Using Scratch to Support Programming Fundamentals

Original Title: Uso de Scratch como apoyo a Fundamentos de Programación

Roberto Muñoz[1], Thiago S. Barcelos[2], Rodolfo Villarroel[3], Ismar Frango Silveira[4]

[1] Escuela de Ingeniería Civil Informática, Universidad de Valparaíso – Valparaíso – Chile
[2] Instituto Federal de Educação, Ciência e Tecnologia de São Paulo (IFSP) – São Paulo, SP – Brasil
[3] Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso – Valparaíso – Chile
[4] Universidade Presbiteriana Mackenzie – São Paulo, SP – Brasil

ARTICLE INFO

Article history:
Received 07 August 2017
Accepted 18 August 2017
Available online 18 August 2017

Keywords:
Programming Fundamentals
Scratch
Programming Learning

ISSN: 2594-5602

DOI:
10.14210/ijcthink.v1.n1.p68

ABSTRACT

INTRODUCTION: *Courses related to the Programming Fundamentals usually have high failure rates. Several initiatives have emerged to address this issue. Among these activities, digital game programming stands out as a promising alternative considering that it is related to a context which is familiar and motivating to students.* ***OBJECTIVE:*** *To design and implement a game design workshop aimed at supporting a Programming Fundamentals discipline.* ***METHOD:*** *The workshop was composed of 12 sessions, and the Scratch programming environment was used. Activities were inspired by Problem-Based Learning and constructionist principles. The code of 24 final projects was analyzed by using a previously defined metric, and retention rates were compared to the level of attendance to the workshop sessions.* ***RESULTS:*** *The results obtained show that students who had high attendance to the workshop also presented high approval rates in the discipline. Additionally, the codes analyzed indicated the development of skills related to Computational Thinking at a high level.* ***CONCLUSIONS:*** *The use of this didactic strategy complemented with the use of Scratch facilitated the development of competencies necessary for programming and also motivated first-year students.*

RESUMEN

INTRODUCCIÓN: *Los cursos relacionados con los Fundamentos de Programación suelen tener altas tasas de fracaso. Varias iniciativas han surgido para abordar esta problemática. Entre estas iniciativas, la programación de juegos digitales se destaca como una alternativa prometedora considerando que está relacionada con un contexto familiar y motivador para los estudiantes.* ***OBJETIVO:*** *Diseñar e implementar un taller de diseño de juegos orientado a apoyar la disciplina de Fundamentos de Programación.* ***MÉTODO:*** *El taller estuvo compuesto de 12 sesiones, y se utilizó el entorno de programación Scratch. Las actividades se inspiraron en el Aprendizaje Basado en Problemas y en los principios construccionistas. Se analizaron los códigos 24 proyectos finales utilizando una métrica previamente definida, y las tasas de retención se compararon con el nivel de asistencia a las sesiones del taller.* ***RESULTADOS:*** *Los resultados obtenidos muestran que los alumnos con alta asistencia al taller también presentaron altas tasas de aprobación en la disciplina. Adicionalmente, los códigos analizados indicaron el desarrollo en un nivel alto de las habilidades relacionadas con el Pensamiento Computacional.* ***CONCLUSIONES:*** *El uso de esta estrategia didáctica complementada con el uso de Scratch facilitó el desarrollo de competencias necesarias para la programación y también motivó a los estudiantes de primer año.*

1. Introducción

Actualmente, en Chile el problema de la retención en carreras asociadas a la Ingeniería Informática es un tema preocupante. En el caso de la carrera de Ingeniería en Informática en la Universidad de Valparaíso, la tasa de retención no logra superar el 50 por ciento en los 3 primeros años. Es por esta razón que a nivel institucional se han realizado acciones para así disminuir la deserción, la cual va principalmente enfocadas a la reorganización de contenidos e incorporación de nuevas herramientas para facilitar su aprendizaje.

La ACM y la IEEE definen que el dominio de los Fundamentos de Programación son una de las competencias principales a ser desarrolladas por los estudiantes de las carreras relacionadas con Ingeniería en Computación (ACM-IEEE Software Engineering 2008, 2008). En el caso de los estudiantes de Ingeniería Civil Informática en la Universidad de Valparaíso reciben su formación en programación en los primeros 2 años a través de tres cursos, Fundamentos de Programación,

Programación 1 y Programación 2. Sin embargo, debido a la baja retención que tiene la carrera, la cantidad de estudiantes que llega a Programación 2, lugar en que se trabaja bajo paradigma orientado a objetos, es mucho menor.

Por otra parte, la construcción de artefactos computacionales por los estudiantes es una estrategia didáctica para el desarrollo de habilidades informáticas que viene siendo frecuentemente utilizada. En ese contexto, los juegos digitales vienen siendo evidenciado como una alternativa altamente promisoria. Esto debido a que los juegos digitales están insertos en el contexto cultural de los nativos digitales. De tal forma Peppler e Kafai (2009) proponen que los estudiantes de nuevas generaciones deberían desarrollar una fluidez en los juegos, es decir una mayor comprensión de sus características como medio digital, por ejemplo, su mecánica y los procedimientos necesarios para su construcción. Algunos relatos de la evaluación de actividades de creación de juegos digitales por estudiantes (Denner, Werner, & Ortiz, 2012; Maloney, Peppler, Kafai, Resnick, & Rusk, 2008) evidencian que la necesidad de incorporar aspectos de interacción en los juegos puede haber contribuido para un uso más intenso de algunas estructuras de programación, o el mismo aprendizaje de algunos conceptos por parte de los estudiantes de manera autónoma, es decir, sin supervisión de instructores.

Con base a lo antes expuesto, este trabajo tuvo como objetivo diseñar e implementar un taller de programación de juegos digitales con Scratch como apoyo a la asignatura de fundamentos de programación. El taller desarrollado se basa en principios del constructivismo y en aprendizaje basado en problemas (ABP). Se evaluaron los artefactos digitales realizados por los estudiantes mediante la rúbrica propuesta por (Moreno-León & Robles, 2015) y analizaron las calificaciones obtenidas por los participantes. Los resultados muestran que aquellos que asisten con una frecuencia alta al taller en su totalidad aprueban la asignatura de Fundamentos de Programación.

2. Trabajos Relacionados

Se ha logrado visualizar en el último tiempo diferentes iniciativas asociadas al uso de programación con Scratch en primeros años de carreras relacionadas con la informática. De hecho, Scratch se concentra entre los 20 lenguajes de programación más populares según el índice TIOBE del 2017 (The Software Quality Compa, 2017), y entre los 7 lenguajes más utilizados por las Universidades Top de Estados Unidos, siendo éste el único lenguaje basado en bloques de esta lista (Guo, 2014).

Por ejemplo en (Rizvi, Humphries, Major, Jones, & Lauzun, 2011), se utiliza Scratch en un curso de Ciencias de la Computación introductorio con el fin de mejorar la retención de estudiantes de alto riesgo, resultando en un aumento sustancial en la matrícula con respecto a años anteriores.

El objetivo del trabajo presentado en (Uludag, Karakus, & Turner, 2011) es resumir la experiencia en la enseñanza en un curso que puede ser considerado como introductorio a las Ciencias de la Computación transversal (denominados CS0/IT0. Con los fundamentos pedagógicos en materia de aprendizaje construccionista y educación informática contextualizada, los autores presentan la motivación y los detalles de un curso que utiliza el lenguaje de programación Scratch, App Inventor, y la robótica con Lego Mindstorms.

En (Rafalski & Santos, 2016), presentan una experiencia del uso de Scratch para la enseñanza de la programación con estudiantes de Ingeniería Eléctrica. Los autores también incorporan elementos tales como GDD (*Document Game Designer*) como base para el desarrollo de los juegos, y además de requerir que los grupos llevase una bitácora del proceso de desarrollo del juego. Para finalizar, los autores describen que los

resultados fueron satisfactorios, además de evidenciar que existió un gran interés e interactividad por parte de los estudiantes durante el proceso de creación de juegos.

En (Gonçalves, Da Silva, Da Luz, & Silva, 2013) relatan la experiencia en el uso de Scratch como herramienta de enseñanza de la programación, además de la construcción de un objeto de aprendizaje *unplugged*. Las actividades fueron realizadas para una carrera de Licenciatura en Computación, en particular en el curso de Fundamentos y Metodologías de Enseñanza de la Informática en la Educación 1. Para finalizar, los autores evidencian algunas dificultades en el uso de Scratch 1.0 tales como interacción entre objetos diferentes, o que haya que replicar el código para todos ellos (ausencia de clones).

3. Estructura del Taller

La estructura constó con 12 sesiones, en la que cada encuentro duró una hora y media. En cada encuentro el docente propuso la construcción de uno o más mecanismos de interacción relacionados con la construcción de un juego digital. Para esto, los estudiantes fueron invitados a explorar conceptos relacionados con desarrollo de juegos (animación de sprites, colisión, controles por teclado y mouse) y de Fundamentos de Programación (variables, mensajes, estructuras condicionales, estructuras repetitivas). A partir de los tópicos a ser cubiertos en la asignatura de FP, se definieron cuatro directrices:

1. La construcción de juegos debe motivar el desarrollo de todas las actividades del taller. Esta directriz se basa en el principio de fluidez de juegos puesto por Peppler y Kafai (2009), que argumentan que el proceso de diseño y construcción de artefactos en el contexto cultural de los juegos digitales puede promover un proceso de reflexión y aprendizaje para los estudiantes.
2. Las actividades deben progresivamente llevar a la construcción de la mecánica de un juego completo. En trabajos anteriores que discuten el desarrollo y conducción de actividades prácticas con el objetivo de fomentar el desarrollo del Pensamiento Computacional. Lee et al. (2011) proponen el esquema Usar – Modificar – Crear, en el cual este principio se basa. Inicialmente a los estudiantes se les presentan programas listos para interactuar y comprender su funcionamiento, a partir de la comprensión obtenida en esa primera etapa, los estudiantes son invitados a introducir modificaciones en su funcionamiento y su apariencia. Finalmente, a medida que los estudiantes adquieren mayor confianza, comienzan a crear sus propios juegos, aplicando los conocimientos adquiridos en etapas anteriores. Cabe destacar que esta estrategia no es aplicada de forma lineal. Un estudiante puede actuar como creador de una etapa del proceso de aprendizaje y en una etapa siguiente, volver a actuar como "usuario" para comprender un nuevo concepto. Sin embargo, la estructura de las actividades y los conocimientos y habilidades a ser movilizados inducen al estudiante a actuar cada vez como "creador" a lo largo del taller. La estrategia todavía tiene impacto en la mantención de un nivel adecuado de desafío.
3. Las actividades deben progresivamente demandar que nuevos conceptos sean explorados por los estudiantes, al mismo tiempo, solicitar que el estudiante utilice nuevamente conceptos explorados anteriormente. La introducción de nuevos conceptos de forma paulatina en las actividades (a través del esquema Usar - Modificar - Crear) tiene como objetivo introducir continuamente nuevos desafíos que induzcan al estudiante a buscar nuevos conocimientos. El proponer nuevos desafíos al mismo tiempo en que conceptos ya explorados son requeridos nuevamente, tiene como objetivo que los estudiantes estén en un estado de flujo

en lo que se refiere a su trabajo autónomo en la construcción de juegos (Nakamura & Csikszentmihalyi, 2009). De la misma forma, se espera que el profesor pueda actuar como facilitador en los momentos en que los estudiantes presenten dificultades puntuales relacionadas, por ejemplo, a un concepto específico. En estos casos, se espera que la secuencia planeada de las actividades mantenga a los estudiantes en la zona de desarrollo proximal (Vygotsky, 1978), permitiendo así su avance en las actividades.

4. La mecánica de los juegos, a pesar de ser simples, debe traer referencia al universo de los juegos "reales" para que sean significativas para los estudiantes. Alineados con una propuesta constructivista, en que la construcción de artefactos digitales por los estudiantes demanda postular de una forma razonablemente autónoma, las actividades del taller siguen el abordaje de aprendizaje basado en problemas (ABP) (2002). Según Merril (2002), el aprendizaje basado en problemas es una estrategia centrada en el estudiante, que trabaja de manera colaborativa en la solución de algún problema, en la cual la figura del profesor sirve como apoyo y la construcción del conocimiento es gradual y empírica. De esta forma, en cada actividad del taller, los estudiantes reciben instrucciones sobre los objetivos propuestos para el juego; además de eso, es presentado un ejemplo del juego propuesto siendo ejecutado y a partir de ahí inician el trabajo. El profesor actúa como un facilitador observando el trabajo e interviniendo a medida que los estudiantes solicitan mayores apoyos.

En las primeras semanas del taller, las actividades son relacionadas con la utilización del entorno Scratch y para reforzar los conceptos vistos en la cátedra de FP.

Los juegos propuestos son: Piedra Papel y Tijera, Simulación de Guerra y prototipos de los famosos juegos Breakout y Pacman. Para finalizar los estudiantes debían presentar su un juego a libre elección.

En la Tabla 1, se presenta la programación de actividades del taller de programación de juegos con Scratch.

Sesión	Actividades / contenido
1	Familiarización con el ambiente Scratch (conceptos sprite y colisión entre sprites)
2	Variables y estructuras repetitivas
3	Estructuras repetitivas y estructuras condicionales
4	Crear juego Piedra-Papel-Tijera
5-6	Crear el juego Simulación de Guerra
7-8	Crear el juego Breakout
9	Pacman – Crear la mecánica básica de los movimientos de los personajes
10-11	Pacman – Implementar las demás características del juego final
12	Presentación del proyecto Final (Temática libre con restricciones)

Tabla 1. Actividades taller de programación

4. Perfil de los Participantes

Se realizó un cuestionario disponible en ("Encuesta de Percepción de Juegos Digitales," 2015) con el fin de determinar el perfil de los participantes y su relación con juegos digitales.

Los participantes casi en su totalidad (78%) correspondieron a estudiantes entre 18 y 19 años. Sólo el 22% de ellos tenían más de 19 años (min: 20 años, máx: 25 años, desviación standard de 1.64). En lo que respecta al sexo, sólo el 12,2% correspondió a femenino y el 87.8% restante a masculino. El 26,8% de los estudiantes provenía de establecimientos municipales (público), el 68,3% de establecimientos particulares subvencionados (financiamiento mixto), y tan sólo el 4,9% de los participantes provenía de colegios particulares (pagados). En lo que respecta a su procedencia, el 63,4% provenía de la región en que la Universidad de Valparaíso se encuentra establecida y el 36,6% de otra región del país.

En lo que respecta a su experiencia en juegos digitales, el 87.8% de ellos declara tener experiencia en juegos digitales, mientras que el 12,20% declara que no. Un elemento a considerar es que de ese 12.2% que declara no tener experiencia en juegos digitales el 7,3% corresponden a mujeres. Este es un elemento que se deberá considerar en una próxima iteración del taller de programación de juegos digitales.

Los estudiantes tenían experiencia en juegos digitales, declaran el 16,67% jugar menos de una hora diaria, el 36,1% entre una y tres horas diarias, el 22,2% entre 3 y 5 horas diarias, el 19,4% entre 5 y 8 horas diarias, el 2,8% entre 8 a 12 horas diarias y 2,8% jugar más de 15 horas diarias.

Cabe mencionar que el 91,7% afirma jugar hace 3 años o más, concentrando un 58,3% a aquellos que juegan hace más de 9 años.

En lo que respecta al tipo de juego preferido las preferencias de los estudiantes se distribuyeron principalmente de la siguiente forma. El 33,3% de preferencias fue a *Role-Playing Games* (RPG), el 16,7% a Acción y Aventura, el 13,9% a *Multiplayer Online Battle Arena* (MOBA) y Shooter en primera persona (FPS) con Deportes, ambos con un 11,1%. Cabe mencionar que los estudiantes tenían 10 opciones de categorías de juegos en las cuales podían marcar sus preferencias.

5. Resultados

El taller se ofreció a los estudiantes durante el primer semestre del año 2015. A todos los estudiantes de primer año matriculados en Fundamentos de Programación se les permitió participar en el taller, sin embargo, su participación no fue obligatoria. El taller fue considerado como una actividad extra en el programa del curso. Sin embargo, se les solicitó la elaboración de un video juego al término de taller.

El videojuego debía contar con una introducción que era una animación creada por los estudiantes, un menú inicial el cual debía contar con 2 botones, uno para iniciar la partida y uno para modificar la dificultad del juego (en base a la dificultad seleccionada se debía provocar cambios en el juego ej: disminuir o aumentar la cantidad de enemigos en base al nivel), y 5 niveles jugables. Además, el usuario debía contar con 3 vidas para avanzar durante el juego, tras llegar a 0 vidas se debía producir el fin del juego y debía comenzar todos los niveles nuevamente. Para finalizar, el fondo del juego debía tener un *scroll* a medida que avanzara con el personaje controlado por el jugador (emulando una resolución de pantalla de 960px. o más). Ejemplos de los trabajos realizados por los estudiantes, son presentados en las figuras 1, 2 y 3.

Figura 1. Juego realizado por estudiante 1.

Figura 2. Juego realizado por estudiante 2.

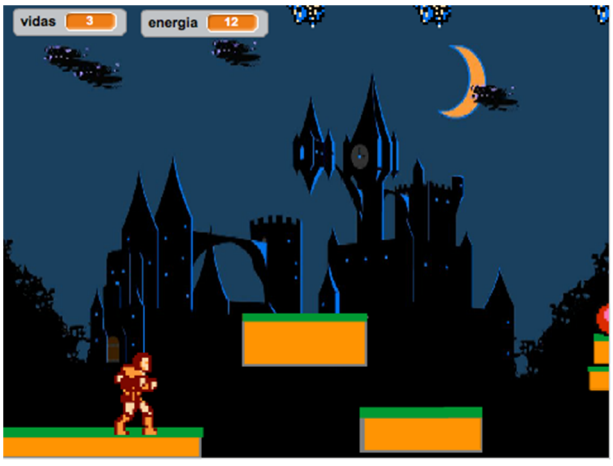

Figura 3. Juego realizado por estudiante 3.

Analizamos los 24 juegos elaborados por los estudiantes en términos de desarrollo de competencias del pensamiento computacional (Wing, 2006). Para ello utilizamos la rúbrica propuesta por (Moreno-León & Robles, 2015), la cual es presentada en la Tabla 1. Los resultados del análisis se pueden visualizar en el gráfico de radar presentado en la Figura 4.

Categoria	Básico – (1 punto)	En desarrollo – (2 puntos)	Competente – (3 puntos)
Abstracción y descomposición de problemas	Más de un programa y más de un objeto	Definición de bloques propios	Uso de clones
Paralelismo	Dos scripts relacionados con la bandera verde	Dos programas en 'tecla presionada', dos programas en 'al presionar' el mismo objeto	Dos programas en 'cuando reciba mensaje', crear clon, dos programas en 'cuando %s es > %', dos programas en 'cuando el escenario cambie a'
Pensamiento lógico	Sí	Si - no	Operaciones lógicas
Sincronización	Esperar	Enviar, cuando reciba mensaje, parar todos, parar programas, parar programas del objeto	Esperar hasta, cuando el escenario cambie a, enviar y esperar
Control de flujo	Secuencia de bloques	Repetir, por siempre	Repetir hasta
Interactividad con el usuario	Bandera verde	Tecla presionada, objeto presionado, preguntar y esperar, bloques de operaciones con ratón	Cuando %s es >%s, vídeo, audio
Representación de datos	Modificadores de propiedades de objetos	Operaciones en variables	Operaciones en listas

Tabla 1. Rúbrica de evaluación de la utilización de conceptos del Pensamiento Computacional.

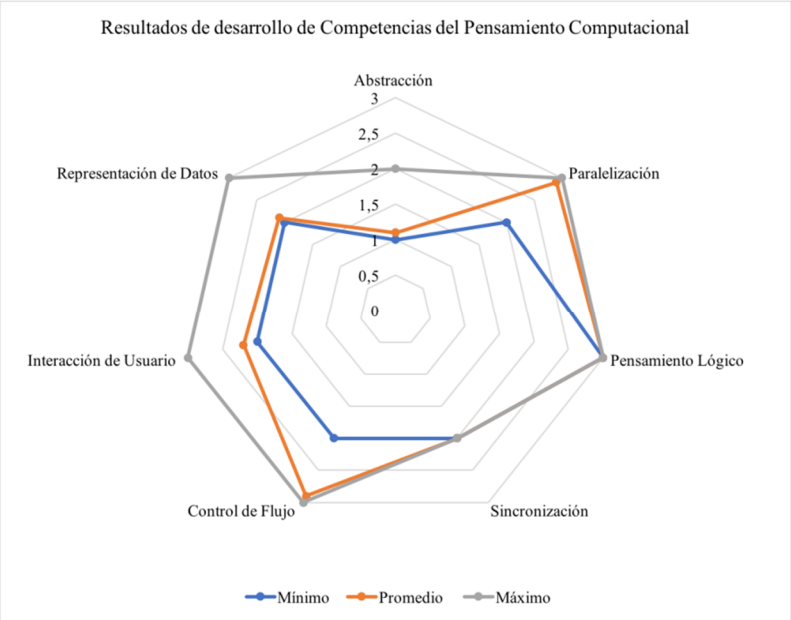

Figura 4. Resultado de desarrollo de competencias del Pensamiento Computacional.

De acuerdo a lo presentado en la Figura 4, es posible visualizar que el trabajo solicitado involucró en promedio la utilización de elementos de complejidad media y alta. Esto también se puede visualizar al momento de explorar los juegos desarrollados por los estudiantes. Por otra parte, al analizar el número de bloques conectados de manera lógica, los resultados muestran que en promedio los estudiantes conectaron 614,7 bloques de código. Las métricas de dispersión son presentadas en la Tabla 2.

Mínimo	Promedio	Máximo	Desviación Estándar
310	614,7	2844	530,1

Tabla 2. Métricas de dispersión asociadas a los juegos desarrollados.

Los resultados presentados en la Tabla 2, si bien muestran que los estudiantes elaboran juegos con diferentes complejidades, al menos en términos de número de bloques utilizados. Al utilizarlos, todos cumplen con las restricciones dadas en un comienzo.

Por otra parte, si bien el desarrollo de competencias del pensamiento computacional no fue el foco del taller, es importante mencionar que estas competencias se encuentran estrechamente relacionadas con los resultados de aprendizaje de la asignatura, por ende, que los estudiantes hayan logrado elaborar juegos de una complejidad alta, es un resultado alentador.

Para finalizar, en la Figura 3, se puede visualizar la relación entre la asistencia al taller en contraste con el estado final en la asignatura.

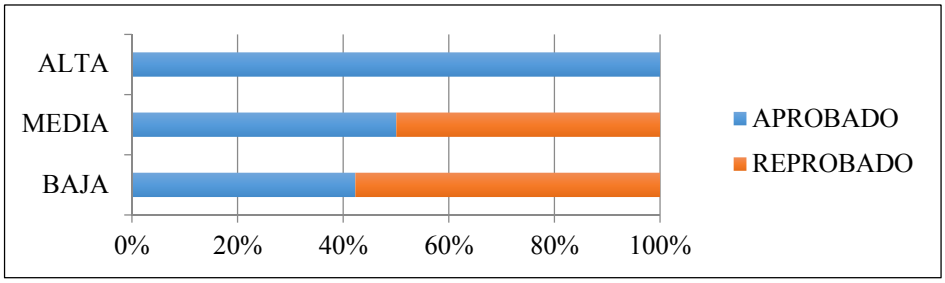

Figura 5. Relación asistencia taller vs estado

De acuerdo a lo presentado en la Figura 5, se puede visualizar que aquellos estudiantes que poseen una asistencia alta (sobre el 66%) al taller en su totalidad aprobaron el curso de Fundamentos de Programación. Sin embargo, el resultado no es claro para aquellos que poseen una asistencia baja (menor al 33%) y media (entre 33% y 66%). Es por esto es que se decidió determinar si existe o no una relación entre el promedio obtenido en el taller y su asistencia. En la Figura 6 se puede visualizar un gráfico de cajas con el promedio, la deviación estándar en cada uno de los grupos de estudiantes mencionados anteriormente.

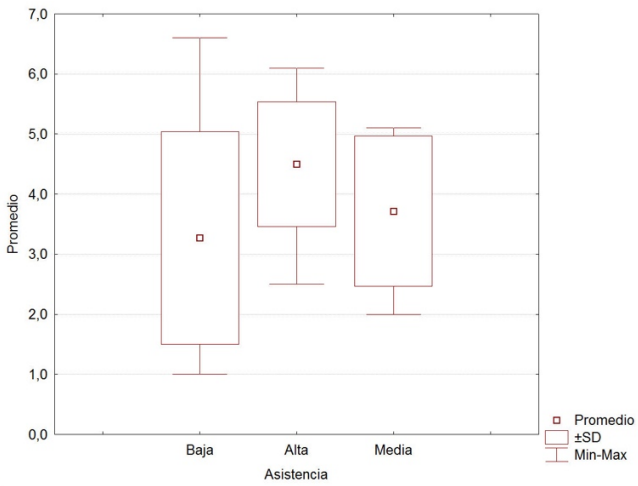

Figura 6. Distribución de grados por nivel de asistencia a el taller.

Es posible visualizar una relación entre la nota promedio de los estudiantes que asisten al curso de Fundamentos de Programación y el nivel de asistencia al Taller, así como una variación menor de las notas a medida que aumenta el nivel de asistencia al taller. A pesar de que el desempeño de los estudiantes podría estar relacionado con un mayor interés y compromiso global por el curso, es que se requiere continuar con la investigación. Sin embargo, la relación identificada anteriormente puede indicar una influencia positiva de las actividades de construcción de juegos en el desempeño académico de los estudiantes.

Por otra parte, si los resultados se comparan con el porcentaje de aprobación con el de otros años, los resultados si son alentadores. Esto debido a que en la segunda aplicación del taller, el aumento en las tasas de aprobación continúa en aumento. Llegando en esta ocasión a un 51% en contraste con el 34% y 40% obtenidos en el en los años 2012 y 2014 respectivamente.

6. Conclusiones

El desarrollo de competencias asociadas a la programación puede ser fomentada con diversas estrategias y herramientas. Bajo esta línea el uso de material didáctico tal como Scratch tiene un alto potencial para desarrollar una aplicación más tangible y concreta de las habilidades requeridas para la programación.

Por otra parte, la retención y aprobación de los estudiantes en los primeros años de las carreras Universitarias siempre será un tema de preocupación, tanto a niveles institucionales como estatales. Por tal razón, se deben buscar estrategias que favorezcan ambos aspectos. Partiendo de esta premisa, desarrollamos un taller de programación de juegos digitales con Scratch para apoyar y complementar la asignatura inicial de Fundamentos de Programación. Si bien la estructura del taller debe refinarse, los resultados son bastante alentadores. Por tal razón, seguiremos buscando alternativas para desarrollar de manera temprana las competencias definidas en los planes de estudio, tomando como directriz que éstas también estén alineadas con los intereses de los estudiantes.

Como trabajo futuro se espera incorporar la integración con placas micro controladoras. Esto con el fin de que los artefactos generados tengan una aplicación concreta al momento de resolver un problema. También se espera poder replicar este taller en otras carreras de la facultad que incorporen Fundamentos de Programación en sus mallas curriculares.

Agradecimentos

Roberto Muñoz es beneficiario de la Beca de Doctorado INF-PUCV 2015.

Referencias

ACM-IEEE Software Engineering 2008. (2008). Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. IEEEComputer Society and Association for Computing Machinery.

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? Computers & Education, 58(1), 240–249. https://doi.org/10.1016/j.compedu.2011.08.006

Encuesta de Percepción de Juegos Digitales. (2015). Retrieved May 24, 2015, from https://es.surveymonkey.com/r/VXSWY7S

Gonçalves, D. A. S., Da Silva, G. M., Da Luz, R. S., & Silva, E. P. (2013). Relato de experiência de alunos do curso de Licenciatura em Computação do IFMG - campus Ouro Branco na utilização de objetos de aprendizagem desplugados e do Scratch como instrumentos no ensino de programação. https://doi.org/10.5753/CBIE.WCBIE.2013.335

Guo, P. (2014, July 7). Python is Now the Most Popular Introductory Teaching Language at Top U.S. Universities | blog@CACM | Communications of the ACM. Retrieved July 31, 2017, from https://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-u-s-universities/fulltext

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., … Werner, L. (2011). Computational thinking for youth in practice. ACM Inroads, 2(1), 32–37. https://doi.org/10.1145/1929887.1929902

Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008). Programming by choice: urban youth learning programming with scratch. In Proceedings of the 39th SIGCSE technical symposium on Computer science education (pp. 367–371). New York, NY, USA: ACM. https://doi.org/10.1145/1352135.1352260

Merril, D. (2002). A Pebble-in-the-Pond Model For Instructional Design. Performance Improvement, 41, 41–46. https://doi.org/10.1002/pfi.4140410709

Moreno-León, J., & Robles, G. (2015). Dr. Scratch: a Web Tool to Automatically Evaluate Scratch Projects. In Proceedings of the 10th Workshop in Primary and Secondary Computing Education (pp. 132–133). London, United Kingdom: ACM Press. https://doi.org/10.1145/2818314.2818338

Nakamura, J., & Csikszentmihalyi, M. (2009). Flow theory and research. In C. R. Snyder & S. J. Lopez, Oxford Handbook of Positive Psychology (2nd ed., pp. 195–206). Oxford: Oxford University Press.

Peppler, K., & Kafai, Y. (2009). Gaming Fluencies: Pathways into Participatory Culture in a Community Design Studio. International Journal of Learning and Media, 1(4), 45–58. https://doi.org/10.1162/ijlm_a_00032

Rafalski, J. D. P., & Santos, O. L. dos. (2016). Uma experiência com a Linguagem Scratch no Ensino de Programação com Alunos do Curso de Engenharia Elétrica (p. 612). https://doi.org/10.5753/cbie.wie.2016.612

Rizvi, M., Humphries, T., Major, D., Jones, M., & Lauzun, H. (2011). A CS0 course using Scratch. J. Comput. Sci. Coll., 26(3), 19–27.

The Software Quality Compa. (2017, July). TIOBE Index. Retrieved July 31, 2017, from https://www.tiobe.com/tiobe-index/

Uludag, S., Karakus, M., & Turner, S. W. (2011). Implementing IT0/CS0 with scratch, app inventor forandroid, and lego mindstorms (pp. 183–190). ACM Press. https://doi.org/10.1145/2047594.2047645

Vygotsky, L. S. (1978). Zone of Proximal Development. In Mind in society: The development of higher psychological processes (pp. 52–91). Oxford: Harvard University Press.

Wing, J. M. (2006). Computational Thinking. Commun. ACM, 49(3), 33–35. https://doi.org/10.1145/1118178.1118215