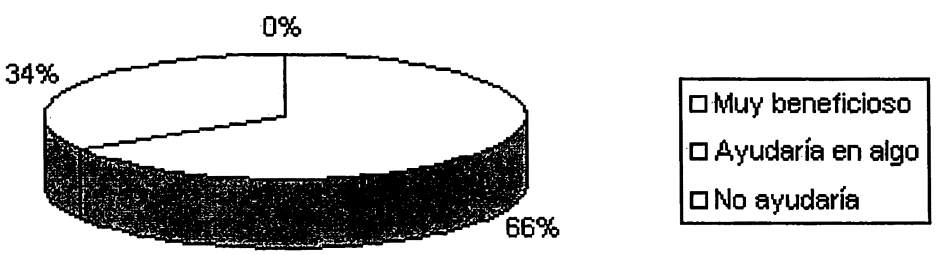
15. **¿Cómo calificarías una metodología de estudio basada en la simulación de los fenómenos físicos mediante una aplicación en PC para apoyar lo desarrollado en papel?**

RESPUESTA	TOTAL
Atractiva	21
Interesante	29
Problemática	0
Aburrida	0
TOTAL DE MUESTRA	**50**



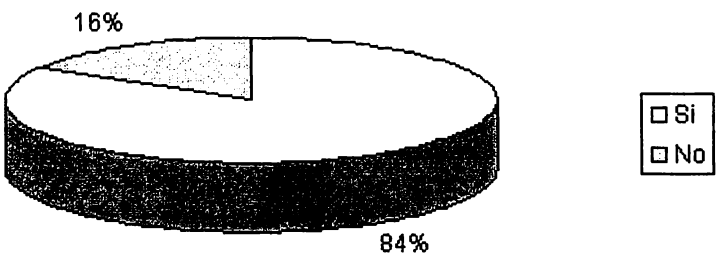
16. **¿Cuan beneficioso crees que sería para ti el poder contar con dicha herramienta de estudio?**

RESPUESTA	TOTAL
Muy beneficioso	33
Ayudaría en algo	17
No ayudaría	
TOTAL DE MUESTRA	**50**



17. ¿Consideras que contar con herramientas de este tipo ayudaría a elevar aún más el nivel académico de la Universidad Don Bosco?

RESPUESTA	TOTAL
Si	42
No	8
TOTAL DE MUESTRA	**50**



ANEXO V. Cuestionario realizado a alumnos de la materia Física I.

iversidad Don Bosco
cultad de Ingeniería
cuela de Computación

DESARROLLO DE SIMULADOR PARA FISICA I

Evaluación de la Aplicación

jetivo de Proyecto: Desarrollar un simulador para la enseñanza de la asignatura Física I para la iversidad Don Bosco en un ambiente gráfico bidimensional, que sea una herramienta útil, fácil y cesible para alumnos y docentes, que sirva de apoyo al estudio de dicha materia.

dicaciones: Subraye la respuesta que más le parezca.

n general, que opinión le merece el funcionamiento del simulador hasta el momento?

celente Muy bueno Bueno Regular Deficiente

ué áreas cree usted que se podrían mejorar?

eoría Simulación Cálculos Ayuda Ninguna

la parte teórica, ¿que aspectos deben mejorar?

ntenido Presentación Estructura Ninguna

gerencias:

_____

_____

_____

la parte de simulación, ¿qué opinión le merece el aspecto gráfico?

celente Muy bueno Bueno Regular Deficiente

cuanto al grado de dificultad operacional, ¿cómo calificaría las simulaciones?

cil Accesible Complicada

¿Qué aspectos mejoraría en las simulaciones?

Captura de Datos Aspecto Gráfico Datos Devueltos Opciones Extras

Sugerencias:

_____

_____

_____

_____

En su opinión ¿Cuan útil puede ser la aplicación en la comprensión de los fenómenos físicos estudiados?

Muy útil De alguna ayuda De ninguna Ayuda

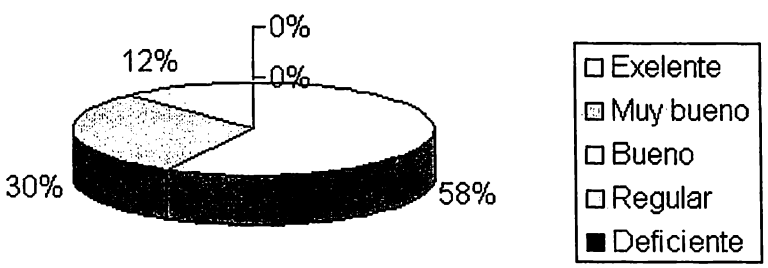
¿Consideraría que contar con herramientas de este tipo ayudaría a elevar aún más el nivel académico de la Universidad Don Bosco?

Ayudaría mucho Ayudaría en algo Ayudaría Poco

ANEXO VI. Tabulación de datos del cuestionario realizado

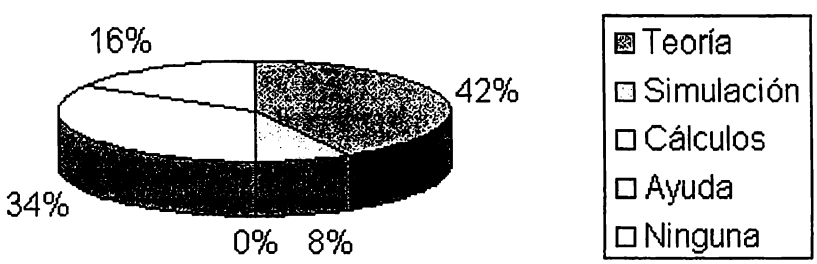
1. **¿En general, que opinión le merece el funcionamiento del simulador hasta el momento?**

RESPUESTA	TOTAL
Excelente	29
Muy bueno	15
Bueno	6
Regular	0
Deficiente	0
TOTAL DE MUESTRA	**50**



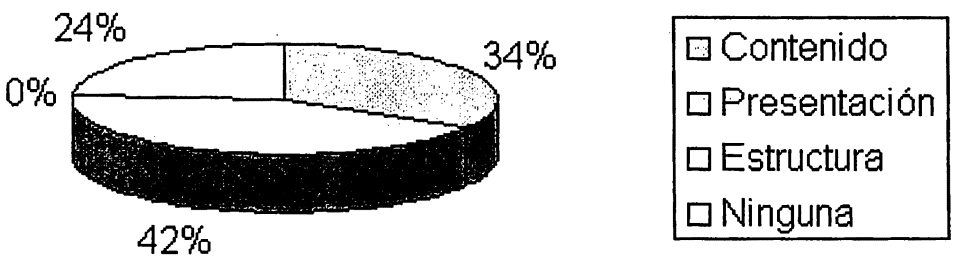
2. **¿Qué áreas cree usted que se podrían mejorar?**

RESPUESTA	TOTAL
Teoría	15
Simulación	10
Cálculos	0
Ayuda	15
Ninguna	10
TOTAL DE MUESTRA	**50**



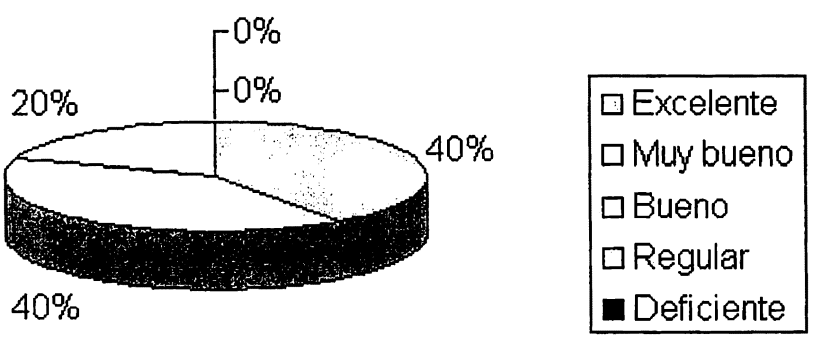
3. **En la parte teórica, ¿que aspectos deben mejorar?**

RESPUESTA	TOTAL
Contenido	17
Presentación	21
Estructura	0
Ninguna	12
TOTAL DE MUESTRA	**50**



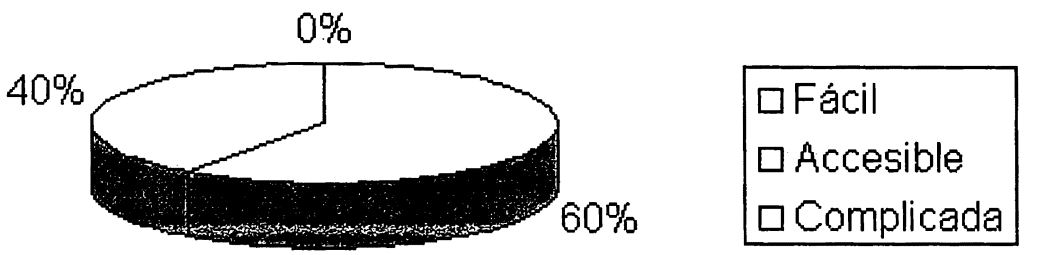
4. **En la parte de simulación, ¿qué opinión le merece el aspecto gráfico?**

RESPUESTA	TOTAL
Excelente	20
Muy bueno	20
Bueno	10
Regular	0
Deficiente	0
TOTAL DE MUESTRA	**50**



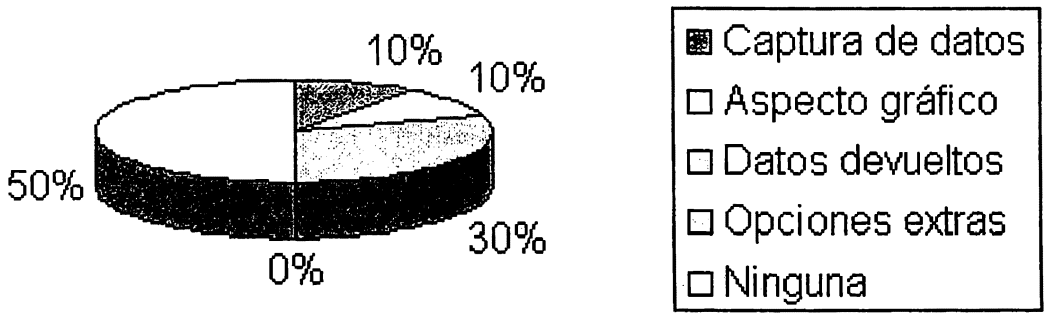
5. **En cuanto al grado de dificultad operacional, ¿cómo calificaría las simulaciones?**

RESPUESTA	TOTAL
Fácil	30
Accesible	20
Complicada	0
TOTAL DE MUESTRA	**50**



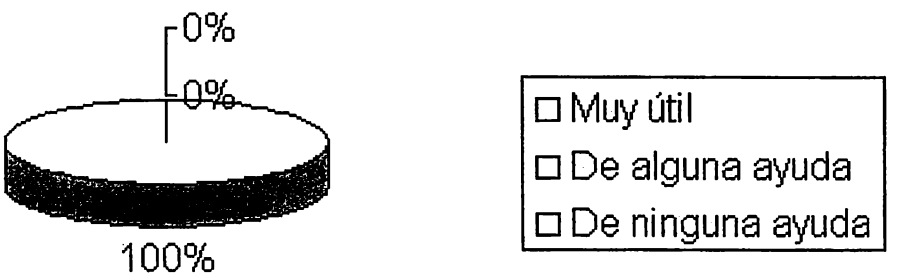
6. **¿Qué aspectos mejoraría en las simulaciones?**

RESPUESTA	TOTAL
Captura de datos	5
Aspecto gráfico	5
Datos devueltos	15
Opciones extra	0
Ninguna	25
TOTAL DE MUESTRA	**50**



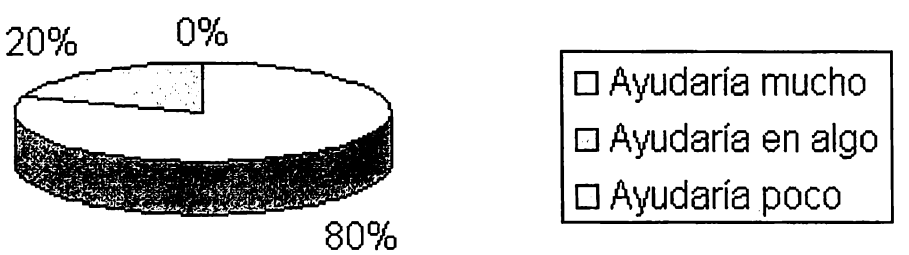
7. **En su opinión ¿Cuan útil puede ser la aplicación en la comprensión de los fenómenos físicos estudiados?**

RESPUESTA	TOTAL
Muy útil	50
De alguna ayuda	0
De ninguna ayuda	0
TOTAL DE MUESTRA	**50**



8. **¿Consideraría que contar con herramientas de este tipo ayudaría a elevar aún mas el nivel académico de la Universidad Don Bosco?**

RESPUESTA	TOTAL
Ayudaría mucho	40
Ayudaría en algo	10
Ayudaría poco	0
TOTAL DE MUESTRA	**50**



Manual de Usuario

INDICE

1. INTRODUCCIÓN 1

1.1 ACERCA DE ESTE MANUAL 1

1.2 EL PROGRAMA "FÍSICA INTERACTIVA" 1

1.3 REQUERIMIENTOS TÉCNICOS 2

1.4 INSTALACIÓN 3

1.5 USO DE LA AYUDA 6

1.6 EL ENTORNO DE TRABAJO 8

2. CONTENIDO TEÓRICO 9

3. SIMULACIONES 10

3.1 BOTONES DE CONTROL 10

3.2 VECTORES 12

3.2.1 DIBUJAR UN VECTOR 13

3.2.2 SUMA DE DOS VECTORES 14

3.2.3 RESTA DE DOS VECTORES 15

3.2.4 MULTIPLICACIÓN DE UN VECTOR POR UN ESCALAR 16

3.3 MOVIMIENTO HORIZONTAL 17

3.3.1 ECUACIONES 18

3.3.2 EJEMPLO 18

3.4 MOVIMIENTO VERTICAL 20

3.4.1 ECUACIONES 21

3.4.2 EJEMPLO 21

3.5 MOVIMIENTO PARABÓLICO 22

3.5.1 ECUACIONES 23

3.5.2 EJEMPLO 23

3.6 MOVIMIENTO CIRCULAR 24

3.6.1 ECUACIONES 25

3.6.2 EJEMPLO 25

.7 LEYES DE NEWTON 26
3.7.1 ECUACIONES 27
3.7.2 EJEMPLO 28
.8 TRABAJO Y ENERGÍA 29
3.8.1 ECUACIONES 30
3.8.2 EJEMPLO 30
.9 COLISIONES 31
3.9.1 ECUACIONES 32
3.9.2 EJEMPLO 33
10 ROTACIÓN 34
3.10.1 ECUACIONES 35
3.10.2 EJEMPLO 35
EVALUADOR 36
INDICE CONCEPTUAL 38

1. INTRODUCCIÓN

1.1 ACERCA DE ESTE MANUAL

Este manual ha sido elaborado para orientar al usuario del simulador "***Física Interactiva"*** en el desarrollo de su contenido teórico, sus prácticas, sus evaluaciones y listado de conceptos, haciendo uso de ésta en el momento que se crea más conveniente, de forma tal que se ofrece un medio facilitador para el proceso del aprendizaje de la materia Física I.

1.2 EL PROGRAMA "FÍSICA INTERACTIVA"

"Física Interactiva" es un programa que provee al usuario el contenido teórico de temas que abarcan siete unidades de la Física I, las cuales se listan a continuación:

- **Unidad I:** Vectores
- **Unidad II**: Movimiento en una dimensión
- **Unidad III:** Movimiento en dos dimensiones
- **Unidad IV:** Fuerza y Movimiento
- **Unidad V**: Trabajo y Energía
- **Unidad VI**: Colisiones
- **Unidad VII**: Rotación

Además permite resolver problemas de carácter teórico en un ambiente gráfico bidimensional, que simula la situación planteada por el usuario para llegar a un resultado cuantitativo.

Así mismo el programa *"Física Interactiva"* permite al usuario evaluar los conocimientos asimilados en una unidad de estudio específica.

También ofrece un listado de conceptos, el cual facilita la búsqueda de conceptos que se relacionan al estudio de una unidad.

Física Interactiva se puede ejecutar como cualquier otra aplicación. Clicando dos veces en el icono correspondiente (ver figura 1.1) en el escritorio.

Fig. No.1.1 Acceso directo

1.3 REQUERIMIENTOS TÉCNICOS

INSTALACIÓN DE FRAMEWORK

- Sistema operativo:
  - Microsoft® Windows® 98
  - Microsoft® Windows® 98 Segunda edición
  - Microsoft® Windows® Millennium Edition
  - Microsoft® Windows NT® 4.0 Workstation con Service Pack 6.0a o posterior
  - Microsoft® Windows NT® 4.0 Server con Service Pack 6.0a o posterior
  - Microsoft® Windows® 2000 Professional
  - Microsoft® Windows® 2000 Server
  - Microsoft® Windows® 2000 Advanced Server
  - Microsoft® Windows® XP Home Edition
  - Microsoft® Windows® XP Professional

  Nota En todos estos sistemas, también es necesario tener Microsoft® Internet Explorer 5.01 o posterior y Microsoft® Windows® Installer 2.0 o posterior.

- Hardware:

Procesador necesario	Procesador recomendado	RAM necesaria	RAM recomendada
Pentium a 90 MHz*	Pentium a 90 MHz o más rápido	32 MB*	96 MB o más
Pentium a 133 MHz*	Pentium 133 o más rápido	128 MB*	256 MB o más

*O el mínimo requerido por el sistema operativo, lo que sea mayor.

INSTALACIÓN DE LA APLICACIÓN

- Instalación previa de Framework
- 80 Mb de espacio en disco duro
- 8 Mb Vram Resolución 1024x768, "8-bit High Color" Screen

1.4 INSTALACIÓN

Su CD de instalación contiene 5 archivos, haga clic en el icono Setup mostrado en la figura 1.2.

Fig. No.1.2 Archivos de instalación

Aparecerá en su pantalla un cuadro que indica que la instalación está siendo preparada para iniciar. (Vea la figura 1.3)

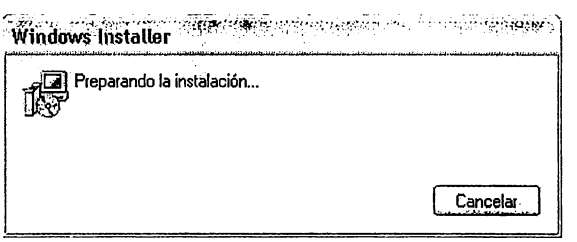


Fig. No.1.3 Instalación inicializada

Se inicializará un asistente que le guiará durante el proceso de la instalación, haga clic en "Siguiente" para continuar. (Vea la figura 1.4)

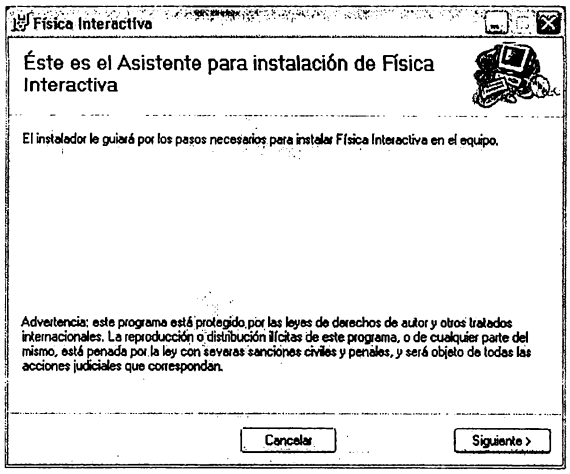


Fig. No.1.4 Asistente de instalación

A continuación debe seleccionar el directorio en el cual desea instalar la aplicación. Si desea cambiar de directorio haga clic en "Examinar", luego haga clic en "Siguiente". (Vea la figura 1.5).

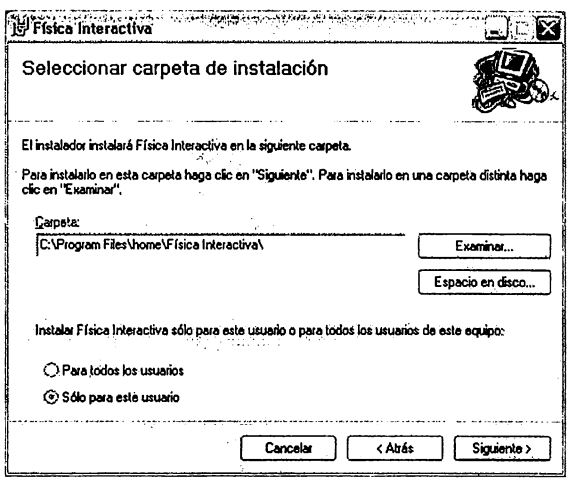


Fig. No.1.5 Directorio de instalación

Una vez hecho éstos pasos, el instalador está listo para iniciar con la instalación de la aplicación, solamente debe hacer clic en el botón "Siguiente". (Vea la figura 1.6).

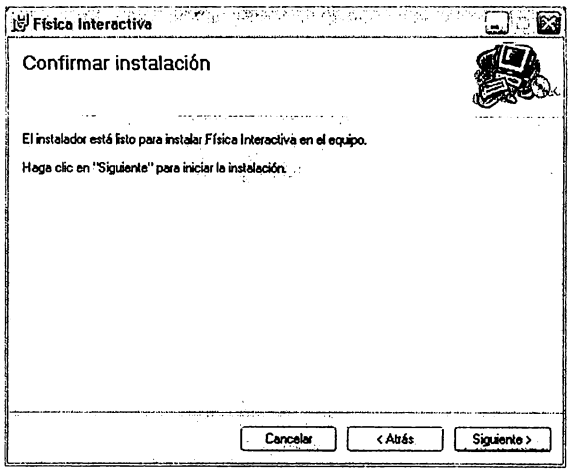


Fig. No.1.6 Confirmación de la instalación

La aplicación comienza a instalarse, este paso puede tardar varios minutos. Cuando finalice la instalación haga clic en "Siguiente".(Vea la figura 1.7).

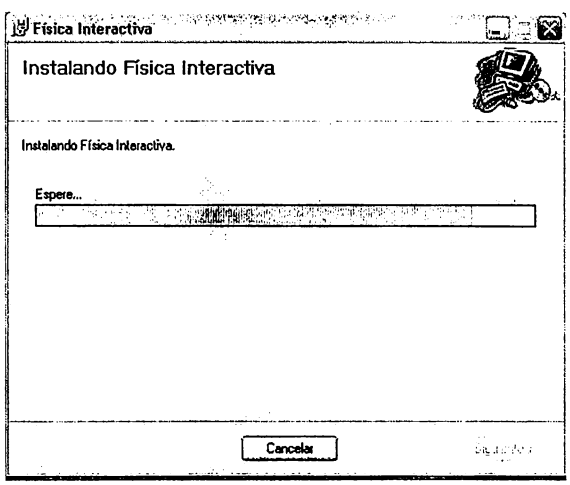


Fig. No.1.7 Instalación de la aplicación

Finalizada la instalación aparecerá una pantalla con un mensaje señalando que la instalación ha sido completada. Debe hacer clic en "Cerrar" para salir (Vea la figura 1.8).

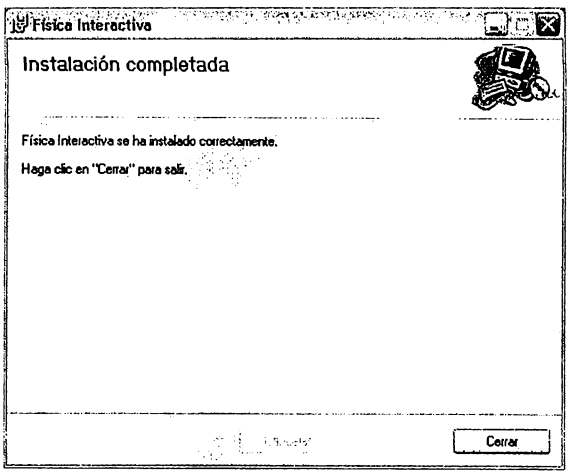


Fig. No.1.8 Instalación completada

1.5 USO DE LA AYUDA

El programa ***"Física Interactiva"*** dispone de una Ayuda con la cual el usuario puede encontrar la información necesaria para el manejo y desarrollo de la aplicación.

Para obtener la ayuda únicamente debe hacer clic sobre el botón que se muestra en la figura 1.9.

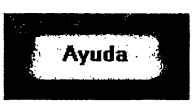


Fig. No.1.9 Botón de Ayuda

Al hacer clic en el botón mostrado en la figura No.1.9 se desplegará un formulario con los temas de ayuda. (Vea figura No.1.10)

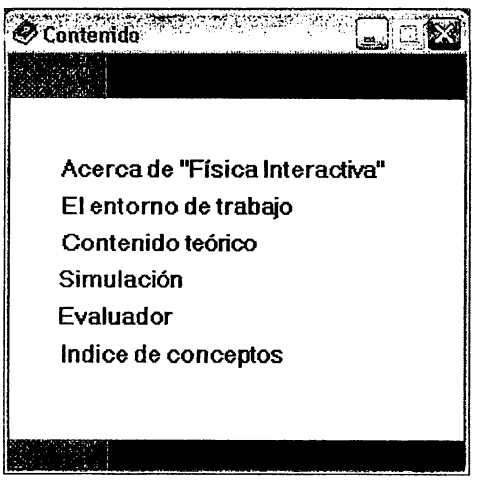


Fig. No.1.10 Contenido de Ayuda

Al hacer nuevamente clic sobre un tema específico, se desplegará un nuevo formulario con la ayuda correspondiente a dicho tema. (Vea figura No.1.11)

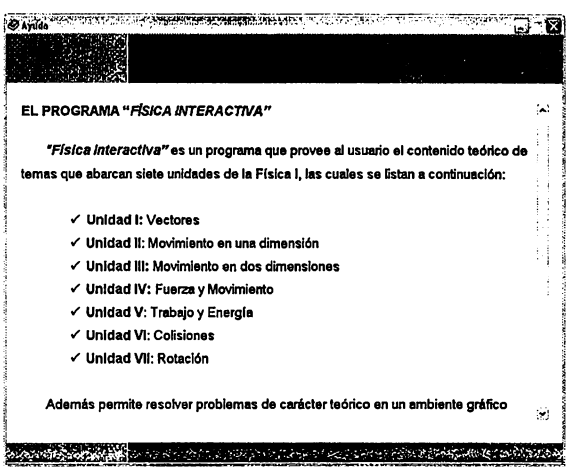


Fig. No.1.11 Formulario de contenido de la Ayuda

Si hace clic en *"Simulación"* se visualizará un formulario con el nombre de todas las prácticas que se pueden realizar.(Vea figura 1.15). Al seleccionar una simulación se abrirá la ayuda. (Ver sección 3).

Fig. No.1.15 Formulario de contenido de Simulación

1.6 EL ENTORNO DE TRABAJO

El entorno de trabajo del programa "Física Interactiva" es sencillo, la figura 1.16 muestra la pantalla principal, los botones de la izquierda sirven para seleccionar la unidad que se desea estudiar. El botón de la unidad activa se vuelve rojo y se muestra un submenú como el de la figura 1.17.

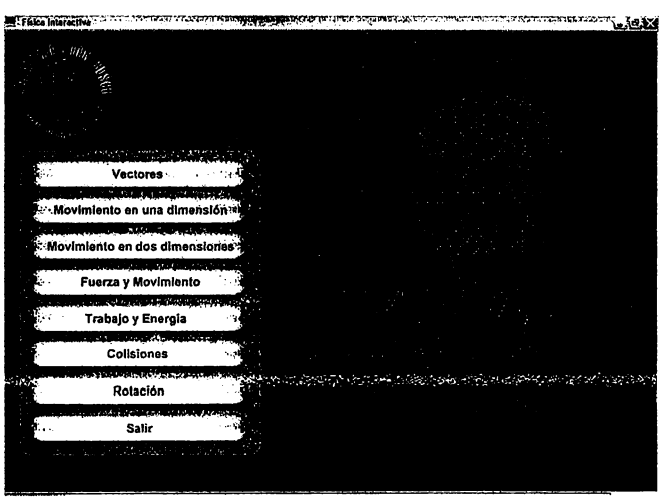


Fig. No. 1.16 Pantalla principal

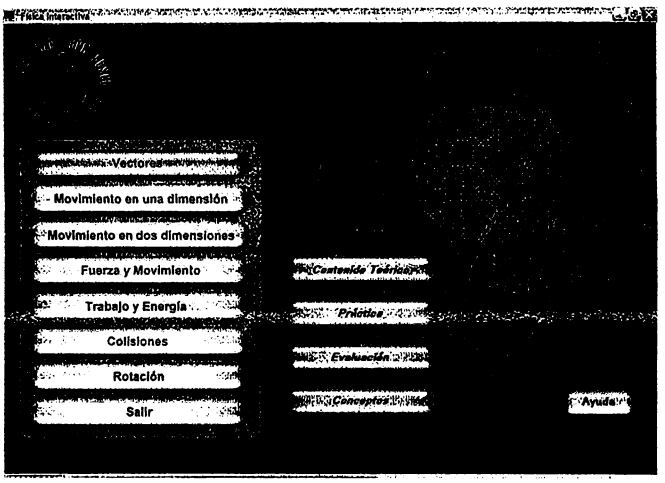


Fig. No. 1.17 Menú de unidades

Éste menú es el mismo para todas las unidades, y su función es la siguiente:

- **Contenido teórico:** Muestra en pantalla el desarrollo de temas teóricos. (Ver sección 2).
- **Práctica:** Muestra las prácticas que están disponibles para la unidad en estudio desplegando un submenú como el de la figura 1.18. (Ver sección 3).

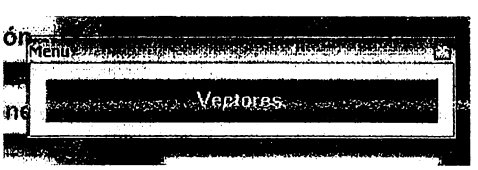


Fig. No. 1.18 Menú de simulaciones

- **Evaluación:** Muestra un formulario que permite realizar una prueba de conocimientos teórico sobre una unidad específica.(Ver sección 4)
- **Conceptos:** Despliega un formulario que facilita la búsqueda de un concepto relacionado a una unidad.(Ver sección 5)

2. CONTENIDO TEÓRICO

Esta sección de la aplicación consta del desarrollo teórico de los temas relacionados a las prácticas, dichos temas abarcan definiciones de conceptos y fórmulas para que puedan ser aplicados en las simulaciones.

La pantalla con la cual se tendrá que familiarizar para esta sección es la mostrada en la figura No.2.1.

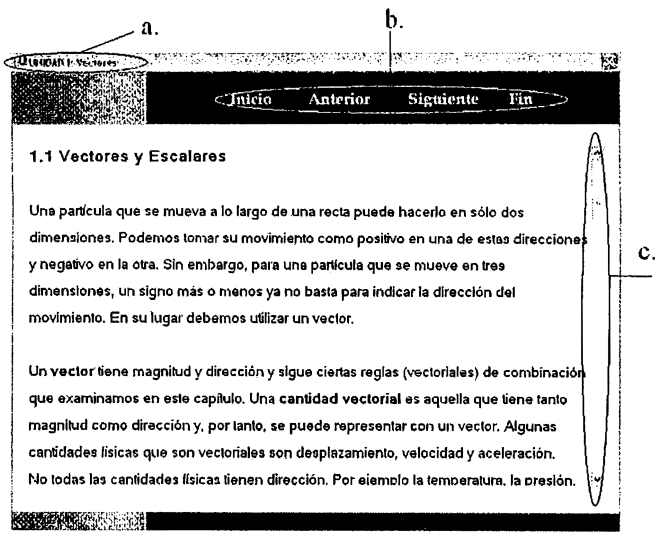


Fig. No.2.1 Desarrollo de contenido teórico

a. Indica el número y nombre de la unidad en estudio.
b. **Botones de desplazamiento:**
   Inicio: Se ubica en el primer tema de la unidad.
   Anterior: Se desplaza al tema anterior de estudio.
   Siguiente: Se desplaza al siguiente tema a estudiar.
   Fin: Se ubica en el último tema de la unidad.
c. **Barra de desplazamiento:** Sirve para desplazarse en forma vertical a través del formulario.

3. SIMULACIONES

Esta sección de la aplicación consta del desarrollo de las simulaciones prácticas existentes para cada unidad de estudio.

A continuación se detalla el funcionamiento de cada una de ellas.

3.1 BOTONES DE CONTROL

A lo largo del desarrollo de las simulaciones se encontrará con Botones de Control que son de uso común en éstas. La función de dichos botones se detalla a continuación.

- El botón de control "Iniciar" o "Evaluar" (vea la figura No.3.1) sirve para que una vez introducidos los parámetros necesarios para el desarrollo de un ejercicio se comience la simulación.

Fig. No.3.1 Botón de control "Iniciar"

- El botón de control "Detener" o "Pausa" (vea la figura No.3.2) sirve para que una vez iniciada la simulación, ésta se pueda detener momentáneamente y poder así observar valores instantáneos.

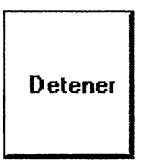


Fig. No.3.2 Botón de control "Detener"

- El botón de control "Seguir" o "Continuar" (vea la figura No.3.3) se visualiza sólo si se ha presionado el botón "Detener" y sirve para continuar el desarrollo de la práctica hasta finalizarla.

Fig. No.3.3 Botón de control "Seguir"

- El botón de control "Tabla de resultados" (vea la figura No.3.4) muestra una tabla (vea la figura No.3.5) que contiene los últimos cinco resultados obtenidos de las prácticas.

Fig. No.3.4 Botón de control "Tabla de resultados"

La tabla de resultados de la figura No.3.5 muestra los últimos cinco resultados obtenidos de las simulaciones. Tenga en cuenta que estos valores permanecen en memoria sólo si la práctica está en ejecución.

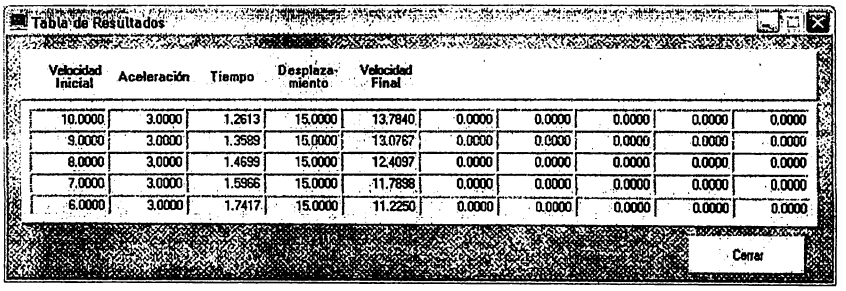


Fig. No.3.5 Tabla de resultados

3.2 VECTORES

Esta práctica consiste en poder asignar las coordenadas en x y y para dibujar en la pantalla los vectores A y B, y así poder realizar diferentes operaciones con ellos.

El entorno con el cual usted debe familiarizarse para el desarrollo de esta práctica es el mostrado en la figura No.3.6:

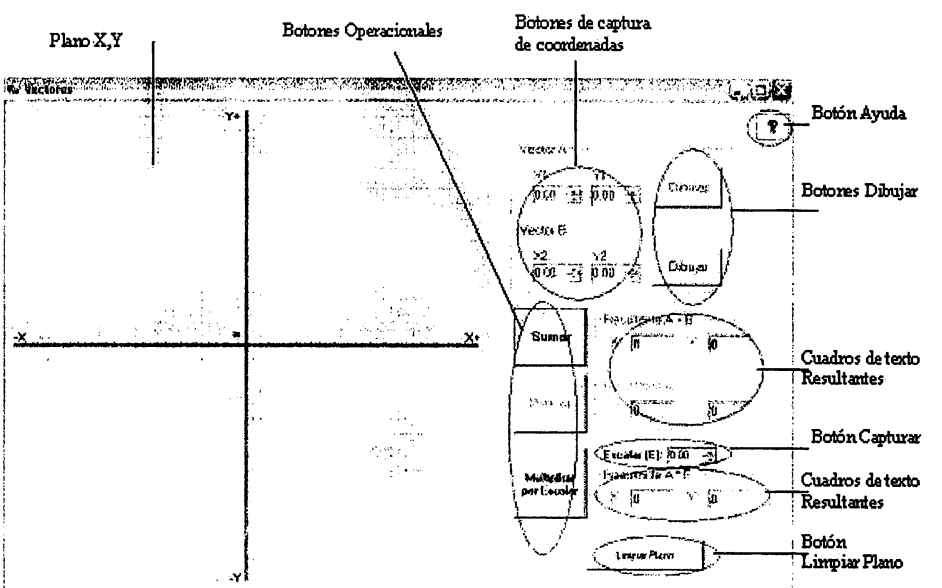


Fig. No.3.6 Simulación de Vectores

- **Plano X,Y:** Plano en el cual se dibujarán los vectores.
- **Botones Operacionales:** Sirven para realizar operaciones según indican Suma, Resta y Multiplicación.
- **Botones de captura de coordenadas:** Sirven para capturar las coordenadas en X y Y de los Vectores A y B.
- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.

- **Botones Dibujar:** Dibuja en pantalla los vectores.
- **Cuadros de texto resultantes:** Muestra las coordenadas en X y Y del vector Resultante.
- **Botón Capturar:** Captura el valor para un escalar.
- **Botón Limpiar plano:** Limpia el plano para nuevas operaciones.

Con este simulador usted podrá crear como máximo dos vectores (A y B) con las coordenadas que desee, sean estas positivas, cero o negativas. Además podrá asignarle un valor a un escalar (E).

Las operaciones que podrá realizar son:

- Suma
- Resta
- Multiplicación por un escalar

3.2.1 DIBUJAR UN VECTOR

El proceso para dibujar vectores es muy sencillo, veamos un ejemplo, dibujemos el vector A con coordenada en X = 6 y coordenada en Y = 8.

Primero debe digitar las coordenadas (X,Y) en los cuadros de captura tal como se muestra en la figura No.3.7.

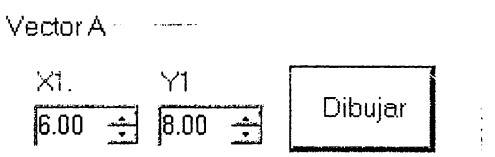


Fig. No.3.7 Dibujando el vector A

Note que la coordenada en X la llamamos X1 y la coordenada en Y es Y1. Para el vector B tendremos X2 y Y2 respectivamente.

Luego de haber digitado las coordenadas simplemente hacemos clic en el botón Dibujar y nuestro vector se dibujará en pantalla como se muestra en la figura No.3.8. El vector A se dibujará en color Rojo, el vector B en Azul.

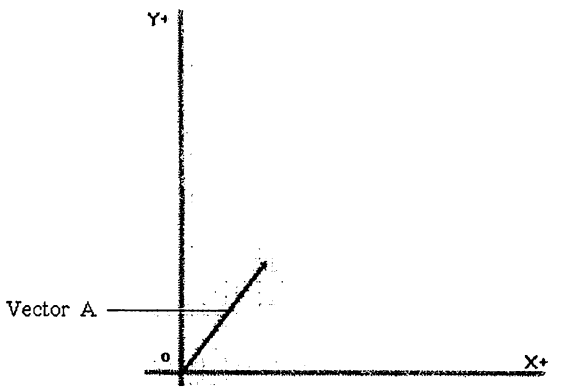


Fig. No.3.8 Vector A dibujado en el plano

Ahora, como ejercicio dibuje el vector B con coordenadas (2,5).

3.2.2 SUMA DE DOS VECTORES

Para sumar dos vectores **A** y **B** se procede de la siguiente manera: a partir del extremo de **A** se lleva el vector **B**; el vector cuyo origen es el origen de **A** y cuyo extremo es el extremo de **B**, es el ***vector suma A + B***.

Entonces para desarrollar la práctica usted debe dibujar el vector **A** y **B**, tal y como se hizo en la sección 3.1.1

Siguiendo el ejemplo de la sección 3.1.1 tenemos las coordenadas del vector **A** y **B** de la siguiente manera:

Vector A: (6,8)

Vector B: (2,5)

Habiendo dibujado los vectores A y B solo debe hacer clic en el botón *Sumar* (Ver figura No.3.9) y el vector Resultante se dibujará en la pantalla en color verde y podrá ver las coordenadas resultantes en los cuadros que aparecen junto al botón *Sumar.*

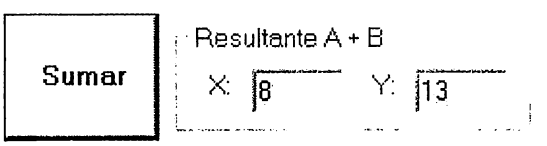


Fig. No.3.9 Botón Sumar y Coordenadas Resultantes

Se preguntará de donde salen las coordenadas resultantes, ¿verdad?, pues bien, simplemente se suman las coordenadas en x del vector A con las coordenadas en x del vector B y así se obtiene la coordenada en x del vector Resultante A + B. De la misma forma se suman las coordenadas en y de los vectores para obtener la coordenada en y del vector Resultante A + B, veamos:

$$x_1 + x_2 = 6 + 2 = 8$$
$$y_1 + y_2 = 8 + 5 = 13$$

En pantalla lo que usted obtendrá como resultado de la Suma será el vector A, B, la proyección de B y A+B tal como se muestra en la figura No.3.10.

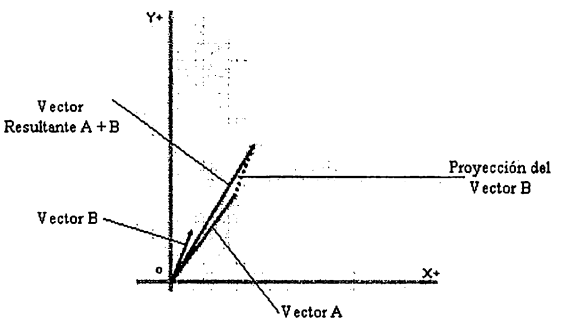


Fig. No.3.10 Vectores A, B y Resultante de la Suma

3.2.3 RESTA DE DOS VECTORES

La resta de dos vectores se lleva a cabo de manera similar a la suma, solamente que en lugar de sumar las coordenadas las restaremos.

Utilizaremos siempre los vectores A y B de la sección 3.1.1:

Vector A: (6,8)

Vector B: (2,5)

Teniendo los vectores A y B dibujados en el plano, proceda a realizar la resta dando clic en el botón *Restar* (Ver figura No.3.11) y el vector Resultante se dibujará en la pantalla en color naranja y podrá ver las coordenadas resultantes en los cuadros que aparecen junto al botón *Restar*.

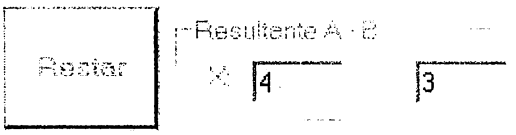


Fig. No.3.11 Botón Sumar y Coordenadas Resultantes

Nuevamente se preguntará de donde salen las coordenadas resultantes, pues bien, es igual a la operación que se hace en la suma, solo que esta vez se resta. Se restan las coordenadas en x del vector A con las coordenadas en x del vector B y así se obtiene la coordenada en x del vector Resultante A - B. De la misma forma se restan las coordenadas en y de los vectores para obtener la coordenada en y del vector Resultante A - B.

$$x_1 - x_2 = 6 - 2 = 4$$
$$y_1 - y_2 = 8 - 5 = 3$$

El vector Resultante de esta operación, el vector A, B y la proyección de B se mostrarán en pantalla como se muestra en la figura No.3.12.

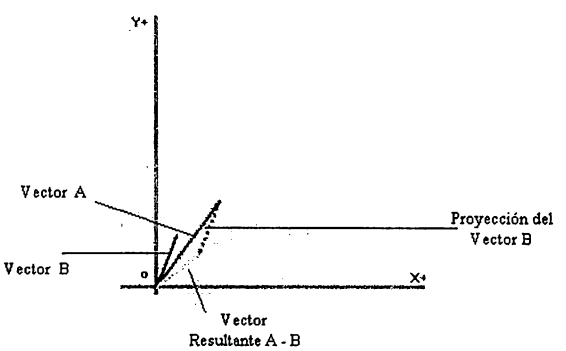


Fig. No.3.12 Vectores A, B y Resultante de la Resta

3.2.4 MULTIPLICACIÓN DE UN VECTOR POR UN ESCALAR

La multiplicación de un vector por un escalar es sencilla, simplemente se multiplican las coordenadas del vector por el escalar y se obtiene una resultante, veamos:

Teniendo dibujado el vector A (6,8) (Ver sección 3.1.1), digite el valor del escalar en el botón de captura que, para nuestro ejemplo tiene un valor de 3 como en la figura No.3.13, luego haga clic en el botón Multiplicar para que se realice la operación. El vector Resultante se dibujará en pantalla en color violeta y al mismo tiempo las coordenadas de este se desplegarán en los cuadros de texto.

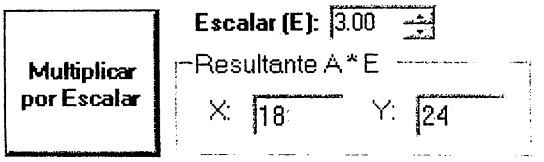


Fig. No.3.13 Recuadro para definición de Escalar, Botón de Multiplicar Y Cuadros de texto para desplegar coordenadas resultantes.

El vector A y el vector Resultante de la Multiplicación se dibujarán en pantalla tal y como se muestra en la figura No.3.14.

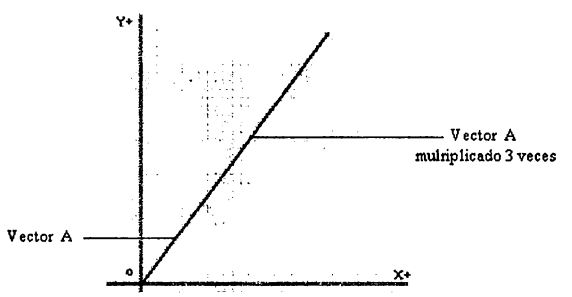


Fig. No.3.14 Vectores A y Resultante de la Multiplicación por un Escalar (E).

3.3 MOVIMIENTO HORIZONTAL

Con esta práctica usted experimentará el movimiento de un objeto sobre una línea recta horizontal y para ello podrá definir una escala de trabajo (en metros), parámetros de inicialización y finalización, además, con los Botones de Control usted podrá como su nombre lo indica mantener el control del desarrollo de la práctica.

El área de trabajo para esta práctica es la mostrada en la figura No.3.15.

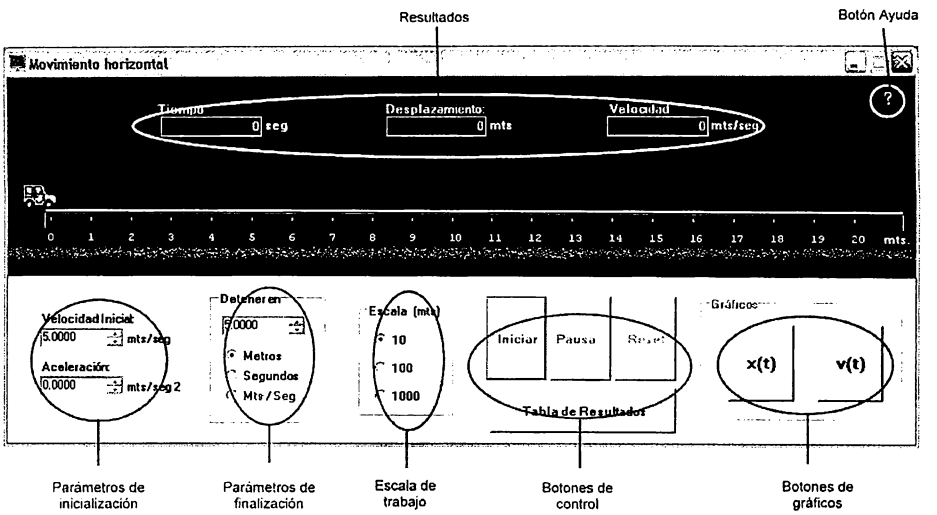


Fig. No.3.15 Simulación de Movimiento Horizontal

- **Resultados:** Muestran el resultado final de la práctica.
- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Parámetros de Finalización:** Captura los parámetros con los cuales desea que la práctica termine.
- **Escala de trabajo:** Permite seleccionar la escala con la cual se desea trabajar, ésta puede ser 10, 100 o 1000 metros.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Botones de gráfico:** Muestra un formulario con las gráficas posición – tiempo ó velocidad – tiempo.

3.3.1 ECUACIONES

Las ecuaciones utilizadas para el desarrollo de la práctica son las que se listan a continuación:

- $v = v_o + at$
- $x - x_o = V_o t + \frac{1}{2} at^2$

3.3.2 EJEMPLO

En una prueba de frenado se observa que un coche es detenido en 3s. ¿Cuál es la distancia de frenado si la velocidad inicial del automóvil era 16.68 m/s y la aceleración –5.56 m/s^2?.

Solución:

Sabemos que el coche tiene como Velocidad Inicial 16.68 m/s y una aceleración de –5.56 m/s^2. Además tenemos un tiempo de 3 segundos. Queremos averiguar la distancia recorrida en ese tiempo.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.16:

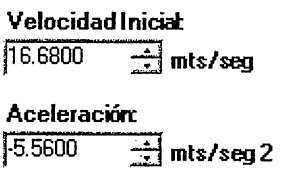


Fig. No.3.16 Parámetros de inicialización

2. Establezca el parámetro de finalización, que para nuestro ejemplo será de 3s. Vea la figura No.3.18:

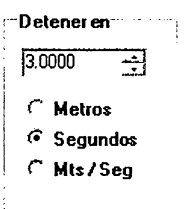


Fig. No.3.18 Parámetros de finalización

3. Defina una escala de trabajo dando clic en la ruedita que corresponde a la escala deseada, para nuestro ejemplo utilizaremos la escala de 100mts. Como en la figura No.3.17:

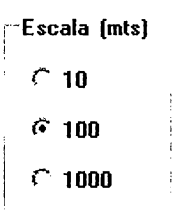


Fig. No.3.17 Escala de trabajo

4. Haga clic en el botón Iniciar y observe que los cuadros de resultados contienen los datos finales del ejercicio, vea la figura No.3.19:

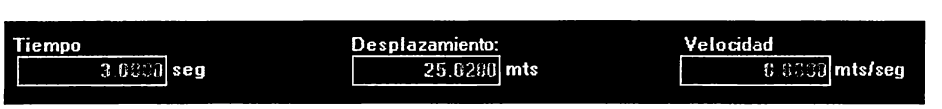


Fig. No.3.19 Resultados

3.4 MOVIMIENTO VERTICAL

En esta práctica usted podrá experimentar el movimiento de un objeto sobre una línea recta vertical, para lo cual tendrá que definir una escala de trabajo (en metros), parámetros de inicialización y finalización, además, con los Botones de Control usted podrá como su nombre lo indica mantener el control del desarrollo de la práctica.

El área de trabajo para esta práctica es la mostrada en la figura No.3.20.

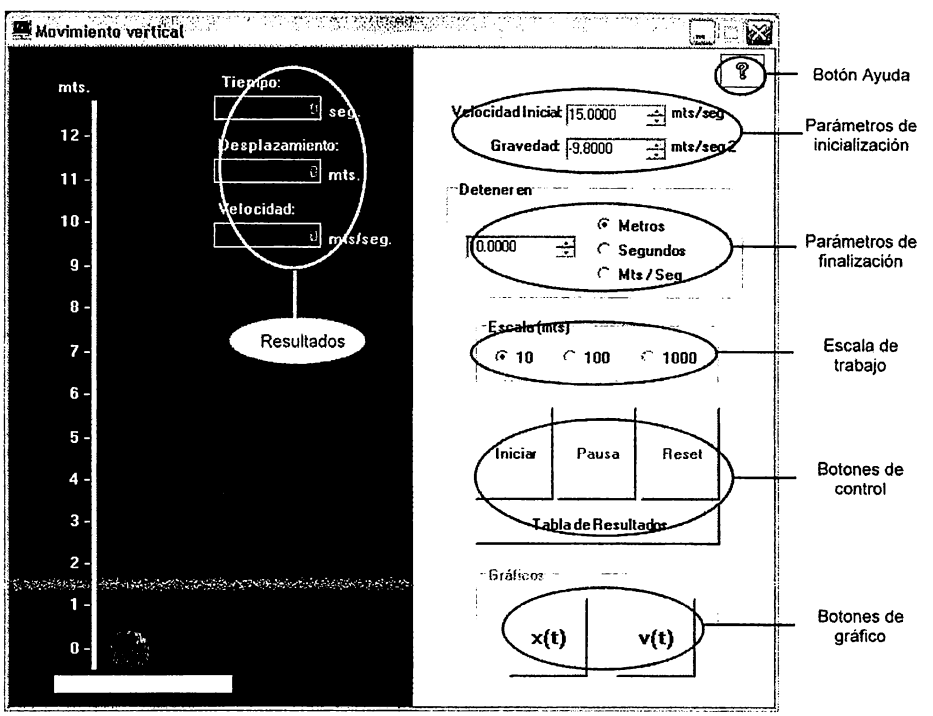


Fig. No.3.20 Simulación de Movimiento Vertical

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Parámetros de Finalización:** Captura los parámetros con los cuales desea que la práctica termine.
- **Escala de trabajo:** Permite seleccionar la escala con la cual se desea trabajar, ésta puede ser 10, 100 o 1000 metros.
- **Botones de control**: Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Botones de gráfico:** Muestra un formulario con las gráficas posición – tiempo ó velocidad – tiempo.
- **Resultados:** Muestran el resultado final de la práctica.

3.4.1 ECUACIONES

Las diferentes ecuaciones utilizadas para el movimiento vertical son las mostradas a continuación:

- $v = v_o - gt$
- $y = v_o t - \frac{1}{2} g t^2$

3.4.2 EJEMPLO

Una pelota de béisbol se lanza hacia arriba con una velocidad inicial de 20 m/s, ¿Cuánto tiempo transcurre para que alcance una altura de 19 m?

Solución:

Sabemos que la velocidad Inicial es de 20 m/s y tenemos un desplazamiento de 19 mts. Queremos averiguar el tiempo que tarda en desplazarse los 19 mts.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.21:

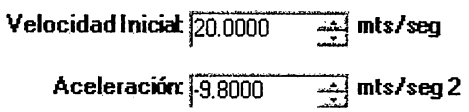


Fig. No.3.21 Parámetros de inicialización

2. Establezca el parámetro de finalización, que para nuestro ejemplo será de 19m. Vea la figura No.3.22:

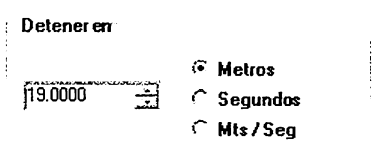


Fig. No.3.22 Parámetros de finalización

3. Defina una escala de trabajo dando clic en la ruedita que corresponde a la escala deseada, para nuestro ejemplo utilizaremos la escala de 100mts. Como en la figura No.3.23:

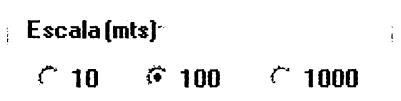


Fig. No.3.23 Escala de trabajo

4. Haga clic en el botón Iniciar y observe que los cuadros de resultados contienen los datos finales del ejercicio, vea la figura No.3.24:

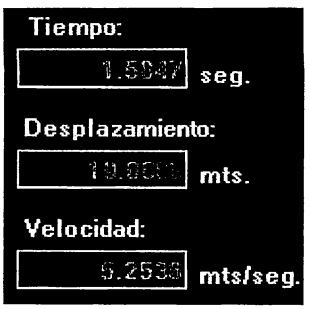


Fig. No.3.24 Resultados

3.5 MOVIMIENTO PARABÓLICO

En esta práctica se experimentará el movimiento de un objeto en una trayectoria parabólica, definiendo una escala de trabajo (en metros), parámetros de inicialización y finalización, además, con los Botones de Control usted podrá como su nombre lo indica mantener el control del desarrollo de la práctica.

El área de trabajo para esta práctica es la mostrada en la figura No.3.25.

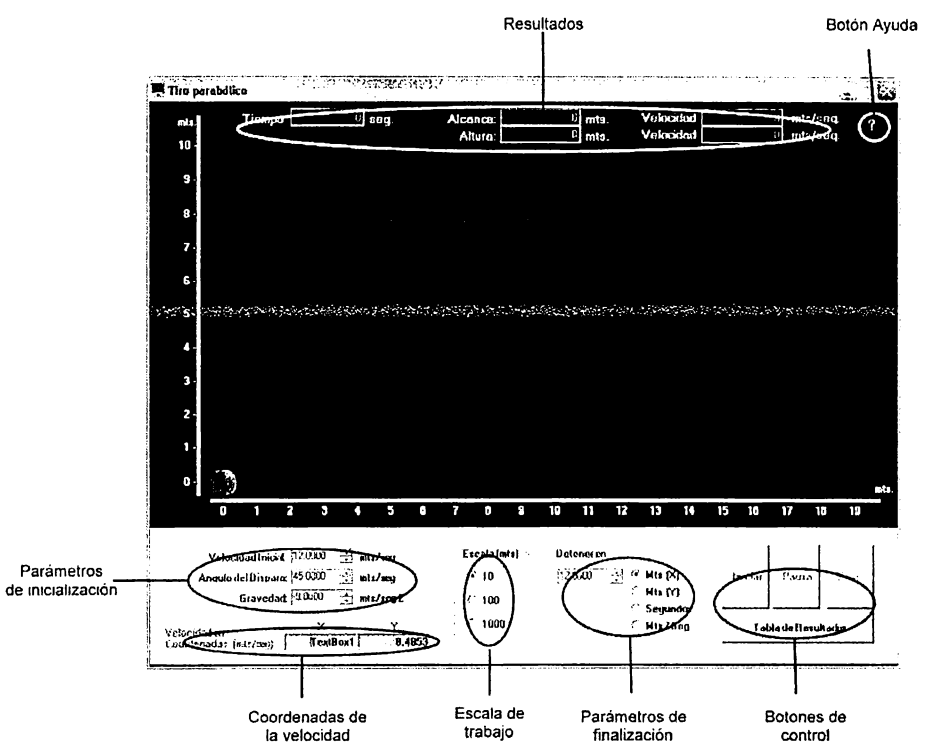


Fig. No.3.25 Simulación de Movimiento Parabólico

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Resultados:** Muestran el resultado final de la práctica.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Escala de trabajo:** Permite seleccionar la escala con la cual se desea trabajar, ésta puede ser 10, 100 o 1000 metros.
- **Parámetros de Finalización:** Captura los parámetros con los cuales desea que la práctica termine.

3.5.1 ECUACIONES

Las diferentes ecuaciones utilizadas para el movimiento parabólico son las mostradas a continuación:

- $y = (v_o t) + \frac{1}{2} a t^2$
- $x = v_x t$
- $v_y = v_o + at$

3.5.2 EJEMPLO

Un bateador golpea una bola de modo que ésta adquiere una velocidad inicial de 37.0 m/s con un ángulo inicial de 53.1° en un lugar donde la gravedad es de 9.8 m/s ^{2}. Después de un tiempo de 6.04s, la bola cae al suelo, ¿Cuál es su velocidad componente sobre el eje x?, ¿cuál es la componente vertical sobre el eje y al término de ese tiempo?.

Solución:

Tenemos que la velocidad inicial es de 37.0 m/s, un ángulo de 53.1° , además conocemos la gravedad que es de 9.8 m/s2 y un tiempo de 6.04s.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.26:

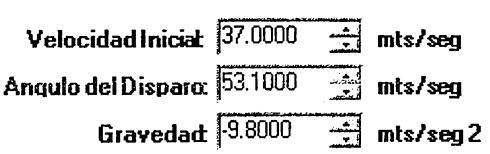


Fig. No.3.26 Parámetros de inicialización

2. Establezca el parámetro de finalización, que para nuestro ejemplo será de 6.04s. Vea la figura No.3.27:

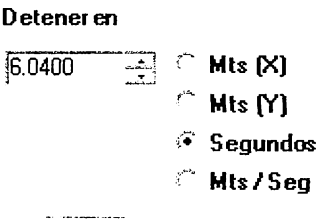


Fig. No.3.27 Parámetros de finalización

3. Defina una escala de trabajo dando clic en la ruedita que corresponde a la escala deseada, para nuestro ejemplo utilizaremos la escala de 100mts. Como en la figura No.3.28:

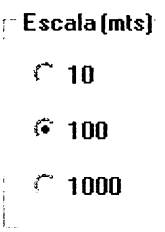


Fig. No.3.28 Escala de trabajo

4. Haga clic en el botón Iniciar y observe que los cuadros de resultados finales del ejercicio, vea la figura No.3.29:

Fig. No.3.29 Resultados

3.6 MOVIMIENTO CIRCULAR

En esta práctica podrá experimentar el movimiento cuando una partícula se mueve en un círculo con rapidez constante.

El área de trabajo para esta práctica es la mostrada en la figura No.3.30.

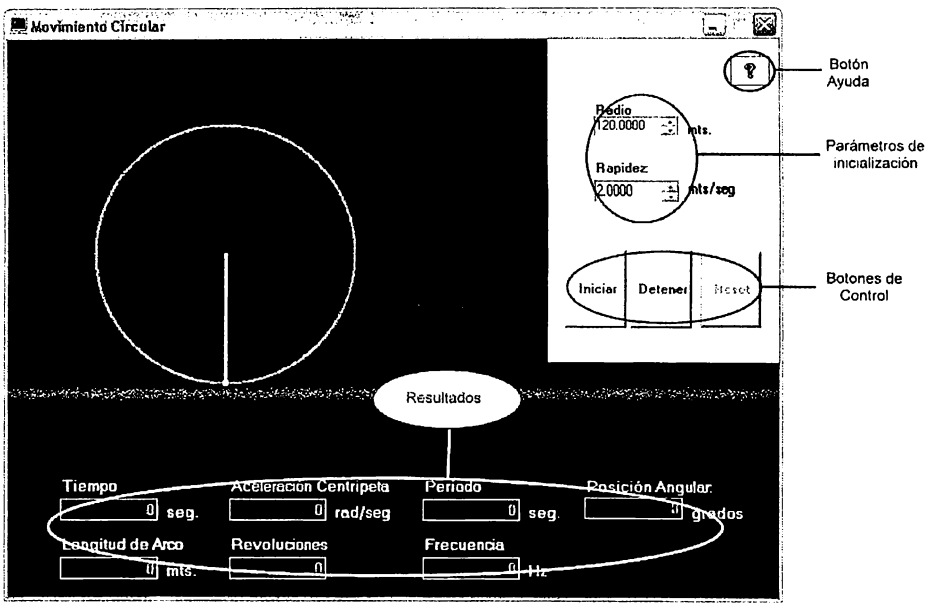


Fig. No.3.30 Simulación de Movimiento Circular Uniforme

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Resultados:** Muestran el resultado final de la práctica.

3.6.1 ECUACIONES

Las diferentes ecuaciones utilizadas para el movimiento circular uniforme son las mostradas a continuación:

- $\theta = (\omega t)$
- $T = \frac{2\pi r}{v}$
- $a_c = \frac{v^2}{r}$

3.6.2 EJEMPLO

En un juego mecánico de feria, los pasajeros viajan con rapidez constante de 7.9m/s en un círculo de 5m de radio, dando una vuelta cada 4s. ¿Qué aceleración tienen?.

Solución:

Tenemos que la velocidad inicial es de 7.9m/s, además conocemos el radio del círculo, que es de 5m. Adicionalmente sabemos que el período es de 4s.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.31:

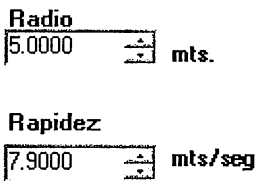


Fig. No.3.31 Parámetros de inicialización

2. Haga clic en el botón Iniciar y observe que la aceleración centrípeta y el período se mantienen en los cuadros de resultados, vea la figura No.3.32:

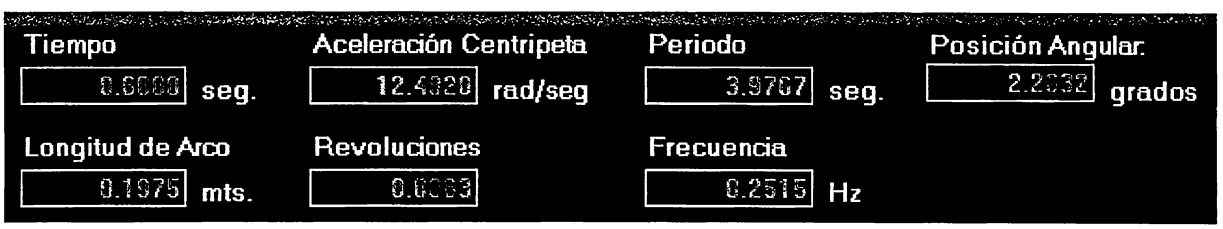


Fig. No.3.32 Resultados

3.7 LEYES DE NEWTON

En esta práctica usted podrá experimentar el movimiento de un sistema, con un bloque deslizante y otro colgante unidos por medio de una polea con masa despreciable en comparación con la masa de los bloques.

El área de trabajo para esta práctica es la mostrada en la figura No.3.33.

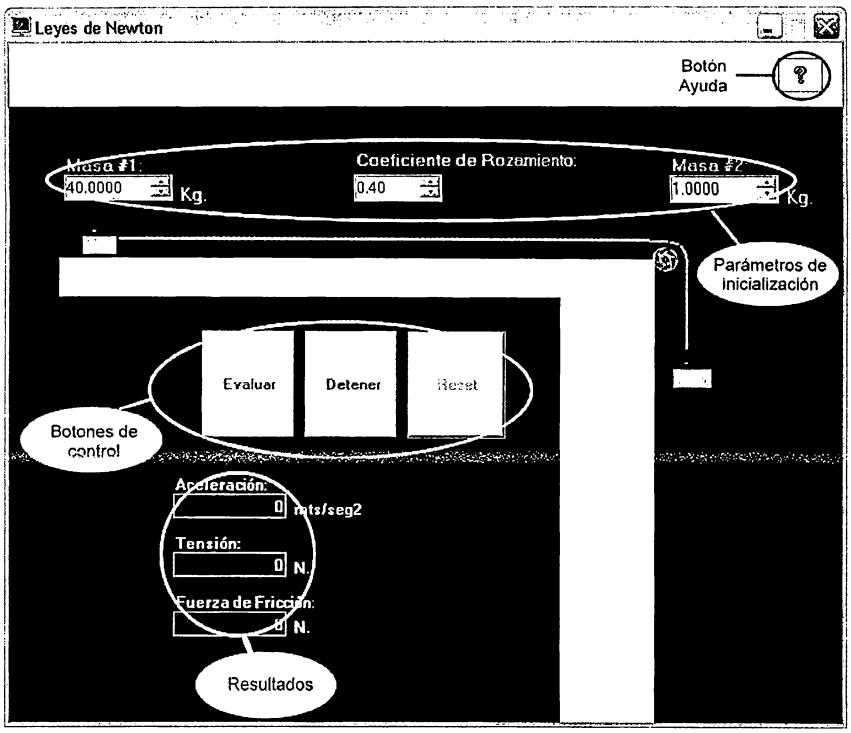


Fig. No.3.33 Simulación de Leyes de Newton

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Resultados:** Muestran el resultado final de la práctica.

3.7.1 ECUACIONES

Las diferentes ecuaciones utilizadas en ésta práctica son las mostradas a continuación:

- $F_r = \mu_r m_1 g$
- $a = \dfrac{((m_2 g) - F_r)}{m_1 + m_2}$
- $F = (m_1 a) + F_r$

3.7.2 EJEMPLO

Un bloque deslizante con masa 3.3kg se encuentra sobre una superficie sin fricción y unido por una cuerda que se enrolla sobre una polea (sin fricción) a un segundo bloque (colgante) con masa 2.1kg. La cuerda y la polea tienen masas despreciables en comparación con los bloques. El bloque colgante cae cuando el deslizante acelera hacia la derecha. Encuentre la aceleración para ambos bloques y la tensión en la cuerda.

Solución:

Tenemos los siguientes valores m1=3.3kg, m2=2.1kg, debido a que se encuentra sobre una superficie sin fricción tenemos que $\mu_r = 0$.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.34:

Fig. No.3.34 Parámetros de inicialización

2. Haga clic en el botón Evaluar y observe que los cuadros de resultados contienen los datos finales del ejercicio, vea la figura No.3.35:

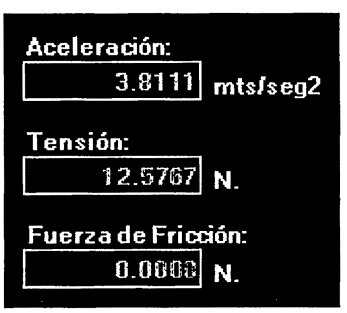


Fig. No.3.35 Resultados

3.8 TRABAJO Y ENERGÍA

En esta práctica usted podrá observar la conservación de la energía mecánica por medio de un resorte que está unido a una masa, con o sin rozamiento.

El área de trabajo para esta práctica es la mostrada en la figura No.3.36.

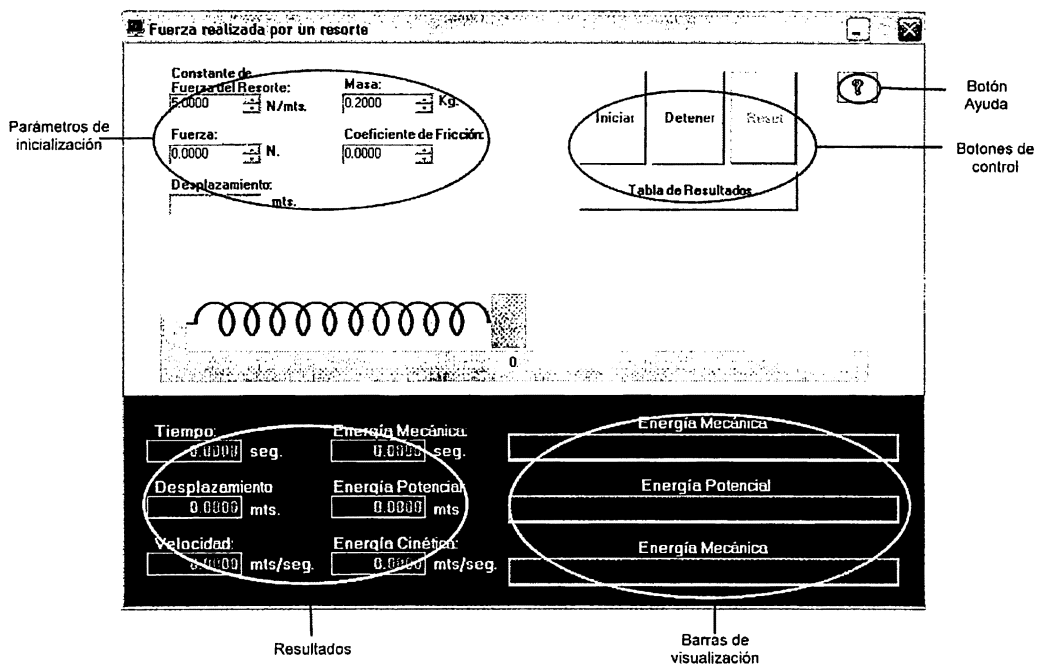


Fig. No.3.36 Simulación de Conservación de la energía mecánica

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Resultados:** Muestran el resultado final de la práctica.
- **Barras de visualización:** Muestran el comportamiento de la conservación de la energía mecánica.

3.8.1 ECUACIONES

Las diferentes ecuaciones utilizadas en ésta práctica son las mostradas a continuación:

- Con rozamiento
  - $\omega = \sqrt{\left(\frac{k}{m}\right) - \left(\frac{F_r^2}{4m^2}\right)}$
  - $x = x_o e^{\left(\frac{-F_r t}{2m}\right)} \cos(\omega t)$
  - $E_{mec} = \frac{1}{2} k x^2 e^{\left(\frac{-F_r t}{m}\right)}$
  - $E_k = E_m - E_p$

- Sin rozamiento
  - $\omega = \sqrt{\left(\frac{k}{m}\right)}$
  - $E_m = \frac{1}{2} k x^2$
  - $x = x_o \cos(\omega t)$
  - $E_p = \frac{1}{2} k x^2 \cos^2(\omega t)$
  - $E_p = \frac{1}{2} k x^2 \sin^2(\omega t)$

3.8.2 EJEMPLO

Un bloque de 2.5kg está unido al extremo de un resorte que tiene constante de resorte de 50N/m. El resorte se comprime 7.6cm. El coeficiente de fricción cinética entre el bloque y la superficie horizontal es de 0.25. Mientras el bloque se lleva al reposo, ¿Cuál es el trabajo realizado por el resorte?

Solución:

Tenemos los siguientes valores: masa de 2.5kg, constante del resorte 50N/m, y el coeficiente de fricción cinética de 0.25.

1. Introduzca los parámetros de inicialización tal y como se muestra en la figura No.3.37:

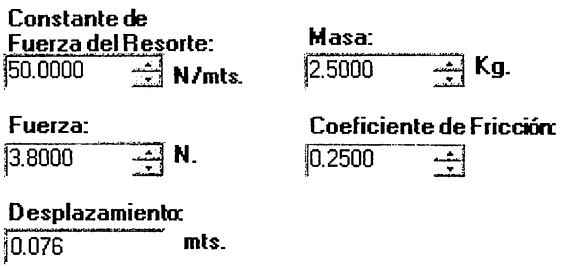


Fig. No.3.37 Parámetros de inicialización

NOTA: La fuerza la calculamos con las flechas arriba y abajo hasta obtener el desplazamiento de 0.076mts.

2. Haga clic en el botón Iniciar y observe que los cuadros de resultados contienen los datos finales del ejercicio, vea la figura No.3.38:

Fig. No.3.38 Resultados

3.9 COLISIONES

Con esta práctica usted podrá experimentar el movimiento cuando dos cuerpos se dirigen a una colisión elástica unidimensional.

El área de trabajo para esta práctica es la mostrada en la figura No.3.39.

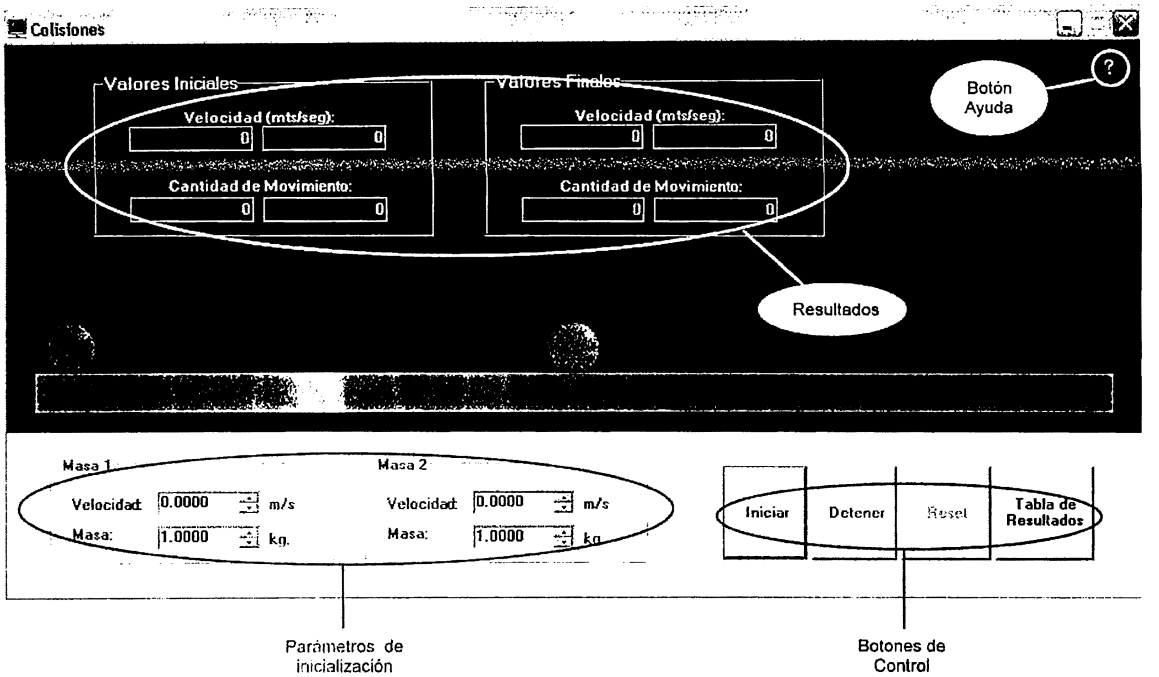


Fig. No.3.39 Simulación de Colisiones

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Resultados:** Muestran el resultado final de la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.

3.9.1 ECUACIONES

Las diferentes ecuaciones utilizadas en ésta práctica son las mostradas a continuación:

- $v_{f1} = \left(\frac{m_1 - m_2}{m_1 + m_2}\right) v_{1i} + \left(\frac{2m_2}{m_1 + m_2}\right) v_{2i}$
- $v_{f2} = \left(\frac{2m_1}{m_1 + m_2}\right) v_{1i} + \left(\frac{m_2 - m_1}{m_1 + m_2}\right) v_{2i}$
- $p = mv$

3.9.2 EJEMPLO

Considere dos esferas metálicas, inicialmente en reposo, la primera con masa = 0.03kg y la segunda 0.075kg. A la esfera 1 se le aplica una velocidad inicial de 1.252m/s. Encuentre la velocidad final para ambas esferas y la cantidad de movimiento lineal.

Solución:

Tenemos los siguientes valores m_1=0.03kg , m_2=0.075kg, V_{1i}=1.252m/s y V_{2i}=0m/s.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.40:

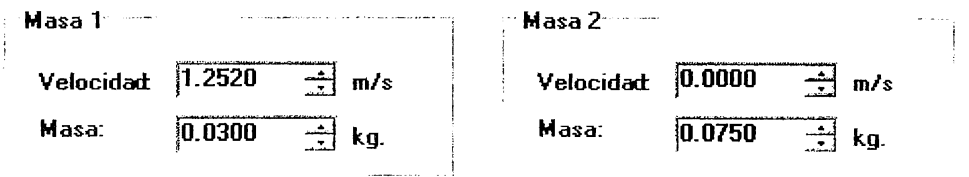


Fig. No.3.40 Parámetros de inicialización

2. Haga clic en el botón Iniciar y observe que los cuadros de resultados contienen los datos finales del ejercicio, vea la figura No.3.41:

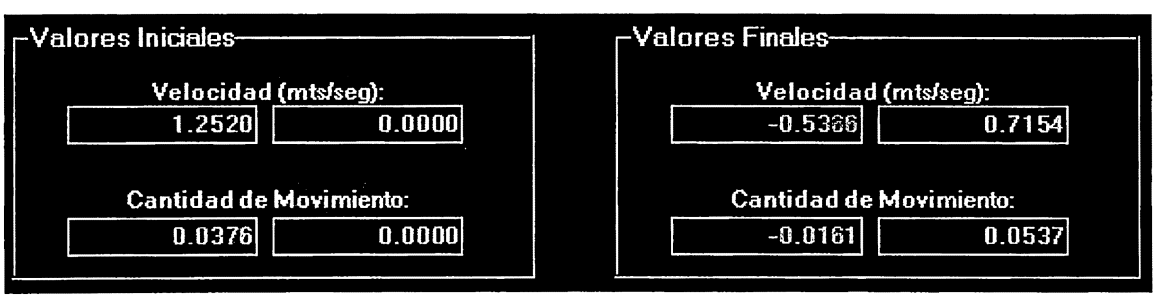


Fig. No.3.41 Resultados

3.10 ROTACIÓN

Con esta práctica usted podrá experimentar el movimiento de rotación con aceleración constante.

El área de trabajo para esta práctica es la mostrada en la figura No.3.42.

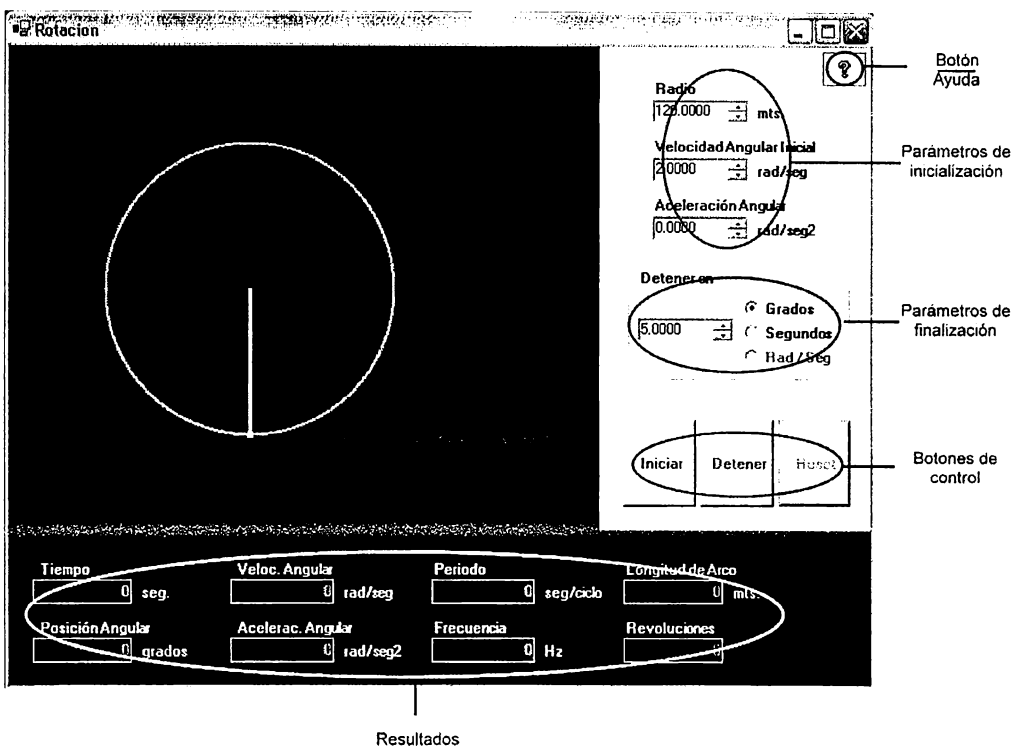


Fig. No.3.42 Simulación de Rotación

- **Botón Ayuda:** Despliega en pantalla ayuda sobre cómo realizar la práctica.
- **Parámetros de Inicialización:** Sirven para capturar los parámetros iniciales de la simulación.
- **Parámetros de Finalización:** Captura los parámetros con los cuales desea que la práctica termine.
- **Botones de control:** Permiten como su nombre lo indica mantener el control de la práctica, con ellos se puede iniciar, detener o reiniciar la práctica en el momento que se desee.
- **Resultados:** Muestran el resultado final de la práctica.

3.10.1 ECUACIONES

- $\theta = (\omega t)$, Si no hay aceleración
- $\theta = \left(\frac{\omega + \omega_o}{2}\right)t$, Si hay aceleración
- $\omega = \omega_o + (\alpha t)$
- $T = \frac{2\pi}{\omega}$

3.10.2 EJEMPLO

Una piedra de afilar, gira a una aceleración angular constante de 0.35rad/s^2. En el tiempo t =0 tiene una velocidad angular de –4.6rad/s y una recta de referencia en ella es horizontal en la posición angular 0. En qué tiempo después de t = 0 está la recta en posición angular de 5 rev.

Solución:

Para el desarrollo del ejercicio no nos interesa saber el radio por lo que lo omitimos, tenemos una aceleración angular constante de 0.35rad/s^2, una velocidad angular de –4.6rad/s, para 5 revoluciones tenemos que son 1800 grados.

1. Introduzca los parámetros de inicialización así como se muestra en la figura No.3.43:

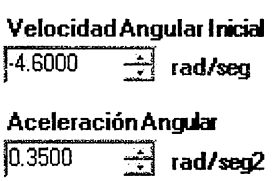


Fig. No.3.43 Parámetros de inicialización

2. Establezca el parámetro de finalización, que para nuestro ejemplo será de 1800 grados. Vea la figura No.3.44.

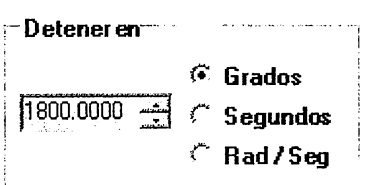


Fig. No.3.44 Parámetros de finalización

3. Haga clic en el botón Iniciar y observe que los cuadros de resultados contienen los datos finales del ejercicio, vea la figura No.3.44:

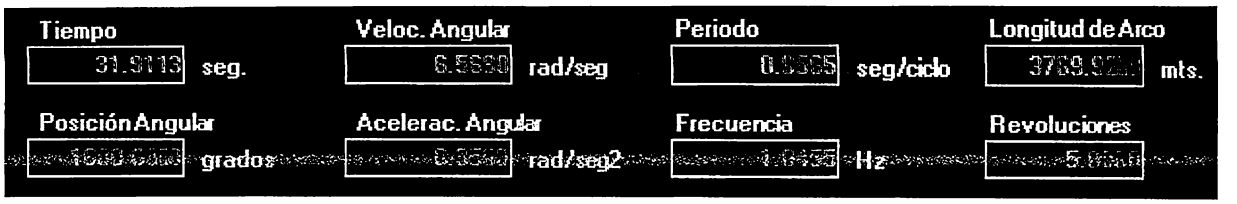


Fig. No.3.44 Resultados

4. EVALUADOR

El evaluador es una aplicación del programa que permite al usuario realizar una prueba de conocimientos que se basa en el contenido teórico de la unidad en estudio.

El proceso de desarrollo de la evaluación es fácil, primero se debe seleccionar el número de preguntas que se desea para la prueba (ver figura 4.1), se tiene tres opciones: 5, 10 y 20 . Si está seguro de realizar la prueba, haga clic en el botón *Continuar*, caso contrario deberá hacer clic en el botón *Cancelar*.

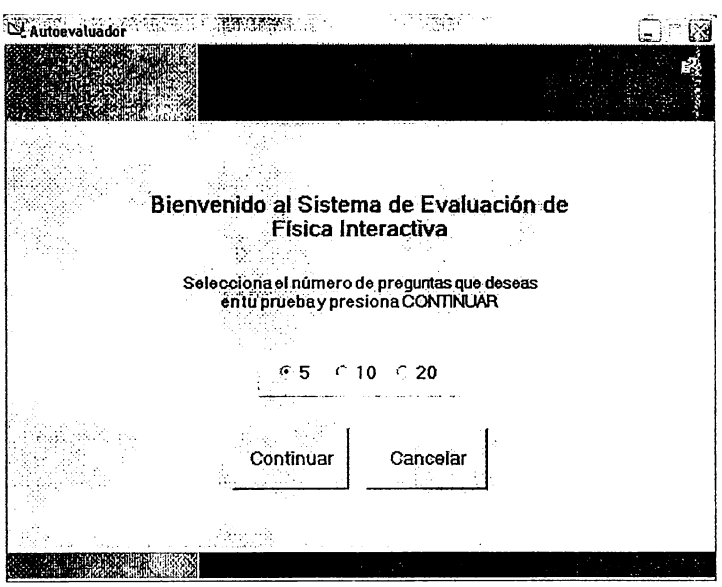


Fig. No.4.1 Pantalla inicial del evaluador

Luego de haber hecho clic en el botón *Continuar*, se le mostrará una pantalla como la mostrada en la figura 4.2.

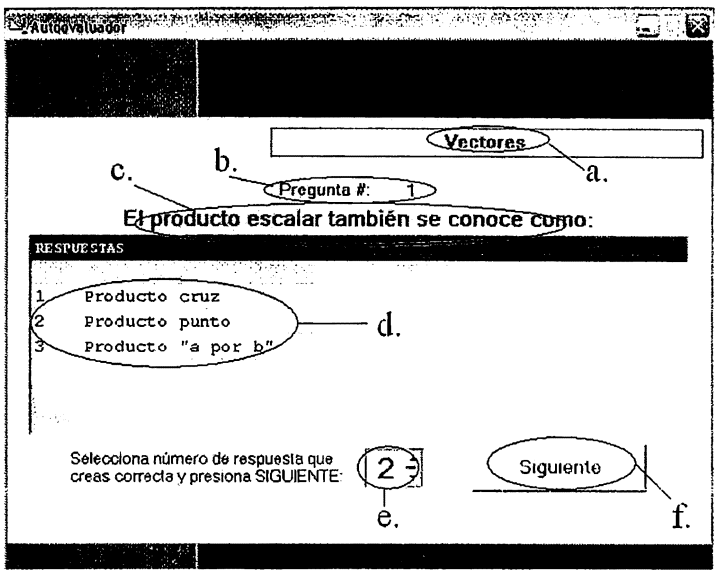


Fig. No.4.2 Pantalla del evaluador

A continuación se le detallan las partes de esta pantalla:

a. Nombre de la unidad que se está evaluando
b. Número de la pregunta en desarrollo
c. Pregunta a desarrollar
d. Posibles respuestas para la pregunta en desarrollo
e. Número de respuesta a seleccionar
f. Botón para ir a la siguiente pregunta

Aquí se va desarrollando una pregunta a la vez, de la siguiente manera: Se le desplegará una pregunta en pantalla (literal c.), para la cual tendrá tres posibles respuestas (literal d.), una de éstas es la correcta y usted podrá seleccionarla dando clic en la flechas arriba y abajo del literal e. Una vez seleccionada su respuesta haga clic en el botón *Siguiente* para continuar con la prueba. Tenga en cuenta que una vez presionado el botón *Siguiente*, usted no podrá regresar a la pregunta anterior.

Cuando haya terminado de contestar las preguntas, se le desplegará un cuadro como el de la figura 4.3, el cual le mostrará el resultado de su prueba.

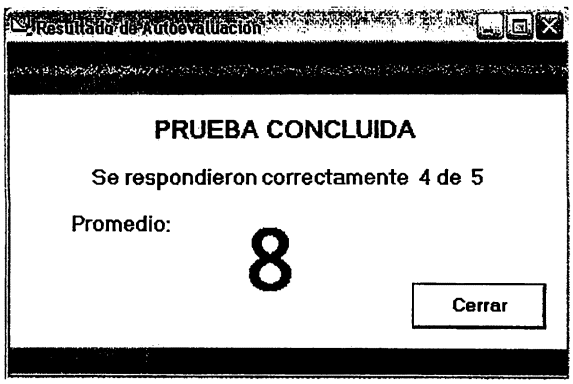


Fig. No.4.3 Cuadro de resultados

Para salirse de la prueba, basta con presionar clic en el botón cerrar.

5. INDICE CONCEPTUAL

El índice de conceptos es una parte del programa que permite al usuario obtener un índice en forma alfabética de los conceptos relevantes relacionados con la unidad de estudio.

La figura 5.1 muestra la pantalla de ésta parte del programa.

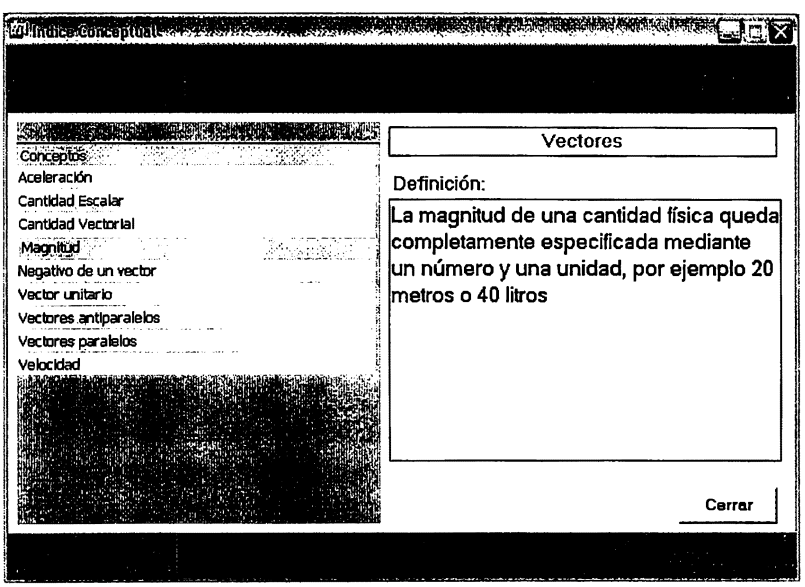


Fig. No.5.1 Índice conceptual

Lo único que debe hacer es seleccionar el concepto de la lista que aparece a la izquierda de su pantalla, ubicando el cursor sobre éste y su respectiva definición será mostrada en la parte derecha de su pantalla.

Manual Técnico

INDICE

1.1 ESTRUCTURA DE INTERFACES 1

1.2 VARIABLES DE ENTORNO 3

1.3 CODIFICACIÓN 3

1.3.1 FORMULARIO: MAIN.VB 3
1.3.2 FORMULARIO: TEORIA.VB 6
1.3.3 FORMULARIO: MENU.VB 7
1.3.4 FORMULARIO: ACELERA.VB 9
1.3.5 FORMULARIO: CAIDALIBRE.VB 13
1.3.6 FORMULARIO: CHOQUE1.VB 16
1.3.7 FORMULARIO: MOVCIRC.VB 19
1.3.8 FORMULARIO: NEWTON.VB 21
1.3.9 FORMULARIO: RESORTE.VB 24
1.3.10 FORMULARIO: ROTACION.VB 26
1.3.11 FORMULARIO: TIPOPARAB.VB 30
1.3.12 FORMULARIO: VECTORES.VB 36
1.3.13 FORMULARIO: AYUDA.VB 38
1.3.14 FORMULARIO: CLDATOS.VB 38
1.3.15 FORMULARIO: EVALUA.VB 39
1.3.16 FORMULARIO: EVALUANOTA.VB 42
1.3.17 FORMULARIO: INDICE.VB 42
1.3.18 FORMULARIO: CONTENIDOHELP.VB 43
1.3.19 FORMULARIO: SIMULACIONHELP.VB 44

1.4 CÓDIGOS DE ERROR 44

1.4.1 DESBORDAMIENTO NUMÉRICO 44
1.4.2 VALOR DE PROPIEDAD NO VÁLIDO 44
1.4.3 ARCHIVO NO ENCONTRADO 44
1.4.4 ARGUMENTO NO VÁLIDO 44
1.4.5 FALLO CONEXIÓN A DATOS 45

1.1 ESTRUCTURA DE INTERFACES

La estructura de interfaces permite visualizar de manera gráfica la jerarquía de las interfaces que componen el sistema Física Interactiva (Ver figura 1.1).

El sistema está compuesto en su estructura de desarrollo por cuatro módulos principales (Contenido Teórico, Simulación, Evaluación, Indice Conceptual) los cuales son accesados a través de un menú principal alojado en Main.vb, el cual aparece dentro del esquema en el nivel superior.

El segundo nivel está compuesto por los archivos Teoria.Vb, el cual despliega el archivo con extensión RTF correspondiente al tema seleccionado; Menu.Vb, el cual contiene el menú de simulaciones disponibles para la unidad en estudio, este a su vez, abre los archivos correspondientes a cada una de las simulaciones quienes tienen acceso a Ayuda.vb y CLDatos.vb.

Continuando con el segundo nivel se puede encontrar los archivos Evalua.vb, que permite ejecutar la prueba de nivel; Indice.vb para ver las lista de conceptos y Ayuda.vb para ayuda general del sistema.

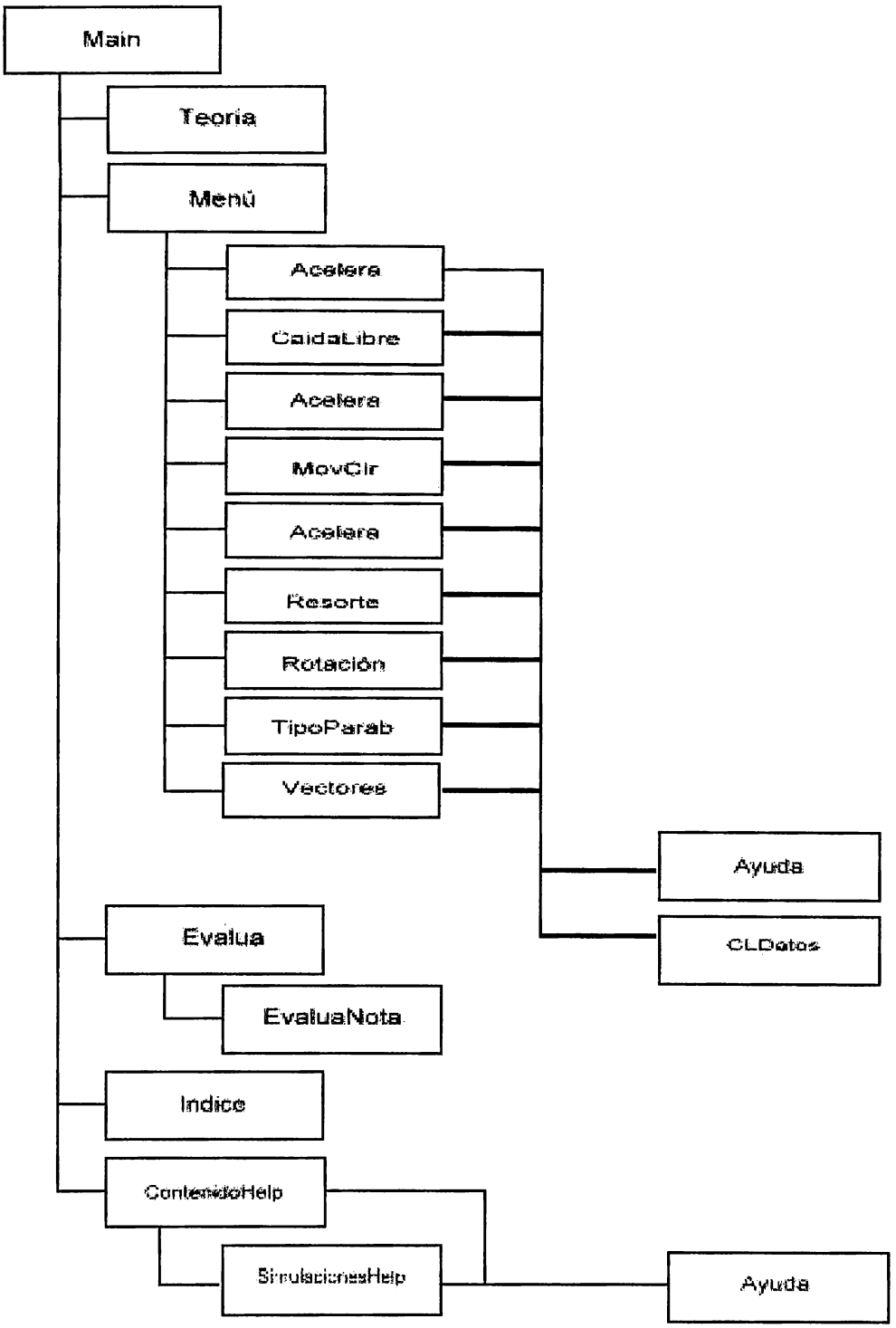


Fig. No. 1.1 Estructura de interfaces

1.2 VARIABLES DE ENTORNO

VARIABLE	DESCRIPCIÓN	CLASE	TIPO
Queunid	Código de unidad en estudio	Integer	Public Shared
TabDt	Matriz de resultados	Double	Public Shared
LabDt	Matriz de encabezados de tabla de resultados	String	Public Shared
X	Representa filas en tablas de resultados	Integer	Public Shared
Y	Representa columnas en tablas de resultados	Integer	Public Shared
Tema	Guarda nombre de archivo actual con contenido teórico	String	Public Shared
Contenido	Guarda nombre de archivo de ayuda actual	String	Public Shared

1.3 CODIFICACIÓN

1.3.1 FORMULARIO: MAIN.VB

Formulario principal del sistema. Contiene menú primario y secundario.

DECLARACION DE VARIABLES:

```
Public Shared QueUnid As Integer
Public Shared TabDt(4, 9) As Double
Public Shared LabDt(0, 9) As String
Public Shared fondo As System.Drawing.Bitmap
Public Shared x, y As Integer
Public Shared tema As String
Public Shared contenido As String
```

CARGA DEL FORMULARIO:

```
Private Sub main_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    QueUnid = 1
    teoria.Visible = False
    Practica.Visible = False
    Evaluacion.Visible = False
    Conceptos.Visible = False
    Ayuda.Visible = False
End Sub
```

LIMPIA VARIABLES DE ENTORNO (TABDT, LABDT):

```
Public Shared Sub Limpiar()
    For x = 0 To 4
        For y = 0 To 9
            TabDt(x, y) = 0
        Next
    Next
    For y = 0 To 9
        LabDt(0, y) = " "
    Next
End Sub
```

CARGA DEL FORMULARIO:

```
Private Sub teoria_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles teoria.Click
    Dim forma As New Teoria()
    forma.Show()
End Sub
```

HABILITA BOTONES DE MENU SECUNDARIO:

```
Private Sub Habilitar()
    teoria.Visible = True
    Practica.Visible = True
    Evaluacion.Visible = True
    Conceptos.Visible = True
    Ayuda.Visible = True
End Sub
```

CONFIGURA COLORES DE MENU PRIMARIO:

```
Private Sub PutColor()
    fondo = New System.Drawing.Bitmap("BotonAmVec.jpg")
    BotonVec.Image = fondo
    fondo = New System.Drawing.Bitmap("BotonAmMov1.jpg")
    BotonMov1.Image = fondo
    fondo = New System.Drawing.Bitmap("BotonAmMov2.jpg")
    BotonMov2.Image = fondo
    fondo = New System.Drawing.Bitmap("BotonAmFza.jpg")
    BotonFza.Image = fondo
    fondo = New System.Drawing.Bitmap("BotonAmTrab.jpg")
    BotonTrab.Image = fondo
    fondo = New System.Drawing.Bitmap("BotonAmCol.jpg")
    BotonCol.Image = fondo
    fondo = New System.Drawing.Bitmap("BotonAmRot.jpg")
    BotonRot.Image = fondo
End Sub
```

ACTIVA UNIDAD DE *VECTORES*:

```
Private Sub BotonVec_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonVec.MouseMove
    PutColor()
    fondo = New System.Drawing.Bitmap("BotonRVec.jpg")
    BotonVec.Image = fondo
    fondo = New System.Drawing.Bitmap("FondoAVectf.jpg")
    Me.BackgroundImage = fondo
    QueUnid = 0
    tema = "TeoVecT1.RTF"
    Habilitar()
End Sub
```

ACTIVA UNIDAD DE *MOVIMIENTO EN UNA DIMENSIÓN*:

```
Private Sub BotonMov1_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonMov1.MouseMove
    PutColor()
    fondo = New System.Drawing.Bitmap("BotonRMov1.jpg")
    BotonMov1.Image = fondo
    fondo = New System.Drawing.Bitmap("FondoAMov1f.jpg")
    Me.BackgroundImage = fondo
    QueUnid = 1
    tema = "TeoMov1T1.RTF"
    Habilitar()
End Sub
```

ACTIVA UNIDAD DE *MOVIMIENTO EN DOS DIMENSIONES*:

```
Private Sub BotonMov2_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonMov2.MouseMove
    PutColor()
    fondo = New System.Drawing.Bitmap("BotonRMov2.jpg")
    BotonMov2.Image = fondo
    fondo = New System.Drawing.Bitmap("FondoAmov2f.jpg")
    Me.BackgroundImage = fondo
    QueUnid = 2
    tema = "TeoMov2T1.RTF"
    Habilitar()
```

```
    End Sub
```

ACTIVA UNIDAD DE *FUERZA Y MOVIMIENTO*:

```
    Private Sub BotonFza_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonFza.MouseMove
        PutColor()
        fondo = New System.Drawing.Bitmap("BotonRFza.jpg")
        BotonFza.Image = fondo
        fondo = New System.Drawing.Bitmap("FondoAFzaf.jpg")
        Me.BackgroundImage = fondo
        QueUnid = 3
        tema = "TeoFzaT1.RTF"
        Habilitar()
    End Sub
```

ACTIVA UNIDAD DE *TRABAJO Y ENERGÍA*:

```
    Private Sub BotonTrab_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonTrab.MouseMove
        PutColor()
        fondo = New System.Drawing.Bitmap("BotonRTrab.jpg")
        BotonTrab.Image = fondo
        fondo = New System.Drawing.Bitmap("FondoATrabf.jpg")
        Me.BackgroundImage = fondo
        QueUnid = 4
        tema = "TeoTraT1.RTF"
        Habilitar()
    End Sub
```

ACTIVA UNIDAD DE *COLISIONES*:

```
    Private Sub BotonCol_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonCol.MouseMove
        PutColor()
        fondo = New System.Drawing.Bitmap("BotonRCol.jpg")
        BotonCol.Image = fondo
        fondo = New System.Drawing.Bitmap("FondoAColf.jpg")
        Me.BackgroundImage = fondo
        QueUnid = 5
        tema = "TeoColT1.RTF"
        Habilitar()
    End Sub
```

ACTIVA UNIDAD DE *ROTACIÓN*:

```
    Private Sub BotonRot_move(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles BotonRot.MouseMove
        PutColor()
        fondo = New System.Drawing.Bitmap("BotonRRot.jpg")
        BotonRot.Image = fondo
        fondo = New System.Drawing.Bitmap("FondoARotf.jpg")
        Me.BackgroundImage = fondo
        QueUnid = 6
        tema = "TeoRotT1.RTF"
        Habilitar()
    End Sub
```

FINALIZA EL PROGRAMA:

```
    Private Sub BotonSalir_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BotonSalir.Click
        End
    End Sub
```

ABRE MENU DE SIMULACIONES (MENU.VB):

```
    Private Sub Practica_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Practica.Click
        Dim formas As New Menu()
```

```
    formas.Show()
  End Sub
```

ABRE FORMULARIO DE EVALUACIÓN (EVALUA.VB):

```
  Private Sub Evaluacion_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Evaluacion.Click
    Dim formas As New Evalua()
    formas.Show()
  End Sub
```

ABRE FORMULARIO DE INDICE CONCEPTUAL (INDICE.VB):

```
  Private Sub Conceptos_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Conceptos.Click
    Dim formas As New Indice()
    formas.Show()
  End Sub
```

1.3.2 FORMULARIO: TEORIA.VB

Despliega contenido teórico de la unidad activada.

CARGA DEL FORMULARIO

```
  Private Sub Teoria_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    RichTextBox1.LoadFile(main.tema)
    Select Case main.tema     SELECCIONA ARCHIVO RTF INICIAL
      Case Is = "TeoVecT1.RTF"
        Me.Text = "UNIDAD I: Vectores"
      Case Is = "TeoMov1T1.RTF"
        Me.Text = "UNIDAD II: Movimiento en una dimensión"
      Case Is = "TeoMov2T1.RTF"
        Me.Text = "UNIDAD III: Movimiento en dos dimensiones"
      Case Is = "TeoFzaT1.RTF"
        Me.Text = "UNIDAD IV: Fuerza y movimiento"
      Case Is = "TeoTraT1.RTF"
        Me.Text = "UNIDAD V: Trabajo y energía"
      Case Is = "TeoColT1.RTF"
        Me.Text = "UNIDAD VI: Colisiones"
      Case Is = "TeoRotT1.RTF"
        Me.Text = "UNIDAD VII: Rotación"
    End Select
  End Sub
```

EXTRAE ARCHIVO DE INICIO A MOSTRAR SEGÚN UNIDAD:

```
Private Sub Inicio_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Inicio.Click
    Select Case main.tema
      'INICIO - TEORIA DE VECTORES
       Case Is = "TeoVecT2.RTF"
        RichTextBox1.LoadFile("TeoVecT1.RTF")
        main.tema = "TeoVecT1.RTF"
        Inicio.Enabled = False
        Anterior.Enabled = False
      Case Is = "TeoVecT3.RTF"
        RichTextBox1.LoadFile("TeoVecT1.RTF")
        main.tema = "TeoVecT1.RTF"
        Inicio.Enabled = False
        Anterior.Enabled = False
      Case Is = "TeoVecT4.RTF"
        RichTextBox1.LoadFile("TeoVecT1.RTF")
        main.tema = "TeoVecT1.RTF"
        Inicio.Enabled = False
        Anterior.Enabled = False
       FUNCION DE SELECCIÓN CONTINÚA HASTA ENCONTRAR EL ARCHIVO CORRESPONDIENTE
```

FUNCIONES SIGUIENTES MUESTRAN IDÉNTICO COMPORTAMIENTO AL ANTERIOR:

```
  Private Sub Siguiente_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Siguiente.Click
      Select Case main.tema
        'SIGUIENTE - TEORIA DE VECTORES
         Case Is = "TeoVecT1.RTF" 'PRIMER TEMA
          RichTextBox1.LoadFile("TeoVecT2.RTF")
          main.tema = "TeoVecT2.RTF"
          Anterior.Enabled = True
          Inicio.Enabled = True
        Case Is = "TeoVecT2.RTF"
          RichTextBox1.LoadFile("TeoVecT3.RTF")
          main.tema = "TeoVecT3.RTF"

        Private Sub Anterior_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Anterior.Click
      Select Case main.tema
        'ANTERIOR - TEORIA DE VECTORES
      Case Is = "TeoVecT1.RTF" 'PRIMER TEMA
          Anterior.Enabled = False
          Inicio.Enabled = False
        Case Is = "TeoVecT2.RTF"
          RichTextBox1.LoadFile("TeoVecT1.RTF")
          main.tema = "TeoVecT1.RTF"
          Anterior.Enabled = False
          Inicio.Enabled = False

Private Sub Fin_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Fin.Click
      Select Case main.tema
        'FIN - TEORIA DE VECTORES
      Case Is = "TeoVecT1.RTF" 'PRIMER TEMA
          Inicio.Enabled = True
          Anterior.Enabled = True
          Siguiente.Enabled = False
          Fin.Enabled = False
          RichTextBox1.LoadFile("TeoVecT6.RTF")
          main.tema = "TeoVecT6.RTF"
        Case Is = "TeoVecT2.RTF"
          RichTextBox1.LoadFile("TeoVecT6.RTF")
          main.tema = "TeoVecT6.RTF"
          Siguiente.Enabled = False
          Fin.Enabled = False
```

1.3.3 FORMULARIO: MENU.VB

Muestra accesos a diferentes formularios de simulación.

DECLARACIONES:

```
  Dim forma1, forma2 As Form
```

ABRE FORMULARIO DE SIMULACIÓN ESPECIFICADA:

```
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    main.Limpiar()
    forma1.Show()
    Me.Hide()
  End Sub
```

ABRE FORMULARIO DE SIMULACIÓN ESPECIFICADA:

```
  Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    main.Limpiar()
    forma2.Show()
    Me.Hide()
  End Sub
```

CARGA EL FORMULARIO:

```
Private Sub Menu_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
```

SELECCIONA UNIDAD Y SIMULACIONES DISPONIBLES PARA ELLA ASIGNA ARCHIVO DE FORMULARIO A CADA BOTON

```
        Select Case main.QueUnid
        Case Is = 0
            Me.Height = 80
            forma1 = New Vectores()
            Button1.Visible = True
            Button1.Text = "Vectores"
            Button2.Visible = False

        Case Is = 1
            Me.Height = 124
            forma1 = New Acelera()
            forma2 = New CaidaLibre()
            Button1.Visible = True
            Button1.Text = "Movimiento Horizontal"
            Button2.Visible = True
            Button2.Text = "Movimiento Vertical"

        Case Is = 2
            Me.Height = 124
            forma1 = New TipoParab()
            forma2 = New MovCirc()
            Button1.Visible = True
            Button1.Text = "Movimiento Parabólico"
            Button2.Visible = True
            Button2.Text = "Movimiento Circular"

        Case Is = 3
            Me.Height = 80
            forma1 = New Newton()
            Button1.Visible = True
            Button1.Text = "Leyes de Newton"
            Button2.Visible = False

        Case Is = 4
            Me.Height = 80
            forma1 = New Resorte()
            Button1.Visible = True
            Button1.Text = "Conservación de la Energía"
            Button2.Visible = False

        Case Is = 5
            Me.Height = 80
            forma1 = New Choque1()
            Button1.Visible = True
            Button1.Text = "Colisión Elástica"
            Button2.Visible = False

        Case Is = 6
            Me.Height = 80
            forma1 = New Rotacion()
            Button1.Visible = True
            Button1.Text = "Rotación"
            Button2.Visible = False

    End Select
End Sub
```

1.3.4 FORMULARIO: ACELERA.VB

Simulación de Movimiento Horizontal

DECLARACIÓN DE VARIABLES:

```
Dim escala As Double  " escala de la regla
Dim Alto As Double    " punto de parada
Dim tiempo, Tt As Double  " Contador de tiempo.  Tiempo de parada
Dim Vo, Vt As Double    " velocidad inicial, Veloc. Parada
Dim A As Double   " aceleracion
Dim Tipo, Det, Xrow As Integer  " Valor de Parada, Ejecuto parada, Fila de Resultdos
Dim PosX, Post As Double " posicion actual de la particula
Dim Vf As Double " velocidad actual o final
```

CARGA EL FORMULARIO:

```
Private Sub Acelera_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Dim i As Integer

        ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
    main.LabDt(0, 0) = "Velocidad Inicial"
    main.LabDt(0, 1) = "Aceleración"
    main.LabDt(0, 2) = "Tiempo"
    main.LabDt(0, 3) = "Desplaza- miento"
    main.LabDt(0, 4) = "Velocidad Final"

        CONFIGURA LA ESCALA
    Xrow = 0
    escala = 40
    For i = 0 To 20 Step 1
      Label1.Text = Label1.Text + "'" + Space(5)
      Select Case i
        Case Is < 10
          Label2.Text = Label2.Text + Str(i) + Space(4)
        Case 10 To 99
          Label2.Text = Label2.Text + Str(i) + Space(3)
        Case 100 To 999
          Label2.Text = Label2.Text + Str(i) + Space(2)
        Case 1000 To 9999
          Label2.Text = Label2.Text + Str(i) + Space(1)
      End Select
    Next
    Label1.Text = Trim(Label1.Text)
    Label2.Text = Trim(Label2.Text)
    Timer2.Interval = 20
    Tipo = 1
    Esc1.Checked = True
  End Sub
```

INCIO DE LA SIMULACIÓN

```
  Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
        OBTIENE DATOS INICIALES
    A = spacel.Value
    Alto = SpStop.Value
    Det = 0
    Vo = SpSpeed.Value
    Part1.Top = 100
    Part1.Left = 12
    tiempo = 0
    Button1.Enabled = True

        CALCULA DATOS DE PARADA
    Select Case Tipo
      Case Is = 1   " en x distancia
        Vt = ((2 * Alto * A) + (Vo ^ 2)) ^ (0.5)
```

```
            If A > 0 Then
               Tt = (Vt - Vo) / A
            Else
               Tt = Alto / Vt
            End If
            Post = Alto
         Case Is = 2  " en x tiempo
            Vt = Vo + (A * Alto)
            Post = (Vo * Alto) + (0.5 * A * Alto * Alto)
            Tt = Alto
         Case Is = 3  " en x velocidad
            Tt = (Alto - Vo) / A
            Post = (Vo * Tt) + (0.5 * A * Tt * Tt)
            Vt = Alto
      End Select
      If A < 0 Then
         Dim myt As Double
         myt = -Vo / A
         If myt < Tt Then
            Vt = Vo + (A * myt)
            Post = (Vo * myt) + (0.5 * A * myt * myt)
            Tt = myt
         End If
      End If
      Button1.Tag = "D"
      Button1.Text = "Detener"

          INICIA EL MOVIMIENTO
      Timer2.Start()
   End Sub

EVENTO DE RELOJ

   Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer2.Tick
      If tiempo >= Tt And Det = 0 Then SI SE CUMPLEN VALORES DE PARADA
                     ENVIA DATOS A TABLA DE RESULATDOS
         main.TabDt(Xrow, 0) = Vo
         main.TabDt(Xrow, 1) = A
         main.TabDt(Xrow, 2) = Tt
         main.TabDt(Xrow, 3) = Post
         main.TabDt(Xrow, 4) = Vt
         If Xrow < 4 Then
            Xrow = Xrow + 1
         Else
            Xrow = 0
         End If
                     ENVIA DATOS A PANTALLA
         Part1.Left = 12 + (Post * escala)
         txtdesp.Text = Format(Post, "#####0.0000")
         txttiempo.Text = Format(Tt, "#####0.0000")
         txtvel.Text = Format(Vt, "#####0.0000")
         Det = 1
         Timer2.Stop()
         Button1.Tag = "S"
         Button1.Text = "Seguir"
      Else
                     CALCULA VALORES INSTANTANEOS
         tiempo = tiempo + (Timer2.Interval / 1000)
         PosX = (Vo * tiempo) + (0.5 * A * (tiempo * tiempo))
         Part1.Left = 12 + (PosX * escala)
         Vf = Vo + (A * tiempo)
                     ENVIA DATOS A PANTALLA
         txtdesp.Text = Format(PosX, "#####0.0000")
         txttiempo.Text = Format(tiempo, "#####0.0000")
```

```
        txtvel.Text = Format(Vf, "#####0.0000")
      End If
    End Sub

CAMBIA VALORES DE ESCALA A 10
    Private Sub Esc1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc1.CheckedChanged
      Dim i As Integer
      Label1.Text = " "
      Label2.Text = " "
      escala = 40
      For i = 0 To 20 Step 1
        Label1.Text = Label1.Text + "|" + Space(5)
        Select Case i
          Case Is < 10
            Label2.Text = Label2.Text + Str(i) + Space(4)
          Case 10 To 99
            Label2.Text = Label2.Text + Str(i) + Space(3)
          Case 100 To 999
            Label2.Text = Label2.Text + Str(i) + Space(2)
          Case 1000 To 9999
            Label2.Text = Label2.Text + Str(i) + Space(1)
        End Select
      Next
      Label1.Text = Trim(Label1.Text)
      Label2.Text = Trim(Label2.Text)
      Timer2.Interval = 20
    End Sub

CAMBIA VALORES DE ESCALA A 100
    Private Sub Esc2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc2.CheckedChanged
      Dim i As Integer
      Label1.Text = " "
      Label2.Text = " "
      escala = 40 / 10
      For i = 0 To 200 Step 10
        Label1.Text = Label1.Text + "|" + Space(5)
        Select Case i
          Case Is < 10
            Label2.Text = Label2.Text + Str(i) + Space(4)
          Case 10 To 99
            Label2.Text = Label2.Text + Str(i) + Space(3)
          Case 100 To 999
            Label2.Text = Label2.Text + Str(i) + Space(2)
          Case 1000 To 9999
            Label2.Text = Label2.Text + Str(i) + Space(1)
        End Select
      Next
      Label1.Text = Trim(Label1.Text)
      Label2.Text = Trim(Label2.Text)
      Timer2.Interval = 60
    End Sub

CAMBIA VALORES DE ESCALA A 1000
    Private Sub Esc3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc3.CheckedChanged
      Dim i As Integer
      Label1.Text = " "
      Label2.Text = " "
      escala = 40 / 100
      For i = 0 To 2000 Step 100
        Label1.Text = Label1.Text + "|" + Space(5)
        Select Case i
```

```
            Case Is < 100
               Label2.Text = Label2.Text + Str(i) + Space(4)
            Case 10 To 99
               Label2.Text = Label2.Text + Str(i) + Space(3)
            Case 100 To 999
               Label2.Text = Label2.Text + Str(i) + Space(2)
            Case 1000 To 9999
               Label2.Text = Label2.Text + Str(i) + Space(1)
         End Select
      Next
      Label1.Text = Trim(Label1.Text)
      Label2.Text = Trim(Label2.Text)
      Timer2.Interval = 120
   End Sub

DETIENE/REANUDA LA EJECUCIÓN
   Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
      If Button1.Tag = "D" Then
         Timer2.Stop()
         Button1.Tag = "S"
         Button1.Text = "Seguir"
      Else
         Timer2.Start()
         Button1.Tag = "D"
         Button1.Text = "Detener"
      End If
   End Sub

INDICA TIPO DE VALOR DE PARADA
   Private Sub Tipo1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo1.CheckedChanged
      Tipo = 1
   End Sub

INDICA TIPO DE VALOR DE PARADA
   Private Sub Tipo2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo2.CheckedChanged
      Tipo = 2
   End Sub

INDICA TIPO DE VALOR DE PARADA
   Private Sub Tipo3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo3.CheckedChanged
      Tipo = 3
   End Sub

CONFIGURA ENTRADA DE DATOS CUANDO LA ACELERACIÓN HA SIDO CAMBIADA
   Private Sub spacel_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spacel.ValueChanged
      If spacel.Value <> 0 Then
         Tipo3.Enabled = True
      Else
         Tipo1.Checked = True
         Tipo3.Checked = False
         Tipo3.Enabled = False
      End If
   End Sub

SETEA VALORES DE INICIO
   Private Sub BtReset_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtReset.Click
      Det = 0
      Part1.Top = 100
      Part1.Left = 12
      tiempo = 0
```

```
        txtdesp.Text = 0
        txttiempo.Text = 0
        txtvel.Text = 0
        Button1.Tag = "D"
        Button1.Text = "Detener"
        Button1.Enabled = False
    End Sub

MUESTRA AYUDA (AYUDA.VB)
    Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
        Dim forma As New Ayuda()
        forma.Show()
    End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
        Dim forma As New CLDatos()
        forma.Show()
    End Sub
```

1.3.5 FORMULARIO: CAIDALIBRE.VB

Simulación de Movimiento Vertical

```
DECLARACIÓN DE VARIABLES:
    Dim escala As Double  " escala de la regla
    Dim Alto As Double    " punto de parada
    Dim tiempo, Tt, Tf As Double  " tiempo instantáneo, parcial, final
    Dim Vo, Vt, V, Vf As Double  " veloc. inicial, parcial, instantánea, final
    Dim A As Double   " aceleracion
    Dim Tipo, Det As Integer  " tipo de valor en que se detiene
    Dim PosX, PosT, PosF As Double " pos. actual de la particula, parcial, final
    Dim XRow As Integer " linea de resultados actual

CARGA DEL FORMULARIO
    Private Sub CaidaLibre_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
            ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
        main.LabDt(0, 0) = "Velocidad Inicial"
        main.LabDt(0, 1) = "Gravedad"
        main.LabDt(0, 2) = "Tiempo"
        main.LabDt(0, 3) = "Altura Máxima"
        main.LabDt(0, 4) = "Velocidad Final"
            DATOS DE INICIO
        escala = 35
        Timer2.Interval = 20
        Esc1.Checked = True
        Tipo = 1
        XRow = 0
    End Sub

INICIA SIMULACIÓN
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
            CAPTURA VALORES INICIALES
        A = spacel.Value
        Alto = SpStop.Value
        Vo = SpSpeed.Value
        Part1.Top = 400
        tiempo = 0
        Button1.Enabled = True
        Det = 0
```

```
        CALCULA VALORES DE PARADA
    Select Case Tipo
      Case Is = 1
        Vt = ((2 * Alto * A) + (Vo ^ 2)) ^ (0.5)
        Tt = (Vt - Vo) / A
        PosT = Alto
      Case Is = 2
        Vt = Vo + (A * Alto)
        PosT = (Vo * Alto) + (0.5 * A * Alto * Alto)
        Tt = Alto
      Case Is = 3
        Tt = (Alto - Vo) / A
        PosT = (Vo * Tt) + (0.5 * A * Tt * Tt)
        Vt = Alto
    End Select
    Vf = Vo * (-1)
    PosF = 0
    Tf = (-Vo / A) * 2
        INCIAL MOVIMIENTO
    Timer2.Start()
  End Sub

EVENTO DE RELOJ
  Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer2.Tick
    tiempo = tiempo + (Timer2.Interval / 1000)
    If tiempo >= Tt And Det = 0 Then  SI SE CUMPLEN VALORES DE PARADA
                ENVIA DATOS A PANTALLA
      Part1.Top = 400 - (PosT * escala)
      txtdesp.Text = Format(PosT, "#####0.0000")
      txttiempo.Text = Format(Tt, "#####0.0000")
      txtvel.Text = Format(Vt, "#####0.0000")
      Det = 1
      Timer2.Stop()
      Button1.Text = "Seguir"
      Button1.Tag = "S"
    Else
      If tiempo >= Tf Then  SI LLEGO A SU FIN
        ENVIA DATOS A TABLA DE RESULTADOS
        main.TabDt(XRow, 0) = Vo
        main.TabDt(XRow, 1) = A
        main.TabDt(XRow, 2) = Tf
        main.TabDt(XRow, 3) = (Vo*(-Vo/A)) + (0.5 * A * ((-Vo / A) ^ 2))
        main.TabDt(XRow, 4) = Vf
        If XRow < 4 Then
          XRow = XRow + 1
        Else
          XRow = 0
        End If
                    ENVIA DATOS A PANTALLA
        Part1.Top = 400 - (PosF * escala)
        txtdesp.Text = Format(PosF, "#####0.0000")
        txttiempo.Text = Format(Tf, "####0.0000")
        txtvel.Text = Format(Vf, "#####0.0000")
        Timer2.Stop()
      Else
                    CALCULA VALORES INSTANTANEOS
        PosX = (Vo * tiempo) + (0.5 * A * (tiempo * tiempo))
        Part1.Top = 400 - (PosX * escala)
        V = Vo + (A * tiempo)
                    ENVIA DATOS A PANTALLA
        txtdesp.Text = Format(PosX, "#####0.0000")
        txttiempo.Text = Format(tiempo, "#####0.0000")
        txtvel.Text = Format(V, "#####0.0000")
      End If
```

```
        End If
    End Sub

DETIENE/REANUDA LA EJECUCIÓN
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        If Button1.Tag = "D" Then
            Timer2.Stop()
            Button1.Tag = "S"
            Button1.Text = "Seguir"
        Else
            Timer2.Start()
            Button1.Tag = "D"
            Button1.Text = "Detener"
        End If
    End Sub

CONFIGURA VALORES DE ESCALA
    Private Sub Esc1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc1.CheckedChanged
        escala = 35
        Lb1.Text = "1 -"
        Lb2.Text = "2 -"
        Lb3.Text = "3 -"
        Lb4.Text = "4 -"
        Lb5.Text = "5 -"
        Lb6.Text = "6 -"
        Lb7.Text = "7 -"
        Lb8.Text = "8 -"
        Lb9.Text = "9 -"
        Lb10.Text = "10 -"
    End Sub

CONFIGURA VALORES DE ESCALA
    Private Sub Esc2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc2.CheckedChanged
        escala = 35 / 10
        Lb1.Text = "10 -"
        Lb2.Text = "20 -"
        Lb3.Text = "30 -"
        Lb4.Text = "40 -"
        Lb5.Text = "50 -"
        Lb6.Text = "60 -"
        Lb7.Text = "70 -"
        Lb8.Text = "80 -"
        Lb9.Text = "90 -"
        Lb10.Text = "100 -"
    End Sub

CONFIGURA VALORES DE ESCALA
    Private Sub Esc3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc3.CheckedChanged
        escala = 35 / 100
        Lb1.Text = "100 -"
        Lb2.Text = "200 -"
        Lb3.Text = "300 -"
        Lb4.Text = "400 -"
        Lb5.Text = "500 -"
        Lb6.Text = "600 -"
        Lb7.Text = "700 -"
        Lb8.Text = "800 -"
        Lb9.Text = "900 -"
        Lb10.Text = "1000 -"
    End Sub
```

```
INDICA TIPO DE VALOR DE PARADA
  Private Sub Tipo1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo1.CheckedChanged
    Tipo = 1
  End Sub

INDICA TIPO DE VALOR DE PARADA
  Private Sub Tipo2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo2.CheckedChanged
    Tipo = 2
  End Sub

INDICA TIPO DE VALOR DE PARADA
  Private Sub Tipo3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo3.CheckedChanged
    Tipo = 3
  End Sub

SETEA VALORES DE INICIO
  Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
    Vo = SpSpeed.Value
    Part1.Top = 400
    tiempo = 0
    Button1.Enabled = True
    Det = 0
    txtdesp.Text = 0
    txttiempo.Text = 0
    txtvel.Text = 0
  End Sub

MUESTRA AYUDA (AYUDA.VB)
  Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
    Dim forma As New Ayuda()
    forma.Show()
  End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
  Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
    Dim forma As New CLDatos()
    forma.Show()
  End Sub
```

1.3.6 FORMULARIO: CHOQUE1.VB

Simulación de Colisiones

```
DECLARACIÓN DE VARIABLES:
  Dim escala As Double  " escala de la regla
  Dim Alto As Double    " punto de parada
  Dim tiempo As Double
  Dim m1, m2 As Double  " valor de las masas
  Dim Vo1, Vo2, Vf1, Vf2 As Double    " velocidad inicial
  Dim A As Double   " aceleracion = 0
  Dim Choque As Bolean  " ocurrio choque
  Dim Tipo, Xrow As Integer  " 0:no se ha detenido
  Dim X1, X2, PosX1, PosX2, Xdist As Double " posicion actual de la particula
  Dim Vf As Double " velocidad actual o final

CARGA DEL FORMULARIO
  Private Sub Choque1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
      ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
    main.LabDt(0, 0) = "Masa1"
```

```
        main.LabDt(0, 1) = "Vo Masa1"
        main.LabDt(0, 2) = "Masa2"
        main.LabDt(0, 3) = "Vo Masa2"
        main.LabDt(0, 4) = "Vf Masa1"
        main.LabDt(0, 5) = "Vf Masa2"

        Dim i As Integer
        Xrow = 0
        escala = 40
        Timer2.Interval = 20
        Tipo = 1
    End Sub

INICIA SIMULACIÓN
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
            CAPTURA VALORES INICIALES
        Choque = False
        A = spacel.Value
        m1 = Spm1.Value
        m2 = spm2.Value
        Vo1 = SpSpeed.Value
        Vo2 = SpSpeed2.Value
            ENVIA DATOS A PANTALLA
        txtVo1.Text = Format(Vo1, "#####0.0000")
        txtVo2.Text = Format(Vo2, "#####0.0000")
        txtmo1.Text = Format(Vo1 * m1, "#####0.0000")
        txtmo2.Text = Format(Vo2 * m2, "#####0.0000")
        Part1.Left = 30
        part2.Left = 440
        X1 = 12
        X2 = part2.Left
        tiempo = 0
        Button1.Enabled = True
        Button1.Tag = "D"
        Button1.Text = "Detener"
            INCIAL MOVIMIENTO
        Timer2.Start()
    End Sub

EVENTO DE RELOJ
    Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer2.Tick
            CALCULA VALORES INSTANTANEOS
        tiempo = tiempo + (Timer2.Interval / 1000)
        PosX1 = (Vo1 * tiempo) + (0.5 * A * (tiempo * tiempo))
        PosX2 = (Vo2 * tiempo) + (0.5 * A * (tiempo * tiempo))
        Part1.Left = X1 + (PosX1 * escala)
        part2.Left = X2 + (PosX2 * escala)
        Vf = Vo1 + (A * tiempo)
        Xdist = part2.Left - Part1.Left
                    SI LAS MASAS SE ENCUENTRAN
        If Xdist <= 40 And Choque = False Then
                    ENVIA DATOS A TABLA DE RESULTADOS
            main.TabDt(Xrow, 0) = m1
            main.TabDt(Xrow, 1) = Vo1
            main.TabDt(Xrow, 2) = m2
            main.TabDt(Xrow, 3) = Vo2
                    CALCULA VALORES INSTANTANEOS
            Vf1=(((m1 - m2) / (m1 + m2)) * Vo1) + (((2 * m2) / (m1 + m2)) * Vo2)
            Vf2 (((2 * m1) / (m1 + m2)) * Vo1) + (((m2 - m1) / (m1 + m2)) * Vo2)
            Vo1 = Vf1
            Vo2 = Vf2
            txtVf1.Text = Format(Vo1, "#####0.0000")
            txtVf2.Text = Format(Vo2, "#####0.0000")
            X1 = Part1.Left
```

```
            X2 = part2.Left
            tiempo = 0
            Choque = True
                        ENVIA DATOS A PANTALLA
            txtvf1.Text = Format(Vo1, "#####0.0000")
            txtvf2.Text = Format(Vo2, "#####0.0000")
            txtmf1.Text = Format(Vo1 * m1, "#####0.0000")
            txtmf2.Text = Format(Vo2 * m2, "#####0.0000")
            main.TabDt(Xrow, 4) = Vf1
            main.TabDt(Xrow, 5) = Vf2
            If Xrow < 4 Then
               Xrow = Xrow + 1
            Else
               Xrow = 0
            End If
         End If
      End Sub

DETIENE/REANUDA LA EJECUCIÓN
      Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
         If Button1.Tag = "D" Then
            Timer2.Stop()
            Button1.Tag = "S"
            Button1.Text = "Seguir"
         Else
            Timer2.Start()
            Button1.Tag = "D"
            Button1.Text = "Detener"
         End If
      End Sub

SETEA VALORES DE INICIO
      Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
         Part1.Left = 30
         part2.Left = 440
         tiempo = 0
         txtVo1.Text = 0
         txtVo2.Text = 0
         txtmo1.Text = 0
         txtmo2.Text = 0
         txtmf1.Text = 0
         txtmf2.Text = 0
         txtvf1.Text = 0
         txtvf2.Text = 0
         Button1.Tag = "D"
         Button1.Text = "Detener"
         Button1.Enabled = False
         Timer2.Stop()
      End Sub

MUESTRA AYUDA (AYUDA.VB)
      Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
         Dim forma As New Ayuda()
         forma.Show()

      End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
      Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
         Dim forma As New CLDatos()
         forma.Show()
      End Sub
```

1.3.7 FORMULARIO: MOVCIRC.VB

Simulación de Movimiento Circular Unifirme

```
DECLARACIÓN DE VARIABLES:
  Dim Superficie As Graphics  " superficie de grafico
  Dim Rectángulo As Rectangle  " area del circulo
  Dim Lapiz As Pen    " tipo linea
  Dim Linea1 As Pen   " linea
  Dim Tiempo, Alto, Tt, T As Double  " tiempo, valor de parada
  Dim Centro, tipo, Det As Integer  " datos de circulo
  Dim Radio, Dv As Integer  " datos de circulo
  Dim Wo, W, Ang, Wt, Angt As Double  " velocidades angulares
  Dim V As Double  " rapidez
  Dim A As Double  " aceleracion
  Dim S As Double  " long de arco
  Dim Px, Py, Qx, Qy, Xrow As Integer  " coordenadas de linea
  Dim rango As Integer  " rango de escala

CARGA DEL FORMULARIO
  Private Sub MovCirc_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    tipo = 1
    Reset()
       ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
    main.LabDt(0, 0) = "Radio"
    main.LabDt(0, 1) = "Rapidez"
    main.LabDt(0, 2) = "Aceleración Centripeta"
    main.LabDt(0, 3) = "Período"
    main.LabDt(0, 4) = "Frecuencia"
  End Sub

INICIA SIMULACIÓN
  Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    Reset()
    Timer1.Start()
    T = (2 * 3.1416 * Radio) / V
    A = (V * V) / Radio
        ENVIA DATOS A TABLA DE RESULTADOS
    main.TabDt(Xrow, 0) = Radio
    main.TabDt(Xrow, 1) = V
    main.TabDt(Xrow, 2) = A
    main.TabDt(Xrow, 3) = T
    main.TabDt(Xrow, 4) = 1 / T
    If Xrow < 4 Then
      Xrow = Xrow + 1
    Else
      Xrow = 0
    End If
  End Sub

EVENTO DE RELOJ
  Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
    Tiempo = Tiempo + (Timer1.Interval / 1000)
    If Tiempo <= 0 Then
      Return
    End If
       PARÁMETROS DEL CIRCULO
    Superficie = Me.CreateGraphics()
    Lapiz = New Pen(Color.Red, 2)
    Linea1 = New Pen(Color.GreenYellow, 3)
    Linea1.EndCap = Drawing.Drawing2D.LineCap.RoundAnchor
    Superficie.Clear(Me.BackColor)
    Rectángulo = New Rectangle(Centro-Radio, Centro-Radio, Radio*2, Radio*2)
    Superficie.DrawEllipse(Lapiz, Rectángulo)
```

```
        CALCULA VALORES INSTANTÁNEOS Y ENVIA A PANTALLA
      Ang = (W * Tiempo)
      txttiempo.Text = Format(Tiempo, "#####0.0000")
      T = (2 * 3.1416 * Radio) / V
      A = (V * V) / Radio
      txtperiodo.Text = Format(T, "#####0.0000")
      txtfrecuencia.Text = Format(1 / T, "#####0.0000")
      txtvelang.Text = Format(A, "#####0.0000")
      txtlong.Text = Format(Radio * Ang, "#####0.0000")
      txtrevs.Text = Format(Ang / (3.1416 * 2), "#####0.0000")
      txtangulo.Text = Format(Ang * 180 / 3.1416, "#####0.0000")
      Px = Centro + (Radio * Math.Sin(Ang))
      Py = Centro + (Radio * Math.Cos(Ang))
        DIBUJA LINEA
      Superficie.DrawLine(Linea1, Centro, Centro, Px, Py)
    End Sub

DETIENE LA EJECUCIÓN
    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
      Timer1.Stop()
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub Tipo1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
      tipo = 1
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub Tipo2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
      tipo = 2
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub Tipo3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
      tipo = 3
    End Sub

SETEA VALORES DE INICIO
    Private Sub Reset()
      Tiempo = 0
      rango = 20
      Centro = 200
      V = spveloc.Value
      W = V / Radio
      Px = Centro
      Py = Centro - Radio
      Qx = 1
      Det = 0
      OBTIENE REFERENCIA
      txttiempo.Text = 0
      txtperiodo.Text = 0
      txtfrecuencia.Text = 0
      txtvelang.Text = 0
      txtlong.Text = 0
      txtrevs.Text = 0
      Dibuja()
    End Sub

DIBUJA CÍRCULO Y LÍNEA
    Private Sub Dibuja()
      Radio = spradio.Value '* rango

      Superficie = Me.CreateGraphics()
      Superficie.Clear(Me.BackColor)
```

```
        Lapiz = New Pen(Color.Red, 2)
        Linea1 = New Pen(Color.GreenYellow, 3)
        Linea1.EndCap = Drawing.Drawing2D.LineCap.RoundAnchor
        ESTABLECE LOS VALORES DEL RECTÁNGULO
        Rectángulo = New Rectangle(Centro - Radio, Centro - Radio, Radio * 2, Radio * 2)
        Superficie.DrawEllipse(Lapiz, Rectángulo)
        Superficie.DrawLine(Linea1, Centro, Centro, Centro, Centro + Radio)
    End Sub

MUESTRA AYUDA (AYUDA.VB)
    Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
        Dim forma As New Ayuda()
        forma.Show()
    End Sub

CAMBIA VALORES DE CIRCULO
    Private Sub spradio_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spradio.ValueChanged
        Dibuja()
    End Sub

DIBUJA CÍRCULO Y LÍNEA
    Private Sub MovCirc_Paint(ByVal sender As Object, ByVal e As System.Windows.Forms.PaintEventArgs) Handles
MyBase.Paint
        Dibuja()
    End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
        Dim forma As New CLDatos()
        forma.Show()
    End Sub
```

1.3.8 FORMULARIO: NEWTON.VB

Simulación de Fuerza y Movimiento

```
DECLARACIÓN DE VARIABLES:
    Dim Lapiz As Pen  " tipo de linea
    Dim Superficie As Graphics  " area de grafico
    Dim puntos(2) As Point  " matriz coordenadas de lineas
    Dim Xo, Yo, Xf, Yf, Xrow As Integer  " coordenadas de lineas
    Dim Xmov, tiempo, PosX, Escala, Vo, Vf, A As Double " valores
    " movimiento, tiempo. desplazamiento, escala, vel inicial, vel final, celeracion
    Dim m1, m2, g, N, Fr, Ur, F As Double
    " masa1. masa2, gravedad, fuerza friccion, coef friccion, tension

CARGA DEL FORMULARIO
    Private Sub Newton_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Reset()
            ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
        main.LabDt(0, 0) = "Masa 1"
        main.LabDt(0, 1) = "Masa 2"
        main.LabDt(0, 2) = "Coef. Rozamiento"
        main.LabDt(0, 3) = "Aceleración"
        main.LabDt(0, 4) = "Tensión"
        main.LabDt(0, 5) = "Fuerza de Fricción"
    End Sub

SETEA VALORES DE INICIO
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles BtnReset.Click
        spmasa2.Value = 1
```

```
        Reset()
    End Sub

EVENTO DE RELOJ
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        If A <= 0 Or Part1.Left >= 484 Then
            Timer1.Stop()
            Return
        End If
            CALCULA VALORES INSTANTANEOS
        tiempo = tiempo + (Timer1.Interval / 1000)
        PosX = (Vo * tiempo) + (0.5 * A * (tiempo * tiempo))
        Xmov = Part1.Left
        Part1.Left = 56 + (PosX * Escala)
        Part2.Top = 250 + (PosX * Escala)
        Vf = Vo + (A * tiempo)
            ENVIA DATOS A PANTALLA
        txtTension.Text = Format(F, "#####0.0000")
        txtFriccion.Text = Format(Fr, "#####0.0000")
        txtveloc.Text = Format(Vf, "#####0.0000")
        txtEnergiaC.Text = Format((0.5) * (m1) * (Vf * Vf), "#####0.0000")
            DIBUJA CUERDA
        Superficie.Clear(Me.BackColor)
        Xo = Xo + (Part1.Left - Xmov)
        Yf = Yf + (Part1.Left - Xmov)
        Superficie.DrawLine(Lapiz, Xo, Yo, 520, 160)
        Superficie.DrawCurve(Lapiz, puntos, 0.6)
        Superficie.DrawLine(Lapiz, 550, 190, Xf, Yf)
        If Xo >= 520 Then
            Timer1.Stop()
        End If
    End Sub

DETIENE LA EJECUCIÓN
    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
        Timer1.Stop()
    End Sub

EJECUTA FUNCIÓN EVAL()
    Private Sub spmasa1_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spmasa1.ValueChanged
        Eval()
    End Sub

EVALUA DATOS
    Private Sub Eval()
            CALCULA VALORES INSTANTANEOS
        m1 = spmasa1.Value
        m2 = spmasa2.Value
        Ur = spcoef.Value
        Fr = Ur * m1 * g
        A = ((m2 * g) - Fr) / (m1 + m2)
        F = (m1 * A) + Fr
        If A > 0 Then  SI EXISTE ACELERACIÓN
                    ENVIA DATOS A TABLA DE RESULTADOS
            main.TabDt(Xrow, 0) = m1
            main.TabDt(Xrow, 1) = m2
            main.TabDt(Xrow, 2) = Ur
            main.TabDt(Xrow, 3) = A
            main.TabDt(Xrow, 4) = F
            main.TabDt(Xrow, 5) = Fr
            If Xrow < 4 Then
                Xrow = Xrow + 1
            Else
```

```
            Xrow = 0
        End If
        txtacel.Text = Format(A, "#####0.0000")
                INCIA MOVIMIENTO
        Timer1.Start()
    Else
        txtacel.Text = Format(0, "#####0.0000")
    End If
  End Sub

EJECUTA FUNCIÓN EVAL()
  Private Sub spcoef_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spcoef.ValueChanged
    Eval()
  End Sub

EJECUTA FUNCIÓN EVAL()
  Private Sub spmasa2_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spmasa2.ValueChanged
    Eval()
  End Sub

SETEA VALORES DE INICIO
  Private Sub Reset()
    Xo = 80
    Yo = 160
    Xf = 550
    Yf = 250
    Part1.Left = 56
    Part2.Top = 250
    Escala = 40
    tiempo = 0
    Vo = 0
    Xmov = 0
    g = 9.8
    txtTension.Text = 0
    txtFriccion.Text = 0
    txtveloc.Text = 0
    txtwork.Text = 0
    txtpower.Text = 0
    Dibuja()
  End Sub

MUESTRA AYUDA (AYUDA.VB)
  Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
    Dim forma As New Ayuda()
  End Sub

DIBUJA LÍNEA
  Private Sub Newton_Paint(ByVal sender As Object, ByVal e As System.Windows.Forms.PaintEventArgs) Handles
MyBase.Paint
    Dibuja()
  End Sub

DIBUJA LÍNEA
  Private Sub Dibuja()
    puntos(0) = New Point(520, 160)
    puntos(1) = New Point(545, 165)
    puntos(2) = New Point(550, 190)
    Superficie = Me.CreateGraphics()
    Superficie.Clear(Me.BackColor)
    Lapiz = New Pen(Color.OrangeRed, 2)
    Superficie.DrawLine(Lapiz, Xo, Yo, 520, 160)
```

```
        Superficie.DrawCurve(Lapiz, puntos, 0.6)
        Superficie.DrawLine(Lapiz, 550, 190, Xf, Yf)
    End Sub

EJECUTA FUNCIÓN EVAL()
    Private Sub Button1_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        Eval()
    End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
        Dim forma As New CLDatos()
        forma.Show()
    End Sub
End Class
```

1.3.9 FORMULARIO: RESORTE.VB

Simulación de Trabajo y Energía

```
DECLARACIÓN DE VARIABLES:
    Dim m, V, Tiempo, k, Xo, X As Double
    " masa, velocidad, tiempo, constante rozamiento, pos. inicial, pos. instantanea
    Dim Ek, Ep, Em, Emx, w, Fr, Wr As Double
    " energia cinetica, energia potencial, energia mecanica, Em inicial
    " veloc. angular, fuerza friccion, trabajo
    Dim RgU, RgK, Xrow As Double  " tamaño graficas energia
    Dim Escala As Integer   " escala de movimiento

CARGA DEL FORMULARIO
    Private Sub Resorte_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Escala = 60
        main.LabDt(0, 0) = "Masa"
        main.LabDt(0, 1) = "Coef. Fza. Resorte"
        main.LabDt(0, 2) = "Fuerza Aplicada"
        main.LabDt(0, 3) = "Coef. Fricción"
        main.LabDt(0, 4) = "Energía Mecánica"
    End Sub

CALCULA ESTIRAMIENTO
    Private Sub spacel_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spK.ValueChanged
        block.Left = 330 + (SpDesp.Value / spK.Value * Escala)
        PicRes.Width = 277 + (SpDesp.Value / spK.Value * Escala)
        txtdespx.Text = SpDesp.Value / spK.Value
    End Sub

INICIA SIMULACIÓN
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
            CALCULA VALORES INICIALES
        m = SpMasa.Value
        k = spK.Value
        Xo = SpDesp.Value / k
        Fr = Spfriccion.Value
        Tiempo = 0
        Emx = (1 / 2) * k * (Xo * Xo)
        If Fr > 0 Then
            w = ((k / m) - ((Fr ^ 2) / (4 * (m ^ 2)))) ^ (0.5)
        Else
            w = (k / m) ^ (0.5)
            Em = Emx
        End If
        PU.Width = 350
```

```
        PK.Width = 350
        main.TabDt(Xrow, 0) = m
        main.TabDt(Xrow, 1) = k
        main.TabDt(Xrow, 2) = SpDesp.Value
        main.TabDt(Xrow, 3) = Fr
        main.TabDt(Xrow, 4) = Emx
        If Xrow < 4 Then
            Xrow = Xrow + 1
        Else
            Xrow = 0
        End If
        Timer1.Start()
    End Sub

EVENTO DE RELOJ
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        Tiempo = Tiempo + (Timer1.Interval / 1000)
            CALCULA VALORES INSTANTANEOS
        If Fr > 0 Then   SI POSEE FRICCION
            X = Xo * Math.Exp((-Fr * Tiempo) / (2 * m)) * Math.Cos(w * Tiempo)
            Em = (1 / 2) * k * (Xo ^ 2) * Math.Exp((-Fr * Tiempo) / m)
            PEm.Width = (Em / Emx) * 350
            Ep = 0.5 * k * X * X
            Ek = Em - Ep
        Else
            X = Xo * Math.Cos(w * Tiempo)
            Ep = (1 / 2) * k * (Xo ^ 2) * (Math.Cos(w * Tiempo)) ^ 2
            Ek = (1 / 2) * k * (Xo ^ 2) * (Math.Sin(w * Tiempo)) ^ 2
        End If
        Wr = (1 / 2) * k * (X ^ 2)
        V = ((2 * Ek) / m) ^ (1 / 2)
        block.Left = 330 + (X * Escala)
        PicRes.Width = 277 + (X * Escala)
            ENVIA DATOS A PANTALLA
        txttiempo.Text = Format(Tiempo, "#####0.0000")
        txtdesp.Text = Format(X, "#####0.0000")
        txtvel.Text = Format(V, "#####0.0000")
        txtEm.Text = Format(Em, "#####0.0000")
        txtEp.Text = Format(Ep, "#####0.0000")
        txtEk.Text = Format(Ek, "#####0.0000")
        txtW.Text = Format(w, "#####0.0000")
        RgU = (Ep / Emx)
        RgK = (Ek / Emx)
        PU.Width = RgU * 350
        PK.Width = RgK * 350
        If Em < 0.0001 Then
            Timer1.Stop()
        End If

    End Sub

DETIENE/REANUDA LA EJECUCIÓN
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
        Timer1.Stop()
    End Sub

CALCULA ESTIRAMIENTO
    Private Sub SpDesp_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
SpDesp.ValueChanged
        block.Left = 330 + (SpDesp.Value / spK.Value * Escala)
        PicRes.Width = 277 + (SpDesp.Value / spK.Value * Escala)
        txtdespx.Text = SpDesp.Value / spK.Value
    End Sub
```

```
SETEA VALORES DE INICIO
  Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
    Reset()
  End Sub

MUESTRA AYUDA (AYUDA VB)
  Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
    Dim forma As New Ayuda()
    forma.Show()
  End Sub

SETEA VALORES DE INICIO
  Private Sub Reset()
    block.Left = 330
    PicRes.Width = 277
    PEm.Width = 350
    PK.Width = 350
    PU.Width = 350
      ENVIA DATOS A PANTALLA
    txttiempo.Text = Format(0, "#####0.0000")
    txtdesp.Text = Format(0, "#####0.0000")
    txtvel.Text = Format(0, "#####0.0000")
    txtEm.Text = Format(0, "#####0.0000")
    txtEp.Text = Format(0, "#####0.0000")
    txtEk.Text = Format(0, "#####0.0000")
  End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
  Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
    Dim forma As New CLDatos()
    forma.Show()
  End Sub
```

1.3.10 FORMULARIO: ROTACION.VB

Simulación de Rotación

```
DECLARACIÓN DE VARIABLES:
  Dim Superficie As Graphics  " area de grafico
  Dim Rectángulo As Rectangle  " area de circulo
  Dim Lapiz As Pen  " tipo linea
  Dim Linea1 As Pen  " linea
  Dim Tiempo, Alto, Tt As Double  " tiempo, valor parada
  Dim Centro, tipo, Det As Integer " centro circulo, tipo parada
  Dim Radio, Dv As Integer  " radio
  Dim Wo, W, Ang As Double  " vel angular, posicion angulo
  Dim Wt, Angt As Double  " Vel Angular
  Dim A As Double " aceleracion angular
  Dim S, T As Double  " long de arco, periodo
  Dim Px, Py, Qx, Qy As Integer  " coordenaras linea
  Dim rango, xrow As Integer  " rango de escala

CARGA DEL FORMULARIO
  Private Sub Rotacion_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    tipo = 1
    Reset()
      ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
    main.LabDt(0, 0) = "Radio"
    main.LabDt(0, 1) = "V.Angular Inicial "
    main.LabDt(0, 2) = "Acel. Angular"
    main.LabDt(0, 3) = "Tiempo"
    main.LabDt(0, 4) = "Pos. Angular"
```

```
        main.LabDt(0, 5) = "V.Angular Inst."
        main.LabDt(0, 6) = "Período"
        main.LabDt(0, 7) = "Frecuencia"
        main.LabDt(0, 8) = "Long. Arco"
        main.LabDt(0, 9) = "Revoluciones"
    End Sub

SETEA VALORES DE INICIO
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        Reset()
    End Sub

INICIA SIMULACIÓN
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
        Reset()
        txtvelang.Text = Format(W, "#####0.0000")
        txtacel.Text = Format(A, "#####0.0000")
            CALCULA VALORES DE PARADA
        Select Case tipo
            Case Is = 1   " ang
                Wt = ((2 * A * Alto) + Wo ^ 2) ^ 0.5
                Tt = (2 * Alto) / (Wt + Wo)
                Angt = Alto
            Case Is = 2  " seg
                Wt = Wo + (A * Alto)
                Angt = ((Wt + Wo) / 2) * Alto
                Tt = Alto
            Case Is = 3  " rad/seg
                Tt = (Alto - Wo) / A
                Angt = ((Alto + Wo) / 2) * Tt
                Wt = Alto
        End Select
        Timer1.Start()
    End Sub

EVENTO DE RELOJ
    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer1.Tick
        Tiempo = Tiempo + (Timer1.Interval / 1000)
        If Tiempo <= 0 Then
            Return
        End If
            CALCULA VALORES GRÁFICOS
        Superficie = Me.CreateGraphics()
        Lapiz = New Pen(Color.Red, 2)
        Linea1 = New Pen(Color.GreenYellow, 3)
        Linea1.EndCap = Drawing.Drawing2D.LineCap.RoundAnchor
        Superficie.Clear(Me.BackColor)
        Rectángulo=New Rectangle(Centro-Radio, Centro-Radio, Radio*2, Radio*2)
        Superficie.DrawEllipse(Lapiz, Rectángulo)
        If Tiempo >= Tt And Det = 0 Then  SI SE CUMPLEN VALORES DE PARADA
            ENVIA DATOS A PANTALLA
                    txttiempo.Text = Format(Tt, "#####0.0000")
            T = 2 * 3.1416 / Wt
            txtperiodo.Text = Format(T, "#####0.0000")
            txtfrecuencia.Text = Format(1 / T, "#####0.0000")
            txtangulo.Text = Format(Angt * 180 / 3.1416, "#####0.0000")
            txtvelang.Text = Format(Wt, "#####0.0000")
            txtlong.Text = Format(Radio * Angt, "#####0.0000")
            txtrevs.Text = Format(Angt / (3.1416 * 2), "#####0.0000")
            Px = Centro + (Radio * Math.Sin(Angt))
            Py = Centro + (Radio * Math.Cos(Angt))
            Det = 1
                    ENVIA DATOS A TABLA DE RESULTADOS
            main.TabDt(xrow, 0) = Radio
```

```
            main.TabDt(xrow, 1) = Wo
            main.TabDt(xrow, 2) = A
            main.TabDt(xrow, 3) = Tt
            main.TabDt(xrow, 4) = Angt
            main.TabDt(xrow, 0) = Wt
            main.TabDt(xrow, 1) = T
            main.TabDt(xrow, 2) = 1 / T
            main.TabDt(xrow, 3) = Radio * Angt
            main.TabDt(xrow, 4) = Angt / (3.1416 * 2)
            If Xrow < 4 Then
               Xrow = Xrow + 1
            Else
               Xrow = 0
            End If
            Timer1.Stop()
         Else
            If A = 0 Then   POSEE ACELERACIÓN?
               Ang = (W * Tiempo)
            Else
               W = Wo + (A * Tiempo)
               Ang = ((W + Wo) / 2) * Tiempo
            End If
                     ENVIA DATOS A PANTALLA
            txttiempo.Text = Format(Tiempo, "#####0.0000")
            T = 2 * 3.1416 / W
            txtperiodo.Text = Format(T, "#####0.0000")
            txtfrecuencia.Text = Format(1 / T, "#####0.0000")
            txtangulo.Text = Format(Ang * 180 / 3.1416, "#####0.0000")
            txtvelang.Text = Format(W, "#####0.0000")
            txtlong.Text = Format(Radio * Ang, "#####0.0000")
            txtrevs.Text = Format(Ang / (3.1416 * 2), "#####0.0000")
            Px = Centro + (Radio * Math.Sin(Ang))
            Py = Centro + (Radio * Math.Cos(Ang))
                     ENVIA DATOS A TABLA DE RESULTADOS
            main.TabDt(xrow, 0) = Radio
            main.TabDt(xrow, 1) = Wo
            main.TabDt(xrow, 2) = A
            main.TabDt(xrow, 3) = T
            main.TabDt(xrow, 4) = Ang
            main.TabDt(xrow, 0) = W
            main.TabDt(xrow, 1) = T
            main.TabDt(xrow, 2) = 1 / T
            main.TabDt(xrow, 3) = Radio * Ang
            main.TabDt(xrow, 4) = Ang / (3.1416 * 2)
            If xrow < 4 Then
               xrow = xrow + 1
            Else
               xrow = 0
            End If
         End If
         Superficie.DrawLine(Linea1, Centro, Centro, Px, Py)
      End Sub

DETIENE LA EJECUCIÓN
      Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
         Timer1.Stop()
      End Sub

INDICA TIPO DE VALOR DE PARADA
      Private Sub Tipo1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo1.CheckedChanged
         tipo = 1
      End Sub
```

```
INDICA TIPO DE VALOR DE PARADA
   Private Sub Tipo2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo2.CheckedChanged
      tipo = 2
   End Sub
INDICA TIPO DE VALOR DE PARADA
   Private Sub Tipo3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo3.CheckedChanged
      tipo = 3
   End Sub

CAMBIO EN LA ACELERACIÓN
   Private Sub spacel_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spacel.ValueChanged
      If spacel.Value > 0 Then
         Tipo3.Enabled = True
      Else
         Tipo3.Enabled = False
      End If
   End Sub

SETEA VALORES DE INICIO
   Private Sub Reset()
      Tiempo = 0
      rango = 20
      Centro = 200
      Wo = spveloc.Value
      W = Wo
      A = spacel.Value
      Px = Centro
      Py = Centro - Radio
      Qx = 1
      Det = 0
      If tipo = 1 Then
         Alto = SpStop.Value * 3.1416 / 180
      Else
         Alto = SpStop.Value
      End If
          ENVIA DATOS A PANTALLA
      txttiempo.Text = 0
      txtperiodo.Text = 0
      txtfrecuencia.Text = 0
      txtangulo.Text = 0
      txtvelang.Text = 0
      txtacel.Text = 0
      txtlong.Text = 0
      txtrevs.Text = 0
      Dibuja()
   End Sub

DIBUJA GRÁFICOS
   Private Sub Dibuja()
      Radio = spradio.Value
      Superficie = Me.CreateGraphics()
      Superficie.Clear(Me.BackColor)
      Lapiz = New Pen(Color.Red, 2)
      Linea1 = New Pen(Color.GreenYellow, 3)
      Linea1.EndCap = Drawing.Drawing2D.LineCap.RoundAnchor
      ESTABLECE LOS VALORES DEL RECTÁNGULO
      Rectángulo = New Rectangle(Centro - Radio, Centro - Radio, Radio * 2, Radio * 2)
      Superficie.DrawEllipse(Lapiz, Rectángulo)
      Superficie.DrawLine(Linea1, Centro, Centro, Centro, Centro + Radio)
   End Sub
```

```
MUESTRA AYUDA (AYUDA.VB)
   Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
      Dim forma As New Ayuda()
      forma.Show()
   End Sub

DIBUJA GRAFICOS SEGÚN CAMBIO DE RADIO
   Private Sub spradio_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
spradio.ValueChanged
      Dibuja()
   End Sub

DIBUJA GRÁFICO
   Private Sub MovCirc_Paint(ByVal sender As Object, ByVal e As System.Windows.Forms.PaintEventArgs) Handles
MyBase.Paint
      Dibuja()
   End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
   Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
      Dim forma As New CLDatos()
      forma.Show()
   End Sub
```

1.3.11 FORMULARIO: TIPOPARAB.VB

Simulación de Movimiento Parabólico

```
DECLARACIÓN DE VARIABLES:
   Dim escala As Double  " escala de la regla
   Dim Alto As Double    " punto de parada
   Dim tiempo, Tt, Tf, Tl As Double " Tiempos
   Dim Vo, Vt, Vf, Vy, Vx As Double  " velocidades
   Dim A As Double   " aceleracion
   Dim Tipo, Det As Integer " tipo de valor de parada
   Dim PosX,Postx,Posfx,Poslx As Double " posicion de la particula en x
   Dim PosY,Posty,Posfy,Posly As Double " posicion de la particula en y
   Dim X1,X2,Y1,Y2 As Integer  "coord. de linea de direccion de disparo
   Dim Linea1 As Pen  " tipo de linea
   Dim Superficie As Graphics  " area de grafico
   Dim XRow As Integer  " fila de tabla de registros actual

CARGA DEL FORMULARIO
   Private Sub TipoParab_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
         ENVIA ENCABEZADOS DATOS A TABLA DE RESULATDOS
      main.LabDt(0, 0) = "Angulo de Disparo"
      main.LabDt(0, 1) = "Velocidad en X"
      main.LabDt(0, 2) = "Velocidad Inicial Y"
      main.LabDt(0, 3) = "Gravedad"
      main.LabDt(0, 4) = "Tiempo"
      main.LabDt(0, 5) = "Alcance"
      main.LabDt(0, 6) = "Altura"
      main.LabDt(0, 7) = "Velocidad en Y"
         CALCULA VALORES DE GRAFICO
      Superficie = Me.CreateGraphics()
      Linea1 = New Pen(Color.Red, 3)
      Linea1.StartCap = LineCap.Round
      Linea1.EndCap = LineCap.ArrowAnchor
      Linea1.DashStyle = DashStyle.Dot
         DATOS DE INICIO
      escala = 35
```

```
        Timer2.Interval = 20
        Tipo = 4
        Esc1.Checked = True
        XRow = 0
    End Sub
INICIA SIMULACIÓN
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
        Iniciar()
        Button1.Enabled = True
            CALCULA VALORES DE PARADA
        Select Case Tipo
            Case Is = 1
                Vt = ((2 * Alto * A) + (Vo ^ 2)) ^ (0.5)
                Tt = (Vt - Vo) / A
                Posty = Alto
            Case Is = 2
                Vt = Vo + (A * Alto)
                Posty = (Vo * Alto) + (0.5 * A * Alto * Alto)
                Tt = Alto
            Case Is = 3
                Tt = (Alto - Vo) / A
                Posty = (Vo * Tt) + (0.5 * A * Tt * Tt)
                Vt = Alto
            Case Is = 4
                Tt = Alto / Vx
                Posty = (Vo * Tt) + (0.5 * A * Tt * Tt)
                Vt = Vo + (A * Tt)
        End Select
            CAPTURA VALORES INICIALES
        Postx = Vx * Tt
        txtvelx.Text = Format(Vx, "#####0.0000")
        Vf = Vo * (-1)
        Posfy = 0
        Tf = (-Vo / A) * 2
        Posfx = Vx * Tf
        Superficie.Clear(Me.BackColor)
        Timer2.Start()
    End Sub

EVENTO DE RELOJ
    Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Timer2.Tick
        tiempo = tiempo + (Timer2.Interval / 1000)
        If tiempo >= Tt And Det = 0 Then  SI SE CUMPLEN VALORES DE PARADA
            Part1.Top = 436 - (Posty * escala)
            Part1.Left = 68 + (Postx * escala)
                    ENVIA DATOS A PANTALLA
            txtdespy.Text = Format(Posty, "#####0.0000")
            txtdespx.Text = Format(Postx, "#####0.0000")
            txttiempo.Text = Format(Tt, "#####0.0000")
            txtvely.Text = Format(Vt, "#####0.0000")
            Det = 1
            Timer2.Stop()
            Button1.Tag = "S"
            Button1.Text = "Seguir"
                    ENVIA DATOS A TABLA DE RESULTADOS
            main.TabDt(XRow, 0) = SpAngulo.Value
            main.TabDt(XRow, 1) = Vx
            main.TabDt(XRow, 2) = Vo
            main.TabDt(XRow, 3) = A
            main.TabDt(XRow, 4) = tiempo
            main.TabDt(XRow, 5) = Postx
            main.TabDt(XRow, 6) = Posty
            main.TabDt(XRow, 7) = Vt
            If XRow < 4 Then
```

```
            XRow = XRow + 1
          Else
            XRow = 0
          End If
        Else
          If tiempo >= Tf Then  SI SE CUMPLEN VALORES DE FIN
            Part1.Top = 436 - (Posfy * escala)
            Part1.Left = 68 + (Posfx * escala)
           ENVIA DATOS A PANTALLA
            txtdespx.Text = Format(Posfx, "#####0.0000")
            txtdespy.Text = Format(Posfy, "#####0.0000")
            txttiempo.Text = Format(Tf, "#####0.0000")
            txtvely.Text = Format(Vf, "#####0.0000")
            Timer2.Stop()
            Tl = (0 - Vo) / A
            Posly = (Vo * Tl) + (0.5 * A * Tl * Tl)
            Poslx = Vx * Tl
                    ENVIA DATOS A TABLA DE RESULTADOS
            main.TabDt(XRow, 0) = SpAngulo.Value
            main.TabDt(XRow, 1) = Vx
            main.TabDt(XRow, 2) = Vo
            main.TabDt(XRow, 3) = A
            main.TabDt(XRow, 4) = tiempo
            main.TabDt(XRow, 5) = Posfx
            main.TabDt(XRow, 6) = Posfy
            main.TabDt(XRow, 7) = Vf
            If XRow < 4 Then
              XRow = XRow + 1
            Else
              XRow = 0
            End If
          Else
                    CALCULA VALORES INSTANTANEOS
            PosY = (Vo * tiempo) + (0.5 * A * (tiempo * tiempo))
            PosX = Vx * tiempo
            Part1.Top = 436 - (PosY * escala)
            Part1.Left = 68 + (PosX * escala)
            Vy = Vo + (A * tiempo)
                    ENVIA DATOS A PANTALLA
            txtdespx.Text = Format(PosX, "#####0.0000")
            txtdespy.Text = Format(PosY, "#####0.0000")
            txttiempo.Text = Format(tiempo, "#####0.0000")
            txtvely.Text = Format(Vy, "#####0.0000")
          End If
        End If
    End Sub


CONFIGURA VALORES DE ESCALA
    Private Sub Esc1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc1.CheckedChanged
        escala = 40
        Lb1.Text = "1 -"
        Lb2.Text = "2 -"
        Lb3.Text = "3 -"
        Lb4.Text = "4 -"
        Lb5.Text = "5 -"
        Lb6.Text = "6 -"
        Lb7.Text = "7 -"
        Lb8.Text = "8 -"
        Lb9.Text = "9 -"
        Lb10.Text = "10 -"
        Lbx1.Text = "1"
        Lbx2.Text = "2"
```

```
        Lbx3.Text = "3"
        Lbx4.Text = "4"
        Lbx5.Text = "5"
        Lbx6.Text = "6"
        Lbx7.Text = "7"
        Lbx8.Text = "8"
        Lbx9.Text = "9"
        Lbx10.Text = "10"
        Lbx11.Text = "11"
        Lbx12.Text = "12"
        Lbx13.Text = "13"
        Lbx14.Text = "14"
        Lbx15.Text = "15"
        Lbx16.Text = "16"
        Lbx17.Text = "17"
        Lbx18.Text = "18"
        Lbx19.Text = "19"
        Timer2.Interval = 20
    End Sub

CONFIGURA VALORES DE ESCALA
    Private Sub Esc2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc2.CheckedChanged
        escala = 40 / 10
        Lb1.Text = "10 -"
        Lb2.Text = "20 -"
        Lb3.Text = "30 -"
        Lb4.Text = "40 -"
        Lb5.Text = "50 -"
        Lb6.Text = "60 -"
        Lb7.Text = "70 -"
        Lb8.Text = "80 -"
        Lb9.Text = "90 -"
        Lb10.Text = "100 -"
        Lbx1.Text = "10"
        Lbx2.Text = "20"
        Lbx3.Text = "30"
        Lbx4.Text = "40"
        Lbx5.Text = "50"
        Lbx6.Text = "60"
        Lbx7.Text = "70"
        Lbx8.Text = "80"
        Lbx9.Text = "90"
        Lbx10.Text = "100"
        Lbx11.Text = "110"
        Lbx12.Text = "120"
        Lbx13.Text = "130"
        Lbx14.Text = "140"
        Lbx15.Text = "150"
        Lbx16.Text = "160"
        Lbx17.Text = "170"
        Lbx18.Text = "180"
        Lbx19.Text = "190"
    End Sub

CONFIGURA VALORES DE ESCALA
    Private Sub Esc3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Esc3.CheckedChanged
        escala = 40 / 100
        Lb1.Text = "100 -"
        Lb2.Text = "200 -"
        Lb3.Text = "300 -"
        Lb4.Text = "400 -"
        Lb5.Text = "500 -"
```

```
        Lb6.Text = "600 -"
        Lb7.Text = "700 -"
        Lb8.Text = "800 -"
        Lb9.Text = "900 -"
        Lb10.Text = "1000 -"
        Lbx1.Text = "100"
        Lbx2.Text = "200"
        Lbx3.Text = "300"
        Lbx4.Text = "400"
        Lbx5.Text = "500"
        Lbx6.Text = "600"
        Lbx7.Text = "700"
        Lbx8.Text = "800"
        Lbx9.Text = "900"
        Lbx10.Text = "1000"
        Lbx11.Text = "1100"
        Lbx12.Text = "1200"
        Lbx13.Text = "1300"
        Lbx14.Text = "1400"
        Lbx15.Text = "1500"
        Lbx16.Text = "1600"
        Lbx17.Text = "1700"
        Lbx18.Text = "1800"
        Lbx19.Text = "1900"
    End Sub

DETIENE/REANUDA LA EJECUCIÓN
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        If Button1.Tag = "D" Then
            Timer2.Stop()
            Button1.Tag = "S"
            Button1.Text = "Seguir"
        Else
            Timer2.Start()
            Button1.Tag = "D"
            Button1.Text = "Pausa"
        End If
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub Tipo1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo1.CheckedChanged
        Tipo = 1
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub Tipo2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo2.CheckedChanged
        Tipo = 2
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub Tipo3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Tipo3.CheckedChanged
        Tipo = 3
    End Sub

INDICA TIPO DE VALOR DE PARADA
    Private Sub RadioButton1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles RadioButton1.CheckedChanged
        Tipo = 4
    End Sub
```

```
INICIA SIMULACIÓN
  Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
    Iniciar()
  End Sub

INICIA SIMULACIÓN
  Private Sub Iniciar()
    A = spacel.Value
    Alto = SpStop.Value
    Part1.Top = 436
    Part1.Left = 68
    tiempo = 0
    Det = 0
    Button1.Enabled = False
    Button1.Text = "Pausa"
    Button1.Tag = "D"
    txtdespy.Text = 0
    txtdespx.Text = 0
    txttiempo.Text = 0
    txtvely.Text = 0
    txtvelx.Text = 0
  End Sub

MUESTRA AYUDA (AYUDA.VB)
  Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
    Dim forma As New Ayuda()
    forma.Show()
  End Sub

MUESTRA TABLA DE DATOS (CLDATOS.VB)
  Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
    Dim forma As New CLDatos()
    forma.Show()
  End Sub

DIBUJA GRÁFICO
  Private Sub SpAngulo_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
SpAngulo.ValueChanged
    Dibuja()
  End Sub

DIBUJA LÍNEA DE DIRECCIÓN DE DISPARO
  Private Sub Dibuja()
    Dim Linea1 As Pen
    Dim Superficie As Graphics
    Dim X1 As Integer
    Dim Y1 As Integer
    Superficie = Me.CreateGraphics()
    Superficie.Clear(Me.BackColor)
    Linea1 = New Pen(Color.Red, 5)
    Linea1.StartCap = LineCap.Round
    Linea1.EndCap = LineCap.ArrowAnchor
    Vo = SpSpeed.Value*Math.Sin(SpAngulo.Value*3.1416/180)
    Vx = SpSpeed.Value*Math.Cos(SpAngulo.Value*3.1416/180)
    X1 = (SpSpeed.Value/4)*Math.Cos(SpAngulo.Value*3.1416/180)*escala
    Y1 = (SpSpeed.Value/4)*Math.Sin(SpAngulo.Value*3.1416/180)*escala
    Superficie.DrawLine(Linea1, 84, 452, 84 + X1, 452 - Y1)
    txtvox.Text = Format(Vx, "#####0.0000")
    txtvoy.Text = Format(Vo, "#####0.0000")
  End Sub
```

DIBUJA GRÁFICO
```
  Private Sub SpSpeed_ValueChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
SpSpeed.ValueChanged
      Dibuja()
  End Sub
```

1.3.12 FORMULARIO: VECTORES.VB

Simulación Vectores

DECLARACIÓN DE VARIABLES:
```
  Dim XRow As Integer  " fila de tabla de registros actual
  Dim Linea1, Linea2, LineaR, LineaA As Pen  " tipos de lineas
  Dim Superficie As Graphics
  Dim X11, Y11, X12, Y12 As Integer  " coordenadas de vector A
  Dim X21, Y21, X22, Y22 As Integer  " coordenadas de vector B
  Dim XR1, YR1, XR2, YR2 As Integer  " coordenadas de vector resultante suma/resta
```

DIBUJA GRÁFICO
```
  Private Sub Form1_Paint(ByVal sender As Object, ByVal e As System.Windows.Forms.PaintEventArgs) Handles
MyBase.Paint
      Dibujar()
  End Sub
```

CARGA DEL FORMULARIO
```
  Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
      Superficie = Me.CreateGraphics()
  End Sub
```

DIBUJA VECTOR A
```
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
      Linea1 = New Pen(Color.Red, 3)
      Linea1.StartCap = LineCap.Round
      Linea1.EndCap = LineCap.ArrowAnchor
      X11 = 265 + (txt_x11.Value * 10)
      Y11 = 265 - (txt_y11.Value * 10)
      X12 = 265
      Y12 = 265
      Superficie.DrawLine(Linea1, X12, Y12, X11, Y11)
  End Sub
```

DIBUJA RESULTANTE VECTOR A + B
```
  Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
      LineaR = New Pen(Color.Green, 3)
      LineaR.StartCap = LineCap.Round
      LineaR.EndCap = LineCap.ArrowAnchor
      LineaA = New Pen(Color.Blue, 3)
      LineaA.StartCap = LineCap.Round
      LineaA.EndCap = LineCap.ArrowAnchor
      LineaA.DashStyle = DashStyle.Dot
      XR1 = 265 + (txt_x11.Value + txt_x21.Value) * 10
      YR1 = 265 - (txt_y11.Value + txt_y21.Value) * 10
      XR2 = 265
      YR2 = 265
      txt_XR.Text = (txt_x11.Value + txt_x21.Value)
      txt_YR.Text = (txt_y11.Value + txt_y21.Value)
      Superficie.DrawLine(LineaR, XR2, YR2, XR1, YR1)
      Superficie.DrawLine(LineaA, 265 + (txt_x11.Value * 10), 265 - (txt_y11.Value * 10), XR1, YR1)
  End Sub
```

DIBUJA VECTOR B

```
Private Sub Button2_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    Linea2 = New Pen(Color.Blue, 3)
    Linea2.StartCap = LineCap.Round
    Linea2.EndCap = LineCap.ArrowAnchor
    X21 = 265 + (txt_x21.Value * 10)
    Y21 = 265 - (txt_y21.Value * 10)
    X22 = 265
    Y22 = 265
    Superficie.DrawLine(Linea2, X22, Y22, X21, Y21)
End Sub
```

DIBUJA RESULTANTE VECTOR A - B

```
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
    LineaR = New Pen(Color.Orange, 3)
    LineaR.StartCap = LineCap.Round
    LineaR.EndCap = LineCap.ArrowAnchor
    LineaA = New Pen(Color.Blue, 3)
    LineaA.StartCap = LineCap.Round
    LineaA.EndCap = LineCap.ArrowAnchor
    LineaA.DashStyle = DashStyle.Dot
    XR1 = 265 + (txt_x11.Value + (txt_x21.Value * -1)) * 10
    YR1 = 265 - (txt_y11.Value + (txt_y21.Value * -1)) * 10
    XR2 = 265
    YR2 = 265
    txt_XR1.Text = (txt_x11.Value + (txt_x21.Value * -1))
    txt_YR1.Text = (txt_y11.Value + (txt_y21.Value * -1))

    Superficie.DrawLine(LineaR, XR2, YR2, XR1, YR1)
    Superficie.DrawLine(LineaA, 265 + (txt_x11.Value * 10), 265 - (txt_y11.Value * 10), XR1, YR1)
End Sub
```

DIBUJA PLANO

```
Private Sub Dibujar()
    Dim Brocha As HatchBrush
    Dim Superficie As Graphics
    Dim Rectángulo As Rectangle
    Dim Lapiz As Pen
    'Obtiene referencia.
    Superficie = Me.CreateGraphics()
    Superficie.Clear(Me.BackColor)
    Lapiz = New Pen(Color.Gray, 4)
    'Establece los valores del rectángulo
    Rectángulo = New Rectangle(10, 10, 510, 510)
    Brocha = New HatchBrush( _
    Drawing.Drawing2D.HatchStyle.Max, Color.LightGray, Color.White)
    Superficie.FillRectangle(Brocha, Rectángulo)
    Superficie.DrawLine(Lapiz, 10, 265, 520, 265)
    Superficie.DrawLine(Lapiz, 265, 10, 265, 520)
End Sub
```

LIMPIA PANTALLA

```
Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click
    Dibujar()
End Sub
```

MUESTRA AYUDA (AYUDA.VB)

```
Private Sub BtnAyuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
BtnAyuda.Click
    Dim forma As New Ayuda()
    main.contenido = "HelpVectores.RTF"
    forma.Show()
End Sub
```

```
MUESTRA TABLA DE DATOS (CLDATOS.VB)
  Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button6.Click
    LineaR = New Pen(Color.Purple, 3)
    LineaR.StartCap = LineCap.Round
    LineaR.EndCap = LineCap.ArrowAnchor
    XR1 = 265 + (txt_x11.Value * spEsc.Value) * 10
    YR1 = 265 - (txt_y11.Value * spEsc.Value) * 10
    XR2 = 265
    YR2 = 265
    txtEscx.Text = (txt_x11.Value * spEsc.Value)
    txtEscy.Text = (txt_y11.Value * spEsc.Value)
    Superficie.DrawLine(LineaR, XR2, YR2, XR1, YR1)
    LineaR = New Pen(Color.Red, 3)
    LineaR.StartCap = LineCap.Round
    LineaR.EndCap = LineCap.ArrowAnchor
    LineaR.DashStyle = DashStyle.Dot
    XR1 = 265 + (txt_x11.Value) * 10
    YR1 = 265 - (txt_y11.Value) * 10
    XR2 = 265
    YR2 = 265
    Superficie.DrawLine(LineaR, XR2, YR2, XR1, YR1)
  End Sub
```

1.3.13 FORMULARIO: AYUDA.VB

Formulario de Ayuda del Sistema

```
CARGA DEL FORMULARIO
  Private Sub Ayuda_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
      CARGA ARCHIVO RTF CON CONTENIDO DE AYUDA
    RichTextBox1.LoadFile(main.contenido)
  End Sub
```

1.3.14 FORMULARIO: CLDATOS.VB

Formulario Resultados Tabulados de Simulacion

```
DECLARACIÓN DE VARIABLES:
  Dim i As Integer   NUMERO DE FILA ACTUAL

CARGA DEL FORMULARIO
  Private Sub CLDatos_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
   ROTULA ENCABEZADOS DE COLUMNAS
    Label1.Text = main.LabDt(0, 0)
    Label2.Text = main.LabDt(0, 1)
    Label3.Text = main.LabDt(0, 2)
    Label4.Text = main.LabDt(0, 3)
    Label5.Text = main.LabDt(0, 4)
    Label6.Text = main.LabDt(0, 5)
    Label7.Text = main.LabDt(0, 6)
    Label8.Text = main.LabDt(0, 7)
    Label9.Text = main.LabDt(0, 8)
    Label10.Text = main.LabDt(0, 9)
      ENVIA VALORES A CADA CUADRO DE TEXTO SEGÚN POSICIÓN EN LA MATRIZ
    txt00.Text = Format(main.TabDt(0, 0), "#####0.0000")
    txt01.Text = Format(main.TabDt(0, 1), "#####0.0000")
    txt02.Text = Format(main.TabDt(0, 2), "#####0.0000")
    txt03.Text = Format(main.TabDt(0, 3), "#####0.0000")
    txt04.Text = Format(main.TabDt(0, 4), "#####0.0000")
    txt05.Text = Format(main.TabDt(0, 5), "#####0.0000")
    txt06.Text = Format(main.TabDt(0, 6), "#####0.0000")
    txt07.Text = Format(main.TabDt(0, 7), "#####0.0000")
```

```
    txt08.Text = Format(main.TabDt(0, 8), "#####0.0000")
    txt09.Text = Format(main.TabDt(0, 9), "#####0.0000")
    txt10.Text = Format(main.TabDt(1, 0), "#####0.0000")
    txt11.Text = Format(main.TabDt(1, 1), "#####0.0000")
    txt12.Text = Format(main.TabDt(1, 2), "#####0.0000")
    txt13.Text = Format(main.TabDt(1, 3), "#####0.0000")
    txt14.Text = Format(main.TabDt(1, 4), "#####0.0000")
    txt15.Text = Format(main.TabDt(1, 5), "#####0.0000")
    txt16.Text = Format(main.TabDt(1, 6), "#####0.0000")
    txt17.Text = Format(main.TabDt(1, 7), "#####0.0000")
    txt18.Text = Format(main.TabDt(1, 8), "#####0.0000")
    txt19.Text = Format(main.TabDt(1, 9), "#####0.0000")
    txt20.Text = Format(main.TabDt(2, 0), "#####0.0000")
    txt21.Text = Format(main.TabDt(2, 1), "#####0.0000")
    txt22.Text = Format(main.TabDt(2, 2), "#####0.0000")
    txt23.Text = Format(main.TabDt(2, 3), "#####0.0000")
    txt24.Text = Format(main.TabDt(2, 4), "#####0.0000")
    txt25.Text = Format(main.TabDt(2, 5), "#####0.0000")
    txt26.Text = Format(main.TabDt(2, 6), "#####0.0000")
    txt27.Text = Format(main.TabDt(2, 7), "#####0.0000")
    txt28.Text = Format(main.TabDt(2, 8), "#####0.0000")
    txt29.Text = Format(main.TabDt(2, 9), "#####0.0000")
    txt30.Text = Format(main.TabDt(3, 0), "#####0.0000")
    txt31.Text = Format(main.TabDt(3, 1), "#####0.0000")
    txt32.Text = Format(main.TabDt(3, 2), "#####0.0000")
    txt33.Text = Format(main.TabDt(3, 3), "#####0.0000")
    txt34.Text = Format(main.TabDt(3, 4), "#####0.0000")
    txt35.Text = Format(main.TabDt(3, 5), "#####0.0000")
    txt36.Text = Format(main.TabDt(3, 6), "#####0.0000")
    txt37.Text = Format(main.TabDt(3, 7), "#####0.0000")
    txt38.Text = Format(main.TabDt(3, 8), "#####0.0000")
    txt39.Text = Format(main.TabDt(3, 9), "#####0.0000")
    txt40.Text = Format(main.TabDt(4, 0), "#####0.0000")
    txt41.Text = Format(main.TabDt(4, 1), "#####0.0000")
    txt42.Text = Format(main.TabDt(4, 2), "#####0.0000")
    txt43.Text = Format(main.TabDt(4, 3), "#####0.0000")
    txt44.Text = Format(main.TabDt(4, 4), "#####0.0000")
    txt45.Text = Format(main.TabDt(4, 5), "#####0.0000")
    txt46.Text = Format(main.TabDt(4, 6), "#####0.0000")
    txt47.Text = Format(main.TabDt(4, 7), "#####0.0000")
    txt48.Text = Format(main.TabDt(4, 8), "#####0.0000")
    txt49.Text = Format(main.TabDt(4, 9), "#####0.0000")
  End Sub
```

ESCONDE DEL FORMULARIO

```
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Me.Hide()
  End Sub
```

1.3.15 FORMULARIO: EVALUA.VB

Formulario de Evaluación

DECLARACIÓN DE VARIABLES:

```
  Dim Nw As Boolean  " indica si la pregunta seleccionada no ha sido propuesta
  Dim x, Mynum, NMax As Integer
  " contador, Numero de pregunta actual, Número máximo de preguntas
  Public Shared Qst(0, 19) As Integer " matriz de almacenamiento de preguntas
  Public Shared acum, Nqst As Double  " acumula preg acertadas, contador de posicion en matriz
  Dim Bmh, Bmd As BindingManagerBase " hace referencia a tablas de preguntas y respuestas
```

CARGA DEL FORMULARIO

```
  Private Sub Evalua_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
```

```
        LLAMA FUNCION QUE CARGA DATO
    Me.Cargar()
        FILTRA DATOS POR UNIDAD ACTIVADA
    Me.BindingContext(DsMain1, "F_Unid").Position = main.QueUnid
    Nqst = 0
        Bmh = Me.BindingContext(DsMain1, "F_Unid.F_UnidF_TsHd")
    Bmd = Me.BindingContext(DsMain1, "F_Unid.F_UnidF_TsHd.F_TsHdF_TsDt")
    NMax = 5
    ToolTip1.SetToolTip(ayuda, "Ayuda")
  End Sub

CREA CONJUNTO DE DATOS TEMPORAL
  Public Sub Cargar()
        CREA CONJUNTO DE DATOS TEMPORAL PARA ALOJAR REGISTROS
            Dim objDataSetTemp As FisicaInt.DSMain
    objDataSetTemp = New FisicaInt.DSMain()
    Try
            RELLENA CONJUNTO DE DATOS TEMPORAL
      Me.FillDataSet(objDataSetTemp)
    Catch eFillDataSet As System.Exception
      Throw eFillDataSet
    End Try
    Try
      DsMain1.Clear()
      DsMain1.Merge(objDataSetTemp)
    Catch eLoadMerge As System.Exception
      Throw eLoadMerge
    End Try
  End Sub

ABRE CONEXIÓN A DATOS Y RELLENA CONJUNTO DE DATOS
  Public Sub FillDataSet(ByVal dataSet As FisicaInt.DSMain)
        DESACTIVA LA COMPROBACIÓN DE REGISTRO DE DATOS
   dataSet.EnforceConstraints = False
    Try
            ABRE CONEXIÓN
      Me.OleDbConnection1.Open()
            INTENTA RELLENAR CONJUNTO DE DATOS
      Me.OleDbDataAdapter1.Fill(dataSet)
      Me.OleDbDataAdapter2.Fill(dataSet)
      Me.OleDbDataAdapter3.Fill(dataSet)
    Catch fillException As System.Exception
      Throw fillException
    Finally
      dataSet.EnforceConstraints = True
      Me.OleDbConnection1.Close()
    End Try
  End Sub

INICIA EL EVALUADOR
  Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    Panel1.Visible = False
   MÁXIMO PREGUNTAS DISPONIBLES
        TextBox4.Text = Bmh.Count()
    For x = 0 To 19
      Qst(0, x) = -1
    Next
   INICIALIZA VALORES
        Nw = True
    Nqst = 0
    acum = 0
    Button3.Enabled = True
    Button2.Enabled = False
```

```
        INICIALIZA MOTOR ALEATORIO
    Randomize()
    Me.NextQst()
  End Sub

PASA A SIGUIENTE PREGUNTA
  Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
        SI ACERTA PREGUNTA AUMENTA CONTADOR
        If NAnswer.Value = Me.BindingContext(DsMain1, "F_Unid.F_UnidF_TsHd.NAnsw").Current Then
      acum = acum + 1
    End If
    Me.NextQst()
  End Sub

GENERA A SIGUIENTE PREGUNTA
  Private Sub NextQst()
        If Nqst = NMax Then  SI LLEGO A ULTIMA PREGUNTA
              MUESTRA FORMULARIO NOTA (EVALUANOTA.VB)
              Dim forma As New EvaluaNota()
      forma.Show()
      Me.Hide()
    Else
              MOTOR DE BÚSQUEDA DE SIGUIENTE PREGUNTA
      Nw = False
      While Nw = False
        Mynum = CInt(Int(((Bmh.Count() * Rnd()) + 0)))
        For x = 0 To 20
          If x = 20 Then
            Nw = True
            Exit For
          End If
          If Qst(0, x) = Mynum Then
            Exit For
          End If
        Next
      End While
              ENVIA DATOS A PANTALLA
      Qst(0, Nqst) = Mynum
      Bmh.Position = Mynum
      Nqst = Nqst + 1
      NAnswer.Maximum = Bmd.Count
      NAnswer.Value = 1
      NAnswer.Focus()
      TextBox1.Text = Format(Nqst, "##")
      If Nqst = NMax Then  SI FUE LA ULTIMA PREGUNTA
        Button3.Text = "Finalizar"
      End If
    End If
  End Sub

CIERRA DEL FORMULARIO
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Me.Hide()
  End Sub

ESTABLECE NÚMERO MAX DE PREGUNTAS A 10
  Private Sub Numq2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Numq2.CheckedChanged
    NMax = 10
  End Sub

ESTABLECE NÚMERO MAX DE PREGUNTAS A 20
  Private Sub Numq3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Numq3.CheckedChanged
```

```
    NMax = 20
  End Sub

ESTABLECE NÚMERO MAX DE PREGUNTAS A 5
  Private Sub Numq1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Numq1.CheckedChanged
    NMax = 5
  End Sub

MUESTRA AYUDA (AYUDA.VB)
  Private Sub ayuda_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ayuda.Click
    Dim forma As New Ayuda()
    main.contenido = "HelpEvaluador.RTF"
    forma.Show()
  End Sub
```

1.3.16 FORMULARIO: EVALUANOTA.VB

Formulario que muestra resultado de evaluación

```
CARGA DEL FORMULARIO
  Private Sub EvaluaNota_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        CALCULA NOTA Y LA MUESTRA EN PANTALLA
    Label1.Text = "Se respondieron correctamente " + Str(Evalua.acum) + " de " + Str(Evalua.Nqst)
    Label3.Text = Str(Evalua.acum / Evalua.Nqst * 10)
  End Sub

ESCONDE EL FORMULARIO
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Me.Hide()
  End Sub
```

1.3.17 FORMULARIO: INDICE.VB

Formulario que muestra listado de conceptos

```
CARGA DEL FORMULARIO
  Private Sub Indice_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        LLAMA FUNCION QUE CARGA DATOS
    Me.Cargar()
    Me.BindingContext(DsMain1, "F_Unid").Position = main.QueUnid
  End Sub

CREA CONJUNTO DE DATOS TEMPORAL
  Public Sub Cargar()
        CREA CONJUNTO DE DATOS TEMPORAL PARA ALOJAR REGISTROS
    Dim objDataSetTemp As FisicaInt.DSMain
    objDataSetTemp = New FisicaInt.DSMain()
    Try
        RELLENA CONJUNTO DE DATOS
      Me.FillDataSet(objDataSetTemp)
    Catch eFillDataSet As System.Exception
      Throw eFillDataSet
    End Try
    Try
      DsMain1.Clear()
      DsMain1.Merge(objDataSetTemp)
    Catch eLoadMerge As System.Exception
      Throw eLoadMerge
    End Try
  End Sub
```

```
ABRE CONEXIÓN A DATOS Y RELLENA CONJUNTO DE DATOS
  Public Sub FillDataSet(ByVal dataSet As FisicaInt.DSMain)
        DESACTIVA LA COMPROBACIÓN DE REGISTRO DE DATOS
    dataSet.EnforceConstraints = False
    Try
              ABRE LA CONEXIÓN DE DATOS
      Me.OleDbConnection1.Open()
              RELLENA EL CONJUNTO DE DATOS
      Me.OleDbDataAdapter1.Fill(dataSet)
      Me.OleDbDataAdapter2.Fill(dataSet)
    Catch fillException As System.Exception
      Throw fillException
    Finally
      dataSet.EnforceConstraints = True
      Me.OleDbConnection1.Close()
    End Try
  End Sub

ESCONDE FORMULARIO
  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Me.Hide()
  End Sub
```

1.3.18 FORMULARIO: CONTENIDOHELP.VB

Lista de Temas Contenidos en Ayuda General

```
ABRE LISTA AYUDA PARA SIMULACIONES
  Private Sub Label5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Simu.Click
    Dim forma As New SimulacionesHelp()
    forma.Show()
  End Sub

ABRE FORMULARIO DE AYUDA PARA EVALUADOR
  Private Sub Eval_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Eval.Click
    Dim forma As New Ayuda()
    main.contenido = "HelpEvaluador.RTF"
    forma.Show()
  End Sub

ABRE FORMULARIO DE AYUDA PARA TEORIA
  Private Sub ConTeo_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ConTeo.Click
    Dim forma As New Ayuda()
    main.contenido = "HelpDesTeo.RTF"
    forma.Show()
  End Sub

ABRE FORMULARIO DE AYUDA GENERAL DE APLICACION
  Private Sub QueEs_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles QueEs.Click
    Dim forma As New Ayuda()
    main.contenido = "HelpQueEs.RTF"
    forma.Show()
  End Sub

ABRE FORMULARIO DE AYUDA PARA MENU PRINCIPAL
  Private Sub Entorno_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Entorno.Click
    Dim forma As New Ayuda()
    main.contenido = "HelpEntorno.RTF"
    forma.Show()
  End Sub
```

1.3.19 FORMULARIO: SIMULACIONHELP.VB

Lista de Temas Contenidos en Ayuda de Simulaciones

```
MUESTRA AYUDA (AYUDA.VB)
  Private Sub Label1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Label1.Click
      Dim forma As New Ayuda()
      forma.Show()
   End Sub
```

1.4 CÓDIGOS DE ERROR

1.4.1 DESBORDAMIENTO NUMÉRICO

MENSAJE: *"ARITHMETIC OPERATION RESULTED IN AN OVERFLOW"*

Indica error en una operación aritmética. Posiblemente se especificó algún parámetro de inicio con un valor no operable dentro de la fórmula ejecutada.

Ej. Inicializar con CERO un parámetro que será divisor en determinada fórmula.

POSIBLE SOLUCIÓN:

Reestablecer valores de entrada. Asegurarse de introducir valores válidos para cada parámetro.

1.4.2 VALOR DE PROPIEDAD NO VÁLIDO

MENSAJE: *"EXCEPCION NO CONTROLADA DEL TIPO SYSTEM.ARGUMENT.EXCEPTION"*

Indica error en valor de propiedad asignada a un determinado objeto.

Ej. Enviar valor 5 a control numérico que posee rango entre 1 y 4.

Este error es posible si se ejecuta los módulo de EVALUADOR o INDICE sin tener datos suficientes en la base de datos.

POSIBLE SOLUCIÓN:

Asegurese de poseer base de datos correcta en carpeta donde está instalada la aplicación.

1.4.3 ARCHIVO NO ENCONTRADO

MENSAJE: *"EXCEPCION NO CONTROLADA DEL TIPO SYSTEM.IO.FILENOTFOUNDEXCEPTION"*

Indica la falta de archivo requerido por la aplicación.

POSIBLE SOLUCIÓN:

Asegurese de poseer archivos completos de la aplicación o ejecute archivo de instalación para repararla.

1.4.4 ARGUMENTO NO VÁLIDO

MENSAJE: *"EXCEPCION NO CONTROLADA DEL TIPO SYSTEM.ARGUMENTEXCEPTION"*

Indica argumento no válido para determinada propiedad.

POSIBLE SOLUCIÓN:

Asegurese de poseer archivos completos de la aplicación o ejecute archivo de instalación para repararla.

1.4.5 FALLO CONEXIÓN A DATOS

MENSAJE: ***"EXCEPCION NO CONTROLADA DEL TIPO SYSTEM.DATA.OLEDB.OLEDBEXCEPTION"***

Indica fallo en la conexión a base de datos FisicaDb.

POSIBLE SOLUCIÓN:

Asegurese de poseer base de datos en carpeta donde está instaladoa la aplicación o ejecute archivo de instalación para volver a instalarla.