CIS1530AP03

Prototipo funcional para apoyar el aprendizaje de la Programación utilizando Gamification

Daniel Felipe Arias Ramírez
Daniel Serrano Reyes

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA DE SISTEMAS
BOGOTÁ, D.C.
2015

CIS1530AP03

Prototipo funcional para apoyar el aprendizaje de la Programación utilizando Gamification

Autor(es):

Daniel Felipe Arias Ramírez

Daniel Serrano Reyes

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, M.Sc.

Página web del Trabajo de Grado

http://pegasus.javeriana.edu.co/~CIS1530AP03

PONTIFICIA UNIVERSIDAD JAVERIANA

FACULTAD DE INGENIERÍA

CARRERA DE INGENIERÍA DE SISTEMAS

BOGOTÁ, D.C.

Noviembre, 2015

PONTIFICIA UNIVERSIDAD JAVERIANA
FACULTAD DE INGENIERÍA
CARRERA DE INGENIERÍA DE SISTEMAS

Rector Magnífico

Jorge Humberto Peláez Piedrahita, S.J.

Decano Facultad de Ingeniería

Ingeniero Jorge Luis Sánchez Téllez

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

Agradecimiento Daniel Serrano

Agradezco a mi familia por su apoyo incondicional durante el transcurso de la carrera, y especialmente el apoyo en todas las decisiones que he tomado. Sin su ayuda no habría sido posible llegar hasta el lugar donde me encuentro.

También agradezco a Fabio, no solo por su apoyo durante la dirección de este trabajo de grado, sino por sus enseñanzas durante toda la carrera, al igual que por su amistad. Agradezco al Capítulo Javeriano ACM, donde encontré mi verdadera pasión. Igualmente al grupo de maratones donde encontré verdaderos amigos y fortalecí mis conocimientos técnicos, en el tema que verdaderamente me apasiona.

Finalmente a Alfredo, por su ayuda en la validación del sistema, no habría sido posible verificar la efectividad de la herramienta sin su ayuda.

Agradecimientos Daniel Arias

Primero a Dios quien me dio la sabiduría y las fuerzas para levantarme cada día y poder seguir adelante, a mi familia; mi mamá, Cecilia Ramírez, mi papá, Pablo Emilio Arias y mi hermano, Pablo Andrés, quienes siempre me apoyaron y confiaron en mí en cada etapa de este recorrido y sin ellos no sería la persona que soy hoy en día y con los que estaré eternamente agradecido.

Agradezco a los profesores del departamento de Ingeniería de Sistemas por brindar todos sus conocimientos y a todos los colegas con los que tuve la fortuna de compartir durante estos cinco años de carrera, especialmente a Paola Huertas, Daniela Prada, Sebastián Ulloa y Daniel Vacca, un excelente equipo de trabajo y más que eso amigos incondicionales quienes fueron un gran apoyo.

También agradezco a Fabio, nuestro director de trabajo de grado por depositar su confianza en nosotros, guiarnos y apoyarnos durante en el desarrollo del proyecto. Finalmente a Alfredo por su ayuda y compromiso en la validación del sistema.

CONTENIDO

Lista de ilustraciones

Lista de tablas

ABSTRACT

Nowadays programming is a highly important activity, because it helps to structure the way of thinking into a systematic process, for the development of tasks and activities. The process of teaching programming today has some deficiencies especially in the area of motivating the students. However, there is not a solution that covers these shortcomings yet. For this reason Game -ludex was created, a support tool that uses gamification techniques and seeks to solve motivational problems that arise in the process of learning and teaching programming classes.

RESUMEN

La programación hoy en día es una actividad de alta importancia, debido a que ayuda a estructurar el pensamiento de una manera sistemática para el desarrollo de tareas y actividades de la vida diaria. El método educativo que se usa en la actualidad tiene algunas deficiencias en cuanto a la enseñanza y la motivación en los estudiantes, sin embargo, todavía no se ha implementado una forma de enseñar que cubra estas falencias. Por este motivo se creó **Game-ludex,** una herramienta de apoyo que hace uso de técnicas de *Gamification* con las cuales se busca resolver los problemas motivacionales que surgen en el proceso de aprendizaje y enseñanza en cursos de programación.

INTRODUCCIÓN

En la presente memoria se desarrolla todo el proceso especificado en la propuesta de trabajo de grado "Sistema para apoyar el aprendizaje de la Programación utilizando *Gamification*", dando como resultado la implementación de Game-Iudex, un prototipo funcional con técnicas de *Gamification*, herramienta de apoyo para la motivación en el aprendizaje de cursos introductorios de programación. En los siguientes párrafos se especificará la distribución y organización del documento.

En la primera sección se describe el contexto sobre el cual se desarrolló el trabajo de grado, la problemática que se identificó, el problema a resolver y la importancia de porqué el desarrollo de una solución acerca de este problema. Continuando con una descripción general del proyecto, se plantearon los objetivos específicos orientados al cumplimiento del objetivo general, y la metodología con la que se realizó el proyecto. La segunda sección detalla los conceptos necesarios para comprender el entorno de la solución, junto al análisis de las soluciones relacionadas a la problemática identificada y porqué la solución desarrollada en este trabajo de grado tiene un valor agregado que la diferencia de las otras.

La tercera sección presenta el análisis de los aspectos a considerar antes de la implementación de la solución, lo que incluye los requerimientos del prototipo, sus especificaciones y su priorización, junto al plan de pruebas con el cual se valida la funcionalidad y el impacto generado por la herramienta desarrollada.

La cuarta sección hace referencia al diseño y desarrollo del prototipo, donde primero se hace un análisis y escogencia tanto de las herramientas base como del *framework* para el desarrollo de la solución, las técnicas de *Gamification* implementadas y los procesos que se llevan a cabo durante el uso de la herramienta junto al desarrollo de la metodología propuesta. Se presenta el diagrama relacional de bases de datos, el diagrama de secuencia y la arquitectura del prototipo para cumplir con los requerimientos especificados en la sección anterior.

La quinta sección, muestra el análisis y los resultados obtenidos según el plan de pruebas descrito en la sección de Análisis. La sexta sección presenta las conclusiones respecto al desarrollo de la solución y los resultados obtenidos, en donde se analiza el impacto obtenido por la herramienta en diferentes áreas y el trabajo a futuro para complementar el prototipo y llegar al alcance deseado sobre las posibilidades que brinda esta solución.

Por último, la séptima y octava sección ofrecen la posibilidad al lector de indagar y ampliar la información sobre los temas tratados, donde se incluyen las referencias y anexos.

I - DESCRIPCIÓN GENERAL

1. Oportunidad, Problemática, Antecedentes

La programación de computadores consiste en definir procesos, actividades y tareas, identificando la serie de pasos lógicos para realizar dicha función, al igual que modelar objetos del mundo real y definir cómo interactúan entre sí [1]. Esta es una actividad de alta importancia hoy en día por varias razones, una de ellas es que el mundo está gobernado por la tecnología, y lo continuará estando cada vez más, por lo que es importante saber cómo funcionan estas tecnologías y entender la forma en la que son creadas las herramientas usadas. Otra razón es quizás la más obvia, poder desarrollar y diseñar estas aplicaciones tecnológicas. Pero la razón más importante no es para enfocar a todos a ser ingenieros de sistemas, sino para que aprendan a pensar algorítmicamente, una habilidad que puede ser usada en cualquier campo del conocimiento [2].

La metodología que se ha venido desarrollando para la enseñanza de esta ciencia tiene algunas deficiencias, debido a que está basada en replicar los conocimientos teóricos que se acaban de aprender, lo cual no motiva a los estudiantes ni los impulsa a relacionar los conceptos y buscar una solución por sus propios medios [3]. Algunas de estas deficiencias son: los estudiantes se frustran fácilmente, no hay sufi-

ciente motivación y muestran falsa percepción respecto a la dificultad de los problemas [4]. Debido a esto se genera una alta deserción de estudiantes en los cursos introductorios y temor a inscribir estas asignaturas por parte de otros estudiantes [4][5], lo cual se convierte en un problema, especialmente en esta área del conocimiento, pues no aporta en la solución de la escasez de profesionales a nivel mundial [6].

Otra de las principales razones que causan el retiro de estas asignaturas es la falta de seguimiento individual del profesor a los estudiantes [5], causando una mala retroalimentación y dificultando el crecimiento de los estudiantes.

Se observa que la deserción en carreras tecnológicas en el país es grande, aún presentando unos de los salarios más altos. Según la revista Forbes [6], los trabajos relacionados con las ciencias de la computación ocupan el segundo puesto en la lista de mejores salarios para recién egresados. En contraste, según el Ministerio de Educación de Colombia, estas mismas carreras presentan los índices de deserción más altos en el país [7]. Esto demuestra una clara desmotivación por parte de los estudiantes.

Una solución posible para este problema es el concepto de *Gamification*; "*Gamification* es un término informal para el uso de elementos de videojuegos en sistemas que no sean juegos, para aumentar la experiencia de usuario, así como su compromiso" [8]. Es una técnica ampliamente usada desde hace varios años con resultados exitosos. El objetivo de este documento es mostrar mediante un prototipo funcional, que la aplicación de este concepto puede ser utilizada también para enseñar y atraer estudiantes al área de la programación.

Formulación del problema que se resolvió

¿Cómo aplicar la teoría de *Gamification* para mejorar el proceso de aprendizaje de programación en universidades y colegios?

Justificación del problema

Otro elemento que se implementa al tiempo con G*amification* es el de evaluación automática [9]el cual consiste en dar retroalimentación automática al código de los estudiantes por medio de un software. Con esto se espera aportar ventajas significativas al estudiante como: mayor rapidez en recibir una respuesta, posibilidad de múltiples intentos, mejor retroalimentación, calificación objetiva, detección de copia, entre otros. La corrección automática se basa en un software al que los estudiantes envían su código (o el resultado de haber ejecutado un conjunto de entrada en el ejercicio), recibiendo la retroalimentación inmediatamente. Esta retroalimentación puede ser tan sencilla como decir si la respuesta es o no correcta, o se pueden incluir análisis más elaborados como por ejemplo el tiempo de ejecución, memoria utilizada, estructuras de datos empleadas, etc., para cursos más complejos como estructuras de datos, programación de computadores, análisis de algoritmos, entre muchos otros.

Esta técnica es muy usada en ambientes competitivos como la International Collegiate Programming Contest (ICPC) [10], organizada por la Association for Computing Machinery (ACM) [11], o el CodeJam [12] de Google por ejemplo.

Un ejemplo de estos resultados se pueden observar en la Universidad de Helsinki [13], en donde el número de estudiantes con calificación mayor a 90% aumentó de 334 a 718, gracias a la utilización de un sistema de calificación automática.

Varias páginas de educación virtual han utilizado conceptos similares a *Gamification* para dictar cursos de diferentes áreas. Esta iniciativa ha sido liderada por algunas de las mejores universidades del mundo, dando su material de forma gratuita a personas ajenas a la universidad. Algunos ejemplos destacados de estas páginas son Coursera [14] con más de 3 millones de usuarios y Khan Academy con más de 10 millones [13]. Estas páginas funcionan bien cuando el estudiante quiere aprender por su propia cuenta, aunque se considera que hace falta el componente de G*amification* en cursos de programación un poco más avanzados que los introductorios. Lo

que este tipo de páginas no abordan es el ser una herramienta de apoyo a una institución educativa, en donde hay un profesor que enseña los materiales necesarios, dejando la herramienta como un material de apoyo para el trabajo individual.

Impacto Esperado

El impacto que tendría el uso de la plataforma **Game-ludex**.

Corto plazo: Tendrá un impacto principalmente en el área de la educación, al ser utilizada para el aprendizaje en cursos introductorios de programación, en donde se espera la disminución de la tasa de reprobación y la tasa de deserción por parte de los estudiantes que ingresen a carreras relacionadas con la ingeniería de sistemas. Con el uso de esta herramienta no solo se pretende motivar al estudiante para que siga resolviendo ejercicios que lo ayudarán durante el transcurso de la asignatura, sino que además se ofrecen facilidades al profesor y al monitor sobre el estado actual del estudiante y el desarrollo de la materia, dando la posibilidad de ofrecer una retroalimentación temprana tanto desde la herramienta como por parte del profesor o el monitor, al estudiante.

Mediano plazo: Se espera extender el impacto generado a corto plazo al implementar el uso de la plataforma en cualquier curso relacionado a la programación, donde los usuarios se vean motivados por el uso de G*amification* notando una mejoría en cuanto al aprendizaje y sus resultados al final del curso.

Largo plazo: Ser la plataforma escogida por todas las instituciones que quieran enseñar programación, también para personas naturales que quieran practicar y reforzar sus habilidades puedan hacer uso de **Game-ludex** como herramienta de estudio.

2. Descripción del Proyecto

Objetivo general

Crear un prototipo funcional que utilice técnicas de Gamification al proceso de aprendizaje, para ser usado en cursos introductorios de programación.

Objetivos específicos

- Seleccionar los factores que dificultan el aprendizaje de la programación que se resolverán y a partir de estos especificar los requerimientos funcionales y no funcionales.
- Seleccionar las técnicas de *Gamification* con las que será desarrollado el sistema basado en los requerimientos.
- Desarrollar un prototipo funcional que cumpla con los requerimientos propuestos.
- Validar el funcionamiento del sistema en estudiantes de cursos introductorios.

3. Metodología

Para el desarrollo del presente trabajo de grado se utilizó una adaptación de la metodología iterativa incremental [15], para poder llevar a cabo las diferentes fases planteadas, donde cada fase se relaciona con un objetivo específico.

Fase 1

Selección de los factores que dificultan el aprendizaje de la programación que se esperan resolver y correspondiente especificación de los requerimientos funcionales y no funcionales.

Actividades:

- Elaboración del estado del arte:
    - Marco Contextual
        - Análisis de las diferentes metodologías y las diferentes problemáticas en la enseñanza de la programación.
        - Concepto de *Gamification* y su uso en diferentes ambientes.
        - E-learning como técnica de aprendizaje.
        - Motivación extrínseca e intrínseca del estudiante.
    - Marco Teórico
        - Análisis de herramientas relacionadas con *Gamification* y la enseñanza de la programación, jueces virtuales, MOOC's [16] y plataformas en línea donde se resuelven problemas de programación.
- Identificación de los requerimientos a tener en cuenta luego del análisis previo y la visión general sobre el prototipo funcional donde se implementa el uso de *Gamification.*
    - Identificación de requerimientos funcionales.
    - Identificación de requerimientos no funcionales.

Resultados:

- Listado de requerimientos (Ver anexo Requerimientos).

Fase 2

Selección de las técnicas de *Gamification* con las que se desarrolló el sistema, basada en los requerimientos.

Actividades:

- Identificación de las técnicas de *Gamification* a ser implementadas con respecto del trabajo de grado "titulado Modelo de la programación utilizando *Gamification*" [17].

- Priorización de requerimientos de acuerdo a las técnicas escogidas y el alcance del prototipo.

Resultados:

- Listado de requerimientos priorizados (Ver anexo Requerimientos).
- Documento con la descripción de las técnicas de *Gamification* implementadas en el sistema.

Fase 3

Desarrollo del prototipo funcional que cumple con los requerimientos propuestos.

Actividades:

- Realización del diseño del prototipo funcional pensando a largo plazo, teniendo en cuenta los requerimientos y su priorización.
    - Diseño de Diagrama de clases.
    - Diseño de Bases de datos.
    - Diseño de Arquitectura del sistema.
- Análisis de las herramientas en las cuales se implementará la solución.
    - Escogencia del lenguaje de programación para desarrollar el prototipo funcional.
    - Análisis y escogencia del framework que permitió el desarrollo e implementación del diseño de acuerdo al lenguaje seleccionado.
    - Escogencia de la herramienta en la cual se mantuvo el versionamiento y almacenamiento del código fuente.
    - Escogencia del servidor de pruebas donde se publicó el prototipo funcional.
- Luego de completar la iteración del diseño, se empezó a desarrollar la implementación siguiendo una metodología de desarrollo basada en issues, los cuales permitían dar un mejor seguimiento al proyecto en cuanto a tareas

pendientes y errores del prototipo, administrando de una mejor manera los esfuerzos requeridos para mejoras o nuevas funcionalidades.

- La agregación de un nuevo issue especificaba un título, una descripción del error o la funcionalidad para agregar o modificar, una etiqueta, con la que se indicaba el grado de prioridad con la que se debía considerar el issue y finalmente se asignaba el responsable de realizar la modificación.
- Para la agregación de nuevos issues, se puso a prueba el prototipo con los estudiantes de una clase de programación de computadores, a quienes se les pidió dar retroalimentación de la plataforma y con esta información se lograban crear nuevos issues.
- Teniendo en cuenta que el profesor y monitor también se veían beneficiados con el uso de la herramienta, se les otorgaron permisos sobre el repositorio en github para la agregación de nuevos issues.
- Habiendo creado y asignado los nuevos issues, estos se implementaban y se probaban en el prototipo. Luego de tener certeza de que el desarrollo realizado solucionaba el issue correspondiente, se realizaba el commit sobre el repositorio, especificando como mensaje los issues que se solucionaban. Finalmente, se realizaba un comentario en el issue indicando cómo se solucionó y se procedía a su cierre.

- **Resultados:**
- Diseño de bases de datos
- Diseño de arquitectura del sistema
- Desarrollo del prototipo (Ver videos)

Fase 4

Validación del funcionamiento del sistema con estudiantes de cursos introductorios.

Actividades:

- Prueba con estudiantes de curso de Programación de Computadores.

- Adición de issues de acuerdo a la retroalimentación de los estudiantes, el monitor y el profesor.
- Encuesta realizada a los estudiantes y al monitor luego de tres semanas de uso de la plataforma.

Resultados:

- Encuesta realizada a estudiantes (Ver anexo Formulario encuestas).
- Encuesta realizada al monitor (Ver Encuesta monitor).

II – MARCO TEÓRICO

Marco Conceptual

Actualmente no existe un consenso sobre qué temas y de qué manera se deberían enseñar en los cursos introductorios de programación [18]. Se mostrará a continuación qué técnicas, temas y problemáticas existen en estos cursos alrededor del mundo. Para esto, se utilizarán como base artículos en lo que se realizaron encuestas a profesores de cursos de programación introductorios en universidades y colegios.

Existen varias metodologías para enseñar la programación, a continuación se presentan cuatro tipos de metodología mencionadas por Essi Lahtinen que resultan ser las más recurrentes en las instituciones [19]:

- **Ambiente libre de sintaxis:** En esta metodología, se busca separar lo referente a escribir código de lo referente a programar, es decir se intenta que el estudiante entienda los conceptos de resolución de problemas, antes de pasar a programar soluciones. Este tipo de metodología se puede hacer incluso sin medios electrónicos [20]. Una vez el estudiante ha entendido los conceptos básicos se procede al uso de un seudocódigo para resolver problemas sencillos [21].

- ***Literacy*:** Es una metodología similar a la libre de sintaxis, en el sentido que primero se enseñan conceptos separados que serán útiles a la hora de programar. Se diferencia de la metodología anterior en que se utiliza un lenguaje de programación para que los estudiantes puedan aplicar los conceptos aprendidos, por lo tanto, es una metodología más práctica, pero también con un componente fuertemente teórico [21].
- **Solución de problemas:** Se enfoca en enseñar a los estudiantes el análisis y diseño en el área de resolución de problemas. Se asume que se trata de una habilidad extrapolable, que se puede aplicar en otros campos una vez sea dominada. Se utiliza usualmente con un solo lenguaje de programación para poner en práctica los conocimientos, pero puede ser aplicada con más de un lenguaje durante varios cursos, o sin el uso de algún lenguaje [21].
- **Computación como interacción:** Este paradigma es un poco diferente a los anteriores. La idea es aprovechar que la mayoría de estudiantes estén familiarizados con gran cantidad de interfaces y programas existentes. La forma tradicional se basa en enseñar a programar con un único hilo de ejecución y con muy poca interacción con el usuario [21]. Este método busca enseñar desde el primer curso una programación más real en donde existan varios hilos de ejecución y una interfaz mucho más clara con el usuario [22].

Vale la pena aclarar que aunque los anteriores son los modelos descritos, la mayor cantidad de profesores e instituciones utilizan una combinación de ellos [19].

Partiendo de los artículos en donde se realizaron encuestas a estudiantes y profesores [21][19], se encontraron los siguientes resultados útiles para analizar la problemática en la enseñanza de la programación:

- En la mayoría de clases que no utilizan un ambiente libre de sintaxis, se utilizan lenguajes usados en la industria. Aunque es importante conocer estos lenguajes, tienen una gran cantidad de funcionalidades y por lo tanto son en ocasiones difíciles de aprender. Según la encuesta, los lenguajes más usados son C++ (73%) y Java (17%) [19].

- Los estudiantes tienden a preferir diferentes métodos para aprender, y consideran que aprenden más de alguna forma específica. Según la encuesta, los estudiantes percibían que estaban aprendiendo significativamente (en orden de mayor a menor aprendizaje) en los siguientes métodos: trabajo individual de tareas, estudio individual, en sesiones prácticas, sesiones en clase de trabajo en grupo, en clases magistrales. Por otro lado, los profesores encuestados encuentran las mejores prácticas en el siguiente orden, de mayor aprendizaje a menor: sesiones prácticas, trabajo individual de tareas, sesiones en clase de trabajo en grupo, estudio individual, clases magistrales [19].
- No hay un currículo estándar y los temas que se enseñan varían mucho entre instituciones, pero los temas que más dificultad presentan los estudiantes son: recursión, eficiencia de algoritmos y estructuras de datos [18].

Aparte de analizar el estado actual de la enseñanza de la programación, también es necesario evaluar cuáles son los principales problemas en la forma en la que actualmente se enseña. Los principales elementos problemáticos encontrados, fueron:

- La enseñanza se basa más en problemas específicos de programación al igual que específicamente en un lenguaje, que en la habilidad de resolver problemas mediante la programación [18]. Aunque muchos profesores saben esto, es difícil dirigir la clase hacia algo totalmente abstracto, dado que los estudiantes pierden motivación y no es fácil para ellos entender los conceptos sin algún tipo de aplicación práctica [21]. La mayor dificultad que se tiene se debe a los lenguajes usados, dado que en su mayoría son lenguajes de alta complejidad, en donde no es sencillo entender la sintaxis [19].
- Los estudiantes son capaces de implementar los programas que los profesores piden para las tareas o ejercicios, pero no tienen el modelo mental de lo que está pasando en la parte de memoria del computador. Conceptos incorrectos de este tipo llevan a problemas en temas más complicados, como los mencionados anteriormente como los más problemáticos [18].
- Los estudiantes obtienen conocimientos muy superficiales, es decir, dado que la clase usualmente se basa en la sintaxis del lenguaje, más que en en-

tender qué está pasando más profundamente, los estudiantes entienden línea por línea del código, pero no pueden extrapolar los conocimientos aprendidos a situaciones ligeramente diferentes a los aprendidos en clase, ni pueden avanzar fácilmente a temas más avanzados [19].

- Hay una gran diferencia en los conocimientos previamente adquiridos, por lo que no es fácil encontrar un punto medio en el que todos estén aprendiendo, y sea retador al mismo tiempo. En una encuesta realizada a más de 500 estudiantes, 58% tenían conocimientos sobre programar [19].

En los últimos años la enseñanza virtual se ha convertido en una de las soluciones para apoyar la educación en masa, resolviendo necesidades que la educación tradicional ha sido incapaz de resolver, como el acceso de las personas distantes a las instituciones educativas o con escasos recursos para hacer pagos de matrículas completas, problemas de tiempo y horarios que puedan abarcar gran cantidad de estudiantes como profesores, o la baja demanda la cual no justifica la apertura de estos cursos [23].

Aunque la educación a distancia resuelve estos problemas, también tiene fallos en cuanto a la poca interacción entre un profesor y su alumno, lo cual dificulta la rectificación de errores al estudiante [24].

Por su parte, el e-learning es una de las tantas técnicas de aprendizaje la cual está basada en la enseñanza de cursos o temas a distancia apoyándose en los avances de las telecomunicaciones. Esta técnica puede utilizarse como herramienta de apoyo a un curso donde los estudiantes pueden entrar libremente y resolver o repasar los temas vistos en clase o como el curso completo en donde la mayoría transmiten los conocimientos por medios multimedia como los videos [25].

Para poder desarrollar un buen entorno virtual para el estudiante, se debe tener en cuenta que todos los seres humanos aprenden de maneras distintas, lo importante es poder estimular de la mejor forma posible estos métodos de aprendizaje en los estudiantes, ofreciendo una mejora en cuanto a la personalización del aprendizaje, lo cual sería muy complejo para la enseñanza tradicional pues es inviable ofrecer un

profesor o un curso distinto que se acomode a los métodos de aprendizaje por cada uno de los integrantes presenciales del curso.

Los pasos a tener en cuenta al momento de desarrollar un curso de enseñanza virtual comprenden la descripción de los contenidos y las técnicas pedagógicas que se usarán para este, adaptándose a las diferentes formas de aprendizaje de los estudiantes, la correlación de los temas y la planificación de las rutas que puede tomar el estudiante para realizar el curso [26]. Esta serie de pasos a tener en cuenta en el desarrollo del curso virtual proveen beneficios en el aprendizaje del estudiante como la retroalimentación temprana y oportuna, la fácil actualización de los contenidos, el uso de contenido multimedia para mejorar la experiencia con el usuario y el desarrollo del curso por parte del estudiante a su propio ritmo. Se debe tener en cuenta que no se trata de la realización de una clase tradicional plasmada en un video [27].

Gamification es un término que se ha convertido en tendencia. Como resultado, ha llegado a una definición estándar en donde se entiende como un proceso que integra características de juegos en ambientes que no son juegos [28]. Estas características se usan de acuerdo al contexto en donde se quieran aplicar, ya que no se ha llegado a una metodología o un modelo específico para realizar con éxito este proceso [29].

Estas características se usan para atraer clientes y mejorar su motivación al hacer uso de estos servicios. Se debe tener en cuenta que el uso de *Gamification* no debe pasar ciertos lineamientos en los que pueda convertirse en un juego [28]. El uso de *Gamification* ha demostrado buenos resultados, aunque se evidencian casos en los que no se tuvo éxito o no se lograron los resultados esperados debido a un mal diseño y al descuido hacia el público al que este iba a ser presentado; en respuesta, esto llevó a una mala comprensión por parte de los participantes y a una colaboración nula en términos de motivación [30].

Gamification es un proceso por el cual el usuario puede sentirse en la presencia de un ambiente de juego, resolviendo o desarrollando actividades del mundo real [31]. El uso de técnicas de *Gamification* se ha desarrollado con éxito en diversos temas

como ambientes de oficina, e-learning, sector educativo, sector bancario, deportes e idiomas por mencionar algunos [32]. Se aplicarán los conceptos de *Gamification* en el sector educativo, específicamente en el apoyo del aprendizaje de la programación en cursos introductorios.

Se ha implementado con éxito G*amification* mediante la integración de insignias, puntos, y tablas de clasificación, aunque no son las únicas técnicas con las que se puede hacer uso de *Gamification* [33]. Estas técnicas se implementan con el fin de mejorar la motivación en el usuario, pues la idea no es colocar un sistema de puntos y permitir que el usuario los gane sin sentido.

Con el uso de *Gamification* se quiere apoyar la motivación del aprendizaje en el estudiante ya que se ha encontrado que la falta de motivación en los estudiantes conlleva a bajas notas [19], el retiro de la materia o deserción en las carreras. Algunas causas de estos problemas se relacionan con las deficiencias en la metodología que se ha venido desarrollando para la enseñanza de esta ciencia, tales como la fácil frustración de los estudiantes, la falta de motivación y la falsa percepción respecto a la dificultad de los problemas [4].

En el proceso de *Gamification s*e pretende aumentar la motivación intrínseca y extrínseca del usuario. La primera puede ser entendida como una motivación interior por lograr una satisfacción personal y aumentar su autoestima. Por otro lado, la motivación extrínseca es el estímulo que se le brinda a una persona para realizar determinada acción [34]. En el proceso de aprendizaje se ha observado que sin motivación los estudiantes presentan fallas en la calidad del aprendizaje [35], por lo que obligar a los estudiantes a realizar ciertas actividades o tareas para la obtención de una nota disminuye su motivación intrínseca, la cual es la que se pretende aumentar con el uso de *Gamification.*

Si se suministra un ambiente controlado para el aprendizaje, pero se deja que el estudiante tenga *libertad* en el desarrollo de las actividades del sistema, se puede mejorar la motivación interna, es decir la motivación intrínseca. Lo anterior permite que el estudiante compita consigo mismo por la autosatisfacción de lograr dar solu-

ción a un problema, conseguir más puntos o poder avanzar más en el mapa para desafiar al sistema y los nuevos retos [36]. Estos resultados se pueden lograr debido a los beneficios que se obtienen del uso de *Gamification* que son: la retroalimentación adecuada y oportuna, la tolerancia al fracaso y una mejor estructura de los objetivos de la clase y de los problemas a resolver [37].

Para mejorar la motivación extrínseca se usan técnicas de *Gamification* como la competencia visualizada en las tablas de clasificación por la obtención de puntos e insignias, y el uso de una comunidad donde el usuario puede mostrar los resultados obtenidos, dado que el ser humano es un ser social y lo que suceda en la sociedad lo afecta directamente [38]. La motivación extrínseca aumenta en el estudiante en el momento de publicar, demostrar sus logros obtenidos y formar competencia con los otros estudiantes de su curso, incluso la competencia entre cursos u otras entidades. Es importante resaltar que se debe manejar bien esta técnica, debido a que la comparación entre usuarios de niveles muy disparejos puede resultar en una colaboración negativa en la motivación del estudiante, dado que los estudiantes que se encuentren en niveles inferiores al ser comparados con otros estudiantes de niveles muy superiores se vean frustrados al intentar alcanzarlos.

Se deberán estructurar muy bien las técnicas a ser utilizadas, para que el sistema pueda dar una motivación duradera y suficiente, debido a que los cursos introductorios de programación tienen un periodo de duración de 16 semanas, se espera que el estudiante haga uso del sistema por la duración de uno o más temas completos, donde se puedan observar resultados de mejoras en el aprendizaje de la programación.

Marco Contextual

Actualmente existen varios trabajos que intentan hacer uso de las ventajas de *Gamification* en distintas áreas. Uno de los campos en el que más se ha intentado motivar a los usuarios es en los ejercicios personales. Aplicaciones como Nike+, que es usada por más de 11 millones de usuarios, usa un sistema de puntos, retos y tablas de posiciones para motivar a sus usuarios [37]. Otro ejemplo en el área del ejercicio es Fitocracy, que usa el sistema de medallas con un componente social para hacer que sus usuarios estén en forma [39].También se utiliza en tareas cotidianas, en donde lo que se busca es que el usuario mejore su productividad en alguna forma, un ejemplo es CARROT, una aplicación de tareas por hacer, donde se cuenta una historia a partir de un personaje para que la persona cumpla a tiempo sus tareas [37].

En el ámbito de mejorar la productividad, también están los sistemas que capacitan al usuario en algún software o servicio. Compañías como Adobe y AutoCAD, han invertido en programas de este tipo, creando cursos que motivan al usuario a continuar aprendiendo y a desarrollar los talleres propuestos en programas como AutoCAD y Photoshop [39].

En el ámbito en el que *Gamification* más ha probado su utilidad es en el de la educación, funcionando tanto en pequeños softwares usados en colegios y universidades como sistemas a gran escala utilizados por millones de usuarios. Ejemplos de pequeños programas usados en colegios, se pueden encontrar para la enseñanza de materias como biología, matemáticas, etc., encontrando una mejora significativa en las calificaciones de los estudiantes. Un ejemplo se puede encontrar en el software McClean y Moreno que enseña biología y programación por medio de un videojuego, obteniendo un incremento en las notas de 30% y 19% respectivamente [39].

Probablemente los mayores exponentes de la educación por medio de *Gamification* son *Code School*, un sistema que enseña a programar virtualmente, haciendo uso

de elementos como puntos y medallas para medir el progreso [39] y *Khan Academy*, "un sistema que gamificó la educación en áreas incluyendo matemáticas, ciencias, historia y economía" [37], usando elementos como medallas y misiones para mantener motivado al usuario [37]. Una de las grandes ventajas *de Khan Academy* es que, dado que funciona con videos, es totalmente personalizado y los estudiantes pueden seguir su propio ritmo, algo que ha cambiado la forma de enseñar tanto dentro como fuera del salón [40], y ha atraído más de 10 millones de visitantes mensualmente [13]. Aunque estas páginas son similares al proyecto descrito, tienen unas diferencias importantes. Khan Academy no tiene un componente fuerte en programación, solo tiene cursos básicos, adicionalmente la idea de esta página, es proporcionar un tema auto contenido, es decir no es una herramienta que apoya una clase sino es la clase como tal. Por su parte, CodeSchool, es pago. La mayor diferencia es que estos sistemas funcionan con auto aprendizaje, es decir el estudiante debe aprender sin ayuda externa al sistema. El sistema descrito en este documento pretende ser una herramienta de apoyo a las clases de programación, dictadas por instituciones.

A continuación se analizarán los trabajos y aplicaciones más importantes en el área que estén cercanamente relacionadas con el uso de *Gamification* y la enseñanza de la programación donde se dará una descripción del programa o herramienta y las características diferenciadoras respecto a sus competidores.

De los principales trabajos relacionados se puede agrupar la mayoría dentro de los denominados MOOCs (Massive Online Open Courses). Dentro de este se analizarán los casos de Udacity, Coursera, Khan Academy y Code school; las siguientes herramientas son UVa, Spoj y Codechef los cuales son jueces virtuales, y por último Codility, Code fights y LeetCode, las cuales son útiles al momento de practicar, ejercitar y retar los conocimientos que se tiene en temas de programación. Enseguida se analiza cada una de las herramientas con más detalle desde el punto de vista del apoyo que se le brinda al usuario y las herramientas con las cuales se cuenta para poder desarrollar con éxito sus actividades, enfocándose en las ventajas y las desventajas que cada una de estas herramientas posee, para ser consideradas al mo-

mento de desarrollar una aplicación para la enseñanza de la programación haciendo uso de técnicas de *Gamification.*

MOOCs

Los Massive Online Open Courses (Cursos en línea masivos y abiertos) son cursos en línea destinados a la participación ilimitada y acceso abierto a través de la web. Gracias al uso de elementos multimedia como son los vídeos, lecturas y cuestionarios, los MOOC proporcionan un ambiente en el cual los estudiantes interactúan con los cursos en los que se inscriben de acuerdo a su ritmo e interés [16][41].

Ventajas

- Dado que son cursos virtuales, por medio de videos o lecturas que se pueden repetir, cada estudiante puede acceder al curso en el momento que lo desee, permitiendo que cada uno tenga su propio ritmo de aprendizaje.
- El uso de *Gamification* permite al estudiante desarrollar el curso sin perder el interés, y con motivación para terminarlo, a la vez que realizan tareas adicionales a las necesarias para aprobar el nivel.
- Foros de usuarios interactivos que ayudan a construir una comunidad para los estudiantes y profesores.

Desventajas

- Pretende reemplazar al profesor, cubriendo el contenido completo de la clase.
- Baja tasa de completitud del curso, alta deserción por parte de los estudiantes [16][42].

Udacity: Es un sitio de educación virtual con ánimo de lucro, en el cual se ofrecen cursos virtuales que pueden ser pagos o gratuitos. Los cursos comprenden temas de tecnología, ciencias de la computación, desarrollo web y de aplicaciones, así como cursos no tecnológicos que desarrollan temas relacionados con la economía y las matemáticas [43].

El flujo de ejecución consiste en que el estudiante escoge un tema y un curso de acuerdo a su nivel, luego comienza con videos introductorios y posteriormente se pone a prueba con algunos cuestionarios o se le da un espacio para que pueda ingresar la solución. Luego de una serie de niveles pasados en el curso, se realizan pequeñas pruebas donde se evalúan los conocimientos adquiridos por el usuario para poder seguir pasando de nivel. Al final se realiza el último examen, donde se certificará si el usuario realizó satisfactoriamente el curso en su totalidad, evaluando los conocimientos, experiencia y habilidades adquiridas durante el mismo.

Ventajas:

- Cursos separados en niveles: principiante, intermedio y avanzado.
- Posibilidad de pagar por un asesor personal.

Desventajas / Características no cubiertas:

- Se muestra toda la solución de un problema luego del video explicativo, más no dan pistas o ayudas al momento de realizar los problemas.

Khan Academy: Es un sitio de educación virtual, que contiene tanto videos como tareas interactivas en diferentes áreas. Dentro de estas áreas se encuentran matemáticas, ciencias de la computación, ciencias naturales, economía, entre otras [44]. Una de las principales herramientas utilizadas para mantener a los estudiantes interesados, es *Gamification*. Ha tenido un gran crecimiento, con más de 10 millones de usuarios actualmente [45].

El flujo de ejecución consiste en que el estudiante escoge una materia y un nivel, aunque en algunas materias se hace un examen inicial, para determinar mejor el nivel del estudiante. Una vez se tiene claro el nivel, usualmente se inicia con un video explicando un nuevo tema, aunque este video puede ser reemplazado por un escrito o un video interactivo dependiendo del área. Una vez recibida la información, se procede a la parte práctica, en donde se deben responder algunas preguntas o resolver ejercicios, para avanzar al siguiente nivel o tema. A medida que los estudiantes resuelven los problemas, ganan una medalla o premios virtuales que pue-

den compartir a través de redes sociales [46]. Estas medallas también son entregadas por tareas adicionales como hacer una pregunta en el foro, responder en el foro, adicionar información al perfil, etc. [47].

Ventajas:

- Se ha creado una comunidad creciente, que permite adicionar contenido rápidamente, al igual que traducirlo a otros idiomas.
- Existe una comunidad entre las personas que toman los cursos, para ayudar a resolver las preguntas sobre temas que no son cubiertos en su totalidad en los videos.
- Completamente gratuito.

Desventajas / Características no cubiertas:

- El contenido en programación se enfoca en diseño web, como actividades basadas en HTML y JavaScript. También tiene contenido sobre ciencias de la programación, en el que enseñan algoritmos básicos y criptografía.
- Aunque existe un módulo para profesores, no es fácil adaptar los contenidos a las clases en los colegios o universidades, dado que los ejercicios o tareas están totalmente basados en los videos o lecturas.

Coursera: Es un sitio de educación virtual, que por medio de alianzas con universidades de todo el mundo, provee cursos virtuales gratuitos.

La metodología del curso varía dependiendo de la institución, aunque en general presentan el mismo formato con videos de cátedras semanales que el estudiante sigue a su propio ritmo y adicionalmente puede hacer las lecturas. Al final de un conjunto de videos, pueden hacer ejercicios de los temas más prácticos, escritos que son calificados por las otras personas inscritas en el curso, o exámenes que usualmente son calificados automáticamente [48].

El flujo de ejecución consiste en que la institución anuncia el curso en la plataforma Coursera y las personas interesadas se inscriben. Una vez comienza el curso se

anuncian diferentes módulos, con una fecha de liberación y una fecha máxima de entrega para los trabajos individuales. Al mismo tiempo se inician foros en el que los estudiantes pueden discutir las tareas y los videos, y en algunos casos son moderados por personas de la universidad. Dependiendo de la calidad de las entregas de los trabajos y la fecha de entrega, se les asigna una calificación, en algunos casos, esta calificación puede llevar a un certificado otorgado por la universidad.

Ventajas:

- Dado que los contenidos no son creados por Coursera, hay una gran variedad de temas disponibles, al igual que diferentes formas de enseñanza.
- Es un servicio gratuito, en algunos cursos es necesario el pago de la verificación de identidad para recibir el certificado, pero en la mayoría de los cursos no es necesario pagar para recibir el contenido.
- Existe una comunidad dentro de cada curso, que ayuda a los estudiantes a avanzar más allá de lo dictado por la universidad.
- Permite el hecho de recibir un certificado de cumplimiento de algunas de las universidades más prestigiosas del mundo.
- Los cursos son dictados, por algunas de las personas más destacadas en los campos.

Desventajas / Características no cubiertas

- El hecho de que cada curso sea dictado por una universidad diferente, crea un sistema desestandarizado, en el que el estudiante tiene que aprender una nueva metodología por cada curso que toma.
- No está pensado para apoyar o complementar un proceso de aprendizaje. La idea se basa en reemplazar el proceso tradicional de aprendizaje, por lo que es difícil adaptarlo a un curso universitario.
- El hecho de que los cursos tengan fechas estrictas, no permite que los estudiantes se inscriban cuando necesiten o les convenga, sino que deben esperar al momento adecuado para comenzar el curso.

Code School: Es un sitio de educación virtual destinado al aprendizaje en línea para desarrolladores a través de contenidos multimedia y de entretenimiento, usando técnicas de juegos para el desarrollo del curso [49].

El flujo de ejecución consiste en escoger un curso de los ofrecidos en la plataforma, para luego comenzar con los videos introductorios, que luego son reemplazados por actividades en donde se desafía el conocimiento y el aprendizaje del estudiante mostrando los resultados en tiempo real al instante que el estudiante envía su respuesta. Una vez completada la actividad o desafío, se felicita o se premia al estudiante otorgando puntos o medallas las cuales puede compartir en su perfil y en distintas redes sociales [50].

Ventajas:

- Cursos especificados en tecnologías web y de programación.
- Blog de discusión para debatir problemas o ideas obtenidas durante el curso.
- Posibilidad de escoger caminos, los cuales permiten desarrollar habilidades específicas.

Desventajas / Características no cubiertas:

- No todos los cursos son gratuitos, hay que pagar para tomar cursos más avanzados.

Jueces Virtuales

Son sistemas en línea para practicar y probar programas en concursos de programación. Muchos de estos sistemas permiten organizar sus propios concursos. El sistema puede compilar y ejecutar código, y prueba las soluciones con los datos preconstruidos. Se pueden juzgar los códigos por restricciones, límite de tiempo, límite de memoria, restricción de seguridad, entre otras. La salida del código se compara con la salida esperada, dando así el resultado de la solución enviada por el programador [51].

UVa: Es un juez automatizado en línea para problemas de programación, creado por la Universidad de Valladolid. Su archivo cuenta con más de 4.300 problemas los cuales son usados en concursos de programación, ofreciendo al usuario poder subir su solución en diferentes tipos de lenguajes y también ofrece la posibilidad de crear concursos de programación mostrando estadísticas y tablero de puntuaciones en vivo [52].

La metodología del uso de esta herramienta consiste en que el usuario se registra en el sistema, luego busca y elige el tipo de problema que desea realizar; se muestra el problema a solucionar, indicando cuáles son las entradas y las salidas esperadas para tener mayor claridad al momento de solucionar el problema. El usuario elige el lenguaje de programación en el cual desarrollará la solución, y puede realizarla en el espacio que otorga la herramienta para la escritura del código o en su propio IDE. Posteriormente, el juez virtual evaluará la solución y mostrará en pantalla si fue aceptado o no, junto con el motivo por el cual no se aceptó la solución.

Ventajas:

- Gran variedad de problemas que desafían al usuario.
- Poder crear problemas para concursos de programación.
- Problemas tomados de concursos de programación.
- Variedad de lenguajes.
- El desarrollo se puede hacer dentro de la herramienta, y es compilado por sus servidores, sin necesidad de algún programa adicional.
- Se muestra si la solución está en cola o por qué no fue aceptada la solución (respuesta incorrecta, límite de tiempo excedido, error de presentación, error en compilación, error en ejecución, límite de memoria excedido, función restringida), lo cual le da pistas al usuario para identificar en dónde pudo haber fallado.
- Se pueden crear concursos o competencias con las propias reglas del usuario.
- Muestra estadísticas del usuario y sus soluciones.

Desventajas / Características no cubiertas:

- El usuario no puede encontrar con facilidad en dónde ocurrió el error.
- Se evalúa la totalidad del programa y no habilidades específicas.
- Se hace uso de un ranking mas no se preocupa por la motivación del usuario; no es propiamente *Gamification*.
- No ofrece al usuario un camino por dónde comenzar a resolver problemas.

Codechef: Es una comunidad virtual de programadores, en donde se alojan competencias de programación. La persona que ingresa a la página puede resolver un problema aislado o hacer parte de una competencia en donde se alojan varios problemas. Para resolver los problemas se puede hacer uso de varios lenguajes de programación, y cada problema tiene unos casos de prueba predeterminados al igual que el tiempo máximo en el que debe terminar la ejecución del programa [53].

Ventajas:

- Es completamente gratuito.
- Contiene una gran cantidad de problemas, dado que personas ajenas a la organización pueden agregar contenido al sitio.
- Es posible participar en foros que permiten la interacción de la comunidad.
- Tiene un editor de código en línea, lo que facilita los envíos.

Desventajas / Características no cubiertas:

- No fue diseñado para que los usuarios aprendan por medio de la herramienta, solamente ejecuta las competencias.
- La forma en la que califica las soluciones es binaria, es decir el problema está bien o mal, no hay estados intermedios. Aunque dice la razón por la cual no fue aprobada la solución, esto puede llevar a desmotivación de los participantes.
- Dado que gran cantidad del contenido es subido por personas independientes, no se garantiza que sean correctos.

- Aunque se pueden crear competencias dentro de una clase, no fue pensado para este fin por lo que puede presentar dificultades para el profesor y los estudiantes.

Spoj: Es un juez virtual, que evalúa automáticamente los programas enviados por sus usuarios registrados. Con más de 13 mil tareas disponibles y soporte en 45 lenguajes de programación, Spoj es una herramienta útil para el autoaprendizaje [54].

La metodología del uso de esta herramienta es la siguiente: el usuario se registra en el sistema, luego de lo cual puede escoger el tipo de problema que desea realizar; se muestra el problema a solucionar, indicando cuales son las entradas y las salidas esperadas para tener mayor claridad al momento de solucionar el problema. El usuario elige el lenguaje de programación que le sea de mayor facilidad para desarrollar la solución, y puede realizarla en el espacio que otorga la herramienta para la escritura del código o en su propio IDE. Posteriormente, el juez virtual evaluará la solución y mostrará el resultado en pantalla si fue aceptado o no, mostrando el motivo por el cual no se aceptó la solución.

Ventajas:

- Gran variedad de problemas que desafían al usuario.
- Problemas tomados de concursos de programación.
- Variedad de lenguajes.
- El desarrollo se puede hacer dentro de la herramienta, y es compilado por sus servidores, sin necesidad de algún programa adicional.
- Se muestra por qué no fue aceptada la solución (respuesta incorrecta, límite de tiempo excedido, error en compilación, error en ejecución), lo cual le da pistas al usuario para identificar en dónde pudo haber fallado.
- Se pueden crear concursos o competencias con las propias reglas del usuario.

Desventajas / Características no cubiertas:

- El usuario no puede encontrar con facilidad en dónde ocurrió el error.

- Se evalúa la totalidad del programa y no habilidades específicas.
- Se hace uso de un ranking mas no se preocupa por la motivación del usuario; no es propiamente *Gamification*.

Herramientas de práctica con conocimientos previos en programación

CodeFights: Es una herramienta en la cual los programadores pueden participar en juegos y competencias, donde deben solucionar un problema en el menor tiempo posible, practicando contra la máquina o entre otros jugadores [55].

El flujo de ejecución consiste en que el programador escoge el lenguaje de programación en el cual desea realizar la batalla. La pelea consiste de 3 rounds, en cada uno de ellos deberá encontrar el error en el código que debería dar la solución al problema, el cual se detalla en las entradas y las salidas correspondientes [56].

Ventajas:

- Uso de *Gamification* para mantener a los usuarios interesados en seguir compitiendo y mejorando sus habilidades.
- Lenguajes de programación más populares (C++, Java, JavaScript, Python).

Desventajas / Características no cubiertas:

- Se requieren conocimientos de programación previos para poder hacer uso de la herramienta.
- Solo se concentra en el reconocimiento de errores.
- El tiempo en que se encuentra el error en el código es el determinante de quién es el ganador, mientras que si hay una solución más eficiente no se tiene en cuenta.

Codility: Es una compañía basada en la contratación de ingenieros de software. La forma en la que realizan las pruebas para programación, consiste en un problema específico y la persona a ser entrevistada puede escoger en qué lenguaje lo desea resolver. Las pruebas dependen de qué habilidad pretende ser evaluada, pero en

general miden las habilidades algorítmicas, y conocimientos de algoritmos y estructuras de datos. La persona puede solicitar algunas ayudas durante la prueba e incluso buscar ayuda en internet. Una vez completado el ejercicio, puede probarlo con algunos datos de ejemplo que contienen la solución, o puede probarlo con casos diferentes y el sistema le dará el resultado, mas no le dirá si es correcto o no. Una vez completada la actividad el sistema realizará varias pruebas independientes para probar elementos como eficiencia y efectividad. Evaluará casos comunes así como casos difíciles para el programa, y estimará la complejidad total del algoritmo propuesto. Dependiendo del resultado de cada prueba, asignará un puntaje total para el algoritmo [57].

Ventajas:

- Tiene un muy buen sistema de calificación, basándose en los elementos que un estudiante de programación debería tener en cuenta en su diseño.
- Hace pruebas individuales lo que permite probar la ejecución en conceptos diferentes, obteniendo más información del candidato, que si se probara el programa como un todo, como lo hacen la mayoría de los sistemas.
- Los ejercicios propuestos, pretenden evaluar una habilidad específica, en un tiempo corto (no más de media hora).
- Permite una gran selección de lenguajes de programación.
- El desarrollo se hace dentro de la herramienta, y es compilado por sus servidores, no se requieren programas adicionales.

Desventajas / Características no cubiertas:

- No es pensado para educación, la idea es medir el conocimiento de una persona después de haber recibido un curso.
- Es una herramienta paga y no está diseñada para ser usada por instituciones educativas sino por compañías, aunque sería muy fácilmente adaptable a su uso en educación.
- No utiliza técnicas de *Gamification*, dado que cada persona tiene un objetivo inmediato al realizar una prueba.

- No utiliza herramientas para trabajo en comunidad.

LeetCode: Es una plataforma para la preparación de entrevistas de técnicas de codificación[58]. Está compuesta por una biblioteca de más de 190 preguntas, las cuales han sido utilizadas previamente en entrevistas por algunas empresas, dependiendo si el usuario ha pagado por una suscripción tiene diferentes privilegios, como acceso a preguntas más difíciles o saber que empresas utilizaron esa pregunta y acceso a cursos donde se explica cómo resolver cierto tipo de problemas con mayor dificultad. También cuentan con algunos cursos, en los que se trata un tema en específico, donde publican una breve introducción en el tema y ejemplos de código con los cuales se puede resolver el problema planteado.

El flujo de ejecución consiste en que el usuario escoge un tipo de problema o pregunta de la biblioteca de preguntas de la plataforma, escoge el lenguaje el cual quiere realizar su solución, la implementa y envía su solución, la plataforma le devuelve el resultado de su solución, indicando si esta fue aceptada, si tiene problemas de compilación, de ejecución o de límite de tiempo.

Ventajas:

- Gran cantidad de preguntas almacenadas.
- Uso de pistas, las cuales no resuelven el problema por sí solas, pero dan una ayuda al usuario en cuanto al enfoque que debería darle a la solución del problema.
- Foros de discusión sobre una pregunta.
- Estadísticas de las soluciones enviadas por el usuario.
- Mide el progreso del usuario en la plataforma.
- Problemas categorizados por niveles: Fácil, Medio y Difícil.
- Indica al usuario el total de casos con los que será probada su solución y en caso de dar una respuesta errónea en la solución indica el número del caso en el que su solución no fue válida, la entrada y la salida de la solución del usuario y la salida esperada por ese caso.

Desventajas / Características no cubiertas:

- No está orientada a la enseñanza de la programación, sino centrada en cómo resolver problemas en específico.
- No utiliza técnicas de *Gamification* para motivar al usuario a resolver más problemas.
- No se indica un camino de problemas a seguir para el usuario.
- No hace uso de videos interactivos en sus cursos.

Luego de analizar los ambientes y escenarios de trabajo para el apoyo del estudiante al momento de aprender y/o practicar temas relacionados con la programación, se tuvieron en cuenta cuáles son las características de cada una de las herramientas basadas en sus ventajas y se implementaron las nuevas funcionalidades fundamentadas considerando sus desventajas o características no cubiertas. El sistema Game-ludex utiliza técnicas de *Gamification* como: tableros de puntuación, logros, medallas y escogencia de un avatar. Todo esto también llevado a las redes sociales donde puedan compartir sus logros, sugerencias y experiencias para poder mantener la comunidad atenta y activa.

Ventajas a tener en cuenta a partir de las herramientas MOOCs	• Uso de videos y lecturas para el repaso de los estudiantes. • Foros de discusión para la ayuda de solución de problemas. • Problemas clasificados por nivel de dificultad. • Contenido de los cursos enfocados a programación. • Curso dictado por los mismos profesores de la Universidad.
Desventajas o características a cubrir de los MOOCs	• El profesor es quien dicta y orienta el curso. • El profesor estará pendiente del

	aprendizaje y entendimiento de cada estudiante del curso. • No se mostrará la solución completa, se otorgarán pistas para que el estudiante pueda desarrollar y encontrar la solución.

Tabla 1. Ventajas y Desventajas a cubrir de los MOOCs

Ventajas a tener en cuenta a partir de los jueces virtuales	• Tener un editor en línea para la facilidad de envíos. • Solo el profesor o el monitor podrán subir los problemas, verificando que estén bien. • Poder crear competencias dentro de un curso con reglas propias del usuario. • Se mostrará al estudiante si fue aceptada la solución o no, y por qué no fue aceptada (respuesta incorrecta, límite de tiempo excedido, error en compilación, error en ejecución).
Desventajas o características a cubrir de los jueces virtuales	Además de mostrar si fue aceptada la solución o no, también se dará retroalimentación más específica al usuario; se mostrará si la cantidad de ciclos son correctos, o si los límites al recorrer una estructura son los apropiados.

Tabla 2. Ventajas y Desventajas a cubrir de los Jueces Virtuales

Ventajas a tener en cuenta a partir de Codility	• Hacer pruebas individuales y no como un todo, para poder probar la ejecución en conceptos diferentes, obteniendo más información del estudiante. • Los ejercicios para evaluar una habilidad específica, en un tiempo corto (no más de media hora).
Desventajas o características a cubrir de Codility	• Adaptar las características de la herramienta para uso en la educación. • Uso de foros y blogs de discusión.

Tabla 3. Ventajas y Desventajas a cubrir de Codility

Ventajas a tener en cuenta a partir de Codefights	• Creación de competencias entre los estudiantes. • Diseño de ejercicios enfocados en encontrar el error en el menor tiempo posible.
Desventajas o características a cubrir de Codefights	• La efectividad y eficacia de la solución del problema también podrá ser determinante para definir el ganador.

Tabla 4. Ventajas y Desventajas a cubrir de CodeFights

Ventajas a tener en cuenta a partir de LeetCode	• Ayudas que contengan ejemplos de cómo solucionar un tipo de problema. • Categorización de problemas por su nivel de dificultad: Fácil, Medio, Difícil.
Desventajas o características a cubrir de LeetCode	• Estructurar un camino o ruta indicada sobre los problemas a resolver. • Agregar pistas a los problemas.

Tabla 5. Ventajas y Desventajas a cubrir de LeetCode

III – ANÁLISIS

Requerimientos

El proceso de requerimientos empleado para construir el presente documento se explica a lo largo de las siguientes secciones. Dicho proceso consta de 2 fases principales: recolección de información y análisis e identificación de requerimientos.

El listado de Requerimientos completos y sus especificaciones, se puede encontrar en el documento de requerimientos (Ver anexo Requerimientos), una hoja de cálculo en donde se representa cada uno de ellos junto a sus especificaciones.

Recolección de información

Esta fase se preocupa por identificar las fuentes de información en que se encuentren relacionados al tema de *Gamification* y la motivación de los estudiantes para el aprendizaje.

Para esta recolección de información se tuvo en cuenta como fuente principal el trabajo de grado titulado "Modelo de la programación utilizando *Gamification*" [17], y como fuentes alternas las siguientes:

- Análisis de plataformas de e-learning como los MOOC's.
- Análisis de Jueces Virtuales.
- Análisis de herramientas para practicar temas relacionados a la programación.

Análisis e identificación de requerimientos

A lo largo de esta fase del proceso de requerimientos se empleó la información obtenida durante la Recolección de información para identificar los requerimientos del sistema tanto funcionales como no funcionales. Esto tuvo como consecuencia la definición más precisa de los límites del producto, una separación puntual de lo que el software hará y lo que no hará.

Identificación de requerimientos

La Identificación de requerimientos tiene como objetivo convertir la información obtenida durante la recolección de información, en requerimientos, y describirlos de acuerdo alcance del producto.

Respecto a los temas de información basados en el trabajo de investigación mencionado anteriormente, se tuvieron en cuenta los siguientes aspectos importantes en cuanto a especificar como requerimientos, para ser implementados en el prototipo:

- **Componentes de la Plataforma**
  - **Módulo de Retos**
    - Personajes
    - Retos
    - Juzgamiento
    - Búsquedas
  - **Módulo de Administración**
    - Gestión de Contenido
    - Gestión de Usuarios
    - Comunidad
    - Estadísticas
  - **Módulo de Jugador**
    - Perfil
    - Habilidades

Especificación de requerimientos

Esta sección complementa la lista general, relacionando a cada requerimiento con los siguientes atributos:

Requerimientos Funcionales: ID, Prioridad, Nombre, Descripción y Estado.

Requerimientos No Funcionales: ID, Prioridad, Tipo, Nombre, Descripción y Estado.

Especificando cada atributo para cada uno de los requerimientos.

Prioridad del Requerimiento

Los requerimientos fueron priorizados con base en el siguiente mecanismo y empleando la hoja Priorización de requerimientos:

1. Se asignó a cada requerimiento un número del 1 al 9 que indica el beneficio que el requerimiento implica para los usuarios, donde 1 indica el mínimo beneficio y 9 el máximo.
2. Se asignó a cada requerimiento un número del 1 al 9 que indica el impacto o la pérdida de valor que implica la no implementación del requerimiento, donde 1 indica el impacto más bajo y 9 el más alto.
3. Se asignó a cada requerimiento un número entre 1 y 9 que indica el esfuerzo de implementar el requerimiento, donde 1 es el mínimo y 9 el máximo.

Tras conocer estos parámetros, se obtienen los siguientes valores:

$$\text{Valor Absoluto} = \frac{\text{beneficio} \times \text{peso beneficio} \ + \ \text{impacto} \times \text{peso impacto}}{\text{peso beneficio} \ + \ \text{peso impacto}}$$

Donde peso beneficio=2 y peso impacto=1. Esto mide el valor de un requerimiento como un promedio ponderado de su beneficio y su impacto. El beneficio es el doble del impacto, para dar prioridad a los beneficios que se puedan aportar al prototipo.

$$\text{Valor relativo} = \frac{\text{Valor absoluto}}{\text{Suma Valores absolutos}} \times 100$$

Indica el valor del requerimiento como un porcentaje del valor de todos los demás.

$$Esfuerzo\ Relativo = \frac{\text{Esfuerzo}}{Suma\ \ esfuerzos} \times 100$$

Indica el esfuerzo del requerimiento como un porcentaje del esfuerzo de todos los demás.

$$Prioridad = \frac{\text{Valor relativo}}{Costo\ relativo}$$

La prioridad será la razón entre el valor relativo y el costo relativo.

El resultado de esta parte se puede detallar en el documento de requerimientos (Ver Anexo Requerimientos).

Según el valor del resultado de la prioridad, se realizó una clasificación de la siguiente manera:

Prioridad	**Descripción**
Alta	Requerimiento indispensable.
Media	Requerimiento no es indispensable, pero brinda beneficios importantes y el nivel de impacto negativo por su ausencia puede crear insatisfacción por parte de los usuarios.
Baja	Requerimiento no es indispensable.

Tabla 6. Prioridades de los Requerimientos

Durante la Especificación de requerimientos se completaron las columnas de importancia que están en la hoja Priorización de requerimientos.

Estado del requerimiento

El estado del requerimiento es una medida del grado de terminación del requerimiento. Puede tener un valor entre los siguientes:

- Diseñado: Indica que el requerimiento tiene una especificación completa y se consideró dentro de la fase del diseño del prototipo; sin embargo, no está implementado.
- Implementado: indica que el requerimiento tiene una especificación completa, se desarrolló y probó en el prototipo.

Estado	Descripción
Diseñado	Requerimiento en una etapa de diseño, sin implementación
Implementado	Requerimiento implementado y probado

Tabla 7. Estados de los Requerimientos

Tipo de requerimiento

Para los requerimientos no funcionales se debe especificar el tipo del requerimiento al que hacen parte. Los diferentes tipos son:

REQUERIMIENTOS NO FUNCIONALES	Usabilidad
	Disponibilidad
	Portabilidad
	Seguridad
	Operatividad
	Interoperabilidad
	Escalabilidad
	Concurrencia
	Mantenibilidad

Tabla 8. Tipos de Requerimientos no funcionales

Listado Requerimientos

A continuación se mostrará el listado general de los requerimientos obtenidos para el desarrollo del prototipo funcional.

Requerimientos Funcionales

ID	Prioridad	Nombre
RF-1	Media	Agregar consejos por estudiantes
RF-2	Media	Escoger Avatar

RF-3	Baja	Escoger personaje
RF-4	Baja	Asignar habilidades
RF-5	Media	Asignar niveles
RF-6	Media	Tableros puntuación
RF-7	Media	Compartir medallas
RF-8	Alta	Premiación con medallas
RF-9	Media	Apertura problemas
RF-10	Alta	Sistema puntuación
RF-11	Media	Seguridad contraseñas
RF-12	Media	Verificación código malicioso
RF-13	Media	Recuperación contraseña
RF-14	Media	Validación correo
RF-15	Alta	Ingreso datos cuenta
RF-16	Alta	Verificación número intentos
RF-17	Baja	Compartir problemas
RF-18	Baja	Guardar curso
RF-19	Alta	Clasificación ejercicios
RF-20	Alta	Registro medallas
RF-21	Alta	Generación de ayudas
RF-22	Baja	Adición lenguajes
RF-23	Media	Envió correos electrónicos
RF-24	Baja	Desactivar cuenta
RF-25	Alta	Consultar cuenta alumno
RF-26	Media	Comprobación Usuario y contraseña
RF-27	Media	Estructuración en niveles
RF-28	Alta	Definir ayudas
RF-29	Alta	Consultar historial envíos
RF-30	Alta	Creación problemas
RF-31	Alta	Enviar solución
RF-32	Media	Roles

RF-33	Media	Generación estadísticas
RF-34	Media	Tablas de posición
RF-35	Media	Generar puntajes
RF-36	Media	Guardar soluciones
RF-37	Baja	Medir complejidad
RF-38	Alta	Medición tiempo ejecución
RF-39	Media	Comparar archivos con solución
RF-40	Media	Generación archivo salida
RF-41	Alta	Cerrar sesión
RF-42	Alta	Consultar perfil alumno
RF-43	Alta	Consultar alumnos en el sistema
RF-44	Alta	Mostrar información intentos estudiante
RF-45	Alta	Creación curso
RF-46	Alta	Iniciar Sesión
RF-47	Media	Creación de Instituciones
RF-48	Alta	Creación de Clases

Tabla 9. Requerimientos Funcionales

Requerimientos No Funcionales

ID	Prioridad	Tipo	Nombre
RNF-1	Media	Concurrencia	Soportar 50 usuarios de plataforma web concurrentes
RNF-2	Media	Operatividad	Compilación C++
RNF-3	Alta	Portabilidad	Navegador Web
RNF-4	Media	Usabilidad	El usuario registrado ingresa al sistema, consulta y edita su perfil
RNF-5	Media	Robustez	Verificación de campos obligatorios
RNF-6	Alta	Seguridad	Seguridad de acceso al sistema

RNF-7	Media	Seguridad	Tiempo activo de la sesión del usuario
RNF-8	Media	Seguridad	Navegación en el sistema
RNF-9	Alta	Seguridad	Manejo de contraseñas

Tabla 10. Requerimientos No Funcionales

Plan de Pruebas

Para la validación del prototipo funcional, se escogió un curso de Programación de Computadores de la Carrera de Ingeniería de Sistemas de la Pontificia Universidad Javeriana. En este curso el estudiante debía profundizar en temas de memoria dinámica y almacenamiento de datos, así como asimilar los conceptos fundamentales sobre las estructuras de datos lineales básicas.

Luego de que los integrantes del curso hicieran uso de la herramienta, se realizaron encuestas para medir las funcionalidades del prototipo y el impacto generado sobre los integrantes del curso, tanto estudiantes como el monitor.

Características del curso:

- ID de la Clase: 1136
- Periodo: 201530
- Profesor: Fabio Antonio Avellaneda Pachón
- Monitor: Alfredo Sebastián Santamaría Gómez
- Número de estudiantes: 18
- Temas vistos antes del uso de la plataforma Game-Iudex:
    - Apuntadores
        - Definición y justificación de apuntadores.
        - Declaración e inicialización de variables tipo apuntador.
        - Aritmética de apuntadores.
        - Cómo pasar parámetros por referencia.

        - Relación entre apuntadores y arreglos.
    - Memoria dinámica
        - Manejo de memoria dinámica (new, delete).
    - Cadenas
        - Implementación de funciones para manejo de cadenas.
        - Funciones estándar de cadenas.
    - Recursión
    - TAD Lista, Pila y Cola
        - Definición.
        - Métodos.
        - Implementación.
        - Aplicaciones.
        - Introducción a Templates.
        - MultiListas.
- Temas restantes antes del uso de la herramienta Game-ludex:
    - Archivos secuenciales y de acceso aleatorio
        - Apertura y cierre de archivos.
        - Creación de archivos.
        - Procesamiento de archivos de texto.
        - Carácter por carácter.
        - Línea por línea.
    - Archivos binarios

Vale la pena aclarar que todos los temas vistos en clase estaban disponibles para su desarrollo en la herramienta Game-ludex, donde los estudiantes pusieron en práctica todos sus conocimientos y habilidades sobre cada uno de los temas propuestos.

Se hizo escogencia de este curso debido a que los estudiantes habían tenido experiencia previa en el uso de jueces virtuales como Boca [59], Spoj [54] y Discant [60], haciendo uso de estos en los diferentes laboratorios desarrollados durante el transcurso de la clase.

El objetivo de realizar las pruebas con este curso consistía en poder dar una aproximación de respuesta a la pregunta propuesta en el trabajo de grado titulado "Modelo de la programación utilizando *Gamification*" [17]:

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

Para que los estudiantes pudieran tener mayor información a la hora de contestar la encuesta, se decidió hacer uso de la plataforma Game-ludex, desde la semana catorce de clases, donde los estudiantes usaron está herramienta para la realización de los laboratorios y talleres de la clase.

Desde la semana catorce y durante las siguientes semanas de clase, se acompañó al curso en el uso de la plataforma para obtener una mayor retroalimentación de cómo se estaba haciendo uso de la herramienta, observar diferentes reacciones por parte de los estudiantes, el monitor y el profesor frente a distintos eventos y también posibles cambios y mejoras propuestas por los integrantes de la clase. Dichas propuestas podían ser escritas como comentarios en la plataforma o expuestas durante el desarrollo de la clase por los mismos estudiantes, el monitor o el profesor. Estas propuestas se pasaban a issues, los cuales se implementaban para la siguiente sesión de la clase, donde los usuarios pudieran hacer uso de las mejoras o adiciones de las nuevas funcionalidades que habían propuesto para la plataforma.

Validación a través de encuestas

Encuesta Estudiantes

El principal objetivo de la encuesta es poder medir el impacto generado por Game-ludex a la variable motivación del aprendizaje en el estudiante. Para esto se realizaron preguntas de acuerdo a la experiencia de uso del estudiante con los diferentes jueces virtuales, comparando las características y funcionalidades de las herramientas utilizadas, indicando cuáles fueron las que más y las que menos les gustaron e indicar con cuál de las herramientas tuvo una mejor experiencia.

Luego de esto, se compararon las diferentes técnicas de *gamification* utilizadas, con el fin de conocer cuál de las herramientas tuvo una mayor incidencia en cuanto a motivar al estudiante para resolver los problemas de la clase.

Las últimas preguntas corresponden a validar si la herramienta les fue útil y dar una aproximación en cuánto pudo Game-Iudex ayudar a la motivación del estudiante para completar los problemas subidos en la plataforma y también las funcionalidades y características que le gustarían que tuviera la plataforma.

Para ver con más detalle el formulario de la encuesta (Ver Anexo Formulario encuestas).

Encuesta al Monitor

Para analizar la utilidad de la herramienta desde un punto de vista diferente al de los estudiantes, se decidió realizar una encuesta al monitor de la clase, en la cual se hace énfasis frente a cómo ayudó la herramienta en cuanto a sus funciones como monitor durante el desarrollo del curso, un breve análisis en cuanto a la ventaja competitiva de Game-Iudex en cuanto a los diferentes jueces virtuales usados en la clase, si se notó una mejoría relacionado a la motivación en los estudiantes y por último un análisis sobre el uso de Game-Iudex como herramienta de apoyo en las distintas clases de programación de las carreras de ingeniería.

Para mayor ver con más detalle el formulario de la encuesta (Ver anexo Encuesta monitor)

IV – DISEÑO Y DESARROLLO DE LA SOLUCIÓN

Herramientas

La herramienta se concibió como una herramienta web principalmente debido a que en ocasiones no es viable para las instituciones educativas instalar una aplicación en los computadores de los laboratorios, así como la necesidad de interacción que debe existir entre los usuarios y un sistema central. Esto presenta la restricción que los usuarios deben estar conectados a internet siempre que hagan uso de la herramienta.

Para el desarrollo de la aplicación se tuvieron en cuenta dos de los lenguajes de programación más utilizados para el desarrollo de aplicaciones web: Ruby y PHP. Se eligió el lenguaje de programación PHP debido a que se tiene más experiencia y practica en su uso, también a la familiaridad en la sintaxis debido a que es muy parecida a otros lenguajes de programación como lo son Java y C++ en los cuales se tiene más conocimientos y práctica. Realizar la implementación en Ruby puede causar que la estimación de tiempo prevista para desarrollar la aplicación no tenga éxito, debido a que se tendría que emplear una considerable parte del tiempo para el aprendizaje del nuevo lenguaje de programación, lo cual está fuera del alance del proyecto [61].

Por otra parte, la cantidad de *frameworks* disponibles para PHP es considerablemente mayor que los existentes para Ruby, así como el tamaño de la comunidad de desarrolladores, lo que beneficia en la búsqueda de ayuda o respuestas a problemas que se presenten al momento de desarrollar la aplicación y que exista una mayor probabilidad de encontrar una respuesta o solución.

Para la decisión de qué framework usar, se evaluaron los más populares existentes para PHP actualmente, los cualesfueron: Yii, Symfony y Laravel. Dado que las tareas que debía realizar el sistema en su gran mayoría no son computacionalmente costosas, y las que sí lo son no son manejadas por PHP, junto con el corto tiempo que se tendría para desarrollar la aplicación, la decisión del framework se basó prin-

cipalmente en la facilidad de aprender y en los módulos integrados en el framework necesarios para el desarrollo de la plataforma. Para determinar esto, se realizó un tutorial de introducción a cada uno de los frameworks, así evaluando su complejidad. Después de esto, se llegó a la conclusión de usar el framework de Laravel, debido principalmente a su facilidad para aprender, la gran cantidad de ayudas online, la estructura de código, ayudas en temas de seguridad y el hecho de que está basado en el patrón modelo vista controlador(MVC).

Para el almacenamiento de datos, se decidió usar un motor de bases de datos relacional, utilizando MySQL como el sistema de gestión, debido a que está totalmente integrado con Laravel, a que es gratuito y de código libre y que es la base de datos de código libre más usada en el mundo [62].

Manejo de usuarios:

Para el manejo autenticación el sistema se apoya principalmente en el módulo de autenticación de Laravel, que incluye:

- Manejo de sesiones, que permite al usuario realizar diferentes acciones únicamente realizando el inicio de sesión una vez.
- Expiración de sesión, controlando un tiempo máximo que el usuario puede usar uso de la cuenta, sin necesidad de ingresar credenciales nuevamente.
- Cifrado de contraseñas utilizando el algoritmo AES256-CBC.
- Recuperación de contraseñas, permitiendo al usuario pedir un token único por correo electrónico permitiéndole cambiar la clave.
- Máximo número de intentos de inicio de sesión por un periodo de tiempo, evitando así ataques.

Adicionalmente a las funciones por defecto, se adicionó un campo extra para tener dos tipos de usuarios: administrador y usuario. El administrador tiene permisos para modificar la configuración general del sistema, además de poder actuar como un usuario normal. Mientras que el usuario no administrador es un usuario que tiene acceso al sistema con restricciones dependiendo de su rol.

Manejo de roles

Adicionalmente al manejo de administrador, se cuenta con diferentes roles para los usuarios. Estos roles están relacionados con las clases, es decir el mismo usuario puede tener varios roles en diferentes clases. Cada rol está asociado con un nivel de privilegio, y cada acción que se realiza en el sistema, requiere un cierto nivel de privilegio para poder ser realizada. Inicialmente el sistema tiene el rol de profesor y de estudiante, los cuales el administrador puede modificar, o crear más roles según sea requerido para cada clase. Al momento de crear una clase, el administrador debe indicar el usuario correspondiente al profesor de la clase, quien posteriormente podrá agregar a sus estudiantes o a otros usuarios con rol de profesor.

Estructura del sistema

Para entender mejor la estructura del sistema se presenta a continuación el diagrama relacional de la base de datos:

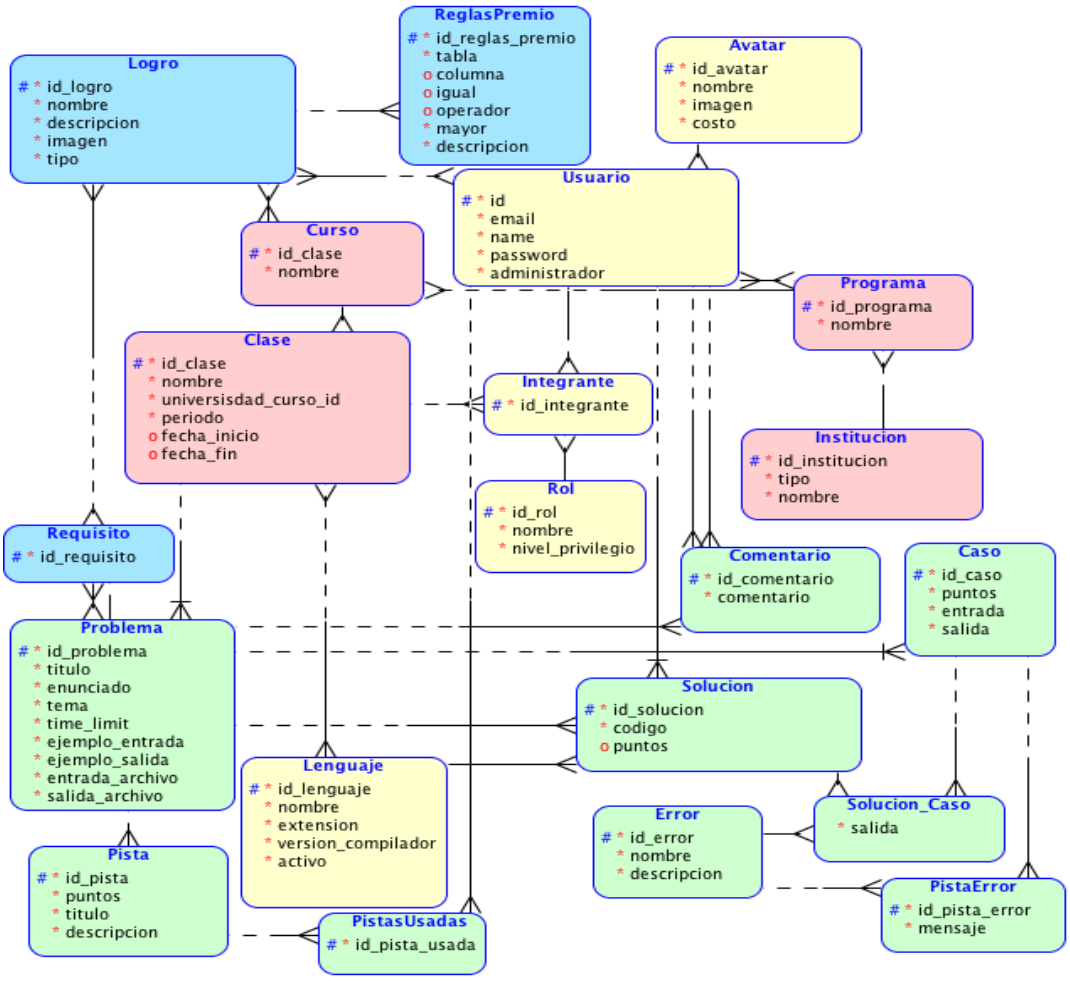


Ilustración 1. Diagrama relacional de base de datos

El sistema está dividido jerárquicamente, en donde el usuario de tipo administrador es quien puede crear las estructuras de más alto nivel. El sistema comienza por institución que está agrupada por programas. Los programas tienen dos funcionalidades: están asociados a los usuarios para poder generar estadísticas y están subdivididos en cursos. A cada curso se le asignan logros que pueden ser usados por los profesores.

Los cursos a su vez, están divididos en clases que representan la unidad máxima manejada por un usuario no administrador, en este caso el profesor. Cada una de las clases está conformada por integrantes, problemas y lenguajes. Los lenguajes

son los lenguajes de programación definidos para la clase, que deben haber sido creados previamente por el administrador.

Los integrantes son los usuarios que pertenecen a la clase, se clasifican como integrantes y no usuarios, para que cada usuario tenga la capacidad de actuar en más de un rol en el sistema.

Los problemas son los desafíos que deberán ser completados por los estudiantes. Un estudiante de una clase está habilitado para leer todos los problemas de la clase, pero solo podrá resolverlos a medida que desbloquee los requerimientos necesarios para tal fin. Un problema está conformado por varias estructuras como soluciones, comentarios, casos, requisitos, pistas y pistas de error.

Cada problema cuenta con una serie de requisitos que son determinados por el profesor. Estos requisitos pueden ser logros desbloqueados o problemas con temáticas similares. El estudiante no podrá resolver el problema hasta que haya completado los requisitos en su totalidad.

Cada solución es un intento del estudiante por resolver un problema, la cual contiene el código fuente enviado por el estudiante y será ejecutado por el servidor, una vez por cada caso de prueba del problema. La idea de esto es que las pruebas sean aisladas y den una mejor retroalimentación al usuario, al igual que mejorar el sistema de puntos. Cada caso contiene una entrada con su correspondiente salida deseada. El sistema guardará la respuesta de esta solución para cada caso y se lo reportará al usuario y al profesor. Para ayudar al estudiante, el profesor puede definir pistas de error, que son pistas asociadas a un caso específico con un tipo de error específico. Estas pistas solo se mostrarán si el usuario obtiene el error en el caso señalado.

Si el profesor desea mostrar una información más general, es posible la creación de una pista, que puede ser comprada por los estudiantes por un costo que será restado de sus puntos en el problema. Los estudiantes también pueden publicar comen-

tarios por cada problema, aunque para ser mostrados al resto de los estudiantes, deben ser previamente aprobados por algún profesor de la clase.

Adicionalmente el administrador puede cambiar configuraciones globales, como lo son los lenguajes de programación permitidos, los tipos de error presentados, los avatares usados por los estudiantes y los logros que ganarán los usuarios.

Proceso de juzgamiento

El principal problema en el proceso de juzgamiento consiste en que algunos problemas pueden tomar varios segundos en ser juzgados dependiendo de su complejidad, sumado a que hay momentos en los que varios estudiantes enviarán soluciones simultáneamente.

Inicialmente se intentaron juzgar los casos secuencialmente, por el mismo proceso que recibe las peticiones web, lo cual causó demoras significativas para los usuarios, no solo los que estaban enviando su solicitud de juzgamiento, sino también para otros usuarios que estaban haciendo cualquier tipo de petición.

Para resolver este problema se plantearon dos posibles soluciones. En primer lugar se analizó la posibilidad de dividir cada caso que debe ser juzgado en un hilo diferente, de manera que un problema con varios casos sería juzgado en el tiempo que tomara el caso más lento. Se decidió no implementar esta solución, en primer lugar por la incapacidad del lenguaje PHP de crear hilos nativamente, aunque existen bibliotecas que permiten simular hilos, pero no era el ideal. Segundo, porque el sistema debería separar esto en subproceso para poder retornar una respuesta temporal al usuario, o el navegador del usuario podría permanecer en estado de espera por un tiempo demasiado largo; sin embargo, dado que el proceso principal debe terminar para retornar al usuario, se pierde control para el manejo de este subproceso.

Finalmente se planteó una alternativa donde las soluciones fueran enviadas a una cola asincrónica, de tal forma que el usuario puede ser notificado posteriormente de

su resultado, mientras la plataforma mantiene control sobre el proceso subyacente. Por lo anterior, se decidió implementar esta solución.

El proceso para el juzgamiento consiste en que el usuario envía su solución, la cual es recibida por el sistema, compilada y guardada con un id único. Este id junto con el id del problema y del usuario son enviados a una tabla de trabajos que es almacenada en la base de datos. Al terminar este proceso, se envía una respuesta al usuario informando que su solución ha sido recibida pero que aún no ha sido juzgada. La única excepción a este procedimiento es si hay un error de compilación; en este escenario, el usuario es informado inmediatamente del error de compilación y dicha respuesta es almacenado como la respuesta a todos los casos del problema.

En el caso de que no haya un error de compilación, la cola tomará los juzgamientos en el orden de llegada y, por cada trabajo, primero crea el archivo de entrada en el mismo directorio donde está el archivo compilado. Posteriormente procede a ejecutarlo, teniendo en cuenta que existen dos formas de recibir la entrada: como entrada estándar en donde el archivo creado será pasado como lectura estándar, o mediante archivos, caso en el cual la ruta del archivo será pasada como parámetro a la función principal y se almacenará en una ruta con permisos de lectura para que el archivo ejecutable pueda leerlo. De igual manera la salida puede ser estándar, en cuyo caso se toma la salida del proceso de ejecución como respuesta, o como salida de archivo, en donde la ruta se pasa como parámetro a la función principal. La ruta donde debe ser almacenado el archivo tiene permisos de escritura para el archivo ejecutable, y el archivo resultante será tomado como respuesta del usuario.

Uno de los principales problemas de correr la ejecución de esta forma es el detectar que el problema está demorándose más tiempo del esperado, por ejemplo debido a un ciclo o una recursión infinita, dado que al proceso que está corriendo la ejecución no le es posible medir el tiempo transcurrido. Para solucionar este problema al mismo tiempo que está corriendo la ejecución del usuario, el sistema crea un subproceso, en donde cada segundo verifica que el proceso principal exista, o en caso contrario indica que terminó exitosamente y debe retornar el mismo valor que retornó el

proceso principal. En caso de que este subproceso haya completado el número de segundos máximos estipulado para este problema, acabará el proceso principal y retornara un código que posteriormente será interpretado para indicar al usuario que dicho caso se ha demorado más tiempo del permitido.

Al terminar la ejecución, el sistema debe revisar que la ejecución no se haya detenido inesperadamente debido a un error en tiempo de ejecución. Para comprobar esto, una vez termina la ejecución el sistema se revisa el archivo de *stderr* generado, buscando palabras claves que indiquen dicho error. En caso de ser encontradas, se le reporta al usuario que el caso en prueba reporta un error en tiempo de ejecución.

Una vez ha sido ejecutado y se cuenta con un archivo de salida del usuario, el sistema intenta arreglar problemas de formato que pueda haber tenido el usuario. Para esto, se borran todas las líneas en blanco que se encuentran en el archivo y se borran los espacios en blanco encontrados al principio y final de cada línea. Al finalizar este proceso se cuentan las líneas del archivo del usuario y del archivo de salida correcto, si hay una diferencia en este número, se le reporta al usuario que hay una diferencia en el número de líneas para este caso.

Después de que se tiene el formato deseado para la salida del usuario, el sistema compara línea por línea con el archivo correcto. Si se encuentra alguna diferencia, se reporta al usuario que es una respuesta errónea. Si el sistema termina de comparar los archivos y no encuentra diferencias, se reporta como respuesta correcta.

Simultáneamente a este proceso, en el navegador del usuario hay un mensaje indicando que aún se están ejecutando los casos. Por medio de peticiones Ajax, el navegador consulta una vez por segundo si todos los casos ya han sido evaluados y cuando reciba una respuesta positiva, recarga la página del navegador mostrando la respuesta al usuario. Si envía más de 15 peticiones, desiste para no inundar el sistema con peticiones, e informa al usuario que prontamente recibirá sus resultados.

En el siguiente diagrama se puede apreciar el proceso principal de envío y juzgamiento de soluciones:

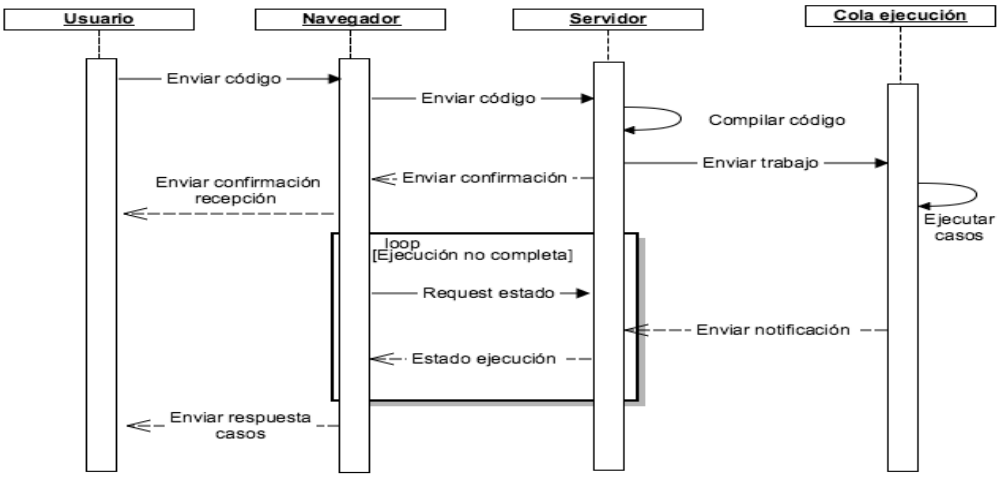


Ilustración 2. Diagrama de secuencia

Diseño elementos de *Gamification*

- **Logros:** El objetivo principal de los logros es que los estudiantes pudieran tener medallas por diferentes logros alcanzados en el sistema. El mayor reto enfrentado en esta característica fue que el administrador debía poder crear los logros sin necesidad de cambiar el código de la aplicación. Para esto, se creó un generador de consultas, en donde el administrador puede generar una serie de reglas, y por cada regla se puede seleccionar una tabla de la base de datos, aunque no todas las tablas están disponibles. Una vez seleccionada la tabla, existe la opción de seleccionar una columna de la tabla y una opción de comparación (=, <, >, <=, >=, <>), contra un valor dado por el administrador. Como último paso se debe introducir cuántos resultados debe devolver la consulta. Lo que el sistema hace es relacionar todas las tablas con el usuario y ejecuta la consulta, contando los resultados y otorgando la medalla a quien cumpla todas las condiciones del logro. Cada vez que el usuario envía una solución, escribe un comentario o compra una pista, el sistema verifica si es merecedor de una medalla. Adicionalmente, el usuario tiene la opción de verificar mediante un botón si ha ganado más medallas, al

igual que el administrador puede hacer esta verificación para todos los usuarios simultáneamente.

- **Pistas:** Estas son ayudas que el profesor otorga al estudiante para cada problema. Al leer la pista el estudiante debe usar los puntos ganados en el problema, a manera de costo que es asignado por el profesor. Estos puntos serán restados de los puntos totales del estudiante. En caso de que el estudiante tenga menos puntos de los requeridos para comprar la pista, el problema asignará cero puntos. Una vez descontados los puntos, el estudiante puede volver a ver la pista cuantas veces quiera.
- **Pistas error:** Son pistas que son asignadas por el profesor a un caso de prueba específico, cuando el usuario obtiene una respuesta en particular. La idea principal de esta funcionalidad es que el profesor indique qué pista es útil desbloquear para solucionar el problema en ese caso específico. Vale la pena aclarar que no es necesario utilizar esta funcionalidad de tal forma, y simplemente dar pistas de manera gratuita para el usuario.
- **Requisitos:** Para cada problema, el profesor puede limitar a quiénes pueden hacer el envío del problema. Para esta funcionalidad el profesor debe seleccionar una serie de problemas o de logros como prerrequisito para enviar soluciones al problema. Aunque el usuario no cumpla con estos requisitos, podrá ver el enunciado, comprar pistas y comentar; la única restricción será que no puede enviar la solución, lo que le impide ganar puntos para el problema.
- **Avatares:** Con el fin de incentivar a los usuarios a ganar puntos, hay unos avatares predeterminados que los usuarios pueden escoger para que los represente en el sistema. Estos avatares son subidos al sistema por el administrador. A cada avatar se le asigna un valor, que será el mínimo número de puntos que el usuario debe tener para poder usar al personaje. Por razones de portabilidad y para facilitar la tarea al administrador del servidor, tanto las imágenes de los avatares como las de los logros son almacenadas en la base de datos como un blob.

- **Comentarios:** Para incrementar la interacción entre estudiantes y para facilitar la retroalimentación hacia el profesor, los estudiantes pueden crear comentarios relacionados a un problema. En estos comentarios el estudiante puede dar ayudas a sus compañeros, o dar su opinión frente al problema. Antes de que el comentario sea publicado, debe haber sido aprobado por un profesor de la clase.
- **Tabla de posiciones:** Esta tabla tiene el fin de incentivar a los estudiantes a trabajar por medio de la competitividad. La tabla está organizada por puntos, y adicionalmente muestra las medallas ganadas para cada estudiante, una barra con el porcentaje del progreso y el avatar de cada estudiante. Uno de los problemas técnicos enfrentados en este requerimiento fue el hecho de que es computacionalmente costoso calcular el progreso general de cada estudiante, cada vez que un usuario consulta la tabla. Para mejorar este proceso se utiliza la técnica de cache, para almacenar el mejor envío de cada estudiante por problema. De esta manera la tabla está actualizada en tiempo real, y es más sencillo para el servidor realizar este cálculo.
- **Casos separados:** Uno de los principales problemas que tienen los jueces virtuales, es la frustración de los estudiantes al no obtener respuesta hasta que la solución pase el 100% de los casos. Para mejorar esto, se dividió cada problema en casos pequeños que van aumentando su dificultad gradualmente. De esta manera, el estudiante recibe puntos, así no haya resuelto el 100% del problema. Para este fin, el profesor crea los casos que considere apropiados por problema, y les asigna puntos que ganara el estudiante al aprobar cada caso.

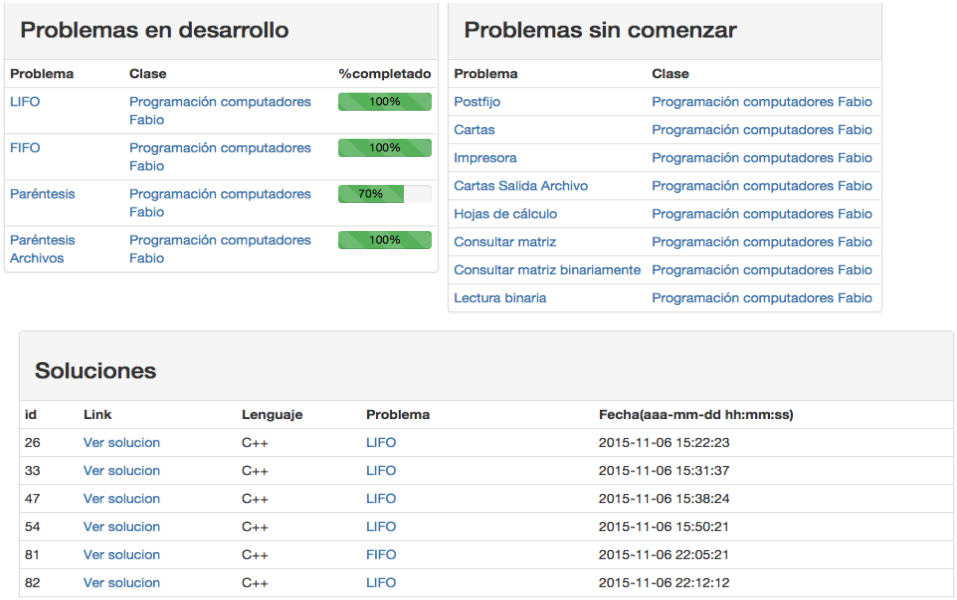
Ventajas para el profesor

Adicionalmente a las ventajas que provee usar un juez virtual en clase, como por ejemplo, la calificación rigurosa de las tareas, el que los estudiantes puedan trabajar fuera de clase, retroalimentación inmediata, y las otras mencionadas anteriormente en este documento, la plataforma Game-Iudex cuenta con ventajas adicionales para el profesor:

- **Comparador de ejecución:** Los profesores de la clase, pueden acceder al envío de un estudiante y además de ver el código enviado, pueden observar la salida obtenida por el sistema al ejecutar el código, al igual que la salida esperada. En los cuadros de texto se puede observar mediante marcas de colores las diferencias en los archivos y las líneas extras. Se puede observar un ejemplo en la siguiente imagen:

Ilustración 3. Comparador ejecución

- **Progreso estudiante:** El profesor puede analizar el progreso de cada estudiante, seleccionándolo de la tabla de posiciones. Ahí encontrará el porcentaje realizado de cada problema al igual que los problemas que aún no ha intentado. También encontrará todas las soluciones enviadas, con links para analizarlas más a fondo, junto con el código fuente, el veredicto de cada caso y su respectivo comparador de soluciones. Se puede observar un ejemplo en la siguiente imagen:



Problemas en desarrollo

Problema	Clase	%completado
LIFO	Programación computadores Fabio	100%
FIFO	Programación computadores Fabio	100%
Paréntesis	Programación computadores Fabio	70%
Paréntesis Archivos	Programación computadores Fabio	100%

Problemas sin comenzar

Problema	Clase
Postfijo	Programación computadores Fabio
Cartas	Programación computadores Fabio
Impresora	Programación computadores Fabio
Cartas Salida Archivo	Programación computadores Fabio
Hojas de cálculo	Programación computadores Fabio
Consultar matriz	Programación computadores Fabio
Consultar matriz binariamente	Programación computadores Fabio
Lectura binaria	Programación computadores Fabio

Soluciones

id	Link	Lenguaje	Problema	Fecha(aaa-mm-dd hh:mm:ss)
26	Ver solucion	C++	LIFO	2015-11-06 15:22:23
33	Ver solucion	C++	LIFO	2015-11-06 15:31:37
47	Ver solucion	C++	LIFO	2015-11-06 15:38:24
54	Ver solucion	C++	LIFO	2015-11-06 15:50:21
81	Ver solucion	C++	FIFO	2015-11-06 22:05:21
82	Ver solucion	C++	LIFO	2015-11-06 22:12:12

Ilustración 4. Progreso estudiante

- **Manejo simultáneo de clases:** El profesor puede manejar varias clases en el mismo servidor y con el mismo usuario, y aun así obtener estadísticas totalmente independientes. Los estudiantes también pueden pertenecer a varias clases y si el administrador lo permite, puede pertenecer incluso a clases de distintas instituciones con el mismo usuario. Los permisos de estar en cada clase son manejados por el rol, que está asociado al usuario y a la clase, por lo que un usuario puede ser estudiante en una clase y profesor en otra simultáneamente.
- **Cuenta simultánea de estudiante:** El profesor puede participar como un estudiante en la clase, o nombrar al monitor como profesor o crear un nuevo rol monitor. De esta manera pueden probar los problemas enviando soluciones y observando el ambiente normal del sistema como estudiantes, pero con la ventaja que no aparecerán en ninguna estadística ni en la tabla de posiciones.
- **Desactivación de información:** La información en el sistema nunca es verdaderamente borrada, siempre se desactiva. Lo que esto significa es que

aunque aún existe en la base de datos, no es visible para ningún usuario. Esto permite al profesor y al administrador usar datos antiguos que ya han sido borrados, al igual que recuperar datos borrados por accidente. Para la implementación de este servicio se hizo uso de la funcionalidad *soft delete* que se encuentra dentro del framework Laravel.

Despliegue

Para el manejo de versiones del código, se hizo uso del sistema Git [63], utilizando la herramienta GitHub [64]. Haciendo uso de un repositorio privado, se almacenó el código y todas sus versiones durante el desarrollo del proyecto. Al finalizar el desarrollo, el sitio de manejo de versiones generó las siguientes estadísticas:

- 3 contribuidores.
- 1 liberación.
- 1 rama, aunque en ocasiones existieron 2 simultáneamente.
- 54 problemas (issues) creados en total.
    - 49 resueltos.
    - 5 aún abiertos.
    - 30 bugs.
    - 25 mejoras.
- 154 commits.
- 291803 líneas de código insertadas (incluyendo dependencias).
- Grafica frecuencia commits:

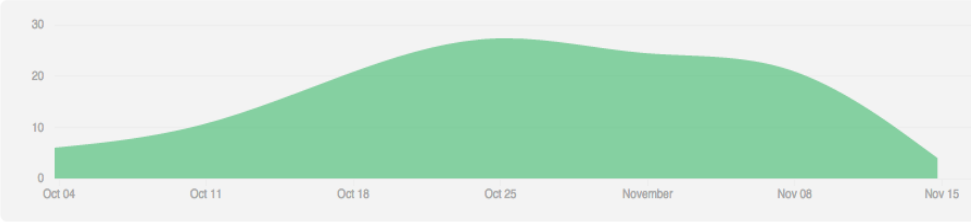


Ilustración 5. Frecuencia commits

- Grafica *punch card*

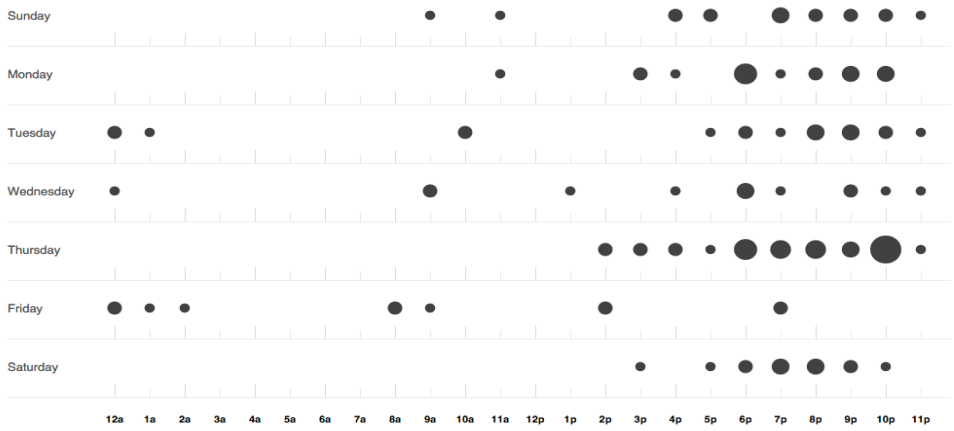


Ilustración 6. frecuencia commits días vs horas

Como dependencias se utilizaron las integradas con el framework Laravel. Adicionalmente se hizo uso de JQuery [65] y Ajax [66] para mejorar procesos de respuesta al usuario, al igual que la librería Bootrstrap [67], para mejorar el aspecto visual de la página. Para que el usuario pudiera introducir de manera más natural código en la página se utilizó un editor de texto gratuito llamado Ace Editor [68], y para que el código guardado en el sistema tuviera formato al ser mostrado al usuario se utilizó SyntaxHighlighter [69].

Para el despliegue de la aplicación, se decidió utilizar un servidor privado con sistema operativo Ubuntu 14.06 64 bits, como máquina virtual, adquirido en la compañía Digital Ocean [70]. El servidor está ubicado en la ciudad de New York y se cuenta con acceso SSH hacia el servidor. Para desplegar el código se utilizó el servicio Git instalado en el servidor, por lo que para desplegar se hace una actualización de la versión de Git por HTTPS hacia el repositorio almacenado en Github.

Las características del servidor han cambiado durante el desarrollo y la etapa de validación. Inicialmente se contaba con un servidor con 512 MB de memoria RAM, 1 CPU, 20 GB de disco de estado sólido y 1 TB de transferencia. Dado que el framework de laravel con la cola en modo de escucha consumen un poco más de 400 MB de memoria RAM, el sistema operativo acababa con el proceso de Laravel tan pron-

to la RAM superaba su límite lo que era una molestia para los usuarios, dado que debían esperar a que fuera reiniciado por un administrador. Este servidor fue ampliado a uno con las mismas características excepto el doble de memoria RAM y el doble de transferencia. Este servidor fue efectivo para gran parte de las validaciones, aunque presentaba un problema cuando los usuarios enviaban soluciones a casos donde el tiempo límite fuera alto (más de 1 segundo), dado que el procesador llegaba al límite ejecutando la solución, volviendo lentas las peticiones de otros usuarios. Por esta razón se decidió avanzar a un servidor con el doble de memoria RAM, es decir 2GB, 3TB de transferencia y lo más importante, 2 CPUs. Este servidor es el actualmente usado y el que se utilizó durante la última etapa de validación.

A continuación se encuentra un diagrama representando una vista general de la arquitectura:

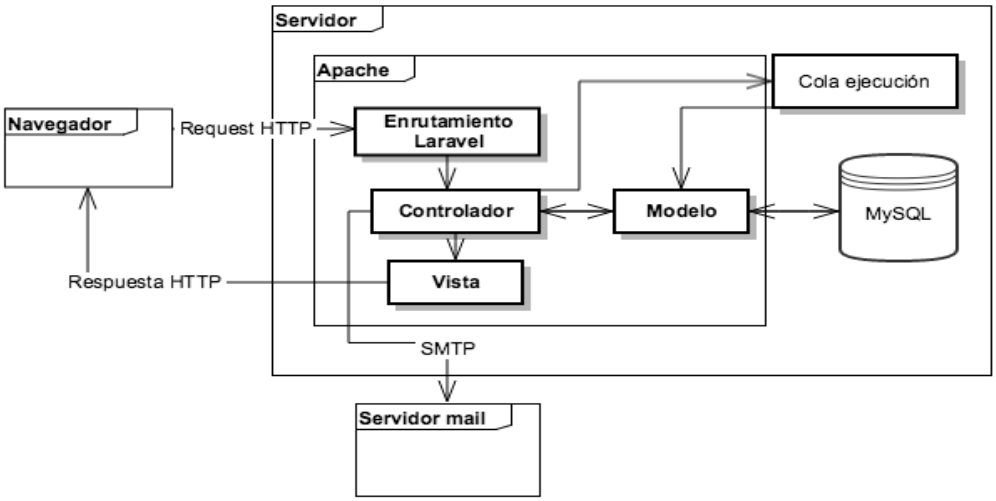


Ilustración 7. Diagrama de despliegue

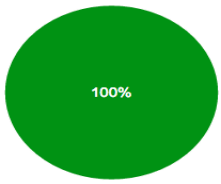
V – RESULTADOS

En esta sección, se analizarán los resultados obtenidos de las encuestas realizadas a los estudiantes y al monitor de la clase, según las especificaciones definidas en la sección de Análisis, donde se desarrolla el plan de pruebas del prototipo funcional.

Encuesta a estudiantes

De una muestra de 12 estudiantes, estás fueron las respuestas a una de las preguntas relacionadas a la experiencia de uso del estudiante con los diferentes jueces virtuales (Ver anexo Resultados encuestas):

¿De los jueces virtuales usados en clase, cuál fue su favorito?



Boca	0	0%
Discant	0	0%
Spoj	0	0%
Game-ludex	12	100%

Lo cual indica la gran acogida por parte de los estudiantes sobre Game-ludex en comparación a las otras herramientas utilizadas durante la clase. Por cada herramienta se les preguntó lo que consideraban como las mayores ventajas y desventajas, para resumir en la siguiente tabla, se pondrán como ventajas y desventajas, conceptos en los que por lo menos dos estudiantes coincidan.

- **Juez Boca**
  - Ventajas
    - Interfaz con globos de colores, representando un envío correcto.
    - Interfaz fácil de usar.
    - Forma de presentar errores.
  - Desventajas
    - Forma de envío de los problemas (archivos comprimidos).
    - Demora en obtener la respuesta.

- **Juez Discant**
  - Ventajas
    - Forma en la que muestra casos de prueba que no son aprobados.
    - Interfaz sencilla.
  - Desventajas
    - Forma de envío de los problemas (archivos comprimidos).
    - Errores en el sistema.
- **Juez Spoj**
  - Ventajas
    - Cantidad de ejercicios.
  - Desventajas
    - Falta de mensajes detallados en caso de envío erróneo.
- **Juez Game-ludex**
  - Ventajas
    - Elementos de gamification como: logros, puntos, medallas, pistas e imágenes.
    - El hecho de que sea didáctico.
    - División de casos de prueba por problema.
  - Desventajas
    - Demora en el servidor para responder.

Adicionalmente, con respecto a las técnicas de *Gamification* utilizadas, todas tuvieron una gran incidencia. En la encuesta se les pidió que calificaron de 1 a 4 qué tan importante fue cada elemento para ellos, siendo 4 lo más importante. Se considera 1 o 2 como un elemento que no tuvo incidencia en el proceso de esta persona y 3 o 4 que tuvo una incidencia significativa. Se resaltaron como más importantes: el sistema de puntos, el hecho de poder desbloquear problemas al avanzar en la clase, tener los problemas divididos en casos más pequeños y el uso de avatares, que fue considerado como importantes por el 91% de los estudiantes. El sistema de compra de pistas fue menos apreciado por los estudiantes siendo valorado como importante por el 50% de ellos.

Con respecto al el proceso de juzgamiento, el 91% de los estudiantes considera pertinente el uso de jueces virtuales en clases de programación. Escribiendo como

razones principales, el hecho de que es una forma objetiva de calificar y que se puede practicar fuera de la clase.

En cuanto a la influencia en la motivación directamente por parte de la herramienta en los estudiantes, respondieron de manera positiva, contestando el 91% que la herramienta los motivó a resolver problemas, a trabajar fuera de clase y a investigar más a fondo los temas tratados en clase, dando como razones principales la competitividad y los elementos amigables de la interfaz. Respecto a estos elementos de la interfaz, se les hizo otra pregunta, pidiendo que indicaran el elemento de la interfaz que más les gusto, obteniendo a los avatares como principal, siendo escogido por el 54% de la clase, seguido por la visualización de logros por el 36%, y la tabla de puntuación con 8%.

Finalmente, para medir la herramienta en su totalidad, se les preguntó a los estudiantes si recomendarían la herramienta para otros cursos de programación y si el uso de la herramienta le ayudó para prepararse para el parcial, en ambos casos el 91% de los estudiantes respondieron positivamente.

Encuesta al monitor de la clase

La recepción por parte del monitor fue positiva, en cuanto al uso de jueces virtuales en clases de programación y el aporte en el cual Game-ludex se diferencia de los otros jueces virtuales debido a su enfoque como herramienta para el apoyo en clases de programación.

Respecto a los beneficios que aporta la herramienta para profesores y monitores de clases de programación, indicó aspectos fundamentales como: tener control de los ejercicios a utilizar en la clase, con la posibilidad de crear casos de prueba independientes, tener un mejor seguimiento acerca del progreso de los estudiantes, la facilidad para comparar con la respuesta esperada y la revisión del código en línea, dando la posibilidad de ofrecer más rápida y mejor retroalimentación de las soluciones enviadas, además de la ayuda para poder realizar calificaciones objetivas.

En cuanto a la contribución del modelo de G*amification* implementado en Gameludex, concluye que el uso de medallas, pistas, avatares y tablas de posiciones los cuales lo hace divertido y motiva a los estudiantes a trabajar dentro y fuera de las horas de clase.

También resalta aspectos a mejorar como la forma en la que se indica a un estudiante que ha ganado un nuevo logro y la generación de estadísticas sobre el desempeño de un estudiante durante la clase, funcionalidades que se tendrán en cuenta para un trabajo futuro.

Para mayor detalle acerca de las respuestas de los estudiantes y el monitor (Ver anexo Encuesta monitor)

Los resultados obtenidos permiten concluir que la motivación de los estudiantes se vio beneficiada por el uso de la plataforma con técnicas de *Gamification,* en el apoyo al proceso de aprendizaje en cursos introductorios de programación.

VI – CONCLUSIONES

Análisis de Impacto del Desarrollo

En esta sección se evaluará el impacto del desarrollo de Game-ludex desde tres perspectivas diferentes.

Disciplinar: El trabajo se enfocó en las siguientes áreas disciplinares como lo son las ciencias de la computación y la educación.

En cuanto a las ciencias de la tecnología se propone una solución de fácil uso y entendimiento, una herramienta web que permite el juzgamiento de problemas de programación de manera automática, e incorpora conceptos de *Gamification* que ha sido una técnica que ha tenido éxito en la mejora de diferentes procesos no solo sistemáticos sino también en actividades cotidianas en contextos de empresas e instituciones, además de guardar los problemas con sus respectivos casos y las soluciones de los estudiantes para su fácil acceso desde cualquier sitio con acceso a Internet.

Respecto al área de la educación, se apoya al estudiante en sus factores de motivación extrínseca e intrínseca, debido a las técnicas de *Gamification* que estimulan este tipo de motivaciones en el estudiante, bien sea por competitividad tratando de permanecer en los primeros puestos de la tabla de puntuaciones, descubrir nuevas funcionalidades en la plataforma, desbloquear nuevos avatares o la premiación con logros y medallas bien sea por su excelente desempeño o por la valoración de esfuerzos al intentar resolver un problema.

Social: el impacto en el área social se puede apreciar en cómo el uso de la herramienta los motivó para seguir resolviendo problemas, incluso se puede reflejar en la colaboración entre los estudiantes al momento de comprar una pista y compartirla con los demás para poder realizar un problema.

Se puede ver reflejado no solo en la calificación del curso, sino también en el progreso que tuvo el estudiante dentro de la herramienta, incluso si no pudo resolver muchos problemas, se puede mirar la cantidad de veces que lo intentó y la mejora entre cada solución enviada, lo cual puede repercutir en la mejora de las calificaciones del estudiante en la materia o en bajar el porcentaje de estudiantes que reprueban la materia.

En un futuro se espera que con el uso de Game-Iudex se pueda aumentar el promedio de notas de los estudiantes en un curso, disminuir el porcentaje de alumnos reprobados y la disminución de alumnos que retiran la materia, o cual puede incidir en el número de estudiantes que ingresen a carreras relacionadas con las ciencias de la computación y tratar de cubrir la demanda de profesionales en tecnología para suplir la oferta de trabajo para el sector de TI del país.

Económico:

En el sector económico se ve reflejado en el uso efectivo de las horas de clases, partiendo de la importancia de las asignaturas en el plan de estudios de las carreras de ingeniería, específicamente en la de sistemas por el número de créditos de la materia, y lo que representa el valor de cada crédito de acuerdo al costo de la matrícula.

Se puede apreciar el impacto en la mejor utilización de las horas de clase por parte de los estudiantes, de los profesores y monitores, para estos últimos dos ofrece la posibilidad de hacer un seguimiento a los estudiantes de manera más fácil, debido a que la herramienta califica de manera automatizada todos los casos de los estudiantes y no hay que revisar problema por problema y caso por caso de cada estudiante, lo cual consume tiempo el cual podría utilizarse de una mejor manera.

También se puede considerar en cuanto a una mejor y más rápida retroalimentación, los diferentes tipos de resultados permiten tanto al profesor, monitor como al estudiante enfocar sus esfuerzos a encontrar un los posibles errores en las soluciones enviadas por el estudiante.

Conclusiones

Para el desarrollo de las conclusiones se tuvieron en cuenta, tanto las encuestas realizadas a estudiantes y al monitor, el desempeño técnico del sistema y el comportamiento observado de los estudiantes en el curso de programación que hicieron uso de la herramienta.

Debido a limitaciones de tiempo, no fue posible hacer una comparación del curso de programación frente a otro donde no se hiciera uso de la herramienta, por lo que no se puede decir con certeza qué tanto afecta a los estudiantes el hecho de hacer uso de una herramienta de juzgamiento en la clase. De acuerdo a lo observado en la clase donde se realizó la verificación, se puede decir que todos los resultados apoyan lo formulado en el marco teórico, indicando que un sistema de juzgamiento en clase de programación beneficia tanto a los profesores como a los estudiantes.

Aunque existen herramientas para el juzgamiento de ejercicios de programación, herramientas del mismo estilo con *Gamification* y herramientas para medir el progreso de los estudiantes en general en clases, no hay una herramienta en el mercado que combine estos conceptos. El resultado final es una herramienta para ser utilizada en clases de programación, que utiliza técnicas de *Gamification* y que permita al estudiante trabajar fuera del horario de clase, manteniendo el registro de progreso del estudiante durante la clase, al mismo tiempo proveyendo herramientas para facilitar las tareas del profesor dentro de la clase.

El hecho de fundamentar el desarrollo con el el trabajo “Modelo para la Motivación del Aprendizaje de la Programación utilizando *Gamification*” por Ricardo José Arenas, permitió tener una base teórica solida al momento de escoger los elementos de *Gamification*. A continuación se presenta un análisis de cada uno de los elementos de G*amification* utilizados en el sistema:

- **Medallas:** Se considera que el hecho de haber creado un generador de consultas facilita en gran parte la labor del profesor de añadir medallas al siste-

ma. Esta es una tarea que puede consumir tiempo inicialmente al profesor, pero dado que las medallas pueden ser usadas en diferentes cursos en distintos periodos, no es una tarea que deba realizarse continuamente. Por otro lado este era un elemento que los estudiantes pareció gustarles, y motivarlos a trabajar, para ganar medallas y tener más que sus compañeros. Una labor importante del profesor en este elemento, es ajustar las medallas al nivel de los estudiantes, dado que no todos los cursos alcanzan el mismo nivel en el mismo periodo de tiempo, las medallas deben ser otorgadas a los estudiantes en un tiempo uniformemente distribuido por la duración del curso.

- **Pistas:** Este elemento fue escogido por los estudiantes como el que menos los motivo a continuar trabajando. Se observó que es difícil para el profesor escribir pistas para los problemas, dado que los estudiantes pueden tener errores diferentes y puede ser desmotivante para los estudiantes, comprar pistas que resuelven un problema diferente al que están teniendo en su código. Otro factor importante que se observó durante las clases es que inicialmente un estudiante compraba la pista y esta era distribuida a todos los miembros de la clase. Cuando los mismos estudiantes descubrieron este comportamiento, se formaban discusiones sobre quién compraría la pista del siguiente problema. Aunque esto fomentó la interacción como grupo, no fue el comportamiento deseado para este elemento.
- **Tablero de puntuación:** Generó una sana competencia entre los estudiantes, dado que podían ver no solo los puntos a comparación de los otros integrantes, sino que también veían el progreso general, las medallas ganadas y su avatar. Un problema encontrado en este elemento, es el hecho que los estudiantes que están al final de la tabla, pierden el elemento de motivación que esta proporciona, dado que es muy difícil para ellos alcanzar las primeras posiciones.
- **Avatares:** Este fue uno de los elementos más motivadores para los estudiantes, debido a que intentaban ganar puntos por el solo hecho de desbloquear un avatar con él se sintieran identificados. Un reto para el profesor pa-

ra mantener este elemento como motivante es el hecho de que deben ser personajes con los que los estudiantes se sientan identificados por lo que debe ser algo muy relacionado con su edad. Adicionalmente los puntos para desbloquear los avatares debe estar estrechamente relacionado con los puntos otorgados en los problemas, para que sea algo retador para los estudiantes.

- **Comentarios:** Dado que el curso en el que se hizo la validación era un curso relativamente pequeño (18 estudiantes), los comentarios no fueron un elemento usado dentro de la clase, dado que les resultaba más fácil comunicarse por otros medios. Este componente fue usado para reportar errores del sistema, o errores en los problemas a los profesores. Se espera que en grupos más grandes o cuando el sistema maneja varias clases del mismo curso simultáneamente este componente sea de gran utilidad para los usuarios.
- **Requisitos:** Aunque no es fácil medir directamente la influencia de este componente en la motivación de los estudiantes, se considera que el hecho de que no hicieran problemas para los que no estaban preparados o no tenían los conocimientos necesarios, hizo que el nivel de frustración fuera menor.
- **División de problemas en casos:** Con base en los resultados de las encuestas y en lo observado en clase, se puede afirmar que este elemento hizo una gran diferencia respecto a la motivación. Una de las principales causas de desmotivación en los jueces virtuales tradicionales, es que el código del estudiante debe funcionar para todos los casos de prueba planteados para obtener una respuesta positiva. El hecho de que sea desglosado en casos pequeños permite al usuario mejorar su código gradualmente. Una de las dificultades para el profesor, es que debe pensar en posibles casos que hagan fallar el código del estudiante, y excluir estas entradas de ciertos casos.

En cuanto a la interfaz gráfica, se contó con la posibilidad de recibir retroalimentación de los estudiantes y de los profesores por 3 semanas, lo que permitió mejorarla haciéndola más dinámica, intuitiva y agradable para el usuario. Con ello, se recibió

una respuesta positiva por parte de los estudiantes frente a la interfaz gráfica, en la encuesta realizada.

Los elementos de ayuda para el profesor y el administrador creados, como lo son el comparador de ejecuciones, el visor de código del estudiante, el generador de consultas para las medallas, aportan una gran ventaja no solo al profesor sino a todos los integrantes, dado que esto resulta en mayores elementos en el sistema, y una mejor y más pronta retroalimentación para los estudiantes por parte del profesor. El profesor cumple un rol importante en este tipo de sistemas usados en clases, y si el hecho de usar un sistema genera una carga extra para el profesor, es probable que no se vean los mismos resultados o que no sea ampliamente usado el sistema.

El servidor utilizado y descrito en la sección de diseño, fue suficiente para el curso en el que fue probado, y sería suficiente para multiplicar el número de usuarios, siempre y cuando las clases no sean en el mismo momento, dado que es este pico de peticiones en el momento de la clase lo que limita al servidor. El hecho de que la herramienta haya sido desarrollada con herramientas gratuitas, facilita la instalación del sistema en diferentes ambientes, al igual que facilita la futura colaboración de otras personas en el proyecto.

Trabajo Futuro

Actualmente la herramienta es funcional, que puede ser utilizada en clases de programación en todo tipo de instituciones. Existen algunos elementos del sistema que pueden ser mejorados significativamente al igual que adiciones que mejorarían la experiencia del usuario en la herramienta.

- **Ejecutador:** Actualmente el sistema ejecuta el código del usuario mediante una serie de acciones *shell*. Esto puede traer problemas al mover la herramienta a un sistema con especificaciones diferentes y no es fácil adaptarlo a requerimientos diferentes que no sea una entrada y salida tradicional. Para mitigar esto se sugiere la creación de un módulo separado que ejecute los programas de los usuarios en un ambiente aislado, con diferentes opciones

de ejecución, mejor medición del tiempo y recursos de memoria, y utilizando una herramienta que permita realizar esta tarea de forma paralela.

- **Analizador de código:** En este momento el veredicto del juez es otorgado únicamente comparando la salida de la ejecución, contra una salida ingresada al juez previamente. Una ayuda grande para el usuario sería un módulo que analizara el código del usuario, entregando información útil como posibles causas de errores y la complejidad general del programa.
- **Sistema de distribución:** Actualmente todas las dependencias del sistema son gratuitas, pero si se desea instalar en un servidor el sistema, hay que instalar todas las dependencias en el mismo servidor independientemente. Se sugiere el uso de un motor de contenedores para una mejor distribución del sistema.
- **Registro:** Actualmente la forma de registrarse en el sistema es crenado una nueva cuenta, donde las credenciales serán almacenadas en el sistema. Para una mejor experiencia de usuario se sugiere el uso de aplicaciones externas de inicio de sesión.
- **Comparador de soluciones:** La herramienta actual para comparar soluciones cumple con los requerimientos planteados, aunque una mejora en este módulo, permitiría al usuario recibir información más detallada de donde puede estar el error, agregando tipos de errores al sistema y eliminando al máximo los errores de presentación.
- **Detector de plagio:** Dado que se tiene almacenado todo el código enviado por los estudiantes, podría ser beneficioso para el profesor y la institución implementar un detector de plagio, en donde se estandarice el código, removiendo elementos personales como nombres de variables o comentarios y se comparen las soluciones generando un porcentaje de posibilidad de copia. Otra opción sería adaptar alguna herramienta que cumpla este propósito en el sistema.

VII- REFERENCIAS Y BIBLIOGRAFÍA

[1] A. Blackwell, "What is programming," in *14th workshop of the Psychology of Programming Interest Group*, 2002, pp. 204–218.

[2] B. Isong, "A Methodology for Teaching Computer Programming: first year students' perspective," *International Journal of Modern Education and Computer Science (IJMECS)*, vol. 6, no. 9, p. 15, 2014.

[3] D. Crow, "Why every child should learn to code," *the Guardian*, 2014.

[4] R. Vivian, K. Falkner, and C. Szabo, "Can everybody learn to code?: computer science community perceptions about learning the fundamentals of programming," 2014, pp. 41–50.

[5] P. Kinnunen and L. Malmi, "Why students drop out CS1 course?," in *Proceedings of the second international workshop on Computing education research*, 2006, pp. 97–108.

[6] S. Adams, "The College Degrees With The Highest Starting Salaries in 2015," *Forbes*, 2015.

[7]. M. de E. Nacional, "Jaque a la deserción," vol. 14, p. 20, 2010.

[8] S. Deterding, M. Sicart, L. Nacke, K. O'Hara, and D. Dixon, "Gamification. using game-design elements in non-gaming contexts," in *PART 2------ Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, 2011, pp. 2425–2428.

[9] L. Malmi, A. Korhonen, and R. Saikkonen, "Experiences in automatic assessment on mass courses and issues for designing virtual courses," *ACM SIGCSE Bulletin*, vol. 34, no. 3, pp. 55–59, 2002.

[10] Icpc.baylor.edu, "The ACM-ICPC International Collegiate Programming Contest," 2015.

[11] Acm.org, "Welcome Association for Computing Machinery," 2015.

[12] Code.google.com, "Google Code Jam," 2015.

[13] G. Bhatt and L.-D. Hackney, "Khan Academy," 2014.

[14] Coursera, “Coursera - Free Online Courses From Top Universities,” 2015.

[15] R. G. Figueroa, C. J. Sol’\is, and A. A. Cabrera, “Metodolog’\ias Tradicionales vs. Metodolog’\ias Ágiles,” *Universidad Técnica Particular de Loja, Escuela de Ciencias en Computación.(En l’\inea), Disponible en: http://adonisnet. files. wordpress. com/2008/06/articulo-metodologia-de-sw-formato. doc*, 2008.

[16] L. Yuan, S. Powell, and J. CETIS, “MOOCs and open education: Implications for higher education,” 2013.

[17] R. J. Arenas Par’\is, “Modelo para la Motivación del Aprendizaje de la Programación utilizando Gamification,” 2014.

[18] C. Schulte and J. Bennedsen, “What do teachers teach in introductory programming?,” in *Proceedings of the second international workshop on Computing education research*, 2006, pp. 17–28.

[19] E. Lahtinen, K. Ala-Mutka, and H.-M. Järvinen, “A study of the difficulties of novice programmers,” in *ACM SIGCSE Bulletin*, 2005, vol. 37, no. 3, pp. 14–18.

[20] C. Fidge and D. Teague, “Losing their marbles: syntax-free programming for assessing problem-solving skills,” in *Proceedings of the Eleventh Australasian Conference on Computing Education-Volume 95*, 2009, pp. 75–82.

[21] S. Fincher, “Session 12a4 What are We Doing When We Teach Programming ? Session 12a4 The ‘ Literacy ’ approach,” pp. 8–12, 1999.

[22] L. A. Stein, “Interactive programming: revolutionizing introductory computer science,” *ACM Computing Surveys (CSUR)*, vol. 28, no. 4es, p. 103, 1996.

[23] A. Gallego Rodr’\iguez and E. Mart’\inez Caro, “Estilos de aprendizaje y e-learning. Hacia un mayor rendimiento académico,” 2003.

[24] C. J. Costa and M. Aparicio, “Analysis of e-learning processes,” in *Proceedings of the 2011 Workshop on Open Source and Design of Communication*, 2011, pp. 37–40.

[25] N. S. Subramanian, S. Anand, and K. Bijlani, “Enhancing E-learning Education with Live Interactive Feedback System,” in *Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing*, 2014, p. 53.

[26] N. Arman, "E-learning materials development: applying and implementing software reuse principles and granularity levels in the small," in *Proceedings of the International Conference on E-Learning, E-Business, Enterprise Information Systems, \& E-Government*, 2010.

[27] Y.-L. Theng, E. Ei Tun, M. M. H. Zaw, S. Y. Y. Cho, C. Miao, M.-Y. Kan, and A. C. Tang, "An empirical study of students' perceptions on e-learning systems," in *Proceedings of the 2nd International Convention on Rehabilitation Engineering \& Assistive Technology*, 2008, pp. 245–249.

[28] A. R. Yohannis, Y. D. Prabowo, and A. Waworuntu, "Defining gamification: From lexical meaning and process viewpoint towards a gameful reality," in *Information Technology Systems and Innovation (ICITSI), 2014 International Conference on*, 2014, pp. 284–289.

[29] B. Heilbrunn, P. Herzig, and A. Schill, "Tools for Gamification Analytics: A Survey," in *Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on*, 2014, pp. 603–608.

[30] A. Domínguez, J. Saenz-de-Navarrete, L. De-Marcos, L. Fernández-Sanz, C. Pagés, and J.-J. Martínez-Herráiz, "Gamifying learning experiences: Practical implications and outcomes," *Computers \& Education*, vol. 63, pp. 380–392, 2013.

[31] D. Rojas, B. Kapralos, and A. Dubrowski, "The missing piece in the gamification puzzle," in *Proceedings of the First International Conference on Gameful Design, Research, and Applications*, 2013, pp. 135–138.

[32] A. Vaibhav and P. Gupta, "Gamification of MOOCs for increasing user engagement," in *MOOC, Innovation and Technology in Education (MITE), 2014 IEEE International Conference on*, 2014, pp. 290–295.

[33] E. D. Mekler, F. Brühlmann, K. Opwis, and A. N. Tuch, "Do points, levels and leaderboards harm intrinsic motivation?: an empirical analysis of common gamification elements," in *Proceedings of the First International Conference on Gameful Design, Research, and Applications*, 2013, pp. 66–73.

[34] M. Iriarte, MOTIVACIÓN 'INTRÍNSECA' Y 'EXTRÍNSECA, 2007.

[35] V. P. DE LA MOTIVACIÓN, "MOTIVACIÓN, APRENDIZAJE Y RENDIMIENTO ESCOLAR."

[36] J. A. Huertas, "Motivación," *Querer aprender*, vol. 1, 1997.

[37] C. Richards, C. W. Thompson, and N. Graham, "Beyond designing for motivation: the importance of context in gamification," 2014, pp. 217–226.

[38] T. Welzer, M. Dru\vzovec, P. Cafnik, M. Z. Venu, and H. Jaakkola, "Awareness of Culture in e-learning," in *Information Technology Based Higher Education and Training (ITHET), 2010 9th International Conference on*, 2010, pp. 312–315.

[39] G. Barata, S. Gama, M. J. Fonseca, and D. Gonçalves, "Improving student creativity with gamification and virtual worlds," in *Proceedings of the First International Conference on Gameful Design, Research, and Applications*, 2013, pp. 95–98.

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

[41] J. R. Citadin, A. Kemczinski, A. V. de Matos, D. C. Robles, and M. do C. D. Freitas, "ANÁLISIS DE HERRAMIENTAS DE COLABORACIÓN CON MOOCs."

[42] D. Yang, T. Sinha, D. Adamson, and C. P. Rose, "Turn on, tune in, drop out: Anticipating student dropouts in massive open online courses," in *Proceedings of the 2013 NIPS Data-Driven Education Workshop*, 2013, vol. 10, p. 13.

[43] Udacity.com, "About Us - Udacity," 2015.

[44] Khan Academy, "Khan Academy," 2015.

[45] C. Thompson, "How Khan Academy is changing the rules of education," *Wired Magazine*, vol. 126, 2011.

[46] Khan Academy, "Khan Academy," 2015.

[47] M. Noer, "One man, one computer, 10 million students: How Khan Academy is reinventing education," *Recuperado de: http://www. forbes. com/sites/michaelnoer/2012/11/02/one-man-one-computer-10-million-students-how-khan-academy-is-reinventing-education*, 2012.

[48] Coursera, "Coursera - Free Online Courses From Top Universities," 2015.

[49] Code School, "About Code School - Code School," 2015.

[50] J. Pereira, "Modelos y técnicas de enseñanza-aprendizaje MOOC aplicables en la formación online universitaria no masiva Extracting MOOC teaching and learning models and techniques for using them in online, non-massive, university courses."

[51] M. A. Revilla, S. Manzoor, and R. Liu, "Competitive learning in informatics: The UVa online judge experience," *Olympiads in Informatics*, vol. 2, pp. 131–148, 2008.

[52] Uva.onlinejudge.org, "UVa Online Judge - Home," 2015.

[53] Codechef.com, "About | CodeChef," 2015.

[54] spoj.com, "Sphere Online Judge (SPOJ) - Info," 2015.

[55] Codefights.com, "Join CodeFights," 2015.

[56] S. Perez, "CodeFights Scores $2.4 Million To Turn Coding Practice Into A Game," *TechCrunch*, 2015.

[57] Codility.com, "Codility's story and team," 2015.

[58] Leetcode.com, "LeetCode Online Judge," 2015.

[59] C. P. De Campos and C. E. Ferreira, "BOCA: um sistema de apoio a competições de programação," in *Workshop de Educação em Computação*, 2004, pp. 885–895.

[60] Dserrano3.github.io, "Discant," 2015.

[61] M.-H. Tzou, "Blurring the boundary between play and work; Gamification and its takeover," 2012.

[62] Dev.mysql.com, "MySQL :: Developer Zone," 2015.

[63] Git-scm.com, "Git," 2015.

[64] GitHub, "Build software better, together," 2015.

[65] jQuery jquery.org, "jQuery," *Jquery.com*, 2015.

[66] jQuery jquery.org, "jQuery.ajax() | jQuery API Documentation," *Api.jquery.com*, 2015.

[67] M. Otto, "Bootstrap The world's most popular mobile-first and responsive front-end framework.," *Getbootstrap.com*, 2015.

[68] Ace.c9.io, "Ace - The High Performance Code Editor for the Web," 2015.

[69] A. Gorbatchev, "SyntaxHighlighter," *Alexgorbatchev.com*, 2014.

[70] Digitalocean.com, "Simple Cloud Infrastructure for Developers | DigitalOcean," 2015.

[71] Help.github.com, "GitHub Glossary - User Documentation," 2015.

[72] A. Deshpande and D. Riehle, "Continuous integration in open source software development," in *Open source development, communities and quality*, Springer, 2008, pp. 273–280.

[73] A. Van Deursen, P. Klint, and J. Visser, "Domain-Specific Languages: An Annotated Bibliography.," *Sigplan Notices*, vol. 35, no. 6, pp. 26–36, 2000.

[74] Cisco, "Protocol Basics: Secure Shell Protocol - The Internet Protocol Journal, Volume 12, No.4," 2015.

[75] Linfo.org, "Shell definition by The Linux Information Project (LINFO)," 2015.

[76] Google Books, "Patent US20080189693 - Remote firmware management for electronic devices," 2015.

VIII - ANEXOS

Glosario

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

Repositorio: Es el elemento más básico dentro de Git. Este contiene todos los archivos y carpetas de un proyecto, y almacena el historial de revisiones [71].

Issues: Es una manera de hacer un seguimiento de las tareas, mejoras y errores en el proyecto. Puedan ser compartidos y discutidos con el resto del equipo. Consta de un título, una descripción y adicionalmente se puede agregar etiquetas y asignaciones a miembros del equipo de trabajo [71].

Commit: Es la acción en la que el desarrollador contribuye una porción de código al repositorio de un proyecto [72].

Framework: Es un conjunto de clases que construye un diseño abstracto de soluciones a problemas relacionados [73].

SSH: Es un protocolo para comunicación segura, relativamente simple [74].

Shell: Es un programa que provee el texto de la interfaz de usuario tradicional de Linux o sistemas operativos Unix similares [75].

Https: Es un protocolo web integrado en los navegadores web, que encripta y desencripta las peticiones y respuestas del usuario [76].

Rama: Una versión en paralelo del repositorio principal. Está contenida dentro del repositorio principal pero no lo afecta [71].

Anexo 1: Requerimientos.

Anexo 2: Issues.

Anexo 3: Formulario encuestas.

Anexo 4: Resultados encuestas.

Anexo 5: Encuesta monitor.