DESARROLLO DE UN SOFTWARE EDUCATIVO QUE SIRVA DE APOYO PARA EL APRENDIZAJE DE LA ASIGNATURA COMPILADORES DEL PROGRAMA DE INGENIERÍA DE SISTEMAS DE LA UNIVERSIDAD DE SAN BUENAVENTURA CARTAGENA

JOHNNY CABARCAS MACHACON

UNIVERSIDAD DE SAN BUENAVENTURA
FACULTAD DE INGENIERIA, ARQUITECTURA, ARTES Y DISEÑO
PROGRAMA DE INGENIERIA DE SISTEMAS
CARTAGENA DE INDIAS D.T.C
2013

DESARROLLO DE UN SOFTWARE EDUCATIVO QUE SIRVA DE APOYO PARA EL APRENDIZAJE DE LA ASIGNATURA COMPILADORES DEL PROGRAMA DE INGENIERÍA DE SISTEMAS DE LA UNIVERSIDAD DE SAN BUENAVENTURA CARTAGENA

JOHNNY CABARCAS MACHACON

Monografía como requisito para optar al
Titulo de Ingeniero de Sistemas

Director
CARLOS CORTES
Ingeniero De Sistemas

UNIVERSIDAD DE SAN BUENAVENTURA
FACULTAD DE INGENIERIA, ARQUITECTURA, ARTES Y DISEÑO
PROGRAMA DE INGENIERIA DE SISTEMAS
CARTAGENA DE INDIAS D.T.C
2013

Nota de Aceptación

_____

_____

_____

_____
Presidente del Jurado

_____
Jurado

_____
Jurado

Cartagena de Índias

ii

Dedico este trabajo especialmente a Dios porque sin su amor y misericordia, esto no hubiera sido posible, a mi Esposa por sus palabras de aliento y apoyo incondicional, a mis padres y hermanos especialmente a mi hermana mayor que siempre estuvo pendiente al desarrollo de este trabajo, gracias a la Universidad de San Buenaventura por brindarme los conocimientos para llegar a obtener el titulo de Ingeniero de Sistemas.

Johnny Cabarcas Machacon

AGRADECIMIENTOS

El autor expresa su agradecimiento a:

A Dios todo poderoso que siempre nos dio sabiduría en la realización de este proyecto.

Carlos Cortes. Ingeniero de Sistemas y Director de la investigación, quien nos impulsó y asesoró para realizar este proyecto.

La Universidad que nos dio la oportunidad de formarnos profesionalmente y ayudo a crecer y a madurar con todas las situaciones vividas en el transcurso de nuestra carrera.

Grupo de docentes del programa de Ingeniería de Sistemas de la facultad de Ingenierías de la Universidad San Buenaventura Cartagena, y a todas esas personas que de una u otra forma estuvieron siempre dispuestos a ayudarnos en la realización de este Proyecto.

La familia porque siempre nos brindo su apoyo y confianza en la realización de este proyecto y durante la carrera de Ingeniería de Sistemas.

RECOMENDACIONES

Durante la investigación surgieron las siguientes recomendaciones:

- Preparar al personal de docentes y estudiantes en el manejo del software para tener un mejor entendimiento relacionado con este.

- Estimular a los estudiantes para que realicen proyectos que beneficien a la institución y contribuyan con el desarrollo de esta asignatura con el fin de mejorar la calidad teniendo en cuenta la ingeniería del software.

- El estudiante tenga bases de esta asignatura, para mayor comprensión de la misma.

CONTENIDO

	Pág.
RESUMEN	9
INTRODUCCIÓN	10
1. PROBLEMA DE INVESTIGACIÓN	16
1.1 PLANTEAMIENTO DEL PROBLEMA	16
1.2 FORMULACIÓN DEL PROBLEMA	17
1.3 JUSTIFICACIÓN	17
1.4 OBJETIVOS	18
1.4.1 Objetivo General	18
1.4.2 Objetivos Específicos	19
2. MARCO REFERENCIAL	20
2.1 INVESTIGACIONES PREVIAS	21
2.2 BASES TEORICAS	22
2.2.1 Compilador.	20
2.2.2 Clasificación De Las Gramáticas	23
2.2.3 Gramáticas Libres Del Contexto.	24
2.2.4 Autómatas Finitos.	26
2.2.5 Ingeniería Del software	26
2.2.6 Software libre	27
2.2.7 Lenguaje Modelado (UML).	28
2.2.8 Lenguaje de Programación	30
2.2.9 Flash Macromedia.	31
2.2.10 PHP	30
2.2.11 MySQL.	31
	32

Pág.

2.3 DEFINICIÓN DE TERMINOS BÁSICOS 33
3. DISEÑO METODOLÓGICO 35
3.1 ENFOQUE, MÉTODO Y TIPO DE INVESTIGACIÓN 35
3.2 DELIMITACIÓN 35
3.2.1 Conceptual. 35
3.2.2 Geográfica 35
3.3 FUENTES DE INFORMACIÓN 35
3.3.1 Primarias. 35
3.3.2 Secundarias. 36
3.4 VARIABLES 36
3.4.1 Identificación de variables. 36
3.4.2 Operacionalización de las Variables 36
3.5 RECURSOS TECNOLÓGICOS 37
3.5.1 Paradigma de Programación. 37
3.5.2 Metodología en la Construcción del Software. 37
3.5.3 Herramientas De Diseño. 38
3.5.4 Herramientas De Desarrollo 39
4. DESARROLLO DEL PROYECTO 40
4.1 COMUNICACIÓN CON EL CLIENTE. 41
4.1.1 Presentación de la Información. 42
4.1.2 Determinación de Requerimientos Funcionales. 42
4.1.3 Determinación de Requerimientos No Funcionales. 42
4.2 PLANIFICACIÓN 42
4.3 ANÁLISIS DE RIESGO 43
4.4 FASE DE CONSTRUCCIÓN Y ADAPTACIÓN DE LA INGENIERÍA 45

4.4.1 Casos de Usos 45
4.4.2 Diseño de Interfaz 68
4.5 FASE DE EVALUACIÓN DEL USUARIO 82
Pág.
4.5.1 Prueba de Campo 82
CONCLUSIONES 91
RECOMENDACIONES 95
BIBLIOGRAFIA 96
ANEXOS 99

RESUMEN

La Asignatura compiladores estudia la forma de cómo reconocer un programa escrito entendible al ojo humano y procesarlo a lenguaje maquina. Esta es un área donde se necesita que el estudiante adquiera destreza, y al no existir un sistema de ayuda que permita al alumno poner en practica los conocimientos aprendidos durante su asistencia a la materia; éste se queda solo con la parte teórica y al momento de ingresar a la vida laboral, el estudiante se encontraría con muchos interrogantes que no sabría resolver. El objetivo principal de este proyecto es realizar un Software que sirva de apoyo para el aprendizaje y practica de esta.

Para el desarrollo COMPISOFT se utilizó la metodología basada en componentes lo cual nos permitió reutilizar un sistema de codificación que ya existía, además pruebas de campo a los estudiantes a través de encuestas y visualización del software.

Los resultados arrojaron que con respecto al diseño de interfaz del usuario es muy amigable, mientras que para la navegabilidad y contenido del software es muy fácil y bueno. Por último el 100% de la población escogida para las pruebas consideran que el software debería utilizado como herramienta básica para el aprendizaje de los estudiantes.

Del proyecto se puede concluir que el 100% de los alumnos a los cuales se les hizo la prueba consideraron que el Laboratorio Virtual sería un buen apoyo pedagógico para mejorar el sistema de aprendizaje y que debería ser utilizado para la enseñanza de la asignatura.

INTRODUCCIÓN

"En 1946 se desarrolló el primer ordenador digital. En un principio, estas máquinas ejecutaban instrucciones consistentes en códigos numéricos que señalan a los circuitos de la máquina los estados correspondientes a cada operación. Esta expresión mediante códigos numéricos se llamó Lenguaje Máquina, interpretado por un secuenciador cableado o por un microprograma. Pero los códigos numéricos de las máquinas son engorrosos. Pronto los primeros usuarios de estos ordenadores descubrieron la ventaja de escribir sus programas mediante claves más fáciles de recordar que esos códigos numéricos; al final, todas esas claves juntas se traducían manualmente a Lenguaje Máquina. Estas claves constituyen los llamados lenguajes ensambladores, que se generalizaron en cuanto se dio el paso decisivo de hacer que las propias máquinas realizaran el proceso mecánico de la traducción. A este trabajo se le llama ensamblar el programa.

En 1954 se empezó a desarrollar un lenguaje que permitía escribir fórmulas matemáticas de manera traducible por un ordenador. Le llamaron FORTRAN (FORmulae TRANslator). Fue el primer lenguaje considerado de alto nivel. Se introdujo en 1957 para el uso de la computadora IBM modelo 704. Permitía una programación más cómoda y breve que lo existente hasta ese momento, lo que suponía un considerable ahorro de trabajo. Surgió así por primera vez el concepto de un traductor, como un programa que traducía un lenguaje a otro lenguaje. En el caso particular de que el lenguaje a traducir es un lenguaje de alto nivel y el lenguaje traducido de bajo nivel, se emplea el término compilador."[1]

Se pretende con la realización de este software que, los estudiantes del Programa de Ingeniería de Sistemas de la Universidad San Buenaventura, adquieran y se apropien de esta asignatura de una manera más fácil e interesante y entiendan en su totalidad los conceptos, sin ninguna clase de inconvenientes. Ya que se han presentado ciertas falencias en el conocimiento de esta asignatura, Se considera que a nivel tecnológico y económico el desarrollo de esta investigación es factible ya que se cuenta con los recursos requeridos para la implementación o culminación del proyecto.

Este proyecto se realizara diseñando el modelo lógico y físico, desarrollando este a través de un lenguaje de programación, por medio de pruebas para detectar errores y hacer sus respectivas correcciones, luego se elaboraran los manuales respectivos.

---

[1]'http://www.webexperto.com/compiladores/historia/introduccion.html. Introducción de un Compilador. Fecha de Visita: 15 de marzo 2006

1. PROBLEMA DE INVESTIGACIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

A nivel mundial se han desarrollado numerosos compiladores capaces de traducir un lenguaje de programación entendible para el ser humano a un lenguaje máquina, este desarrollo ha venido evolucionando a través del tiempo desde un FORTRAN (FORmulae TRANslator), hasta lo que conocemos hoy en día como la máquina virtual de java.

El aprendizaje de Compiladores en América latina ha sido de gran importancia para los desarrolladores debido a la gran necesidad de ir a la vanguardia de los grandes países en materia de tecnología ya que ésta va evolucionando día a día de una manera extraordinaria.

Los compiladores han sido utilizados a lo largo de la historia como una herramienta capaz de interpretar un lenguaje de programación y traducirlo a lenguaje maquina.

En Colombia, se encuentran Universidades que están capacitando a los estudiantes del Programa de Ingeniería de Sistemas, dentro de su pensum académico en la asignatura Compiladores, especialmente en lo que tiene que ver con la Teoría de Autómatas, Gramáticas Libres de Contextos y Analizadores Sintácticos, algunas con alto rendimiento académico y otras realmente con muchas deficiencias, en Cartagena no es la excepción, especialmente en la Universidad de San Buenaventura.

La Asignatura Compiladores es importante a la hora de saber como se maneja un Lenguaje de Programación y a la vez hace parte del desarrollo intelectual de un Ingeniero de Sistemas, haciendo parte de la esencia del profesional. Por experiencia, observación directa y entrevista, se han identificado algunas falencias en la Universidad de San Buenaventura Cartagena para la comprensión y el desarrollo de los temas mencionados en este proyecto. La asignatura como tal contiene temas de gran importancia y en algunos libros e Internet se muestra de forma compleja al igual que la asignatura, de manera que, un gran número de estudiantes de esta Institución Educativa, no comprenden en su totalidad al investigar los temas ya mencionados y necesitan apoyo de manera didáctica para su comprensión. Esta situación hace que los investigaciones quieran aportar a esta asignatura ya que es muy importante para la formación y desarrollo académico de los estudiantes, quienes reconocen lo necesario del aprendizaje, y así comprender como trabaja de forma adecuada un compilador, además, asignaturas como esta son determinantes a la hora de escoger o elegir una formación superior (Postgrado).

De ahí que se requiera Desarrollar un Software Educativo que sirva de Apoyo para el Aprendizaje de la asignatura compiladores para los estudiantes del programa Ingeniería de Sistemas de la Universidad de San Buenaventura Cartagena, haciendo énfasis en los temas Teoría de Autómatas, Gramáticas Libres de Contexto y Analizadores Sintácticos.

1.2 FORMULACIÓN DEL PROBLEMA

¿Cómo desarrollar un software educativo que sirva de apoyo para el aprendizaje de la asignatura compiladores del programa de Ingeniería de Sistemas de la Universidad de San Buenaventura Cartagena?

1.3 JUSTIFICACIÓN

En el programa de Ingeniería de Sistemas, en la Universidad de San Buenaventura Cartagena, es importante el estudio de la asignatura compiladores debido a que aporta grandes conocimiento en manejo de lenguajes de programación y sus componentes, el aprendizaje de esta asignatura se hace complicado ya que se debe tener claros muchos conceptos como la teoría de conjuntos, gramáticas entre otros, además el docente solo cuenta con los libros de la universidad, esto se hace dispendioso para los estudiantes, por tal motivo se hace necesario el uso de una herramienta que sirva de apoyo para el entendimiento de esta asignatura.

Desde el ámbito institucional La Universidad de San Buenaventura Cartagena plantea la investigación en el ámbito profesional como *"Una actividad presente en todas las áreas del saber para la formación de docentes y estudiantes; para el desarrollo en ciencias y tecnología y en las disciplinas sociales, humanas y solución de los problemas de las regiones y del país"*[2], Esta investigación desde el enfoque profesional es importante ya que al realizar un proyecto le brinda a la institución una aplicación cumpliendo así una labor social siguiendo los lineamientos filosóficos establecidos por parte de la Universidad de San Buenaventura Cartagena.

Esta propuesta está enmarcada EN EL PROGRAMA DE INGENIERÍA DE SISTEMAS, dentro de la Línea de Investigación Informática Educativa, y el área temática Desarrollo de Software Educativo. Ésta Área Temática es de gran importancia debido a su contenido y las herramientas que le permiten realizar un Software Educativo que sirva de apoyo para el aprendizaje de temas como Autómatas, Gramáticas Libres de Contexto y Analizadores Sintácticos. Debido a esto es el deseo de aportar una herramienta que ayude de manera clara a los estudiantes al entendimiento y aprendizaje de esta asignatura tan interesante como es

[2] Proyecto Educativo Bonaventuriano PEB. Pág. 85. Universidad de San Buenaventura.

Compiladores, de esta manera se pretende hacer uso del área temática para contribuir o apoyar el programa de Ingeniería de Sistemas de la Universidad de San Buenaventura Cartagena, a través de un Software Educativo.

El Software Educativo estará conformado por cinco (5) partes, organizado de la siguiente manera:

- ✓ Introducción teórica de cada uno de los temas a abordar
  - Definición de Autómatas Finitos Determinista y No Determinista
  - Definición de Gramáticas Libre o Independientes del Contexto
  - Definición de Analizadores Sintácticos
  - Todos los temas que el tutor quiera ingresar para su desarrollo.

- ✓ Ejemplos claros y demostrativos de cada uno de los temas mencionados anteriormente.

- ✓ Ejercicios propuestos por el Software de cada uno de los temas para un mejor entendimiento.

- ✓ Ayuda acerca del Software, aquí se mostrara de forma mas explicita algunas de las inquietudes que se podrían presentar a la hora de utilizar el software.

Lo que se pretende con ésta investigación es aportar no sólo los conocimientos sobre el tema sino también un Software que ayude de manera fácil, ágil el aprendizaje de una Asignatura tan importante como lo es Compiladores enmarcada en el pensum de el programa de Ingeniería de Sistemas de La Universidad de San Buenaventura Cartagena.

1.4 OBJETIVOS

1.4.1. Objetivo General

Desarrollar un Software Educativo que sirva de Apoyo para el Aprendizaje de la asignatura compiladores, mediante la plataforma de Windows, utilizando como herramientas de diseño Lenguaje Modelado Unificado (UML).

1.4.2 Objetivo Específicos

- Identificar los problemas y los requerimientos del usuario por medio de entrevista con el docente encargado de la asignatura para realizar el análisis.

- Diseñar el modelo lógico y físico teniendo en cuenta el análisis utilizando el lenguaje de modelado de un Software (UML).

- Desarrollar la aplicación a través de un Diseño Arquitectónico

- Aplicar pruebas de campo para medir el grado satisfacción a través de una encuesta que lleve el análisis estadístico.

- Elaborar el manual de sistema y de usuario a través de un documento para la respectiva instalación y utilización o manejo correcto del Software por parte del usuario.

- Analizar los costos que conllevará la elaboración del proyecto.

2. MARCO REFERENCIAL

2.1 INVESTIGACIONES PREVIAS

- **El Proyecto DISEÑO Y DESARROLLO DE UN SOFTWARE COMO HERRAMIENTA PEDAGÓGICA PARA SIMULAR EL COMPORTAMIENTO DE LOS MODELOS MATEMÁTICOS EN EL DESARROLLO DEL TEMA TEORÍA DE COLAS, EN LA ASIGNATURA INVESTIGACIÓN DE OPERACIONES II DE LA FACULTAD DE INGENIERÍA DE SISTEMAS DE LA UNIVERSIDAD DE SAN BUENAVENTURA CARTAGENA.** Este proyecto se realizó con el objetivo de desarrollar una herramienta que sirva de apoyo, para profundizar los componentes de la Investigación de Operaciones y mejorar las deficiencias en las actividades de VIII semestre. El cual fue desarrollado por los estudiantes del Programa de Ingeniería de Sistemas Víctor Fabián Castro Pérez y Geonavis Pájaro Blanco de la universidad de San Buenaventura Cartagena en el año 2003.

  El proyecto mencionado anteriormente, sigue con el lineamiento teórico implementado en el desarrollo del proyecto actual; de esta manera da un aporte a esta investigación pero enfocada a otra asignatura y semestre de la institución.

- **El Proyecto DISEÑO Y DESARROLLO DE UN INTERPRETE DE ALGORITMO DE PROGRAMACIÓN ESTRUCTURADA QUE SIRVA DE APOYO A LA ASIGNATURA COMPUTACIÓN I DE LA UNIVERSIDAD DE SAN BUENAVENTURA CARTAGENA**, desarrollada por la estudiante Luz Stella Garzón.

  El cual se realizó con el objetivo de desarrollar una herramienta que sirva de apoyo para la asignatura de Computación I, y para el buen aprendizaje por parte de los estudiantes del programa Ingeniería de Sistemas.

  Este proyecto sirve de aporte a la actual investigación ya que, define y maneja los conceptos de compiladores tales como: analizadores sintácticos, analizadores léxico, estos conceptos son base fundamental para el desarrollo de este proyecto.

2.2 BASES TEÓRICAS

La información teórica que servirá de apoyo parea la realización del siguiente proyecto es

2.2.1 Compilador: es un programa que lee el lenguaje fuente y lo traduce a lenguaje maquina, permitiendo ver al usuario como esta el programa que realizó en cuantos errores, sin ejecutarlo; este también permite compilar o examinar el código de forma independiente para después enlazarlo.

“A grandes rasgos un compilador es un programa que lee un programa a escrito es un lenguaje, el lenguaje fuente, y lo traduce a un programa equivalente en otro lenguaje, el lenguaje objeto. Como parte importante de este proceso de traducción, el compilador informa a su usuario de la presencia de errores en el programa fuente.

Clasificación de Compiladores

El programa compilador traduce las instrucciones en un lenguaje de alto nivel a instrucciones que la computadora puede interpretar y ejecutar. Para cada lenguaje de programación se requiere un compilador separado. El compilador traduce todo el programa antes de ejecutarlo. Los compiladores son, pues, programas de traducción insertada en la memoria por el sistema operativo para convertir programas de cómputo en pulsaciones electrónicas ejecutables (lenguaje de máquina). Los compiladores pueden ser de:

- una sola pasada: examina el código fuente una vez, generando el código o programa objeto.
- pasadas múltiples: requieren pasos intermedios para producir un código en otro lenguaje, y una pasada final para producir y optimizar el código producido durante los pasos anteriores.
- Optimación: lee un código fuente, lo analiza y descubre errores potenciales sin ejecutar el programa.
- Compiladores incrementales: generan un código objeto instrucción por instrucción (en vez de hacerlo para todo el programa) cuando el usuario teclea cada orden individual. El otro tipo de compiladores requiere que todos los enunciados o instrucciones se compilen conjuntamente.
- Ensamblador: el lenguaje fuente es lenguaje ensamblador y posee una estructura sencilla.
- Compilador cruzado: se genera código en lenguaje objeto para una máquina diferente de la que se está utilizando para compilar. Es perfectamente normal construir un compilador de Pascal que genere código para MS-DOS y que el compilador funcione en Linux y se haya escrito en C++.
- Compilador con montador: compilador que compila distintos módulos de forma independiente y después es capaz de enlazarlos.
- Autocompilador: compilador que está escrito en el mismo lenguaje que va a compilar. Evidentemente, no se puede ejecutar la primera vez. Sirve para hacer ampliaciones al lenguaje, mejorar el código generado, etc.
- Metacompilador: es sinónimo de compilador de compiladores y se refiere a un programa que recibe como entrada las especificaciones del lenguaje para el que se desea obtener un compilador y genera como salida el compilador para ese lenguaje. El desarrollo de los meta compiladores se encuentra con la dificultad de unir la generación de código con la parte de

análisis. Lo que sí se han desarrollado son generadores de analizadores léxicos y sintácticos. Por ejemplo, los conocidos: generador de analizadores léxicos LEX: generador de YACC: analizadores sintácticos desarrollados para UNIX. Los inconvenientes que tienen son que los analizadores que generan no son muy eficientes.

- Descompilador: es un programa que acepta como entrada código máquina y lo traduce a un lenguaje de alto nivel, realizando el proceso inverso a la compilación.

Funciones de un compilador

A primera vista, la diversidad de compiladores puede parecer abrumadora. Hay miles de lenguajes fuente, desde los lenguajes de programación tradicionales, como FORTRAN o Pascal, hasta los lenguajes especializados que han surgido virtualmente en todas las áreas de aplicación de la informática. Los lenguajes objeto son igualmente variados; un lenguaje objeto puede ser otro lenguaje de programación o el lenguaje de máquina de cualquier computador entre un microprocesador y un supercomputador. A pesar de existir una aparente complejidad por la clasificación de los compiladores, como se vio en el tema anterior, las tareas básicas que debe realizar cualquier compilador son esencialmente las mismas. Al comprender tales tareas, se pueden construir compiladores para una gran diversidad de lenguajes fuente y máquinas objeto utilizando las mismas técnicas básicas.

Nuestro conocimiento sobre cómo organizar y escribir compiladores ha aumentado mucho desde que comenzaron a aparecer los primeros compiladores a principios de los años cincuenta. Es difícil dar una fecha exacta de la aparición del primer compilador, porque en un principio gran parte del trabajo de experimentación y aplicación se realizó de manera independiente por varios grupos. Gran parte de los primeros trabajos de compilación estaba relacionada con la traducción de fórmulas aritméticas a código de máquina.

En la década de 1950, se consideró a los compiladores como programas notablemente difíciles de escribir. EL primer compilador de FORTRAN, por ejemplo, necesitó para su implantación de 18 años de trabajo en grupo (Backus y otros [1975]). Desde entonces, se han descubierto técnicas sistemáticas para manejar muchas de las importantes tareas que surgen en la compilación. También se han desarrollado buenos lenguajes de implantación, entornos de programación y herramientas de software.

Con estos avances, puede hacerse un compilador real incluso como proyecto de estudio en un curso de un semestre sobre diseño sobre de compiladores. ”[3]

2.2.2 Clasificación De Las Gramáticas: estas se dividen en cuatro tipos las cuales son: Recursivamente Enumerable, Sensibles al contexto, Libres de contexto, Regulares; las gramáticas libres o independientes de contexto pueden ser reconocido por un autómata deterministico o no deterministico esta es la gramática en la que se centrara este proyecto.

“En 1959, Noam Chomsky clasificó las gramáticas en cuatro tipos de lenguajes y esta clasificación es conocida como la jerarquía de Chomsky, en la cual cada lenguaje es descrito por el tipo de gramática generado. Estos lenguajes sirven como base para la clasificación de lenguajes de programación. Los cuatro tipos son: lenguajes recursivamente enumerables, lenguajes sensibles al contesto, lenguajes libres de contexto y lenguajes regulares. Dichos lenguajes también se identifican como lenguajes de tipo o, 1, 2 y 3.

Existe una exacta correspondencia entre cada uno de estos tipos de lenguajes y particulares arquitecturas de máquinas en el sentido que por cada lenguaje de tipo *T* hay una arquitectura de máquina *A* que reconoce el lenguaje de tipo *T* y por cada arquitectura *A* hay un tipo *T* tal que todos los lenguajes reconocidos por *A* son de tipo *T*. La correspondencia entre lenguajes y arquitectura son mostrados en la siguiente tabla” [4]

TIPO	LENGUAJES	TIPO DE MAQUINA
0	Recursivamente Enumerable	Máquina de Turing
1	Sensibles al contexto	Autómata Lineal Acotado
2	Libres de contexto	Autómatas de Pila
3	Regulares	Autómatas Finitos y Expresiones Regulares

Tabla 1. Los cuatro tipos de Gramáticas.

2.2.3 Gramáticas Libres Del Contexto: como su nombre lo dice estas gramáticas son libre o independientes, las cuales son reconocidas por los autómatas que se van a tratar en este proyecto.
“Estas gramáticas, conocidas también como gramáticas de tipo 2 o gramáticas independientes del contexto, son las que generan los lenguajes libres o independientes del contexto. Los lenguajes libres del contexto son aquellos que

[3] http://www.upseros.com/compiladores/introduccion/caracter05.php . Compiladores. Fecha de Visita: 30 de marzo de 2006.
[4] http://www.uned.es/guias12c/clasificacion/082301.pdf . Clasificación de las Gramáticas. Fecha de Visita: 30 de marzo de 2006.

pueden ser reconocidos por un autómata de pila determinístico o no determinístico.
Como toda gramática se definen mediante una cuadrupla G = (N, T, P, S), siendo
- N es un conjunto finito de símbolos no terminales
- T es un conjunto finito de símbolos terminales N ∩T = Ø
- P es un conjunto finito de producciones
- S es el símbolo distinguido o axioma S ¤(N U¤T)

En una gramática libre del contexto, cada producción de P tiene la forma

$$A \longrightarrow \omega \qquad \begin{cases} A \text{ ¤ } \text{¤}N \text{ U } \text{¤}\{S\} \\ \omega \text{ ¤¤ } \text{¤}(N \text{ U } T)^* - \{\varepsilon\} \end{cases}$$

Es decir, que en el lado izquierdo de una producción pueden aparecer el símbolo distinguido o un símbolo no terminal y en el lado derecho de una producción cualquier cadena de símbolos terminales y/o no terminales de longitud mayor o igual que 1.
La gramática puede contener también la producción S ⟶¤¤¤¤¤¤si el lenguaje que se quiere generar contiene la cadena vacía.” [5]

2.2.4 Autómatas Finitos: Estos autómatas están representados por un grafico que tiene un comienzo y un final, el cual representa una trayectoria donde se muestra como resultado un autómata finito.

“Un autómata finito es un conjunto de nodos y aristas que representan trayectorias para generar una expresión bajo un alfabeto. Un diagrama de transición es un autómata finito.
Los autómatas finitos se clasifican en:

- Autómatas finitos no determinísticos. NFA ¤
- Autómatas finitos determinísticos. DFA ¤

Autómata Finito No Determinístico

Un NFA esa un modelo matemático que consiste de:
1.- Un conjunto de estados.
2.- Un conjunto de símbolos de entrada.
3.- Una función de transición que corresponde pares estado-símbolo a conjuntos de estados.
4.- Un estado So que denota como el estado inicial.
5.- Un conjunto de estados F que denotan los estados de aceptación o finales.

[5] http://www.exa.unicen.edu.ar/catedras/ccomp1/Apunte5.pdf Gramáticas libres del contexto. Fecha de Visita: 30 de marzo de 2006.

Un NFA no tiene restricciones para que exista más de una transición con el mismo nombre a diferentes estados, por lo que en una representación tabular no es posible determinar de manera única el estado destino para un símbolo determinado.
Por ejemplo, el siguiente diagrama representa la expresión (a | b)* a b b

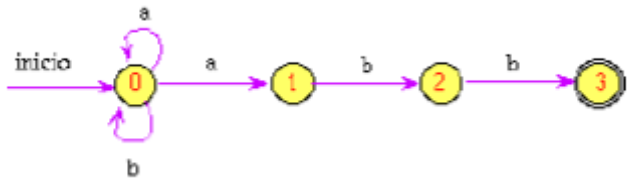


Figura 1. Ejemplo Autómata Finito No Determinístico

no podemos determinar a qué estado se avanza del estado 0 con el símbolo a; puede ser al propio 0 o al 1 pero ¿bajo qué condición? No está determinado. Incluso podría existirtransiciones sin ninguna restricción, que se denominan transicionesepsilon porque no requieren un símbolo determinado ocurra para realizarse. Nuevamente, esto se vuelve indeterminado en una representación tabular.

Autómata Finito Determinístico

Un DFA es un caso especial de NFA en el que ningún estado tiene transiciones para diversos estados bajo el mismo símbolo y no se permiten transiciones epsilon. Los diagramas de transición son autómatas determinísticos.
Por ejemplo, el DFA de (a | b)* a b b puede ser:

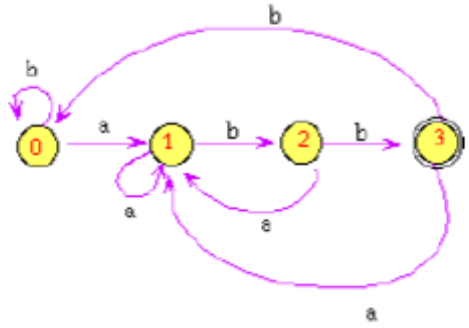


Figura 2. Ejemplo Autómata Finito Determinístico

Que podría ser muy aproximado al diagrama de transición que construiríamos para la expresión.

Para llegar de la expresión regular al autómata determinístico se emplea el método de Thompson que permite hacerla transformación y preparar la construcción del analizador de léxico." [6]

2.2.5 Ingeniería Del software: Así se conoce en otros países la Ingeniería de Sistemas la cual pretende no solo el diseño y desarrollo de programas de computación sino también la administración de proyectos a nivel empresariales.
"Hoy en día el software tiene un doble papel. Es un producto y al mismo tiempo el vehículo para hacer una entrega de un producto. Como producto hace entrega de la potencia informática del Hardware informático. Como vehículo utilizado para hacer entrega del producto, el software actúa como la base de control de la computadora (sistemas operativos), la comunicación de la información (redes), y la creación y control de otros programas (herramientas de software y Entornos).
El software transforma datos personales (p. Ej.: transacciones financieras de una persona) Para que los datos sean mas útiles en un contexto local; gestiona información comercial para mejorar la competitividad, proporciona el acceso a redes de información por todo el mundo (Internet) y también proporciona el medio de adquirir información en todas sus formas.

El software se desarrolla, no se fabrica en un sentido clásico aunque existen varias similitudes entre el desarrollo del software y la construcción del hardware ambas actividades son fundamentalmente diferentes.
En ambas actividades la buena calidad se adquiere mediante un buen diseño, el software no es susceptible a los males del entorno que hacen que el hardware se Estropee, cuando un componente de hardware se estropea, se sustituye por una pieza de repuesto.
No hay piezas de repuesto para el software, cada fallo en el software indica un error en el diseño o en el proceso mediante el que se tradujo el diseño a código maquina ejecutable, la mayoría del software se construye a medida, en vez de ensamblar componentes existentes se puede comprar software ya desarrollado, pero solo como una unidad completa, no como componentes que pueden reensamblarse en nuevos programas.

Aplicaciones del software

- Software de sistemas.
  Es un conjunto de programas que han sido escritos para servir a otros programas.
-
- Software de tiempo real.

[6] http://www.conocimientosweb.net/dcmt/ficha2185.html Autómatas finitos. Fecha de Visita: 30 de marzo de 2006.

- El software que mide, analiza, controla sucesos del mundo real conforme ocurren, se denomina de tiempo real.

- Software de gestión.
  El procesamiento de información comercial constituye la mayor de las áreas de aplicación del software.

- Software de ingeniería y científico.
  Esta caracterizado por los algoritmos de manejo de números.

- Software residente en memoria.
  Reside en memoria de solo lectura y se utiliza para controlar productos y sistemas de los mercados industriales y de consumo.

- Software de computadoras personales.
  Tiene miles de aplicaciones.

- Software de inteligencia artificial.
  Hace uso de los algoritmos no numéricos para resolver problemas complejos para los que no son adecuados los cálculos o el análisis directo."[7]

2.2.6 Software libre o "free software": Es la oportunidad que se le brinda al usuario de tomar un programa y ejecutarlo de manera libre en su computador personal u otro y realizar con él diferentes tareas o modificaciones.

"Permite al usuario copiar, modificar y distribuir copias del programa. El código fuente está disponible. Las compañías pueden distribuir copias por una cantidad de dinero. El término "Free software" es con frecuencia mal traducido e interpretado. La palabra inglesa 'free' puede significar dos cosas: gratis o libre. Lo que se quiere decir con "free software" es software libre, y por eso en español se usa ambos términos indistintamente (y no software gratis). Ahora bien, ¿a qué se refiere 'libre' en software libre? Se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, instalar, cambiar y mejorar el programa tantas veces como quieran. En concreto se consideran cuatro niveles de libertad:

- La libertad de usar el programa, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades. El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que puedes ayudar a tu vecino.

[7] http://www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html Ingeniería del Software Fecha de visita: 31 de Marzo de 2006

- La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto.

Con software libre la libertad que no se tiene es la de restringir estos derechos a otros usuarios, es decir, la libertad de eliminar libertades. Si se distribuye una copia o una modificación de un programa libre todos los usuarios disponen de los derechos antes citados.

Si un programa es libre, entonces puede ser potencialmente incluido en un sistema operativo libre tal como GNU, o sistemas GNU/Linux libres.
El desarrollo comercial del software libre ha dejado de ser inusual; el software comercial libre es muy importante" [8]

2.2.7 Lenguaje Modelado (UML): Este es el Lenguaje Modelado Unificado es una herramienta que le permite al programador armar de manera visual el diseño arquitectónico de un proyecto.

"UML es un lenguaje para especificar, construir, visualizar y documentar los artefactos de un sistema de software orientado a objetos (OO). Un artefacto es una información que es utilizada o producida mediante un proceso de desarrollo de software

UML se quiere convertir en un lenguaje estándar con el que sea posible modelar todos los componentes del proceso de desarrollo de aplicaciones. Sin embargo, hay que tener en cuenta un aspecto importante del modelo: no pretende definir un modelo estándar de desarrollo, sino únicamente un lenguaje de modelado.

Modelado de objetos.

En la especificación del UML podemos comprobar que una de las partes que lo componen es un metamodelo formal. Un metamodelo es un modelo que define el lenguaje para expresar otros modelos. Un modelo en OO es una abstracción cerrada semánticamente de un sistema y un sistema es una colección de unidades conectadas que son organizadas para realizar un propósito específico. Un sistema puede ser descrito por uno o más modelos, posiblemente desde distintos puntos de vista.

Una parte del UML define, entonces, una abstracción con significado de un lenguaje para expresar otros modelos (es decir, otras abstracciones de un sistema, o conjunto de unidades conectadas que se organizan para conseguir un propósito). Lo que en principio puede parecer complicado no lo es tanto si

[8] http://www.softwarelibre.cl/drupal/ Software Libre Fecha de visita: 31 de Marzo de 2006

pensamos que uno de los objetivos del UML es llegar a convertirse en una manera de definir modelos, no sólo establecer una forma de modelo, de esta forma simplemente estamos diciendo que UML, además, define un lenguaje con el que podemos abstraer cualquier tipo de modelo.

Artefactos para el Desarrollo de Proyectos

Un artefacto es una información que es utilizada o producida mediante un proceso de desarrollo de software. Pueden ser artefactos un modelo, una descripción o un software. Los artefactos de UML se especifican en forma de diagramas, éstos, junto con la documentación sobre el sistema constituyen los artefactos principales que el modelador puede observar.

Se necesita más de un punto de vista para llegar a representar un sistema. UML utiliza los diagramas gráficos para obtener estos distintos puntos de vista de un sistema:

- Diagramas de Implementación.
- Diagramas de Comportamiento o Interacción.
- Diagramas de Casos de USO.
- Diagramas de Clases.

Ejemplo de algunos de los diagramas que utiliza UML.

Diagramas de Implementación Se derivan de los diagramas de proceso y módulos de la metodología de Booch, aunque presentan algunas modificaciones. Los diagramas de implementación muestran los aspectos físicos del sistema. Incluyen la estructura del código fuente y la implementación, en tiempo de implementación. Existen dos tipos:
Diagramas de componentes, Diagrama de plataformas despliegue

Características del UML: Además de los conceptos extraídos de métodos anteriores, se han añadido otros nuevos que vienen a suplir carencias antiguas de la metodología de modelado. Estos nuevos conceptos son los siguientes:

- Definición de estereotipos: un estereotipo es una nueva clase de elemento de modelado que debe basarse en ciertas clases ya existentes en el metamodelo y constituye un mecanismo de extensión del modelo.
- Responsabilidades.
- Mecanismos de extensibilidad: estereotipos, valores etiquetados y restricciones.
- Tareas y procesos.
- Distribución y concurrencia (para modelar por ejemplo ActiveX/DCOM y CORBA).

- Patrones/Colaboraciones.
- Diagramas de actividad (para reingeniería de proceso de negocios)
- Clara separación de tipo, clase e instancia.
- Refinamiento (para manejar relaciones entre niveles de abstracción). Interfaces y componentes" [9]

2.2.8 Lenguaje de Programación: es una herramienta que permite codificar en un lenguaje entendible para el usuario y la maquina es decir escribir un programa.

"Lenguajes de Programación: Su función es proporcionar instrucciones al sistema de la computadora para que pueda realizar una actividad de procesamiento. Cada lenguaje de programación utiliza un grupo de símbolos o reglas que tiene un significado específico a eso se le llama sintaxis.

EVOLUCIÓN DE LOS LENGUAJES DE PROGRAMACIÓN:

1. Lenguajes de primera generación
2. Lenguajes de segunda generación
3. Lenguajes de tercera generación
4. Lenguajes de cuarta generación

- Lenguajes de programación orientados a objetos
- Smalltalk
- C++
- Java
- Lenguajes de programación visual

SELECCIÓN DE UN LENGUAJE DE PROGRAMACIÓN

- El mejor lenguaje de programación para usarse en un programa en particular incluye equilibrar las características funcionales del lenguaje con aspectos como costo, control y complejidad.
- Un factor importante que debe considerarse al seleccionar cualquier lenguaje de programación es la cantidad de control directo que se necesita para operar el hardware.

TRADUCTORES DE LENGUAJE

- Traductor de lenguaje. Software de sistemas que convierte un código fuente del programador en su equivalente en lenguaje maquina.

[9] http://www.dcc.uchile.cl/~psalinas/uml/introduccion.html Lenguaje Modelado Unificado (UML) Fecha de visita: 31 de Marzo de 2006

- Código fuente. Código de programación de nivel alto escrito por el programador.

- Código objeto. Otro nombre para el código de lenguaje maquina.

- Intérprete. Traductor de lenguajes que convierte, ala vez, una sentencia de programa a un código de máquina.” [10]

2.2.9 Flash Macromedia : Es una herramienta avanzada para desarrollar animaciones. Es una aplicación sencilla de uso y a la vez potente y cuyos resultados son unos archivos de tamaño pequeño, lo que permite agilizar su descarga.

Además incorpora nuevas acciones en las animaciones y permitiendo una mayor y más compleja interactividad, sin necesidad de desarrollar programación aparte. Dispone también de una mejora de la interfase convirtiendo al programa en una herramienta de fácil manejo.

Flash Macromedia trabaja con un lenguaje de programación llamado **ActionScript** que es un lenguaje de programación orientado a objetos (OOP), utilizado en especial en aplicaciones Web animadas realizadas en el entorno Macromedia Flash. ActionScript es un lenguaje de script, no requiere la creación de un programa completo para que la aplicación alcance los objetivos.
El lenguaje está basado en especificaciones de estándar de industria ECMA-262, un Standard para Javascript, de ahí que ActionScript se parezca tanto a Javascript.

2.2.10 PHP. Es un lenguaje de script incrustado dentro del HTML. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características especificas de si mismo. La meta del lenguaje es permitir rápidamente a los desarrolladores la generación dinámica de páginas.

PHP “*(acrónimo recursivo de "PHP: Hypertext Preprocessor", originado inicialmente del nombre PHP Tools, o Personal Home Page Tools) es un lenguaje de programación interpretado, con licencia OpenSource. Debido al diseño de PHP, también es posible crear aplicaciones con una interfaz gráfica para el usuario (GUI), utilizando la PHP-GTK. También puede ser usado desde la Línea de comandos, como Perl o Python.*

Su interpretación y ejecución se da en el servidor en el cual se encuentra almacenada la página y el cliente solo recibe el resultado de la ejecución. Cuando el cliente hace una petición al servidor para que le envíe una página

[10] http://www.programacion.com/introduccion/evolucion/csharp/ Lenguaje de Programación Fecha de visita: 1 de Abril de 2006

web, enriquecida con código PHP, el servidor interpretará las instrucciones mezcladas en el cuerpo de la página y las sustituirá con el resultado de la ejecución antes de enviar el resultado a la computadora del cliente. Además es posible utilizarlo para generar archivos PDF, Flash o JPG, entre otros.

Permite la conexión a numerosas bases de datos de forma nativa tales como:

- *MySQL*
- *Postgres*
- *Oracle*
- *ODBC*
- *IBM DB2*
- *Microsoft SQL Server*
- *SQLite*

Lo cual permite la creación de Aplicaciones web muy robustas.
PHP tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX, Linux, Windows y Mac OS X, y puede interactuar con los servidores de web más populares" [11].

Entre las ventajas que posee PHP podemos citar:

- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- Leer y manipular datos desde diversas fuentes, incluyendo datos que pueden ingresar los usuarios desde formularios HTML.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una muy buena documentación en su página oficial).
- Es Libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.

2.2.11 MySQL. Es uno de los Sistemas Gestores de bases de Datos (SQL) más populares, ya que se encuentra desarrollada bajo la filosofía de "Código Abierto". Está siendo utilizado por muchos desarrolladores, ya que permite una durabilidad por muchos años de la base de datos en ambientes exigentes.

El servidor de base de datos MySQL se acopla mucho con PHP y el servidor Web APACHE. Los tres son la combinación perfecta para la creación de páginas web dinámicas.
Las características principales de MySQL son:

- Rápido
- Seguro
- Fácil de usar
- Velocidad
- Conectividad
- Seguridad

[11] http://es.wikipedia.org/wiki/PHP

2.3 DEFINICIÓN DE TÉRMINOS BÁSICOS

SOFTWARE: Conjunto de programas y sistema disponibles en un ordenador.
APLICACIÓN: Cada uno de los programas que, una vez ejecutados, permiten trabajar con el computador. Los procesadores de textos, hojas de cálculo, bases de datos, programas de dibujo y paquetes estadísticos, entre otros, son aplicaciones.

COMPILADOR: Un compilador lee el código fuente creado en un determinado lenguaje de programación, lo interpreta, comprueba su sintaxis y traduce a lenguaje o código máquina toda la serie de instrucciones, generando el archivo ejecutable final (programa compilado)

LENGUAJE: Cualquier lenguaje artificial que puede utilizarse para definir una secuencia de instrucciones para su procesamiento por un ordenador o computadora.

SISTEMA: Cualquier conjunto de dispositivos que colaboran en la realización de una tarea.

PROGRAMA: En informática, sinónimo de software, el conjunto de instrucciones que ejecuta un ordenador o computadora.

CÓDIGO: En la Teoría de la Información, es la forma que toma la información que se intercambia entre la Fuente (el emisor) y el Destino (el receptor) de un lazo informático. Implica la comprensión o decodificación del paquete de información que se transfiere.

HARDWARE: conjunto de elementos materiales que componen un ordenador. En dicho conjunto se incluyen los dispositivos electrónicos y electromecánicos, circuitos, cables, tarjetas, armarios o cajas, periféricos de todo tipo y otros elementos físicos.

ALGORITMO: es un conjunto de operaciones y normas que deben seguirse para resolver un problema. Los algoritmos se codifican en sistemas comprensibles para los motores de búsqueda. El lenguaje algorítmico, es por tanto, más cercano a ser entendido por una máquina que por las personas.

INFORMACIÓN: es un conjunto organizado de datos, que constituyen un mensaje sobre un determinado ente o fenómeno. De esta manera, si por ejemplo organizamos datos sobre un país (número de habitantes, densidad de población, nombre del presidente, etc.) y escribimos por ejemplo, el capítulo de un libro, podemos decir que ese capítulo constituye información sobre ese país.

PROGRAMACIÓN: al acto de crear un programa de computadora, un conjunto concreto de instrucciones que una computadora puede ejecutar. El programa se escribe en un lenguaje de programación, aunque también se pueda escribir directamente en lenguaje de máquina, con cierta dificultad. Un programa se puede dividir en diversas partes, que pueden estar escritas en lenguajes distintos.

NODOS: Puntos en los cuales se ubican equipos de procesamiento en una red, ya los cuales están conectados los enlaces de la misma.

INFORMÁTICA: es la ciencia del tratamiento automático de la información mediante un computador (llamado también ordenador o computadora).

ENSAMBLADOR: lenguaje de programación de medio nivel, el cual es traducible directamente a lenguaje máquina. Está constituido por un juego de instrucciones característico de cada máquina con distinta arquitectura.

3. DISEÑO METODOLÓGICO

3.1 ENFOQUE, MÉTODO Y TIPO DE INVESTIGACIÓN

Este proyecto se fundamenta en la investigación aplicada, la cual es una de las modalidades de investigación de la Universidad de San Buenaventura ya que adquieren nuevos conocimientos que se aplican en la realización de un software produciendo un aporte al desarrollo de la tecnología informática.

El método es cualitativo ya que nos permite hacer una aproximación global de las situaciones y el descubrimiento e identificación de nuevas ideas, pensamientos, sentimientos, conocimientos preliminares y comprensión de estas ideas y objetos, es abierta, de Plazos Relativamente cortos, Preguntas y respuestas subjetivas de contenido representativo.

El tipo de investigación es proyectivo por que tiene por objeto, el diseño, la propuesta o creación de modelo que permita solucionar una necesidad de tipo práctico (Proyectos Factibles), ya que este proyecto busca construir una herramienta que sirva de apoyo para los estudiantes de la asignatura Compiladores de la Universidad de San Buenaventura Cartagena.

3.2 DELIMITACIÓN

3.2.1 Conceptual: La investigación a realizar hace parte de la Ingeniería de Sistemas específicamente con el desarrollo de un software Educativo.

El Software a realizar será de gran ayuda para los estudiantes de Universidad de San Buenaventura Cartagena, en especial a los de décimo semestre, este Software se desarrollará haciendo énfasis en la Teoría de Autómatas, Gramáticas Libres de Contexto y Analizadores Sintácticos, para un entendimiento de estos temas que servirán para el enriquecimiento profesional.

3.2.2 Geográfica: El proyecto ésta enmarcado en el área espacial de la Universidad de San Buenaventura Cartagena ubicado en el barrio de Ternera en la cuidad de Cartagena.

3.3 FUENTES DE INFORMACIÓN

3.3.1 Primarias: Para la realización de esta investigación se contara con la ayuda de personas especializadas, entrevistas a docentes, pruebas de campo las que sean necesarias para las correcciones del software, y de esta manera tener una mejor orientación con respecto al proyecto.

3.3.2 Secundarias: Durante el desarrollo del presente proyecto de investigación se utilizaran Libros especializados en Desarrollo e Ingeniería de Software, Compiladores y Lenguaje de programación, revistas de actualidad, periódicos e Internet, consultas de proyectos realizados sobre el tema de investigación.

3.4 VARIABLES

3.4.1 Identificación de variables: Las variables utilizadas en el presente proyecto de investigación son:
Características del Software educativo
Aspectos de la asignatura Compiladores

3.4.2 Operacionalización de Variables

VARIABLE	DEFINICIÓN	DIMENSIÓN	INDICADOR
Características del Software educativo	Son programas para ordenador creados con la finalidad específica de ser utilizados como medio didáctico, es decir, para facilitar los procesos de enseñanza y de aprendizaje.	Análisis	Identificación del problema y de los requerimientos.
		Diseño	Creación y documentación del modelo físico y lógico.
		Implementación	Codificación, instalación y pruebas.
Aspectos de la asignatura Compiladores	Es aquel que lee el código fuente creado en un determinado lenguaje de programación, lo interpreta comprueba sus sintaxis lo traduce a lenguaje o código maquina toda la serie de instrucciones, generando el archivo ejecutable final (Programa Compilado).	Contenido	Analizador léxico Analizadores sintácticos Gramáticas libres del contexto Autómatas Finitos
		Metodología	Clases en el aula parte teórica, clases en la sala de sistemas parte practica.
		Sistema de evaluación	Pruebas en la sala de informática, quices, trabajos, exposiciones

Tabla 2. Variables

3.5 RECURSOS TECNOLÓGICOS

3.5.1 Paradigma de programación: El paradigma de programación a utilizar es la programación orientada a objetos.

"La programación orientada a objetos proporciona ventajas como: desarrollo rápido, fácil mantenimiento, reutilización de diseños y códigos, reduce los riesgos en el desarrollo de sistemas complejos, facilita la construcción de sistemas distribuidos, entre otras. Sin embargo, a menudo requiere un gran conocimiento de los mecanismos de diseño y el desarrollo de la clase inicial, la cual, en muchos casos puede consumir mucho tiempo de elaboración.

Los lenguajes orientados a objetos tienen una ventaja en la manera en que los objetos se crean y se conectan. Desde una perspectiva de programación, los objetos son entidades con características descriptivas y capacidades funcionales encapsuladas en unidades autónomas. Un mecanismo estándar para crear nuevos objetos e interconectarlos (proporcionarles interfaces) es una característica inherente de los lenguajes orientados a objetos."[12]

3.5.2 Modelo de construcción de software: Para la realización de este proyecto el modelo de construcción de software que se piensa utilizar es RUP "El Proceso Unificado Racional (Rational Unified Process), es un proceso de desarrollo de software y junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo)
- Pretende implementar las mejores practicas en Ingeniería de Software
    - Desarrollo iterativo
    - Administración de requisitos
    - Uso de arquitectura basada en componentes
    - Control de cambios
    - Modelado visual del software
    - Verificación de la calidad del software

El RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que

[12] http://www..lawebdelprogramador.com/programacion/oriobj/pe.htm Programación Orientada A Objetos Fecha de visita 2 de Abril de 2006

desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

El RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al final de cada ciclo, cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- inicio: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos
- elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos
- construcción: se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario
- transición: se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requerimientos a ser analizados." [13]

3.5.3 Herramientas De Diseño. ***UML (Unified Modeling Language – Lenguaje Unificado del Modelado)*** es la herramienta que se utilizó para el diseño de este software ya que el lenguaje elegido debería ser:

- "*Suficientemente expresivo, de manera que sea posible expresar los aspectos del diseño que será necesario tratar, y que reflejen de forma que tengan sentido, los cambios en el diseño que se lleven a cabo durante el desarrollo como cambios en el modelo.*
- *Suficientemente fácil de utilizar, de forma que el lenguaje de modelado ayude a tener un conocimiento claro en vez de proporcionar el camino para tener dicho conocimiento claro.*
- *Inequívoco, para que el lenguaje de modelado ayude a resolver malos entendidos en vez de presentar más.*
- *Soportado por herramientas adecuadas, de manera que el esfuerzo de los desarrolladores pueda utilizarse en un trabajo que requiera su habilidad, no en un trabajo rutinario como crear un diagrama con una herramienta de dibujo.*
- *Generalmente utilizado, por gran variedad de razones. Por supuesto cuanto más general sea la utilización de un lenguaje es más probable que se cumplan los cuatro puntos anteriores. También:*
  - *cuando se incorpora gente nueva en el proyecto, es una ventaja si ya conocen el lenguaje de modelado en vez de tener que aprenderlo.*
  - *Para hacer diseño basado en componentes hay que ser capaz de leer las descripciones de los componentes, y cuanto más rápido y fácil se pueda hacer es más barato tener en cuenta un componente. Cuanto más general*

[13] http://es.wikipedia.org/wiki/RUP RUP (Proceso Unificado Racional) Fecha de visita 2 de Abril de 2006

sea la utilización de un lenguaje de modelado, mayor es la posibilidad de que sea el mismo que el escritor del componente decidió utilizar"[14].

Por otra parte es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. Cubre tanto aspectos conceptuales (procesos de negocio, funciones del sistema) como cosas concretas (clases, esquemas de bases de datos, componentes reutilizables).

3.5.4 Herramientas de Desarrollo. Para este proyecto fue necesario utilizar diferentes herramientas indispensables para el desarrollo del software, entre las cuales se tiene:

APACHE

Es un servidor web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos. Las diferentes plataformas y los diferentes entornos, hacen que a menudo sean necesarias diferentes características o funcionalidades, o que una misma característica o funcionalidad sea implementada de diferente manera para obtener una mayor eficiencia. Apache se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular.

Este diseño permite a los administradores de sitios web elegir que características van a ser incluidas en el servidor seleccionando que módulos se van a cargar, ya sea al compilar o al ejecutar el servidor.

Apache nos brinda muchas ventajas como son:

- Soportar de una forma más fácil y eficiente una amplia variedad de sistemas operativos.
- El servidor puede personalizarse mejor para las necesidades de cada sitio web.

MySQL Server: "es la base de datos de código fuente abierto más usada del mundo. Su ingeniosa arquitectura lo hace extremadamente rápido y fácil de personalizar. La extensiva reutilización del código dentro del software y una aproximación minimalística para producir características funcionalmente ricas, ha dado lugar a un sistema de administración de la base de datos incomparable en velocidad, compactación, estabilidad y facilidad de despliegue. La exclusiva separación del core server del manejador de tablas, permite funcionar a MyQSL bajo control estricto de transacciones o con acceso a disco no transaccional ultrarrápido." [15]

---

[14] BOOCH, Grady; JACOBSON, Ivar; RUMBAUGH, James. Utilización de UML en Ingeniería del Software con Objetos y Componente. Madrid; Addison Wesley Iberoamericana, p. 55-56

[15] http://www.software-shop.com/Productos/MySQL/mysql.html MySQL Server Fecha de visita 2 de Abril de 2006

“PHP es un lenguaje de programación usado generalmente para la creación de contenido para sitios web. PHP es el (acrónimo recursivo de "PHP: Hypertext Preprocessor", inicialmente PHP Tools, o, Personal Home Page Tools) es un lenguaje interpretado usado para la creación de aplicaciones para servidores, o creación de contenido dinámico para sitios web, y últimamente también para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando la librería GTK+.”[16]

[16] http://es.wikipedia.org/wiki/PHP PHP **H**ypertext **P**reprocessor Fecha de visita 5 de Abril de 2006

4. DESARROLLO DEL PROYECTO

Este proyecto utiliza la ***Metodología de Construcción*** el Modelo de Basado en Componentes por que permite "*reutilizar piezas de código conllevando a diversos beneficios como las mejoras a la calidad, la reducción del ciclo de desarrollo y el mayor retorno sobre la inversión*"[17], permitiendo así tomar menos tiempo en la entrega del proyecto, la cual te permite realizar una organización ordenada y disciplinada el desarrollo de la aplicación.

Este proyecto se había descrito como herramienta de programación JAVA, pero realizando un análisis más detallado por parte de los investigadores, se llegó a la conclusión que también se utilizaría la herramienta FLASH MX Professional 2004 de MACROMEDIA por las ventajas que nos brinda y que describiremos más adelante y que a su vez trabaja con ActionScript que es el lenguaje que FLASH para añadir interactividad básica o compleja a aplicaciones, como son botones entre otras.

La justificación para acompañar el proyecto con otra herramienta son:

- Limitaciones de conocimiento acerca de la herramienta JAVA por parte de los investigadores.
- Herramienta con la que pueden crearse desde animaciones simples hasta complejas aplicaciones Web interactivas
- Las aplicaciones de Flash se pueden enriquecerse con imágenes, vídeo, etc.
- Flash incluye muchas funciones que la convierten en una herramienta con muchas prestaciones entre ellas tenemos: la posibilidad de arrastrar y soltar componentes de la interfaz de usuario, comportamientos incorporados que añaden código ActionScript al documento y varios efectos especiales.
- Cuando creas un documento de Flash, este al guardarse, tiene la extensión .fla. Una vez que está preparado para desarrollar su contenido de Flash, lo publica, creando un archivo con una extensión .swf. Flash Player, que se describe en la siguiente sección, ejecuta el archivo SWF

Al no haber inconvenientes, para el desarrollo del sistema se decidió Acompañar la herramienta JAVA por FLASH MX Professional MX, ya que el uso de este software le proporciona mayor interés y atractivo a este proyecto LVTD.

Para la realización de un sistema efectivo, óptimo y que cumpla con las necesidades exigidas, se ha tomado la metodología, herramientas y paradigmas Orientadas a Objetos, realizando así cada uno de los pasos que presenta la

[17] Desarrollo de Software basado en Componentes.
http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/MTJ_3985.asp

Ingeniería de Software y así reduciendo los errores que se pueden presentar en un software.

Después de haber estudiado los modelos de construcción de software se escogió el Modelo Basado en Componentes ya que cuenta fases bien definidas que ayudaron a conseguir el objetivo final de este proyecto.
Después de haber estudiado los modelos de construcción de software se escogió el Modelo Basado en Componentes ya que cuenta fases bien definidas que ayudaron a conseguir el objetivo final de este proyecto.

El modelo de Desarrollo Basado en Componentes se adapta especialmente a *UML (Unified Modeling Language – Lenguaje Unificado del Modelado* ya que éste es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software, este *Modelo Basado en Componentes* contiene varias actividades o fases que son. Comunicación con el Cliente, Planificación, Análisis de Riesgos, Construcción y Adaptación de la Ingeniería y Evaluación del cliente. Por medio de estos se identificaron los requerimientos del sistema propuesto.

4.1 COMUNICACIÓN CON EL CLIENTE.

La realización de una investigación detallada se da a partir de libros, consultas en Internet, personas especializadas en el área de compiladores, Diseño de Software y Metodologías de Estudios, esta fueron las bases para la realización de recolección de información para el desarrollo del proyecto.
Se hizo una entrevista al docente de la cátedra Compiladores Ing. Carlos cortes y tutor de este proyecto, el cual planteó las necesidades que se tienen en dicha cátedra, una de ellas es la herramienta que sirva de apoyo en esta cátedra tan importante.

Esta investigación detallada y profunda conllevó a una serie de pautas, que los investigadores tendrán en cuenta para la buena presentación y calidad de la aplicación.

4.1.1 Presentación de la Información: A través del análisis con el docente de la cátedra Compiladores Ing. Carlos Cortes, se llegó a la conclusión de qué esperan tanto el docente como el estudiante del software y se llegaron a los siguientes ítems:

- El software debe contener ***Conceptos Teóricos*** referentes a cada tema que se vaya a tratar durante el transcurso de la cátedra.
- **Ejemplo**s para el mejor entendimiento de cada uno de los temas
- **Ejercicios** de autómatas finitos con graficas que pueda manipular.

- Se deben ***evaluar*** a los estudiantes para saber que tanto han aprendido y que las preguntas sean escogidas por el tutor.
- Debe existir una **ayuda** que explique al usuario, ya sea docente o alumno, el funcionamiento de cada simulador.
- El Administrador debe tener la posibilidad de actualizar la evaluación, pero para el ingreso debe tener una contraseña.

4.1.2 Determinación de Requerimientos Funcionales. El proceso que se llevó a cabo para identificar los requerimientos que definen la funcionalidad del software llevó a los siguientes resultados:

- Presentar información sobre conceptos básicos acerca de las transmisiones de datos.
- El sistema debe controlar el acceso según el perfil del usuario.
    - Debe existir una interacción entre el estudiante y el sistema, Utilizando ejemplos y ejercicios.
- El programa debe permitir actualizar (ingresar) las preguntas para las evaluaciones.
- Ayudar al estudiante a resolver las dudas que tenga sobre la manipulación de la aplicación.
-

4.1.3 Determinación de Requerimientos No Funcionales. Para que el software tenga un buen desempeño y buen funcionamiento, debe presentar las siguientes características.

- Robusto
- Fácil de Utilizar
- Flexible
- Heterogeneidad

4.2 PLANIFICACIÓN

Una buena planificación de un buen proyecto da resultados óptimos, es por esto que a continuación se muestra la tabla las actividades que se realizaron durante el desarrollo del proyecto.

Actividad	Tiempo*	Recurso	Personas Involucradas
Recopilación de la Información	1 semana 19 May-11 – 26 May- 11	Encuesta y Observación	Docente y Alumnos de la Cátedra
Análisis de la Información	2 semanas 12 ago-11 a 26 Ago-11	Equipos de Cómputos	Investigadores
Diseño del Software	2 meses Oct -11 a 3 Dic 20-11	Libro UML, Equipos de Cómputos	Investigadores y Tutor
Actividad	**Tiempo**	**Recurso**	**Personal Involucrado**
Desarrollo del Software	3 Meses 1 Enero-12 a 29 Marzo-12	Software Apache, MySQL,PHP,Flash Macromedia, ActionScript ,Equipos de Cómputos, Tutoriales	Investigadores
Pruebas Piloto y de Campo	1 semanas 15 Marzo-13 a 25 Mar-13	Encuesta, Equipos de Computo	Personal Especializado y estudiantes de la cátedra

Tabla 3. Planificación

4.3 ANÁLISIS DE RIESGO

Se determinaron los riesgos que se pueden presentar durante el desarrollo y ejecución de la aplicación, y que si no se tienen en cuenta pueden causar un rechazo por parte de los clientes del programa, entre ellos tenemos:

- **Tecnológicos**
- **Económicos**
- **Tiempo**
- **Humanos**

Estos riesgos, que podrían aparecer antes, durante y después del desarrollo del producto, afectarían directamente a la calidad del mismo o al rendimiento del software en desarrollo. Entre ellos se detectaron los siguientes:

- Interfaz no agradable a usuario.
- No existe conexión con la base de datos.
- Incompatibilidad con el sistema.
- Desactivación Temporal del Servidor.

Riesgos	Causas	Efectos	Estrategias de Prevención	Plan de Contingencia
Interfaz no agradable a usuario.	Mal diseño de la interfaz Gráfica.	Poco interés en el software.	Mostrar aplicación al cliente con regularidad.	Volver a diseñar la Interfaz.
No existe conexión con la base de datos.	Mala configuración del servidor.	No se guardan las preguntas.	Actualizar la evaluación constantemente.	Manual para la configuración del servidor.
Incompatibilidad con el sistema.	No existe el hardware ni el software requerido.	No se puede presentar el software como se debe.	Especificar en la Ayuda los requerimientos de hardware y software.	Asegurarse que los equipos cumplan con los requerimientos antes de instalar el software.
Desactivación Temporal del Servidor.	No existe energía auxiliar de apoyo.	Poco interés en el software.	Obtener un dispositivo de almacenamiento de energía.	

Tabla 4. Análisis de Riesgos.

4.4 FASE DE CONSTRUCCIÓN Y ADAPTACIÓN DE LA INGENIERÍA

4.4.1 Casos de Usos

Descripción del Proceso.

Después de reunir y analizar todos los requerimientos, se logró reconocer lo que el usuario necesitaba y además la forma de cómo representar el software ante éste. Para lograrlo se utilizó los diagramas de casos de usos, los cuales representan la forma de como un usuario o Actor que opera con el sistema en desarrollo, la interacción que existe entre cada caso de uso y el tiempo que transcurre.

Como primera medida se identificaron las personas que tendrán acceso directo con el software Compisoft, para la elaboración de los respectivos diagramas.

- ✓ Alumno
- ✓ Docente ó Administrador (en este caso utilizaremos el nombre Administrador)

En Segunda medida se identifican los casos de usos para cada usuario.

Actor Alumno:

- ✓ Selecciona la opción deseada.

Actor Administrador

- ✓ Selecciona la opción Administrador.

Después de haber identificado los casos de usos, se procederá a describir los Diagramas de Caso de Uso general y el detallado.

Diagrama de Caso de Uso General

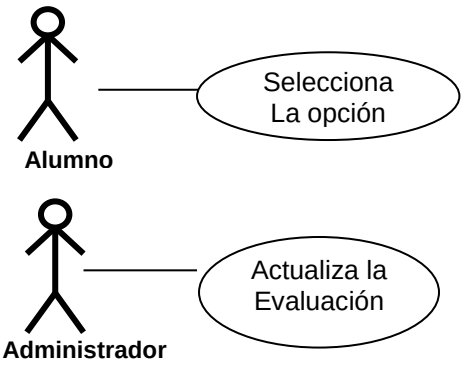


Figura 3. Diagrama Caso de Uso General

La figura anterior describe al Actor Alumno con la opción de seleccionar, entre una de las diferentes opciones que presenta la interfaz principal del software COMPISOFT, como son el de seleccionar la opción de Teoría, *Ejemplos Ejercicios, Videos, Ayuda, Créditos* tomando así la opción que desea Estudiar (uno de los casos de uso podría ser *Teoría)*. Mientras que el Actor Administrador tiene la opción de Agregar Preguntas activando el Caso de uso *Actualizar Preguntas.*

Diagrama de Caso de Uso Detallado

Esta sección presenta los casos de usos de forma más detallada, con el objetivo de que el lector de este proyecto observe las interacciones que surgen entre los elementos (Actores, Casos de Usos, entre los mismos Casos de Usos y el Sistema).

Para comprender los casos de usos detallados, se explicará cada uno por separado (Ver Figura 2 y 3). Además se ha decidido identificar cada caso de uso con la abreviatura **"CU"** (Casos de Usos), acompañado de un número, y así diferenciarlos uno del otro. Ej. CU1, CU2, CU3 etc.

CU1: Seleccionar Teoría
CU2: Seleccionar Ejemplos
CU3: Seleccionar Ejercicios
Extendiendo a su vez los casos de usos
CU8: Seleccionar Software de aprendizaje Autómata Finito
CU9: Seleccionar Ejercicios de Auto evaluación
CU4: Seleccionar Videos
CU5: Seleccionar Ayuda
CU6: Seleccionar Créditos
CU7: Seleccionar Admin.
Extendiendo a su vez del caso de uso
CU10: Seleccionar actualizar contenido teórico
CU11: Seleccionar Añadir preguntas
CU12: Seleccionar activar o desactivar preguntas
CU13: Seleccionar actualizar preguntas
CU14: Seleccionar Borrar preguntas
CU15: Seleccionar consultar a un estudiante
CU16: Seleccionar añadir contenido teórico

Diagrama de Caso de Uso Detallado Alumno

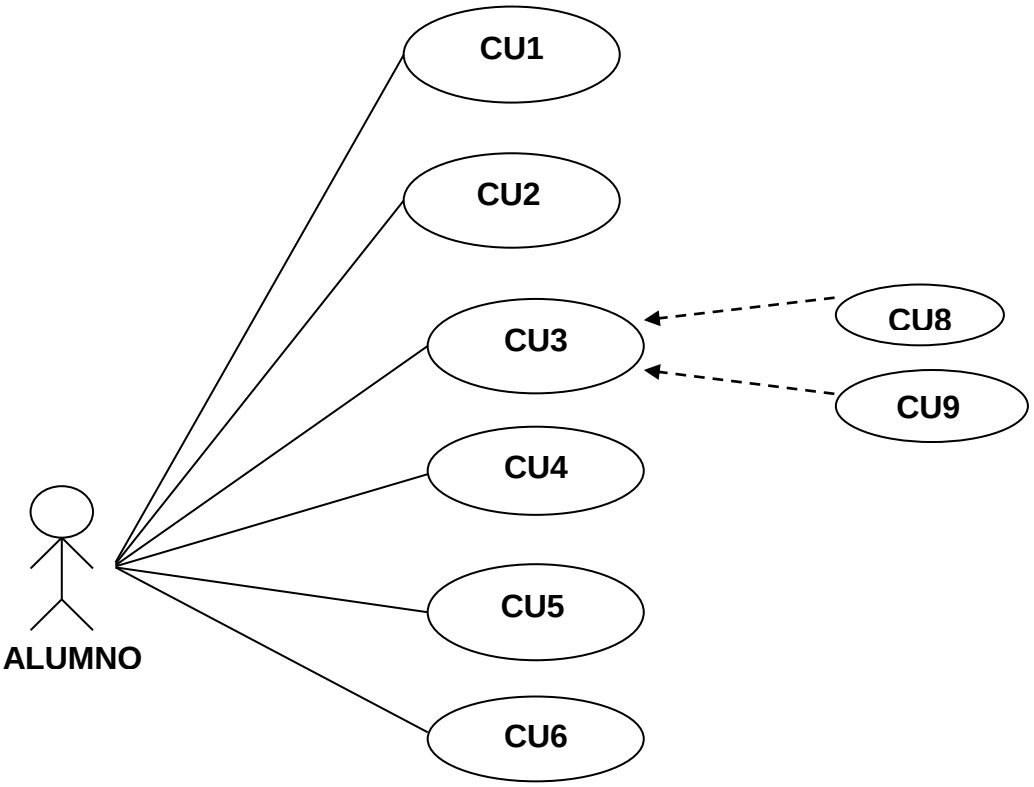


Figura 4. Diagrama Caso de Uso Detallado Alumno

Diagrama de Caso de Uso Detallado Administrador

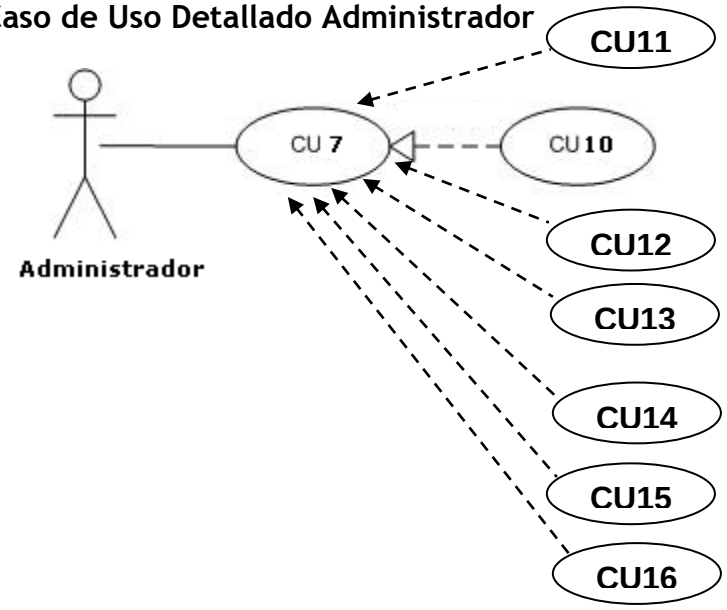


Figura 5. Diagrama Caso de Uso Detallado Administrador

Especificación De Caso De Uso

CASO DE USO: Seleccionar Teoría
DESCRIPCIÓN: El Actor Alumno puede escoger este caso de uso *"Seleccionar Teoría"* cuando se vea en la necesidad de estudiar o repasar los temas incluidos en el software Compisoft.
ACTOR PRINCIPAL: Alumno
PRECONDICION: Ninguna
GARANTÍA DE ÉXITO: El sistema muestra la información de la teoría

FLUJO BASICO: 1. El sistema muestra un submenú con las siguientes opciones (Definiciones básicas, gramáticas independientes del contexto, analizadores sintácticos, autómatas finitos determinista y no determinista) 2. El usuario elige la opción deseada. 3. El sistema examina la opción seleccionada. 4. El sistema carga la información sobre la opción seleccionada.
FLUJO ALTERNATIVO: ninguno

Tabla 5. Descripción de Caso de Uso Seleccionar Teoría

CASO DE USO: Seleccionar Ejemplos
DESCRIPCIÓN: El caso de uso "Seleccionar Ejemplos" puede ser escogido por el Actor Alumno cuando éste desee observar ejemplos relacionados con algunos de los temas descritos en el caso de uso "Seleccionar Teoría". El actor tiene la facultad de escoger entre los diferentes Ejemplos dados en Compisoft.
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: Que se tenga una buena comprensión de la parte teórica dada en la sección Teoría.

GARANTÍA DE ÉXITO: El sistema muestra los ejemplos de dos temas que hacen parte de la teoría.
FLUJO BASICO: 1. El sistema muestra las diferentes opciones. 2. El usuario elige la opción deseada (Ejemplos). 3. El sistema muestra las opciones. 4. El usuario elige la opción deseada. 5. El sistema identifica la opción seleccionada. 6. El sistema carga los ejemplos de la opción seleccionada.
FLUJO ALTERNATIVO: ninguno

Tabla 6. Descripción de Caso de Uso Seleccionar Ejemplos

CASO DE USO: Seleccionar Ejercicios
DESCRIPCIÓN: El caso de uso "Seleccionar Ejercicios" puede ser escogido por el Actor Alumno cuando éste desee realizar prácticas relacionadas con algunos de los temas descritos en el caso de uso "Seleccionar Teoría". El actor tiene la facultad de escoger entre los diferentes Ejercicios que muestra Compisoft.
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: Que se tenga una buena comprensión de la parte teórica dada en la sección Teoría.

GARANTÍA DE ÉXITO: El sistema muestra los ejercicios de algunos temas que hacen parte de la teoría.
FLUJO BASICO: 1. El sistema muestra las diferentes opciones. 2. El usuario elige la opción deseada (Ejercicios). 3. El sistema muestra diferentes opciones sobre los ejercicios de la teoría. 4. El usuario elige opción. 5. El sistema identifica la opción seleccionada. 6. El sistema carga los ejercicios de la opción seleccionada.
FLUJO ALTERNATIVO: ninguno

Tabla 7. Descripción de Caso de Uso Seleccionar Ejercicios

CASO DE USO: Seleccionar Software de aprendizaje Autómata Finito
DESCRIPCIÓN: El caso de uso "Seleccionar Auto visual" puede ser escogido por el Actor Alumno cuando desee realizar prácticas relacionadas con el tema de los autómatas finitos determinista y no determinista.
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: Que se tenga una buena captación y comprensión de la parte teórica dada en la sección Teoría.
GARANTÍA DE ÉXITO: El sistema carga la interfaz del programa Auto Visual.

FLUJO BASICO: 1. El sistema carga el programa. 2. El alumno realiza los pasos para hacer un autómata. 3. El sistema carga y muestra si lo realizado es o no un AFD O AFN.
FLUJO ALTERNATIVO: ninguno

Tabla 8. Descripción de Caso de Uso Seleccionar Software de aprendizaje Autómata Finito

CASO DE USO: Seleccionar Ejercicios de Auto evaluación
DESCRIPCIÓN: Este caso de uso "*Seleccionar Ejercicios*" será activado por el Actor Alumno para realizar las evaluaciones respectivas de los diferentes temas o la teoría establecida en este programa y así determinar su nivel académico.
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: El Actor Alumno haya pasado satisfactoriamente la parte de Teoría, al igual que la de Ejemplos. El tema del cual se va a resolver el examen se encuentre en el Compisoft.

GARANTÍA DE ÉXITO: El sistema muestra la evaluación de los temas que hacen parte de Compisoft.
FLUJO BASICO: 1. El sistema muestra las diferentes opciones. 2. El usuario elige la opción deseada. 3. El sistema genera las correspondientes preguntas. 4. El usuario responde. 5. El usuario selecciona ver resultados. 6. El sistema evalúa Respuestas. 7. El Sistema Muestra Resultados.
FLUJO ALTERNATIVO: 6.1 El sistema verifica si todas las preguntas están respondidas, si no es correcto, muestra mensaje y regresar a 4.

Tabla 9. Descripción de Caso de Uso Seleccionar Ejercicios de Auto evaluación

CASO DE USO: Seleccionar Videos
DESCRIPCIÓN: El caso de uso "Seleccionar Videos" puede ser escogido por el Actor Alumno cuando éste desee observar videos relacionados con algunos de los temas descritos en el caso de uso "Seleccionar Teoría". El actor tiene la facultad de escoger entre los diferentes videos dados por Compisoft.
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: Que se tenga una buena conexión a Internet.
GARANTÍA DE ÉXITO: El sistema muestra los videos de algunos temas que hacen parte de la teoría.

FLUJO BASICO: 1. El sistema muestra las diferentes opciones. 2. El usuario elige la opción deseada (Videos). 3. El sistema muestra las opciones. 4. El usuario elige la opción deseada. 5. El sistema identifica la opción seleccionada. 6. El sistema carga el video de la opción seleccionada.
FLUJO ALTERNATIVO: ninguno

Tabla 10. Descripción de Caso de Uso Seleccionar Videos

CASO DE USO: Seleccionar Ayuda
DESCRIPCIÓN: El caso de uso "Seleccionar Ayuda" puede ser escogido por el Actor Alumno cuando éste desee resolver alguna inquietud sobre el programa.
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: Ninguna
GARANTÍA DE ÉXITO: El sistema muestre de una manera clara los diferentes pasos que se pueden realizar en Compisoft en caso de alguna duda.

FLUJO BASICO: 1. El sistema muestra las diferentes opciones. 2. El usuario elige la opción deseada (Ayuda). 3. El sistema identifica la opción seleccionada. 4. El sistema carga el interfaz de la opción seleccionada.
FLUJO ALTERNATIVO: ninguno

Tabla 11. Descripción de Caso de Uso Seleccionar Ayuda

CASO DE USO: Seleccionar Créditos
DESCRIPCIÓN: El caso de uso "Seleccionar Créditos" puede ser escogido por el Actor Alumno el cual mostrara los autores de Compisoft
ACTOR PRINCIPAL: Alumno
PRECONDICIÓN: Ninguna
GARANTÍA DE ÉXITO: El sistema los autores de Compisoft
FLUJO BASICO: 1. El sistema muestra las diferentes opciones. 2. El usuario elige la opción deseada (Créditos). 3. El sistema identifica la opción seleccionada. 4. El sistema carga la interfase de la opción seleccionada.
FLUJO ALTERNATIVO: ninguno

Tabla 12. Descripción de Caso de Uso Seleccionar Créditos

CASO DE USO*:* Admin.
DESCRIPCIÓN: Este caso de uso *"admin."* es escogido por el Actor Administrador cuando necesite ingresar las preguntas que hacen o harán parte de la sección de evaluación.
ACTOR PRINCIPAL: Administrador
PRECONDICIÓN: Ninguna
GARANTÍA DE ÉXITO: El sistema muestra la interfase donde se a actualizar la evaluación.
FLUJO BASICO: 1. El sistema carga la interfase de actualización. 2. El Administrador ingresa la contraseña de ingreso. 3. El sistema carga otra interfase donde se va a agregar la(s) pregunta(s) del test evaluativo.
FLUJO ALTERNATIVO: 2.1 El sistema verifica la contraseña, si ésta es invalida mostrar mensaje, ir a 2

Tabla 13. Descripción de Caso de Uso Seleccionar Actualizador de Preguntas

CASO DE USO: Agregar Pregunta
DESCRIPCIÓN: Este caso de uso *"Agregar Pregunta"* es escogido por el Actor Administrador cuando necesite Agregar preguntas al test de Compisoft.
ACTOR PRINCIPAL: Administrador
PRECONDICIÓN: Ninguna
GARANTÍA DE ÉXITO: El sistema guarda en la base de datos la nueva pregunta agregada.
FLUJO BASICO: 1. El sistema carga el escenario. 2. El Administrador selecciona el grupo al cual agregara las preguntas 3. El Administrador ingresa las nuevas preguntas y sus respuestas. 4. El administrador asigna el grupo encargado de responder esas preguntas. 5. El sistema guardar en una base de datos las preguntas nuevas.
FLUJO ALTERNATIVO: 3.1 Si una de las casillas de preguntas y/o respuesta no es llenada, mostrar mensaje y enviar a 2.

Tabla 14. Descripción de Caso de Uso Seleccionar Agregar Preguntas

Diagramas de Secuencia

Los diagramas de secuencia representan la forma de como estos objetos interactúan entre sí y que muestra de forma detallada todas aquellas operaciones que se realizan, los mensajes enviados y en qué momento. Pero lo más importante del diagrama de secuencia es que todo está en función del tiempo. En este diagrama el tiempo avanza "hacia abajo" y los objetos que intervienen se colocan de izquierda a derecha todo depende del orden en que participan.

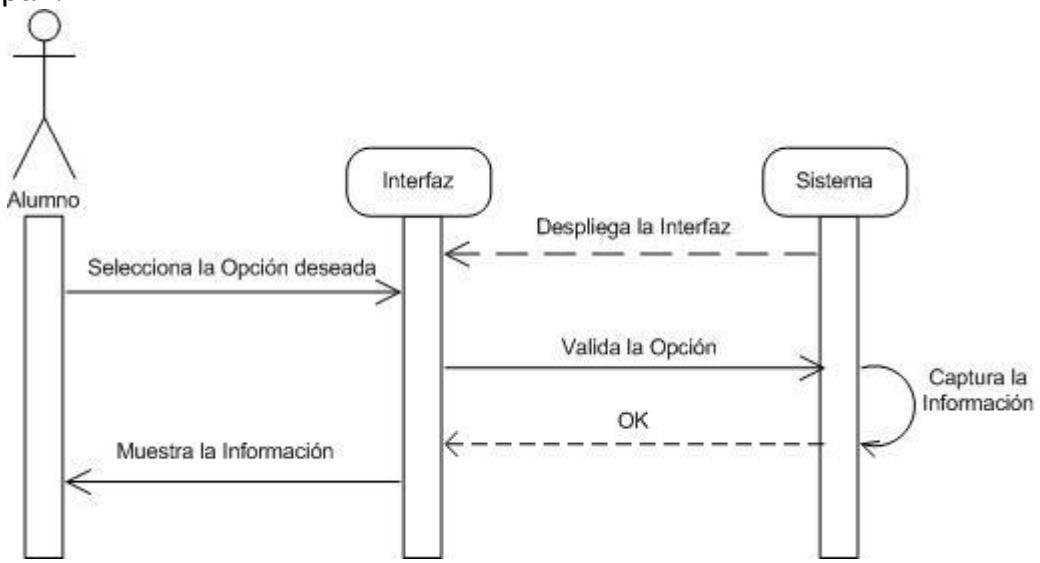


Figura 6. Flujo para Seleccionar Teoría.

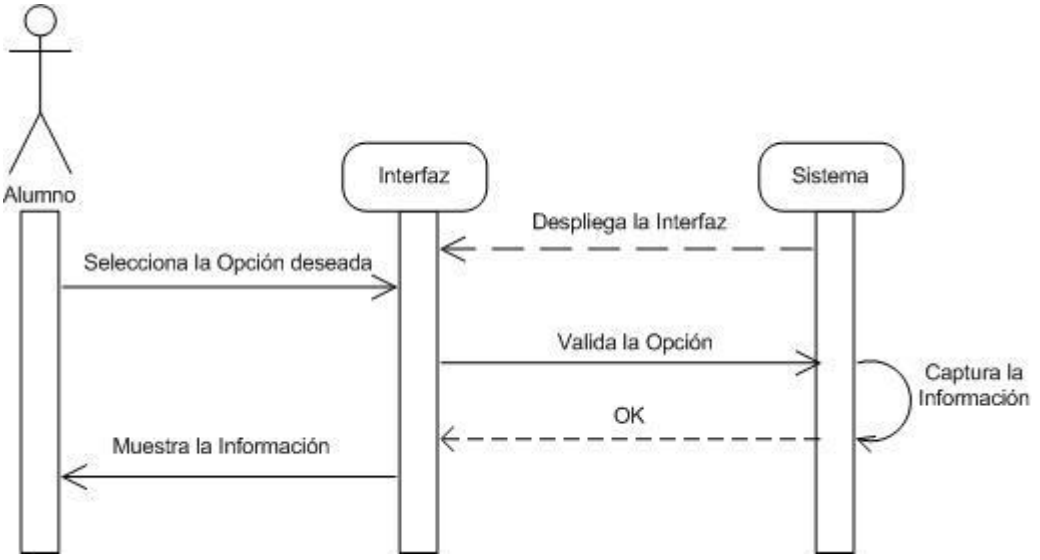


Figura 7. Flujo para Seleccionar Ejemplos.

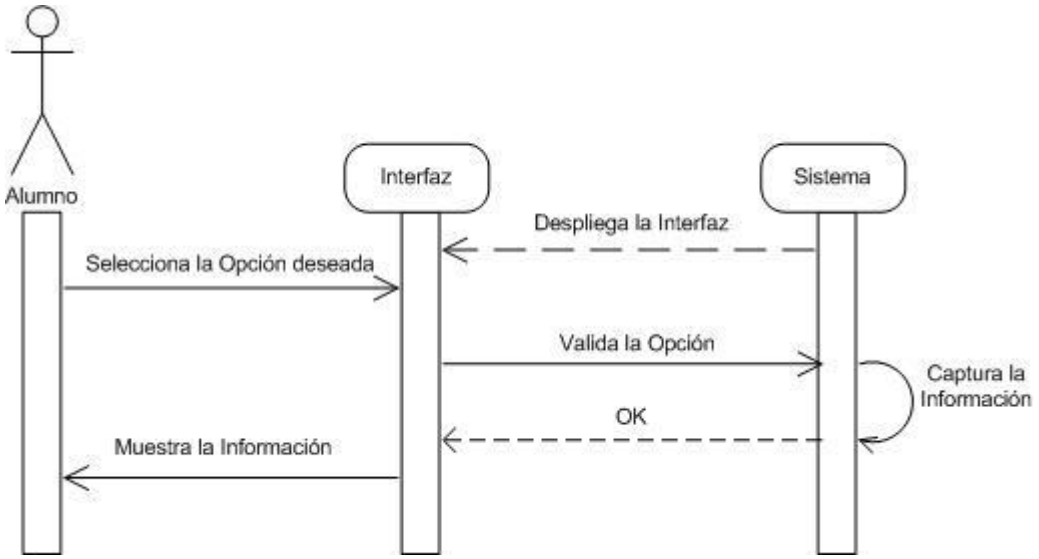


Figura 8. Flujo para Seleccionar Ejercicios.

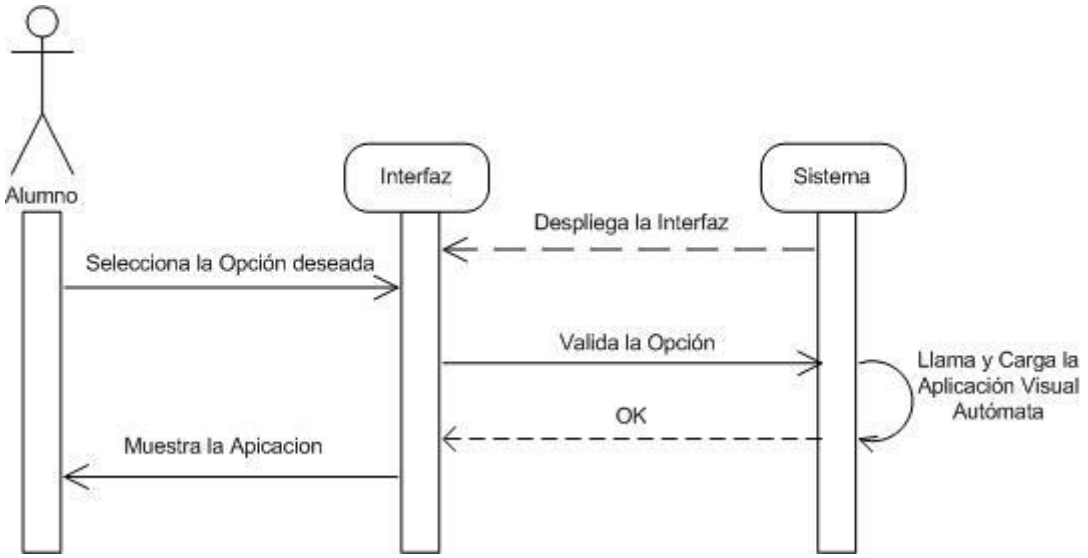


Figura 9. Flujo para Seleccionar Software de aprendizaje Autómata Finito

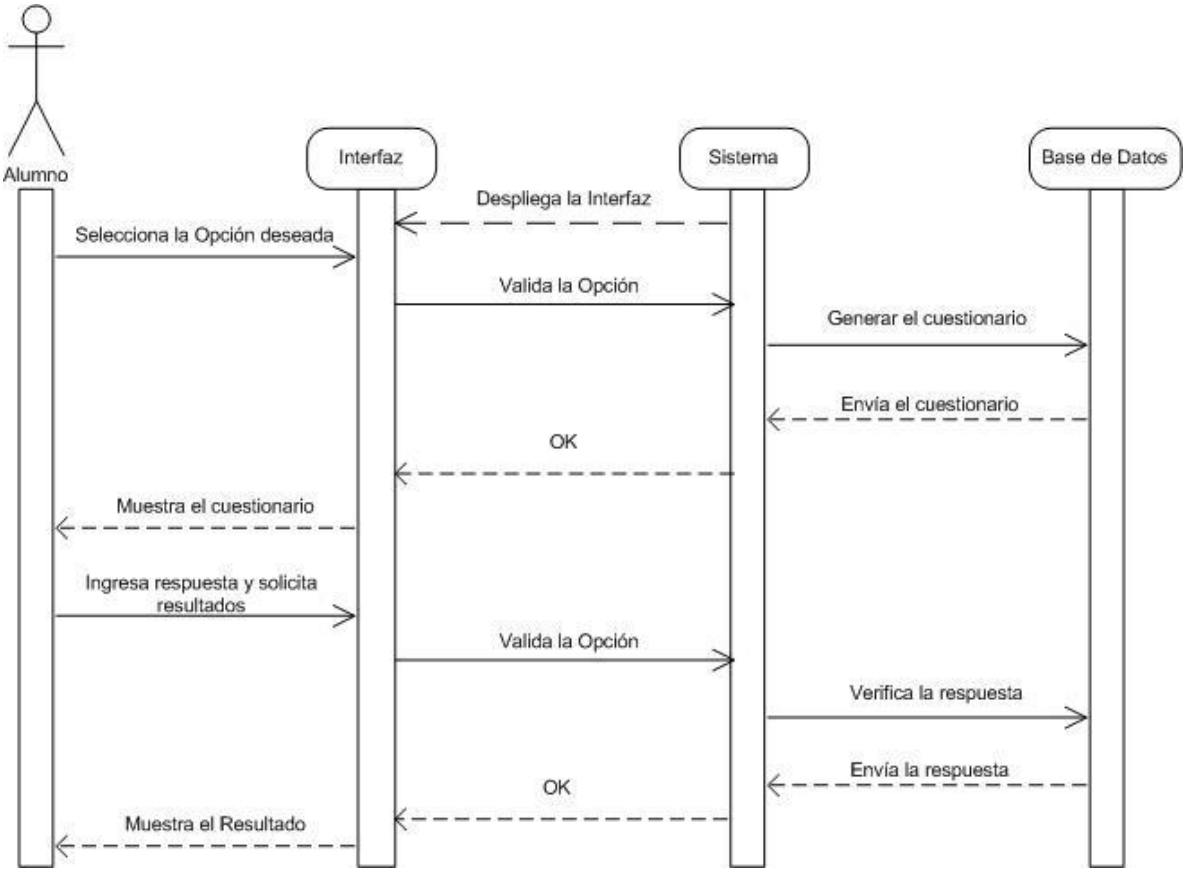


Figura 10. Flujo para Seleccionar Auto Evaluación.

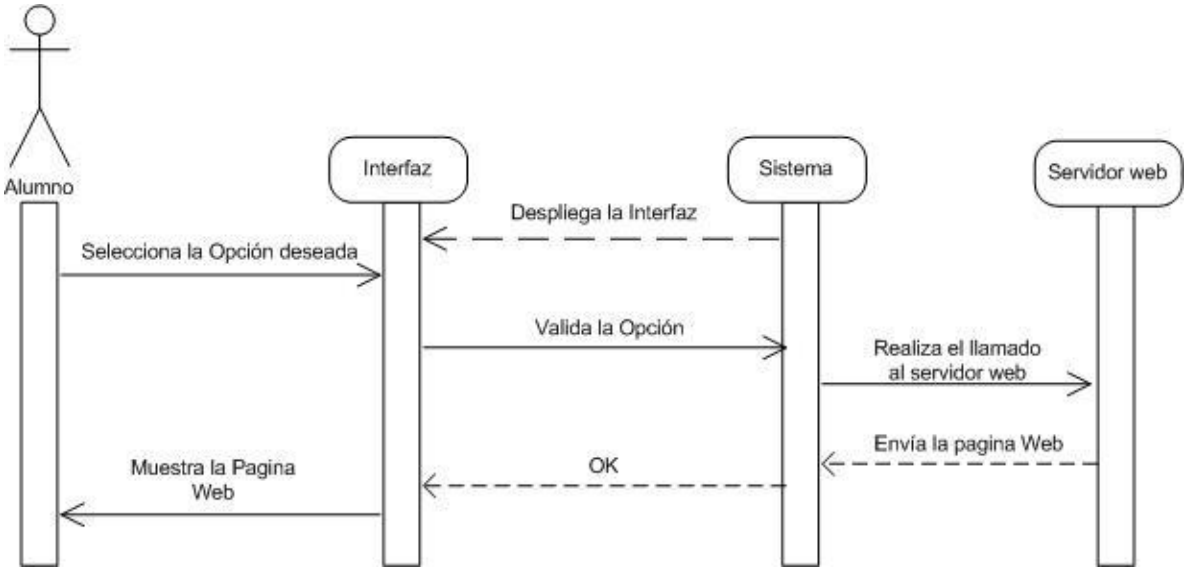


Figura 11. Flujo para Seleccionar Videos.

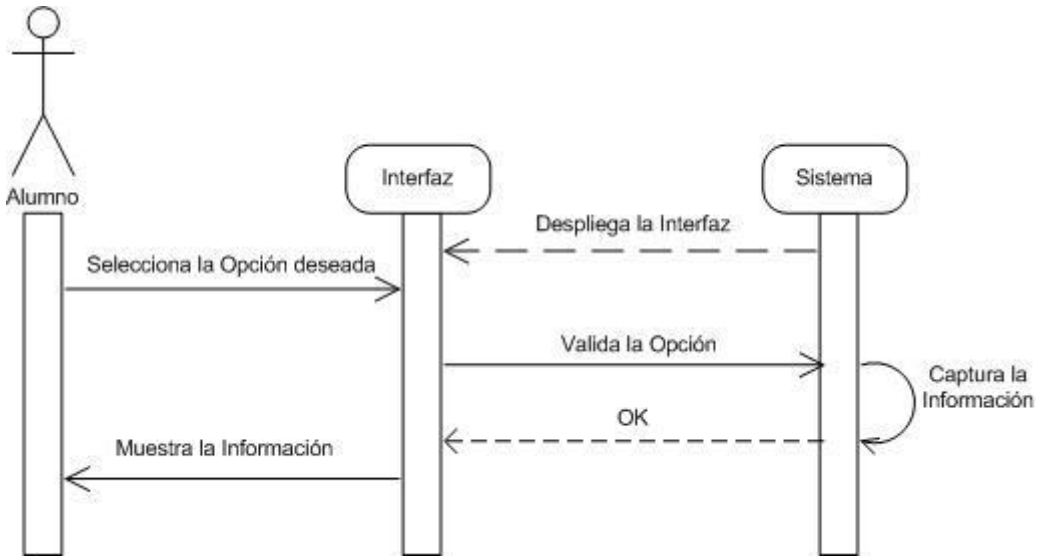


Figura 12. Flujo para Seleccionar Ayuda.

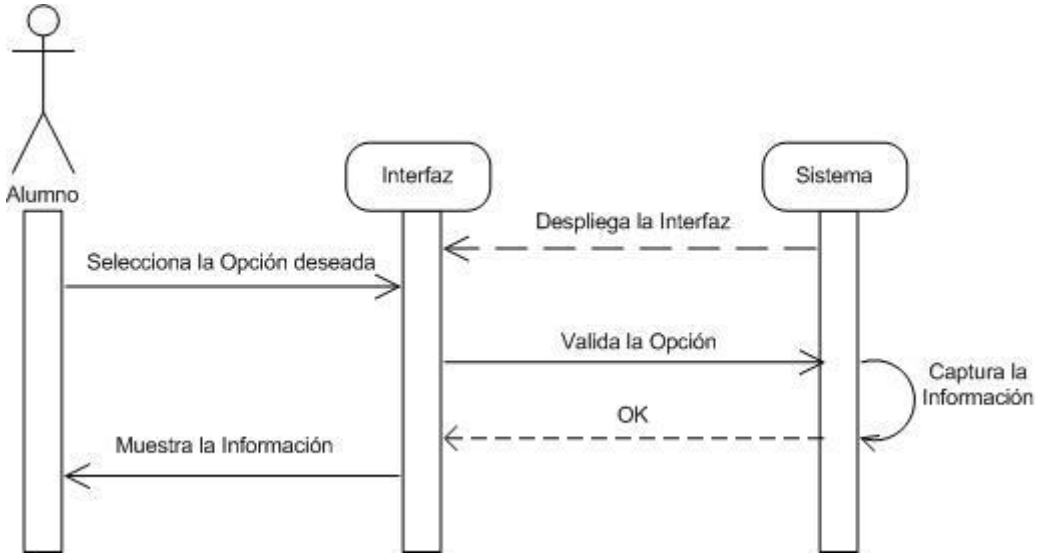


Figura 13. Flujo para Seleccionar Créditos.

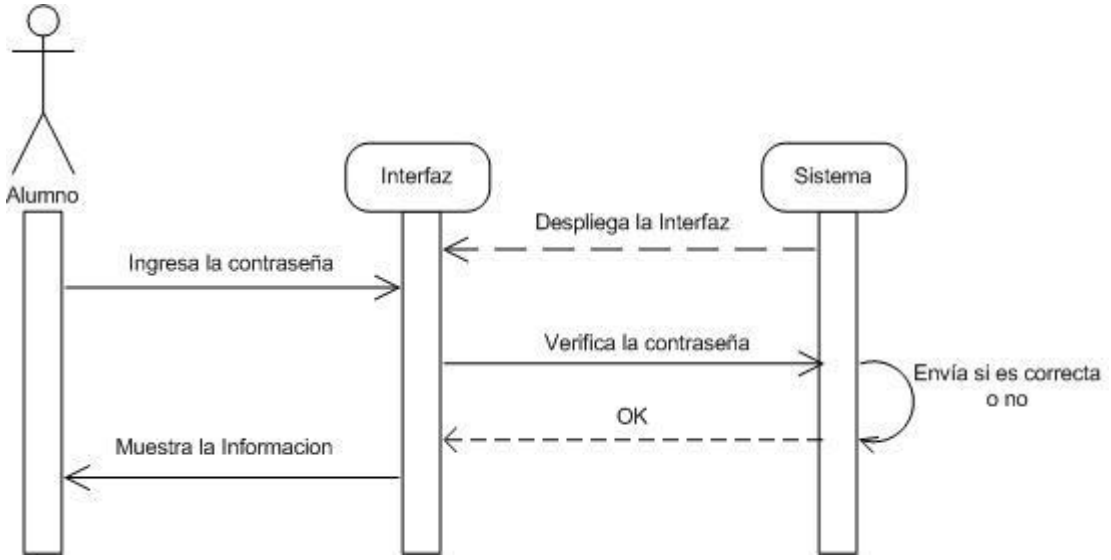


Figura 14. Flujo para Seleccionar Admin.

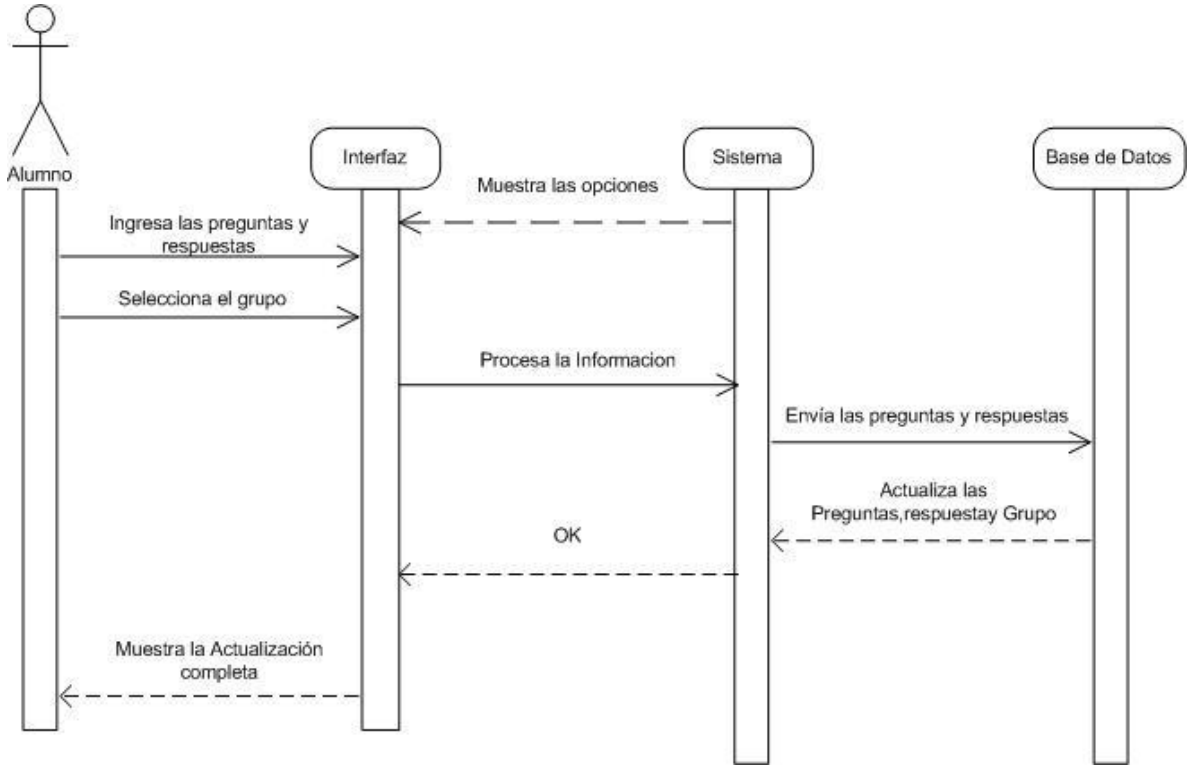


Figura 15. Flujo para Seleccionar Agregar peguntas.

Clases con sus atributos y propiedades

Teoría
-Definiciones básicas -Gramáticas Independientes del contexto -Analizadores Sintácticos -Afd y Afnd
- Mostrar opciones -Escoger opciones -Atrás

Tabla 15. Clase Teoría

Ejemplos
-gramáticas Independientes del Contexto -Expresiones Regulares -Autómatas Finitos
- Mostrar opciones -Escoger opciones -Atrás

Tabla 16. Clase Ejemplos

Pregunta
-Descripción -Respuesta correcta -Respuesta usuario
-Mostrar -Verificar respuesta -Mostrar respuesta correcta

Tabla 17. Clase Pregunta

Administración
-Contraseña -Pregunta -Respuesta -Grupo
-Verificar Contraseña -Modificar Preguntas -Modificar Respuestas -Limpiar Preguntas -Limpiar Respuestas -Salir

Tabla 18. Clase Administración

Cuestionarios
-Preguntas -Respuestas -Colección de aciertos -Desaciertos
-Mostrar consulta -Ocultar consulta -Calcular aciertos -Calcular desaciertos

Tabla 19. Clase Cuestionario

Mapa de Navegabilidad

El mapa que se muestra a continuación describe como el usuario se desplazará a través del software Compisoft y lo que encontrará durante todo el recorrido. Al igual esta grafica describe exactamente los temas y puntos que se desarrollan y encontraran dentro del software.

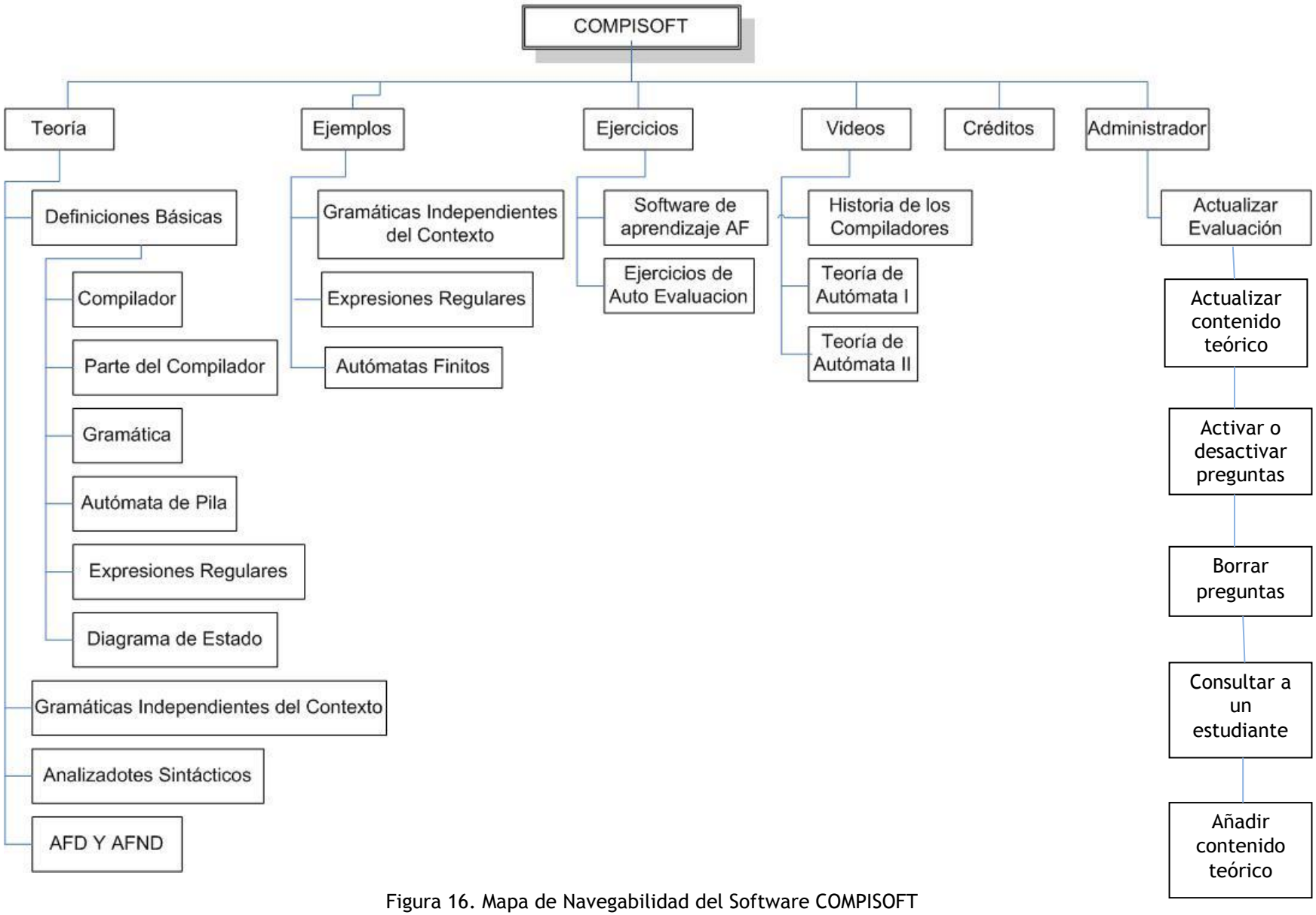


Figura 16. Mapa de Navegabilidad del Software COMPISOFT

Diagrama de Componentes

"Un diagrama de componentes muestra la organización y las dependencias entre un conjunto de componentes. Los diagramas de componentes cubren la vista de implementación estática de un sistema"[18] .

El Diagrama de Componentes se usa para modelar la estructura del software, incluyendo las dependencias entre los componentes de software, los componentes de código binario, y los componentes ejecutables. En el Diagrama de Componentes se modelan componentes del sistema, a veces agrupados por paquetes, y las dependencias que existen entre componentes (y paquetes de componentes).

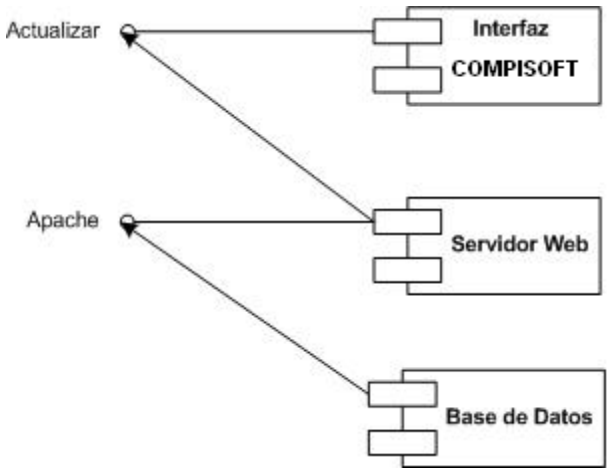


Figura 17. Diagrama de Componentes General

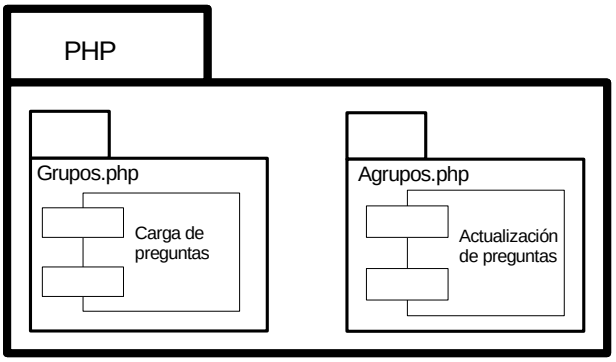


Figura 18. Paquete PHP

[18] PRESSMAN, Roger. Ingeniería Del Software, Un Enfoque Práctico. 5ta edición. Mc Graw Hill. Ingeniería del software. Páginas. P. 75. Página. 22

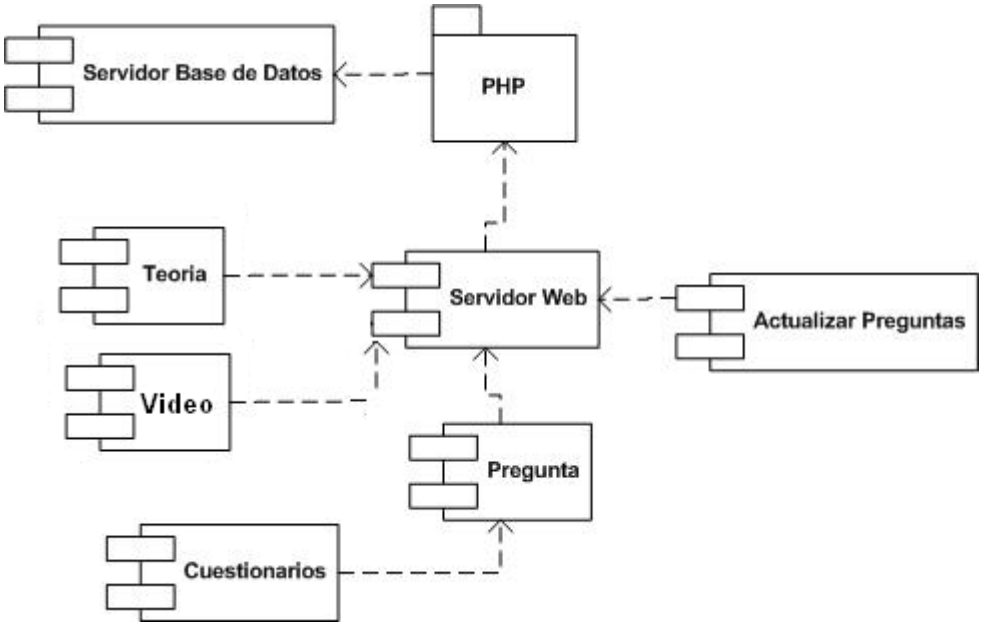


Figura 19. Diagrama de Componente Detallado

Diagrama de Despliegue

Los diagramas de despliegue son utilizados para visualizar y especificar sistemas clientes-servidor, además aparecen cuando se modelan los aspectos físicos de los sistemas orientados a objetos. Un diagrama de despliegue muestra la configuración de nodos que actúan en la ejecución y de los componentes que se encuentran en ellos.

La Figura 18 muestra los elementos que intervienen en este diagrama dados en esta aplicación. Como podemos observar, existe un nodo llamado ***USUARIO*** en el que éste a través del browser o navegador realiza una petición al servidor utilizando una sentencia HTML. El servidor a su vez realiza un llamado a la aplicación de nombre y extensión ***.swf**, en el cual se encuentran las diferentes interfases y códigos de la aplicación. El paquete **Aplicación** contiene los nodos Teoría, Ayuda, Videos, Evaluación y Actualizar Pregunta, en donde los nodos **Evaluación** y **Actualizar Preguntas** realizan la ejecución de la base de datos a través de un código php. Para finalizar todas las interfases son ejecutadas como respuesta del servidor APACHE en el navegador del usuario.

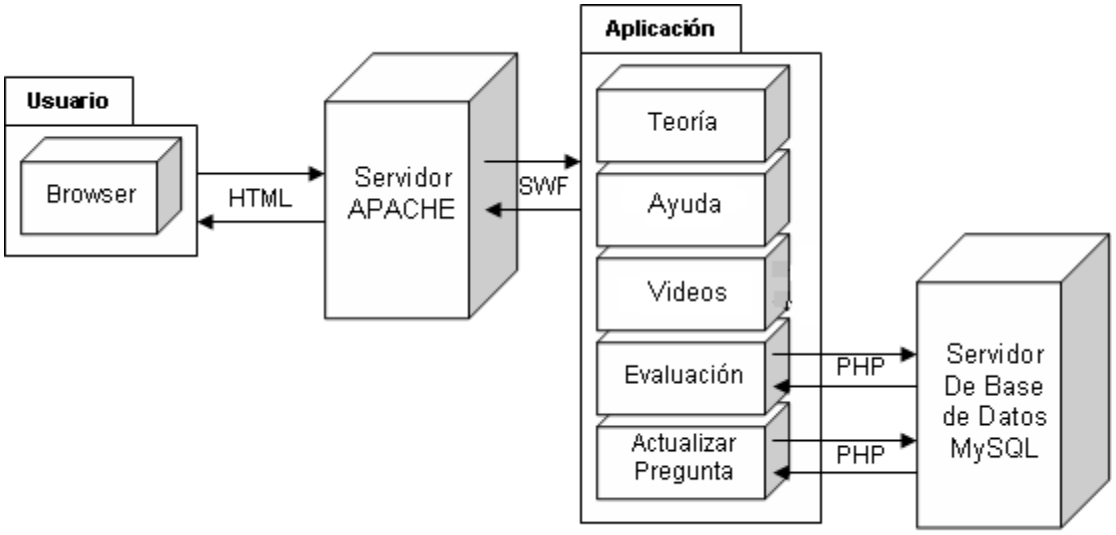


Figura 20. Diagrama de Despliegue

Modo de Relaciones

En la tabla creada en el software MySQL la cual se observa en la Figura 19 , es utilizada para almacenar las preguntas que el estudiante resolverá en la sección Auto Evaluación , es decir para todos los temas en la parte teórica existe una base de datos la cual contiene una tabla que controla las preguntas de dicho capitulo. En la grafica siguiente se puede ver los campos; tipos de datos y otros contenidos en la tabla.

	Campo	Tipo	Collation	Atributos	Nulo	Predeterminado	Extra	Acción
	numero	int(2)			No		auto_increment	
	pregunta	mediumtext	utf8_spanish_ci		No			
	respuesta	char(1)	utf8_spanish_ci		No			

Figura 21 Tabla Pregunta de la Base de Datos Cuestionario

La tabla consta de 3 campos: ***numero*** (identificación de la pregunta, es la llave primaria), ***pregunta*** (descripción o almacenamiento de la pregunta a formular) y por último ***respuesta*** (es la respuesta correcta a la pregunta formulada).

4.4.2 Diseño de Interfaz.

El diseño de Interfaz de Usuario, de un programa es un conjunto de elementos hardware y software de una computadora que presentan información al usuario y le permiten interactuar con la información y con el computadora.
A continuación se describe paso a paso la interacción entre los usuarios finales y el sistema, a través de ventanas que explican las funciones y opciones del software de manera detallada.
Durante el proceso de realización de la aplicación fue asignado el nombre "COMPISOFT" el cual significara Software de Compiladores.

Para que la aplicación pueda observarse en la computadora se debe tener en cuenta una serie de archivos y que se deben copiar en el servidor Web, es decir el equipo donde se va a alojar el software.

De tal forma es importante copiar de forma correcta todos estos archivos, ya que de ellos depende el buen funcionamiento y ejecución del software, para una mejor explicación usted debe dirigirse al Manual de Usuario.

Para las estaciones de trabajos (clientes), se debe tener instalado el programa Flash Player 8.0

Pantalla principal de COMPISOFT



Figura 22. Interfaz Pantalla Inicial

Al Iniciar el software muestra la pantalla anterior (Figura 20). Aquí el usuario encontrará las diferentes opciones como Teoría, Ejemplos, Ejercicios, Videos, Ayuda, Créditos, Admin, de las cuales el usuario podrá elegir para trabajar.
Cada uno de los botones se encuentran de forma visual y el usuario no encontrará ningún impedimento para manipularlo.

Cabe resaltar que en la opción "ADMIN", le permitirá agregar preguntas para las Auto evaluaciones. Este botón solo es utilizado por el Administrador del Software.



Figura 23. Interfaz de Teoría

El botón "Teoría" despliega una serie de opciones donde el usuario podrá escoger entre cualquiera de las diferentes teorías que conforman la aplicación COMPISOFT como se muestra en la figura anterior. Se le recomienda cerrar la ventana ya que no tiene ningún botón realizar esta operación.

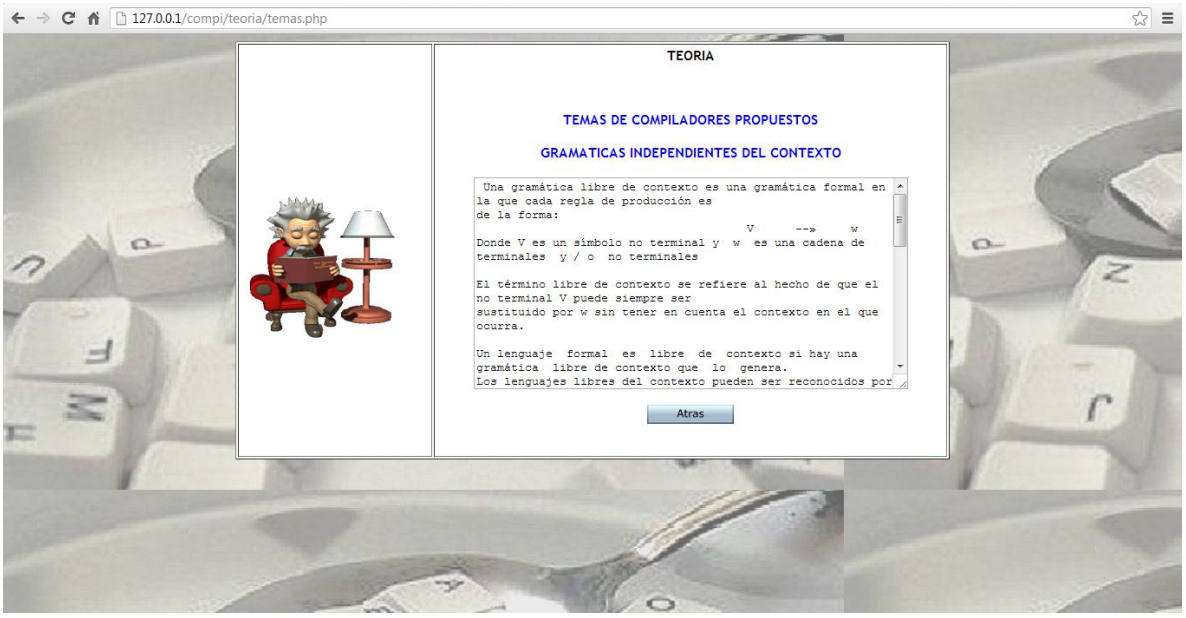


Figura 24. Interfaz de Definiciones de los temas

Esta interfaz se activa cuando el usuario escoge la opción “Seleccionar” la cual despliega otra serie de opciones donde el usuario podrá escoger entre cualquiera de estos conceptos que son de gran importancia antes de entrar de lleno a cada uno de los temas que conforman la aplicación COMPISOFT como se muestra en la figura anterior.



Figura 26. Interfaz de Ejemplos

Titulo de opción escogida.
Botón para desplazarse en este caso muestra la pantalla de la Figura 20.
Opciones que deseo o puedo escoger.

El botón ejemplos muestra la interfaz de la figura anterior, la cual despliega una serie de opciones donde el usuario podrá escoger la que mas desee y así mostrar los ejemplos de los temas enmarcados en esta aplicación.

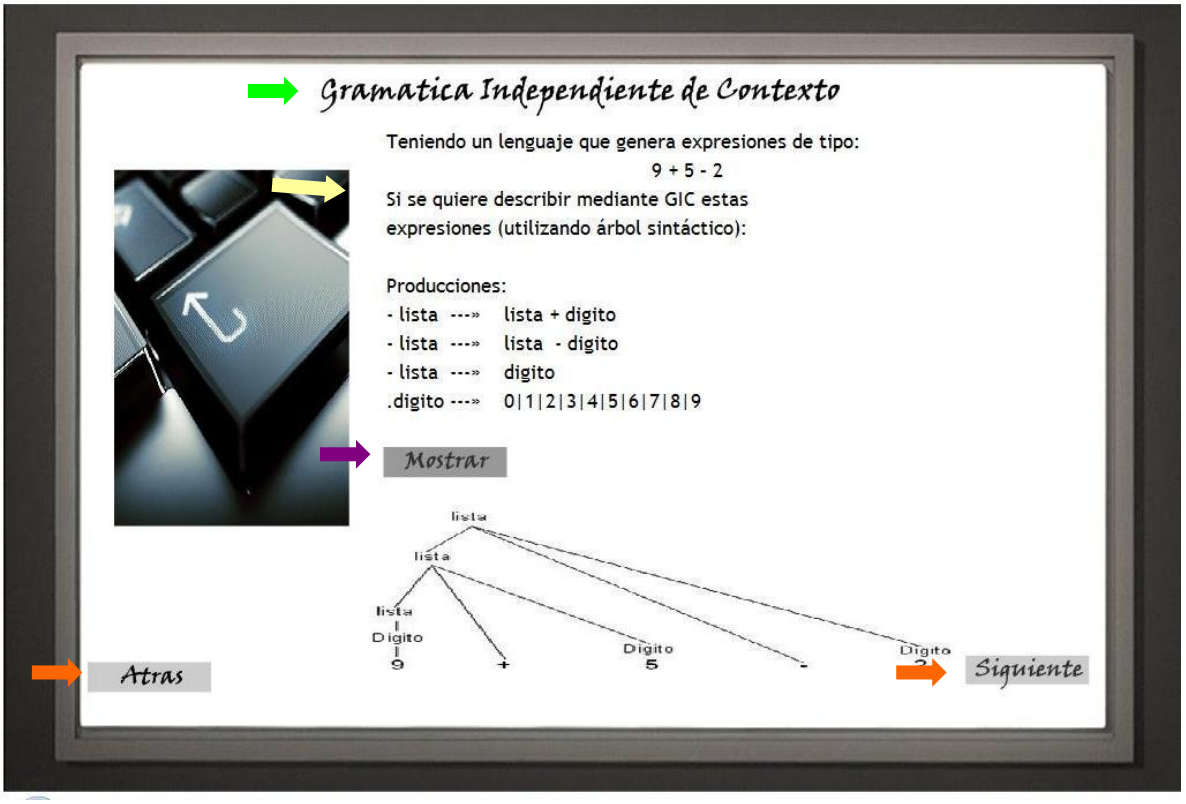


Figura 27. Interfaz de Gramáticas Independientes del Contexto (Ejemplos)

Titulo del tema al cual sale el ejemplo.
Enunciado del ejemplo.
Botón mostrar, al darle clic me muestra el árbol de manera detallada.
Botón atrás o siguiente que me permite moverme en las diferentes interfases de ejemplos.

La Interfaz anterior se activa al escoger una de las opciones del botón Ejemplos, esta muestra de manera detallada un ejemplo del tema escogido en este caso el de gramática.



Figura 28. Interfaz de Ejercicios

Titulo de la opción escogida.
Botón de Inicio, el cual me muestra la pantalla inicial de la aplicación
Las 2 Opciones que puedes escoger.

Al Iniciar el software muestra la interfaz de la Figura 20 si se escoge la opción Ejercicios muestras la interfaz anterior la cual tiene 2 opciones de donde el usuario toma la decisión de cual quiere escoger.

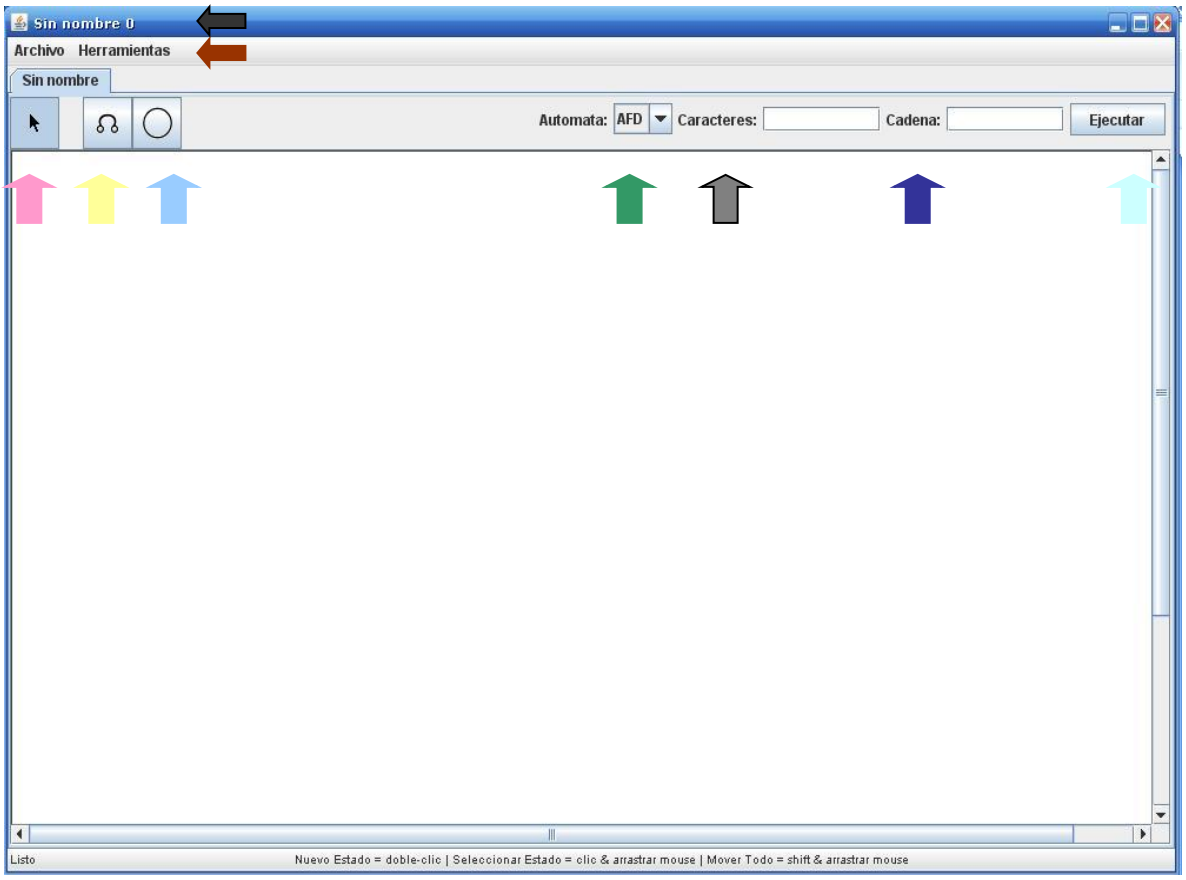


Figura 29. Interfaz de Software de aprendizaje Autómata Finito

Activa el puntero del Mouse.

Con esta opción realizas Aristas.

Con esta opción realizas los Nodos.

Aquí puedes escoger la opción de convertirlo en autómata finito determinista o no determinista.

Agregar la cadena

Agregar el carácter

Verifica que la cadena digitada pertenezca al AFD OAFND

Barra de menús.

Nombre del ejercicio realizado

En la figura anterior representa la interfaz del Software de aprendizaje Autómata Finito. En esta aplicación el usuario agregara nodos aristas para realizar ejercicios de manera grafica, además digitar la cadena y los caracteres para luego al dar clic en ejecutar, usuario visualizará si la cadena digitada correspondiente o no al Autómata Finito Determinista o No Determinista.

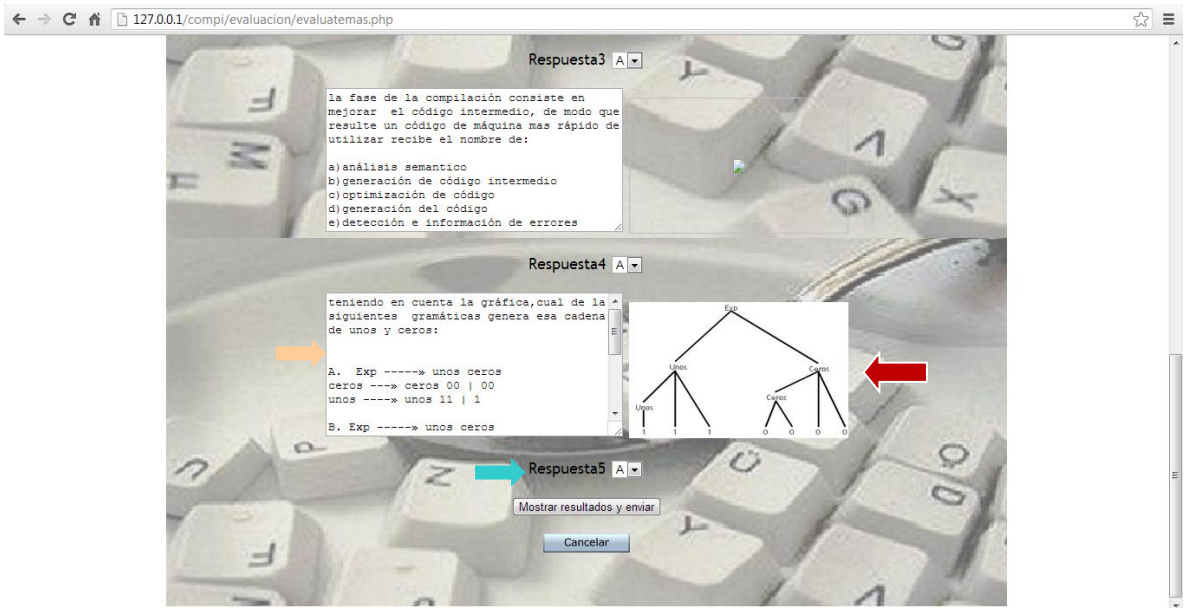


Figura 30. Interfaz de Autoevaluación

- Preguntas de la Auto evaluación.
- En este campo escribe la respuesta que considera correcta.
- Grafica dela pregunta en caso de que tenga alguna

En esta interfaz (Figura 30) aparecen, 5 preguntas en total y sus casillas de respuesta, en donde el estudiante colocará la respuesta que él considere correcta.
Cuando esté seguro de haber respondido todas las preguntas haga clic en el botón Resultados, que se encuentra en la parte inferior de la pantalla.



Figura 32. Interfaz Video

➡ Titulo de la opción escogida.
➡ Diferentes opciones que puedes escoger.
➡ Botón Atrás se utiliza para desplazarse en la aplicación en este caso a la interfaz principal de Compisoft.

Esta figura muestra los tres videos que se pueden observar, teniendo en cuenta que necesitas Internet para tener acceso a ellas.

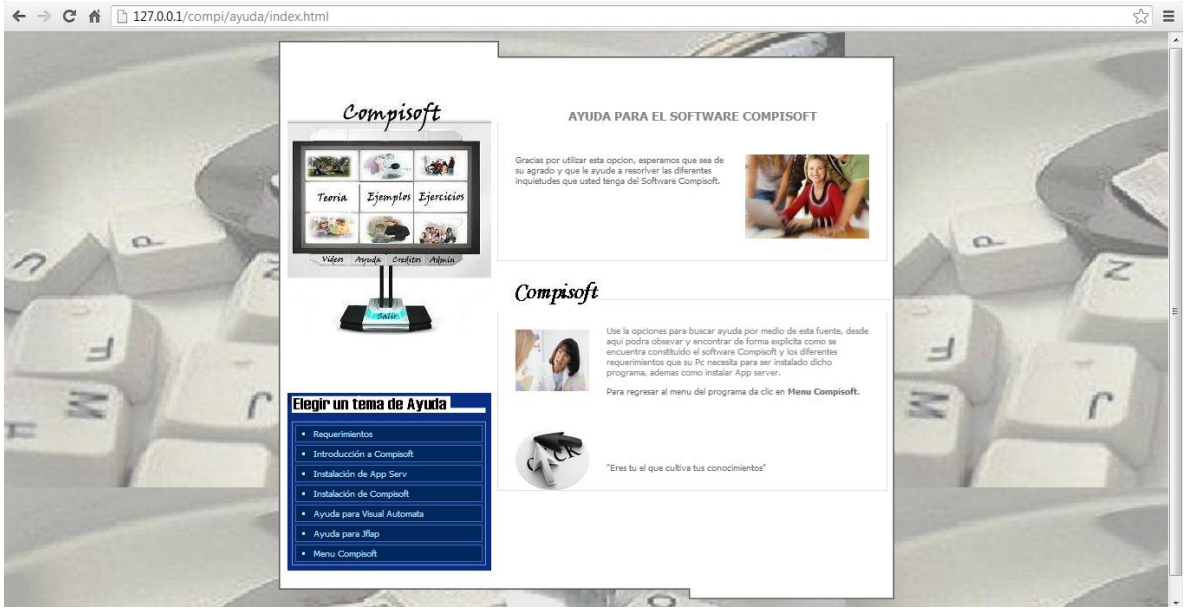


Figura 33. Interfaz Ayuda

La figura anterior muestra de forma clara todos los pasos e inquietudes que el usuario pueda tener al utilizar la aplicación.



Figura 34. Interfaz Créditos

- Titulo de la opción escogida.
- Botón Atrás se utiliza para desplazarse en este caso a la interfaz Principal de Compisoft

La figura anterior muestra las personas que hicieron parte de este proyecto.

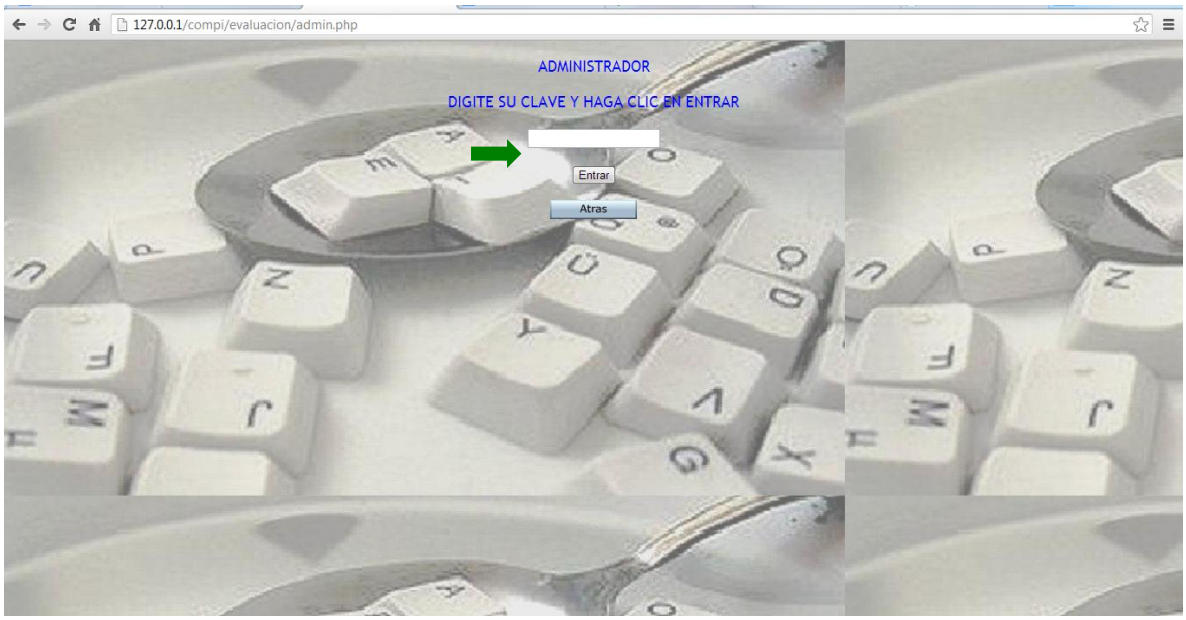


Figura 35. Interfaz Admin

- Opción para ingresar la clave o contraseña.

En la siguiente interfaz el docente que en este caso es el Administrador debe digitar la clave para ingresar a todas las opciones que el tutor tiene para manejo de la preguntas, contenido teórico y consulta de estudiantes etc. después que ingrese la clave debe dar clic en aceptar para que verifique si es o no la correcta.

4.5 FASE DE EVALUACIÓN DEL USUARIO

El objetivo general de este proyecto es realizar un Software que sirva de apoyo para el aprendizaje de la materia compiladores. Por tal razón para verificar que la aplicación no tuviera errores a nivel de programación y que cumpliera con todas funciones y requerimientos propuestos en dicho proyecto, se realizaron las pruebas de campo a estudiantes.

Esta prueba se realizó el día 25 de abril de 2008 a continuación, se hará el análisis de cada uno de los resultados obtenidos en la prueba.

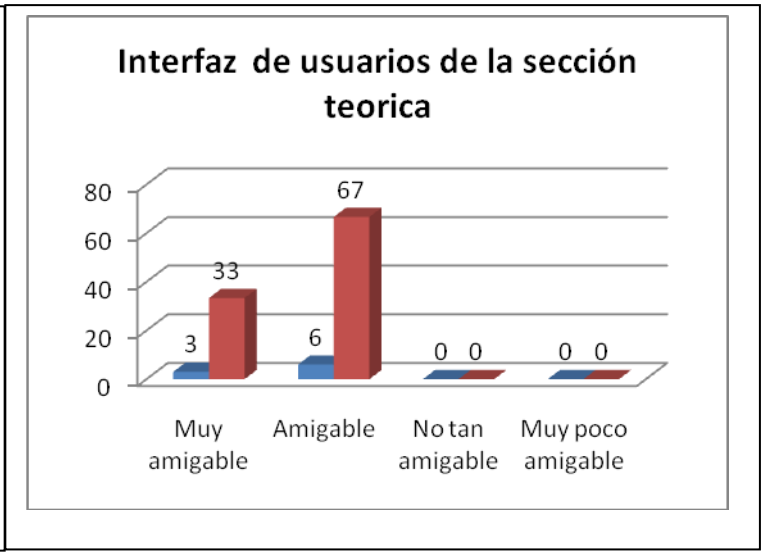
4.5.1 Prueba de campo. El objetivo de esta encuesta se basa en comprobar que COMPISOFT cumpla con todos los requerimientos solicitados tales como una interfase, navegabilidad, y por último su contenido. Esta encuesta fue realizada a un grupo de 9 estudiantes de décimo semestre de la Universidad San buenaventura Cartagena, conocedores ya de esta asignatura.

A continuación se hará una tabulación y análisis de los datos obtenidos en la encuesta.

Análisis Prueba de Campo

1. Cree usted que la interfaz de usuario de la sección teórica es:

	CANTIDAD	%
Muy amigable	3	33
Amigable	6	67
No tan amigable	0	0
Muy poco amigable	0	0
Total	9	100

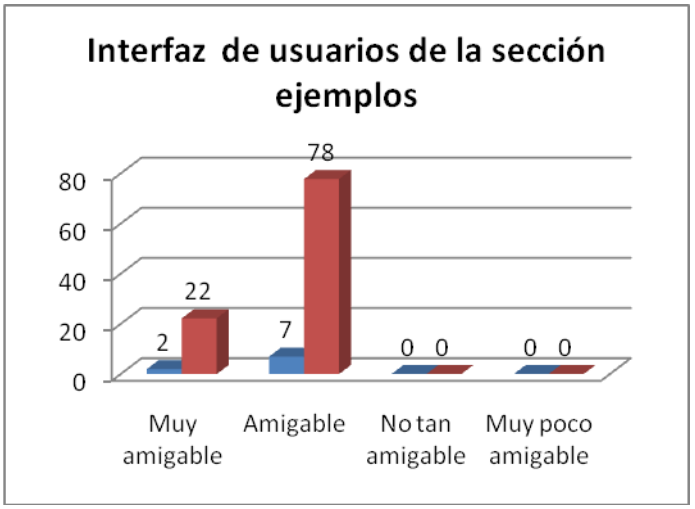


Grafica 1. Interfaz de usuarios sección teórica

Los resultados demuestran que la interfaz de esta sección es agradable a la vista del usuario.

2. Cree Usted que la Interfaz de usuario de la sección de ejemplos es:

	CANTIDAD	%
Muy amigable	2	22
Amigable	7	78
No tan amigable	0	0
Muy poco amigable	0	0
Total	9	100

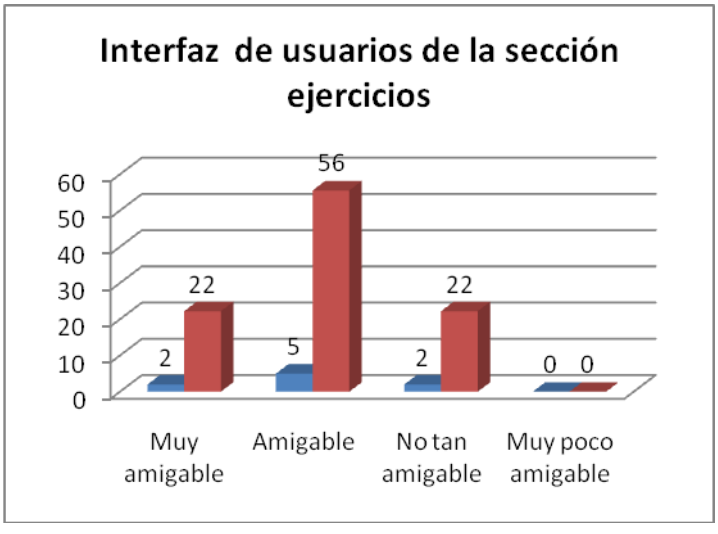


Grafica 2. Interfaz de usuarios sección ejemplos

Los resultados muestran que la interfaz de la sección ejemplos es agradable a la vista del usuario.

3. Cree Usted que la Interfaz de usuario de la sección de ejercicios es:

	CANTIDAD	%
Muy amigable	2	22
Amigable	5	56
No tan amigable	2	22
Muy poco amigable	0	0
Total	9	100

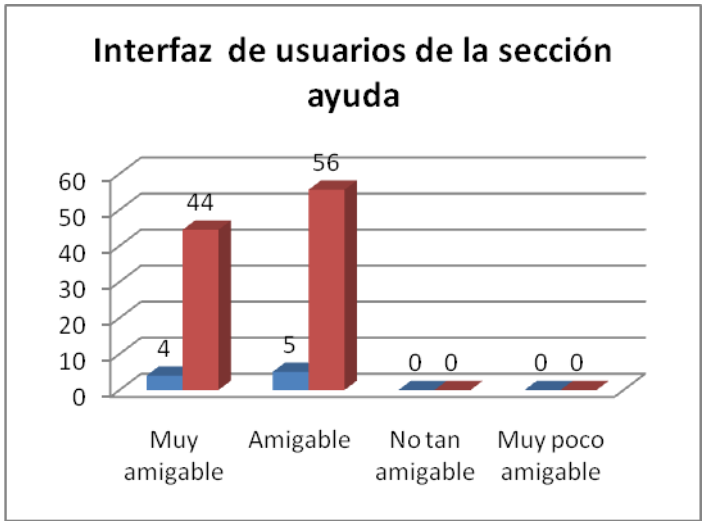


Grafica 3. Interfaz de usuarios sección ejercicios

Los resultados muestran que la interfaz de la sección ejercicios es agradable a la vista del usuario.

4. Cree Usted que la Interfaz de usuario de la sección de ayuda es:

	CANTIDAD	%
Muy amigable	4	44
Amigable	5	56
No tan amigable	0	0
Muy poco amigable	0	0
Total	9	100

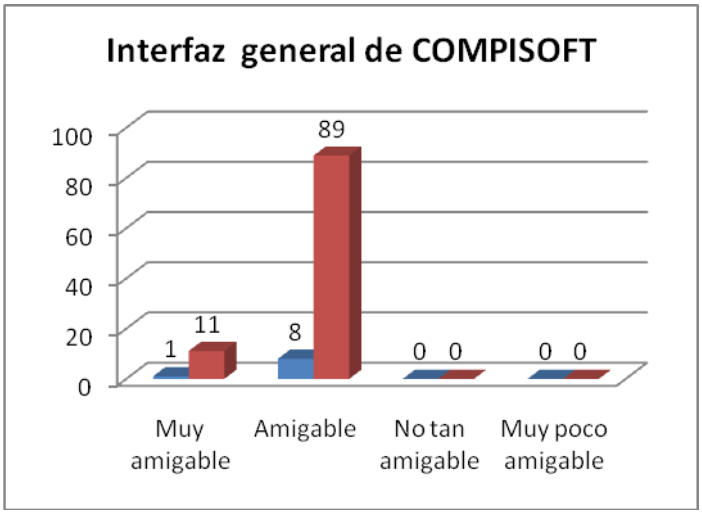


Grafica 4. Interfaz de usuarios sección Ayuda

Los resultados muestran que la interfaz de la sección ejercicios es agradable a la vista del usuario.

5. De forma general cree usted que la Interfaz de COMPISOFT es:

	CANTIDAD	%
Muy amigable	1	11
Amigable	8	89
No tan amigable	0	0
Muy poco amigable	0	0
Total	9	100

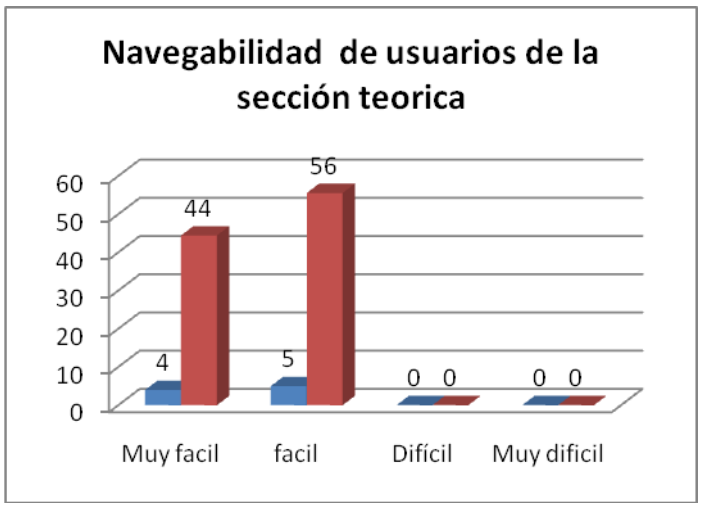


Grafica 5. Interfaz general Compisoft

Los datos muestran que la interfaz general de COMPISOFT es agradable a la vista del usuario.

6. Cree usted que la Navegabilidad de usuario de la sección teórica es:

	CANTIDAD	%
Muy fácil	4	44
Fácil	5	56
Difícil	0	0
Muy difícil	0	0
Total	9	100

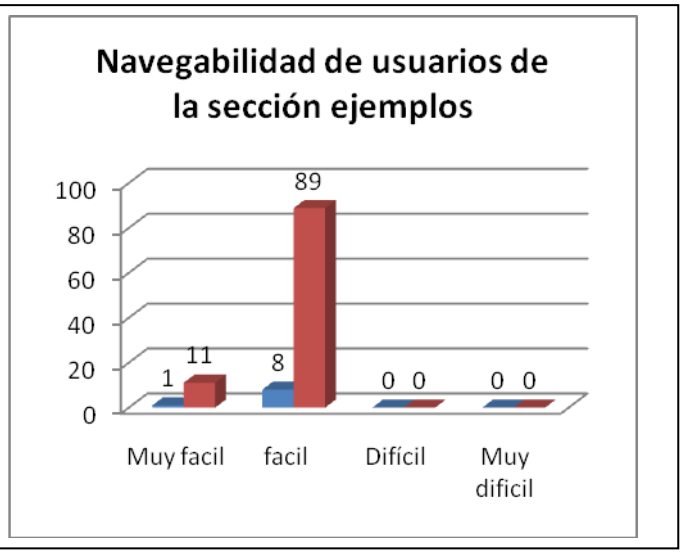


Grafica 6. Navegabilidad de usuarios sección teórica

Los datos muestran que la navegabilidad de usuario de la sección teórica es fácil al usuario

7. Cree Usted que la Navegabilidad de usuario de la sección de ejemplos es:

	CANTIDAD	%
Muy fácil	1	11
Fácil	8	89
Difícil	0	0
Muy difícil	0	0
Total	9	100

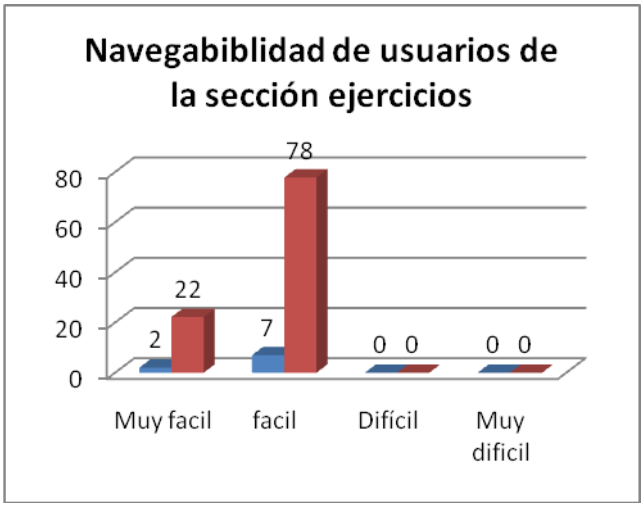


Grafica 7. Navegabilidad de usuarios sección ejemplos

Los datos muestran que la navegabilidad de usuario de la sección ejemplos es fácil al usuario

8. Cree Usted que la Navegabilidad de usuario de la sección de ejercicios es:

	CANTIDAD	%
Muy fácil	2	22
Fácil	7	78
Difícil	0	0
Muy difícil	0	0
Total	9	100

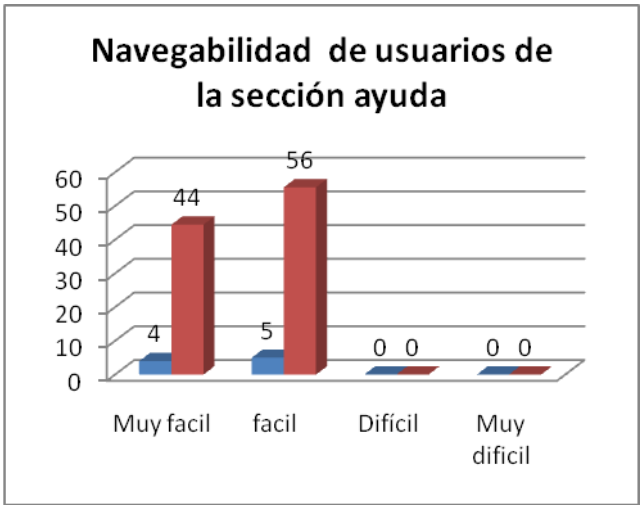


Grafica 8. Navegabilidad de usuarios sección ejercicios

Los datos muestran que la navegabilidad de usuario de la sección ejercicios es fácil al usuario

9. Cree Usted que la Navegabilidad de usuario de la sección de ayuda es:

	CANTIDAD	%
Muy fácil	4	44
Fácil	5	56
Difícil	0	0
Muy difícil	0	0
Total	9	100

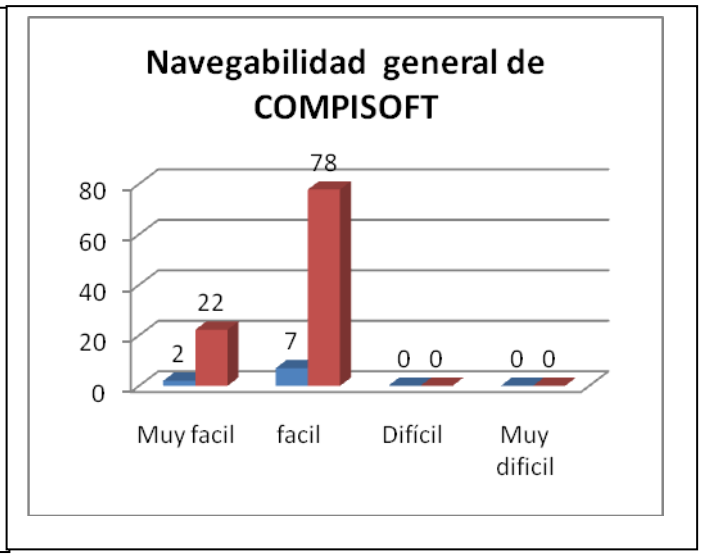


Grafica 9. Navegabilidad de usuarios sección Ayuda

Los datos muestran que la navegabilidad de usuario de la sección ayuda es fácil al usuario.

10. De forma general cree usted que la Navegabilidad de COMPISOFT es:

	CANTIDAD	%
Muy fácil	2	22
Fácil	7	78
Difícil	0	0
Muy difícil	0	0
Total	9	100

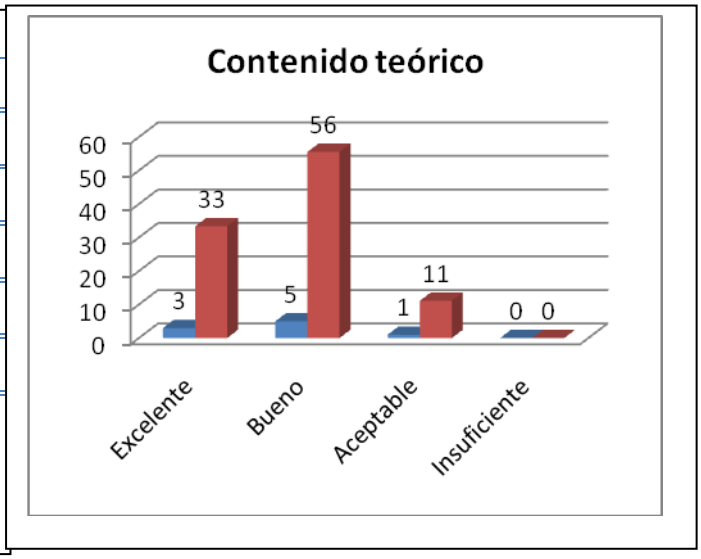


Grafica 10. Navegabilidad general Compisoft

Los datos muestran que la navegabilidad de COMPISOFT es fácil al usuario

11. Cree usted que el contenido teórico es:

	CANTIDAD	%
Excelente	3	33
Bueno	5	53
Aceptable	1	11
Insuficiente	0	0
Total	9	100

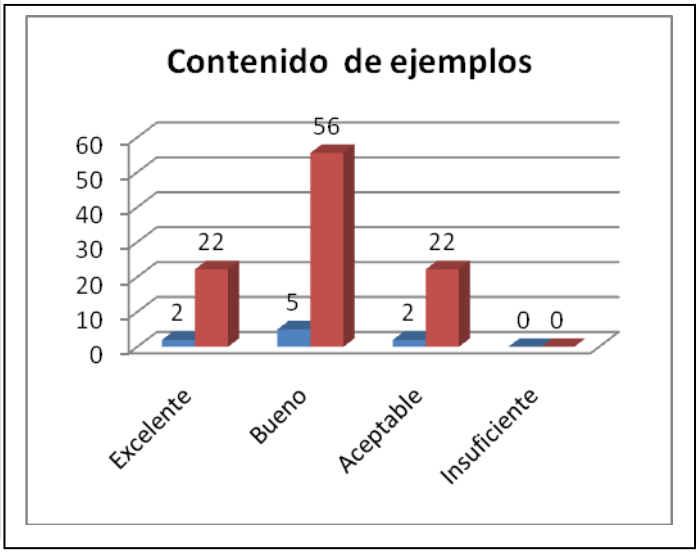


Grafica 11. Contenido teórico

Los datos muestran que el contenido teórico es bueno al usuario.

12. Cree Usted que el contenido de ejemplos es:

	CANTIDAD	%
Excelente	2	22
Bueno	5	56
Aceptable	2	22
Insuficiente	0	0
Total	9	100

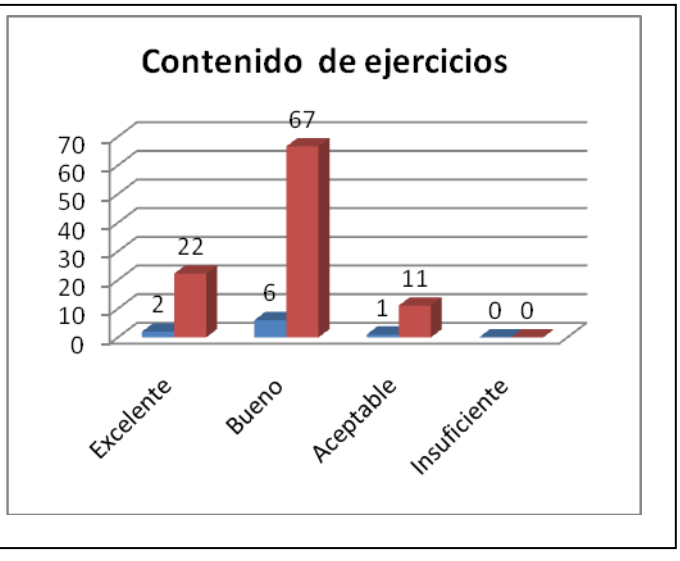


Grafica 12. Contenido ejemplos

Los datos muestran que el contenido de ejemplos es bueno al usuario

13. Cree Usted que el contenido de los ejercicios es:

	CANTIDAD	%
Excelente	2	22
Bueno	6	67
Aceptable	1	11
Insuficiente	0	0
Total	9	100

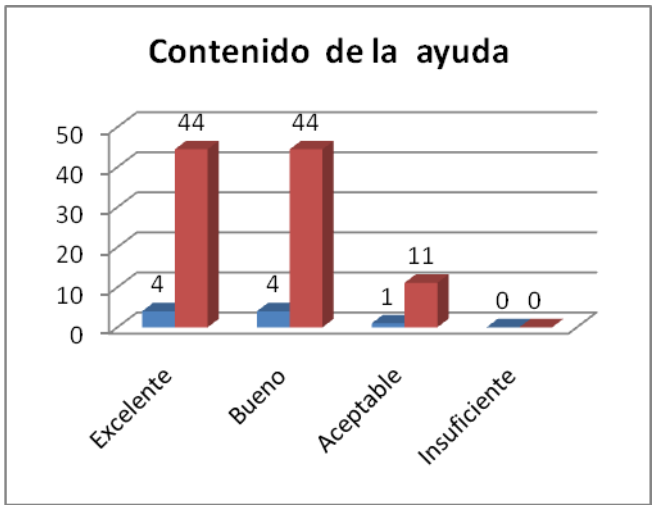


Grafica 13. Contenido ejercicios

Los resultados muestran que el contenido de ejercicios es bueno al usuario

14 Cree Usted que el contenido de ayuda es:

	CANTIDAD	%
Excelente	4	44
Bueno	4	44
Aceptable	1	11
Insuficiente	0	0
Total	9	100

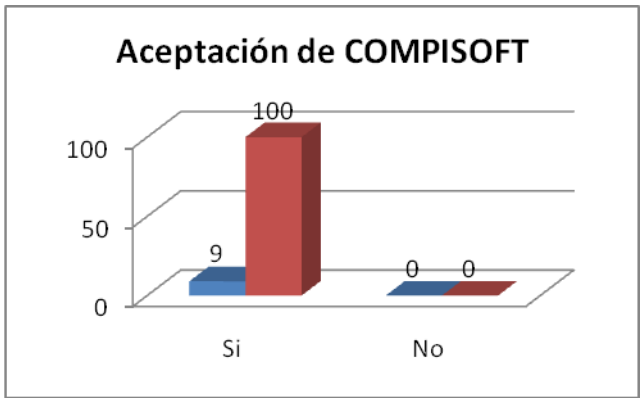


Grafica 14. Contenido Ayuda

Los resultados muestran que el contenido de la ayuda es bueno al usuario

15 Cree usted que el programa COMPISOFT Servirá de apoyo para el aprendizaje de la asignatura compiladores para los estudiantes del programa de Ingeniería de Sistemas de la Universidad San Buenaventura Cartagena.

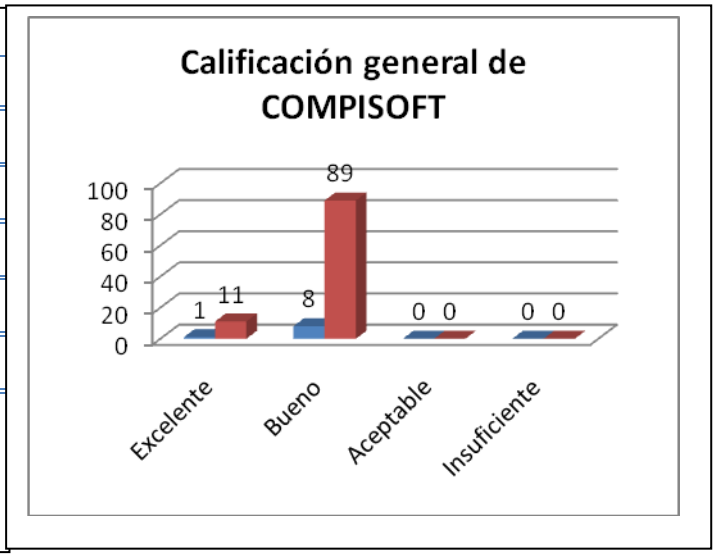
	CANTIDAD	%
Si	9	100
No	0	0
Total	9	100



Los resultados muestran que el 100% de los encuestados están de acuerdo que COMPISOFT si servirá de apoyo para el aprendizaje de la asignatura Compiladores.

16. En general como considera usted el programa COMPISOFT

	CANTIDAD	%
Excelente	1	11
Bueno	8	89
Aceptable	1	0
Insuficiente	0	0
Total	9	100



Grafica 16. Calificación general de Compisoft

Los resultados arrojados en el análisis de los datos estadísticos de las pruebas de campo realizadas, fueron muy positivas ya que se pudo determinar que el software es amigable a la vista, de fácil uso para el usuario final, además que su contenido es el adecuado para lograr el objetivo general del proyecto que es servir como herramienta de apoyo para el aprendizaje de la asignatura Compiladores.

CONCLUSIONES

Al Terminar esta investigación y el desarrollo del software educativo se concluye:

El diseño y desarrollo del software educativo permitió construir una herramienta tecnológica utilizando como medio de apoyo el aprendizaje, a partir de los diferentes procesos, teniendo en cuenta la temática de la cátedra de compiladores, lo cual permite a estudiante, profesores e investigadores mejorar sus conocimientos sobre esta cátedra, interactuando de una forma mucho más amigable, además se puede observar los diferentes eventos que se presentan en compiladores, lo que da como resultado una mejor motivación la realizar investigaciones relacionadas con esta cátedra.

Para la realización del diseño y desarrollo del simulador, se inició con un proceso de estudio y análisis de los diferente temas que hacen parte de compiladores, en dicho proceso se determinó dividir en *temas* donde se mostraran los temas mas básicos de la cátedra, *ejemplos* donde de forma grafo se muestra algunos ejercicios de algunos temas, *ejercicios* donde se utiliza una aplicación de Autómatas Finitos y una Auto evaluación, *Administrador* donde el docente pueda actualizar la evaluación, y otras como *Ayuda* y *Créditos*.

Luego de haber estudiado y analizado los temas de la cátedra de Transmisión de datos, se identificaron diferentes requerimientos, se determinó la parte del diseño gráfico de las interfaces que es muy importante para lograr capturar la atención del usuario y así utilice esta herramienta, una interfaz gráfica que sea agradable a la vista, de fácil manejo y navegabilidad son aspectos que se tuvieron en cuenta al momento de realizar el software. Otro de los requerimientos que se determino en esta fase fue: el contenido teórico y el contenido evaluativo debe ser claro y conciso para que el usuario no tienda a confundirse y entienda como los desarrolladores del software la expresaron.

Luego de haber identificado cada uno de los requerimientos que se tuvieron en cuenta para el desarrollo de la aplicación, se procedió a realizar el diseño de COMPISOFT, este se realizó con el lenguaje unificado de Modelado UML, el cual permitió a los desarrolladores de la aplicación realizar un diseño práctico y de fácil entendimiento para la construcción del software.

Luego del realizar el diseño en UML se procedió a realizar el diseño de las interfaces del software lo que permitió a los desarrolladores crear una aplicación agradable a la vista.

Al haber realizado el diseño, se realizó el desarrollo del software tomando como punto de referencia el modelo de desarrollo basado en componentes, utilizado

en la Ingeniería del Software, dicho modelo de desarrollo contiene varias fases que le permitieron a los desarrolladores construir una aplicación de manera más fácil y en menor tiempo, debido a que la principal característica de este modelo es la reutilización de código, lo cual brinda al modelo de desarrollo las características como evolutivo e iterativo proporcionando ventajas en cuanto a calidad del software producido, del mismo modo el modelo de desarrollo basado en componentes fue de gran utilidad debido a que la herramienta de desarrollo Flash Macromedia MX y su lenguaje de programación ActionScript se basan en la creación de componentes reutilizables.

En la etapa de desarrollo del software los desarrolladores se identificaron con los grandes beneficios de la herramientas Flash Macromedia y ActionScript, al momento de desarrollar la interfaz gráfica, debido a que es necesario mostrar los procesos de una manera clara, agradable a la vista, para que el usuario del software le sea más fácil comprender el contenido de esta herramienta. Del mismo modo se identificaron algunas falencias del las herramientas mencionadas anteriormente, dicha falencia es que no pueden conectarse a una base de datos directamente, lo que indujo a los desarrolladores utilizar una herramienta que sirviera de enlace entre la aplicación realizada en flash y la base de datos del software, la herramienta que permitió este enlace fue PHP, la utilización de PHP, fue muy importante para los desarrolladores del proyecto ya que además de proporcionar el enlace entre la aplicación hecha en Flash y la base de datos desarrollada en MySQL, como se menciono anteriormente, también permitió afianzar los conocimientos de esta herramienta tan importante en la programación Web de gran auge en la actualidad. La utilización de MySQL como motor de base de datos de permitió a los desarrolladores del software afianzar conocimientos acerca de esta herramienta que es la más utilizada a nivel mundial cuando se trata de desarrollar aplicaciones Web, que como se menciono anteriormente esta teniendo un gran auge en la actualidad, además de su gran compatibilidad con PHP.

Al realizaron las correspondientes pruebas pilotos y de campo para validar la funcionalidad del software, las cuales permitieron identificar algunos falencias de la validación y funciones innecesarias, las cuales fueron corregidas y permitieron que la aplicación tuviera un mejor desempeño.

Finalmente en los resultados obtenidos, la totalidad de estudiantes y especialistas encuestados están de acuerdo con un porcentaje del 100%, en que sea utilizado el software COMPISOFT como apoyo a la cátedra de transmisión de datos, cumpliendo con los objetivos planteados por los desarrolladores del software.

RECOMENDACIONES

Universidad:

La universidad debe promover la investigación sobre el campo de la tecnología y programación, para la innovación de nuevas herramientas de estudios que ayuden al mejoramiento del proceso Enseñanza-Aprendizaje, como es la creación de Software Educativos enfocados a las asignaturas de esta Universidad.

Estudiantes:

Buscar nuevas metodologías para el proceso de enseñanza-aprendizaje, y vayan de la mano de las nuevas tecnologías de la programación, por que gracias a ellas se han realizado día a día laboratorios virtuales llevados más a la realidad.

Bibliografía

BACA URBINA, Gabriela. Evaluación de proyectos. 4ta edición. México DF. MC Graw-Hill. 2001. 383 p.

GALVIS MANQUERA, Álvaro H**.** Ingeniería de Software Educativo. Primera edición. Santa Fe de Bogota. Edición Unian. 1997

INSTITUTO COLOMBIANO DE NORMAS TÉCNICAS Y CERTIFICACIÓN, Documentación. Numeración de divisiones y subdivisiones en documentos escritos 2ª actualización. Santa Fe de Bogota DC. ICONTEC.2002. 34 p NTC 1486.

MANUAL DE ORIENTACIÓN Y REGLAMENTACIÓN PARA TRABAJO DE GRADO. CBI. Formulación de la Propuesta.

NASSIR, Caín. Preparación Y Evaluación De Proyecto. 3era edición.
Mc Graw Hill 1997.

PROYECTO EDUCATIVO BONAVENTURIANO –P.E.B. Universidad de San Buenaventura. Pág. 85.

PRESSMAN, Roger S. Ingeniería del Software: Un Enfoque Practico. 4ta edición Madrid: Mc Graw Hill, 1997.

TAMAYO Y TAMAYO. Metodología de la Investigación. Colombia: Mc Graw Hill, 1995.

http://www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html
Ingeniería de software.
Fecha de Visita: 15 de marzo de 2008

http://www.icontec.org.co/normas/ShowResults.asp
GUÍA PARA LA ELABORACIÓN Y PRESENTACIÓN DE TRABAJOS ESCRITOS.
Fecha de Visita: 25 de marzo de 2008

http://www.monografias.com/trabajos5/desof/desof.html
PROCESO DE DESARROLLO DE SOFTWARE
Fecha de Visita: 15 de marzo de 2008

http://www.el-directorio.org/index.php/%C2%BFQu%C3%A9_es_Software%3F
¿QUÉ ES SOFTWARE?
Fecha de Visita: 16 de marzo de 2008

ANEXOS

ANEXO A
UNIVERSIDAD DE SAN BUENAVENTURA
PROGRAMA DE INGENIERÍA DE SISTEMAS
PRUEBA DE CAMPO A ESTUDIANTES

Fecha_____ nombre_____
semestre_____

Proyecto: **DESARROLLO DE UN SOFTWARE EDUCATIVO QUE SIRVA DE APOYO PARA EL APRENDIZAJE DE LA ASIGNATURA COMPILADORES DEL PROGRAMA DE INGENIERÍA DE SISTEMAS DE LA UNIVERSIDAD DE SAN BUENAVENTURA CARTAGENA**

Objetivo: comprobar que el programa compisoft cumpla con todos los requerimientos solicitados tales como una interfase, navegabilidad y su contenido.

INTERFAZ

1. Cree usted que la interfaz de usuario de la sección teórica es:

Muy amigable ☐ No tan amigable ☐

Amigable ☐ Muy poco amigable ☐

2. Cree Usted que la Interfaz de usuario de la sección de ejemplos es:

Muy amigable ☐ No tan amigable ☐

Amigable ☐ Muy poco amigable ☐

3. Cree Usted que la Interfaz de usuario de la sección de ejercicios es:

Muy amigable ☐ No tan amigable ☐

Amigable ☐ Muy poco amigable ☐

4. Cree Usted que la Interfaz de usuario de la sección de ayuda es:

Muy amigable ☐ No tan amigable ☐

Amigable ☐ Muy poco amigable ☐

5. De forma general cree usted que la Interfaz de COMPISOFT es:

Muy amigable ☐ No tan amigable ☐

Amigable ☐ Muy poco amigable ☐

NAVEGABILIDAD

6. Cree usted que la Navegabilidad de usuario de la sección teórica es:

Muy Fácil ☐ Difícil ☐

Fácil ☐ Muy Difícil ☐

7. Cree Usted que la Navegabilidad de usuario de la sección de ejemplos es:

Muy Fácil ☐ Difícil ☐

Fácil ☐ Muy Difícil ☐

8. Cree Usted que la Navegabilidad de usuario de la sección de ejercicios es:

Muy Fácil ☐ Difícil ☐

Fácil ☐ Muy Difícil ☐

9. Cree Usted que la Navegabilidad de usuario de la sección de ayuda es:

Muy Fácil ☐ Difícil ☐

Fácil ☐ Muy Difícil ☐

10. De forma general cree usted que la Navegabilidad de COMPISOFT es:

Muy Fácil ☐ Difícil ☐

Fácil ☐ Muy Difícil ☐

CONTENIDO

11. Cree usted que el contenido teórico es:

Excelente ☐ Aceptable ☐

Bueno ☐ Insuficiente ☐

12. Cree Usted que el contenido de ejemplos es:

Excelente ☐ Aceptable ☐

Bueno ☐ Insuficiente ☐

13. Cree Usted que el contenido de los ejercicios es:

Excelente ☐ Aceptable ☐

Bueno ☐ Insuficiente ☐

14. Cree Usted que el contenido de ayuda es:

Excelente ☐ Aceptable ☐

Bueno ☐ Insuficiente ☐

OTROS

15. Cree usted que el programa COMPISOFT Servirá de apoyo para el aprendizaje de la asignatura compiladores para los estudiantes del programa de Ingeniería de Sistemas de la Universidad San Buenaventura Cartagena.

Si ☐ No ☐

16. En general como considera usted el programa COMPISOFT

Excelente ☐ Aceptable ☐

Bueno ☐ Insuficiente ☐

Observaciones y/o Recomendaciones