COMPARACIÓN DE HERRAMIENTAS PARA LA ENSEÑANZA DE LA PROGRAMACIÓN

Patricia Zavaleta Carrillo
Universidad Autónoma del Carmen
pzavaleta@pampano.unacar.mx

Damaris Pérez Cruz
Universidad Autónoma del Carmen
dperez@pampano.unacar.mx

J. Felipe Cocón Juárez
Universidad Autónoma del Carmen
jcocon@pampano.unacar.mx

Elvia Elvira Morales Turrubiates
Universidad Autónoma del Carmen
emorales@pampano.unacar.mx

Resumen

Éste artículo presenta el resultado de una investigación sobre las diferentes herramientas, métodos y técnicas que se emplean en el proceso de enseñanza / aprendizaje de la programación. Incluye una descripción de los trabajos de investigación relacionados con el desarrollo y uso de herramientas, métodos y técnicas para la enseñanza de la programación en Universidades versus Facultad de Ciencias de la Información (FCI) de la Universidad Autónoma del Carmen (UNACAR), con el propósito de proponer aquellos que son aplicables en la enseñanza de la programación dentro de la Facultad.

Palabra(s) Clave(s): Aprendizaje, enseñanza, programación.

1. Introducción

En la programación de computadoras se requieren de varias competencias y habilidades relacionadas con la capacidad de manipular un conjunto de abstracciones interrelacionadas para resolver problemas [1], para enseñar a programar, se tiene primero que enseñar cómo solucionar problemas, especificar los pasos a seguir para la solución del problema y posteriormente se pueden utilizar las herramientas para escribir el algoritmo o programa. El método propuesto por George Polya [2] es una guía muy utilizada en la resolución de problemas, formada por cuatro pasos: 1) comprender el problema, 2) diseñar un plan de solución, 3) ejecutar el plan y 4) examinar la solución obtenida, lo que se puede corresponder con la metodología para el desarrollo de programas en sus fases básicas. En áreas relacionadas con la enseñanza en áreas computacionales se han hecho diferentes investigaciones de las cuales se derivan técnicas, metodologías y herramientas computacionales que buscan apoyar a los profesores y alumnos en la enseñanza y aprendizaje de la programación y áreas de la computación y desarrollo de sistemas [3, 4].

En la búsqueda de mejores resultados en la enseñanza de la programación, en la FCI se llevó a cabo una investigación para contextualizar la problemática de la enseñanza, analizando los métodos, las técnicas y las herramientas que facilitan el proceso de enseñanza aprendizaje de la programación. A continuación se listan los pasos seguidos en la investigación:

a) Se realizó investigación exploratoria sobre publicaciones relacionadas con la difusión de investigación y experiencias en todo lo relacionado con la enseñanza de la programación.
b) Se elaboró listado descriptivo de los diferentes métodos, técnicas y herramientas encontradas.
c) Se compararon y clasificaron las herramientas computacionales, métodos o técnicas descritas en los artículos para considerar su aplicabilidad en los cursos de programación de la FCI.
d) Se analizó la información concentrada en la investigación descrita en [25], y se elabora un cuestionario para identificar los métodos, técnicas y

herramientas que fueron usados en sus cursos de programación el cual es aplicado a los alumnos de la FCI. El instrumento se aplicó a 84 alumnos activos en la FCI durante el ciclo escolar Agosto-diciembre de 2012, dicha población consta de alumnos que pertenecen a las generaciones de cursos del modelo de aprendizaje basado en experiencias y los del modelo basado en competencias.

e) Finalmente se compararon los métodos, técnicas y herramientas utilizados en la FCI contra los mencionados en los artículos seleccionados, para realizar una propuesta de diferentes métodos, técnicas y herramientas que pueden aplicarse en los diferentes cursos de la FCI.

2. La programación en Instituciones de Educación

Para satisfacer el incremento en el uso de las computadoras a nivel Nacional e Internacional en todas las áreas laborales y profesionales, se han creado diferentes carreras que forman profesionistas en diferentes áreas de la computación, informática y desarrollo de sistemas computacionales. Estas carreras llevan como base al menos un curso de Programación y con ello la preocupación de cómo se debe impartir, lo que ha originado distintas investigaciones y propuestas relacionadas con la enseñanza de programación.

Relacionado con la enseñanza de programación a nivel básico, en Colombia, desarrollaron una herramienta llamada DFD, el cual es un editor e intérprete de algoritmos básicos representados en diagramas de flujo [5], pretendía introducir un método didáctico e interactivo para la enseñanza y el estudio del comportamiento de algoritmos.

En la Universidad de Stanford, se desarrolló MiniJava un lenguaje de programación para enseñar POO [6], es un subconjunto restringido de la versión estándar de java diseñado para reducir el factor intimidación en estudiantes cuando se encuentran con sistemas tan grandes como el medioambiente de java.

En la Universidad de Talca en Chile [7], proponen una metodología para aprender a programar, consiste en: comprender primero la arquitectura del computador, exponiendo sus características internas y estudiando la forma en que ejecuta los

programas. Los alumnos trabajan con el lenguaje ensamblador de un computador ficticio, que cuenta con un conjunto de instrucciones muy simple y limitado. Posteriormente, la mayor parte del curso se concentra en la programación en el lenguaje Java.

El proyecto IDEFIX [8] consiste en el desarrollo de una plataforma que facilite la enseñanza de la programación en diferentes lenguajes mediante la utilización de Internet. El sistema permite la realización de prácticas de laboratorio mediante la creación de un entorno dinámico de desarrollo. En este entorno, los estudiantes tienen acceso a los enunciados de los ejercicios de programación y los evalúa de forma automática.

En el Departamento de Ciencias e Ingeniería de la Computación de la Universidad Nacional del Sur, se desarrolló un ambiente de aprendizaje con un editor interactivo de algoritmos, un constructor automático de trazas, y un traductor de programas en lenguaje *Pascal* [9]. Para minimizar las etapas que no aportan demasiado interés en lo que se refiere a la programación, específicamente la resolución del problema, al diseño y a la formulación del programa, el editor de algoritmos ayuda al alumno en la especificación del mismo y permite comprobar que esté correcto con la detección de errores mediante la confección automática de trazas. El editor cuenta con plantillas de las diferentes estructuras de control, datos de entrada, datos de salida, comprobación de tipos de datos; cuenta con un corrector sintáctico y semántico.

En la tesis [10], desarrollo un medioambiente de programación llamado *Storytelling Alice*, que da en nivel secundaria, una primera experiencia con programación de computadoras, con el propósito de presentar la programación como un significado con el fin de contar historias, como actividad de motivación, la herramienta incluye tres tipos de soportes, para habilitar a los usuarios para crear historias: el primero, animaciones de alto nivel que permiten el uso de personajes sociales que pueden interactuar con otros, segundo, una galería de personajes y elementos de escena que ayudan a encontrar historias, finalmente un tutorial basado en historia.

El Dr. Andrew Scott, profesor asistente de la Universidad de Western Carolina, desarrolló *Programimate*, una herramienta en java disponible en línea, para la

enseñanza de programación visual orientada a programadores novatos. Su objetivo es hacer más fácil la asimilación de la programación en su etapa inicial, proporcionando un entorno amigable, fácil de usar y sin la complejidad de la escritura de la sintaxis de un lenguaje [11].

De igual modo, en la Universidad Politécnica del Estado de México, se desarrolló una herramienta sustentada en los cuatro grandes paradigmas de la programación (Paradigma Imperativo, Paradigma Funcional, Paradigma Basado en reglas, Paradigma Orientado a Objetos) partiendo de la hipótesis de "la reducción del tiempo en aprender a programar, está relacionada positivamente con el uso de la metodología de diseño de algoritmos y de una herramienta automatizada en un sistema de información", [12].

Otro medioambiente que se desarrolló para enseñar programación a los niños es *Scratch* un entorno para enseñar conceptos básicos de programación, fue desarrollado en el MIT (Instituto Tecnológico de Massachuset), permite comenzar a entender conceptos como ciclos, control de flujo, señales, etc. Este entorno es completamente gráfico. Los juegos que se construyen son animaciones a las que se les coloca comportamiento mediante programas, y los programas se construyen mediante elementos que se arrastran y pegan como piezas de LEGO. *Scratch* está construido sobre *Squeak*, que es un ambiente para desarrollo multimedia hecho en lenguaje Smalltalk [13]. Aunque fue creado para niños, es utilizado en algunas universidades como una herramienta para facilitar el aprendizaje de la programación.

Algunos investigadores [14], proponen en una primera aproximación, asociar niveles de la jerarquía de *Bloom* con herramientas de programación concretas. Basados en la tabla de niveles de experimentación y los niveles de conocimiento de *Bloom*, hicieron una clasificación de herramientas de programación adecuadas para cada nivel.

Con respecto a la enseñanza de la programación en el libro de Programación Estructurada [15], se propone enseñar Programación Estructurada antes que Programación Orientada a Objetos (POO) y la Visual, ya que la POO se basa en la programación estructurada y va más allá, la POO incluye los elementos básicos de

la estructurada como son los tipos de datos básicos, las estructuras de control, módulos y funciones. Además menciona que un error que se comete cuando se enseña a programar es: enseñar directamente sobre un lenguaje, lo que supone que el alumno tenga que aprender la lógica de programación y la sintaxis del lenguaje al mismo tiempo.

Martin Blom y otros [16], de la Universidad de Karlstad, proponen un método de programación basada en contratos para forzar los aspectos semánticos y lo experimentan en un curso, para ver las ventajas que tal método pueda tener sobre las habilidades del estudiante. Un aspecto importante cuando se enseña tecnología OO es la semántica de la programación, muchas veces es tradicional que se enseñe sobre sintaxis y mecanismos del leguaje y cualquiera que sea la semántica se le da menos tiempo. Presentan el método y un experimento llevado a cabo en un curso sobre un trabajo de proyecto y java para comparar el método con un método de programación estándar. Los estudiantes recibieron el método positivamente quienes dijeron que trabajar con el método resulto muy natural.

El Simulador Gráfico Asistido por Computador Para la Enseñanza de Programación de Lenguajes Estructurados (SIGACLE), su principal objetivo es el desarrollo de habilidades cognoscitivas en estudiantes de pregrado en las materias de programación de computadores. El objetivo pedagógico es presentar al estudiante vistas diferentes de lo que sucede en la máquina al ejecutar un programa. El ambiente gráfico muestra el estado de la memoria y la forma en que las instrucciones del programa lo afectan, permite comprender las acciones internas que ocurren en el computador [17].

Chesñevar del Departamento de Computación, de la Universidad Nacional del Sur en Argentina [1], propone los mapas conceptuales cómo una técnica didáctica que facilita a los alumnos una mayor comprensión y vinculación de conceptos relacionados con la elaboración de programas, añadiendo atributos visuales para facilitar la distinción de conceptos.

En la Universidad Autónoma de Aguascalientes implementaron una estrategia instruccional [18] basada en el modelo de Robert Gagne, en la que concretan el enfoque metodológico de la estrategia de diseño de solución en dos puntos

principales, el primero es un enfoque de estructuración del problema ó diseño de soluciones en la que sugieren la fijación de principio generales (objetivos y criterios) y segundo, la aplicación del paradigma práctica reflexiva, analizando y auto-reflexionando sobre los programas generados; para que de esta manera se afinen los "esquemas" generados.

En la Universidad del Istmo en Oaxaca, se implementó ABEA (Asistente Básico en la Enseñanza de Algoritmos), software educativo que busca ser un elemento auxiliar en la enseñanza en cursos de introducción a la programación, poniendo énfasis en la resolución de algoritmos, propiciando que el estudiante desarrolle su capacidad de abstracción de forma gradual y sin perderse en detalles de un lenguaje especifico. El diseño de la herramienta sigue la heurística de resolución de problemas de Polya fomentando en el estudiante buenas prácticas de programación desde el análisis del problema hasta la prueba y verificación de la solución algorítmica [19].

Aparte de lo mencionado anteriormente, se han desarrollado varias herramientas computacionales que apoyan en la enseñanza de cursos afines al área de las ciencias computacionales. Villalobos y otros [20] describen un conjunto de herramientas computacionales que permiten a un estudiante experimentar con los conceptos introducidos en el curso de estructuras de datos. Otro ejemplo es un sistema tutorial interactivo de estructura de datos [21] que muestra mediante animaciones gráficas el comportamiento de las operaciones básicas sobre las estructuras de datos clásicas (listas, pilas, colas y árboles binarios).

En el Departamento de Sistemas Informáticos y Programación de la Universidad Complutense de Madrid desarrollaron una herramienta informática para la visualización interactiva de estructuras de datos y esquemas algorítmicos [22], en su herramienta pretenden mostrar la separación entre especificación e implementación de una estructura de datos y proporcionar ejemplos de utilización de dichas estructuras, la herramienta cuenta con los tipos de datos: pilas, colas, árboles binarios de búsqueda, árboles *AVL*, colas de prioridad, tablas ordenadas y tablas dispersas.

En el Departamento de Sistemas Computacionales de la Universidad Autónoma de Baja California Sur [23], presentan una herramienta didáctica que proporciona al alumno un entorno de programación en el cual puede diseñar algoritmos básicos para el manejo de listas ligadas en pseudocódigo, y comprobar su funcionamiento y desempeño de manera visual.

Para el curso de ingeniería de software, Alex Baker y otros [24] proponen un juego de cartas educacional, que liga la teoría y la práctica con los fenómenos que ocurren en los procesos de ingeniería de software del mundo real. Las cartas están diseñadas para enseñar aquellos procesos que no son suficientemente subrayados en las lecturas y proyectos. Simulan los procesos desde la especificación de requerimientos hasta el producto desarrollado. El juego incluye además, la competencia, cada jugador toma el papel de administrador de proyecto y deberá completar el proyecto antes que alguno de sus oponentes lo hagan, también se incluye aprendizaje colaborativo, el juego es físico (se juega con interacción cara a cara entre los jugadores).

En la Universidad Nacional de Colombia, proponen “El Juego de la Consistencia” [3], que trabaja el concepto de consistencia entre diagramas de UML (*Unified Modeling Language*), los diagramas que se usan en el Juego de la Consistencia son: diagramas de casos de uso, diagramas de clase, diagramas de secuencia. La consistencia pretende que los requisitos de una especificación no se contradigan entre sí. El objetivo del juego es llenar correctamente plantillas predefinidas de cuatro diagramas (esquema pre-conceptual, diagrama de clases, diagrama de casos de uso y diagrama de secuencias), que corresponden al modelo verbal de un problema específico.

En la tabla 1 se presentan las investigaciones descritas anteriormente, indicando de acuerdo a la literatura revisada, si es una herramienta, una metodología o una técnica.

3. La Programación en la FCI

Para determinar los Métodos, técnicas y herramientas que se aplican en la enseñanza de la programación, se diseñó un cuestionario con preguntas aplicadas

en el aula a los alumnos de las carreras de Informática y Computación [25]. El resultado obtenido es el siguiente:

- El lenguaje de programación más usado en la mayoría de los cursos es el lenguaje C/C++ y su principal medioambiente de desarrollo es *borland*, después el lenguaje java y su medioambiente es *NetBeans*[1].
- El lenguaje C es el que predomina en los cursos iniciales y el lenguaje java para los cursos de programación orientada a objetos y la programación visual.
- También mencionan el uso de herramientas para modelado UML como Dia. En cursos relacionados con ensamblador se utilizó en la mayoría PC-Spim y Emu8086.

Tabla 1 Clasificación de investigaciones de enseñanza de programación.

Herramienta	**Técnica /Estrategia**
Herramienta DFD [5]	Mapas conceptuales para enseñar programación [1]
MiniJava [6]	El juego de la consistencia [3]
Editor de algoritmos (genera código en pascal) [9]	Programación Estructurada. Un enfoque algorítmico [15].
Scratch [12]	Estrategia Instruccional [18]
Storytelling Alice [10]	Juego para enseñar I.S. [24]
Programimate [11]	
Herramienta. Implementa 4 paradigmas de programación [12]	
Scratch [13]	
SIGACLE [17]	
ABEA [19]	**Método**
Laboratorio flexible de estructuras de datos [20]	Conocimiento de la arquitectura de computadoras [7]
Tutorial gráfico interactivo de estructuras de datos [21]	Proyecto IDEFIX [8]
Herramienta visualización interactiva de estructuras de datos [22]	Taxonomía de Bloom [14]
Herramienta didáctica para diseñar algoritmos básicos [23]	Metodología basada en contratos [16]

4. Recomendaciones para la FCI

Con base a la investigación de las herramientas que son empleadas para enseñar programación, se listan en la tabla 2, métodos, técnicas y/o herramientas

[1] NetBeans https://netbeans.org/

que pueden ser útiles a los profesores, como apoyo en la enseñanza de los diferentes cursos de programación impartidos en las carreras de Ingeniería en Computación, Ingeniería en Sistemas Computacionales e Ingeniería en Diseño Multimedios de la FCI.

Tabla 2 Sugerencias para cursos de la FCI.

Sugerencia	**Curso**
Laboratorio flexible de estructuras de datos [20]	Estructuras de datos
Tutorial gráfico interactivo de estructuras de datos [21]	Estructuras de datos
Alice	Programación gráfica I
Scratch	Programación gráfica I
Herramienta DFD	Programación I
MiniJava	P.O.O.
ABEA	Programación I
Conocimiento de la arquitectura de computadoras [7]	Programación I

5. Descripción de 3 herramientas sugeridas FreeDFD 1.1.

Editor e intérprete de diagramas de flujo permite editar, ejecutar y depurar algoritmos representados como diagramas de flujo. Útil en la enseñanza de algoritmos básicos, también incluye recursividad, modularidad y arreglos de varias dimensiones que permiten construir algoritmos complejos. Es software libre y puede ser redistribuido y/o modificado bajo los términos de la Licencia Pública General de GNU publicada por la Free Software Foundation, en cualquiera de sus versiones. Sus diagramas de flujo siguen los estándares utilizando símbolos que se conectan por medio de flechas, para establecer la secuencia de una determinada operación o funcionalidad. Los algoritmos desarrollados pueden ser guardados y recuperados en disco y pueden ser impresos en diferentes tamaños.
Con el uso de plantillas predeterminadas facilita la construcción de diagramas de flujo, se pueden agregar símbolos y flechas para establecer secuencias de instrucciones; proporciona una barra de herramientas en la que se pueden seleccionar los distintos elementos que se quieran ir agregando al modelo (figura 1). Soporta las estructuras de programación: “Asignación”, “Ciclo mientras”, “Ciclo para”, “Decisión”, “Lectura”, “Llamada”, “Salida” y ”Nuevo Subprograma”. También permite trabajar con expresiones complejas que involucren constantes, variables, funciones y operadores.

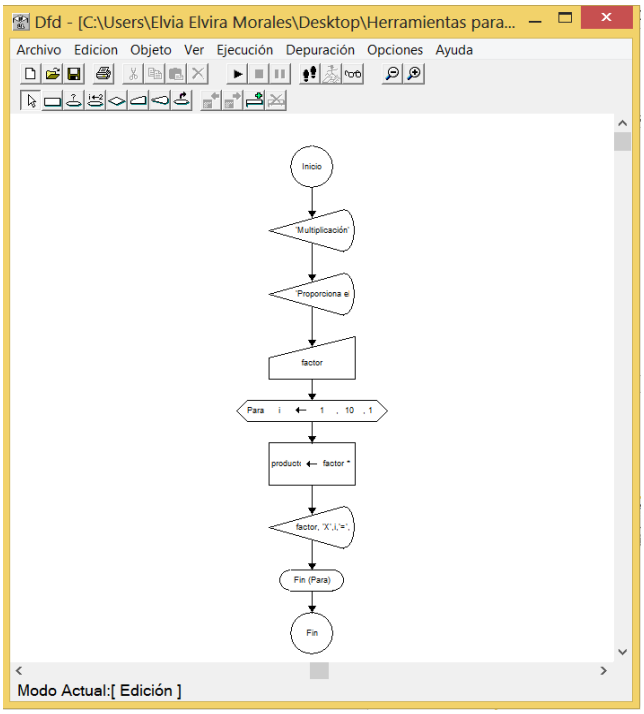

Figura 1 Algoritmo de ejemplo generado en Dfd.

Permite la utilización de arreglos de cualquier dimensión. Existen tres tipos de datos: Real, Cadena de Caracteres y Lógico; además 24 operadores y 19 funciones que operan con datos de estos tipos.

Permite un ambiente de programación estructurada con la utilización de subprogramas que permiten el paso de argumentos por referencia y por valor, con lo que se pueden elaborar y estudiar algoritmos que impliquen recursividad directa e indirecta o simples llamadas para modularizar los procesos [5]. Proporciona la opción de probar los algoritmos elaborados, con las opciones de: "Detener**"** que permite pausar la ejecución del diagrama de flujo, "Depuración de paso simple" que permite una depuración paso a paso, "Ejecutar hasta" permite puntos de ruptura en la ejecución del diseño del diagrama de flujo y "Evaluar" que funciona como una calculadora. Los errores generados en cualquier lugar del diagrama, son detectados y mostrados a través de mensajes, indicando el lugar en el que se presentan. El diagrama de flujo generado sirve como algoritmo que guía la elaboración del programa en la etapa de la codificación usando un lenguaje de programación estructurada.

Programinate 2

Es una herramienta en línea para la enseñanza de programación visual, orientada a programadores novatos, de uso libre, desarrollada en java por el Dr. Andrew Scott [11]. Su objetivo es hacer más fácil la asimilación de la programación en su etapa inicial, proporcionando un entorno amigable, fácil de usar y sin la complejidad de la escritura de la sintaxis de un lenguaje. El entorno (figura 2) permite realizar un diagrama de flujo, generar su código en lenguaje (*Java, VB.NET, VB6, Pascal y JavaScript*), corrida paso a paso y comportamiento de las variables durante la ejecución. En cuanto a programación, permite declarar variables simples y arreglos, realizar procesos, entrada y salida de datos, condicionales simples y dobles e iteraciones con ciclos for y while.

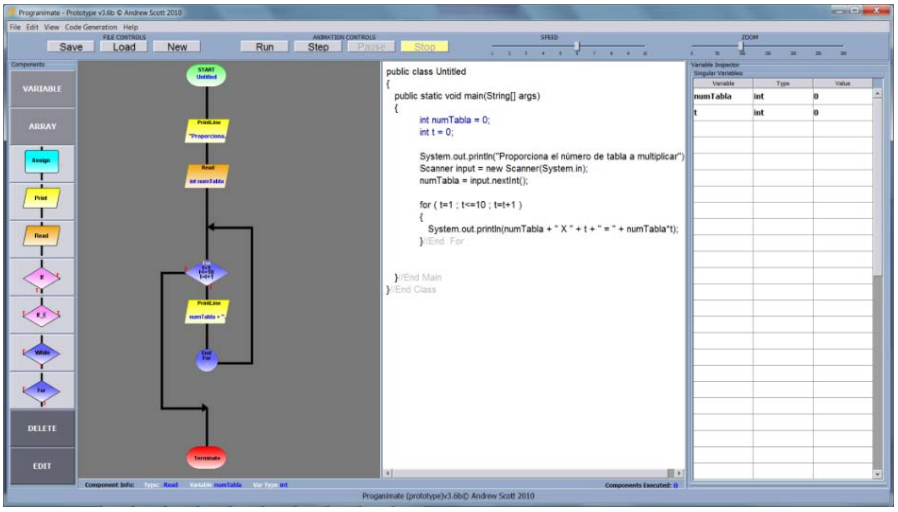

Figura 2 Algoritmo de ejemplo generado en *programinate*.

Scratch

Entorno de aprendizaje que facilita el aprendizaje autónomo de un lenguaje de programación. La gramática está basada en una colección de bloques de programación gráficos. El usuario los coloca juntos para crear programas, como con las piezas de *Lego*, conectores en los bloques indican cómo deben juntarse, el usuario puede comenzar por simplemente pensar con las piezas, colocándolas juntas en diferentes secuencias, hacer combinaciones y ver qué pasa.

[2] Página de ProgrAnimate disponible al 21 de Junio de 2016.

Los bloques de *Scratch* están conformados para encajar de manera que tengan sentido sintáctico. Las estructuras de control se basan en C. La forma de bloques de salida es acorde a los tipos de valores que devuelven: óvalos para números y hexágonos para Booleanos. Bloques condicionales (*if y repeat-until*) tienen forma hexagonal, indicando que un booleano es requerido.

Scratch es altamente interactivo mezclando gráficos, animaciones, fotos, música y sonido, solo hay que hacer clic sobre una pila de bloques y se empieza a ejecutar su código inmediatamente. La herramienta permite: a) hacer cambios a una pila que se está ejecutando, por lo que es fácil de experimentar con nuevas ideas de forma incremental e iterativa, y b) crear hilos paralelos, al crear múltiples pilas de bloques. El objetivo es hacer la ejecución en paralelo tan intuitiva como la ejecución secuencial y su interfaz es semejante a un escritorio físico (figura 3).

La mayoría de los lenguajes de programación (y los cursos de ciencias computacionales) privilegian la planeación de arriba hacia abajo, antes que la de abajo hacia arriba. El énfasis en el diseño iterativo e incremental se alinea con el estilo de desarrollo de *Scratch*.

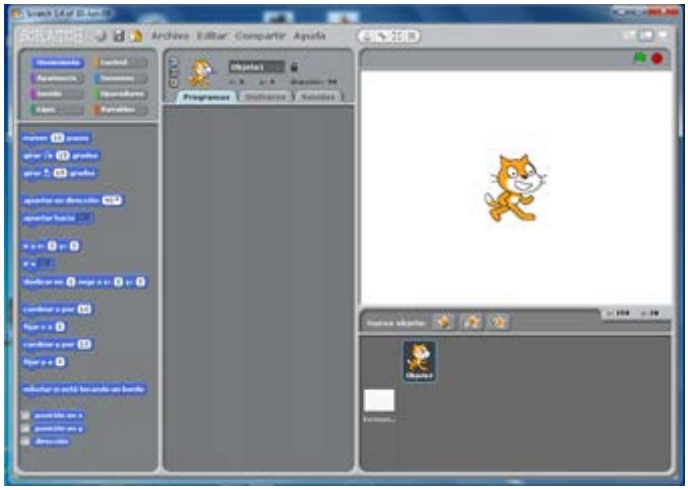

Figura 3 Ejemplo de programa en *S*cratch.

Antes de lanzar Scratch en 2007, se probó continuamente con prototipos en entornos del mundo real, revisando una y otra vez basado en la retroalimentación y sugerencias obtenidas de las pruebas.

7. Conclusiones

Existe la necesidad de mejorar la enseñanza/aprendizaje de la programación en las carreras relacionadas con la informática y ciencias computacionales, tanto a nivel nacional como internacional, en relación a esta necesidad se han realizado diversas investigaciones y propuestas de métodos, técnicas y/o herramientas que apoyan la enseñanza en cursos de programación.

El uso de algunas de estas herramientas de programación puede contribuir a elevar los índices de aprobación en los cursos de programación.

Por parte de las entidades correspondientes en cada una de las escuelas y universidades se tienen que revisar los perfiles de cada programa de estudio y en base a estos, proponer el tipo de paradigma y el lenguaje de programación que contribuyan al logro de los objetivos de dicho programa de estudio.

8. Bibliografía y Referencias

[1] C.I. Chesñevar, “Utilización de mapas conceptuales en la enseñanza de la programación”. Lidecc, 2000.

[2] G. Polya, Cómo plantear y resolver problemas. Reprint. 1965. Trillas. 1965.

[3] C.M. Zapata, M.I. Duarte Herrera, “Consistency game: a didactic strategy for software engineering”. Rev. Téc. Univ. Zulta Vol. 31, No.1. 2008. Pp. 3-12.

[4] L. Mandow, J. Coego, F. Villalba, “Herramienta de software para la enseñanza de algoritmos de búsqueda”. III Jornadas de Innovación Educativa y Enseñanza Virtual en la Universidad de Málaga. 2009.

[5] F. Cárdenas Varela, N. Castillo Izquierdo, E. Daza Castillo, "Editor e intérprete de algoritmos representados en diagramas de flujo". Informática educativa UNIANDES – LIDIE, Vol. 11, No. 1, 1998. pp. 101-106,

[6] E. Roberts, “An overview of MiniJava”. Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education SIGCSE '01. 2001. Pp. 1-5

[7] F. Meza, R. Garrido, C. Astudillo, Un Lenguaje de Bajo Nivel como Apoyo al Aprendizaje en el Primer Curso de Programación de las Carreras de Ingeniería. 2002.

[8] J.E. Labra Gayo, M. Morales Gil, R. Turrado C. Plataforma de enseñanza de lenguajes de programación a través de Internet: Proyecto IDEFIX. 2002. Universidad de Oviedo. España.

[9] N. Moroni, P. Señas, “Estrategias para la enseñanza de la programación”. Primeras Jornadas de Educación en Informática y TICS en Argentina-JEITICS 2005.

[10] C. Kelleher, Motivating Programming: using storytelling to make computer programming attractive to middle school girls. 2006. Carnegie Mellon University. Pittsburgh.

[11] A. S. Scott, M. Watkins, D. McPhee, “Progranimate – A Web Enabled Algorithmic Problem Solving Application”. Proceedings of the 2008 International Conference on E-Learning, E-Business, Enterprise Information Systems, and E-Government. July 2008.

[12] M. A. Cruz-Chávez, J. C. Zavala-Díaz (Eds): CICos 2008, Una herramienta y técnica para la enseñanza de la programación. ISBN: 978-607-00-0165-9. 2008.

[13] M. Resnick, J. Maloney, A. Monroy-Hernández, et all, "Scratch: Programming for All". Communications of the ACM. Vol. 52. No. 11. November 2009. Pp. 60-67.

[14] I. Hernán Lozada, C. A. Lázaro Carrascosa, J. A. Velázquez Iturbide, "Hacia el diseño de herramientas educativas de programación basadas en la taxonomía de Bloom". Proc. Of 5° Simposio Internacional en Informática Educativa (SIIE). 2003. Pp. 761-765.

[15] L. López Román, Programación Estructurada. Un enfoque algorítmico. ISBN: 9701508564, 2da edición. 2003. Alfaomega. México.

[16] M. Blom, E. J. Nordby, A. Brunstrom, “Teaching semantic aspects of OO programming”. Fifth Worshop on Pedagogies and Tools for Assimilating Object-Oriented Concepts, OOSLAP. 2001.

[17] G. F. Cendales, M. P. Díaz, R. J. Barros, "SIGACLE: Simulador grafico asistido por computador para la enseñanza de programación de lenguajes estructurados". IV Congresso RIBIE. 1998.

[18] J. P. Cardona, F. J. Álvarez, J. Muñoz, Estrategia Instruccional para la enseñanza de lenguaje de programación introductorio implementada con objetos de aprendizaje. 2007. Universidad Autónoma de Aguascalientes.

[19] J. J. Arellano Pimentel, O. Nieva García, “ABEA, Herramienta de apoyo en la enseñanza de algoritmos y habilidades de programación”. VI Semana Nacional de Ingeniería en Electrónico (SENIE). Octubre 2010. Pp. 193-202.

[20] J. Villalobos, D. Pérez, J. Castro, C. Jiménez, "Construcción de un Laboratorio Flexible de Estructuras de Datos". En: XIII Congreso Iberoamericano de Educación Superior en Computación. Octubre 2005.

[21] M. J. Cabrero Canosa, J. Blanco Gutiérrez, "Tutorial gráfico interactivo de estructuras de datos". XIII Jornadas de Enseñanza Universitaria de la Informática (JENUI). 2007.

[22] C. Segura, I. Pita, "Una herramienta para el estudio de Estructura de Datos y algoritmos". III Jornada Campus Virtual UCM. 2007.

[23] J. A. Sandoval Bringas, M. Carreño León, I. Estrada Cota, F. Aispuro, J. Suarez Villavicencio, Utilización de una herramienta didáctica para cursos básicos de EdD. 2006. Universidad Autónoma de Baja California Sur. Baja California Sur.

[24] A. Baker, E. Oh Navarro, and A.V. Der Hoek, "An Experimental Card Game for Teaching Software Engineering Processes". Journal of System and Software. Vol. 75. ISSUES 1-2. Feb 2005. Pp 3-16.

[25] P. Zavaleta Carrillo, D. Pérez Cruz, Pérez Rejón, José A, Métodos, técnicas y herramientas para la enseñanza de la programación. Marzo 2014. Universidad Autónoma del Carmen. Campeche, México. Pp. 18-23.

7. Autores

M.C. Zavaleta Carrillo Patricia obtuvo su título de Maestría en Ciencias en Ciencias Computacionales con especialidad en Ingeniería de Software en el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET) en Cuernavaca, Morelos, México.

M.C. Pérez Cruz Dámaris obtuvo su título de Maestría en Ciencias Computacionales con especialidad en Ingeniería de Software en la Universidad de las Américas Puebla (UDLAP), San Andrés Cholula, Puebla, México.

Dr. José Felipe Cocón Juárez obtuvo su título de Doctor por la universidad Rey Juan Carlos mediante el programa de doctorado en Tecnologías de Información y Sistemas Informáticos impartida en la Universidad Rey Juan Carlos- Comunidad de Móstoles, Madrid, España

Dra. Elvia Elvira Morales Turrubiates obtuvo su título de Doctora en Sistemas Computacionales en la Universidad del Sur, Chiapas, México.