CIS1410IS11

Modelo para la Motivación del Aprendizaje de la Programación utilizando Gamification

Ricardo José Arenas París

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA DE SISTEMAS
BOGOTÁ, D.C.
2014

CIS1410IS11
Modelo para la Motivación del Aprendizaje de la Programación utilizando Gamification

Autor:
Ricardo José Arenas París

MEMORIA DEL TRABAJO DE GRADO REALIZADO PARA CUMPLIR UNO DE LOS REQUISITOS PARA OPTAR AL TÍTULO DE INGENIERO DE SISTEMAS

Director
Fabio Antonio Avellaneda Pachón
Asesor
Cristian David Romero Melgarejo

Jurados del Trabajo de Grado
Rafael Andrés González Rivera
Esteban Ocampo Flórez

Página web del Trabajo de Grado
http://pegasus.javeriana.edu.co/~CIS1410IS11

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA DE SISTEMAS
BOGOTÁ, D.C.
Mayo, 2014

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA DE SISTEMAS

Rector Magnífico

Padre Jorge Humberto Peláez Piedrahita, S.J.

Decano Académico Facultad de Ingeniería

Ingeniero Jorge Luis Sánchez Téllez

Decano del Medio Universitario Facultad de Ingeniería

Padre Antonio José Sarmiento Nova S.J.

Director de la Carrera de Ingeniería de Sistemas

Ingeniero Germán Alberto Chavarro Flórez

Director Departamento de Ingeniería de Sistemas

Ingeniero Rafael Andrés González Rivera

Artículo 23 de la Resolución No. 1 de Junio de 1946

"La Universidad no se hace responsable de los conceptos emitidos por sus alumnos en sus proyectos de grado. Sólo velará porque no se publique nada contrario al dogma y la moral católica y porque no contengan ataques o polémicas puramente personales. Antes bien, que se vean en ellos el anhelo de buscar la verdad y la Justicia"

AGRADECIMIENTOS

Agradezco a mi familia por todo su apoyo durante toda la carrera y durante el desarrollo de este trabajo. Sin sus sacrificios, su comprensión y su ayuda, no estaría tan cerca de completar esta etapa de mi vida.

También doy las gracias a Fabio y a Cristian por su acompañamiento constante y su compromiso con el trabajo para asegurar que tuviera la mejor calidad posible. No es común encontrar un grupo de directores que esté dispuesto a reunirse un domingo a las 9pm para asegurarse que el trabajo sea todo lo que puede ser. Agradezco también a Nicolás por toda la asesoría en un área en la que no teníamos tanto conocimiento. Sin su ayuda este trabajo nunca hubiera finalizado.

A Fabio además le agradezco su amistad y guía durante toda la carrera, Y aunque sus consejos en ocasiones significaron dolores de cabeza, siempre terminaron ayudándome a crecer profesionalmente y como persona.

Agradezco también al Capítulo Javeriano ACM, en especial al grupo de Maratones de Programación. Nunca había conocido un grupo de personas tan comprometidas con su universidad y con su carrera. Gracias a ellos encontré mi pasión por la programación, un hobby que se convirtió en una carrera profesional, y un grupo de amigos que siempre ha estado presente y me ha mantenido apuntando hacia lo más alto.

Por último agradezco a los profesores del departamento de Ingeniería de Sistemas por todos los conocimientos y oportunidades que me brindaron durante estos cinco años de carrera.

Contenido

Tabla de Figuras

ABSTRACT

This work describes a model for the learning of programming using Gamification techniques to solve the motivational problems behind the learning and teaching of programming. The proposed model uses the D6 Gamification framework to ensure its sane and proper design, along with educational principles that ensure the model is useful for students. A prototype for a software tool was developed using the WordPress CMS and the BadgeOS plugin to show the technical feasibility of implementing the model. A series of experts of different areas judged the model to be an innovative approach to trying to teach programming and recommend its implementation as a tool to support a traditional programming course.

RESUMEN

Este trabajo describe un modelo para el aprendizaje de la programación utilizando técnicas de *Gamification* para resolver problemas motivacionales que surgen en el proceso de su aprendizaje y enseñanza. El modelo utiliza el framework D6, asegurando que el diseño de los elementos de *Gamification* sea apropiado, junto con principios pedagógicos para asegurar su utilidad para los estudiantes. Se realizó un prototipo de una herramienta computacional, incluyendo algunos elementos del modelo para demostrar la factibilidad técnica de su implementación. Un grupo de expertos determinó la validez del modelo y recomendó su implementación como herramienta para apoyar un curso tradicional de programación.

RESUMEN EJECUTIVO

La programación y la algoritmia son temas que están tomando una importancia cada vez mayor en nuestra sociedad. Algunos autores incluso afirman que aprender a programar es tan importante como aprender a leer y escribir. Sin embargo, la enseñanza de estos conceptos ha presentado dificultades desde distintas áreas, especialmente las del estilo de aprendizaje necesario para aprender programación, la desmotivación de los estudiantes y los conceptos específicos del área.

Gamification, el uso de mecánicas de juegos en diferentes contextos, ha surgido como una propuesta para incrementar el compromiso y mejorar la motivación de los participantes en diferentes procesos.

Este trabajo presenta un modelo que integra elementos de Gamification al proceso de aprendizaje de la programación con el fin de resolver algunos de los problemas antes mencionados.

Se realizó una investigación sobre las técnicas de Gamification existentes y de los problemas en el aprendizaje de la programación que estas técnicas podrían ayudar a resolver. Se eligió un proceso de diseño para sistemas gamificados que ha sido ampliamente documentado y utilizado para asegurar el diseño riguroso del modelo. Para demostrar que es posible apoyar la implementación del modelo mediante una herramienta tecnológica, se implementó un prototipo de una herramienta de software que incluye algunos de los elementos del modelo y que es posible extender para incluir nuevas funcionalidades.

Se solicitó a tres expertos validar el modelo desde su experiencia en las áreas de la pedagogía y de la enseñanza de la programación. Ellos determinaron que el modelo presenta suficientes bases teóricas y es razonable esperar que se puedan resolver los problemas motivaciones de los estudiantes mediante su implementación.

Se concluye que utilizar Gamification para resolver problemas motivacionales es una opción interesante e innovadora, y es apropiado realizar una implementación completa del modelo para apoyar un curso de programación.

Se plantean posibilidades de realizar trabajos futuros en los que se extienda el prototipo planteado para implementar la totalidad del modelo, trabajar en conjunto con profesionales de la pedagogía para la creación de contenido, mejoras en el sistema de calificación del prototipo y de personalización y recomendación de contenido para los estudiantes.

INTRODUCCIÓN

Se ha descubierto que uno de los factores con mayor influencia en las dificultades de los estudiantes para el aprendizaje de la programación es la motivación [1, 2]. Por su parte, Gamification ha surgido en los últimos años como una propuesta para mejorar la interacción, el compromiso y la motivación de quienes participan en diferentes procesos [3, 4]. Este documento presenta el trabajo realizado para aplicar estas técnicas al aprendizaje de la programación.

En primer lugar se presenta información general del proyecto, donde se presenta la problemática que se deseaba abordar, así como los objetivos planteados para él junto con la metodología propuesta para cumplirlos. A continuación se presentan las bases teóricas bajo las cuales se desarrolló el proyecto. Se recopila una definición de Gamification y las mecánicas de juegos que hacen parte de esta técnica, se exploran los problemas en el aprendizaje de la programación que han sido identificados por varios autores para determinar cuáles de ellos podrían ser solucionados con este concepto, y se describen varios procesos de diseño reconocidos para implementar Gamification, de los cuales se elige uno para realizar la propuesta.

Posteriormente se utiliza el framework D6 de Kevin Werbach [5] para plantear el modelo y se hace la sustentación teórica, desde la óptica de la pedagogía, de por qué el modelo puede ayudar a solucionar los problemas de aprendizaje mientras brinda métodos para asegurar que se realice una apropiación profunda y duradera de conocimientos para quienes lo utilicen.

Luego, para demostrar la viabilidad técnica del desarrollo del modelo, se presenta un prototipo de una plataforma de software que implementa algunos de sus aspectos. Se examinan algunas herramientas base que proveen elementos de ‘gamification’, así como los criterios de selección utilizados para tomar una decisión sobre cuál de ellas utilizar para el prototipo. También se presentan los componentes del modelo que se decidió implementar en el prototipo, así como una explicación de cómo fueron desarrolladas.

Enseguida se presenta la validación realizada por expertos de las áreas de la pedagogía y de la educación de la programación para determinar la relevancia y consistencia teórica del modelo, así como su concepto sobre su pertinencia para motivar a los estudiantes a aprender programación.

Finalmente se presenta la discusión y conclusiones encontradas en el diseño del modelo propuesto, junto con ideas para trabajos futuros que podrían ser llevados a cabo para agregar valor al modelo.

I - DESCRIPCIÓN GENERAL DEL TRABAJO DE GRADO

1. Oportunidad, Problemática, Antecedentes

1.1. Descripción de la problemática y Antecedentes

La programación y la algoritmia son dos temas que todo estudiante de ciencias e ingeniería debería conocer pues son áreas de conocimiento que en esta era están a la par con las matemáticas, la lectura y la escritura [6]. "Así como el conocimiento básico en lenguajes nos ayuda a comunicarnos y las matemáticas básicas nos ayudan a pensar de forma cuantitativa, el pensamiento computacional nos ayuda a procesar información y tareas de una forma eficiente y sistemática" [7].

Sin embargo, la enseñanza y el aprendizaje de la programación han presentado ciertas dificultades tanto para los profesores como para los estudiantes y en general, no se ha logrado alcanzar su objetivo de desarrollar este pensamiento de alto nivel que puede ayudarles a resolver problemas de cualquier índole [8]. Algunos de los problemas específicos de aprender esta habilidad son:

1. Aprender a descomponer un problema en partes más pequeñas e independientes unas de otras [8][9].
2. Utilizar un lenguaje con instrucciones y estructuras arbitrarias para resolver un problema [10][9].
3. Encontrar y corregir errores en el código desarrollado [10][9].

Debido a estos y otros problemas [10][11], el aprendizaje de la programación crea algunas veces frustración en el estudiante, la cual hace que la tasa de deserción al aprender este tema sea alta.

En Colombia, las carreras afines a las ciencias de la computación tienen los segundos índices de deserción más altos del país [12]. 50% de los estudiantes que ingresan a un programa de educación superior en esta área se retiran antes de completar el quinto semestre y solo el 35.5% pasa del décimo semestre. Adicionalmente, según el Observatorio Laboral para la Educación, entre el año 2001 y el 2011, el número de aspirantes para iniciar

un programa de educación en estas áreas se ha reducido en casi un 50% [13]. Estas cifras contrastan con estudios que ubican a la ingeniería de sistemas y carreras afines entre los primeros cinco puestos en el listado de carreras más demandadas por las empresas colombianas [14].

Se encontró que se han utilizado varias estrategias para atacar los problemas que surgen al aprender y enseñar a programar. Entre estas se hallan:

- La creación de lenguajes intermedios que tratan de hacer más fácil la transición del lenguaje natural a los lenguajes de programación [15][16].
- Herramientas donde se programa gráficamente mediante la manipulación de personajes y de un entorno virtual [17][18].
- Juegos de video donde se usa la programación como un elemento propio del juego [19].

El primer tipo de solución asume que la dificultad para programar surge del aprendizaje de un nuevo lenguaje, sin tener en cuenta los problemas inherentes mencionados con anterioridad [8][9][10].

Los otros dos tipos intentan responder a la frustración al convertir la programación en un juego, pero se basan en la premisa que para hacer que la programación sea divertida, los estudiantes deben utilizar entornos y lenguajes que los hagan programar sin escribir código, de tal forma que no pueden diseñar sus propios algoritmos y se encuentran limitados por lo que la herramienta les permite hacer [20].

Sin embargo, el éxito que han demostrado tener este tipo de soluciones [20][21] en ayudar a los estudiantes a aprender a programar y a desarrollar conceptos básicos de diseño de algoritmos hace evidente que aplicar mecánicas de juego para aprender es efectivo y es un campo que vale la pena explorar más a fondo.

Adicionalmente, en Colombia se han desarrollado metodologías pedagógicas que intentan resolver los problemas de aprendizaje de la programación. Una de estas es el proyecto CUPI2 de la Universidad de los Andes [22]. CUPI2 se basa en cuatro componentes: el aprendizaje incremental, el aprendizaje basado en problemas, el uso de herramientas tecnológicas y la comunidad de enseñanza de la programación que se ha construido para

dar soporte a la comunidad académica. El impacto en los estudiantes es medido principalmente mediante la disminución en la mortalidad en los cursos que implementan la metodología y encuestas de satisfacción de los estudiantes. Aunque la metodología tiene una apropiada fundamentación teórica y ha tenido un impacto significativo, estas métricas no dan cuenta de los conocimientos o habilidades que han adquirido los estudiantes.

1.2. **Formulación**

¿Es posible usar los elementos y la mecánica de los juegos para motivar el aprendizaje de la programación por parte de los estudiantes?

1.3. **Justificación**

Gamification es el uso de técnicas de diseño de juegos en contextos que no son juegos [23][3]. Esto significa que se pueden tener algunas bondades de los juegos como la diversión, la tolerancia al fracaso y el compromiso [3] sin necesidad de convertir la actividad completamente en un juego.

En los últimos años, se han realizado estudios que muestran que aplicar elementos de *gamification* en el proceso educativo ayuda a aumentar el interés de los estudiantes por los temas estudiados, a disminuir la frustración y a que se esfuercen por obtener mejores calificaciones [24][25].

Fue pertinente entonces realizar una investigación que determine si en el contexto colombiano, integrar algunos elementos de *gamification* en el aprendizaje de la programación puede ayudar a mitigar algunos de los problemas mencionados en secciones anteriores y motivar a los estudiantes para que aprovechen los beneficios que trae adquirir estos conocimientos.

2. **Descripción del Proyecto**

2.1. **Objetivo general**

Crear un modelo que aplique técnicas de *gamification* al proceso de aprendizaje de conceptos básicos de programación y algoritmia.

2.2. **Objetivos Específicos**

- Elegir las técnicas de *Gamification* a utilizar y los problemas de aprendizaje de la programación que se intentarán resolver.
- Diseñar un modelo que mitigue los problemas elegidos mediante el uso de los elementos de *Gamification* seleccionados.
- Desarrollar un prototipo de una plataforma que implemente el modelo planteado.
- Diseñar y aplicar un protocolo de validación del Modelo.

2.3. **Método que se propuso para satisfacer cada fase metodológica**

Esta sección presenta las actividades que se realizaron para cumplir con cada uno de los objetivos propuestos. Cabe aclarar que cada objetivo específico representó una fase metodológica del trabajo.

2.3.1. **Fase 1**

Elegir las técnicas de *Gamification* a utilizar y los problemas de aprendizaje de la programación que se intentarán resolver.

Actividades

- Elaborar el Estado del arte en los temas de:
    - Gamification.
    - Aplicación de Gamification en entornos educativos y específicamente en áreas de ciencias de la computación y programación.
    - Problemas típicos en el aprendizaje de la programación y estrategias para resolverlos.
    - Frameworks de diseño para Gamification.
- Seleccionar los problemas de aprendizaje que se resolverán mediante el modelo.
- Seleccionar los elementos de *gamification* a utilizar en el modelo.
- Elegir un framework de diseño de *Gamification* para el desarrollo del modelo.

2.3.2. **Fase 2**

Diseñar un modelo que mitigue los problemas elegidos mediante el uso de los elementos de *Gamification* seleccionados.

Actividades

- Estudiar y definir cómo aplicar los elementos de *gamification* seleccionados para resolver los problemas de aprendizaje.
- Diseñar un modelo que integre los elementos de *gamification* al proceso de aprendizaje de conceptos básicos de programación utilizando el framework de diseño seleccionado.

2.3.3. **Fase 3**

Desarrollar un prototipo de una plataforma que implemente el modelo.

Actividades

- Recopilar información de plataformas similares.
- Identificar las necesidades que debe satisfacer la plataforma.
- Elegir las herramientas de software que se usarán para el desarrollo del prototipo.
- Especificar los elementos de la plataforma que serán implementados por el prototipo.
- Realizar y validar el diseño del prototipo.
- Implementar el prototipo.
- Corregir errores en el prototipo.

2.3.4. **Fase 4**

Aplicar un protocolo de validación al modelo.

Actividades

- Elegir la metodología de validación a utilizar.

- Diseñar el protocolo de validación para el modelo propuesto.
- Realizar la validación del modelo.

II – Fundamentación Teórica

1. Gamification y Mecánicas de Juego

Según Deterding, el primer uso del término Gamification surgió en el año 2008 en el sector de medios digitales [3]. Desde esa época, su uso se ha extendido a muchos otros dominios y han surgido numerosas definiciones para tratar de acotar su significado.

Una de las definiciones más aceptadas es la de Deterding, quien dice que "es el proceso de usar el pensamiento y las mecánicas de juegos en contextos distintos a estos con el fin de resolver problemas y comprometer a los usuarios" [3]. Por su parte, Lee lo define como "el uso de elementos, dinámicas y frameworks de juegos para promover comportamientos deseados en contextos distintos a los juegos" [24]. Una tercera definición por Erenli lo describe como "intentar aprovechar el poder motivacional de los juegos y aplicarlo al mundo real"[23].

Las técnicas de Gamification funcionan combinando elementos familiares para la mayoría de personas nacidas después de 1970, cuando los videojuegos empezaron a surgir como un medio popular de entretenimiento [26]. Al combinar estos elementos en el diseño de la experiencia para los participantes, se espera hacer más factible que la persona esté más involucrada y atienda con más frecuencia determinada acción. Al combinar estos elementos de tal forma que un proceso sea una experiencia disfrutable para su participante, se espera hacer más factible que la persona esté más involucrada y siga ejecutando el proceso con frecuencia.

Con este objetivo en mente, se han tratado de separar los elementos de los juegos que pueden resultar de utilidad para 'gamificar' un proceso. Una clasificación de estos elementos propuesta por Werbach y Hunter [5] los divide en tres tipos:

1.1. **Dinámicas**

Estos son los elementos más conceptuales y de alto nivel de un juego. Son los que le dan significado al juego y hacen que toda la experiencia tenga sentido. Entre las dinámicas posibles se pueden encontrar:

- **Restricciones:** Cuáles son las restricciones que impone el juego sobre el mundo real. Algunas de las reglas del juego entran en esta categoría.
- **Emociones:** Cuáles emociones busca inspirar el juego en sus jugadores.
- **Narrativa:** Cuál es la historia que el juego intenta contar, y de qué manera debe ser entregada al jugador.
- **Progresión:** Cómo el juego hace que el jugador sienta que ha obtenido nuevas habilidades y que el tiempo gastado en el juego ha sido recompensado.
- **Relaciones:** Cómo permite el juego establecer relaciones con otros jugadores o no jugadores.

1.2. **Mecánicas**

Estos elementos son aquellos que permiten realizar las dinámicas y hacen que el juego avance. Entre las mecánicas se pueden encontrar, aunque no están limitadas a:

- **Retos:** El jugador avanza en el juego al superar obstáculos y resolver problemas.
- **Competencia:** La competencia con otros jugadores puede hacer que un jugador reconozca su desempeño en el juego y genere relaciones de rivalidad.
- **Cooperación:** La cooperación entre jugadores permite avances más rápidos y establece relaciones diferentes a la rivalidad.
- **Retroalimentación:** Cuando el jugador recibe respuestas a sus acciones siente que su participación en el juego importa.
- **Recompensas:** Consiste en los premios que obtiene un jugador al cumplir con los objetivos, seguir las reglas, participar en el juego, entre otras. Muchos de los componentes de los juegos están enfocados a recompensar al jugador.
- **Estados de Victoria:** Cuáles son las condiciones que permiten a un jugador progresar de un nivel a otro o avanzar en la narrativa.

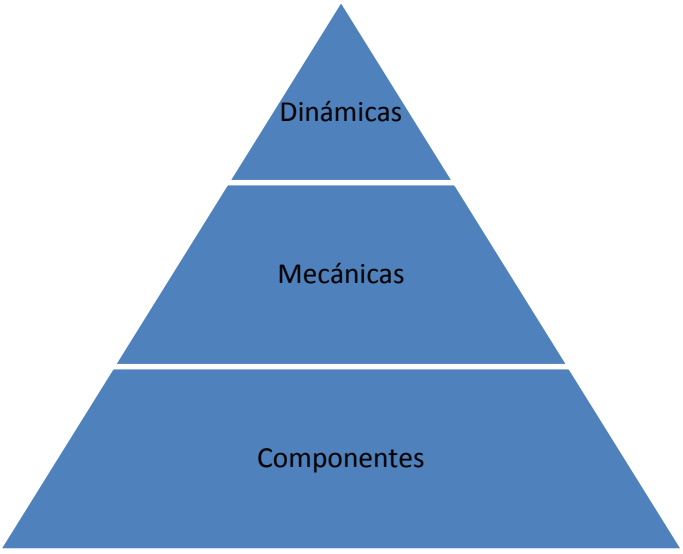
1.3. **Componentes**

Estos son los elementos de más bajo nivel que permiten implementar las mecánicas y las dinámicas de una manera específica. Algunos de los componentes utilizados en los juegos son:

- **Logros:** El jugador es recompensado por una acción específica.
- **Avatares:** Es la representación del jugador en el mundo del juego.
- **Medallas:** Son representaciones visuales de los logros obtenidos por el jugador.
- **Colecciones:** Permite al jugador ver el acumulado de logros u otro contenido desbloqueado al que se ha hecho merecedor.
- **Desbloqueo de Contenido:** A medida que el jugador avanza en el desarrollo del juego y adquiere nuevas habilidades, se le permite acceder a contenido que antes era restringido.
- **Tableros de puntaje:** Permite al jugador compararse directamente con otros jugadores.
- **Niveles:** Es una forma de mostrar el avance del jugador. Un jugador de mayor nivel generalmente representa que tiene más habilidad que un jugador de nivel bajo.
- **Puntos:** Es una medida objetiva de qué tanto ha avanzado un jugador, además, funciona como un símbolo de estatus.
- **Búsquedas:** Son retos o problemas puntuales que debe resolver el jugador con el objetivo de recibir alguna recompensa o retroalimentación.

La relación entre estos tres tipos de elementos se puede observar en la Figura 1:

Figura 1 Mecánicas de juego



2. **Problemas en el Aprendizaje de la Programación**

La programación y la algoritmia son dos temas que todo estudiante de ciencias e ingeniería debería conocer pues son áreas de conocimiento que en esta era están a la par con las matemáticas, la lectura y la escritura [6]. Sin embargo, los estudiantes universitarios no aprecian estos temas debido a distintas razones dependiendo del contexto en el que se les trata de enseñar [1]. Por esta razón se han realizado estudios y diseñado diferentes estrategias para lograr que más estudiantes traten de aprender estos temas y para determinar las causas por las cuales algunos estudiantes presentan dificultades para aprenderlos [8–11].

Entre los problemas se pueden encontrar los diferentes estilos de aprendizaje de los estudiantes, los problemas inherentes en los contenidos de los cursos y los problemas motivacionales.

2.1. **Estilos de Aprendizaje**

La clasificación mejor conocida para los estilos de aprendizaje es aquella que los divide en estilo superficial y de profundidad. Los estudiantes adquieren preferencia por alguno de los

estilos dependiendo de las capacidades de cada uno, de sus experiencias previas de aprendizaje y de los mecanismos educativos de profesores previos [28].

Aquellos estudiantes que poseen un estilo de aprendizaje superficial se concentran principalmente en la memorización de definiciones y conceptos [29]. Este tipo de aprendizaje funciona bien para ciertas áreas que poseen un cuerpo de conocimiento amplio, por ejemplo la historia o la medicina.

En contraste, los estudiantes que tienen un estilo de profundidad se concentran en conseguir un entendimiento de los temas [28]. Este estilo es especialmente útil cuando es necesario realizar análisis, solución de problemas y generación de nuevo conocimiento.

Ambos estilos suelen ser necesarios para el aprendizaje. En la mayoría de las áreas teóricas, primero es necesario memorizar ciertos conocimientos base, para luego utilizar estas bases en conocimientos más profundos.

Esto sin embargo no aplica a la programación, donde contrario a pasar de un estilo al otro, ambos estilos son necesarios a la vez. No es útil aprender toda la sintaxis y las reglas de un lenguaje de programación de manera separada y luego estudiar la solución de problemas. Tener que aplicar ambos estilos al mismo tiempo no es algo con lo que la mayoría de estudiantes estén familiarizados [29].

2.2. **Contenidos Específicos**

Otro reto del aprendizaje de la programación está en la apropiación de los temas concretos. Diferentes estudios han demostrado cómo algunos de estos temas son más o menos complicados de aprender para un estudiante [9–11], [14]. Además, los contenidos específicos en el aprendizaje de la programación se deben al hecho que la mayoría de los conceptos necesarios para utilizar un lenguaje de programación con el fin de resolver un problema no tienen un referente en el mundo real [31].

2.3. **Motivación**

La motivación es un concepto abstracto y difícil de medir de manera objetiva, pero que tiene un impacto significativo en el proceso de aprendizaje [32]. Específicamente en el aprendizaje de la programación, tres de los factores principales que afectan la motivación de los estudiantes son:

- El estudiante espera poder programar cualquier tipo de software después de tomar una clase de programación, lo cual hace que se frustren sus expectativas al descubrir que no es así [33].
- El estudiante no encuentra una relación entre la profesión elegida y la programación [1].
- El estudiante cree que se necesita cierta habilidad innata para programar y si no la posee, no podrá aprender sin importar cuánto lo intente [29].

La motivación es generalmente categorizada según su fuente: la motivación extrínseca es aquella que surge a partir de un estímulo o una fuente externa. Esta hace que el estudiante quiera aprender para lograr un objetivo, ya sea lograr el éxito profesional, vencer a sus compañeros académicamente, o impresionar a un profesor [34]. La motivación intrínseca, por el contrario, proviene del interior del estudiante, y hace que este quiera aprender por la satisfacción de hacerlo [34].

Para lograr la apropiación de conocimientos, es ideal lograr que el estudiante se sienta motivado intrínsecamente. Se ha encontrado que aquellos estudiantes intrínsecamente motivados tienen mejor desempeño que aquellos cuyas motivaciones son principalmente extrínsecas [29].

Según la teoría de la autodeterminación, existen tres formas de lograr que una persona adquiera motivación intrínseca hacia una tarea [35]:

Competencia: La persona siente que posee habilidad para realizar la tarea requerida.

Autonomía: La persona siente que tiene la posibilidad de elegir qué y en qué orden aprende algo.

Relación: La persona puede utilizar los conocimientos adquiridos para relacionarse con otras personas.

3. Trabajos Previos en el Área

Se han realizado diferentes trabajos para tratar de llevar los juegos y *Gamification* al salón de clase. Lee y Hammer proponen que vale la pena explorar el uso de técnicas de *Gamification* en la educación como herramienta motivacional para incrementar la participación estudiantil en diferentes actividades que hacen parte de una asignatura [24].

Herramientas como *Khan Academy*, que mezclan conceptos como medallas y logros con la experiencia de aprendizaje, han demostrado cómo la inclusión de elementos de *Gamification* ha servido para resolver el problema principal de los cursos masivos en línea (MOOCs): la deserción de los estudiantes que toman los cursos [36].

También se han aplicado técnicas de *Gamification* a los entornos educativos tradicionales. Al reemplazar las notas con niveles de experiencia, o los créditos extra con puntos, se ha intentado resolver problemas tanto cognitivos como motivacionales. Estos intentos han tenido resultados mixtos [17], [22-24].

Una investigación realizada por Jayasinghe y Udeni descubrió que los estudiantes responden positivamente ante los elementos de juego combinados con sus clases tradicionales [37]. Otros intentos de 'gamificar' el contenido de un curso de ingeniería han resultado en mayor participación estudiantil y un incremento en la cantidad de estudiantes que toman el curso, mas no se determinó una correlación entre 'gamificar' el contenido y una mejora en el desempeño de los estudiantes [40].

Por otra parte, específicamente en el área de programación, un experimento realizado por Thomas y Berkling concluyó que no es suficiente añadir puntos y medallas a un curso para lograr el efecto de incrementar la motivación ya que para los estudiantes la clase deja de ser importante cuando se le agregan estas mecánicas sin un objetivo y un proceso de diseño bien definido [39].

4. **Frameworks de Diseño de Gamification**

Aunque utilizar *Gamification* sigue siendo una metodología relativamente nueva, han surgido diferentes frameworks que proponen una serie de pasos y herramientas para guiar la aplicación de mecánicas de *Gamification* dentro de un proceso. Entre estos se encuentran:

4.1. **Framework D6 [5]**

Este framework, elaborado por Kevin Werbach y Dan Hunter presenta seis pasos a seguir para crear un sistema 'gamificado'. Los primeros tres pasos están orientados a definir el contexto donde se va a implementar el sistema y a determinar si utilizar *Gamification* es una buena solución para el sistema. Los siguientes tres están orientados a describir el sistema 'gamificado' y a justificar cómo y mediante cuáles mecánicas de juegos el sistema va a mantener motivado al usuario. Los pasos del framework son:

4.1.1. **Definir los Objetivos de Negocio:**

En este paso se deben definir los objetivos del sistema. Estos objetivos deben resumir por qué se está 'gamificando' la actividad y qué se espera obtener al hacerlo. En esta sección se hace énfasis en explicar lo que se desea obtener del sistema y no en cómo se va a llevar a cabo.

4.1.2. **Delinear los Comportamientos Objetivo**

En este paso se deben describir las acciones que se desea los jugadores lleven a cabo al usar el sistema y las métricas que se usarán para determinar si estos comportamientos se están dando. Los comportamientos descritos deben buscar alcanzar los objetivos de negocio y las métricas deben proveer al jugador con retroalimentación de algún tipo para que él sepa si está llevando a cabo las acciones esperadas de su parte.

4.1.3. **Describir a los Jugadores**

Este paso busca que el diseñador del sistema entienda a sus usuarios (jugadores) potenciales para que en pasos posteriores pueda diseñar la mejor experiencia posible para ellos. En este paso el diseñador ya debe empezar a describir qué tipo de mecánicas de juegos va a utilizar en su sistema, haciendo énfasis en cuáles serían las más efectivas para los jugadores descritos.

4.1.4. **Determinar los Ciclos de Actividad**

En este paso se explora con más detalle cómo se va a motivar a los jugadores mediante ciclos que comprometan a los usuarios con el sistema y ciclos que permitan al jugador adquirir más experiencia de manera que perciba lo aprendido o logrado desde el momento cuando empezó a utilizar el sistema hasta el momento actual. Es apropiado describir el tipo de retroalimentación que recibirá el jugador, la forma como se hará que nuevos usuarios adopten el sistema y cómo se logrará que usuarios existentes sigan utilizándolo.

4.1.5. **¿Dónde se Encuentra la Diversión?**

Este es el paso más abstracto de todos. Asegurarse que el sistema 'gamificado' sea divertido de utilizar para el usuario es una de las claves de éxito para un proyecto de *gamification*. Aquí conviene explorar cómo funcionaría el sistema si no hiciera uso de recompensas extrínsecas (puntos, medallas, logros, entre otras) con el fin de evitar que el sistema caiga en la trampa de sólo utilizar ese tipo de mecánicas.

4.1.6. **Desplegar las Herramientas Adecuadas.**

En este paso se explican cuáles son las mecánicas de juego que utilizará el sistema, cómo será la experiencia del jugador al utilizarlo y cómo asegurarse que cumplirá los objetivos.

4.2. **Framework Octalysis** [41]

Este framework indica que todas las actividades realizadas en el proceso de 'gamificar' una actividad deben estar enfocadas hacia uno de ocho núcleos principales que motivan a los jugadores a seguir haciendo parte del juego.

4.2.1. **Significado Épico y Llamado a la Acción**

Sentir que ha sido elegido para cumplir con un rol importante y que lo que está haciendo tiene importancia.

4.2.2. **Desarrollo y Logro**

El jugador siente que está progresando, sobreponiéndose a los obstáculos y obteniendo habilidades.

4.2.3. **El Empoderamiento de la Creatividad y la Retroalimentación**

El poder ejercer la creatividad y recibir retroalimentación para reaccionar.

4.2.4. **Sentido de la posesión**

Sentirse dueño de algo motiva al jugador a tratar de mejorar su posesión y obtener más.

4.2.5. **Influencia Social**

La competencia, la cooperación, la aceptación y, en general, todos los aspectos sociales del juego.

4.2.6. **La Impaciencia y la escasez**

No poder obtener algo inmediatamente motiva al jugador a pensar en cómo obtenerlo.

4.2.7. **Curiosidad y Falta de Predictibilidad**

Cuando no se sabe lo que va a suceder a continuación, creando una sensación de curiosidad.

4.2.8. **La pérdida y Evitación**

Por naturaleza, el ser humano está motivado a evitar perder algo que es de su posesión.

4.3. **Framework MDA** [42]

Este Framework fue desarrollado para el diseño de juegos, pero también ha sido utilizado para la creación de sistemas '*gamificados*' [5]. Sus autores lo plantean como una aproximación formal al entendimiento de los juegos. El framework descompone el diseño de juegos en los tres componentes esenciales que le dan su nombre.

4.3.1. **Mecánicas**

Describen los componentes particulares del juego, al nivel de su representación de datos y algoritmos.

4.3.2. **Dinámicas:**

Describe el comportamiento de las mecánicas durante la ejecución del juego y su interacción con el jugador.

4.3.3. **Estéticas**

Describe las respuestas emocionales que se desea sean invocadas en el jugador cuando este interactúa con el juego/sistema *'gamificado'.*

III – MODELO PROPUESTO

Para el desarrollo de este modelo, se ha seleccionado el framework D6, ya que tiene en cuenta aspectos importantes tales como cuáles son los objetivos del sistema 'gamificado' y todos los demás pasos están alineados a cumplir esos objetivos.

1. Definir los Objetivos de Negocio:

- Borrar el mito de que la programación es una actividad especializada no útil para la vida diaria.
- Motivar a los estudiantes de cursos de programación a que estudien y practiquen los conocimientos adquiridos en las clases.
- Fomentar el autoaprendizaje para minimizar la dependencia de los conceptos transmitidos en el salón de clases.
- Incrementar el número de estudiantes que ven utilidad en adquirir habilidades de programación.

2. Delinear los Comportamientos Objetivo

Se espera que los jugadores utilicen este sistema para resolver más problemas de programación respecto a los resueltos durante el desarrollo de un curso universitario de programación. Esto lleva a que las métricas utilizadas para determinar si este comportamiento se está presentando tengan todas que ver con el número de problemas resueltos. Indicadores como el promedio de problemas resueltos cada día, el promedio de la cantidad de problemas resueltos respecto al número de usuarios registrados, entre otros, representan mayor éxito del sistema entre mayor sea su valor.

3. Describir a los Jugadores

Para este modelo de gamification, los jugadores serán descritos de dos formas: demográficamente y por la clasificación de jugadores descrita por Richard Bartle [43].

3.1. **Descripción Demográfica**

El modelo está dirigido principalmente a estudiantes universitarios de primeros semestres de ingeniería quienes, en su mayoría, se encuentran entre los 16 y los 19 años. Para el propósito de este modelo, estos estudiantes pueden ser divididos en dos grupos principales:

3.1.1. **Estudiantes de Ingeniería de sistemas y afines:**

Algunos tienen conocimientos básicos de programación cuando entran a la universidad, pero la mayoría no los tienen [44]. Al iniciar sus estudios universitarios, tienen ciertas ideas de lo que significa programar. Ellos son especialmente vulnerables a la desmotivación por expectativas frustradas al encontrarse haciendo aplicaciones que no tienen que ver con las expectativas que los medios de comunicación les han inculcado.

3.1.2. **Estudiantes de otras ingenierías:**

Estos estudiantes son especialmente vulnerables a la desmotivación porque tienen la idea que la programación no sirve para sus carreras [45].

3.2. **Descripción según tipo de jugador**

Utilizando la clasificación de jugadores de Richard Bartle [43] es posible separar los usuarios en cuatro grupos según lo que cada uno busca principalmente en los juegos. Esta clasificación es usada ampliamente para describir los potenciales usuarios de un proceso ‘gamificado’ [5].

3.2.1. **Perfeccionistas:**

Para los jugadores que quieran sobresalir y hacer todo lo que ofrece el sistema, habrá muchos problemas para mantenerlos ocupados. Logros, medallas y los múltiples caminos deberán servir para mantener enganchado a este tipo de jugador.

3.2.2. **Exploradores:**

Para los jugadores que disfrutan explorando la herramienta y encontrando lo que puede ofrecer, se pueden incluir logros ocultos. Hacer que algunos sean obvios y parezcan obtenidos al azar, como al enviar un problema correctamente al primer intento o por encontrar más de una solución al mismo problema puede hacer que este tipo de jugador pase más tiempo utilizando la herramienta buscándolos.

3.2.3. **Asesinos:**

Este tipo de jugador disfruta de la competencia y de saber que es el mejor. Los tableros de puntaje y los problemas difíciles que presenta el modelo sirven para mostrar su superioridad y son especialmente efectivos con ellos.

3.2.4. **Socializadores:**

Dar la posibilidad de compartir los logros y niveles alcanzados, así como comentar en los problemas o buscar una forma de compartir soluciones con otros usuarios que ya hayan solucionado un problema dado puede ser atractivo para los jugadores de este tipo.

4. **Determinar los Ciclos de Actividad**

4.1. **Ciclos de Progreso (Progression Loops)**

El progreso en el sistema está basado en la noción que el jugador puede elegir entre distintos “personajes”, que representan distintos caminos y temas de aprendizaje.

Cada personaje tiene un camino diferente con problemas diferentes que deben ser resueltos por el estudiante. Estos problemas tienen un contexto o “historia” con el objetivo de poner al jugador en el contexto de su personaje y de tratar de que se sienta identificado con él. Por ejemplo, en el camino de un personaje llamado “programador novato”, se podría esperar encontrar problemas introductorios de programación que debe resolver un estudiante para superar un curso de programación.

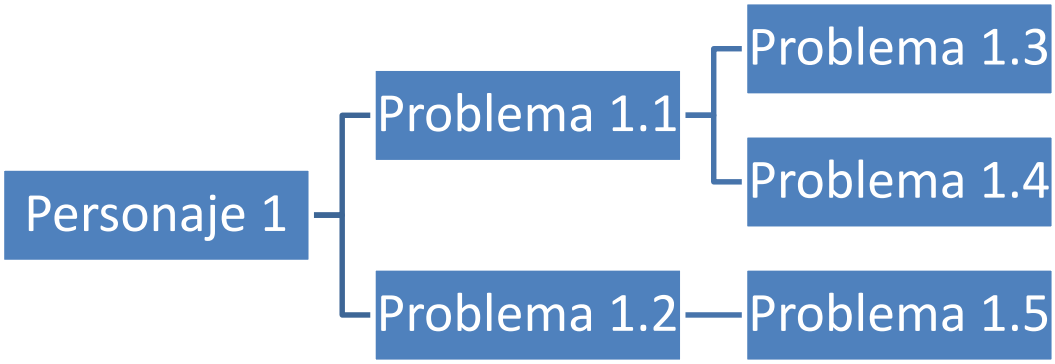
Al momento de iniciar un camino, solo los problemas más básicos de ese camino estarán desbloqueados. A medida que el jugador complete estos problemas básicos, se desbloquearán problemas de mayor dificultad.

Solo algunos de estos problemas son necesarios para completar el camino del personaje. Otros problemas funcionarán como retos bono que tendrán mayor dificultad al problema promedio del camino, pero entregarán recompensas adicionales.

Cuando el jugador completa un problema correctamente, recibe puntos de experiencia. A medida que el usuario acumula estos puntos y cuando llega a ciertos umbrales predeterminados, el jugador aumenta su nivel. Esto refuerza al jugador en el hecho que ha adquirido experiencia desde que ha empezado a utilizar el sistema.

La relación entre los personajes y los problemas se puede observar en el ejemplo descrito a continuación e ilustrado en la Figura 2. Para completar al personaje 1, el jugador debe resolver los problemas 1.1. – 1.5. Sin embargo, para poder intentar resolver los problemas 1.3 y 1.4, el jugador deberá haber completado el problema 1.1.

Figura 2 Personajes y Problemas



4.2. **Ciclo de Compromiso**

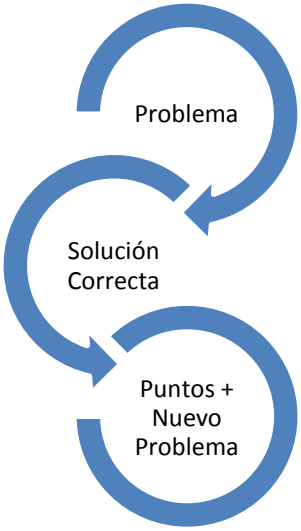
El sistema está basado en la solución de problemas de programación, así que el ciclo de actividad principal se centra en esto.

En cualquiera de los caminos disponibles, cuando el jugador decide solucionar un problema, se le presenta una introducción que contiene el enunciado del problema y la información de contexto de éste. En esta sección es importante que el diseñador del problema indique en qué situaciones de la vida real se pueden presentar este tipo de problemas. A continuación el jugador debe solucionar el problema y enviar la respuesta al sistema para que sea calificada. Si la respuesta que provee el jugador es incorrecta, se le informa al jugador que su solución no es la esperada y se le da la oportunidad de recibir una ayuda antes de volver a intentar resolverlo de nuevo.

En caso de que la respuesta sea correcta, se felicita al jugador por solucionar correctamente el problema, se le muestran algunas estadísticas relacionadas a este (número de personas que resolvieron el problema antes que él, qué tan rápido pudo escribir la solución al problema, qué tan rápido se ejecutó su programa, entre otras posibles) y se le presenta una explicación contextual detallada de cómo se utilizan los conceptos utilizados para resolver el problema en situaciones de la vida real.

Una vez solucionado el problema, se desbloquean nuevos problemas para que el usuario los resuelva, repitiendo el ciclo de planteamiento, solución y retroalimentación. Una representación gráfica de este tipo de ciclo se puede observar en la Figura 3.

Figura 3 Ciclo de Compromiso



5. ¿Dónde se Encuentra la Diversión?

Para determinar cómo el jugador encuentra la diversión dentro del sistema, se va a utilizar la clasificación propuesta por Nicole Lazzaro, quien plantea cuatro aspectos clave que debe tener un juego para hacer que sus jugadores sientan apego y se sientan comprometidos emocionalmente con este [46]. Es importante que un juego que quiera tener una audiencia diversa, explote más de uno de los aspectos. Este modelo utiliza tres de ellos para describir cómo el jugador se va a divertir utilizándolo.

5.1. Diversión Difícil

La diversión difícil se refiere a la diversión que surge cuando a una persona se le presenta un reto y disfruta resolviéndolo. Los retos hacen que los jugadores se enfoquen en la actividad que realizan, elijan una estrategia de solución y recompensan el progreso, haciendo que el jugador tenga una sensación de triunfo personal. Los retos deben ser balanceados mediante niveles de dificultad y progreso para mantener un buen nivel de frustración y recompensa.

Este aspecto es el que predomina en este modelo. Se espera que los jugadores se sientan comprometidos con solucionar los retos que se les presentan. Los problemas de programación son problemas concretos que muy a menudo tienen más de una estrategia de solución posible, con lo cual el jugador puede explotar su creatividad y su capacidad de solución de problemas, a la vez que se intenta convertir la frustración de encontrar un problema difícil en un reto que proporcione satisfacción cuando se logre solucionar.

5.2. Diversión Fácil

La diversión fácil es aquella que se obtiene al simplemente disfrutar las actividades del juego. Cuando se incluyen aspectos de diversión fácil, se facilita la inmersión y la curiosidad del jugador, a la vez que hace que este explore el juego y considere diferentes opciones y aproximaciones para jugar.

Aunque este aspecto no será explotado directamente puesto que este modelo no describe el funcionamiento de un juego sino de un sistema ‘gamificado’, se encontrará presente en

el hecho que el jugador puede explorar la herramienta, los diferentes personajes, las diferentes habilidades y otros aspectos libremente.

5.3. **El Factor Social**

Este aspecto se refiere al disfrute que obtienen los jugadores cuando interactúan con otras personas dentro o por fuera del juego. El factor social crea emociones y vínculos tan importantes que algunas personas incluso juegan juegos que no les gustan simplemente para pasar tiempo con sus amigo [46]. La rivalidad, el trabajo en equipo y los objetivos en común son estrategias de los juegos para explotar estas emociones.

El factor social está presente en este modelo mediante tres aspectos principales. Para explotar el altruismo y las sensaciones de bienestar generadas al ayudar a los demás, cuando un jugador logra solucionar un problema podrá escribir consejos para otros jugadores que no lo hayan resuelto aún. Adicionalmente, el jugador podrá compartir sus logros, los problemas resueltos y otros aspectos de su experiencia. Por último, diferentes jugadores pueden unirse para resolver retos de alta dificultad que requieran las habilidades de personas que hayan desbloqueado distintos tipos de jugadores.

6. **Desplegar las Herramientas Adecuadas.**

6.1. **Triada PBL (conocida por sus siglas en inglés: Points, Leaderboards, Badges)**

Aunque no es la mecánica central de este modelo, estos tres elementos siguen siendo útiles para recompensar a los jugadores cuando alcanzan un objetivo dentro del sistema. Los puntos de experiencia funcionan como un valor cuantitativo que permite al jugador saber qué tanto ha avanzado desde el punto cuando empezó a utilizar el sistema.

Los tableros de puntaje servirán para motivar a los jugadores más competitivos, mostrando por cada problema cuáles son los mejores tiempos de solución, quiénes son los usuarios con más problemas terminados, etc.

Las medallas servirán como recompensa por alcanzar algunos logros importantes, por ejemplo se podrán asignar medallas por resolver todos los problemas pertenecientes a un personaje.

6.2. **Personajes**

Como se mencionó en la sección de Determinar los Ciclos de Actividad, el jugador podrá elegir entre una variedad de personajes que representarán diferentes aproximaciones y usos de la programación. Estos personajes son la implementación de este modelo para permitir al jugador tener diferentes opciones de lo que desea aprender en un momento dado. Por ejemplo, si un estudiante quiere saber qué le puede ofrecer la programación para crear sitios web, puede resolver algunos problemas del personaje "Desarrollador Web". Si otro estudiante hasta ahora está aprendiendo a programar, puede empezar a utilizar el personaje "programador básico" y resolver problemas básicos de programación.

6.3. **Niveles**

Los niveles representan cuantitativamente la experiencia que ha adquirido un jugador utilizando la herramienta. Un jugador de nivel 5 ha resuelto más ejercicios, y por lo tanto ha practicado más en comparación con un jugador de nivel 3. Los jugadores obtienen puntos al resolver problemas. Entre más difícil es el problema, más puntos obtiene. Cuando el jugador obtiene un número predeterminado de puntos, sube de nivel. Este nivel sirve como refuerzo para el jugador para que sienta que va en buen camino para llegar a la maestría del sistema.

6.4. **Habilidades**

Cada vez que un jugador sube de nivel, tiene la opción de obtener una habilidad que lo ayudará a mejorar su rendimiento al resolver problemas. Estas habilidades permiten que el jugador sienta que puede personalizar su experiencia en el sistema y que tiene libertad de elegir aquello que le será más beneficioso y sirven como otro recordatorio del camino que ha recorrido desde que comenzó como un usuario nuevo del sistema.

6.5. **Retos**

En el corazón del sistema están los retos o problemas que se le presentan al jugador. Cada personaje tiene sus propios retos independientes de los de otro personaje. Como se describe en los ciclos de actividad, al aceptar un reto el jugador recibe un enunciado que le presenta un contexto y lo que se espera que resuelva. A continuación debe codificar su solución y enviarla al sistema para ser calificada. El jugador recibe retroalimentación sobre su solución: en caso de ser incorrecta se le dan consejos para solucionar el problema y se le pide que intente de nuevo. En caso de solucionar correctamente el problema, se le informa que ha superado el reto, se otorgan los puntos de experiencia correspondientes y se le otorgan las medallas correspondientes en caso que el jugador se haya hecho merecedor de alguna.

Los retos podrán ser diseñados para ser resueltos por un solo jugador, pero también pueden existir retos en los que se busca la participación de varios jugadores para resolver problemas muy complejos para una sola persona. Como los retos de programación de la vida real, estos problemas grupales requerirían la participación de jugadores de diferentes niveles y que hayan desbloqueado diferentes personajes para ser resueltos.

La Figura 4 muestra una representación gráfica de las entradas, los fundamentos teóricos, las herramientas utilizadas y las salidas esperadas del modelo.

Figura 4 Representación Gráfica del Modelo

IV – METODOLOGÍA PEDAGÓGICA

A continuación se presentan los fundamentos teóricos, desde la óptica de la pedagogía, de por qué el modelo puede ayudar a solucionar los problemas de aprendizaje mientras que brinda métodos para asegurar que se realice una apropiación profunda y duradera de conocimientos para quienes lo utilicen.

1. Metodología Constructivista

El constructivismo es una corriente de la filosofía de la educación. Sostiene que las habilidades y el conocimiento se construyen a partir de la exploración activa que la persona hace de su entorno [47]. Además, indica que la mejor forma para aprender es mediante problemas y enigmas atractivos, que despierten un sentido de propósito en quienes los solucionan [31, 32].

Estas afirmaciones planteadas por el constructivismo se pueden mapear directamente a elementos del modelo. Se busca fomentar el aprendizaje de los jugadores mediante la práctica y la solución de problemas, a la vez que intenta personalizar su experiencia educativa al permitírseles la libre exploración de la herramienta y la selección de los temas a aprender mediante el sistema de personajes. Por último, se intenta dar un sentido de propósito a los jugadores al ofrecer un contexto para cada uno de los personajes y de los problemas que se presentan en los caminos de estos.

2. Aprendizaje para la comprensión.

El psicólogo y pedagogo Howard Gardner propone que la educación debería enfocarse menos hacia la memorización de conceptos y de fórmulas, y más hacia la comprensión de los temas por parte del estudiante. Para lograr este objetivo, plantea cuatro propuestas para estimular la comprensión [49]. Este modelo posee elementos que implementan cada una de estas propuestas:

2.1. **Aprender de las instituciones adecuadas**

El aprendizaje no solo se lleva a cabo en el salón de clases sino mediante la experiencia y la aplicación práctica de los conocimientos, por ejemplo, los museos de ciencia interactivos. Este modelo describe un sistema complementario al salón de clases donde el estudiante puede resolver problemas prácticos de programación con un objetivo definido.

2.2. **Afrontar directamente las concepciones erróneas**

Para la mayoría de las personas, el hecho de que una creencia arraigada sea puesta en duda como mínimo llama su atención, y puede llevar a mejorar la comprensión al tratar de descubrir una creencia mejor.

Es un objetivo directo del modelo lograr que los estudiantes rompan con los preconceptos sobre la utilidad de la programación y provee la forma para que se encuentren nuevas creencias y para que el estudiante sienta que obtiene lo que desea de la programación mediante el sistema de personajes.

2.3. **Múltiples vías de acceso a la comprensión**

Las personas poseen distintos tipos de mentes, las cuales trabajan con diferentes combinaciones de representaciones mentales. Esto causa que cada persona aborde y llegue a dominar los conocimientos y materiales curriculares de una manera personal. Esto significa que para que la mayoría de los estudiantes mejoren su nivel de comprensión, debería haber varias formas de acceder a ella.

Mediante el uso de los personajes y las habilidades, los estudiantes pueden personalizar su experiencia educativa para lograr encontrar la mejor forma de aprender.

2.4. **Un marco de referencia que facilite la comprensión**

Este marco de referencia fue diseñado por Gardner y otros profesores de la universidad de Harvard como una propuesta para colocar la comprensión en un primer plano en el sistema educativo. Ofrece una metodología pedagógica concreta que se puede aplicar a estudiantes

de diferentes edades y con diferentes estilos de aprendizaje. La metodología consta de 4 componentes, todos compatibles con el modelo propuesto:

2.4.1. **Establecer Objetivos de Comprensión**

Se deben establecer objetivos claros que permitan medir el avance en la comprensión. El sistema de personajes busca dar modularidad a la comprensión de la programación de tal forma que cada personaje tenga un objetivo concreto de enseñanza.

2.4.2. **Establecer Temas Generativos o Cuestiones Esenciales**

Se deben plantear temas esenciales para el aprendizaje del área que se estudia. Estos temas además de esenciales deben ser relevantes y atractivos para los estudiantes.

El sistema de personajes y retos permite al estudiante seleccionar los temas que desea estudiar y en qué orden. Cada personaje busca enseñar una rama distinta de la programación para lograr eventualmente la maestría del estudiante en el área.

2.4.3. **Establecer los Ejercicios de Comprensión**

Los estudiantes deben estar conscientes de qué es lo que tienen que saber y sobre lo que tienen que hacer. A su vez, ellos deben tener claros los criterios que se usarán para evaluarlos.

Los problemas de programación se prestan mucho para este tipo de metodología ya que aunque por lo general pueden ser solucionados de varias formas, la respuesta que arrojan debe ser exacta y funcionar en varios casos de prueba para ser aceptada como correcta. Así, al leer un ejercicio, el estudiante tiene claro lo que debe hacer para resolverlo.

2.4.4. **Evaluación Continua**

El estudiante debe ser evaluado y debe recibir retroalimentación de su desempeño de manera continua. Esto permite que con el tiempo sea capaz de evaluar su desempeño sin necesidad de la intervención de terceros.

Cuando los estudiantes envían la solución a un reto, reciben retroalimentación inmediata sobre su desempeño, informándoles si su solución fue correcta o no, y en caso de que no lo sea, informa cuáles fueron los errores presentados.

3. **Principios de Aprendizaje en los Juegos** [50]

Los principios de aprendizaje en los juegos según James Gee [50] son un conjunto de trece principios de diseño de videojuegos que, además de crear una mejor experiencia en el juego, ayudan al jugador a aprender las mecánicas de este. Gee sostiene que estos principios, al ser aplicados en un contexto educativo son una herramienta para lograr que los juegos sean un mecanismo válido de aprendizaje. En esta sección además se describe cómo el modelo propuesto implementa estos principios para asegurar que la experiencia del jugador ayude al aprendizaje.

Los trece principios están divididos en tres categorías según su finalidad:

3.1. **Darle poder a los jugadores**

Estos principios son utilizados para que los jugadores sientan que están en control del juego y de su experiencia de aprendizaje.

3.2. **Principios de Solución de Problemas**

Estos principios son utilizados para que el jugador mejore sus capacidades de solución de problemas.

3.3. **Aprendizaje Profundo**

Estos principios son utilizados para lograr que el entendimiento que logra el jugador sea profundo y duradero.

Las tablas 1, 2 y 3 presentan los 13 principios y cómo se encuentran reflejados en el modelo para demostrar que aunque la experiencia esté diseñada con elementos de juegos, proporciona mecanismos de aprendizaje para las personas que lo utilicen.

Tabla 1. Principios para darle poder a los jugadores

Principio	Descripción	Implementación en el modelo
Co-Diseño/Agente	El estudiante debe creer que las decisiones que toma en el juego tienen importancia.	El modelo plantea la contextualización de los problemas que deben resolver los jugadores con el objetivo de que sientan que lo que están haciendo tiene importancia dentro y fuera del sistema.
Personalización	El contenido del juego debe permitir al jugador usar diferentes estilos de juego para triunfar.	Este principio se encuentra aplicado tanto en los personajes como en las habilidades que puede elegir el jugador.
Identidad	Crear metas y roles definidos y claros, dando una o varias identidades al jugador.	Los jugadores podrán elegir entre diferentes personajes con sus propios enfoques y problemas. Adicionalmente, los retos que deben ser resueltos por el jugador están claramente establecidos dentro de cada personaje.
Manipulación	Permitir al jugador manipular el juego para facilitar la inmersión. Ofrecer herramientas para involucrar su mente y cuerpo en el juego.	Dado que el modelo describe un sistema 'gamificado' y no un juego como tal, este principio no se ve aplicado directamente. Sin embargo, si un jugador se siente retado por un problema, puede verse inmerso en él.

Tabla 2. Principios para fomentar la solución de Problemas

Principio	Descripción	Implementación en el modelo
Problemas Bien Ordenados	El estudiante debe creer que las decisiones que toma en el juego tienen importancia.	Los problemas iniciales de cada personaje empiezan siendo fáciles y van aumentando en dificultad. El jugador puede ver claramente el orden en que deben resolver los problemas.
Agradablemente Frustrantes	Los problemas deben representar retos y no castigar los errores, a la vez que deben permitir ser resueltos eventualmente después de retar la mente.	Como este sistema se plantea como una herramienta adicional a un curso de programación y no como parte de este, el jugador no siente miedo de equivocarse al resolver un problema. Si el jugador comete un error al resolver un problema, podrá intentarlo de nuevo sin recibir penalizaciones. Por último, es responsabilidad de quien cree los retos para cada jugador hacer que los problemas sean difíciles para él, pero que tengan una solución clara.
Ciclos de Habilidad	Los problemas deben crear ciclos de reto, práctica, conocimiento, maestría.	El subsistema de retos del sistema se encarga de que el jugador empiece su camino a la maestría resolviendo problemas que se vuelven cada vez más complejos, profundizando sus conocimientos cada vez que resuelve un problema.
Información bajo demanda y cuando se necesita	La información verbal es más valiosa cuando se da justo cuando se necesita y/o cuando el jugador la solicita.	El jugador puede acceder a su información de perfil para recibir información sobre su progreso. Se ofrece retroalimentación al jugador cuando este resuelve uno de los retos.

Pecera	Empezar con poca complejidad e irla aumentando a medida que se superan niveles.	El sistema de retos permite al diseñador crear los caminos de sus personajes de tal forma que los retos iniciales sean sencillos de resolver y aumenten en complejidad a medida que el jugador adquiere experiencia. A medida que obtiene experiencia, sube de nivel, y completa retos, otras mecánicas son desbloqueadas.
Cajas de Arena	El jugador debe poder explorar dentro del juego sin ser castigado fuertemente por hacerlo.	El sistema permite al jugador cambiar de personaje, cambiar sus habilidades, y tratar de resolver los retos tantas veces como lo desee.
Habilidades como Estrategias	Las habilidades adquiridas tienen un significado en el contexto del juego y se usan como estrategia para resolver problemas o cruzar metas.	Las habilidades adquiridas tienen un significado en el contexto del juego y se usan como estrategia para resolver problemas o cruzar metas.

Tabla 3 Principios para lograr un aprendizaje Profundo

Principio	Descripción	Implementación en el modelo
Pensamiento Sistémico	Hacer que el jugador logre construir un modelo mental del problema.	Cuando el estudiante está tratando de resolver un problema, este debe crear un modelo mental del mundo donde se desarrolla el problema. Al contextualizar la importancia de resolverlo, el estudiante traslada ese modelo mental al mundo real.
Significado como Acción	El jugador aprende mediante acciones que realiza en el juego y no mediante definiciones o explicaciones que se le dan.	El jugador aprende lo que debe hacer utilizando el sistema y adquiriendo experiencia resolviendo los retos planteados por este.

V – PROTOTIPO DE PLATAFORMA

Uno de los aspectos importantes al planear la *gamificación* de un proceso mediante el uso de un sistema de software es demostrar que su implementación es técnicamente factible [5]. Para cumplir este requisito, se creó un prototipo de una plataforma web que implementa algunos de los aspectos principales del modelo de tal forma que pueda ser extendido en un futuro para implementar los elementos restantes e incluso agregar nuevas características a medida que se haga necesario.

En esta sección se presentan el proceso de selección de las herramientas tecnológicas que se utilizaron para el desarrollo, el análisis realizado para el diseño de la plataforma, así como la selección de características que fueron implementadas en el prototipo.

1. **Herramientas Tecnológicas**

Para decidir las herramientas tecnológicas que se usarían para desarrollar el prototipo, se realizó un listado de las características principales que debía poseer la plataforma cuando fuera implementada en su totalidad. Se buscó utilizar herramientas extensibles que incluyeran características de *gamification* como base para la plataforma, de tal modo que el desarrollo se pudiera enfocar en los aspectos propios de esta, y no en los aspectos comunes de la mayoría de sistemas *gamificados*. A continuación se presentan las herramientas investigadas y los criterios utilizados para elegir una de ellas como base para el desarrollo del prototipo.

1.1. **Criterios de Selección**

1.1.1. **Plataforma**

Debido a la necesidad de atender múltiples estudiantes de manera concurrente, y de mantener el sistema permanentemente actualizado con nuevo contenido, se decidió que la mejor opción era implementar el sistema bajo una arquitectura cliente-servidor, específicamente en forma de una aplicación web. De esta forma, solamente el servidor web

debe ser actualizado periódicamente y los clientes pueden acceder al sistema mediante cualquiera de los diferentes navegadores web.

1.1.2. **Licencia y Costo**

Ya que ninguna de las herramientas investigadas implementaba las características necesarias en su totalidad, uno de los criterios de selección para la herramienta base a utilizar fue determinar si era posible agregarle nuevas funcionalidades. Aquellas herramientas publicadas bajo licencias libres permiten examinar y modificar su código fuente por lo que pueden ser extendidas indefinidamente. En contraste, no está permitido modificar el código de herramientas con licencia propietaria, por lo que su extensibilidad depende de otros mecanismos, como proveer un servicio web o una interfaz de programación de aplicaciones (API, por sus siglas en inglés). Adicionalmente se examinó el costo de utilizar la herramienta para determinar si era posible su uso teniendo en cuenta las limitaciones del proyecto.

1.1.3. **Modificabilidad y Existencia de API**

A continuación se examinó qué tan sencillo era extender nuevos elementos a la herramienta para determinar si era factible utilizarla como base para la plataforma. Adicionalmente se investigó si la aplicación proveía un API con el cual fuera posible comunicarse desde los otros componentes de la plataforma.

Bajo este criterio, las herramientas de software libre demostraron una clara ventaja, ya que debido a su propia naturaleza, permiten la inclusión de nuevas funcionalidades y la interoperabilidad con otras plataformas.

1.1.4. **Soporte para Logros, Puntos, Medallas y Tableros (PBL)**

Los elementos PBL siguen siendo una de las estrategias más comunes en los sistemas *gamificados*. Aunque no es el punto principal de la plataforma, estos elementos proveen a los usuarios con recompensas de naturaleza extrínseca que pueden ayudar a mantenerlos enganchados para usar los aspectos principales.

1.1.5. **Disponibilidad de Documentación**

Se evaluó qué tan sencillo era encontrar documentación sobre la herramienta con el objetivo de minimizar la curva de aprendizaje.

1.1.6. **Componente Social**

Una de las características de la plataforma es la posibilidad de que los usuarios se comuniquen entre sí para resolver los diferentes problemas propuestos. En las herramientas investigadas, se determinó si es posible facilitar la comunicación entre usuarios con el fin de implementar las características sociales de la plataforma

1.1.7. **Estadísticas**

Con el fin de determinar si los usuarios están llevando a cabo los comportamientos objetivo, uno de los puntos considerados para determinar la utilidad de las herramientas disponibles fue la existencia de un módulo de estadísticas que permitiera obtener datos sobre el uso de la plataforma que pudieran llegar a determinar cuáles objetivos de negocio se están cumpliendo mediante los comportamientos delineados.

1.1.8. **Retos**

Ya que los retos son una parte importante del modelo ya que son la mecánica principal mediante la cual se desea controlar el progreso y el compromiso de los jugadores, se consideró cuáles herramientas poseían alguna característica que facilitara su implementación.

1.1.9. **Mecánicas Extra**

Por último se examinaron las mecánicas extra que proveían las herramientas investigadas, con el objetivo de documentar qué tipo de extensiones permitirían realizar a la plataforma en caso de ser elegidas. Las herramientas comerciales investigadas demostraron poseer ventajas en este aspecto, teniendo un gran número de facilidades adicionales incluidas,

mientras que las herramientas de software libre invitaban a los usuarios a que ellos mismos extendieran la aplicación.

1.2. **Herramientas Investigadas**

Se investigaron siete herramientas que proveen elementos de Gamification en plataformas web.

1.2.1. **ITPrism Gamification Platform**[51]

Es una extensión gratuita para el Sistema Gestor de Contenido Joomla[52], desarrollada por la compañía ITPrism. La plataforma provee algunos de los elementos más populares para *gamification* como puntos, medallas, niveles y recompensas. Se le pueden agregar elementos y funcionalidades adicionales. Como todas las extensiones de Joomla, es una herramienta de software libre.

1.2.2. **USERINFUSER**[53]

Una herramienta desarrollada por AppScale systems [54]. Son una serie de widgets que se pueden agregar a un sitio web existente. Incluye elementos como medallas, puntos, notificaciones en vivo y tableros de puntaje.

1.2.3. **BadgeOS**[55]

Es un plugin gratuito y libre para el Gestor de Contenidos WordPress[56]. Permite a los usuarios de un sitio web completar tareas, demostrar logros y obtener medallas. Incluye un motor para personalizar los pasos requeridos para obtener un logro. Es extensible mediante su sistema de Add-Ons y es compatible con la mayoría de los temas y demás Plugins de WordPress.

1.2.4. **Open Glaze**[57]

El proyecto Open Glaze es un conjunto de herramientas libres y gratuitas enfocadas en agregar una capa de elementos de juegos a cualquier sitio web. Las herramientas pueden

ser utilizadas en conjunto o independientemente, más no incluyen los elementos más populares para realizar 'gamification'. No ha visto mayores avances desde que la iniciativa fue comenzada en el año 2011.

1.2.5. **Herramientas Comerciales de Gamification**

Las últimas herramientas investigadas son representativas de una tendencia que ha empezado a surgir en el ámbito de Gamification: son herramientas comerciales ofrecidas por compañías que prometen incrementar el tráfico de cualquier sitio web por una cuota mensual. Aunque existe un gran número de estas herramientas, todas parecen compartir las mismas características. Para este proyecto, se extrajeron las características de las siguientes herramientas de este tipo: Behavior Platform de BadgeVille [58], la plataforma Nitro de Bunchball [59], y el Engage Engine de IActionable [60].

1.3. **Análisis de Herramientas**

Una vez determinados los criterios, se procedió a realizar el análisis de las herramientas elegidas. Los resultados del análisis se encuentran en las tablas 4 - 6, y una versión completa en el anexo 2 . Un recuadro de color verde indica que la herramienta cumple favorablemente con la característica, el color amarillo indica que la cumple parcialmente, y el color rojo indica que no la cumple.

Teniendo en cuenta los resultados de la comparación, se eligió utilizar el plugin BadgeOS como base para el prototipo. Se eligió utilizar esta herramienta ya que es gratuita y libre, lo cual hace posible que sea modificada, posee la mayoría de elementos necesarios para desarrollar el prototipo, cuenta con un sistema interesante de gestión de pasos para obtener los logros y cuenta con documentación y soporte suficiente para ayudar al desarrollo. También se tuvo en cuenta que WordPress, la plataforma donde se ejecuta el plugin, es un gestor de contenidos para páginas web que implementa facilidades tales como el registro y gestión de usuarios y facilita la creación de contenido, disminuyendo el tiempo de desarrollo y mantenimiento de características que hacen parte de todas las aplicaciones web, más no son propias de la plataforma.

Tabla 4 Comparación de Herramientas I

Herramienta	Plataforma	Licencia	Costo	Puntos	Medallas	Tableros	Componente Social
ITPrism.com Gamification Platform	Joomla	GPL	Gratis	Sí	Sí	Mediante Extensión	Mediante Extensión
USERINFUSER	AppScale	GPL	Gratis	Sí	Sí	Sí	No
BadgeOS	WordPress	GPL	Gratis	Sï	Sí	Mediante Plugin adicional premium (tiene un costo de $50)	Conectado con el plugin BuddyPress
Open Glaze	Varias	GPL	Gratis	No	No	No	Foros, Publicaciones en redes sociales
BadgeVille Behavior Platform	Web	Propietaria	$3000-$5000 al mes	Sí	Sí	Sí	Integración a redes sociales
Bunchball Nitro	Web	Propietaria	$2000-$4000 al mes	Sí	Sí	Sí	Integración a redes sociales
IActionable Engage Engine	Web	Propietaria	Depende del tráfico del sitio	Sí	Sí	Sí	Integración a redes sociales

Tabla 5 Comparación de Herramientas II

Herramienta	API Disponible	Modificabilidad	Estadísticas	Niveles	Retos
ITPrism.com Gamification Platform	Sí	Open Source, extensiones Joomla	Mediante Extensión	Sí	No
USERINFUSER	No	Open Source, Google Apps	Sí	Sí	No
BadgeOS	Sí	Open Source, plugins WordPress	Mediante Extensión	Sí	Implementado mediante búsquedas (ir de un link a otro, obtener cierto número de medallas)
Open Glaze	No	No	Sí	No	No
BadgeVille Behavior Platform	No	Desarrollo a la medida	Sí	Sí	Implementado mediante búsquedas (ir de un link a otro, obtener cierto número de medallas)
Bunchball Nitro	No	Desarrollo a la medida	Sí	Sí	Sí
IActionable Engage Engine	Disponible para clientes	Desarrollo a la medida	Sí	Sí	No

Tabla 6 Comparación de Herramientas III

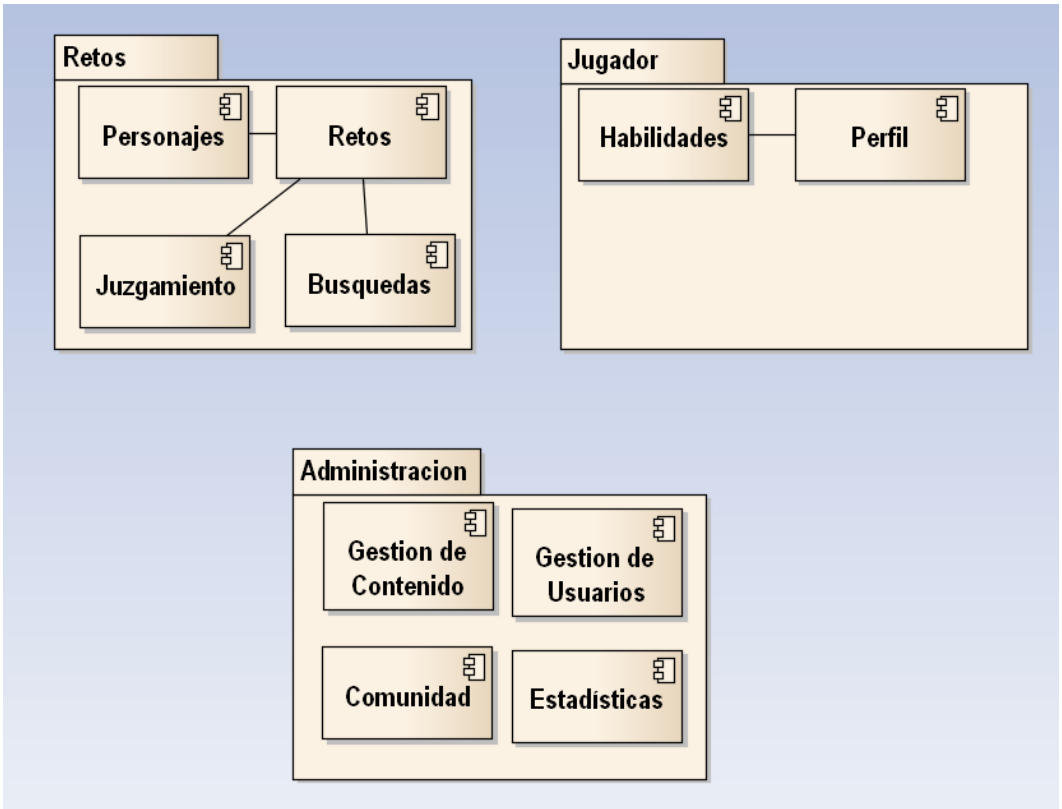
Herramienta	Mecánicas Extra	Observaciones
ITPrism.com Gamification Platform	Rangos, Notificaciones, Actividades, Perfiles	Se ajusta a cualquier template de Joomla
USERINFUSER	Caja de trofeos, Notificaciones	Funciona en Appscale y otros proveedores de servicios en la nube
BadgeOS	Manejo de comunidad, Perfiles	Se ajusta a la mayoría de Temas de WordPress
Open Glaze	No incluye elementos de PBL a propósito. Es un conjunto de herramientas pequeñas que pueden funcionar entre sí como una capa de elementos de juego sobre redes sociales	Está en desarrollo y aunque promete incluir más mecánicas de juego en el futuro, en este momento no ofrece muchas posibilidades
BadgeVille Behavior Platform	Búsquedas, recompensas, notificaciones en tiempo real	Estas tres herramientas son plataformas comerciales de gamification que son desarrolladas para aumentar la lealtad de los visitantes de las páginas web de las empresas que los contratan. Cobran una cuota mensual dependiendo del tráfico que se haya obtenido en la página. Son las herramientas más sencillas de utilizar y con más mecánicas, pero su costo las hace prohibitivas para el contexto de este proyecto.
Bunchball Nitro	Perfiles, notificaciones, newsfeeds	
IActionable Engage Engine	Metas, notificaciones	

2. Componentes de la Plataforma

Para el diseño de la plataforma, se determinaron los componentes necesarios para implementar los ciclos de progreso y compromiso del modelo. Adicionalmente, se determinó que debería haber algunos componentes adicionales para el manejo de la plataforma.

La siguiente figura muestra un diagrama con los componentes que fueron detectados como necesarios para el desarrollo de la plataforma.

Figura 5 Componentes de la plataforma



2.1. **Módulo de Retos**

Este módulo agrupa todas las funcionalidades que tienen que ver con el ciclo de compromiso del modelo. Maneja la creación de personajes y retos, así como los componentes necesarios para saber si un jugador ha resuelto un problema y por lo tanto ha completado el camino de un jugador.

2.1.1. **Personajes**

Este componente maneja la creación de los diferentes personajes que se agregan a la plataforma, así como de los retos necesarios para progresar en el camino de cada personaje.

2.1.2. **Retos**

Contiene la información de los retos tal como a cuál personaje pertenecen, el enunciado del reto, la información del contexto, y los datos para que sea posible juzgar un envío de un usuario.

2.1.3. **Juzgamiento**

Es el encargado de determinar si la solución que envía un jugador para solucionar un reto es correcta o no.

2.1.4. **Búsquedas**

Las búsquedas son logros transversales que no están necesariamente relacionados con un personaje o con un reto. Por ejemplo se puede crear un logro que se obtenga por registrarse en el sistema, o por resolver tres problemas del personaje A y cinco del personaje B.

2.2. **Módulo de Jugador**

Como su nombre lo indica, este módulo contiene todos los componentes que gestionan a los jugadores. Contiene la información de perfil del jugador y las habilidades que ha elegido al subir de nivel.

2.2.1. **Perfil**

Contiene la información de perfil de un jugador, tal como su nombre de usuario, su avatar, los puntos, niveles, medallas y logros que ha obtenido.

2.2.2. **Habilidades**

Maneja las habilidades que un jugador ha obtenido al subir de nivel y se encarga de aplicar los bonos correspondientes a cada habilidad.

2.3. **Módulo de Administración**

Este módulo controla los elementos adicionales de la plataforma como la creación de contenido adicional a los retos, la gestión de usuarios, y el manejo de los aspectos sociales.

2.3.1. **Gestión de Contenido**

Se encarga de manejar todo el contenido que no tenga que ver con los retos o jugadores, tal como noticias y contenido estático de la página.

2.3.2. **Gestión de Usuarios**

Maneja todos los procesos logísticos de registro y autenticación de usuarios en la plataforma.

2.3.3. **Comunidad**

Este componente contiene los aspectos sociales de la plataforma. Controla los comentarios que los usuarios realizan sobre los retos, los tableros de puntaje y el motor que controla la formación de equipos para resolver retos grupales.

2.3.4. **Estadísticas**

Permite recolectar y visualizar estadísticas de uso del sistema que permitan dar cuenta de los objetivos propuestos y de las métricas determinadas para evaluar el éxito del modelo.

3. **Elementos Implementados en el Prototipo**

Para el desarrollo del prototipo, se eligieron aquellos aspectos de la plataforma que representaban los principales retos técnicos para la implementación del modelo, así como las funcionalidades principales que demuestran la posibilidad de un desarrollo completo del sistema.

3.1. **Módulo de Administración**

Este componente permite al administrador manejar la edición de personajes, retos y habilidades. A su vez, permite ver estadísticas sobre el uso del sistema como cuáles usuarios han obtenido qué logros o los personajes que han desbloqueado.

La sección de administración de logros, medallas, personajes y problemas es implementada mediante las facilidades de administración que provee la instalación estándar de WordPress para el manejo de sitios y mediante la sección de administración de BadgeOS.

3.2. **Perfil del Jugador**

El perfil del jugador es implementado mediante la administración de usuarios de WordPress, combinado con el plugin BuddyPress [61] y la extensión de comunidad de BadgeOS, que permite agregar información sobre los puntos y logros obtenidos al perfil de usuario.

3.3. **Personajes**

En BadgeOS, la unidad de trabajo principal es el logro (achievement). De esta unidad se desprenden todas las demás. Se pueden crear tipos personalizados de logro, asignarle una medalla, así como modificar los pasos necesarios para que un usuario se haga merecedor a un logro. En el prototipo, los personajes, los retos y las búsquedas son un tipo personalizado de logro.

Los personajes se completan cuando el jugador ha obtenido los logros correspondientes a los retos que hacen parte del jugador. Debido a la flexibilidad que provee BadgeOS, un jugador puede tener como requisito obtener otro logro de tipo jugador, o incluso obtener otros tipos de logro.

3.4. **Retos**

Los logros están implementados de dos formas en el prototipo. Por un lado, los problemas son un tipo personalizado de página de WordPress (custom post type en el lenguaje de la

plataforma). Eso permite agregarle información al reto tal como un enunciado, la información de contexto, y las entradas y salidas que debe recibir un programa que solucione el problema.

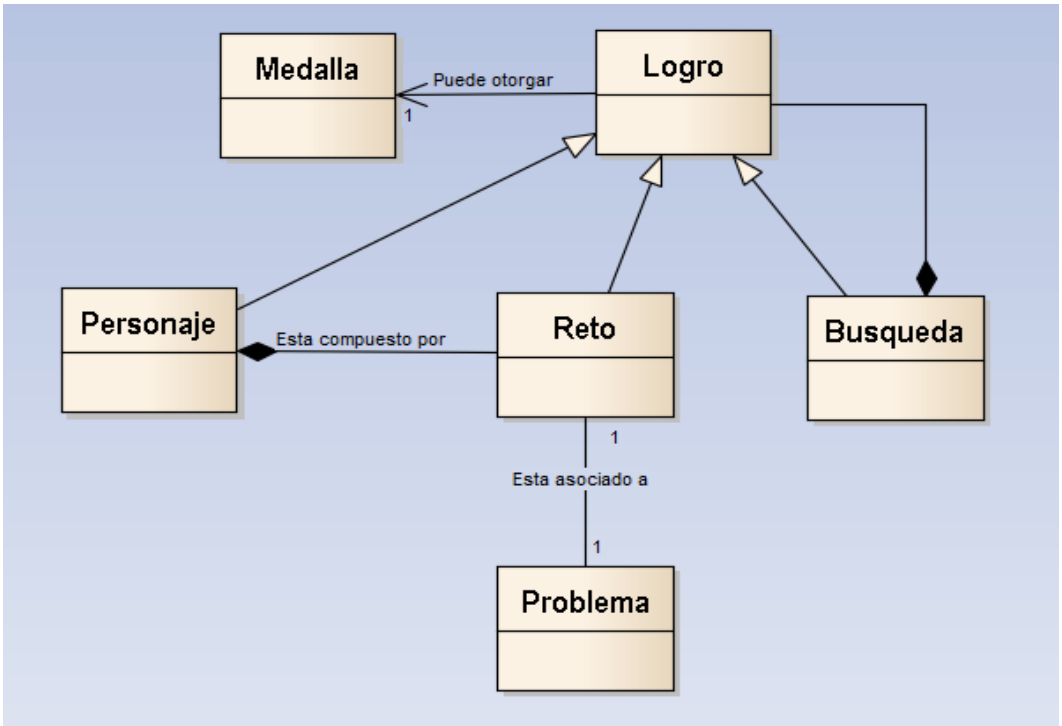
La segunda parte es un tipo personalizado de logro de BadgeOS llamado reto. Cuando el administrador crea un problema, se crea su reto correspondiente automáticamente. Cuando el usuario soluciona el problema, le es otorgado el logro de forma automática.

3.5. **Búsquedas**

Las búsquedas son un tercer tipo de logro personalizado. Este tipo de logro aprovecha el poder del motor de pasos de BadgeOS junto con el sistema de logros para permitir otorgarlos cuando el usuario realiza todo tipo de acciones, desde crear comentarios, resolver retos, desbloquear personajes, o registrarse en el sistema.

La figura 5 explica la interacción entre logros, personajes, retos, búsquedas, problemas y medallas.

Figura 6 Logros, Retos, Personajes y Búsquedas



3.6. **Juzgamiento**

Para determinar si la solución que envía el usuario para resolver un problema es correcta o no, era necesario desarrollar un componente que compilara y ejecutara el código enviado, y que fuera capaz de juzgar el envío. Para efectos del prototipo, el juzgamiento se realiza ejecutando el código que envía el usuario para revisión. Al programa resultante de la compilación se le ingresan los datos de entrada que han sido programados dentro del reto y finalmente se comparan las salidas obtenidas con las esperadas. Para realizar la compilación y ejecución, se decidió utilizar el Sphere Engine de ideone.com [62]. Este es un servicio web que expone el compilador en línea ideone y que permite compilar y ejecutar código de forma remota, así como recibir los resultados de la solicitud. La comparación de las salidas obtenidas por el programa del usuario y la salida esperada se realizan caracter por caracter, excluyendo los espacios y líneas en blanco al final de la salida. Así solamente los programas cuya salida concuerde exactamente con la salida esperada recibirán crédito.

VI – VALIDACIÓN DEL MODELO

Ya que la efectividad de la totalidad del modelo no puede ser comprobada sin ser implementada en un salón de clases como complemento al desarrollo de un curso de programación, y debido a que este tipo de prueba se encuentra fuera del alcance de este trabajo, se decidió realizar la validación del modelo bajo los méritos de su consistencia interna, su fundamentación teórica y su sensatez como herramienta educativa. Si los resultados eran positivos, se podría determinar que vale la pena continuar con la implementación del modelo en su totalidad y realizar las pruebas con estudiantes.

1. Metodología de Validación

Para realizar la validación, se decidió utilizar la metodología de la triangulación [63]. En esta metodología, se elige un número de expertos capacitados (preferiblemente un número impar) para evaluar los méritos del modelo propuesto lo examinan y dan un veredicto acerca de diferentes aspectos. Para este trabajo, se solicitó la colaboración de tres expertos con diferentes perfiles y conocimientos en las áreas de pedagogía y programación.

A estos expertos se les presentó un artículo con los detalles del modelo y las bases teóricas sobre las cuales se fundamenta, y se les solicitó diligenciar un formulario con los aspectos que debían evaluar. Una vez obtenidos los resultados, se analizaron con el objetivo de determinar los puntos fuertes del modelo, los aspectos por mejorar, y así llegar a una conclusión acerca de los puntos expuestos con anterioridad.

Adicionalmente, el artículo fue sometido al XXII Congreso Iberoamericano de Educación Superior en Computación en el marco de la XI Conferencia Latinoamericana en Informática, a realizarse en septiembre de 2014.

2. Evaluadores

2.1. Sara Méndez

Psicóloga de la Pontificia Universidad Javeriana, Magíster en Desarrollo Educativo y Social del Centro Internacional de Desarrollo Humano, docente de psicología educativa y

desarrollo psicológico, con 10 años de experiencia en procesos de formación para educadoras iniciales y políticas públicas en primera infancia.

2.2. **Luis Manuel Silva**

Psicólogo y Magíster en Historia de la Pontificia universidad Javeriana, candidato a Doctor en Psicología de la Universidad Nacional de Irlanda, con 10 años de experiencia docente en evaluación psicológica, análisis experimental de la conducta en la carrera de psicología y en la maestría de psicología clínica.

2.3. **Hernando Hurtado**

Ingeniero de sistemas de la Universidad Nacional, Magíster en Educación con énfasis en Cognición y Creatividad de la universidad Javeriana. Lleva más de 10 años como docente en diversas universidades del país, incluida la Universidad Javeriana. Es el actual jefe de la sección de programación y desarrollo de Software del Departamento de Ingeniería de Sistemas de la Universidad Javeriana.

3. **Formulario de Validación**

El formulario de validación se puede encontrar en el anexo 3. El formulario fue desarrollado con la asesoría del profesor del Departamento de Psicología de la Pontificia Universidad Javeriana Nicolás Gualteros. Se encuentra basado en el formulario usado para validar los artículos que se envían a consideración para ser publicados en la revista de investigación de la Facultad de Psicología y al formato de evaluación para que un artículo sea aceptado en el Congreso Latinoamericano de Estudios en Informática CLEI.

El formulario evalúa aspectos de forma del documento, tales como la validez y formalidad de la metodología de investigación, así como la relevancia de los temas estudiados. Adicionalmente, se evalúan aspectos más globales al solicitar la opinión del evaluador acerca de la pertinencia de la estrategia planteada por el modelo.

4. **Resultados Obtenidos**

Los formularios diligenciados por los expertos se pueden encontrar en el Anexo 4.

Cabe resaltar que dos de los evaluadores consideraron que todos los aspectos a evaluar estaban presentes en el documento de una manera satisfactoria, lo cual significa que están de acuerdo con la relevancia, metodología y fundamentación teórica del modelo.

Uno de los evaluadores opinó que no se han examinado las repercusiones teóricas y/o prácticas del desarrollo del modelo, y recomienda su implementación en forma de un Curso Abierto Masivo en Línea. (MOOC, por sus siglas en inglés). Esto da a entender que existen múltiples maneras de aprovechar el modelo aun sin haber sido consideradas desde un principio.

5. **Análisis de los Resultados**

La recepción de los evaluadores hacia el modelo fue positiva. Están de acuerdo con que el modelo posee una apropiada fundamentación teórica, realizando una revisión literaria apropiada.

Adicionalmente, están de acuerdo en que la metodología propuesta concuerda con posiciones contemporáneas en pedagogía y recomiendan su implementación en trabajos futuros. Resaltan en particular el enfoque hacia el auto aprendizaje, la contextualización para que los conocimientos tengan sentido para el aprendiz, la retroalimentación inmediata de los aciertos y errores, la elección individual del ritmo de aprendizaje y los contenidos a estudiar, junto con la posibilidad de comparar sus logros con los de los demás estudiantes, llegando incluso a interactuar con ellos para resolver problemas.

También concuerdan en que debido al uso de principios de diseño de Gamification y la revisión de literatura realizada, se disminuye la posibilidad de fracaso al implementar el modelo.

Los resultados obtenidos permiten concluir que según el criterio de expertos en las áreas de la pedagogía y la programación, el modelo es pertinente para resolver los problemas que intenta resolver, específicamente los problemas motivacionales, y que vale la pena implementar el modelo como apoyo a un curso de programación tradicional.

VII – CONCLUSIONES, RECOMENDACIONES Y TRABAJOS FUTUROS

1. Conclusiones

Gamification es una técnica novedosa y que se encuentra en auge como método para incrementar la participación y la motivación de los usuarios de los procesos que son 'gamificados'.

En el área de la educación, se ha utilizado en salones de clase para motivar la participación estudiantil en diferentes actividades con resultados mixtos pero alentadores. No se ha encontrado una relación directa entre el uso de técnicas y elementos de juegos en el salón con una mejora en el desempeño académico de los estudiantes, pero sí se ha encontrado que una aplicación correcta de las técnicas lleva a incrementar la participación y despertar motivación por el área donde se implementa.

Se propone un modelo utilizando esta técnica para motivar el aprendizaje de la programación. En contraste con otras implementaciones de *Gamification* en el salón de clases, esta utiliza un framework formal y reconocido de diseño para *Gamification* con el fin de incrementar las posibilidades de éxito al ser adoptado y lo combina con teorías establecidas de aprendizaje para lograr una experiencia educativa significativa. El framework específicamente elegido se enfoca en mantener presentes los objetivos planteados durante todo el proceso y en plantear métricas significativas para determinar si está funcionando. A diferencia de otros modelos educativos utilizados para el aprendizaje de la programación, este utiliza una medida objetiva para determinar su éxito, basándose siempre en la solución de problemas de programación por parte de los aprendices.

A diferencia de muchos de los cursos de programación tradicionales, se hace énfasis en el autoaprendizaje con una metodología que mantiene al estudiante practicando sus habilidades de solución de problemas constantemente y le informa sobre el sentido de lo que está aprendiendo a su propio ritmo. Estas características permitirán la apropiación profunda y duradera del conocimiento por parte de los estudiantes.

El modelo propuesto no busca ser la fuente esencial del aprendizaje sino un complemento que promueva la motivación intrínseca por el aprendizaje. Este diseño permite que el estudiante no se sienta obligado a utilizar el sistema 'gamificado', así evitando la degradación de una actividad que en un principio debía ser divertida a una que desmotiva.

Adicionalmente se presenta el desarrollo de un prototipo de una herramienta de software que implementa parcialmente el modelo propuesto con el fin de demostrar su factibilidad técnica. El prototipo puede funcionar como base para un desarrollo posterior que incluya todos los elementos y pueda ser utilizado en un salón de clases como herramienta de apoyo al aprendizaje.

El modelo fue puesto a la consideración de expertos en las áreas de la programación y de la pedagogía. Los expertos estuvieron de acuerdo en que el modelo presenta una alternativa novedosa y con suficientes bases teóricas para justificar su implementación, con un amplio prospecto de trabajos que pueden surgir a partir de él.

2. **Trabajos Futuros**

2.1. **Creación de los Personajes y Retos**

La creación de diferentes personajes y retos que representen los diferentes enfoques de la programación deberán ser diseñados para la implementación final del modelo. Se recomienda que este diseño se lleve a cabo por equipos multidisciplinares de las áreas de la programación, la pedagogía y el diseño.

Los retos y personajes son el corazón del modelo, así que el éxito de su adaptación podría depender de la calidad del contenido que se le presenta al estudiante.

2.2. **Extensión del Prototipo**

El prototipo desarrollado puede ser extendido para implementar los aspectos del modelo que no se encuentran presentes en este momento. Específicamente la recolección de estadísticas, los retos grupales y el motor de habilidades.

2.3. **Validación con Estudiantes**

Una vez el modelo haya sido implementado en su totalidad, es necesario realizar una validación con estudiantes para determinar si la metodología planteada tiene el efecto esperado.

2.4. **Mejora del motor de Juzgamiento**

Uno de los puntos fuertes del modelo es la retroalimentación constante que recibe el estudiante. Entre más detallada sea la retroalimentación que recibe el estudiante cuando comete un error, será más fácil para él aprender de sus errores y sobreponerse a ellos.

2.5. **Personalización de Contenidos**

Aprovechando los trabajos recientes realizados en la universidad sobre personalización de la información y motores de recomendación, se puede aplicar un sistema de personalización para el sistema donde según los retos y personajes y el perfil del estudiante, se le recomienden los próximos retos a ser solucionados.

2.6. **Explorar Nuevas Implementaciones del Modelo**

Uno de los expertos que evaluó el modelo recomendó el desarrollo de un MOOC basado en la metodología propuesta. Es posible que haya otras alternativas sin explorar que puedan hacer uso de los conceptos presentados en el modelo.

VIII - REFERENCIAS

[1] T. Jenkins, "The motivation of students of programming," *ACM SIGCSE Bull.*, vol. 33, no. 3, pp. 53–56, Sep. 2001.

[2] M. Feldgen and O. Clua, "New motivations are required for freshman introductory programming," in *33rd Annual Frontiers in Education, 2003. FIE 2003.*, 2003, vol. 1, pp. T3C_24–T3C_24.

[3] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining gamification," *Proc. 15th ...*, pp. 9–15, 2011.

[4] J. Hamari, J. Koivisto, and H. Sarsa, "Does Gamification Work? -- A Literature Review of Empirical Studies on Gamification," *2014 47th Hawaii Int. Conf. Syst. Sci.*, pp. 3025–3034, Jan. 2014.

[5] K. Werbach and D. Hunter, *For the Win: How Game Thinking Can Revolutionize Your Business*. 2012.

[6] J. M. Wing, "Computational Thinking CS @ CMU and Grand Vision for the Field," 2006.

[7] J. J. Lu and G. H. L. Fletcher, "Thinking about computational thinking," *ACM SIGCSE Bulletin*, vol. 41, no. 1. p. 260, 2009.

[8] D. Sleeman, "The challenges of teaching computer programming," *Commun. ACM*, vol. 29, no. 9, pp. 840–841, Sep. 1986.

[9] A. Robins, J. Rountree, and N. Rountree, "Learning and Teaching Programming : A Review and Discussion Learning and Teaching Programming : A Review," no. February 2014, pp. 37–41, 2010.

[10] I. Milne and G. Rowe, "Difficulties in Learning and Teaching Programming — Views of Students and Tutors," pp. 55–66, 2002.

[11] C. S. Dweek, "Motivational Processes Affecting Learning," vol. 41, no. 10, pp. 1040–1048, 1986.

[12] Ministerio de Educación Nacional, "Educación Superior, Boletín Informativo," *14*, p. 20, 2010.

[13] "Observatorio Laboral para la educación." [En línea]. Disponible en: http://redes.colombiaaprende.edu.co/ntg/observatorio/index.htm. [Consultado: 21-May-2014].

[14] "¿Cuáles son las carreras con mayor demanda laboral en Colombia?" [En línea]. Disponible en: http://noticias.universia.net.co/en-portada/noticia/2011/08/11/856011/cuales-son-carreras-demanda-laboral-colombia.html. [Consultado: 21-May-2014].

[15] "Edpcs | Free Development software downloads at SourceForge.net." [En línea]. Disponible en: http://sourceforge.net/projects/edpcs/. [Consultado: 21-May-2014].

[16] "PsiCoder | Free Home & Education software downloads at SourceForge.net." [En línea]. Disponible en: http://sourceforge.net/projects/psicoder/. [Consultado: 21-May-2014].

[17] "Scratch - Imagine, Program, Share." [En línea]. Disponible en: http://scratch.mit.edu/. [Consultado: 21-May-2014].

[18] "Alice.org." [En línea]. Disponible en: http://www.alice.org/index.php. [Consultado: 21-May-2014].

[19] S. Esper, S. R. Foster, and W. G. Griswold, "CodeSpells : Embodying the Metaphor of Wizardry for Programming," pp. 249–254, 2013.

[20] G. Fesakis and K. Serafeim, "Influence of the familiarization with 'scratch' on future teachers' opinions and attitudes about programming and ICT in education," *ACM SIGCSE Bull.*, vol. 41, no. 3, p. 258, Aug. 2009.

[21] C. Kelleher, R. Pausch, and S. Kiesler, "Storytelling alice motivates middle school girls to learn computer programming," *Proc. SIGCHI Conf. Hum. factors Comput. Syst. - CHI '07*, p. 1455, 2007.

[22] J. A. Villalobos, "Proyecto Cupi2 : un enfoque multidimensional frente al problema de enseñar y aprender a programar 7," pp. 45–64, 2009.

[23] K. Erenli, "The impact of gamification: A recommendation of scenarios for education," in *2012 15th International Conference on Interactive Collaborative Learning (ICL)*, 2012, pp. 1–8.

[24] J. J. Lee and J. Hammer, "Gamification in Education: What , How , Why Bother?," *Acad. Exch. Q.*, vol. 15, no. 2, pp. 1–5, 2011.

[25] C. Santos, S. Almeida, L. Pedro, M. Aresta, and T. Koch-Grunberg, "Students' Perspectives on Badges in Educational Social Media Platforms: The Case of

SAPO Campus Tutorial Badges," *2013 IEEE 13th Int. Conf. Adv. Learn. Technol.*, pp. 351–353, Jul. 2013.

[26] M. Guzdial and E. Soloway, "Teaching the Nintendo generation to program," *Commun. ACM*, vol. 45, no. 4, pp. 17–21, 2002.

[27] S. Garner, P. Haden, and A. Robins, "My Program is Correct But it Doesn ' t Run : A Preliminary Investigation of Novice Programmers ' Problems," 2003.

[28] F. MARTON and R. SÄLJÖ, "ON QUALITATIVE DIFFERENCES IN LEARNING: I-OUTCOME AND PROCESS*," *Br. J. Educ. Psychol.*, vol. 46, no. 1, pp. 4–11, Feb. 1976.

[29] T. Jenkins, "On the difficulty of learning to program," *Proc. 3rd Annu. Conf. …*, pp. 53–58, 2002.

[30] E. Lahtinen, K. Ala-Mutka, and H.-M. Järvinen, "A study of the difficulties of novice programmers," *ACM SIGCSE Bull.*, vol. 37, no. 3, p. 14, Sep. 2005.

[31] P. Kinnunen and B. Simon, "My program is ok – am I? Computing freshmen's experiences of doing programming assignments," *Comput. Sci. Educ.*, vol. 22, no. 1, pp. 1–28, Mar. 2012.

[32] M. Feldgen and O. Clua, "Games as a motivation for freshman students to learn programming," in *34th Annual Frontiers in Education, 2004. FIE 2004.*, 2004, pp. 1079–1084.

[33] D. Gries, "Where is programming methodology these days?," *ACM SIGCSE Bull.*, vol. 34, no. 4, p. 5, Dec. 2002.

[34] L. Lumsden, "Student Motivation To Learn. ERIC Digest, Number 92.," 1994.

[35] E. Deci and R. Ryan, *Handbook of Self-determinatino Research*. Rochester University Press, 2004.

[36] B. C. Thompson, "How Khan Academy Is Changing the Rules of Education," 2011.

[37] U. Jayasinghe and A. Dharmaratne, "Game based learning vs. gamification from the higher education students' perspective," *… , Assess. Learn. …*, no. August, pp. 683–688, 2013.

[38] S. Sheth, J. Bell, and G. Kaiser, "Increasing Student Engagement in Software Engineering with Gamification," pp. 1–2.

[39] C. Thomas and K. Berkling, "Redesign of a Gamified Software Engineering Course Step 2 Scaffolding : Bridging the Motivation Gap," *Interact. Collab. Learn. ...*, no. September, pp. 525–530, 2013.

[40] G. Barata, S. Gama, J. Jorge, and D. Gonçalves, "Engaging Engineering Students with Gamification An empirical study," 2013.

[41] "Octalysis: Complete Gamification Framework - Yu-kai Chou." [En línea]. Disponible en: http://www.yukaichou.com/gamification-examples/octalysis-complete-gamification-framework/#.U10xxPldXSg. [Consultado: 27-Apr-2014].

[42] R. Hunicke, M. Leblanc, and R. Zubek, "MDA : A Formal Approach to Game Design and Game Research."

[43] "Richard A. Bartle: Players Who Suit MUDs." [En línea]. Disponible en: http://mud.co.uk/richard/hcds.htm. [Consultado: 21-Mar-2014].

[44] A. J. Gomes, A. N. Santos, and A. J. Mendes, "A study on students' behaviours and attitudes towards learning to program," *Proc. 17th ACM Annu. Conf. Innov. Technol. Comput. Sci. Educ. - ITiCSE '12*, p. 132, 2012.

[45] S. Fincher, "Session 12a4 What are We Doing When We Teach Programming ? Session 12a4 The ' Literacy ' approach," pp. 8–12, 1999.

[46] N. Lazzaro, "Why we play games: Four keys to more emotion without story," 2004.

[47] H. Gardner and K. Davis, *The App Generation: How Today's Youth Navigate Identity, Intimacy, and Imagination in a Digital World*. Yale University Press, 2013, p. 256.

[48] E. Von Glasersfeld, "Constructivism in education," 1989.

[49] H. Gardner, *The Disciplined Mind: Beyond Facts and Standardized Tests, the K-12 Education That Every Child Deserves*. Penguin Group (USA) Incorporated, 2000, p. 296.

[50] J. P. Gee, "What video games have to teach us about learning and literacy," *Comput. Entertain.*, vol. 1, no. 1, p. 20, Oct. 2003.

[51] "Gamification platform and game mechanics manager | Joomla! extension." [En línea]. Disponible en: http://itprism.com/free-joomla-extensions/ecommerce-gamification/game-mechanics-platform. [Consultado: 24-May-2014].

[52] "Joomla! The CMS Trusted By Millions for their Websites." [En línea]. Disponible en: http://www.joomla.org/. [Consultado: 24-May-2014].

[53] "userinfuser - Open Source Gamification Platform - Google Project Hosting." [En línea]. Disponible en: https://code.google.com/p/userinfuser/. [Consultado: 24-May-2014].

[54] "AppScale: The Open Source App Engine." [En línea]. Disponible en: http://www.appscale.com/. [Consultado: 24-May-2014].

[55] "BadgeOS." [En línea]. Disponible en: https://badgeos.org/. [Consultado: 24-May-2014].

[56] "WordPress › Blog Tool, Publishing Platform, and CMS." [En línea]. Disponible en: http://wordpress.org/. [Consultado: 24-May-2014].

[57] "Open Glaze." [En línea]. Disponible en: http://www.openglaze.com/. [Consultado: 24-May-2014].

[58] "Behavior Platform | Badgeville, The #1 Gamification Platform." [En línea]. Disponible en: http://badgeville.com/products/behavior-platform. [Consultado: 25-May-2014].

[59] "Bunchball Nitro Gamification Platform with Nitro Studio | Bunchball." [En línea]. Disponible en: http://www.bunchball.com/products/nitro. [Consultado: 25-May-2014].

[60] "Engage Engine | IActionable." [En línea]. Disponible en: http://iactionable.com/engage-engine/. [Consultado: 25-May-2014].

[61] "BuddyPress.org." [En línea]. Disponible en: https://buddypress.org/. [Consultado: 25-May-2014].

[62] "Ideone.com - Online Compiler and IDE >> C/C++, Java, PHP, Python, Perl and 40+ other compilers and interpreters." [En línea]. Disponible en: https://ideone.com/sphere-engine. [Consultado: 25-May-2014].

[63] E. Bonilla-Castro and P. Rodríguez Sehk, *Más allá del dilema de los métodos*. 2005, p. 284.

VII – ANEXOS

Los anexos 2-4 están disponibles para ser descargados en la página web del trabajo de grado: http://pegasus.javeriana.edu.co/~CIS1410IS11/memoria.html

1. Glosario

API (Interfaz de Programación de Aplicaciones): Es una especificación de cómo funciona un producto de software. También provee una serie de rutinas y herramientas para utilizar dicho software.

Cliente-Servidor: Modelo de aplicaciones en el cual las tareas se dividen entre un proveedor de recursos, llamado servidor, y los consumidores de esos recursos, llamados clientes.

Framework: Una estructura o plataforma sobre la que se apoya algo más. Se usa como base para la algo que se construye.

Gestor de Contenido: Es un programa de computador que gestiona la creación, edición y publicación de contenido para un sitio web. También controlan el funcionamiento del sitio y ciertas funcionalidades como el registro de usuarios.

MOOC: Curso en línea enfocado hacia la participación ilimitada y acceso abierto a través de internet.

Plugin: Es un componente de software que extiende una aplicación existente para incluir una funcionalidad específica.

Widget: Es un elemento de software con funcionalidades específicas que se puede agregar a un sitio web para desarrollar alguna tarea con información que extrae de otras páginas web.

2. **Cuadro de Comparación de Herramientas**
3. **Formulario de Validación**
4. **Formularios Diligenciados por los Expertos**

ANEXO 2

CARTA DE AUTORIZACIÓN DE LOS AUTORES
(Licencia de uso)

Bogotá, D.C., Julio 29 de 2014

Señores
Biblioteca Alfonso Borrero Cabal S.J.
Pontificia Universidad Javeriana
Cuidad

Los suscritos:
Ricardo José Arenas París , con C.C. No 1018406517
_____ , con C.C. No _____
_____ , con C.C. No _____

En mi (nuestra) calidad de autor (es) exclusivo (s) de la obra titulada:
Modelo para la motivación del Aprendizaje de la programación utilizando Gamification
(por favor señale con una "x" las opciones que apliquen)
Tesis doctoral ☐ Trabajo de grado ☒ Premio o distinción: **Si** ☐ **No** ☐
cual: _____
presentado y aprobado en el año 2014 , por medio del presente escrito autorizo (autorizamos) a la Pontificia Universidad Javeriana para que, en desarrollo de la presente licencia de uso parcial, pueda ejercer sobre mi (nuestra) obra las atribuciones que se indican a continuación, teniendo en cuenta que en cualquier caso, la finalidad perseguida será facilitar, difundir y promover el aprendizaje, la enseñanza y la investigación.

En consecuencia, las atribuciones de usos temporales y parciales que por virtud de la presente licencia se autorizan a la Pontificia Universidad Javeriana, a los usuarios de la Biblioteca Alfonso Borrero Cabal S.J., así como a los usuarios de las redes, bases de datos y demás sitios web con los que la Universidad tenga perfeccionado un convenio, son:

AUTORIZO (AUTORIZAMOS)	SI	NO
1. La conservación de los ejemplares necesarios en la sala de tesis y trabajos de grado de la Biblioteca.	X	
2. La consulta física (sólo en las instalaciones de la Biblioteca)	X	
3. La consulta electrónica - on line (a través del catálogo Biblos y el Repositorio Institucional)	X	
4. La reproducción por cualquier formato conocido o por conocer	X	
5. La comunicación pública por cualquier procedimiento o medio físico o electrónico, así como su puesta a disposición en Internet	X	
6. La inclusión en bases de datos y en sitios web sean éstos onerosos o gratuitos, existiendo con ellos previo convenio perfeccionado con la Pontificia Universidad Javeriana para efectos de satisfacer los fines previstos. En este evento, tales sitios y sus usuarios tendrán las mismas facultades que las aquí concedidas con las mismas limitaciones y condiciones	X	

De acuerdo con la naturaleza del uso concedido, la presente licencia parcial se otorga a título gratuito por el máximo tiempo legal colombiano, con el propósito de que en dicho lapso mi (nuestra) obra sea explotada en las condiciones aquí estipuladas y para los fines indicados, respetando siempre la titularidad de los derechos patrimoniales y morales correspondientes, de

acuerdo con los usos honrados, de manera proporcional y justificada a la finalidad perseguida, sin ánimo de lucro ni de comercialización.

De manera complementaria, garantizo (garantizamos) en mi (nuestra) calidad de estudiante (s) y por ende autor (es) exclusivo (s), que la Tesis o Trabajo de Grado en cuestión, es producto de mi (nuestra) plena autoría, de mi (nuestro) esfuerzo personal intelectual, como consecuencia de mi (nuestra) creación original particular y, por tanto, soy (somos) el (los) único (s) titular (es) de la misma. Además, aseguro (aseguramos) que no contiene citas, ni transcripciones de otras obras protegidas, por fuera de los límites autorizados por la ley, según los usos honrados, y en proporción a los fines previstos; ni tampoco contempla declaraciones difamatorias contra terceros; respetando el derecho a la imagen, intimidad, buen nombre y demás derechos constitucionales. Adicionalmente, manifiesto (manifestamos) que no se incluyeron expresiones contrarias al orden público ni a las buenas costumbres. En consecuencia, la responsabilidad directa en la elaboración, presentación, investigación y, en general, contenidos de la Tesis o Trabajo de Grado es de mí (nuestro) competencia exclusiva, eximiendo de toda responsabilidad a la Pontifica Universidad Javeriana por tales aspectos.

Sin perjuicio de los usos y atribuciones otorgadas en virtud de este documento, continuaré (continuaremos) conservando los correspondientes derechos patrimoniales sin modificación o restricción alguna, puesto que de acuerdo con la legislación colombiana aplicable, el presente es un acuerdo jurídico que en ningún caso conlleva la enajenación de los derechos patrimoniales derivados del régimen del Derecho de Autor.

De conformidad con lo establecido en el artículo 30 de la Ley 23 de 1982 y el artículo 11 de la Decisión Andina 351 de 1993, *"Los derechos morales sobre el trabajo son propiedad de los autores"*, los cuales son irrenunciables, imprescriptibles, inembargables e inalienables. En consecuencia, la Pontificia Universidad Javeriana está en la obligación de RESPETARLOS Y HACERLOS RESPETAR, para lo cual tomará las medidas correspondientes para garantizar su observancia.

NOTA: <u>Información Confidencial:</u>
Esta Tesis o Trabajo de Grado contiene información privilegiada, estratégica, secreta, confidencial y demás similar, o hace parte de una investigación que se adelanta y cuyos resultados finales no se han publicado. Si ☐ No ☒
En caso afirmativo expresamente indicaré (indicaremos), en carta adjunta, tal situación con el fin de que se mantenga la restricción de acceso.

NOMBRE COMPLETO	No. del documento de identidad	FIRMA
Ricardo José Arenas París	1013406517	

FACULTAD: Ingeniería
PROGRAMA ACADÉMICO: Ingeniería de Sistemas

ANEXO 3
BIBLIOTECA ALFONSO BORRERO CABAL, S.J.
DESCRIPCIÓN DE LA TESIS O DEL TRABAJO DE GRADO
FORMULARIO

TÍTULO COMPLETO DE LA TESIS DOCTORAL O TRABAJO DE GRADO	
Modelo para la motivación del Aprendizaje de la Programación	
SUBTÍTULO, SI LO TIENE	
Utilizando Gamificación	
AUTOR O AUTORES	
Apellidos Completos	**Nombres Completos**
Arenas París	Ricardo José
DIRECTOR (ES) TESIS O DEL TRABAJO DE GRADO	
Apellidos Completos	**Nombres Completos**
Avellaneda Pachón	Fabio Antonio
FACULTAD	
Ingeniería	
PROGRAMA ACADÉMICO	
Tipo de programa (seleccione con "x")	
Pregrado: X	Especialización:	Maestría:	Doctorado:
Nombre del programa académico	
Ingeniería de Sistemas	
Nombres y apellidos del director del programa académico	
Germán Alberto Chavarro Flore	
TRABAJO PARA OPTAR AL TÍTULO DE:	
Ingeniero de Sistemas	
PREMIO O DISTINCIÓN *(En caso de ser LAUREADAS o tener una mención especial)*:	

CIUDAD	AÑO DE PRESENTACIÓN DE LA TESIS O DEL TRABAJO DE GRADO	NÚMERO DE PÁGINAS
Bogotá	2014	86

TIPO DE ILUSTRACIONES (seleccione con "x")

Dibujos	Pinturas	Tablas, gráficos y diagramas	Planos	Mapas	Fotografías	Partituras
		X				

SOFTWARE REQUERIDO O ESPECIALIZADO PARA LA LECTURA DEL DOCUMENTO

Nota: En caso de que el software (programa especializado requerido) no se encuentre licenciado por la Universidad a través de la Biblioteca (previa consulta al estudiante), el texto de la Tesis o Trabajo de Grado quedará solamente en formato PDF.

MATERIAL ACOMPAÑANTE					
TIPO	DURACIÓN (minutos)	CANTIDAD	FORMATO		
			CD	DVD	Otro ¿Cuál?
Vídeo	6	2			MP4
Audio					
Multimedia					
Producción electrónica		1			Prototipo de Software
Otro Cuál?					

DESCRIPTORES O PALABRAS CLAVE EN ESPAÑOL E INGLÉS

Son los términos que definen los temas que identifican el contenido. *(En caso de duda para designar estos descriptores, se recomienda consultar con la Sección de Desarrollo de Colecciones de la Biblioteca* Alfonso Borrero Cabal S.J *en el correo biblioteca@javeriana.edu.co, donde se les orientará).*

ESPAÑOL	INGLÉS
Gamificación	Gamification
Programación	Programming
Educación	Education
Motivación	Motivation

RESUMEN DEL CONTENIDO EN ESPAÑOL E INGLÉS

(Máximo 250 palabras - 1530 caracteres)

Este trabajo describe un modelo para el aprendizaje de la programación utilizando técnicas de Gamificación para resolver los problemas motivacionales que surgen en el proceso de enseñanza y aprendizaje de la motivación.

This work describes a model for the learning of programming using gamification techniques to solve the motivational problems that appear during the teaching and learning process of programming.