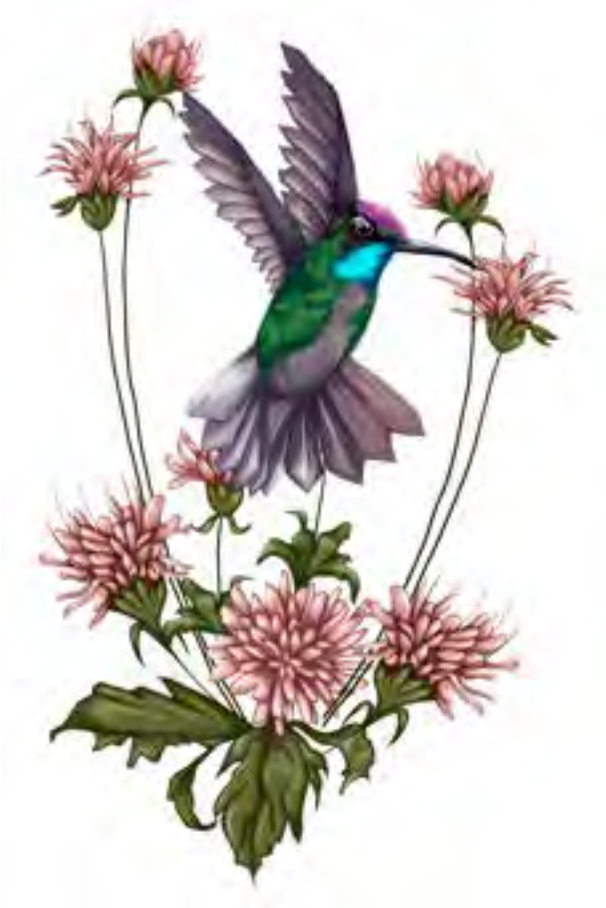


- Capítulo 11 -

Juego para la resolución de problemas en programación de computadores

Autores

- **Edwin Andrés Niño Velásquez.** Es Ingeniero de Sistemas de la Universidad Nacional de Colombia y candidato a Magíster en Ingeniería de Sistemas y Computación de la Universidad Nacional de Colombia. Profesor de planta del Politécnico Grancolombiano desde 2010. *Correspondencia: eaninove@poligran.edu.co*

- **Javier Fernando Niño Velásquez.** Es Ingeniero de sistemas de la Universidad Nacional de Colombia, Especialista en Economía de la Universidad de los Andes y Especialista en Evaluación Social de Proyectos de la Universidad de los Andes. Magíster en Ingeniería Industrial de la Universidad de los Andes. Profesor de planta del Politécnico Grancolombiano desde 2012. *Correspondencia: jninovel@poligran.edu.co*

Resumen

La programación de computadores requiere de un pensamiento analítico, lógico y matemático que se adquiere en la carrera de Ingeniería. Se pretende que los estudiantes apliquen tales conocimientos matemáticos en la resolución de problemas y en la escritura de algoritmos. Para ello, se crea una herramienta de software que consiste en un juego estructurado por niveles de dificultad, o problemas que reten a los estudiantes y los lleven a analizarlos y resolverlos. El juego se diseña para aprendices y cuenta con sesiones de entrenamiento, dadas por un maestro que le orienta empleando elementos de storytelling educativo y problemas de ejemplo para que el estudiante analice. Luego tiene una sesión de entrenamiento que consiste en cinco problemas relacionados, para terminar el nivel enfrentado a un monstruo o virus informático que debe combatir. Cada nivel presenta mayor dificultad y el estudiante aprende por medio de bloques de programación. En el nivel final para "salvar el mundo", debe resolver un problema que debe programar en pseudocódigo. El propósito es que el estudiante desarrolle una lógica de programación más que aprender un lenguaje. El juego es una herramienta motivadora para el estudiante, que va más allá de lo lúdico convirtiéndose en un verdadero reto educativo.

Palabras clave

Videojuegos, gamificación, aprendizaje basado en juegos, programación por bloques, programación de computadores.

Introducción

El módulo de programación de computadores es un módulo teórico-práctico de tres créditos que pertenece a la Facultad de Ingeniería, Diseño e Innovación. Se oferta a todos los estudiantes de los programas de Ingeniería de Software

y Tecnología en Logística en la modalidad virtual. Es el módulo introductorio para que los estudiantes adquieran, no sólo una lógica de programación, sino también los elementos básicos para escribir códigos y aprender algunas de las estructuras de almacenamiento de la información. Los estudiantes que cursan este módulo ya tienen unas bases de pensamiento algorítmico; por lo tanto, se busca que en programación de computadores apliquen los conocimientos previos y los que van adquiriendo en el transcurso del mismo. Se pretende que identifiquen las características claves de un problema en un contexto dado, realicen el modelado algorítmico y su posterior implementación en un lenguaje de programación.

Este proyecto trata de la creación de un videojuego que apoya el desarrollo de habilidades en resolución de problemas y programación, para estudiantes que están iniciando su proceso de formación en el área de la ingeniería. También pretende que el estudiante realice todo el proceso anteriormente mencionado, para que (de manera lúdica) adquiera las habilidades propias de un programador y las competencias de un ingeniero.

Marco teórico

Problemática que atiende

El módulo de programación de computadores que se encuentra en la etapa de formación de los ingenieros de software y contempla los primeros acercamientos de los estudiantes a la resolución de problemas usando un computador. En el módulo virtual existen diversos materiales educativos que se centran en análisis e interpretación de códigos ya elaborados por terceros, esto es importante en el proceso de formación del futuro ingeniero, sin embargo, una de las competencias fundamentales para un programador es enfrentarse a la resolución de un problema por medio de la escritura de código, lo que implica construir un programa desde cero, es decir, crear, diseñar y elaborar la solución del problemas desde su propio proceso de razonamiento.

Este proyecto busca precisamente solventar la dificultad que hay en los ambientes virtuales, permitiendo a los estudiantes modelar un problema, construir código a través de bloques y, finalmente, llegar a la escritura del código. Como estrategia para ello, se han empleado los videojuegos porque permiten desarrollar en los jugadores cualidades como: la persistencia, la atención a los detalles, la proactividad, la creatividad, la mejora en las

capacidades perceptivas y de toma de decisiones, la capacidad de resolución a problemas, el aumento de las capacidades según el nivel de dificultad, etc. que resultan ideales en un aprendizaje para aplicar en la vida.

Antecedentes

Con el crecimiento de la oferta y demanda de los cursos abiertos y masivos en línea (MOOC, por sus siglas en inglés), ha surgido un reto respecto a la capacidad de los tutores humanos para atender a un estudiantado activo en cualquier horario y día de la semana. El reto se refiere a dos aspectos principalmente: la calificación de envíos de código que pueden envolver logros parciales (cuando el código tiene errores), y el acompañamiento tutorial de los estudiantes cuando se encuentran atascados en la resolución de un problema. Estos aspectos serían imposibles de enfrentar únicamente por medio de la intervención de un tutor e incluso de un pequeño equipo de tutores (Sharma y cols., 2018) (Glassman, 2014). De manera similar a los MOOC, los cursos virtuales de la Institución Universitaria Politécnico Grancolombiano agrupan una gran cantidad de estudiantes bajo la responsabilidad de un único tutor; por ello, resulta interesante tratar el problema de una tutoría automatizada.

Respecto al proceso de calificación y/o retroalimentación, existen múltiples propuestas enfocadas en la aplicación de distintas técnicas y metodologías. Algunas de estas propuestas se centran únicamente en la ejecución de los programas enviados por estudiantes para algunos casos de prueba y, de esta manera, determinan si el programa es correcto o no (Pieterse, 2013). Una de las críticas a este método es que se enfoca enteramente en los resultados y no analiza otros aspectos relevantes como el estilo de escritura de código, esto impide proveer una mejor realimentación al estudiante. En consecuencia, se ha usado la comparación de árboles sintácticos abstractos (ATS, por su sigla en inglés), con el fin de analizar el significado del código por medio de la estructura de los programas que los estudiantes envían (Huang y cols., 2013). Otros autores apuntan a hacer seguimiento de las trazas de variables, es decir, ver el cambio de valores que toman las variables durante la ejecución del código. Por ejemplo, Glassman y cols. (2015) combinan el seguimiento de trazas con la formación de clusters para identificar soluciones similares.

El uso de videojuegos para la escritura de código no es exclusivo de este proyecto. Existen múltiples casos de éxito en la fundamentación de la programación de computadores usando ambientes lúdicos. Es común

emplear diferentes estrategias lúdicas en la programación, sobre todo si esta es realizada por niños. Entre ellos tenemos ejemplos de lenguajes como*: Logo*[25] o *Turbo Karel*[26] , donde el usuario aprende a programar jugando.

Aunque la estrategia no se centra solo en niños, existen portales donde los usuarios pueden ingresar y mientras van aprendiendo la escritura de código van participando en un juego. Entre ellos tenemos: *Codecombat*[27], *Code. org*[28], *Human Resource Machine*[29]; mientras el usuario aprende código en Python y Javascript.

Algunas herramientas permiten la construcción del código de forma textual y otras por bloques como *Scratch*[30] y *Blockly*[31]. Scratch es una herramienta elaborada por el Instituto Tecnológico de Massachusetts. Otras universidades, como la de Greenwich, han realizado juegos para el desarrollo del pensamiento computacional. Kazimoglu y cols. (2012) afirman que el juego diseñado fortalece la adquisición de habilidades de programación (como construcción de algoritmos, depuración y simulación). El objetivo del juego es ayudar a un robot a escapar; para ello, los estudiantes programan un algoritmo de solución.

Es probable que una de las ventajas de la herramienta propuesta en este proyecto es que trabaja con bloques y con seudocódigo. Y es fundamental porque, más que trabajar un lenguaje, se pretende construir a nivel pedagógico una lógica de programación que le sirva al estudiante para llevarla y aplicarla en cualquier lenguaje.

Otra diferencia que tiene el proyecto realizado es que cuenta con una narrativa donde el estudiante debe asumir el papel del héroe o hacker y combatir a un cracker o programador mal intencionado, buscando redimir el término que ha sido mal empleado. También debe enfrentar diversos virus informáticos. Para ello, cuenta con maestros que son personajes importantes en la historia de la computación; dado que otro de los propósitos educativos es que, paralelamente al juego, aprenda algunos datos históricos de relevancia para su profesión.

25 https://ccl.northwestern.edu/netlogo/

26 http://www.jarosovi.cz/turbo-karel/

27 https://codecombat.com/

28 https://code.org/

29 https://tomorrowcorporation.com/humanresourcemachine

30 https://tomorrowcorporation.com/humanresourcemachine

31 https://blockly-games.appspot.com/

Estrategia de aprendizaje

Para construir esta estrategia de aprendizaje fue necesario revisar los posibles inconvenientes que puede tener un estudiante en un módulo de Introducción a la programación; por ejemplo, la diferencia de nivel que pueden tener los estudiantes en sus habilidades. Para atenuar tales diferencias se estructura un juego que tiene diversos niveles de dificultad; de manera que los estudiantes avanzados lo sientan como un reto, y los estudiantes iniciales recorran toda la ruta de aprendizaje. Cada problema que resuelve el estudiante aumenta el nivel de dificultad que enfrenta. Esto es acorde con la cultura gamer, que es muy similar a la espiral del aprendizaje. Un gamer tiene diversos rasgos como la persistencia, la atención a los detalles, la proactividad, aumento en la creatividad, en las capacidades perceptivas y de toma de decisiones, que resultan ideales para aplicar en la vida.

En las nuevas hipermediaciones se hace necesario generar estrategias didácticas mediadas con herramientas de software, donde el estudiante recree realidades y le permitan desempeños para diversos contextos. Si a esas herramientas le incorporamos metodologías didácticas "ludificadas" con niveles motivacionales como los que logran los videojuegos, se puede incidir en los circuitos neurales asociados a los procesos de aprendizaje (Reig, 2013, p.40). Es decir, con un juego como el que se propone en este proyecto, podremos generar una enseñanza más profunda y significativa; una centrada en la aplicación del conocimiento, que conciba la enseñanza como una urdimbre entre saber, realidad, actuación y tecnología.

Para el diseño pedagógico del juego se tuvo en cuenta que en el módulo de Programación de Computadores, al ser introductorio, algunos estudiantes no tienen conocimientos previos. Por ende, no tienen una apropiación del pensamiento computacional necesario para realizar las soluciones que son presentadas como secuencias de instrucciones y algoritmos. Se hace necesario que el estudiante pueda pensar como un programador al resolver un problema. Con este fin, el juego se diseñó con cuatro etapas: 1. Comprender: que se centra en entender el problema antes de empezar a resolverlo. Para ello, se crean tres maestros (Alan Turing, Ada Lovelace y George Boole) que explican y resuelven un problema inicial. 2. Practicar: este componente tiene un elemento práctico muy fuerte. Allí el estudiante resuelve diferentes problemas y va pasando de un nivel a otro, cada uno con mayor dificultad. Como elemento de gamificación, el estudiante puede conseguir estrellas que después cambiará por ayudas. 3. Resolver: aquí el estudiante se enfrenta al monstruo de nivel y puede cambiar

sus monedas por ayudas. Además, el avatar del jugador (a nivel de animación 3D) tiene una lucha contra el monstruo cuando supera el problema. Los tres niveles iniciales son para resolver problemas. No obstante, se trabaja con bloques de código que debe elegir y ordenar. Ahora viene la fase más compleja que es la 4. Competir: aquí el estudiante resuelve el problema escribiendo en pseudocódigo. Es decir, ya tiene que programar, este se traduce a un lenguaje de programación de alto nivel Python y allí se compila para comprobar si el programa es correcto. Si es así, vence al cracker, que es el villano del juego.

Otro inconveniente previsto en el aprendizaje es que el estudiante se centre en la resolución del problema y no en el lenguaje de programación. Para esto, el proyecto se desarrolla en seudolenguaje. El estudiante analiza los bloques y si es capaz de llegar a la solución, puede aplicarla en cualquier lenguaje de programación. Se pretende que el estudiante analice el problema, aplique métodos de solución y adquiera una lógica de programación, más allá de manejar un lenguaje.

Otro reto que superar es que los estudiantes adquieran las competencias, habilidades y conocimientos para el desarrollo de tareas de programación. Por ello, la estrategia se diseña para iniciar con la lectura y explicación de problemas; luego el análisis del código para la solución y, posteriormente la elección de los bloques de programación. El proceso es el mismo durante 3 niveles con 5 pruebas cada uno, lo que cambia es la complejidad del problema. Como elemento lúdico se cuenta con un mapa que lo lleva de problema a problema, igual que muchos juegos para teléfono móvil. Por último, se hace una competencia para que el estudiante sea quien escriba el código y resuelva el problema. Lo más importante en el proceso de aprendizaje, es que el estudiante vaya progresivamente de un pensamiento concreto a un estadio de representación conceptual y simbólica más adecuada al pensamiento (Araújo y Chadwick, 1988, p.40-41); y termine con la actuación de su conocimiento solucionando adecuadamente el problema.

Es parte de la naturaleza del módulo de Programación de Computadores requerir de la acumulación de habilidades y de conocimientos. Es decir, es muy difícil avanzar a lo largo del módulo si no se ha comprendido y dominado los temas anteriores. A esa misma naturaleza responde el juego elaborado. Por lo tanto, el estudiante percibirá un aumento constante en la cantidad de aspectos que debe dominar y en la dificultad de los problemas que le están retando a medida que avanza. Este aumento en la carga cognitiva se hace como un constructo multidimensional que incluye la carga mental, las tareas realizadas,

la sensación de reto y la motivación. Una tarea motivante aumenta de manera intrínseca el esfuerzo mental invertido por los estudiantes. Por ello, el juego es tan significativo en el proceso de la información porque aumenta carga y la motivación para lograr el rendimiento deseado (Wenhao y cols., 2013, p.60).

La metodología activa que se emplea para el proyecto es el aprendizaje basado por problemas (ABP), donde el propósito es generar para cada estudiante una situación problémica personalizada. Con este fin, el software se crea de tal manera que el docente o tutor pueda ingresar diferentes problemas categorizados por tipo y nivel de dificultad. Estos problemas entran a una base de datos y el software los asigna al azar al estudiante. Esto implica que dos jugadores del mismo módulo pueden estar en el mismo nivel y jugar dos problemas distintos. Lo anterior ya que es fundamental para el proceso que el estudiante se enfrente solo a la solución del problema.

El software toma del ABP que plantea una situación problémica, el estudiante analiza el problema y lo resuelve organizando adecuadamente los bloques de programación. De esa manera, pasa cinco pruebas y se enfrenta al virus o monstruo de nivel (que es un problema más complejo). Si aprueba pasa al siguiente nivel. El último nivel se enfrenta al cracker que es el malo del juego. Para este proceso, el estudiante escribe el código y lo prueba en Python; ya es un producto que le permite al estudiante corroborar el aprendizaje obtenido en el proceso. En este punto, hablamos de los metaobjetivos de la estrategia, porque tienen diversas finalidades de aprendizaje y permiten el desarrollo de habilidades de aprendizaje, la activación de procesos cognitivos y la transferencia de metodologías de acción intelectual (Restrepo, 2005, p.11).

La gamificación tiene unos componentes motivacionales que potencializan el aprendizaje. También incide en su implicación (engagement) y adopta estrategias propias de las mecánicas de juego, lo que permite crear experiencias lúdico-didácticas que puedan optimizar los resultados (Villalustre y del Moral, 2015, p.17). Para que lo anterior sea posible, se hace necesario que el juego cuente con: 1. Descripción de la misión o desafío, que la realiza un personaje denominado Database; 2. Instrucciones o reglas de juego claras. En cada escenario, el estudiante recibe las indicaciones de lo que debe realizar; 3. Identificación de retos o niveles. Cada nivel cuenta con un antagonista diferente y el estudiante debe vencerlo; 4. Asignación de puntos o premios. Los premios son criptomonedas que después puede comprar, son ayudas ante el reto del monstruo de nivel. De igual manera, los puntos se asignan de acuerdo con el desempeño del estudiante. Aquí es claro que se pretende que el juego

no sea punitivo, es decir, que no castigue los errores, sino reconozca el proceso del estudiante; y por último, 5. Competencia, es uno de los elementos más importantes de la gamificación por su componente de exigencia al estudiante. En el juego se cuenta con una competencia para vencer al que programa los virus que es un cracker llamado Anthrax.

El juego, además de ser una estrategia de aprendizaje basado en juegos, contiene diversos elementos narrativos muy importantes que no se toman al azar. Por ejemplo, los maestros son científicos que realizaron aportes muy significativos a la programación; los estudiantes hacen el papel de hackers y el antagonista es un cracker (para darle el uso apropiado al término); los monstruos de cada nivel son tomados de virus informáticos reales. Todo lo anterior, con el fin de que el juego tenga elementos que apoyen los problemas y la estrategia pedagógica. La lúdica toma cada vez mayor importancia en la educación porque según Duarte (2003): "(…) parece escapar a la pretensión instrumentalista que caracteriza a la escuela" y se centra en el placer y la exploración, que facilita en el estudiante el autoaprendizaje y la autorregulación.

Modelo para la simulación

La aplicación ha sido construida en seudocódigo con la intención de eliminar las particularidades de los lenguajes de programación. De esta manera, cada estudiante se enfocará en lo que es más importante en el módulo: resolver problemas por medio del uso de un computador. Para asegurar la independencia de todos los lenguajes de programación, se usan palabras y términos que cualquier estudiante con conocimientos básicos de matemáticas y de la lengua castellana puede fácilmente identificar. Asimismo, se han eliminado símbolos que indiquen el inicio o final de estructuras o conjuntos de instrucciones y se han reemplazado por la indentación del código, algo que resulta mucho más genérico y con mayor facilidad para el usuario.

El simulador está destinado a cultivar las habilidades de los estudiantes en un paradigma de programación llamado programación estructurada. La programación estructurada se basa enteramente en la utilización de tres estructuras de control fundamentales:

1. **Secuenciales:** ejecución de una instrucción tras otra.
2. **Condicionales:** ejecución de uno de dos conjuntos de instrucciones de acuerdo con el valor que tome una expresión lógica.

3. **Iterativas:** ejecución de un conjunto de instrucciones mientras que el valor resultante de una expresión lógica sea VERDADERO.

Además de estas estructuras de control, el simulador permitirá al estudiante conocer y manipular dos estructuras básicas de datos: los arreglos y las matrices. Arreglos y matrices son formas de organizar los datos de manera que puedan ser operados. El conocimiento de estas estructuras habilitará al estudiante para resolver una gama más amplia de problemas. Adicionalmente, servirá de fundamentación para el módulo Estructuras de Datos que deberán cursar algunos de los estudiantes que aprueben el módulo de Programación de Computadores.

El juego se diseñó para que sea trabajado por bloques. Este es un paradigma que permite conectar las piezas de código de manera similar al armado de un rompecabezas. En cada bloque se encuentran elementos relacionados con las tres estructuras de control básicas de la programación estructurada, anteriormente reseñadas.

De igual manera, el juego pretende desarrollar en los estudiantes las siguientes capacidades en programación:

- Resolución de problemas aplicando las estructuras básicas de control, de datos y recursión.
- Identificar las entradas y salidas del problema para una mayor comprensión del mismo.
- Desarrollar un pensamiento algorítmico que le permita al estudiante enfrentarse a diversos retos computacionales.
- Identificar variables o elementos relevantes de un problema y modelarlos matemáticamente.
- Discriminar instrucciones y datos irrelevantes para proponer la solución más depurada de un problema.

Finalmente, el tema de los métodos recursivos se implementará como una forma de pensar alternativa al uso de instrucciones iterativas en los programas. Este tema se introducirá por medio del uso de funciones matemáticas recursivas que son traducidas a métodos recursivos. El estudiante, después de terminar el juego, debería ser capaz de identificar cuáles son las características de los métodos recursivos y de implementarlas con éxito.

Desarrollo de la experiencia de aprendizaje

Cuando se habla de un módulo introductorio a la programación siguiendo una programación estructurada es importante considerar, según Moroni y Señas (1996): a) ¿Cómo expresar con precisión el problema a resolver?; b) ¿Qué estrategia de resolución elegir?; c) ¿Cómo plasmar la resolución en un algoritmo? y d) ¿Cómo escribir el programa correspondiente en un lenguaje de programación?

Para expresar con precisión el problema a resolver se incluye en el juego una narrativa que explica el problema y orienta al estudiante en la solución. Luego debe ordenar los bloques como si fueran puzzles y dar solución al problema. Si lo hace de manera correcta, los bloques quedan en verde; si corresponden a la solución, pero están mal ubicados, salen en amarillo; y si no corresponden, salen en rojo. De esa manera, el estudiante puede analizar cómo fue su lógica de programación. Para esto se creó una plantilla que permite agregar problemas al juego, de manera que a futuro se vuelva una base de datos de problemas de programación para resolver.

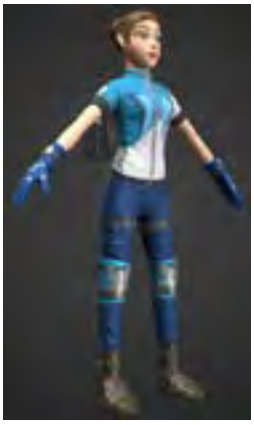
La estrategia de resolución que se estableció para los tres primeros niveles fue por bloques. Lo que va en aumento es la complejidad de los problemas. La última prueba se hace por seudocódigo, para que sea el estudiante quien construya la solución. El juego le da el problema y el estudiante debe resolverlo escribiendo el seudocódigo. Este campo inicialmente se planteó como un enfrentamiento en línea entre estudiantes; sin embargo, no se continuó con esta idea por las características de la modalidad virtual. Si era una olimpiada, surgían estas preguntas: ¿Cómo se hacía con los estudiantes que no ingresaran? ¿Cómo se declaraba un ganador si no había competencia? Por lo tanto, se prefirió dejar que el estudiante escribiera el seudocódigo y el tutor evaluara su desempeño.

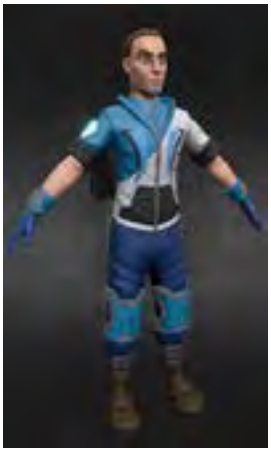
Para plasmar la solución, se trabajan los bloques; y para evitar que el programa sea de un lenguaje, se hizo en pseudocódigo. Obviamente se va a programar lo que permita traducir el programa a Python.

Con estos puntos claros se construyeron los problemas y los bloques de solución en la plantilla. Se crearon los mockups para cada uno de los niveles. Asimismo, se realizó la narrativa para cada uno de los maestros que guían al estudiante.

Uno de los procesos más complejos fue el modelado y animación 3D. Cada personaje tuvo que hacerse por separado conservando la estética, según las solicitudes dadas en el guion. La Figura 6, muestra el modelado de algunos de los personajes del juego. A la izquierda están los avatares que puede elegir el estudiante, a la derecha los monstruos de los dos primeros niveles:

Héroes





Villanos

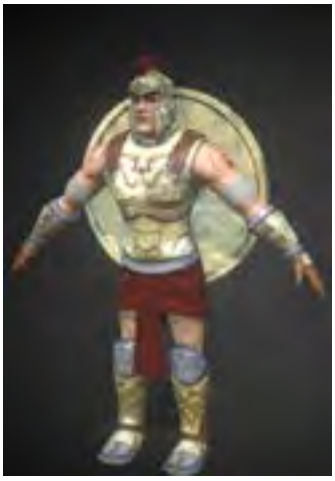


Figura 6. Personajes del juego

Fuente: elaboración propia

El proceso de programación se integró con las animaciones 3D realizadas en Maya. El proyecto se programó con *Unity* y *Python*.

Resultados obtenidos

El empleo de un juego para programación no es un evento nuevo. Los investigadores de este proyecto ya venían utilizando experiencias basadas en juegos para sus cursos presenciales. Allí los aprendices compiten en un formato similar a una copa deportiva, donde se van eliminado hasta obtener un campeón. La diferencia para el caso de este proyecto es su estructura apta para una modalidad de aprendizaje virtual, donde el individuo juega contra la máquina y debe hacerlo durante el transcurso de su módulo. Debido a la dificultad de sincronía entre los estudiantes de esta modalidad, el formato de competencia cambia de copa deportiva a superación de pruebas que aumentan la dificultad a superar. En este punto no se le da tanta relevancia a la competencia con el otro, sino que se enfoca en el reto personal.

Este cambio de formato representó para los proponentes una modificación significativa del diseño, tanto desde la perspectiva pedagógica como desde la estructura y programación del juego. En un inicio, solo se tenían concebidos dos niveles por bloques y dos niveles de competencia tipo copa. Entonces, se analizó desde lo educativo qué pasa con el estudiante que es eliminado; cómo definir que una persona lo hace mejor que otra; quién le asegura al estudiante A que el estudiante B va a estar en la competición o viceversa, etc. Estas son dinámicas que deben tenerse en cuenta cuando se diseña un juego para estudiantes en línea. Esto es muy diferente a los jugadores en línea (gamers), donde el jugador se enfrenta con quien esté en ese momento; a nivel pedagógico se hace necesario que todos los estudiantes jueguen y eliminarlos es contraproducente para una continuidad en la adquisición de la competencia. Por eso se establece un diseño de problemas que reten al estudiante.

Los propósitos perseguidos en la implementación del simulador y que fueron presentados en la sección de Antecedentes se refieren al proceso de calificación / retroalimentación y al acompañamiento tutorial. La calificación se obtiene de manera automática y corresponde al nivel de logro de cada estudiante. La retroalimentación ocurre de manera inmediata, pues cada jugador puede ver reflejado de manera gráfica el tipo de error que está cometiendo en la resolución de los retos. Finalmente, el acompañamiento tutorial mejora gracias a la existencia de tres espacios diferentes: instrucción por medio del

ejemplo; práctica libre en un proceso que ofrece recompensas y no castiga; y comprobación de los conocimientos en un ambiente flexible.

Conclusiones

La metodología del aprendizaje basado en problemas aplicada en un juego se vuelve, no solo retadora, sino motivante para el estudiante. Este debe enfrentarse a los problemas desde el juego, lo que elimina barreras al afrontarlo; permitiéndole una resolución desde la tranquilidad, así como reforzar los conocimientos aprendidos con situaciones que requieren su pensamiento analítico y creativo. Por eso la filosofía de este juego es evitar ser punitivo. Sin embargo, sí pretende comprometer al estudiante con su aprendizaje; y, sobre todo, busca a un individuo capaz de resolver eficientemente problemas por medio de la programación de computadores.

El simulador propuesto funciona a manera de juego didáctico que mezcla dos etapas: la aventura y la competencia. Durante la etapa de aventura, el estudiante enfrenta un proceso individual que le permite mantener un ritmo propio de aprendizaje, mediante el uso de una herramienta que le permite desarrollar la lógica sin depender directamente de las reglas de un lenguaje. Por otro lado, la etapa de competencia entre pares promueve el compromiso y aumenta la motivación de los estudiantes. La presente experiencia acude a estos dos tipos de juego para tener oportunidad de cautivar a un mayor número de estudiantes.

El proceso narrativo de un juego es una excelente excusa para involucrar elementos que pueden estar desagregados de la naturaleza del curso, pero que siguen siendo relevantes en la formación disciplinar. En el caso específico que se presenta, se introdujeron aspectos históricos del desarrollo de la computación por medio de la presentación de tres personajes relevantes en ese aspecto. Asimismo, se trata una problemática asociada a la seguridad informática: la existencia de virus informáticos y de los crackers, personas que irrumpen en sistemas informáticos con el fin de dañarlos. Este tipo de contenido es de difícil presentación en un curso regular de Programación de Computadores, pero se adecua muy bien al uso de narrativas digitales.

El juego fue diseñado con el objetivo de incrustarse en un ambiente virtual. Es decir, en un espacio en el que puede haber muchos más estudiantes que en la modalidad presencial sin necesidad de coincidir en un momento preciso

del tiempo. Sin embargo, es suficientemente versátil como para ser usado en cursos muchos más grandes como los MOOC o como extensión de la instrucción docente del aula en un curso presencial. Asimismo, sería pertinente en cursos de extensión.

Referencias bibliográficas

Araújo, J. B. y Chadwick, C.B. (1988). Tecnología educacional. *Teorías de la instrucción. Paidós.*

Duarte, J. (2003). Ambientes de aprendizaje. Una aproximación conceptual. *Revista Estudios Pedagógicos,* 29, 97-113. http://dx.doi.org/10.4067/S0718-07052003000100007

Glassman, E. L. (5-8 de octubre de 2014). Interacting with massive numbers of student solutions. En H. Benko (Presidencia), *27th Annual ACM Symposium on User Interface Software and Technology* - UIST'14, llevado a cabo en Honolulú. https://dl.acm.org/doi/pdf/10.1145/2658779.2661167

Glassman, E. L., Fischer, L., Scott, J. y Miller, R. C. (8-11 de noviembre de 2015). Foobaz: Variable Name Feedback for Student Code at Scale. En C. Latulipe (Presidencia), 28th *Annual ACM Symposium on User Interface Software & Technology llevado a cabo en Charlotte*, E.E.U.U. https://dl.acm.org/doi/pdf/10.1145/2807442.2807495

Huang, J., Piech, C., Nguyen, A. y Guibas, L. J. (9-13 de julio de 2013). Syntactic and Functional Variability of a Million Code Submissions in a Machine Learning MOOC. En N. Le (Presidencia), *AIED Workshops* llevados a cabo en Memphis. https://www.researchgate.net/profile/Zachary_Pardos/publication/299707248_First_Annual_Workshop_on_Massive_Open_Online_Courses/links/5a188e43aca272df080a864c/First-Annual-Workshop-on-Massive-Open-Online-Courses.pdf#page=30

Kazimoglu, C., Kiernan, M., Bacon, L. y Mackinnon, L. (2012). A serious game for developing computational thinking and learning introductory computer programming. *Procedia - Social and Behavioral Sciences*, 47, 1991-1999. https://doi.org/10.1016/j.sbspro.2012.06.938

Moroni, N. y Señas, P. (1996). Un entorno para el aprendizaje de la programación. En el *II Congreso Argentino de Ciencias de la Computación* llevado a cabo en Bahía Blanca, Argentina.

Pieterse, V. (2013). Automated Assessment of Programming Assignments. *CSERC*, 13, 4-5. https://www.researchgate.net/profile/Vreda_Pieterse/publication/262328132_Automated_Assessment_of_Programming_Assignments/links/54c8a76b0cf289f0ced09629.pdf

Reig, D. (2013). Describiendo al hiperindividuo, el nuevo individuo conectado. En D. Reig y L.F. Vílchez (Eds.). *Los jóvenes en la era de la hiperconectividad: tendencias, claves y miradas* (pp. 21-81). Fundación Telefónica y Fundación Encuentro.

Restrepo, B. (2005). Aprendizaje basado en problemas (ABP) una innovación didáctica para la enseñanza universitaria. *Revista Educación y Educadores,* 8, 9-20.

Sharma, S., Agarwal, P., Mor, P. y Karkare, A. (2018). *TipsC: Tips and Corrections for Programming* MOOCs. https://arxiv.org/pdf/1804.00373.pdf

Villalustre, L. y del Moral, M. (2015). *Gamificación*: Estrategia para optimizar el proceso de aprendizaje y la adquisición de competencias en contextos universitarios. *Digital Education Review*, 27, 13-31. https://doi.org/10.1344/der.2015.27.13-31

Wenhao, D. H., Tristan, J. y Seung-Hyun, C. (2013). Impact of online instructional game features on college students perceived motivational support and cognitive investment: A structural equation modeling study. *Internet and Higher Education Journal,* 17, 58–68.