

**DESARROLLO DE UN SOFTWARE INTERPRETE DE ALGORITMOS PARA  
FACILITAR LA ENSEÑANZA DE LOS FUNDAMENTOS PROGRAMACIÓN**

Luís Roberto Olascoaga Surmay  
Daniel José Salas Álvarez

Universidad de Córdoba  
Facultad de Ciencias Básicas e Ingenierías  
Departamento de Ingeniería de Sistemas y Telecomunicaciones



# **DESARROLLO DE UN SOFTWARE INTERPRETE DE ALGORITMOS PARA FACILITAR LA ENSEÑANZA DE LOS FUNDAMENTOS PROGRAMACIÓN**

**Lic. Esp. Luís Roberto Olascoaga Surmay**  
**Investigador grupo SOCRATES**  
**Universidad de Córdoba**  
[Luisolascoaga@sinu.unicordoba.edu.co](mailto:Luisolascoaga@sinu.unicordoba.edu.co)

**Ing. Mag. Daniel José Salas Álvarez**  
**Investigador grupo SOCRATES**  
**Universidad de Córdoba**  
[dsalas@sinu.unicordoba.edu.co](mailto:dsalas@sinu.unicordoba.edu.co)

## **Resumen**

El propósito de este trabajo de investigación es diseñar e implementar un software intérprete de algoritmo (INALGORI) como herramienta didáctica que facilite el proceso de enseñanza de la algoritmia en el programa de Licenciatura en Informática y Medios Audiovisuales de la Universidad de Córdoba. De esta manera los estudiantes y docentes contarán con un software en español que les facilite la tarea de codificación, revisión sintáctica, análisis de la ejecución, realización de pruebas de escritorio, ejecución paso a paso, detección de errores lógicos y análisis de flujo de ejecución que contribuya de manera significativa en su proceso de aprendizaje como fundamento para desempeñarse apropiadamente en el área de programación. El software presenta una interfaz atractiva y sencilla de utilizar, además, está provisto de un sistema de ayuda en línea, manual de usuario y documentación de las diferentes etapas de su desarrollo, para lo cual se ha tenido en cuenta la teoría de lenguajes compilados, interpretes e Ingeniería del Software, tomando aspectos relacionados con el lenguaje de modelado unificado (UML), análisis y diseño orientado a objetos, desarrollo de componentes y modelo de tres capas.

## **Palabras Claves**

Enseñanza, Aprendizaje, Ingeniería del Software, Algoritmo, Interprete, UML.

## **Abstract**

## **Keywords**

## 1. Antecedentes

La asignatura de fundamentos de algoritmia del programa de Licenciatura en Informática y Medios Audiovisuales introduce al estudiante en el mundo de la programación y en su desarrollo se manifiesta una serie de dificultades como la pérdida de la asignatura por un número considerable de estudiantes; hecho que se atribuye principalmente a la complejidad de la algoritmia, al rigor lógico de los conceptos, dificultad en el análisis e interpretación de problemas, rigidez en las normas sintácticas para escribir el código y además por la variabilidad misma de los ejercicios; ya que estos pueden ser completamente diferente a los resueltos en clase aunque contemplen los mismos conceptos y reglas para resolverlos. Además de esto el léxico empleado para escribir la solución de los ejercicios es variable, acorde con los criterios de docentes, autores y en últimas de los textos seguidos para la temática; lo cual conlleva a una diferencia significativa entre la metodología del docente y la aplicada por los libros, recursos de Internet y personas a las que acuden los estudiantes para buscar asesoría y ayuda.

En este sentido, observamos que no existe un software adecuado que tenga un léxico y sintaxis “normalizada”, que verifique la correcta ejecución de un algoritmo mediante la prueba de escritorio y con el cual el estudiante pueda transcribir y revisar el algoritmo; que así mismo le facilite al docente la revisión y prueba de los ejercicios, tarea que se hace tediosa por que los grupos son normalmente numerosos. De esta manera vemos que actualmente no se cuenta con una herramienta para algoritmos capaz resaltar la sintaxis de las sentencias acorde con su naturaleza para que el estudiante pueda recordar y diferenciar bien su significado y aplicación reduciendo así el número de errores que comente, y que además le preste un sistema de ayuda en línea para facilitar la escritura correcta de las sentencias. Se hace necesario entonces desarrollar un software para esta asignatura que cumpla con los requerimientos señalados anteriormente; que incluya un sistema de seguimiento de variables y que almacene los resultados de las entradas y salidas aplicadas para posibilitar un proceso de ejecución del algoritmo mas controlado. De igual manera es requerido un sistema de ejecución paso a paso del algoritmo, facilitando el análisis del flujo de ejecución especialmente en algoritmos largos o complejos que contengan sentencias condicionales y repetitivas.

## 2. Marco Teórico.

En el desarrollo del Software INALGORI se abordan diferentes teorías y conceptos tales como la algoritmia, la teoría de compiladores e intérpretes y la Ingeniería del Software, entre otros.

Los algoritmos se definen como un conjunto de pasos, procedimientos o acciones que nos permiten alcanzar un resultado o resolver un problema [5]. Estos pasos o instrucciones deben tener tres características esenciales: Precisión, Determinismo y finitud [4]. Los algoritmos de acuerdo con la naturaleza de las instrucciones se clasifican en dos grupos [4]: Algoritmos cualitativos, descritos por medio de palabras y variables empleados en la realización de una actividad. Algoritmos cuantitativos, que involucran cálculos numéricos usados en la obtención de una respuesta. En la elaboración de un algoritmo se siguen tres etapas: análisis del problema, construcción y verificación del algoritmo [5]. El software INALGORI se ajusta al tipo de algoritmo cuantitativo y en su construcción se han implementado las instrucciones propias de un algoritmo, tales como: variables, constantes, ciclos, sentencias de entrada y salida, entre otras.

Para la ejecución del algoritmo se requiere de la aplicación de la teoría de compiladores e intérpretes. Los compiladores son programas que traducen de un lenguaje a otro tomando como entrada un programa escrito en lenguaje fuente y produciendo un programa en lenguaje máquina [8]. De esta manera, los compiladores traducen programas escritos en un lenguaje de alto nivel a código intermedio o código de máquina [10]. Los intérpretes no generan código de máquina sino que analizan y ejecutan directamente cada proposición del código fuente [10]; un intérprete es un traductor de lenguaje que ejecuta el programa fuente inmediatamente en vez de generar un

código de máquina [8]. El lenguaje o código fuente para INALGORI son todas las instrucciones del algoritmo también llamadas pseudo código.

En análisis, diseño e implementación de INALGORI se tuvo en cuenta los fundamentos de la ingeniería del software, el proceso unificado y UML. La ingeniería del Software es un conjunto de normas, técnicas y procedimientos para el desarrollo de software.

Por su parte el proceso unificado es un proceso de desarrollo de software configurable que se adapta a proyectos de tamaño y complejidad variados, que guía al equipo de trabajo en cómo administrar el desarrollo iterativo de un modo controlado mientras se balancean los requerimientos del negocio, el tiempo y los riesgos del proyecto [12]. Este proceso guió la construcción de la arquitectura del software en las vistas de negocio, aplicación, información y tecnológica.

Entre tanto el UML (lenguaje de modelado unificado) es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema de software [3].

### **3. Estado del arte.**

Desde las décadas de los 60, 70 y 80 los esfuerzos en la construcción de nuevos lenguajes de programación se orientaron a desarrollar lenguajes fáciles de aprender, con depuración en línea, ayudas visuales, sintaxis sencillas y que paralelamente se ajustaran más a las características del problema que a la arquitectura de las computadoras. Esta necesidad pronto se extendió a programas especiales para la ejecución de algoritmos surgiendo así herramientas como Karel, que es un lenguaje algorítmico para programar a un robot (Karel) mediante instrucciones sencillas y bien estructuradas que son fáciles de entender y utilizar. También tenemos a LCC (Lea Code Compiler) que es un compilador de algoritmos que implementa estructuras secuenciales, selectivas e iterativas, tipos lógicos, aritméticos, vectores, matrices y conjuntos, parámetros por valor y por referencia. En Paraguay se diseñó SL, que incluye en los algoritmos el uso de estructuras de datos y programación modular. SL permite la definición de tipos de datos básicos, registros y arreglos unidimensionales y multidimensionales. Los subprogramas comprenden funciones y subrutinas, que aceptan parámetros por valor o por referencia. En el contexto Nacional el intérprete de algoritmos más conocido fue desarrollado por Cesar Becerra Santamaría, que venía adjunto a su libro Fundamento de Algoritmos. A nivel Regional, podemos señalar a DFD, que es un programa para el diseño e implementación de algoritmos expresados en diagramas de flujo (DF), que incorpora opciones de depuración facilitando la localización de errores de ejecución y lógicos habituales. A nivel local tenemos a PILATOX desarrollado en la Universidad de Córdoba que incluye operaciones como seguimiento de variables y ejecución paso a paso.

### **4. Construcción de INALGORI.**

INALGORI, se desarrolló fundamentado en el modelo Proceso Unificado, que está conformado por cuatro fases: iniciación también conocida como concepción, elaboración, construcción y transición, para cada una de ellas se asocia un flujo de trabajo que está conformado por las siguientes etapas: Especificación de Requisitos, Análisis, Diseño, Implementación y Pruebas.

En la aplicación del Modelo Proceso Unificado en INALGORI se han utilizado varias iteraciones, lo cual refleja un nivel granularidad fino, dado el refinamiento sucesivo en los flujos de trabajo.

#### **4.1 Especificación de requisitos.**

El desarrollo del software INALGORI tuvo en cuenta la metodología seguida para el desarrollo de software orientado a objetos, tal como lo es Proceso Unificado, las normas de la IEEE y los

diagramas de UML. En términos de la especificación de requisitos del sistema INALGORI consideramos los siguientes elementos: Objetivos del sistema, Requisitos de Información, restricción de Información y diagramas de casos de usos, este proceso está fundamentado en la norma IEEE 830.

#### **4.1.1 Objetivos del sistema.**

Dentro de los objetivos trazados para el sistema INALGORI tenemos la implementación de las instrucciones formales de un algoritmo incluyendo los tipos de datos, declaración de variables y constantes, sentencias de entrada y salida, operadores lógicos, relacionales, aritméticos y de bits, condicionales, ciclos, arreglos y matrices; además se agregaron funciones complementarias para operaciones matemáticas, conversión de tipos y bases numéricas.

Otro objetivo consistió en la gestión del código del algoritmo de una forma sencilla con una interfaz de documento múltiple para editar varios archivos al mismo tiempo en los cuales se resaltan en texto de colores diferentes las palabras reservadas discriminadas por grupos; así por ejemplo los tipos de datos y los operadores se muestran en colores diferentes. Dentro de este objetivo se incluyó la posibilidad de auto completar el código acorde con la combinación de teclas que permiten desplegar y escribir las instrucciones mas usadas en un algoritmo para facilitar su correcta y rápida escritura por parte del usuario.

Para el módulo de revisión sintáctica el objetivo consistió en la revisión del código del algoritmo para detectar posibles errores, resaltando en color marrón la línea donde se encuentran los errores detectados sin hacer especificación alguna sobre la naturaleza de los errores. En este punto el software genera una lista ordenada en la parte inferior de la ventana en la que se muestra los números de las líneas de cada uno de los errores y el usuario haciendo doble click en la lista se posesiona el cursor línea y carácter del error en cuestión.

Finalmente, para el módulo de ejecución el objetivo trazado fue la ejecución, prueba y seguimiento del código correspondiente de las instrucciones del algoritmo. De esta forma el docente o el estudiante pueden seleccionar el modo de ejecución continua o un modo de ejecución paso a paso; caso para el cual el programa resalta en rojo la línea del código por la cual va la ejecución, desplegando además, el número de líneas y total de instrucciones ejecutadas. Para facilitar el análisis de la ejecución del algoritmo, el software dispone de una tabla en la que se muestra el contenido de las variables registrando dinámicamente sus cambios, teniendo en cuenta que finalizada la ejecución del algoritmo esta información puede guardarse en un archivo plano separado por tabulador que consta del nombre de cada variable con su respectivo valor final. Igualmente es posible almacenar en un archivo de texto la información de la consola de salida y la entrada del usuario.

#### **4.1.2. Requisitos de Información**

Este aspecto define los datos a ser almacenados por INALGORI. Un primer requisito de información es el léxico del programa; representado por la definición de las sentencias, tipos de datos, símbolos especiales, operadores, precedencia de operadores y funciones internas empleados para los algoritmos. El siguiente requisito de información es la gestión de archivos; que consiste en guardar en archivos externos los algoritmos. En cuanto al resaltado sintáctico del código, el requisito de información consistió en almacenar los atributos de la fuente y colores que resaltan partes del código como palabras reservadas, operadores, tipos de datos, símbolos especiales etc. Finalmente, tenemos el requisito de información sobre consola de salida, que consistió en guardar en archivos de texto los resultados de la salida y entrada de la ejecución de un algoritmo, así como también el estado final de las variables con su nombre y contenido.

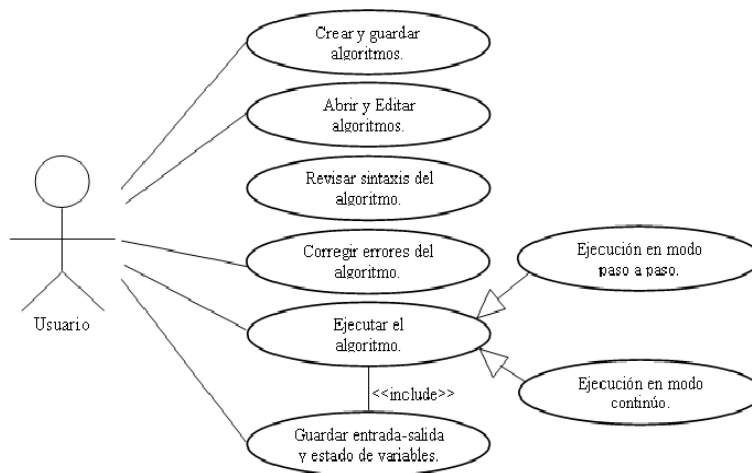
### 4.1.3. Restricciones de Información

Este aspecto contemplamos las validaciones y condiciones deseables de cada requisito de información; así por ejemplo una de las restricciones de información consideradas consistió en codificar y decodificar los archivos de los algoritmos creados y editados por el usuario en el programa para garantizar el formato de la información que espera encontrar INALGORI al abrir y ejecutar el algoritmo. Así mismo, INALGORI define como extensión única de los archivos el termino .alg. Para la ejecución del algoritmo, se espera que el código haya sido previamente guardado y además que se cumpla con el requisito de restricción correspondiente a un análisis correcto de la sintaxis del algoritmo en cuanto al requisito de información léxica.

### 4.1.4. Diagramas de casos de usos.

En el diagrama, el actor usuario representa al estudiante o al docente. Las especificaciones muestran que el usuario puede crear, guardar, abrir y editar archivos del código fuente del algoritmo. Luego el usuario efectúa la revisión sintáctica del código localizando errores y haciendo las correcciones respectivas. Finalmente, el diagrama ilustra el proceso de ejecución del algoritmo que se realiza de dos formas posibles. La primera la llamamos continua o normal, y en ella el algoritmo se ejecuta instrucción por instrucción sin hacer pausa entre cada sentencia ejecutada. El segundo modo de ejecución permite realizar una depuración del código ya que se ejecuta línea a línea y el usuario decide cuando continuar con la siguiente línea. En este tipo de ejecución el usuario observa el flujo de ejecución del algoritmo y el cambio del contenido de las variables. Finalizada la ejecución del algoritmo el usuario puede guardar en archivos de texto la información de la consola de entrada/salida y el nombre de las variables con su contenido.

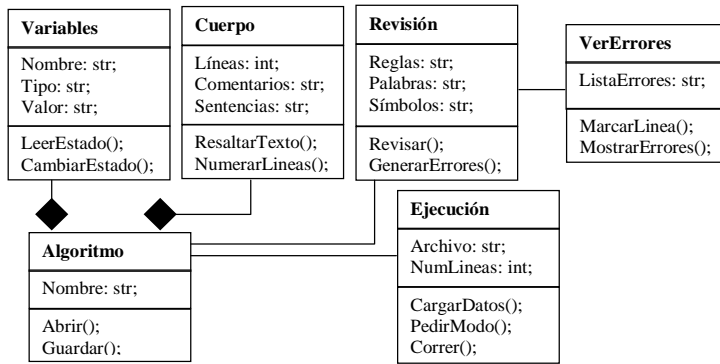
Figura 1. Diagrama de casos de uso.



### 4.2. Diseño.

El diseño del sistema INALGORI se llevó a cabo siguiendo el proceso unificado desarrollado mediante actividades que establecieron diseños más amplios de los requerimientos de los usuarios, lográndose una funcionalidad más estructura y modelo de dominio del problema que reflejara los planos software, obteniendo finalmente una vista del sistema De esta manera, en la fase de diseño se definió el modelo de dominio de INALGORI con los módulos y clases del sistema.

Figura 2. Diagrama de clases.



### 4.3. Implementación

Para la implementación del sistema INALGORI se tuvieron en cuenta los lineamientos del modelo Proceso Unificado, tomando base los aspectos relacionados con la herramienta de desarrollo, formato de archivos para almacenar los algoritmos, librerías, módulos y elementos externos como objetos del sistema operativo y librerías DLL.

La implementación, en general se llevó a cabo el siguiente orden de pasos:

- Proceso de selección de herramientas.
- Estándar de codificación.
- Codificación.

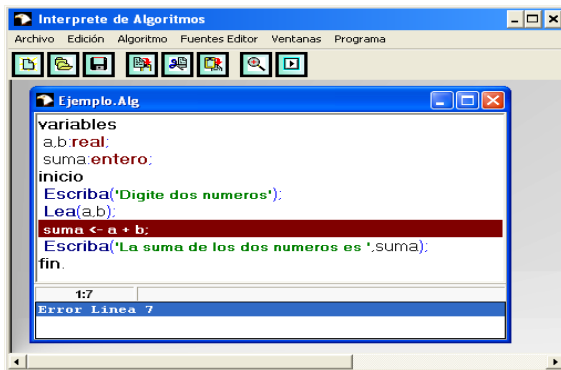
En cuanto al proceso de selección de la herramienta de desarrollo se tuvo en cuenta las consideraciones presentadas en la siguiente tabla.

Tabla 1. Criterios para selección de herramientas.

Herramientas de Programación	CRITERIOS									
	Paradigmas				Características					
	Herencia	Herencia Múltiple	Polimorfismo	Encapsulado	Portable	Económico	Integrado al WEB	Seguro	Robusto	Tecnología asociada
Smalltalk	✓	✓	✓	✓				✓	✓	
C++	✓	✓	✓	✓				✓	✓	
Eiffel	✓		✓	✓				✓	✓	
Delphi	✓	✓	✓	✓		✓	✓		✓	
Visual Basic	✓		✓	✓						
Java	✓		✓	✓	✓	✓	✓	✓	✓	✓

De acuerdo con los criterios estudiados y teniendo en cuenta otros aspectos tales como la experiencia en las herramientas, curva de aprendizaje y las licencias con que cuenta la universidad se procedió a la selección de herramientas. Acorde a lo anterior, los lenguajes más adecuados eran Java y Delphi, optándose por este último dado que sus diferencias con Java no eran significativas para la realización del proyecto. En consecuencia, la codificación fue estandarizada, ya que las clases, métodos y atributos se desarrollaron conforme a los lineamientos establecidos por el lenguaje.

Figura 3. Interfaz INALGORI.



#### 4.4. Pruebas

Para el desarrollo de este trabajo se efectuaron una serie de pruebas tendientes a asegurar los objetivos del mismo y haciendo los ajustes necesarios en cada etapa de su ejecución. En cuanto a la construcción del software se efectuaron pruebas de caja negra, caja blanca y de integración. Estas pruebas nos permitieron conocer debilidades en la funcionalidad del sistema, las oportunidades de optimización del código y la eficiencia en su ejecución. En relación con las pruebas con la población objetivo se tomó como muestra un grupo de 30 estudiantes del curso de algoritmia del programa de informática de la universidad de córdoba. Entre las características de la población tenemos que su edad esta entre 18 y 24 años, siendo el 73.3% mujeres y el 26.6% hombres. Aproximadamente el 85% del curso son repitentes de los cuales mas de la mitad ha reprobado el curso en mas de una ocasión. Los problemas presentados por los estudiantes se establecieron mediante una prueba aplicada para determinar los conocimientos y habilidades de los estudiantes en la aplicación de conceptos fundamentales necesarios para la solución de problemas de algoritmia. La aplicación de la prueba fue llevada a cabo por dos docentes de la asignatura, desarrollando ejercicios en los que se observó el proceso de codificación, revisión, ejecución y evaluación de problemas de algoritmos. Los resultados de esta prueba se presentan en la siguiente tabla.

Tabla 2. Prueba de conocimientos.

Concepto	Mal	%	Bien	%
Tipos de datos	9	30	21	70
Declaración de variables	12	40	18	60
Declaración de constantes	4	13,3	26	86,7
Sentencia de lectura/escritura	3	10	27	90
Operadores lógicos	19	63,3	11	36,7
Operadores relacionales	11	36,7	19	63,3
Operadores aritméticos	7	23,3	23	76,7
Condicionales simples	17	56,7	13	43,3
Condicionales anidados	21	70	9	30
Ciclo para	17	56,7	13	43,3
Ciclo mientras	22	73,3	8	26,7
Ciclo repita	24	80	6	20
Vectores	19	63,3	11	36,7
Matrices	23	76,7	7	23,3
Pruebas de escritorio	26	86,7	4	13,3

En los resultados expresados por la tabla observamos dificultades serias en la escritura (sintaxis)

y correcta utilización (semántica) de las instrucciones que forman parte de un algoritmo haciendo difícil su mecanización por parte del estudiante. Así mismo vemos que el proceso más complejo para el estudiante es la realización de pruebas de escritorio, con lo que consecuentemente se le dificulta el análisis, corrección y sustentación de los ejercicios delante del profesor y del grupo, causando una mala calificación por parte de este último. Igualmente docentes y estudiantes atribuyeron los malos resultados a la falta de herramientas didácticas y de software especializado para esta área.

## **5. Resultados Alcanzados**

Con la implementación y aplicación de INALGORI en el curso de fundamentos de programación encontramos que se facilitó el aprendizaje de los estudiantes por que el software presenta un conjunto bien definido de instrucciones que el usuario puede mecanizar y escribir adecuadamente. Este aspecto se vio mejorado por el despliegue automático en una lista de las sentencias más usadas, que se resaltan en colores diferentes lo cual permitió una mecanización sencilla por parte del usuario acerca de la sintaxis de los algoritmos. Consecuentemente la herramienta redujo el número de errores cometidos por los estudiantes. En términos de la ejecución del algoritmo se observó un mejor rendimiento en el estudiante en cuanto al análisis y depuración del algoritmo debido a la implementación de modo de ejecución paso a paso con la respectiva presentación de las variables y su contenido. Igualmente el docente se vio favorecido ya que el software le reduce el tiempo y errores de revisión de los algoritmos pudiendo probarlos rápidamente con diferentes datos de entrada.

## **6. Conclusiones**

Después de realizar el INALGORI podemos señalar los siguientes aspectos a manera de conclusiones:

- El uso de un conjunto finito de instrucciones y reglas sintácticas bien definidas y constantes a lo largo del curso facilita el aprendizaje de los estudiantes en la asignatura de algoritmia.
- La aplicación de la herramienta cumplió con una normalización adecuada del léxico y la sintaxis de las sentencias empleadas en la construcción de los algoritmos por parte de docentes y estudiantes.
- La herramienta INALGORI facilita el análisis de las variables del algoritmo durante su ejecución al presentar sus contenidos en forma permanente.
- La presentación del contenido de las variables facilita la detección y corrección de errores lógicos y de ejecución del algoritmo.
- El software INALGORI le permite al estudiante realizar las pruebas de escritorio del algoritmo de una manera más rápida y precisa.
- El almacenamiento en un archivo de texto de los resultados finales de la ejecución de los algoritmos le permitieron al docente revisar en forma más exacta y rápida los ejercicios resueltos por los estudiantes.
- El resaltado y despliegue automático de las sentencias y palabras reservadas del algoritmo por el software INALGORI redujo el número de errores sintácticos cometidos por los estudiantes y el tiempo empleado en la digitación de los mismos.
- La revisión sintáctica automática y detección de líneas e instrucciones con errores mejoró el proceso de depuración de algoritmos en los estudiantes y docentes.
- Para los estudiantes resultó más fácil observar el flujo de ejecución de un algoritmo y la naturaleza del comportamiento de las sentencias utilizadas, apoyándose en el sistema de ejecución paso a paso incorporado en el software.

## **7. Trabajo Futuro**

Después de realizar la presente investigación, se proyecta a futuro incorporar los módulos e instrucciones necesarias para adecuar una metodología de algoritmos orientadas a objetos, así

como también implementar en INALGORI la capacidad de construir diagramas de clases en forma gráfica y generar código de manera automática para implementar los métodos de las clases y demás conceptos de la orientación a objetos, teniendo en cuenta una interfaz en la que el usuario ingrese un conjunto mínimo y sencillo de datos para la generación del código que implemente las clases y sus relaciones.

### **Referencias Bibliográficas**

- [1] AHO A.; HOTPCROFT J.; ULLMAN J. Estructuras de datos y algoritmos. Editorial Addison – Wesley Iberoamericana 1988.
- [2] AHO A.; SETHI, R.; ULLMAN, J. Compiladores, Principios Técnicas y Herramientas. Editorial Addison - Wesley Iberoamericana 1990.
- [3] BOOCH GRADY, JACOBSON IVAR, RUMBAUGH JAMES. El lenguaje unificado de Modelado. Editorial Addison wesley,. 1999.
- [4] CARO PINEDA, SILVINA. Lógica de programación y algoritmos. Fundación Universitaria Boyacá. Tunja 2003. 335 Pág.
- [5] CAIRO, Oswaldo. Metodología de la Programación, Algoritmos, Diagramas de Flujo y Programas Tomo I. Editorial Computec. México 1995. 227 pág.
- [6] JOYANES, AGUILAR LUIS. Fundamentos de Programación, Algoritmos Y Estructuras de Datos. Editorial Mc Graw Hill. Madrid 1996. 707 pág.
- [7] KENNETH C., LOUDEN. Construcción de compiladores, principios y práctica.
- [8] TANEMBAUM, ANDREW, Sistemas Operativos Modernos , Prentice Hall, Mexico, 2003
- [9] TEUFEL B. CHMIDT S.; TEUFEL, T. Compiladores. Conceptos fundamentales. Editorial Addison - Wesley Iberoamericana 1995.
- [10] STALING, WILLIANG, Sistemas Operativos, Prentice Hall, México, 2003
- [11] <http://www.angelfire.com/scifi/jzavalar/apuntes/IngSoftware.html>
- [12] <http://www.willydev.net/descargas/articulos/general/IngSoftware.aspx>
- [13] <http://www.rational.com>