Explorando estrategias alternativas de aprendizaje en la universidad

Pedro A. Willging

Facultad de Ciencias Exactas y Naturales – Universidad Nacional de La Pampa

pedro@exactas.unlpam.edu.ar

Resumen: Se describen varias estrategias utilizadas en un curso universitario de grado del área de las ciencias informáticas para el aprendizaje de lenguajes de programación. Además de los fundamentos y el encuadre teórico, se pasa revista a los resultados observados en las experiencias recientes y las que están en desarrollo. Se discuten y analizan implicancias y generalizaciones para otros ámbitos educativos y áreas de conocimiento.

I- Introducción

El aprendizaje de los lenguajes de computación está viviendo un renacer en los últimos años a consecuencia de anuncios de reformas en planes curriculares, que incluyen asignaturas de informática en la enseñanza obligatoria de los niños desde edades muy tempranas, y también por la necesidad de la industria que requiere mano de obra calificada en el área del software y la ingeniería informática (CODE.org; Curtis, 2013). El saber elaborar código para programar computadoras y otros dispositivos electrónicos se ha convertido en parte esencial de la nueva alfabetización digital.

Pero tanto el fracaso como el abandono son desafortunadamente frecuentes en los cursos de informática/ciencias de la computación (Bennedsen & Caspersen,2007; Cernuda del Río, Hevia, Suárez, & Gayo, 2013). Por ello es que los profesores de ciencias de la computación están a la búsqueda de métodos para enseñar lenguajes de programación en modos más eficientes y atrapantes, especialmente para aquellos estudiantes cuyas carreras no son las ciencias de la computación/informática, pero que tienen en su curricula el requerimiento de tomar cursos de

computación o programación. Propuestas innovadoras, como el uso de robots educativos podrían ser la fuente de experiencias de aprendizaje memorables y efectivas. Ello puede luego generar ganancias cognitivas y mejoras de aprendizaje significativas (Bergin & Reilly, 2005). Cualquier progreso en el desempeño de los estudiantes que transiten estas nuevas aproximaciones metodológicas se transformarán en experiencias exitosas que pueden ser replicadas en otras situaciones educativas.

Con mayor frecuencia las carreras de grado (de áreas distintas a la informática) incluyen cursos de lenguajes de programación o software computacional que son obligatorios para completar los requisitos de graduación. Esos estudiantes en general no están muy motivados a tomar cursos de computación, ya que los ven como algo complementario y es difícil lograr que se involucren con su aprendizaje (Forte & Guzdial, 2005). Algunos estudiantes dejan el curso antes de finalizarlo, y regresan al año siguiente para intentarlo nuevamente, lo que produce retraso en la graduación, frustración, e incluso abandono de la institución. Esta es la razón principal por la cual se necesitan métodos de aprendizaje mas eficaces.

II- Marco teórico/antecedentes

Comencemos por considerar el éxito y fracaso en los cursos de computación. “Pasar o fracasar” los cursos del primer año en la universidad “juega in rol mayor en el moldeado de las expectativas futuras y las emociones” de los estudiantes (Hawi, 2010, p. 1127). Hay una variedad de factores que pueden señalarse para justificar el alto grado de deserción que se observa en los cursos iniciales de informática/computación. Algunos de esos factores se relacionan con las expectativas de los estudiantes acerca de los cursos de informática, que son distintas de la realidad, demandando entre otras cosas un sólido bagaje en matemática, niveles de abstracción elevados y considerables habilidades para resolución de problemas en general. Las tareas que se proponen en los cursos de programación están organizadas para que los estudiantes desarrollen habilidades cognitivas que posiblemente no hayan desarrollado antes en los colegios secundarios

(Hernandez et al, 2010). El proceso de aprender un lenguaje de programación es una tarea a completar de manera lenta y gradual (Dijkstra, 1988) y los lenguajes de programación disponibles hoy en día son muy complejos, con construcciones sintácticas adecuadas para programadores profesionales, pero inadecuadas como herramientas para enseñar los fundamentos básicos de la programación (Jenkins, 2002; Motil & Epstein, 1998; Robillard & DeLine, 2011)

Existe una escasez de estudios de investigación que examinen las atribuciones causales para los logros en la disciplina de la programación de computadoras. Uno de esos pocos estudios recientemente publicado encontró que los estudiantes con alto rendimiento atribuyen su éxito a una “estrategia apropiada de aprendizaje”. La muestra en este caso fueron 45 estudiantes que cursaron “Programación de Computadoras 1”, un curso introductorio de programación para estudiantes de negocios en la Universidad Notre Dame del Líbano. Entre las razones mas citadas por los estudiantes para éxito o fracaso están: estrategia de aprendizaje, falta de estudio y método de enseñanza apropiado (Hawi, 2010).

Consideremos ahora algunas de las estrategias que se intentan para facilitar el aprendizaje de los lenguajes de programación, incluyendo los usos de tecnologías de diversa índole para arribar a metodologías más eficaces.

El uso de las computadoras en educación tiene ya un recorrido histórico mensurable. Más reciente es la incorporación de la conectividad. Las tecnologías de la información y comunicación (TIC) facilitan el desarrollo de ambientes de aprendizaje que invitan a colaborar con otros de un modo mas informal, flexible y exploratorio. Estos son lugares apropiados para incentivar la experimentación y la reflexión acerca de los estilos de aprendizaje (Coll & Monereo, 2008). Los ambientes formativos basados en computadoras proveen características específicas y distintivas que los hacen apropiados para estrategias de enseñanza. Dentro de las características que los convierten en poderosas herramientas para enseñar a aprender está la necesidad de planear, definir y revisar las decisiones para obtener los resultados esperados. Otras es la promoción de una interacción dinámica con los objetos de aprendizaje y sujetos que comparten sus

producciones y diálogos mientras adquieren información y conocimiento. Este proceso les permite observar los cambios, aprender a partir de los errores, descubrir las acciones mentales llevadas a cabo, y actuar como dispositivos meta-cognitivos que muestran el recorrido del trabajo realizado (Coll & Marti, 2001; Badia & Monereo, 2005). Las capacidades multimediales e hipermediales de las tecnologías computacionales aumentan las posibilidades de aprender nuevas maneras de manejo de conocimiento debido a la versatilidad de los formatos de representación de datos y la facilidad de crear y modificar las redes de conocimiento (Badia & Monereo, 2008).

Los estudiantes que habitan las aulas del siglo 21 son parte de la generación de "juegos de pantalla" que se caracteriza por la búsqueda de estímulo multimedial y multisensorial constante, la permanente necesidad de novedad para evitar el aburrimiento, la ubicuidad de las pantallas, la devolución inmediata a las acciones con respuestas de tipo reacción, y la disponibilidad de una variedad de dispositivos digitales móviles para redes sociales entre otras características (Prensky, 2007; Aldrich, 2009).

De acuerdo a un reporte sobre los usos y hábitos de los jugadores de video-juegos españoles, el 23% de los españoles son jugadores activos, el 71.5% de los hogares tiene una computadora, y el 34.7% tiene al menos una consola de juegos (GfK Emer, 2009). Los niños comienzan a usar juegos de consola y computadora antes del ingreso al sistema escolar, mayoritariamente para entretenimiento. Junto con Internet y la TV, los video-juegos, juegos de computadora y móviles son las fuentes preferidas de entretenimiento de la sociedad juvenil. Ellos se pasan largas horas en frente de las pantallas cada día.

El entretenimiento en el siglo 21 involucra una mezcla de interacciones virtuales y reales por medio de dispositivos como la Wii (una consola de video-juego producida por la compañía Nintendo), mundos imaginarios y fantasía. En algunos casos el entretenimiento puede trasmitir contenidos educativos o motivar indagación científica. La combinación de educación y entretenimiento ha sido denominada como "edutainment" (por la combinación de los vocablos en ingles "education" -educación- y "entertainment"- entretenimiento-), y hoy en día: "aprendizaje y

entretenimiento ya no son actividades separadas" (Burbules, 2009). La investigación ha mostrado que las historias de ciencia ficción motivan a los estudiantes a emprender carreras científicas. Del mismo modo que las historias de *sci-fi*, los juegos prometen estimular la imaginación, elevar la curiosidad, promover discusión y debate, y permitir la experimentación e indagación (Squire & Jenkins, 2004).

Los juegos pueden utilizarse como una estrategia educativa, fundada en la motivación inducida en los niños y su interés natural por el juego (Morfi & Minetti, 2010). El aprendizaje basado en los juegos ha sido listado como una de las tecnologías educativas a ser adoptadas en los próximos años (Johnson, Smith, Willis, Levine, & Haywood, 2011). Los investigadores afirman que ejecutar juegos puede mejorar: atención visual, comprensión de reglas, concentración, razonamiento, resolución de problemas, habilidades sociales, y desarrollo intelectual (Gee, 2003). Niños de 5 a 7 años de una escuela de La Rioja (Argentina) participaron de una prueba piloto en la que utilizaron 3 juegos "orientados a entrenar procesos cognitivos de control inhibitorio, memoria de trabajo y planificación" (Lopez y Rosenfeld, 2012, p.5) creados con la asistencia de un grupo de investigación de neurociencias. Por medio de la asistencia del programa Conectar-Igualdad (un programa similar a One Laptop Per Child- ver http://one.laptop.org- implementado en los colegios públicos de la Argentina- http://www.conectarigualdad.gob.ar-), que fue la plataforma que permitió la difusión de la experiencia, se pudo comprobar que es posible implementar de manera masiva estos juegos.

Otra experiencia concreta de utilización de juegos en el ámbito universitario que puede citarse es la de Hernández *et al*.(2010), en la cual estudiantes de un curso de programación inicial en la Universidade Cruzeiro do Sul (San Pablo, Brasil) usan un motor de juegos para crear sus propios juegos de computadora, aprendiendo las habilidades de programación fundamentales sin necesidad de aprender la sintaxis ni las particularidades de ningún lenguaje de programación en particular.

Como antecedente local, en la Facultad de Ciencias Exactas y Naturales de la Universidad

Nacional de La Pampa, se utilizó el software Scratch en un curso-taller de resolución de problemas, destinado a ingresantes de la Facultad (Willging, Astudillo, & Bast, 2010). Esta experiencia se continúa repitiendo en los cursos para ingresantes todos los años, debido al éxito de la propuesta, en la cual se ha logrado constatar que la herramienta utilizada es intuitiva, simple, de bajo consumo de recursos computacionales, y que genera, entre los estudiantes y docentes un clima de competencia y juego, con mucha motivación y entusiasmo.

A pesar de que la investigación acumulada muestra que las tecnologías educativas tales como la hipermedia o los juegos pueden mejorar el aprendizaje de los estudiantes y de que se han hecho muchos esfuerzos para preparar a los estudiantes de carreras de profesorados y magisterio para que integren las tecnologías en sus prácticas diarias, los profesores (maestros) no son aun capaces de producir experiencias aúlicas donde la tecnología es empleada con éxito (Angeli & Valanides, 2005). Esto puede deberse al hecho de que la implementación de las tecnologías como herramientas pedagógicas involucra un cambio en el rol del profesor, moviéndose del rol tradicional centrado en el profesor hacia un modelo de aprendizaje centrado en el que aprende (Jonassen, 2000; Hannafin & Land, 1997). Este cambio de paradigma no ocurre espontáneamente, y por lo tanto los programas para formadores deberían incorporar oportunidades de capacitación para mejorar las habilidades de enseñanza en contextos basados en tecnología (Kramarski & Michalsky, 2010).

III- Aproximación metodológica/procedimientos

El planteo de la propuesta pedagógica se enmarca bajo el principio de que nuevas metodologías o experiencias innovativas pueden ser aplicadas para motivar e involucrar a los estudiantes a aprender de manera entusiasta temas o conceptos que pueden percibir no son el foco principal de sus estudios (Gold, 2010). En este trabajo, se describe una secuencia de experiencias desarrolladas en un curso de lenguaje de programación. Este es un curso introductorio de programación, requerido para los estudiantes de grado de la licenciatura en física. Se implementaron modos alternativos de desarrollo de actividades aúlicas. El objetivo central de las

innovaciones metodológicas fue lograr que los estudiantes se involucraran activamente con su propio aprendizaje al mismo tiempo que lograr un clima de aprendizaje lúdico y creativo.

Para contextualizar las condiciones que sitúan la experiencia educativa-investigativa, se trata de un grupo de estudiantes en un número que oscila año con año entre 5 y 10. El lenguaje de programación que exige la currícula es lenguaje C, y para las actividades prácticas se utiliza el entorno de desarrollo integrado Dev-C++ (ver http://orwelldevcpp.blogspot.com.ar/). Los estudiantes tienen clases en el laboratorio de computación, donde cada uno trabaja en alguno de los equipos disponibles (PCs) o bien en sus propias computadoras portátiles. Los estudiantes están acostumbrados en general al uso del campus virtual de la Facultad, pues son usuarios en algún curso previo (cuanto menos lo han utilizado en el curso para ingresantes, que tiene una porción en formato virtual), y acceden desde sus hogares a los materiales y la ejercitación, empleando sus propias computadoras para trabajar en la resolución de los problemas de las actividades prácticas.

Además del ambiente de trabajo para lenguaje C, se utiliza el software Scratch, que es un lenguaje de programación “tipo lego” con el que se pueden crear animaciones y juegos; los cuales pueden compartirse en la web por medio del sitio http://scratch.mit.edu. Scratch es un proyecto desarrollado por el Lifelong Kindergarten Group en el Laboratorio de Medios del Massachusetts Institute of Technology (MIT), y por medio de su uso, niños y jóvenes de todo el mundo crean y comparten sus proyectos mientras aprenden conceptos matemáticos y computacionales, a pensar creativamente, a razonar sistemáticamente y a trabajar colaborativamente (Resnik et al., 2009).

La filosofía de la iniciativa es que por medio de los “Proyectos de Ejemplo”, los aprendices pueden experimentar e iniciar los sus primeros proyectos propios (¡copiar a los demás está bien!).

Figura 1: Captura de pantalla del entorno de programación del software Scratch.

El robot utilizado en el curso es un Modulo N6 de RobotGroup (ver http://www.robotgroup.com.ar/index.php/productos/131-robot-n6#documentación), equipado con un controlador DuinoBot compatible con Arduino, y puede ser programado de forma nativa (software residente en el microcontrolador del robot) tanto en C/C++ como en miniBloq (Minibloq es un ambiente gráfico de programación para Multiplo™, Arduino™, dispositivos computacionales físicos y robots ver http://blog.minibloq.org/) . También es posible programarlo en otros lenguajes interactivos de alto nivel de código abierto.

```
void setup()
{
  float speed = 0;
  motor0.setClockwise(false);

  while(speed < 100)
  {
    motor0.setSpeed(speed);
    motor1.setSpeed(-speed);
    delay(1000);

    speed = speed + 10;
  }

  while(speed > 0)
  {
    motor0.setSpeed(speed);
    motor1.setSpeed(-speed);
    delay(1000);
    speed = speed - 10;
  }
}

void loop()
{}
```

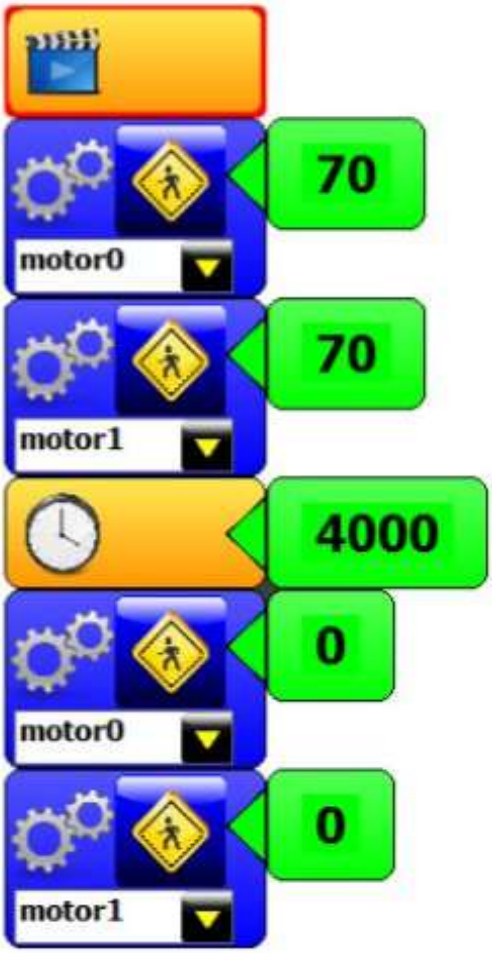

Figura 2: Capturas de pantalla con muestras de código generados en los ambientes de programación Arduino y Multiplo.

IV- Resultados

Se describen a continuación los cambios implementados en el curso y los resultados obtenidos con las distintas innovaciones aplicadas. El relato se divide en tres *actos*, que representan los diferentes ciclos de incorporación y ajuste de la metodología, recursos y actividades.

Primer acto

Tradicionalmente, las clases de programación tienen una estructura que incluye la presentación teórica de los conceptos por parte del profesor, siguiendo algún texto de cabecera y la posterior resolución de ejercicios prácticos donde los estudiantes aplican esos conceptos. Las evaluaciones consisten en la resolución de ejercicios similares a los que integran los trabajos prácticos durante la cursada.

El primer cambio en la metodología del curso consistió en la utilización del software Scratch durante las primeras 2 semanas de clase, como una manera lúdica de introducir a los estudiantes a los principios de programación. Esto permitió introducir de manera inmediata a los estudiantes en un ambiente de programación, ya en los primeros 15 minutos de clase, los estudiantes pueden crear una animación que ha sido programada por ellos mismos. Esto constrasta notablemente con el modo tradicional en que se “enseña a programar”, ya que allí, los estudiantes llegan a elaborar un programa y ver sus resultados recién después de un largo proceso escalonado de acumulación de conceptos tales como constantes, variables, asignaciones, comparaciones lógicas, etc.

Otro cambio introducido en el curso fue la incorporación de un proyecto final como elemento de evaluación. Este proyecto final consiste en la preparación de una actividad que se pretende integradora de lo aprendido durante el curso. Los estudiantes definen su propio proyecto, que en general es un programa, un poco mas extenso y complejo que los que han desarrollado en las prácticas, que resuelve algún problema relacionado con la Física. Se incorpora además, el uso de una plataforma virtual (Moodle), como elemento soporte de las actividades presenciales, con foros para consultas, y con materiales complementarios (ver Figura 3).

Figura 3: Captura de pantalla que muestra el espacio virtual del curso de programación.

Segundo acto

Los cambios introducidos en la metodología, si bien fueron en principio muy “tímidos” y con una gran preponderancia de los procedimientos tradicionales, alentaron a continuar agregando modificaciones en la curricula. Se consolidó el uso de Scratch, que permite a los estudiantes la inmersión en un lenguaje de programación de manera inmediata, para recién después iniciarlos en el aprendizaje de los conceptos básicos de programación estructurada y particularmente del lenguaje C. Como un nuevo elemento de innovación, se experimentó con un robot programable, el cual puede ser controlado por medio de código programado en Arduino. El lenguaje de programación del robot, si bien tiene particularidades relacionadas con el movimiento de las ruedas, los sensores de distancia, los leds o los sonidos que emite la placa que lo controla, es

muy próximo al lenguaje C con el que los estudiantes están ya familiarizados al momento en que se les presenta la propuesta. Se propone a los estudiantes la creación de programas sencillos para hacer “funcionar” al robot como parte de su proyecto final. Es así que los estudiantes son provistos con el robot, los manuales en formato digital y un tutorial básico para que ellos mismos investiguen su funcionamiento y testeen una batería de programas incluidos en el tutorial. Esta actividad se realiza en formato grupal, con todos los estudiantes participando de modo colaborativo. Los estudiantes prepararon una serie de programas para que el robot realice un circuito con un recorrido sobre una pista que ellos mismos armaron (ver Figura 4), para presentar en una jornada de ciencias (la facultad tienen eventos periódicos en los cuales ofrece a la comunidad -particularmente para estudiantes de las escuelas de nivel primario y secundario- actividades de demostración de lo que se realiza en los laboratorios y clases). De este modo, los estudiantes debieron presentar y explicar el funcionamiento del robot y el código que lo controla a otros estudiantes y público que asistió a la jornada científica.

Figura 4: Pista y robot dispuestos para la demostración.

Tercer Acto

Con la incorporación de los cambios en la metodología que significaron el uso de una plataforma virtual (Moodle), el programa Scratch, el robot programable, y los proyectos finales como parte de la evaluación, se han logrado mejoras notables en la producción de los estudiantes, su motivación y los logros obtenidos. En la siguiente iteración del modelo de mejoras de la metodología, se va a incorporar el armado de una computadora a partir de un kit (ver Kano project http://www.kano.me/), que funciona con la filosofía del armado de legos. Aquí se propone que los estudiantes vean desde "adentro" como funciona una computadora y como es posible programar aplicaciones de un modo muy simple. Entre las ventajas de esta innovación, se cuenta que es un proyecto global de "crowdsourcing" destinado a difundir la programación entre personas de todas las edades, especialmente los niños, que hace la programación algo entretenido y motivador. Esta es una experiencia aun en desarrollo, se planea completarla en el transcurso de el presente ciclo lectivo y tener los primeros resultados para el año próximo.

V-Conclusiones

Se ha descripto una experiencia de incorporación de tecnología y cambio en las metodologías de trabajo y aprendizaje en un curso introductorio de lenguajes de programación. La investigación llevada a cabo toma registro de actividades de innovación pedagógica desarrolladas a lo largo de tres años, con cambios graduales en la curricula, aumentando la distancia con el enfoque tradicional de enseñanza a medida que se afianzaron los procedimientos y se ajustaron las actividades de acuerdo al feedback y los resultados obtenidos en cada uno de los ciclos previos. La cuarta iteración de este modelo de integración tecnológica y mejoramiento del ambiente de aprendizaje continuará profundizando en la senda marcada, esto es, centrar el aprendizaje en el estudiante, motivarlo a producir evidencias de su aprendizaje, que puedan ser compartidas y valoradas por sus colegas, incentivar la creatividad y desarrollar los procesos cognitivos que despiertan la utilización de herramientas novedosas para el aprendizaje.

Si bien algunos de los recursos tecnológicos empleados en esta experiencia se adaptan para el caso particular del aprendizaje de los lenguajes de programación, la metodología general y el

modelo de integración y cambio en la curricula puede ser adoptado y ejercitado en otros campos de conocimiento.

VI-Referencias/Bibliografía

Aldrich, C. (2009). Learning online with games, simulations, and virtual worlds. Strategies for online instruction. San Francisco, CA: Jossey-Bass.

Angeli, C., & Vanalides, N. (2005). Preservice teachers as ICT designers: an instructional design model based on an expanded view of pedagogical content knowledge. Journal of Computer-Assisted Learning, 21(4), 292-302.

Badia, A., & Monereo, C. (2005). Aprender a aprender a través de Internet. En C. Monereo (Coord.), Internet y competencias básicas (pp. 51-71). Barcelona: Graó.

Badia, A. & Monereo, C. (2008). La enseñanza y el aprendizaje de estrategias de aprendizaje en entornos virtuales. En C. Coll & C. Monereo (Eds) Psicologia de la educacion virtual (348-367). Madrid: Ediciones Morata, SL.

Bergin, S., & Reilly, R. (2005). The influence of motivation and comfort-level on learning to program, en Proceedings 17th Annual Workshop of the Psychology of Programming Interest Group, 293-304, University of Sussex, Brighton, UK.

Bennedsen, J., & Caspersen, M. E. (2007). Failure rates in introductory programming, ACM SIGCSE Bulletin, 39 (2), Jun. 2007 [doi>10.1145/1272848.1272879]

Burbules, N. (Mayo 24, 2009). Artículo en Clarin, Suplemento Zona, 38-39.

Cernuda del Río, A., Hevia, S., Suárez, M. C., & Gayo, D. (2013). Un estudio sobre el absentismo y el abandono en asignaturas de programación, ReVision, 6(1), http://www.aenui.net/ojs/index.php?journal=revision&page=article&op=viewArticle&path[]=114&path[]=180

CODE http://code.org/promote

Coll, C., & Martí, E. (2001). La educación escolar ante las nuevas tecnologías de la información y la comunicación, en C. Coll, J. Palacios y A. Marchesi (comps.), Desarrollo psicológico y educación. 2. Psicología de la educación escolar, Madrid, Alianza, 623-655.

Coll, C. y Monereo, C. (Eds.) (2008). Psicología de la educación virtual. Aprender y enseñar con las Tecnologías de la Información y la Comunicación. Madrid: Morata.

Curtis, S. (4 de noviembre de 2013). Teaching our children to code: a quiet revolution. The Telegraph, Tech, Technology News, http://www.telegraph.co.uk/technology/news/10410036/Teaching-our-children-to-code-a-quiet-revolution.html

Dijkstra, E. W. (1988). On the Cruelty of Really Teaching Computing Science (EWD-1036). E.W. Dijkstra Archive. Center for American History, University of Texas at Austin. http://www.cs.utexas.edu/users/EWD/ewd10xx/EWD1036.PDF

Forte, A., & Guzdial, M. (2005). Motivation and nonmajors in computer science: identifying discrete audiences for introductory courses, IEEE Transactions on Education, 48(2), 248-253.

GfK Emer. (2009). Estudio aDeSe 2009 “Usos y hábitos de los videojugadores españoles”. Obtenido de la web, Noviembre 12, 2011, http://www.adese.es/pdf/PPThabitos122009.pdf

Gee, J. P. (2003). What Video Games Have to Teach Us about Learning and Literacy. New York: Palgrave Macmillan.

Gold, N. (2010). Motivating Students in Software Engineering Group Projects: An Experience Report, Innovation in Teaching and Learning in Information and Computer Sciences, 9(1), 10-19.

Hannafin, M. J., & Land, S. M. (1997). The foundations and assumptions of technology-enhanced student-centered learning environments. Instructional Sciences, 25, 167-202.

Hawi, N. (2010). Causal attributions of success and failure made by undergraduate students in an introductory-level computer programming course, Computers & Education, 54, 1127-1136.

Hernandez, C.C., Silva, L., Alencar Segura, R., Schimiguel, J., Fernández Paradela Ledón, M., Naito Mendes Bezerra, L., & Frango Silveira, I. (2010). Teaching Programming Principles through a Game Engine, CLEI Electronic Journal, 13 (2), Universidade Cruzeiro do Sul, São Paulo, Brazil, http://www.clei.org/cleiej/papers/v13i2p3.pdf

Jenkins, T. (2002). On the difficulty of learning to program. Proceedings of 3rd Annual LTSN_ICS Conference. U.K. The Higher Education Academy, 53-58.

Johnson, L., Smith, R., Willis, H., Levine, A., & Haywood, K., (2011). The 2011 Horizon Report. Austin, Texas: The New Media Consortium.

Jonassen, D. H. (2000). Computers as mindtools for schools: Engaging critical thinking (2nd ed.). Upper Saddle River, NJ: Prentice-Hall.

Kramarsky, B., & Michalsky, T. (2010). Preparing preservice teachers for self-regulated learning in the context of technological pedagogical content knowledge. Learning and Instruction, 20, 434-

447.

Lopez y Rosenfeld, M. (2012). Mate Marote: plataforma educativa de juegos para el entrenamiento de competencias cognitivas. Tesis de Licenciatura, Universidad Nacional de Buenos Aires. Disponible: *www.dc.uba.ar/inv/tesis/licenciatura/2012/**lopezyrosenfeld**.pdf*

Morfi, M. L., & Minetti, M. V. (2010). Historia del juego. Learning Review Latinoamérica, 32(9).

Motil, J., & Epstein, D. (1998). JJ: A language designed for beginners (less is more). Disponible: http://www.publicstaticvoidmain.com/JJ_A_Language_Designed_For_Begin-ners_LessIsMore.pdf.

Prensky, M. (2007). Digital game-based learning. St. Paul, MN: Paragon House.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for All. Communications of the ACM, 52 (11), 60-67.

Robillard, M.P. & DeLine, R. (2011). A field study of API learning obstaclesEmpirical Software Engineering, 16(6), 703-732, Springer, US. http://dx.doi.org/10.1007/s10664-010-9150-8

Squire, K. & Jenkins, H. (2004). Harnessing the power of games in education. Insight (3)1, 5-33.

Willging, P.A., Astudillo, G. J., & Bast, S. (2010). Aprender a programar (¿y a pensar?) jugando. Memorias V Congreso de Tecnología en Educación y Educación en Tecnología, TE&ET 2010, Universidad Nacional de la Patagonia Austral, El Calafate, Santa Cruz, Argentina.