ENSEÑANDO PROGRAMACIÓN ORIENTADA A OBJETOS USANDO UN AMBIENTE 3D.

Olga Lucía Monroy Vecino

omonroy@unab.edu.co

Tel: (57)7-6436111 Ext. 338

Daniel Arenas Seleey

darenass@unab.edu.co

Tel: (57)7-6436111 Ext. 195

Resumen

El problema fundamental que intenta abordar esta investigación es la dificultad que presentan los estudiantes en el aprendizaje de la programación de computadores en los primeros niveles de educación superior. Como consecuencia, se presenta en las universidades un alto nivel de deserción o pérdida en estos niveles y frustración cuando los estudiantes no pueden alcanzar las competencias requeridas en estos cursos.

Como una alternativa de solución al problema enunciado, se plantea el uso de la herramienta Alice para la enseñanza de la Programación Orientada a Objetos (POO), buscando combatir la poca motivación de los estudiantes y en muchos casos, las dificultades para entender y aplicar los conceptos básicos de la lógica de programación, particularmente de la orientación a objetos. De manera paralela, se usará el lenguaje Java, para no dejar de lado el rigor de la escritura de código usando un lenguaje de programación. Esta estrategia fue evaluada con los estudiantes del primer curso de Programación de la Universidad Autónoma de Bucaramanga. Los resultados mostraron que utilizar Alice mejora la comprensión de los conceptos básicos de la programación orientada a objetos y el uso de las estructuras de control.

Palabras Clave: Alice, Introducción a la Programación, Java, POO.

Introducción

En las Universidades de todo el mundo se ha observado una fuerte disminución en el interés de cursar carreras del área de ciencias computacionales o sistemas de

información y en particular en Colombia, de estudiar Ingeniería de Sistemas (o Ingeniería Informática), así como un alto nivel de deserción en los primeros niveles. En Estados Unidos, el número de graduados del área había declinado en más de un 60% entre el año 2000 y el 2004, según estudios realizados por Carnegie Mellon University, por lo que un equipo de investigadores de dicha universidad, decidió crear una herramienta para tratar de resolver este problema, cambiando fundamentalmente la forma como se enseña programación. Así nació el proyecto Alice, que es un desarrollo en Java, que permite crear animaciones con la utilización de objetos 3D, como un lenguaje de programación con propósitos educativos, de código abierto, orientado a objetos, incluyendo un entorno de desarrollo integrado de fácil utilización, permitiendo arrastrar y soltar los elementos requeridos en el lugar que se desee. Alice les permite a los estudiantes ver de forma inmediata como se ejecuta su programa de animación y la relación entre las líneas del programa y el comportamiento de los objetos. Con la manipulación de los objetos en el mundo virtual, los estudiantes ganan experiencia en el uso de los conceptos fundamentales que normalmente se enseñan en un curso introductorio de programación. Toda la información del proyecto Alice se encuentra disponible en su página oficial[1].

Con este proyecto se busca diseñar una metodología de enseñanza de la programación de computadores en primer nivel, que utilice prácticas que logren desarrollar en el estudiante las habilidades, interés y competencias asociadas al análisis, abstracción, identificación de variables, escritura de sentencias algorítmicas, entre otras, usando Alice, a la par que se desarrollan habilidades para la escritura de código usando un lenguaje de programación orientado a objetos como Java; por lo que las prácticas integrarán los dos ambientes de trabajo: la herramienta en 3D y la interface de desarrollo para un lenguaje orientado a objetos.

Este artículo inicia con una descripción general del trabajo, mostrando cómo la enseñanza de la programación presenta una problemática común en muchas universidades. Con este panorama, se explica cómo ha evolucionado la programación y cómo ha sido enseñada según las teorías y metodologías propias de cada momento. Posteriormente se describe la metodología aplicada en la investigación y en la sección de discusión, se analizan los resultados, para finalizar con las conclusiones del proyecto.

Panorama en las Universidades. El problema de los altos niveles de deserción estudiantil o de pérdida de cursos en el área de programación de computadores en carreras como Ciencias Computacionales, Ingeniería de Software, Ingeniería Informática, Tecnologías de la Información, Ingeniería de Sistemas o similares, y en general en programas de Ingeniería que tienen la asignatura de introducción a la programación dentro del ciclo básico de su plan de estudios, ha sido abordado por numerosas universidades del mundo, en los últimos años (especialmente del 2000 en adelante), con resultados similares, que se evidencian en los siguientes casos:

- Un estudio realizado en la Universidad de California (UCLA – University of

[1] Alice. An Educational Software that teaches students computer programming in a 3D environment. www.alice.org

California at Los Angeles) por el Higher Education Research Institute, muestra que la popularidad de las ciencias computacionales ha descendido significativamente en los últimos años, disminuyendo en un 60% entre el 2000 y 2004 (Vegso, 2005).

- El Departamento de Ciencias Computaciones de la Universidad de Illinois, en Chicago, muestra niveles de deserción en promedio de 25% en estudiantes de primer año (Talton, 2006).

- En la Universidad de California, Berkeley , se realizó un análisis durante 8 años (2002 al 2009), y se encontraron porcentajes de deserción alrededor del 20%, con algunos picos como en primavera del año 2006, que llegó a estar por encima del 40%, para estudiantes del curso introductorio de ciencias computacionales (Lewis, 2010).

- En la Universidad de Pensilvania, solo el 50% de las mujeres inscritas en los primeros cursos de programación en el 2003, continuaron sus estudios en esta línea, lo que motivó a la realización de ajustes en el currículo, políticas y prácticas para aumentar la persistencia de sus estudiantes. Se encontró como un factor muy importante un bajo nivel en las competencias de entrada para enfrentar a compañeros con más experiencia (Manco, 2008).

- En el Departamento de Programación - Facultad de Informática, de la Universidad Nacional del Comahue, Argentina, en los últimos años se vienen desarrollando acciones tendientes a mejorar las condiciones de permanencia de los estudiantes, dados los altos niveles de deserción en los cursos de enseñanza de la programación. Para esto, se propuso el proyecto “Tópicos Avanzados en Programación de Computadoras”. (López et al.,2013).

En el ámbito nacional, el Ministerio de Educación ha realizado varios estudios sobre el ingreso y la permanencia de los estudiantes de todas las áreas de conocimiento, mostrando para el caso de Ingeniería de Sistemas, Telemática y Afines, niveles de deserción del 27.36% en primer semestre, del 55.24% a quinto Semestre y de 64.45 a décimo semestre, siendo el porcentaje más alto el del primer nivel. (Ministerio de Educación Nacional, 2010). La situación en la Facultad de Ingeniería de Sistemas de la Universidad Autónoma de Bucaramanga (Unab) no es diferente.

Por otra parte, se han probado diversas iniciativas con el mismo propósito de mejorar la retención estudiantil en los cursos introductorios de programación, como la práctica de “Pair Programming” o programación colaborativa, en la cual dos programadores trabajan colaborativamente en un computador, en el diseño de un mismo algoritmo, la codificación y la prueba (Li, et al., 2012). Esta herramienta pedagógica se ha investigado ampliamente (Preston, 2006) y (Salleh et al.,2011), con los siguientes hallazgos:

- Pair Programming ayuda a mejorar la tasa de retención y de aprobación de los

cursos introductorios de programación.
- Pair Programming mejora la calidad de los programas, la confianza en sus propios desarrollos y el trabajo se hace más divertido.
- Adicionalmente, Pair Programming ayuda a los estudiantes que tienen un nivel de conocimientos bajo al ingreso a la Universidad.

Ya en la Unab, en algunos grupos de Programación de Computadores, con estudiantes de segundo semestre, se trabajó con un grupo piloto con esta herramienta pedagógica y se lograron buenos resultados en algunos casos. Ahora se propone utilizar una herramienta gráfica de diseño de algoritmos, incorporando también trabajo colaborativo en algunas prácticas del curso.

Desarrollo de la Investigación

Propuesta: A partir del segundo semestre de 2009, se inició el trabajo con Alice en las sesiones prácticas del curso de Fundamentos de Programación (con estudiantes de primer semestre), teniendo una percepción muy positiva por parte de los estudiantes y un incremento en el nivel académico de los grupos, aunque aun se observa dificultad al iniciar la escritura de código en el lenguaje de programación Java. La propuesta es ampliar la aplicación de Alice, y mejorar el uso del lenguaje de programación, mediante el diseño de prácticas que cubran las temáticas del curso, aprovechando las ventajas del entorno gráfico, y la robustez del lenguaje de programación. El uso de Alice permite explicar de una forma clara los conceptos básicos de la programación orientada objetos, haciendo uso de los elementos que trae incorporados la herramienta. Así, por ejemplo, se puede explicar qué es un objeto visualizándolo y manipulándolo mediante los métodos y funciones que vienen programados o con los nuevos métodos que el usuario puede crear. Seleccionando un objeto, se pueden mostrar los atributos del mismo, o modificar sus valores, de manera más amigable y tangible que como lo hacen los lenguajes de programación tradicionales.

En la figura 1, se muestra el objeto tortuga en Alice, y algunos de los métodos que este puede ejecutar, así como el botón <create new method>, que permite ampliar el conjunto de acciones que el objeto puede hacer. Se observa también la pestaña de propiedades que muestra las características del objeto y la pestaña de funciones que despliega una serie de operaciones útiles para la construcción de los métodos.

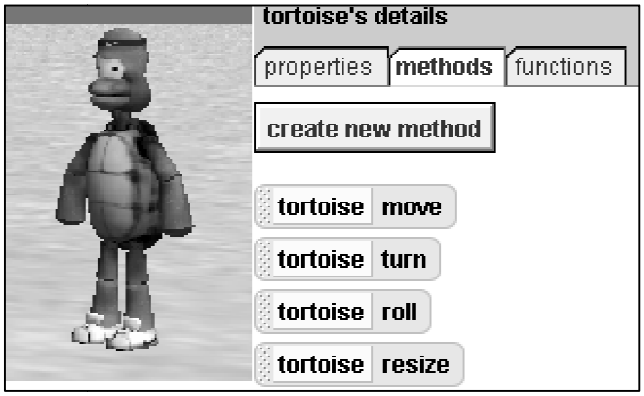

Figura 1. Objeto Tortuga y detalle de métodos.

Cada método contiene una serie de pasos <instrucciones> que se ejecutan secuencialmente para completar una acción. En la figura 2 se observa el método **Saludar** creado para el objeto tortuga, compuesto por tres pasos o instrucciones, que se ejecutan en secuencia.

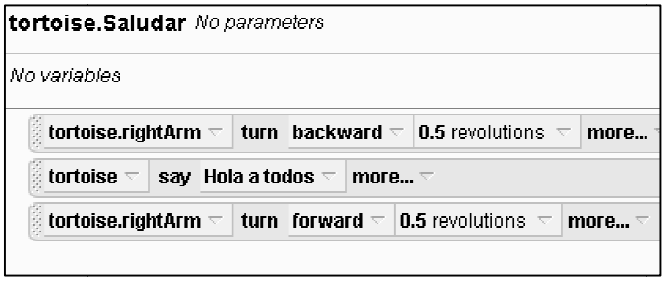

Figura 2. Método saludar del objeto tortuga.

En este método la tortuga sube su brazo derecho, dice “Hola a todos” y vuelve a bajar el brazo a la posición inicial. De una manera muy sencilla el estudiante puede ver el efecto que tiene cada paso y la razón del orden en que se colocan, verificando el concepto de algoritmo, como un conjunto de pasos ordenados para realizar una acción. Las explicaciones que involucran actividades visuales captan más fácilmente la atención de los jóvenes.

En la Figura 3, se muestra la forma como Alice describe una estructura de repetición con un número finito de iteraciones, que para el ejemplo, logra que el objeto tortuga, mostrado en la Figura 1, realice un salto con el pie derecho o con el pie izquierdo dependiendo del valor del índice del ciclo. La función **IEEERemainder of** calcula el residuo de la división entre dos valores enteros, en este caso el index y la constante 2. De esta forma se determina si el index es par o impar, dependiendo del residuo de la división.

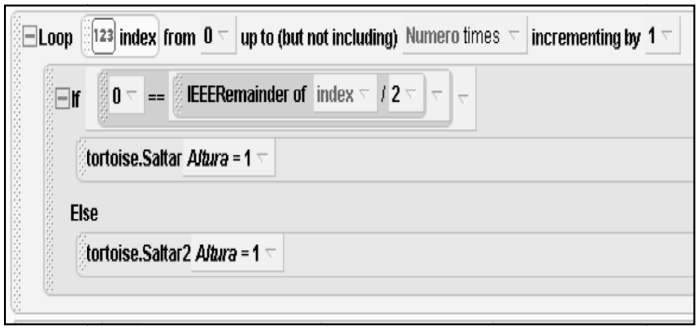

Figura 3. Ciclo de repetición en el entorno de Alice.

Si el índex tiene un valor par, el objeto tortuga ejecutará el método **Saltar,** y si tiene un valor impar, ejecutará el método **Saltar2**, con una elevación de un metro. Estas acciones usando Java, tendrían la siguiente forma:

```
for (index = 0 ; index < numero; index ++)
{
  if (index % 2 == 0)
  tortuga.Saltar(1);
  else
  tortuga.Saltar2(1);
}
```

Que implicaría primero la construcción del objeto tortuga con todos sus atributos y métodos, mientras que en Alice, el objeto ya existe, dentro de una amplia galería de objetos disponibles. Indudablemente, para los estudiantes es más agradable ver el objeto con todos sus movimientos y poder comprobar el efecto de cada una de las acciones que se adicionen, que solo observar líneas de código y una ejecución en consola, que además supone que el estudiante conoce la sintaxis propia del lenguaje.

¿Por qué Alice? Aunque existen otras herramientas similares a Alice como Scratch[2], Greenfoot[3], entre otras, se han encontrado más casos documentados del uso de Alice, con buenos resultados, en cursos introductorios a nivel universitario, por lo que se decidió usar Alice y verificar los resultados de la aplicación.

Alice es un entorno de programación innovador, disponible libremente para ser usada como una herramienta educativa para estudiantes en su primer contacto con la programación orientada a objetos. En Alice, los objetos 3D pueblan un mundo virtual y los estudiantes crean un programa para animar los objetos. Esta herramienta fue desarrollada por los investigadores de la Universidad Carnegie Mellon, entre los que se destacan Randy Pausch (fallecido en Julio del 2008), Wanda Dann (actual líder del grupo de trabajo de Alice), Steve Cooper, entre otros, y continúa liberando nuevas y

[2]Scratch, herramienta desarrollada por MIT. Disponible en http://scratch.mit.edu/
[3] Greenfoot, herramienta desarollada por la Universidad de Kent (Inglaterra). Disponible en http://www.greenfoot.org/

más completas versiones, con la colaboración de otros investigadores y estudiantes, y el patrocinio de grandes empresas como EA – Electrionic Arts, Oracle, Sun Microsystems, NSF - The Nacional Science Foundation, Google, The Hearst Foundation, Darpa, Disney, entre otras. La versión estable más difundida es la 2.2, y en Marzo del 2013 fue liberada la versión Alice 3.1, en el SIGCSE'13 en Denver[4].

El principal objetivo de Alice es lograr que la primera experiencia en la programación sea agradable, mediante un aprendizaje visual en lugar de texto plano. Con Alice, el usuario arrastra objetos 3D y las acciones deseadas, logrando una animación sin errores. Esto se traduce en que el usuario puede ver lo que ocurre con cada línea de código de manera inmediata.

Los objetos 3D en el mundo virtual pueden moverse, girar, cambiar de color o tamaño, reaccionar a eventos creados para el ratón o el teclado y mucho más. Su interfaz interactiva genera instrucciones al arrastrar y soltar elementos gráficos (drag and drop). Estas instrucciones se asemejan a líneas de código de lenguajes de programación como **Java o C++**. Al ver en forma inmediata cómo se ejecutan las animación, los estudiantes pueden entender con mayor facilidad la relación entre el código y el comportamiento de un objeto. Su entorno visual mejora la retención y el aprendizaje, evitando la frustración de una sintaxis mal utilizada. (Garrido, 2008) y (Dann, 2006).

Alice 2.2 puede utilizarse en Windows (7, Vista, XP, 2000) Mac (OS X 10.3 en adelante). Los requerimientos de máquina son mínimos: Procesador Pentium II (o equivalente), tarjeta gráfica de 16 bit y resolución 1024x768, tarjeta de sonido y 512 RAM como mínimo (recomendado 1 M). No requiere instalación.

En numerosas universidades e institutos de educación superior, se han realizado estudios para comprobar la efectividad del uso de Alice en las aulas, en rendimiento y disminución de los índices de deserción, especialmente en estudiantes considerados "en riesgo", es decir, quienes ingresan sin ninguna preparación previa en el área de la programación o con niveles bajos en matemáticas, como el estudio realizado en Saint Joseph's University (SJU) e Ithaca College (IC) (Moskal et al., 2003).

Tres de los principales creadores de Alice, Pausch, Dann y Cooper, realizaron una investigación, buscando el mejor método de llegarles primero a estudiantes que inician su primer curso de programación, entre estos métodos estudiaron: Programación imperativa, programación funcional y programación con objetos, concluyendo que usando Alice como primer acercamiento a la programación con objetos se lograba reducir el nivel de complejidad, diseño preliminar de objetos y visualizar los objetos en un contexto significativo (Cooper, et al. 2003).

Resultados

[4] SIGCSE, ACM Special Interest Group of Computer Science Education. Marzo 6-9 de 2013. http://www.sigcse.org/sigcse2013/

En una primera fase, se ha realizando la búsqueda y documentación de experiencias satisfactorias del uso de Alice a nivel universitario, y de Java de forma paralela, para con esta información y los criterios propios, diseñar las primeras prácticas con Alice, para ser desarrolladas por estudiantes de los cursos de Fundamentos de Programación. Utilizando los resultados obtenidos de la aplicación de estas prácticas, se han rediseñado dichas prácticas, reforzando los temas donde se haya presentado mayor dificultad para los estudiantes. A la par de estas actividades, se han creado programas en Java que evidencian los conceptos fundamentales de la programación orientada a objetos, acordes con las prácticas realizadas con Alice.

Con los cursos finalizados, se recopilaron las mejores prácticas de cada uno de los temas que forman parte del contenido del curso de Fundamentos de Programación, desarrolladas por docentes y estudiantes, para proponer una metodología de aplicación, que sirva de guía del curso, para posteriores aplicaciones, y que se espera sea mejorada y ampliada con las experiencias en las aulas y los aportes de estudiantes y demás docentes en siguientes semestres.

La metodología de enseñanza de la programación se implementó con dos grupos piloto, a lo largo de dos semestres académicos, para poder contrastar los resultados obtenidos con el histórico que se tiene de los logros alcanzados por los estudiantes en los cursos de Fundamentos de la Programación en semestres anteriores. Actualmente, ya se cuenta con un conjunto de prácticas de ejemplo hechas por los docentes de la Universidad, y con los estudiantes, ya se han realizado encuestas a los estudiantes de los grupos piloto, para conocer su apreciación sobre la usabilidad y la utilidad de la herramienta, y se ha introducido paulatinamente la creación de código java para resolución de tareas.

De la encuesta hecha a un grupo de 29 estudiantes, se concluye en forma general, que una herramienta como Alice cuenta con un muy buen nivel de aceptación. A la pregunta relacionada con la facilidad de uso de la herramienta, el 66% de los estudiantes respondió que la herramienta es sencilla pero requiere de mucha práctica (respuesta b), mientras el 31% respondió que es muy fácil de usar (respuesta a). En otra pregunta se indagó sobre si recomendaría el uso de Alice en los cursos introductorios de programación, y el 76% del grupo respondió de forma afirmativa. Con los dos grupos se ha podido verificar que la herramienta es aceptada por los estudiantes, les ayuda a entender conceptos abstractos de la programación, y los mantiene motivados en las clases.

Conclusiones

Con la aplicación de Alice en los cursos introductorios de programación, se ha evidenciado el incremento en la motivación en las clases, mejoras en la asimilación de conceptos, aunque no se han visto mejoras significativas en el momento de escribir los programas de la manera tradicional, es decir, usando el lenguaje de programación. Por lo tanto, el reto es lograr no solo que los estudiantes hagan buen uso de una

herramienta para programación en 3D como Alice, y asimilen los conceptos fundamentales, sino que sean capaces de realizar los programas en Java con la misma facilidad que pueden operar un objeto en el entorno gráfico.

La aplicación de las prácticas con los estudiantes usando Alice y los proyectos de clase creados por ellos, mostraron dominio sobre conceptos fundamentales de programación, así como una actitud más positiva y dinámica en las clases, pero el rechazo a escribir código java aun se presenta. Como tarea a seguir, el propósito es continuar en la búsqueda de herramientas y metodologías para ser utilizadas en los primeros cursos de programación, que ayuden a mitigar las deficiencias con las que llegan los estudiantes, la falta de motivación, y el rechazo para el desarrollo de aplicaciones escribiendo código en los programas. Las experiencias de otras universidades deben seguir siendo monitoreadas.

Para continuar en la búsqueda de buenas prácticas, actualmente se está trabajando en la incorporación de talleres para la creación de videojuegos de nivel básico usando java, donde el estudiante pueda usar la programación mientras crea un producto que lo motive.

Referencias

Vegso, J. (2005). *Interest in CS as a major drops among incoming freshmen*. Computing Research News, 17(3). Recuperado de http://www.cra.org/CRN/online.html.

Talton, J., Peterson, D., Kamin, S., Israel, D., & Al-muhtadi , J. (2006). *Scavenger Hunt: Computer Science Retention Through Orientation*. Department of Computer Science. University of Illinois at Urbana-Champaign. Recuperado de http://slice.cs.illinois.edu/pubs/scavenger-hunt.pdf

Lewis, C. (2010). *Attrition in Introductory Computer Science at the University of California, Berkeley*. Electrical Engineering and Computer Sciences. Technical Report No. UCB/EECS-2010-132-2010. Recuperado de http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-132.html

Manco, R. (2008). *Improving the Persistence of first-year undergraduate women in Computer Science*. University of Pennsylvania. Recuperado de http://www.seas.upenn.edu/~rpowell/documents/sigcse204-powell.pdf

Ministerio de Educación Nacional. República de Colombia. (2010) *Jaque a la Deserción.* Boletín informativo No. 14. Febrero de 2010. Recuperado de http://menweb.mineducacion.gov.co/educacion_superior/numero_14/media/ES14_web.pdf

Moskal, B., Lurie, D. & Cooper, S. (2003). *Evaluating ther Effectiveness of a New Instructional Aproach.* Recuperado de

http://www.alice.org/publications/EvaluatingTheEffectivenessOfANewApproach.pdf

Cooper, S., Dann, W. & Pausch, R. (2003). *Teaching Objects-first In Introductory Computer Science*. Recuperado de http://www.alice.org/publications/TeachingObjects-firstInIntroductoryComputerScience.pdf.

Li, Z., Plaue, C. & Kraemer, E. (2012). *A Spirit of Camaraderie: The Impact of Pair Programming on Retention*. Department of Computer Science- The University of Georgia Athens, GA USA. Recuperado de http://cs.uga.edu/~zhen/Paper/cseet13.pdf

Preston, D. (2006). *Using collaborative learning research to enhance pair programming pedagogy*, SIGITE News., vol. 3, pp. 16--21, jan 2006. Recuperado de http://dl.acm.org/citation.cfm?id=1113381

Salleh, N., Mendes, E. & Grundy, J. (2012) *Empirical Studies of Pair Programming for CS/SE Teaching in Higher Education: A Systematic Literature Review*, IEEE Trans. Softw. Eng., vol. 37, pp. 509- -525, jul 2011. Recuperado de http://dl.acm.org/citation.cfm?id=2052488

Garrido, J. (2008). *Alice: the programming language*. Sudbury, US. Ed. Jones and Bartlett Publishers.

Dann, W. (2006). *Learning to program with Alice*. Saddle River, US. Ed. Pearson Prentice Hall.

López, L., Amaro, S., Godoy, I., Alonso de Armiño, A., & Leiva, M. (2013) Tópicos Avanzados en la Programación de Computadoras. Recuperado el 20 de Agosto de 2013 de http://sedici.unlp.edu.ar/handle/10915/27422