

# XI EVENTO INTERNACIONAL “MATECOMPU 2009” “LA ENSEÑANZA DE LA MATEMÁTICA Y LA COMPUTACIÓN”

## Ambiente virtual y Lenguaje de Domino Específico para la enseñanza de la programación

Lic. José Albert Cruz Almaguer<sup>1</sup> email: [jalbert@uci.cu](mailto:jalbert@uci.cu)

Procedencia: Grupo de Investigación de Programación Avanzada (GIPA),  
Universidad de las Ciencias Informáticas, Cuba

### **Resumen**

Aprender a programar computadoras es un tarea difícil, para los que nos dedicamos a enseñarla nunca serán pocos los esfuerzos ni variantes didácticas que hagamos para facilitarlo; conceptos como abstracción, estado, instrucción, estructura de control, flujo de ejecución, toman todos mucho tiempo en ser asimilados a cabalidad por los estudiantes. El presente trabajo pretende aportar herramientas que ayuden al entendimiento de dichos conceptos, constituye una variante al trabajo presentado en FIMAT XXI<sup>2</sup>, a partir de presentar una adaptación al entorno Scratch y lograr un pequeño lenguaje de programación con construcciones sintácticas en Español y posibilidad de actuar sobre un ambiente en ejecución que presente varios objetos con los que se pueda interactuar.

### **Introducción**

En la Universidad de las Ciencias Informáticas se usan, como medio de iniciar a los estudiantes en la programación, los lenguajes C++, Java y C#; cualquiera de estos lenguajes exigen manejar símbolos que constituyen en sí mismos un problema: **if**, **for**, **static**, **void** son sólo algunos ejemplos; especialistas de renombre<sup>3</sup> y diferentes investigaciones<sup>4</sup> se han referido a la importancia del lenguaje y el entorno de desarrollo que se usen en los primeros momentos de la enseñanza y han coincidido en que pueden influir sobre el proceso.

---

<sup>1</sup> Profesor Asistente de la Universidad de Ciencias Informáticas, Cuba

<sup>2</sup> José Albert Cruz Almaguer. PSEUDOCÓDIGO EJECUTABLE PARA LA ENSEÑANZA DE LA ALGORTIMIZACIÓN, FIMAT XXI, mayo de 2009.

<sup>3</sup> Bertrand Meyer. Object-Oriented Software Construction, capítulo 29, páginas 938-940. Prentice Hall PTR, 2nd edition, 1997.

<sup>4</sup> Michael Kölling. The problem of teaching object-oriented programming Partes I y II: Languages. Journal of Object-Oriented Programming, 1999.

Varios autores<sup>5</sup> han considerado necesario el diseño de lenguajes específicos con el único objetivo de enseñar los conceptos básicos de programación, lenguajes con una sintaxis simple que permita centrarse en el problema a resolver y dejar de lado cuestiones de otro tipo.

### **Consideraciones sobre la programación**

Un curso de programación se enfrenta al reto de detallar procesos, aprender a programar es despertar la habilidad intelectual de la previsión, es generar una actitud de anticipar permanentemente las consecuencias de las instrucciones dadas para que se desarrolle en el tiempo un proceso.

Diseñar procesos en detalle implica poner en concordancia ideas: conceptos, requisitos, consideraciones iniciales, etc; esas características exigen gran coordinación mental para lograr plasmarlas en un artefacto que operará independientemente de su creador, de ahí la enorme importancia de usar metáforas, medios, y/o ejemplos adecuados para enseñarlo. Por otra parte, como señalara Dijkstra: *el dominio de la lengua materna (junto a las matemáticas) es la habilidad más importante que necesita un programador*; los símbolos con los que nos expresamos al programar tienen relación con la forma y facilidad con que pensamos, esto hace deseable el uso de palabras y frases en Español para escribir especificaciones, codificar instrucciones y demás actividades asociadas a la construcción de soluciones software.

### **Roles de los lenguajes de programación**

El uso de los lenguajes en el aprendizaje de la programación puede verse en tres dimensiones fundamentales<sup>6</sup>:

- *Medio para instruir a la computadora*: viéndolo como un lenguaje de máquina de altísimo nivel, constituyendo una abstracción del medio de cómputo capaz de resolver muchos problemas. Con la atención en aspectos de la ejecución de los programas, manejo de la memoria etc. (Nivel de implementación)
- *Gestión de la descripción del programa (fuentes)*: usándolo para entender el programa en su conjunto a partir de los recursos de

---

<sup>5</sup> Rosana Satorre et al. *Un modo de entender la programación. X Jornadas de Enseñanza Universaria de la Informática, JENUI 2004, 2004.*

<sup>6</sup> J.L. Knudsen and O.L. Madsen. *Teaching Object-Oriented Programming is more than Teaching Object-Oriented Programming Languages. DAIMI-PB 251, 1990.*

administración del encapsulamiento, modularidad entre otros. (Nivel de especificación)

- *Modelado conceptual*: el más alto grado de modelado de conceptos, relaciones... del conocimiento en el dominio de aplicación en su máxima expresión. (Nivel conceptual)

Salvo muy raras excepciones (que usan la 3ra<sup>7</sup>) los cursos introductorios se basan en desarrollar las dos primeras. Para potenciar la No 1 es deseable que las construcciones sintácticas posean símbolos fáciles de aprender y que no colisionen con significados ya conocidos (como el tan llevado y traído = usado para asignar); con palabras conocidas a la perfección (el uso del Inglés como idioma base para las palabras reservadas deja en desventaja a los que no tengan éste como idioma materno); y estructuras para el control del flujo de ejecución que tengan una clara traducción desde sus componentes sintácticas hasta la especificación de lo que describen como proceso. Como ejemplos del último caso tenemos, en el caso de la iteración, la forma de **while** camuflado que tiene el **for** en los lenguajes de la familia C en oposición al **for** de Pascal, y en el caso de Eiffel la presencia de un **loop** con lugares para ubicar el componente variante de la iteración así como aserciones, entre otras cosas.

En el caso de la segunda dimensión son particularmente útiles las posibilidades declarativas de los lenguajes y en particular la posibilidad de expresar precondiciones, postcondiciones e invariantes; es mediante esto que más se puede desarrollar la muy importante habilidad de leer código que tan poca atención recibe en el diseño de los cursos iniciales.

### **Programar actuando sobre ambientes virtuales**

Estas características de la exigente actividad intelectual de programar computadoras hacen importante el uso de medios que sean estimulantes, motivadores... y que sobre todo, le den al estudiante la posibilidad de ver reflejadas con facilidad las consecuencias de cada línea de código que escriba.

La inclusión de multimedios para estimular el aprendizaje se encuentra entre las variantes más utilizadas, es común en los juegos didácticos en computadoras la

---

<sup>7</sup> Michael E. Caspersen Jens Bennedsen. *Teaching Object-Oriented Programming -- Towards Teaching a Systematic Programming Process*. En: *Proceedings of the Eighth Workshop on Pedagogies and Tools for the Teaching and Learning of Object-Oriented Concepts, 18th European Conference on Object-Oriented Programming (ECOOP 2004)*, junio de 2004, Oslo.

inclusión de vídeo y sonido para estimular varios sentidos a la vez y facilitar la adquisición del conocimiento. En el ámbito de la enseñanza de la programación a novicios se ha empleado la animación (el movimiento) de objetos como aliada para, al lograr realizar ejercicios vistosos, estimular el aprendizaje.

Algunos casos exitosos consultados son:

- Currículo invertido<sup>8</sup>: uso de la librería Traffic.
- Curso de Smalltalk: Squeak: Learn Programming with Robots por Stéphane Ducasse
- Guido van Robot: posibilidad de manipular un robot mediante código escrito en Python.
- Cupi2: Proyecto de una universidad colombiana que desarrolla medios didácticos para enseñar programación.

## Scratch

Scratch es un lenguaje de programación desarrollado por el Lifelong Kindergarten Group en el MIT Media Lab, que facilita enormemente la creación de historias interactivas, juegos y animaciones; posee un entorno de desarrollo rico en cuanto a integración de medios (imágenes, sonidos, animaciones) lo que lo hace ideal para introducir a los niños en el fascinante mundo de la programación. Desde su salida al público en mayo del 2007 ha creado muchas expectativas al proveer al usuario no especializado en informática de un medio de creación de software, fue motivado por la idea de lograr habilidades creativas y tecnológicas en los Computer Clubhouses<sup>9</sup>; es un producto de código abierto basado y desarrollado en Squeak, una versión moderna de Smalltalk, posee una interfaz vía sockets que permite la comunicación con otros sistemas su objetivo original es conectar un dispositivo denominado PicoBoard con el entorno pero en general puede ser usado para cualquier cosa que los desarrolladores interesados necesiten siempre que conozcan el protocolo de comunicación asociado.

Aunque su objetivo inicial ha sido los niños nada impide que se utilice en la enseñanza de programación a futuros profesionales del desarrollo de software, la

---

<sup>8</sup> Bertrand Meyer. *TOUCH OF CLASS: Learning to program well with objects and contracts*. Springer-Verlag, 2009.

<sup>9</sup> Yasmin Kafai, Mitchel Resnick and John Maeda. *A Networked, Media-Rich Programming Environment to Enhance Technological Fluency at After-School Centers in Economically-Disadvantaged Communities*, 200. Disponible online en: <http://web.media.mit.edu/~emres/papers/scratch-proposal.pdf>

universidad de Harvard fue la primera en hacerlo<sup>10</sup> y desde el curso 2008-2009 lo utilizamos en la Universidad de Ciencias Informáticas (UCI) en un curso propedéutico de Algoritmización previo a las asignaturas curriculares de la disciplina de Técnicas de Programación.

### ***Desarrollo***

A partir del anterior estudio se concluyó en la conveniencia de potenciar los roles 1 y 2 anteriormente enunciados así como de un entorno que poseyera objetos con los que se pueda interactuar y cuyas respuestas tengan una connotación gráfica.

### ***Adaptaciones al Scratch***

Una de las características del Scratch que lo hacen de fácil acceso a principiantes es su estilo *arrastrar-y-soltar* de bloques (para codificar las acciones) de rompecabezas que solamente logran ensamblarse si son sintácticamente correctos, durante la primera experiencia en la aplicación (curso 2008-2009) nos dimos cuenta que las etiquetas de varios de los bloques no correspondían con exactitud a los términos utilizados en la literatura o eran totalmente distintos por lo que traían confusión en las posteriores clases (ver Tabla 1).



Bloque en Scratch	Significado
	Para representar la asignación ( En Pascal: <i>numero := 4;</i> )
	Para representar una variación del valor actual ( En Pascal: <i>numero := numero + 2;</i> )

Tabla 1. Ejemplos de bloques con semántica no acorde a sus etiquetas.

Dada la publicación del código fuente para uso no comercial es posible la modificación del entorno para su ajuste o extensión; en nuestro caso decidimos adaptar algunas de las estructuras de control y sobre todo las de manipulación de variables. En la Tabla 2 se muestra la versión original y la modificada de algunos de los bloques.

<sup>10</sup> Henry H. Leitner David J. Malan. *Scratch for budding computer scientists*. 2007. Disponible online en: <http://www.eecs.harvard.edu/~malan/publications/fp079-malan.pdf>




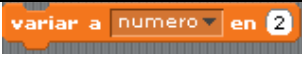



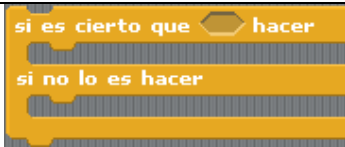
Bloques originales	Bloques modificados
	
	
	
	

Tabla 2. Ejemplos de las modificaciones realizadas.

## El lenguaje

Aunque el programar a base de bloques es ideal para los primeros pasos nunca bastará en la formación de un profesional del desarrollo de software, el uso de lenguajes a base de texto es la norma y una introducción a la programación debe llevar por lo tanto el uso de un lenguaje de este tipo.

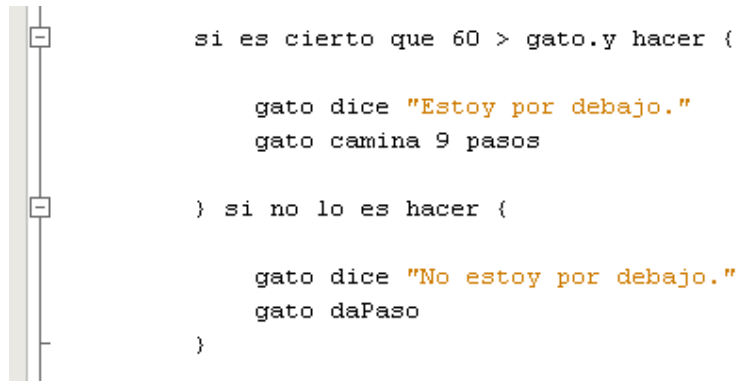
Son múltiples las opciones a la hora de escoger: C#, Java, Python, Ruby, Pascal y un muy largo etcétera, sin embargo, cualquiera de ellos implicaría un enorme salto a partir del Scratch tanto en sintaxis como en semántica; es por ello que se decidió crear un *Lenguaje de Dominio Específico* (LDE) con construcciones equivalentes a las utilizadas en Scratch.

### *Lenguajes de Dominio Específico*

Lenguajes como Java y C# son denominados lenguajes de propósito general pues fueron creados para resolver problemas de cualquier tipo (que admitan una solución algorítmica), los LDE son lenguajes concebidos para tareas particulares (en dominios de aplicación especiales), ejemplos de ellos constituyen el HTML y el SQL.

En nuestro caso se utilizó el lenguaje Scala para desarrollar un LDE con estructuras de control equivalentes a las de Scratch y con objetos que permiten interactuar con programas Scratch, siendo por lo tanto un lenguaje cuyo dominio se restringe a la

escritura de códigos que interactúen con Scratch. En la Figura 1 se muestra la sintaxis de las estructuras si-sino del LDE.

The image shows a snippet of Scratch code. On the left, there is a vertical timeline with two square markers. The code is written in a light gray font. It starts with 'si es cierto que 60 > gato.y hacer {'. Inside this block, there are two lines: 'gato dice "Estoy por debajo."' and 'gato camina 9 pasos'. This is followed by '}' si no lo es hacer {'. Inside this second block, there are two lines: 'gato dice "No estoy por debajo."' and 'gato daPaso'. The entire structure is enclosed in a final closing brace '}'.

**Figura 1.** Fragmento de código del LDE en el que se muestra el uso de la estructura si-entonces-sino clásica bajo otra sintaxis.

### **Enseñanza de conceptos de la Programación Orientación a Objetos (POO)**

Son muchos los debates alrededor de si se debe empezar un curso de programación utilizando la POO o no, en general no existe un consenso absoluto al respecto y el éxito o fracaso al parecer depende de la manera en que se aplique el paradigma escogido y no en el paradigma en sí mismo<sup>11</sup>.

En Scratch no hay soporte para conceptos tales como métodos o clases por lo que no se puede usar para hablar de POO, sin embargo soporta el trabajo con eventos: recepción y envío de mensajes, esto hace que pueda ser usado para la enseñanza del principio de la ejecución de sistemas OO: *la realización de cómputo mediante el intercambio de mensajes entre objetos*.

---

<sup>11</sup> Michael E. Caspersen Jens Bennedsen and Michael Kölling, editores. *Reflections on the Teaching of Programming, Methods and Implementations*, volumen 4821 de LNCS, Capítulos 3: *Experiences with Functional Programming in an Introductory Curriculum* y 6: *Transitioning to OOP/Java — A Never Ending Story*, Springer-Verlag, 2008.

```
gato dice "No estoy por debajo."
```

Figura 2. Instrucción en el LDE



Figura 3. Tratamiento del mensaje correspondiente al método *dice*



Figura 4. Consecuencia de procesar el mensaje en el entorno.

El código de la instrucción de la Figura 2 consiste en texto que se escribiría siguiendo una sintaxis: *<objeto> <métodos/mensajes> <parámetro>*, separados por espacios en blanco.

### **Diseño de un curso de programación para principiantes**

A partir de la experiencia acumulada durante 4 cursos en la UCI utilizando un esquema en el que: primero se desarrolla un curso de nivelación de algoritmia sin codificar en un lenguaje formal; para luego pasar a uno de Introducción a la Programación en el que, utilizando un Lenguaje de Programación Orientado a Objetos de propósito general, se escriben programas del tipo: *dada las edades de tres personas determine la mayor* y cuya respuesta consiste siempre en un mensaje texto sobre una consola de color negro, se ha notado cierta frustración de los estudiantes al no obtener los resultados con los que soñaban y que consideraban los normales antes de entrar a la Universidad, fenómeno ya observado en otros entornos<sup>12</sup> dado que nuestros actuales estudiantes han crecido en medio de los juegos de video y de películas de ciencia ficción.

Se ha desarrollado un marco de trabajo en el que se pueden enseñar los conceptos iniciales utilizando Scratch con todas las facilidades de creación de animaciones, uso

---

<sup>12</sup> Michela Pedroni, Bertrand Meyer, *The Inverted Curriculum in Practice, Proceedings of SIGCSE 2006, ACM, Houston, Texas.*



de multimedia y construcción de los programas que posee, todo ello creando un ambiente virtual acorde a la imaginación e intereses de cada estudiante. En esa etapa de adquisición de los principios del pensamiento algorítmico y de las técnicas más básicas de programación se vería el paso de mensajes como una manera natural de comunicación para luego plantearla como pilar en la concepción de los sistemas OO.

Una vez fijados estos conceptos dentro del entorno gráfico se pasaría a la interacción con los ambientes virtuales creados en Scratch desde un lenguaje basado en textos que use las mismas palabras que tenían los bloques utilizados anteriormente: el LDE; en este esquema se trabaja desde el primer día en un entorno gráfico en el que se pueden hacer con facilidad programas que puedan lograr *moverse un ratón que al tocar a un gato diga: ¡no me vas a comer!*

### ***Conclusiones***

En este trabajo se han presentado algunas de las modificaciones realizadas a Scratch y parte de un Lenguaje de Dominio Específico concebidas para desarrollar un curso de Introducción a la Programación que:

- Utilice siempre interfaz de ventanas y objetos interactivos
- Inicie a los estudiantes en la codificación de instrucciones mediante el ensamblado de bloques con mensajes claros asociados
- Codifique en modo texto utilizando un lenguaje cuyas palabras pertenezcan al Español
- Utilice objetos como medio de asimilación de algunos de los principales conceptos de OO: mensajes/métodos y objetos

Además se ha mostrado una manera de utilizar un entorno que posibilita niveles de interacción entre los estudiantes de manera que estimule el aprendizaje al lograrse resultados gráficos.

En los últimos tiempo el país ha dirigido enormes esfuerzos para convertir a la Industria del Software en una importante fuente de recursos, lograr altos niveles de asimilación de los contenidos de programación en nuestras universidades juega un importante papel para alcanzar este objetivo.

### ***Recomendaciones***

El proyecto es una iniciativa individual en el marco de la Maestría en Ciencias de la Computación de la UCLV, mención Software Educativo, para implementarse adecuadamente requiere desarrollar abundantes materiales extras que incluyen: manuales de usuario, ejercicios de diferentes niveles de complejidad y un folleto que enuncie y desarrolle (libro de texto) los conceptos usados de manera que los estudiantes puedan aprender con mayor rigor; el completamiento de todo esto mas la aplicación a un número mucho mayor de estudiantes serían imprescindibles para la validación y perfeccionamiento de este trabajo.