International Studies on Law and Education 29/30 mai-dez 2018
CEMOrOc-Feusp / IJI-Univ. do Porto

Studio-Based Learning, una metodología al servicio de la enseñanza de los lenguajes de programación

García-Jiménez, Sergio[1]; Roig-Vila, Rosabel[2]

[1]Instituto de Enseñanza Secundaria Jiménez de la Espada, Cartagena (Murcia), sergio.garcia3@murciaeduca.es

[2]Universidad de Alicante, rosabel.roig@ua.es

Resumen: En el presente artículo se describe la metodología *Studio-Based Learning*, cuyas características ayudan que los alumnos aprendan de forma activa y colaborativa. Veremos cómo favorece que se cumplan todos los niveles de la taxonomía de Bloom y analizaremos su idoneidad en la enseñanza de los lenguajes de programación. Por último, describiremos algunos ejemplos en los que se ha aplicado este método con éxito.
Palabras Clave: Studio-Based Learning, lenguajes de programación, taxonomía de Bloom.

Abstract: The present paper describes the Studio-Based Learning methodology, the characteristics of which help students to learn both actively and collaboratively. Our focus will be placed on the extent to which this methodology contributes to the fulfillment of all the levels in Bloom's taxonomy, additionally analyzing its suitability for the teaching of programming languages. The paper will conclude with a description of several examples where this method has already been successfully applied.
Keywords: Studio-Based Learning, programming languages, Bloom's taxonomy.

1. Introducción

Las metodologías basadas en *Studio-Based* tienen su origen en la escuela pública de EEUU (Lackney, 1999). Aunque no se denominó *Studio-Based Learning* (SBL), a finales del siglo XIX se implementó en Massachusetts, un modelo que abogaba por centrarse más en el alumno. Aunque se conoció como método Quincy, la motivación y el interés de los estudiantes eran primordiales, se trataba de un currículo integrado, de aprender haciendo y de expresarse a través del arte.

El SBL es una metodología de enfoque constructivista, es decir, afirma que el estudiante construye el conocimiento de manera activa, no pasivamente por medio de clases o libros de texto (Ben-Ari, 2001). Este modelo ha sido utilizado en muchas disciplinas especialmente en arte y arquitectura (Brown, 2006; Lopez, 2003), de manera que, para incentivar el aprendizaje activo, los alumnos, bien por grupos o de manera individual, solucionan problemas propuestos.

En el presente artículo se hará una descripción de en qué consiste el método, sus principales ventajas en educación y se describirá detalladamente a fin de que pueda ser implementado por cualquier lector.

2. Una descripción del SBL

El nombre de Studio-Based Learning[1] trata de evocar la imagen de estudiantes trabajando en proyectos reales en un aula de estudio, a la vista de otros estudiantes y de los profesores. El modelo funciona mejor cuando se realiza en un aula dedicada, pero esto no es imprescindible, ya que este modelo se puede adaptar a infinidad de entornos. Para implementar con éxito el SBL, se requieren grupos de alumnos dispuestos a colaborar y trabajar en equipo y un docente cuya principal responsabilidad es la de diseñar experiencias de aprendizaje (Ahmad & Gestwicki, 2013; Gestwicki & Ahmad, 2011).

El modelo necesita dos elementos claves, que los alumnos diseñen su propia representación de su propio aprendizaje y que esta representación sea compartida con sus iguales y con expertos tanto en un contexto formal como informal. Para obtener una realimentación tanto de sus compañeros como del profesor, se utilizan las denominadas "*design crits*" (abreviatura de *critiques*), donde se presentan las soluciones a las que los alumnos han llegado. Esto incentiva la reflexión personal y la interacción social.

Estas "*design crits*" permiten a los estudiantes adquirir habilidades de comunicación que les serán de gran utilidad en el futuro. Desarrollarán la capacidad de presentar, racionalizar, depurar y criticar soluciones a problemas concretos (Hundhausen, Narayanan, & Crosby, 2008).

Derivado de la disciplina de la arquitectura (Brown, 2006), el SBL es un enfoque pedagógico que aplica técnicas desde la arquitectura y el arte a la ciencia, este el caso de la informática (véase la Figura 1). SBL es una variación del aprendizaje basado en problemas (*Problem-Based Learning*, PBL) (McKeachie & Svinicki, 2006). El PBL, implica exponer un problema a resolver, lo más probable es que los alumnos trabajen en grupo, y, posiblemente, al final, se discutirán las soluciones. SBL es similar al PBL, pero le añade otro enfoque, los utensilios y actividades creados por los alumnos, como base para la discusión (McKeachie & Svinicki, 2006). Las sesiones de discusión, conocidas como "*design crits*", tienen lugar tras el proceso de búsqueda de soluciones al problema clave. El objetivo principal de ambos estilos de aprendizaje, SBL y PBL, es mezclar la teoría que se está estudiando con la práctica y la implementación real.

Según Carter y Hundhausen (2011), SBL contiene cuatro características clave. Son los siguientes:

1. Las tareas se asignan por medio de proyectos.
2. Se realizan evaluaciones periódicas, tanto formales como informales, por medio de las "*design crits*".

[1] https://sites.google.com/site/appinventorsbl/about/studio-based-learning

3. Los estudiantes participan por pares en la evaluación y la crítica.
4. Las "*design crits*" deben girar en torno a actividades creadas por la propia disciplina.

El SBL facilita el proceso de realimentación, que es esencial para cualquier proceso de aprendizaje. Además, asegura el aprendizaje activo, por medio de prácticas en el laboratorio (McKeachie & Svinicki, 2006). El aprendizaje activo se centra en mantener a los estudiantes activos e involucrados en el proceso de aprendizaje. Por otro lado, no sólo ayuda a los estudiantes a aprender el material que se enseña, sino que también les ayuda a aprender cómo pensar así, y adquirir el conocimiento (McKeachie & Svinicki, 2006; Rotenberg, 2010). Transforma el aprendizaje en un objetivo intrínseco para los estudiantes que, a su vez, aumenta su motivación para aprender y los transforma en aprendices activos, incluso fuera de la institución educativa. SBL se centra en la creación de un ambiente de clase activa en la que se introducen los conceptos y se retienen mejor los conceptos; un ambiente donde los estudiantes están activos, participativos, e interactúan con el profesor y, más importante aún, con otros estudiantes. Muchos trabajos de investigación han demostrado que los estudiantes aprenden más de sus compañeros de lo que aprenden de su profesor (McKeachie & Svinicki, 2006). En comparación con los cursos tradicionales de disertación, se demostró que el SBL era más beneficioso en términos de calificaciones y resultados y términos de actitud de los estudiantes (Carter & Hundhausen, 2011).

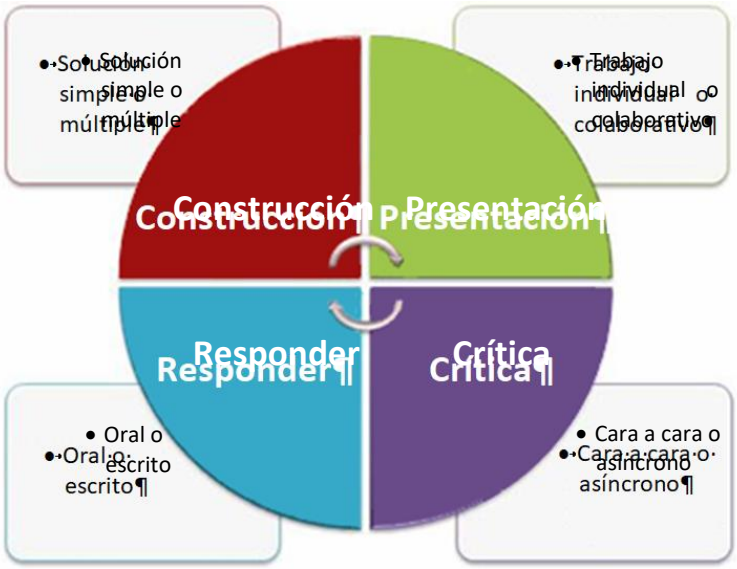

Figura 1: Modelo de *Studio-Based Learning*.

La taxonomía de Bloom es un esquema, que implica el dominio cognitivo para la clasificación de los objetivos del curso y conjuntos de preguntas y problemas para la discusión en clase o exámenes (Rotenberg,

2010). Se clasifica en seis niveles que van desde el más simple al más complejo (Reynolds, Livingston, & Willson, 2009). El SBL reúne casi todos los niveles de la taxonomía de Bloom (Sutherland & Bonwell, 1996); dando una experiencia más completa de formación. El SBL, además, incorpora práctica, análisis, síntesis, y evaluación; en los niveles más altos en la taxonomía (Sutherland & Bonwell, 1996). Las clases tradicionales no exceden de los dos primeros niveles de la taxonomía de Bloom (Reynolds et al., 2009; Rotenberg, 2010). Esos dos niveles combinados implican solamente memorizar y entender, no se extiende la experiencia de aprendizaje para incluir la aplicación y el análisis.

La gran versatilidad de este modelo permite adaptarlo a las necesidades particulares de cada caso, siendo independiente de la tecnología utilizada, lo que permitirá utilizar aquella con la que el profesor se sienta más cómodo (Hundhausen et al., 2008).

Además, al ser un modelo de enfoque constructivista y de acuerdo con algunos autores (Ben-Ari, 2001), se supone que ésta técnica va a tener más éxito que aquellas con un enfoque tradicional.

Todo esto, hace pensar que el SBL tiene todas las características para ser un modelo de éxito, encontrando en la literatura experiencias donde los estudiantes respondieron muy favorablemente (Carter & Hundhausen, 2011). De hecho, hay muchos estudios implementados en SBL con resultados muy positivos (Ahmad & Gestwicki, 2013; Docherty, Sutton, Brereton, & Kaplan, 2001; Estey, Long, Gooch, & Gooch, 2010; Gestwicki & Ahmad, 2011; Hendrix, Myneni, Narayanan, & Ross, 2010; Hundhausen et al., 2008; Hundhausen, Agrawal, Fairbrother, & Trevisan, 2009; Hundhausen, Agrawal, Fairbrother, & Trevisan, 2010).

3. El SBL aplicado a la enseñanza de un lenguaje de programación

La enseñanza de los lenguajes de programación por medio de metodologías tradicionales, resulta del todo insuficiente a la hora de preparar a los alumnos para su desempeño a nivel profesional y laboral. En este contexto, no sólo es suficiente con saber programar, además, el programador tiene la necesidad de disponer de suficientes habilidades comunicativas, colaborativas y de pensamiento crítico (Hundhausen et al., 2010). El mayor problema con el que nos encontramos en este aspecto, es que los cursos de programación, muchas veces son impartidos utilizando las mismas metodologías que se utilizan en otras áreas como matemáticas e ingeniería (Docherty et al., 2001). Cooper y Cunningham (2010) creen que la educación de las matemáticas se centra en la teoría y las técnicas formales, con poca conexión con las muchas áreas de las que vienen o en las que se pueden usar esas teorías o técnicas. Docherty et al. (2001) sostienen, sin embargo, que la informática tiene características que la hacen diferente de esas dos disciplinas; por ejemplo, los criterios para medir el éxito en un curso de informática no están claros. Afirman que la informática como disciplina está más cerca de las ciencias de diseño, tales como la arquitectura, por lo tanto, sostienen que se deben seguir los métodos de enseñanza utilizados en este tipo de disciplinas de diseño, como

por ejemplo el *Studio-Based Learning* (SBL). Carter y Hundhausen (2011) comparten este argumento y afirman que el SBL es fácilmente adaptable a la disciplina de la informática. Por lo tanto, otro enfoque que hoy se está implementando en los cursos de informática, es el *Studio-Based Learning* (SBL).

Una de las razones del bajo interés por la informática y las bajas tasas de retención es la imagen antisocial que tiene la disciplina (Hundhausen et al., 2008), la imagen de un informático como una persona, cuestionada socialmente, sentado frente al ordenador todo el día. Hundhausen et al. (2008) afirman que esta imagen se debe a la asociación de la informática con la programación, ya que muchos de los primeros cursos de informática tienden a centrarse principalmente en la programación y el aprendizaje de la sintaxis de un lenguaje. La industria exige programadores eficaces que hagan este enfoque comprensible e incluso necesario. Sin embargo, incluso después de tomar muchos cursos de informática que se centran en la programación, los estudiantes todavía tienen pobres conocimientos de programación (Woodley & Kamin, 2007). Woodley y Kamin (2007) sostienen que el aprendizaje hábil de programación es muy parecido a aprender a escribir bien: el estudiante tiene que recibir información detallada, regrabar y recibir más realimentación. La escritura puede no ser la mejor analogía aquí, ya que se realiza principalmente en forma aislada; los profesionales en informática trabajan en equipos de diseño, no sótanos, por lo que las habilidades de colaboración son esenciales. Sin embargo, de manera similar a la escritura, se puede decir, que la programación requiere mucha práctica con realimentación. Esto implica que el profesor tendría que evaluar meticulosamente cada ejercicio y escribir amplia realimentación y repetir el proceso de nuevo. Ese proceso podría ser agotador, así como llevar mucho tiempo. El SBL, con su entorno social y sus sesiones de "*design crits*" periódicas, pueden ser una solución. Lewis, Yasuhara y Anderson (2011) argumentan que la integración del aprendizaje cooperativo y colaborativo podría cuestionar los estereotipos culturales y establecer expectativas más precisas sobre las profesiones de informática.

Hundhausen et al. (2008) encuentran que SBL encaja perfectamente en cursos de informática de todos los niveles. Una de las razones por las que la mayoría de los estudios ponen en práctica el enfoque SBL es para dar a los estudiantes una experiencia similar a la realidad de la industria (Carter & Hundhausen, 2011) . Estudios recientes han demostrado que el SBL mejora la motivación y el interés de los estudiantes en informática, así como ayuda a los estudiantes a desarrollar sus habilidades de comunicación (Hundhausen et al., 2008). Las experiencias de aprendizaje de los estudiantes han sido en general positivas en entornos de SBL (Carbone & Sheard, 2002), y aunque los estudios de impacto longitudinales de SBL están todavía en curso, los resultados preliminares son positivos. En general, estas implementaciones mostraron un futuro muy prometedor en el tratamiento de los problemas en los cursos introductorios a la informática.

4. Algunos casos donde se ha implementado el SBL con éxito

Dadas las ventajas y la versatilidad con la que cuenta el modelo, un gran número de universidades y centros educativos se han decantado por implementar de manera exitosa este modelo. Veamos algunos casos.

En Australia, la Universidad de Monash, como parte de un programa de Tecnología de la Información (TI), puso en práctica el SBL en una materia trocal anual (Carbone & Sheard, 2002). La asignatura implementaba un plan de estudios integrados, que obligaba a los alumnos a utilizar de manera transversal contenidos y competencias de otras materias básicas. El objetivo de la asignatura era preparar a los estudiantes para sus carreras de TI. Para la experiencia se realizó en tres ambientes distintos. El primero consistió en dos estudios en los que se realizó el trabajo en equipo, el segundo fue cibercafé para reuniones informales y sociabilización y por último una sala de reuniones diseñada como un espacio profesional para consultas, reuniones y presentaciones. La última semana del primer semestre se llevó a cabo una encuesta en la que participaron 132 estudiantes de primer año. Los resultados de la encuesta indicaron la preferencia de este modelo sobre las clases impartidas de forma tradicional, sobre todo debido a su enfoque práctico de aprendizaje. Los estudiantes de primer año indicaron que era una experiencia de aprendizaje positiva y encontraron el acogedor ambiente de estudio, útil para su aprendizaje. Estos resultados indican que el SBL se puede aplicar con éxito en cursos introductorios.

En Urbana-Champaign en la Universidad de Illinois, aprovecharon las características del SBL para enseñar informática a un grupo de estudiantes de nivel intermedio (Woodley & Kamin, 2007). La experiencia consistió en clases de una hora y una sesión de discusión (*design crits*) de dos horas con frecuencia semanal. Cada dos semanas se les daba tareas de programación vagamente especificadas y, por último, un proyecto final que duró cuatro semanas. Las sesiones de discusión consistieron en presentaciones de 20-25 minutos durante los que cinco estudiantes presentaban sus trabajos. Woodley y Kamin encontraron que estas sesiones de discusión tienen varios efectos. Para evitar hacer el ridículo, los alumnos trabajaron más duro en sus presentaciones, además, dedicaron mayor esfuerzo en escribir el código de forma clara a fin de poder explicarlo más fácilmente, "lo que produce casi siempre, un código mejor" (p. 533), y, al depender de ellos mismos, se evitaron las trampas. Las sesiones de discusión fueron dirigidas por profesores, ayudantes graduados o estudiantes universitarios que obtuvieron buenos resultados en anteriores ediciones de este curso. Como prueba del éxito de la experiencia, los investigadores utilizaron los testimonios de los alumnos que aseguraban que el curso fue de ayuda para mejorar su manera de programar. Aparte de eso, este estudio no presentó ningún resultado concreto.

En la Universidad de Auburn en otoño de 2008, Hendrix et al. (2010) adaptaron el SBL en un curso de 2º de informática sobre estructuras de datos. Además de dar clases teóricas, implementaron SBL en las sesiones de laboratorio del curso donde los estudiantes se reunían dos veces por semana para trabajar en proyectos en grupo. El lenguaje de programación utilizado en este curso fue el Java. Para evaluar su enfoque, los investigadores realizaron un

pre y post-test para evaluar el dominio de los conceptos de la informática básica, así como un pre y post "Cuestionario de Estrategias de Aprendizaje y Motivación" (CEAM) y un pre y post "Cuestionario de Sentimiento de Comunidad" (SCQ) para evaluar las actitudes. Como se esperaba, al comparar el antes y después de los ensayos, los estudiantes mostraron una mejoría en su aprendizaje. Al comparar el antes y después del CEAM y el SCQ, los resultados mostraron un aumento estadísticamente significativo en la motivación intrínseca, la motivación extrínseca, la auto-eficacia, el aprendizaje por pares, la regulación del esfuerzo, y en las escalas del SCQ. En la primavera de 2009, los investigadores recogieron datos de un mismo curso en el que se enseñó de forma tradicional. Compararon el post-test del curso SBL con el post-test del curso tradicional, no se encontraron diferencias estadísticamente significativas. Sin embargo, cuando se correlacionaron las notas de los estudiantes, se encontró un aumento estadísticamente significativo en los estudiantes que realizaron el curso adaptado con SBL. Esto indicó que, si bien ambos cursos enseñan de manera suficiente en cuanto al dominio de conceptos básicos de informática, el supuesto adaptado con SBL hizo un trabajo mejor en el proceso de enseñanza (Carter & Hundhausen, 2011). Hendrix et al. (2010), sin embargo, no aplicaron CEAM en el curso tradicional.

En la Universidad Estatal de Washington, Hundhausen et al. (2009) quisieron aplicar el SBL asignando a los estudiantes tareas de programación individuales. Para ello, se desarrollaron lo que ellos llamaron "Revisión de Código Pedagógica" (RCP), en el que los estudiantes revisan entre ellos las soluciones individuales de código, y luego se juntan en equipos para identificar, analizar y registrar los problemas de código (Hundhausen et al., 2010). El RCP se llevó a cabo en sesiones de laboratorio, por medio de la contratación de estudiantes graduados. Sin embargo, aun conservaban las clases teóricas, como parte de su enfoque. Hundhausen et al. (2009) afirmaron que su enfoque sería proporcionar las habilidades de colaboración, comunicación y pensamiento crítico, fundamentales para la profesión de programador, a los estudiantes que destacasen y tuviesen totalmente adquiridas las habilidades de programación. Los expertos evaluaron la enseñanza de conceptos informáticos por medio de este enfoque en la primavera de 2008 en un curso de informática usando C como lenguaje de programación. Catorce estudiantes participaron en este estudio (12 hombres y 2 mujeres). Las RCPs se llevaron a cabo tres veces durante el semestre. Después de cada RCP se pasó una encuesta final. Los resultados indicaron que los estudiantes, al final del semestre, parecían centrarse más en los problemas de diseño que en los de depuración y formato. Al avanzar el curso, éstos comprobaron la mejora en la calidad de sus trabajos y encontraron beneficiosa la colaboración.

De nuevo en la primavera de 2009, Hundhausen et al. (2010) repitieron el estudio en el mismo curso. En este estudio, sin embargo, los investigadores compararon su enfoque experimental con el tradicional (es decir, sin RCPs). Además de las encuestas finales realizadas en el estudio anterior, se realizó un pre y post-test, un pre y post-CEAM y un SCQ, y una entrevista final. De un total de 176 alumnos matriculados en ambos cursos, 87 en el experimental y 89 en el tradicional, sólo 42 de estos estudiantes participaron en las encuestas

CEAM y SCQ, y 12 en la entrevista final. Se realizó un ANOVA con los datos recogidos sin diferencias estadísticamente significativas.

Hundhausen et al. (2010), sin embargo, señalaron algunas tendencias hacia resultados significativos. En los resultados de las encuestas, encontraron una disminución significativa en la escala de "Autoeficacia" en el curso tradicional, y un aumento en la escala de "Aprendizaje por pares" en el curso experimental en comparación con el tradicional. En las prácticas aumentó el rendimiento, tanto en el curso experimental como en el tradicional. Los resultados de las entrevistas, indicaron que los estudiantes disfrutaron de los cursos y las actividades de programación que más aportaban a su aprendizaje. Sin embargo, los estudiantes participantes del curso experimental necesitaron recibir comentarios positivos en mayor medida que los del tradicional y se sintieron más cómodos colaborando entre ellos. Con respecto a la RCP, la mayoría de los estudiantes las encontraron útiles y realizaron las prácticas en equipo.

En la Universidad de Victoria en Canadá, Estey et al. (2010) implementaron el enfoque SBL en un curso de diseño de juegos. El propósito de su estudio era promover la colaboración y la comunicación entre los estudiantes. Su enfoque, además, no era puramente SBL ya que, como parte del método de enseñanza los investigadores daban clases. Como en otros estudios anteriores, se utilizaban las clases para introducir la nueva información y se complementaba con un enfoque de SBL en el laboratorio. La gestión del curso se llevó a cabo por medio de Moodle: se gestionaban los grupos, creaban foros, y daban cuestionarios online. La herramienta utilizada para desarrollar los proyectos de juegos de los estudiantes fue Flash. Los investigadores pasaron las primeras semanas introduciendo a los estudiantes en el uso de Flash y ActionScript a través de una serie de tutoriales. En este curso, se pidió a los estudiantes que completasen dos proyectos importantes, uno en la mitad del semestre y otra cerca del final. En ambos proyectos los estudiantes crearon juegos a partir de cero en cinco etapas: concepto del juego, prototipos de tres juegos, y una presentación práctica. Cada etapa implicó un proceso de revisión por pares. Al final del semestre, los investigadores implementaron una encuesta en la que los estudiantes podían expresar su acuerdo o desacuerdo a las preguntas, por medio de una escala de Likert de cinco puntos. Los resultados indicaron que los estudiantes respondieron positivamente al curso. Según las encuestas, los estudiantes mejoraron sus perspectivas y su motivación y, así como en el sentido de comunidad gracias al enfoque SBL.

Göttel y Schild (2011) estudiaron también un método para cultivar la creatividad en un curso de diseño de juegos. Su método era muy similar a la metodología de SBL. El objetivo de su trabajo era que los participantes probaran los juegos centrándose en la creatividad, en lugar de en los aspectos técnicos. En este curso de diseño de juegos, los estudiantes fueron divididos en tres grupos en los que se requirió que cada grupo crease un juego. El curso se llevó a cabo en una sala a la que los investigadores llamaron "cuarto de la creatividad 5555". Al igual que en con las "*design crits*", el propósito de esta sala era proporcionar un espacio para que los miembros del grupo se reuniesen y discutiesen sus proyectos en un entorno social sin necesidad de ordenadores.

Por otra parte, los estudiantes utilizaban este espacio para crear prototipos de sus juegos y presentarlos en cuatro etapas. Además, los investigadores reservaron dos aulas de ordenadores tradicionales, aunque, ninguno de los estudiantes los usó, prefiriendo llevar sus propios portátiles al "cuarto de la creatividad 5555".

Los investigadores se reunieron a menudo con los estudiantes en esta sala y les explicaron diferentes temas, como técnicas para el intercambio de ideas. Los investigadores observaron que los estudiantes disfrutaron con la idea de tener un entorno creativo específico. Göttel y Schild encontraron que se hacía necesario tener un lugar de reunión creativo y lúdico, especialmente en un contexto de diseño de juegos. Afirman que aunque medir la creatividad es complicado, los juegos resultantes de la experiencia contenían características no convencionales, haciendo el juego atractivo para el usuario. Se observó que los diseños del juego estaban bien pensados y estructurados y ampliamente revisados. Sin embargo, afirmaron que no se apreciaba una buena planificación y organización del trabajo en equipo, cuando se encontraban en la sala 5555. Junto con sus observaciones, los investigadores llevaron a cabo tres encuestas al final del semestre. Se llevaron a cabo dos encuestas a los estudiantes que hicieron el curso y una en los visitantes que asistieron a las demostraciones de los proyectos finales de los estudiantes. Los resultados de las encuestas indican que los visitantes reconocieron elementos creativos en los juegos y, que los estudiantes apreciaron la atmósfera informal y agradable que les permitía intercambiar información con los otros participantes. El estudio mostró que el "cuarto de la creatividad 5555" desempeño un papel activo en el proceso de diseño y el desarrollo del juego.

Cabe mencionar, sin embargo, que la mayoría de los estudios que hasta ahora han implementado SBL, lo han usado para reforzar material de clase (Carter & Hundhausen, 2011). Vale la pena investigar; un caso en el que la metodología de enseñanza principal utilizada en el curso sea SBL y las clases se utilicen para reforzar las sesiones de SBL. Se hacen necesarios estudios comparativos adicionales (Carter & Hundhausen, 2011), donde SBL sea comparado con otras metodologías. Creen, por otra parte, que, la interacción entre las clases y el SBL debe estudiarse más a fondo. Este estudio se centra en el SBL como la metodología de enseñanza principal, usando pequeñas clases sólo como refuerzo para las sesiones de SBL.

5. Bibliografia

Ahmad, K., & Gestwicki, P. (2013). Studio-based learning and app inventor for android in an introductory CS course for non-majors. *Proceeding of the 44th ACM Technical Symposium on Computer Science Education,* 287-292.

Ben-Ari, M. (2001). Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching, 20*(1), 45-73.

Brown, J. S. (2006). New learning environments for the 21st century: Exploring the edge. *Change: The Magazine of Higher Learning, 38*(5), 18-24.

Carbone, A., & Sheard, J. (2002). A studio-based teaching and learning model in IT: What do first year students think? *ACM SIGCSE Bulletin, 34*(3) 213-217.

Carter, A. S., & Hundhausen, C. D. (2011). A review of studio-based learning in computer science. *Journal of Computing Sciences in Colleges, 27*(1), 105-111.

Cooper, S., & Cunningham, S. (2010). Teaching computer science in context. *Acm Inroads, 1*(1), 5-8.

Docherty, M., Sutton, P., Brereton, M., & Kaplan, S. (2001). An innovative design and studio-based CS degree. *ACM SIGCSE Bulletin, 33*(1) 233-237.

Estey, A., Long, J., Gooch, B., & Gooch, A. A. (2010). Investigating studio-based learning in a course on game design. *Proceedings of the Fifth International Conference on the Foundations of Digital Games,* 64-71.

Gestwicki, P., & Ahmad, K. (2011). App inventor for android with studio-based learning. *Journal of Computing Sciences in Colleges, 27*(1), 55-63.

Göttel, T., & Schild, J. (2011). Creativity room 5555: Evoking creativity in game design amongst CS students. *Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education,* 98-102.

Hendrix, D., Myneni, L., Narayanan, H., & Ross, M. (2010). Implementing studio-based learning in CS2. *Proceedings of the 41st ACM Technical Symposium on Computer Science Education,* 505-509.

Hundhausen, Agrawal, Fairbrother, & Trevisan. (2009). Integrating pedagogical code reviews into a CS 1 course: An empirical study. *ACM SIGCSE Bulletin, 41*(1), 291-295.

Hundhausen, Agrawal, Fairbrother, & Trevisan. (2010). Does studio-based instruction work in CS 1?: An empirical comparison with a traditional approach. *Proceedings of the 41st ACM Technical Symposium on Computer Science Education,* 500-504.

Hundhausen, Narayanan, & Crosby. (2008). Exploring studio-based instructional models for computing education. *ACM SIGCSE Bulletin, 40*(1), 392-396.

Lackney, J. (1999). A history of the studio-based learning model. *Retrieved March, 8*, 2012.

Lewis, C. M., Yasuhara, K., & Anderson, R. E. (2011). Deciding to major in computer science: A grounded theory of students' self-assessment of ability. *Proceedings of the Seventh International Workshop on Computing Education Research*, 3-10.

Lopez, E. (2003). McKeachie's teaching tips: Strategies, research, and theory for college and university teachers (review). *The Review of Higher Education, 27*(2), 283-284.

McKeachie, W. J., & Svinicki, M. (2006). *McKeachie's Teaching Tips: Strategies, Research, and Theory for College Students and University Teachers*. New York: Houghton Mifflin Company.

Reynolds, C. R., Livingston, R. B., & Willson, V. (2009). *Measurement andAssessment in education* (2nd ed.). New Jersey: Pearson.

Rotenberg, R. (2010). *The art and craft of college teaching: A guide for new professors and graduate students* Left Coast Press.

Sutherland, T. E., & Bonwell, C. C. (1996). Using active learning in college classes: A range of options for faculty. *New Directions for Teaching and Learning*, (67)

Woodley, M., & Kamin, S. N. (2007). Programming studio: A course for improving programming skills in undergraduates. *ACM SIGCSE Bulletin*, *39*(1) 531-535.

Recebido para publicação em 16-09-17; aceito em 15-10-17