ALGORITMOS Y PROGRAMACIÓN EN LA EDUCACIÓN ESCOLAR

PONENCIA

Temática del trabajo:
TIC, cognición, aprendizaje y currículo

JUAN CARLOS LÓPEZ GARCÍA

FUNDACIÓN GABRIEL PIEDRAHITA URIBE

editor@eduteka.org
Tel 316-1877
Cali - Colombia

RESUMEN

Utilizar lenguajes de programación en ambientes escolares mediante cursos de Algoritmos y Programación, además de ayudar a desarrollar pensamiento algorítmico, exige que los estudiantes atiendan aspectos importantes de la solución de problemas. En Educación Básica, realizar este tipo de actividades en el aula se ha dificultado por la carencia de materiales que apoyen iniciativas con este enfoque. Conciente de la importancia del tema, la Fundación Gabriel Piedrahita Uribe ha venido trabajando en este campo durante los últimos cuatro años en el Instituto Nuestra Señora de la Asunción (INSA) de Cali. Producto de esta experiencia es la elaboración tanto de una Guía para docentes de Informática, como de un Cuaderno de Trabajo; la primera utilizando el entorno de programación MicroMundos y el segundo con ejemplos y actividades para los estudiantes. Por otra parte, más recientemente y con la misma finalidad anterior, se inició otra experiencia en la Universidad Icesi de Cali con el fin de elaborar y validar materiales para aprender a utilizar un entorno de programación alterno como Scratch; experiencia esta en la que participa un grupo de docentes de Informática pertenecientes a tres Instituciones Educativas de la ciudad.

ALGORITMOS Y PROGRAMACIÓN EN LA EDUCACIÓN ESCOLAR

1. ANTECEDENTES

Existe actualmente, dentro de la comunidad educativa, un consenso general a nivel mundial sobre la necesidad de superar el tipo de enseñanza basada en la transmisión de contenidos y reemplazarla por desarrollar capacidades. Investigaciones y estudios recientes proponen diversos conjuntos de habilidades que la educación debe fomentar para que los estudiantes puedan tener éxito en el mundo digital y globalizado en el que van a vivir. Este planteamiento exige, sin dilaciones, implementar estrategias que contribuyan efectivamente al desarrollo de esas habilidades consideradas como fundamentales para la educación en el Siglo XXI (21stcenturyskills, 2004).

En la mayoría de conjuntos de habilidades propuestos figura el desarrollo de la destreza para solucionar problemas; por esta razón, se requiere seleccionar estrategias efectivas para ayudar a que los estudiantes la adquieran. Para atender esta necesidad, el uso de lenguajes de programación constituye una alternativa efectiva pues compromete a los estudiantes en la consideración de diversos aspectos importantes para solucionar problemas, tales como: decidir sobre la naturaleza del problema, seleccionar una representación que ayude a resolverlo y monitorear tanto sus propios pensamientos (metacognición) como las estrategias de solución utilizadas, aspectos estos que deben desarrollarse desde la edad temprana.

No debemos perder de vista que solucionar problemas con ayuda del computador puede constituirse en excelente herramienta para adquirir la costumbre de enfrentar problemas de manera rigurosa y sistemática, aun, cuando no se utilice un computador para solucionarlos. David Moursund (2006), experto en el tema, se basó en sus propios experimentos y en la teoría de los cuatro estados de desarrollo cognitivo planteada por Piaget, para sugerir que en la etapa de las operaciones concretas los niños empiezan a

manipular lógica y sistemáticamente símbolos en un computador y aprenden a apoyarse en software para resolver un rango amplio de problemas y tareas de tipo general. De esta manera, ganan habilidad considerable tanto en la utilización de lenguajes de programación, como en la manipulación de ambientes gráficos. Posteriormente, en la etapa de operaciones formales, los estudiantes demuestran su inteligencia mediante el uso lógico de símbolos relacionados con los conceptos abstractos que demanda la solución de problemas por medio del computador.

Por otra parte, además de ayudar a fomentar la habilidad para solucionar problemas, un curso de Algoritmos y Programación puede contribuir efectivamente en el desarrollo del Pensamiento Algorítmico de los estudiantes. Este se refiere a la elaboración y uso de algoritmos que puedan favorecer la solución de una clase especifica de problema o la realización de tareas de un determinado tipo. Este pensamiento incluye elementos como: descomposición funcional, repetición (iteración y/o recursión), organización de datos (registro, campo, arreglo, lista, etc), generalización y parametrización, diseño por descomposición de un problema en partes más pequeñas y manejables (top-down) y refinamiento (NRC, 2004).

Dado que un programa se compone de uno o más procedimientos, con instrucciones paso a paso que pueden ejecutarse en un computador, utilizar el diseño de procedimientos que solucionen o ayuden a solucionar problemas con diferentes niveles de dificultad es un recurso que puede aprovechar el docente para captar el interés de los estudiantes. Por ejemplo, asignar la tarea de elaborar un procedimiento que dibuje un cuadrado es relativamente sencilla. Pero el proyecto puede aumentar su complejidad si se añaden nuevas especificaciones como diseñar otro procedimiento que ejecute cuatro veces el primero para dibujar (sin levantar el lápiz) cuatro cuadrados adyacentes entre si que conformen un "cuadrado mayor". Posteriormente, el proyecto puede crecer si se agrega un nuevo procedimiento que ejecute en cascada los procedimientos anteriores y permita girar el "cuadrado mayor" cierto número de grados de manera que se dibuje una flor geometrizada como la que se muestra en la Figura 1.

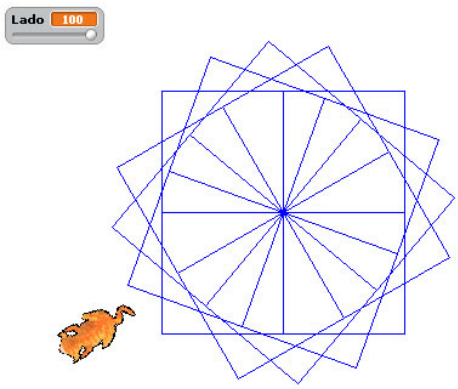

Figura 1: Dibujo realizado mediante la ejecución en cascada de procedimientos

Al igual que en el ejemplo anterior y para mantener motivados a los estudiantes, se pueden diseñar proyectos de clase interesantes cuyas tareas y retos tengan una

complejidad progresiva; proyectos en los que cada reto nuevo parta de la construcción anterior. En resumen, los procedimientos constituyen un tipo particular de tarea que busca solucionar problemas específicos y que al desarrollarlos ponen en juego el pensamiento algorítmico.

Las anteriores, son razones de peso que se encuadran además dentro de los objetivos de la educación; en contraposición con la idea peregrina de utilizar lenguajes de programación en la educación Básica con el objeto de desarrollar en los estudiantes competencias laborales específicas; esto es, formar programadores. Resulta de la mayor importancia insistir en esta orientación debido a que la mayoría de las metodologías utilizadas en educación Básica para realizar cursos de Algoritmos y Programación, son herencias de la educación superior y muchos de los docentes que las usan dedican la mayor parte del tiempo a enseñar los vericuetos de los lenguajes de programación. Hablar hoy de aprender a diseñar y construir aplicaciones complejas, implica una labor titánica que está fuera del alcance de la educación Básica ya que necesariamente demanda hacer uso de lenguajes profesionales (C++, C#, Java, Etc) y abordar alguno de los enfoques de programación complejos que se utilizan en la industria del software (programación orientada a objetos, programación funcional, etc).

Por lo tanto, en este nivel educativo, es recomendable utilizar ambientes de programación como Logo, más fáciles de utilizar y que permiten realizar procedimientos (subrutina o subprograma en forma de algoritmo para resolver una tarea específica) basados en tres estructuras básicas: secuencial, condicional e iterativa.

2. PROPUESTA DE LA FUNDACIÓN GABRIEL PIEDRAHITA URIBE

En palabras de Seymour Papert (1980), creador de Logo, este se define como "un lenguaje de programación más una filosofía de educación" y esta última se relaciona con frecuencia con el "constructivismo" o "aprendizaje a través del descubrimiento"; esto es, aprender haciendo. La Fundación Gabriel Piedrahita Uribe (FGPU) propone la utilización de dos herramientas basadas en Logo con el fin tanto de ayudar a desarrollar el pensamiento algorítmico de los estudiantes como de darles la oportunidad de atender aspectos importantes de la solución de problemas. La primera de ellas es Scratch (http://scratch.mit.edu) desarrollada por el grupo "Lifelong Kindergarten" (http://llk.media.mit.edu) del Laboratorio de Medios del MIT; la segunda es MicroMundos (http://www.micromundos.com), desarrollada por la compañía canadiense LCSI (http://www.micromundos.com/company/profile.html).

Scratch, herramienta gratuita, permite que a partir de proyectos planteados por el docente los estudiantes expresen sus ideas de manera creativa incorporando texto, sonido, vídeo, animación e imágenes. El entorno de programación incluye conceptos recogidos de herramientas como e-Toys, LogoBlocks y MicroMundos (Resnick, Kafai & Maeda, 2007). El lenguaje se basa en Logo y en lugar de tener que escribir instrucciones se interactúa con él arrastrando y soltando bloques gráficos similares a los ladrillos de Lego. Estos bloques, que encajan unos en otros (autoencajables), solo ajustan si son sintácticamente correctos. Esto permite al estudiante concentrarse en la solución algorítmica en lugar de la sintaxis. Una ventaja adicional es que los proyectos elaborados se pueden compartir a través de la página Web de Scratch (http://scratch.mit.edu); a la fecha, hay en ella 126.033 proyectos compartidos que fuera de poderse descargar

libremente, permiten ver su código (scripts). Otra ventaja importante de la herramienta es que los bloques de instrucciones pueden cambiarse a diferentes idiomas, incluyendo el español, sin que la funcionalidad de los scripts se afecte; lo que permite por ejemplo, que las instrucciones de un proyecto elaborado por un estudiante ruso se puedan cambiar automáticamente al español (Prudencio, 2007). En conclusión, Scratch es una buena opción para iniciar a los estudiantes, de cualquier edad, en el mundo de la programación. De hecho, en la Universidad de Harvard realizaron una investigación que corroboró los beneficios de utilizar Scratch como aprestamiento del lenguaje Java en clases de programación, por facilitar la transferencia de conceptos a otros lenguajes (Malan & Leitner, 2007).

Por su parte, MicroMundos, es una poderosa herramienta multimedia dirigida a niños de 6 a 12 años, que permite crear proyectos incorporando vídeos, fotografías, sonidos, gráficos, textos y animación; lo que la convierte en herramienta de trabajo ideal para realizar proyectos educativos con enfoque constructivista.

En MicroMundos, los procedimientos contienen instrucciones que se inician con el comando "para" y que el computador ejecuta automáticamente, una tras otra, hasta encontrar el comando "fin". Esto demanda que los estudiantes planifiquen, formulen hipótesis y anticipen qué sucederá, antes de determinar los comandos que conforman un procedimiento particular, para poder escribirlo, ejecutarlo y por último, comprobar si produjo el resultado esperado. Tal vez, la única desventaja de este software es que tiene costo y muchas instituciones educativas no tienen presupuesto para comprarlo. Sin embargo, también son numerosas las que lo tienen ya licenciado, por ejemplo las del Distrito Especial de Bogotá (http://www.redacademica.edu.co) o las escuelas rurales del departamento de Caldas (http://evirtual.recintodelpensamiento.com/escuelavirtual/).

Aquellas Instituciones que ya poseen MicroMundos o que disponen de presupuesto para adquirirlo, primero, pueden usar como aprestamiento, Scratch y posteriormente, trabajar con MicroMundos para implementar aspectos más avanzados de la programación, no disponibles en Scratch, como son, entre otros: la recursividad, la interactividad con el usuario vía teclado y el manejo de procedimientos con parámetros. Para concluir, las dos herramientas promueven lo que Piaget (1964) denominó "la conquista de la difícil conducta de la reflexión" que se inicia a partir de los siete u ocho años cuando el niño deja de actuar por impulso y empieza a pensar antes de proceder.

3. EXPERIENCIA EN EL INSTITUTO NUESTRA SEÑORA DE LA ASUNCIÓN

En los últimos cuatro años lectivos, se ha llevado a cabo un curso de Algoritmos y Programación con estudiantes de los grados 4° y 5° del Instituto Nuestra Señora de la Asunción - INSA (http://www.insa-col.org) de Cali integrándolo con el área de Matemáticas. Durante este tiempo se le han realizado ajustes buscando la mejor forma tanto de secuenciar los contenidos como de abordar la solución de problemas matemáticos. Como resultado de esta experiencia, se ha evidenciado que cuando los estudiantes resuelven problemas matemáticos retadores con un lenguaje de programación basado en Logo, este pone a prueba la verdadera comprensión que tienen respecto a los conceptos matemáticos involucrados en las soluciones. Este hallazgo, realizado en los dos primeros años, implicó asegurar que los estudiantes de primaria desarrollaran, previa y efectivamente, competencias tanto en comprensión lectora, como en los temas

matemáticos básicos con los que iban a trabajar (geometría).

El trabajo que ahora presenta la FGPU, fruto de más de cuatro años de labor, reflexión, consulta y afinamiento, comprende una *Guía* dirigida a docentes que tienen a su cargo el área de Informática entre los grados 4° y 9° y de un *Cuaderno de Trabajo* enfocado en ofrecer ejercicios para los estudiantes. Estos documentos recogen la experiencia con la que se han logrado integrar en el INSA los lenguajes de programación, mediante las clases de Algoritmos y Programación, en el desarrollo de habilidades para solucionar problemas. La Guía esta compuesta por cuatro unidades y aunque en INSA se utiliza con estudiantes de grados 4° y 5°, puede perfectamente trabajarse con estudiantes de secundaria, basta con que el docente ajuste los ejemplos y ejercicios a las capacidades de sus estudiantes.

En la primera de las unidades que la componen, se presenta un enfoque para educación básica orientado a la solución de problemas con ayuda del computador. En ella, se plantean las cuatro fases que componen el ciclo de programación que a su vez concuerdan con las operaciones mentales descritas por Polya (1957) para resolver problemas matemáticos: Analizar el problema (entender el problema), Diseñar un algoritmo (trazar un plan), Traducir el algoritmo a un lenguaje de programación (ejecutar el plan) y Depurar el programa (revisar). Cabe destacar el marcado énfasis que la Guía hace en la primera fase: Analizar el problema hasta lograr la mejor comprensión posible de este. Para ello, se lo debe formular claramente, especificar los resultados que se desean obtener, identificar la información disponible (datos), determinar las restricciones y definir los procesos necesarios para convertir los datos disponibles (materia prima) en la información requerida (resultados solicitados). Los textos universitarios consultados, aunque reconocen la importancia del análisis de problemas, omiten explicar la forma de trabajarlo en el aula y a esto le dedican únicamente pocas páginas.

La segunda unidad se concentra en los conceptos básicos requeridos para llevar a cabo la fase dos: Diseñar algoritmos que permitan resolver problemas mediante pasos sucesivos y organizados en secuencia lógica (pensamiento algorítmico). Se tratan temas como: ¿qué es un algoritmo?, formas comunes de representarlos (seudocódigo y diagrama de flujo) y, conceptos básicos de programación (variable, constante, identificador, palabra reservada, contador, acumulador, tipos de datos, operadores y expresiones).

La tercera unidad atiende las fases dos y tres del ciclo de programación: diseñar un algoritmo y traducirlo a un lenguaje de programación. En esta se exponen las tres estructuras de control básicas, conocidas como secuencial, iterativa (repetición) y condicional (decisión, selección). Adicionalmente, se explican los fundamentos de programación en el área de procedimientos de MicroMundos.

La última unidad está dedicada a la cuarta fase del ciclo de programación: Depurar procedimientos. Para un estudiante resulta muy difícil, en los primeros intentos, elaborar procedimientos perfectos y la dificultad aumenta a medida que los problemas se vuelven más complejos. Después de traducir el algoritmo a un lenguaje de programación, el procedimiento resultante se debe probar y se deben validar sus resultados (revisión). Este proceso se conoce como depuración y desde el punto de vista educativo estimula en los estudiantes la curiosidad, la perspectiva y la comunicación, además de promover valores como responsabilidad, fortaleza, laboriosidad, paciencia y perseverancia.

Por otra parte, los Anexos de la Guía están compuestos por: el resumen de comandos de MicroMundos utilizados en los diferentes ejemplos; un esquema de los temas tratados; un plan de trabajo con la sucesión de contenidos para trabajar la Guía en el aula y una propuesta curricular que los apoya. Esta propuesta no solo es completa sino que esta planteada para realizarse durante un año lectivo, además está secuenciada de manera que obedezca al orden descrito en el Anexo 3, fruto de las experiencias más recientes en INSA.

En resumen, el principal objetivo de esta propuesta no consiste en lograr que los estudiantes se conviertan en programadores hábiles, consiste más bien en darles la oportunidad de desarrollar pensamiento algorítmico y habilidad para solucionar problemas, capacidades estas que les serán útiles a lo largo de la vida. Esto es posible de alcanzar cuando se *"enseña"* al computador cómo realizar acciones o tareas mediante procedimientos en los que se utilicen estructuras secuenciales, iterativas y/o condicionales. Se logra lo anterior, analizando problemas, diseñando algoritmos, traduciéndolos a un lenguaje de programación y depurando procedimientos sencillos, todo con el objeto de solucionar los problemas planteados. En este orden de ideas, los ambientes de aprendizaje enriquecidos con computadores apoyan muy bien el supuesto de que la mejor manera de aprender es enseñar, confirmado por muchos docentes que aseguran que solo cuando han tenido que explicar un tema a otros, verdaderamente lo han entendido (Jonassen & Reeves, 1996). En estos ambientes, los estudiantes tratan de enseñar al computador lo que ellos deben aprender; autores como Seymour Papert (1980) resaltan que el hecho de tratar de enseñar mejora procesos cognitivos y ayuda a desarrollar habilidades de expresión y de solución de problemas.

Una forma de lograr el manejo de estructuras y conceptos básicos de programación es mediante instrucción acompañada por ejemplos abundantes, ejercicios retadores y actividades que requieran aplicar lo aprendido. Esta propuesta también incluye un Cuaderno de Trabajo que, además de ejemplos y actividades dirigidos a estudiantes de los grados 4° a 9°, contiene algunos de los conceptos básicos de programación expuestos en la Guía para que los tengan a mano. Los ejemplos y actividades propuestos en el Cuaderno de Trabajo, corresponden a temas matemáticos sencillos diseñados para lograr que los estudiantes aprendan a: analizar un problema, descomponerlo en partes, ordenarlas lógicamente, diseñar un algoritmo que represente una solución, traducir el algoritmo a un lenguaje de programación como Logo y verificar la respuesta.

Como propuesta de solución a los escollos que se presentaron en INSA, vale la pena aclarar que la secuencia óptima para presentar a los estudiantes los temas de Algoritmos y Programación, debe ordenarse de modo diferente al expuesto en la Guía. De la misma forma en que algunos expertos aconsejan secuenciar la enseñanza de las matemáticas, la sucesión de contenidos de Programación debe planearse de acuerdo a una estructura helicoidal, en la que los distintos temas se retomen en distintas ocasiones a lo largo del proceso de aprendizaje, de manera que el estudiante pueda comprender e interiorizar progresivamente dichos contenidos. Por lo tanto, en el "Plan de Trabajo" (Anexo 3) se sugiere una secuencia que ayudará a evitar que la primera e importante fase del ciclo de programación, analizar problemas, se convierta en pesada y tediosa para los estudiantes.

Por último y luego de varios años de realizar este curso en INSA se obtuvieron los siguientes resultados: los estudiantes pusieron a prueba la real comprensión de los conceptos matemáticos involucrados en las soluciones planteadas; mejoraron la interpretación de problemas; acompañaron las soluciones con el planteamiento del problema, el análisis de requerimientos y la identificación de datos disponibles; identificaron fácilmente qué hacer cuando se enfrentan a problemas matemáticos; mejoraron en la elaboración de procedimientos secuenciales ó paso a paso, para algunos conceptos matemáticos; demostraron mayor interés por explorar, conocer y utilizar el computador para resolver problemas matemáticos y mejoró en ellos la comprensión de variable, constante, operador y expresión.

4. EXPERIENCIA EN LA UNIVERSIDAD ICESI

Todo parece indicar que la falta de interés que los estudiantes de grado 11 demuestran hacía las ingenierías obedece a la mala formación escolar en Ciencias y Matemáticas. Deficiencia que evidenció claramente la última prueba Pisa (2006), en la cual Colombia ocupó el puesto 53 entre 57 países (OCDE, 2008). A esta realidad debemos agregarle que los cursos de informática y muy especialmente los de programación que se llevan a cabo en los colegios no motivan a los estudiantes a contemplar Ingenierías como la de Sistemas, como opción de vida profesional; por el contrario, esos cursos tienen a espantarlos. Este problema lo detectó el programa de Ingeniería de Sistemas de la Universidad Icesi de Cali, cuyo director se acercó a la FGPU para invitarla a realizar un trabajo conjunto que ayudara a revertirlo.

Así se realizó la búsqueda de una herramienta de programación, preferiblemente basada en Logo, atractiva para los niños, estable en su funcionamiento, fácil de aprender a utilizar y gratuita, para que el costo no fuera obstáculo en su implementación.

Inicialmente se probó el lenguaje KPL (http://phrogram.com/kpl.aspx) pero a los pocos día vendieron la compañía que lo desarrollo y la nueva lo empezó a cobrar. Ante este revés, se reinició la búsqueda que finalmente dio como resultado encontrar Scratch (http://scratch.mit.edu), herramienta altamente valorada en ambientes escolares, que cumple con creces las características deseadas en un lenguaje de programación para estudiantes de educación Básica. En el proceso, también se ensayó Alice (http://www.alice.org), otra herramienta de programación pero orientada a objetos, desarrollada por la Universidad Carnegie Mellon (http://www.cmu.edu). Se estimó, sin embargo, que era apropiada para utilizarse con estudiantes de Básica secundaria o media y que sería altamente deseable que los que ya hubiesen pasado por un curso básico de Algoritmos y Programación con Scratch la usaran.

Una vez definida Scratch como herramienta para iniciar la programación en educación escolar, se inició un proyecto de grado en el que se desarrollan una serie de materiales tanto para que los docentes aprendan a utilizarla, como para que puedan trabajarla con los estudiantes en el aula. Los materiales mencionados comprenden instructivos ilustrados para cada lección (en formato PDF) y videos que muestran cómo elaborar, en 8 lecciones, un juego básico de Super Mario.

Concientes del valor de tener experiencias reales de aula con Scratch, se convocó a 11 docentes de Informática de tres instituciones educativas de Cali (INSA, Comfandi y Corporación Educativa Popular) con el fin de validar estos materiales. La validación

consistió en realizar paso a paso las instrucciones propuestas en los guiones de cada una de las lecciones (8 en total), para poder hacerles las observaciones que cada uno considerara pertinentes.

Vale la pena anotar que en dos de las instituciones educativas (INSA y Comfandi) se trabaja con MicroMundos en primaria y las opiniones de estos docentes resultan especialmente valiosas. Una vez concluida la elaboración de los materiales, estos se pondrán a disposición de los docentes hispanoamericanos, de manera gratuita, en la siguiente dirección de EDUTEKA: http://www.eduteka.org/AlgoritmosIcesi.php

5. DESCARGAS

En EDUTEKA (http://www.eduteka.org/AlgoritmosProgramacion.php), sitio Web de la FGPU, se encuentran disponibles para descarga gratuita, desde Mayo de 2007, los siguientes documentos de Algoritmos y Programación: Guía para docentes (PDF, 1.1MB, 71 Páginas); Cuaderno de Trabajo para estudiantes (PDF, 694KB, 38 Páginas); Ejemplos de MicroMundos (ZIP, 550KB).

A continuación presentamos la Tabla 1 que muestra la cantidad de descargas mensuales que ha tenido cada uno de esos documentos. De los materiales de Scratch no tenemos todavía estadísticas pues aún no se han publicado; esperamos poder hacerlo en septiembre de 2008.

Tabla 1: Estadística de descargas de documentos sobre Algoritmos y programación (en formato PDF)

	Guía Docente	Cuaderno Trabajo Estudiantes	Ejemplos MicroMundos
May-07	34.771	19.245	2.332
Jun-07	17.323	9.031	988
Jul-07	14.324	7.965	963
Ago-07	12.165	7.120	577
Sep-07	17.439	8.676	767
Oct-07	14.699	7.687	602
Nov-07	8.767	4.500	127
Dic-07	4.905	2.403	235
Ene-08	8.104	3.831	490
Feb-08	12.409	6.607	255
Mar-08	10.122	4.730	343
Abr-08	13.313	7.086	335
TOTAL	**168.341**	**88.881**	**8.014**

REFERENCIAS

21stcenturyskills, (2004). Logros indispensables para los estudiantes del siglo XXI. Extraído el 6 de mayo de 2008 desde http://eduteka.org/SeisElementos.php

Jonassen, D. & Reeves, T. (1996). Learning with technology: Using Computers as cognitive tools. Handbook of research for educational communications and technology (pp. 693-719). New York: Macmillan.

Malan, D., & Leitner H. (2007). Scratch for Budding Computer Scientists. Extraído el 20 de Marzo de 2008 desde http://www.eecs.harvard.edu/~malan/publications/fp079-malan.pdf

Moursund, D. (2006). Computational Thinking and Math Maturity: Improving Math Education in K-8 Schools. Extraído el 30 de eEero de 2005 desde http://uoregon.edu/~moursund/Books/ElMath/ElMath.html

NRC, (2004). Being fluent with information technology. Extraído el 30 de Enero de 2005 desde http://www.nap.edu/html/beingfluent/

OCDE, (2008). Evaluación Pisa 2006. Extraído el 15 de Abril de 2008 desde http://www.eduteka.org/Pisa2006Prueba.php

Papert S. (1980). Mindstorms: children, computers, and powerful ideas. New York: BasicBooks.

Piaget, J. (1977). Seis estudios de psicología. Barcelona: Seix Barral.

Polya, G. (1957). How to Solve It. New Jersey: Princeton University Press.

Prudencio, M. (2007). Scratch, una herramienta lúdica de iniciación a la programación. Linux Magazine, 28, 78-82. Obtenido el 8 de Noviembre de 2007, desde http://www.linux-magazine.es/issue/28/078-082_ScratchLM28.crop.pdf

Resnick, M., Kafai, Y., & Maeda, J. (2007). A Networked, Media-Rich Programming Environment to Enhance Technological Fluency at After-School Centers in Economically-Disadvantaged Communities. Extraído el 28 de Enero de 2008 desde http://web.media.mit.edu/~mres/papers/scratch-proposal.pdf