UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMÁTICA

TESINA DE LICENCIATURA

Título: Juego serio como actividad de autoevaluación de los alumnos y su integración con un entorno virtual de enseñanza y aprendizaje
Autores: Federico Héctor Archuby
Director: Dra. Cecilia Verónica Sanz
Codirector: Dra. Patricia Mabel Pesado
Carrera: Licenciatura en Sistemas

Resumen

La utilización de juegos en escenarios educativos es una práctica naturalizada, especialmente en los niveles iniciales de la educación formal. Las tecnologías digitales han evolucionado de forma tal que los videojuegos se han puesto también en el abanico de posibilidades para desarrollar estrategias educativas innovadoras. Debido a la proliferación de los dispositivos móviles que ocurrió en los últimos años, han empezado a surgir investigaciones para aprovechar las posibilidades de los juegos serios sobre este tipo de dispositivos.

En la presente tesina, se ha desarrollado un juego serio para dispositivos móviles orientado a la autoevaluación de los estudiantes y con posibilidades de integración con Entornos Virtuales de Enseñanza y Aprendizaje (EVEA). Se realizó una revisión bibliográfica sobre juegos serios, y su relación con la educación, haciendo énfasis en aquellos pensados para autoevaluación y en dispositivos móviles. También se relevaron diferentes motores de juegos como posibilidades para el desarrollo del juego creado en esta tesina.

El desarrollo es un juego serio llamado "Desafiate", que se integra con la herramienta de autoevaluación del EVEA IDEAS. Con este juego se desarrolló una experiencia con alumnos y docentes del módulo de EPA del curso de ingreso para las carreras que dicta la facultad. Se presentan los resultados y las conclusiones del trabajo.

Palabras Claves

Educación, Autoevaluación, Juegos Serios, Dispositivos móviles, Unity 3D, Entornos virtuales de Enseñanza y Aprendizaje.

Conclusiones

- *A raíz de las experiencias realizadas se observa una actitud positiva de docentes y alumnos hacia la incorporación de juegos serios en procesos de enseñar y aprender. Consideran que incluirlos como estrategia de evaluación es una idea innovadora y que les ha gustado.*
- *Desafiate ha tenido buena aceptación, rescatándose por parte de los alumnos el uso de dispositivos móviles como un aspecto interesante, al igual que la idea de utilizar un juego para su autoevaluación. Los docentes han valorado también este último aspecto.*

Trabajos Realizados

- *Revisión bibliográfica sobre el uso de juegos serios móviles y orientados a la autoevaluación.*
- *Análisis de diferentes motores de juegos, y selección de uno a utilizar para el desarrollo de un juego serio móvil.*
- *Diseño del juego serio Desafiate, juego de tipo preguntas y respuestas para dispositivos móviles orientado a la autoevaluación.*
- *Implementación e integración de Desafiate con el EVEA IDEAS.*
- *Desarrollo de experiencias con alumnos y docentes del módulo de EPA del curso de ingreso y análisis de los resultados.*

Trabajos Futuros

- *Evolucionar Desafiate considerando aspectos de usabilidad y jugabilidad recogidos de las experiencias ya realizadas.*
- *Desarrollar nuevas experiencias incluyendo a alumnos y docentes de otras disciplinas.*
- *Avanzar en la integración de Desafiate con otros EVEAs*
- *Profundizar la investigación sobre juegos serios para procesos de autoevaluación.*

Fecha de la presentación: **Marzo de 2017**

Agradecimientos

A mi familia por su apoyo a lo largo de mi carrera y por el esfuerzo hecho para permitirme realizar mis estudios.

A Rocío por toda su ayuda y por estar siempre presente y acompañándome.

A mis directoras, Cecilia Sanz y Patricia Pesado, por permitirme realizar esta tesina bajo su dirección y por todos los conocimientos y herramientas que me brindaron, sin los cuales el presente trabajo no hubiera podido ser realizado.

A las personas que colaboraron en las pruebas realizadas, ya que sin ellas hubiese sido imposible evaluar el trabajo.

A mis amigos y compañeros de la facultad con los que compartí tantas horas de estudio y cursadas.

A los miembros del III-LIDI que siempre estaban dispuestos a darme una mano y al instituto, por facilitarme el acceso a los recursos necesarios para realizar este trabajo.

A todos ellos, GRACIAS.

ÍNDICE GENERAL

CAPÍTULO 1

INTRODUCCIÓN. OBJETIVOS Y MOTIVACIÓN

1.1. Objetivos

1.1.1. Objetivo general

Investigar las posibilidades de los juegos serios móviles en educación y de su integración a los entornos virtuales de enseñanza y aprendizaje para favorecer la autoevaluación de los alumnos en procesos educativos.

1.1.2. Objetivos específicos

1. Estudiar las potencialidades de los juegos serios (en particular móviles) en procesos educativos

2. Analizar experiencias de utilización de juegos móviles en educación

3. Investigar herramientas que permitan el desarrollo de juegos serios móviles

4. Realizar un desarrollo de un juego serio orientado a la autoevaluación de los alumnos

5. Integrar el juego serio desarrollado al EVEA IDEAS a partir de su módulo de autoevaluación

6. Analizar la usabilidad de juego desarrollado y de la integración realizada

1.2. Motivación

El avance de las tecnologías digitales ha ido evolucionando rápidamente desde las últimas décadas. Esta evolución ha hecho que muchas disciplinas y actividades de la vida cotidiana sufrieran cambios o se expandieran con el propósito de aprovechar sus posibilidades. La educación no ha sido ajena a esto (Garcia Aretio, Ruiz Corbella, y Domínguez Figaredo, 2007). De Benito Crosetti y Salinas Ibañez (2016) afirman que "La agenda de investigación en el campo de la Tecnología Educativa (TE) está caracterizada por experimentar un ritmo acelerado en las novedades y temas emergentes, por ofrecer un gran abanico de elementos de interés y por estar fuertemente influenciada por las modas..." y por esto resulta importante profundizar en su investigación.

Entre los años 90 e inicios del 2000, las universidades se hicieron testigos del avance y proliferación de entornos virtuales de enseñanza y aprendizaje (EVEA). Estos sistemas centrados en la web abrieron el camino para que docentes y alumnos pudieran compartir y vivenciar procesos educativos mediados por estas tecnologías, con la utilización de un conjunto de herramientas (de comunicación, de distribución de contenidos, de gestión, de evaluación, de estadísticas, entre otros), integradas en un mismo espacio y con posibilidad de realizar un seguimiento de las actividades de los alumnos (Sanz, 2015). Actualmente existen muchos EVEA, algunos con gran difusión en la comunidad académica: Moodle, E-ducativa, Chamilo, Blackboard, y otros que han sido desarrollos propios de las universidades como WebUNLP (su evolución en el sistema IDEAS), Portal America, entre otros. Estos entornos permiten complementar la tarea realizada en el aula. Los docentes pueden crear un curso en ellos y los alumnos puedan acceder a través de un navegador web, y, dependiendo de lo que permita el sistema y haya diseñado el docente, pueden tener algunas de las herramientas anteriormente nombradas en un solo lugar. Muchos de ellos cuentan con herramientas de autoevaluación con capacidad de generar preguntas de verdadero o falso y de opciones múltiples, con corrección automática, que son aprovechadas por numerosos docentes.

Por otra parte, los juegos —y los videojuegos, en particular— han constituido una herramienta para el aprendizaje del comportamiento y de la actitud, y han sido utilizados en la educación desde hace años (Marcano Lárez, 2014). A la hora de categorizar los juegos es útil tener en cuenta la función principal del juego. Así los juegos serios según Michael y Chen (2005) son aquellos juegos que se usan para educar, entrenar e informar (Abt, 1970). En (Marcano Lárez, 2014) se afirma que los últimos avances en las ciencias cognitivas en general apoyan a los principios del aprendizaje que los juegos incorporan. Destacan el carácter activo y el rol protagónico que adquiere el alumno en la resolución de problemas en tiempo real y con un *feedback* inmediato. Explican que los videojuegos siguen estrategias de diseño para generar ambientes virtuales que atrapen las capacidades perceptivas del usuario, que les produzcan gratificación sensorial (citando a Crawford, 2003) y que generen la sensación de inmersión en la que el jugador "sienta" que puede "participar". Esta interactividad que

ofrecen los videojuegos, logran la atracción, la inmersión, la emoción y motivación que son factores fundamentales en el proceso de aprendizaje de una persona.

Asimismo, en Herrero et al. (2014) también se explica que los juegos serios pueden ser útiles como herramientas metacognitivas para los estudiantes, o para los profesores con el fin de sensibilizarse más sobre el grado en que sus estudiantes son capaces de usar y aplicar conceptos y teorías científicas en una situación contextualizada y citan el trabajo de Nilsson y Jakobsson (2010).

No obstante, en Sung y Hwang (2013) se citan varios estudios que demuestran que sin un diseño apropiado el uso de juegos serios podría tener efectos negativos. En el mismo artículo puede verse cómo, con un diseño apropiado, las ventajas de los juegos serios pueden aumentar aún más. La parte psicológica del usuario, especialmente la relacionada con la motivación y el disfrute, es un punto importante a tener en cuenta cuando se diseñan juegos, como puede verse en el análisis de diferentes *frameworks* de diseño, en Mora Carreño, Riera, Gonzales, y Arnedo-Moreno (2015). El experimento de Giannakos (2013) en el cual se demuestra la importancia de estos conceptos en el aprendizaje mediante juegos serios refuerza esta hipótesis.

Dentro de la industria del videojuego, algo que ha ganado mucha fuerza es el área de juegos móviles. Uno de los motivos es el incremento que ha tenido la penetración de los *smartphones* en el mundo. Por lo que en varios ámbitos académicos se está aprovechando esto para el desarrollo de juegos serios educativos.

En este trabajo se propone la integración de un juego serio a un entorno virtual de enseñanza y aprendizaje (IDEAS) para acompañar a la autoevaluación del alumno en la aplicación de conceptos, recuperación de conocimientos abordados en un determinado proceso educativo, integración de temáticas, entre otros. El juego serio tomará preguntas o problemas a resolver que los docentes hayan creado en la herramienta de autoevaluación de IDEAS, pero que presentará al alumno en forma de desafíos a resolver como parte del juego, ofreciendo elementos lúdicos que contribuyan al mismo tiempo al entretenimiento, a la motivación, a su participación activa, y al *feedback* inmediato. El juego podrá ser accedido desde dispositivos móviles para su mejor aprovechamiento desde cualquier lugar y momento.

1.3. Metodología de trabajo propuesta

Para dar cumplimiento a los objetivos propuestos, se comenzará haciendo un relevamiento de la bibliografía para investigar el estado del arte en relación a la temática de juegos serios móviles en el escenario educativo, experiencias de utilización de juegos serios móviles en educación y herramientas para su desarrollo. También se revisarán antecedentes de juegos serios orientados a la autoevaluación de los alumnos.

Luego, basándose en este relevamiento, se procederá a realizar el diseño y desarrollo de un juego serio orientado a la autoevaluación de los alumnos, teniendo en cuenta diferentes juegos serios actuales que se correspondan con la temática de preguntas y respuestas. De esta forma se buscará aprovechar las tendencias actuales para hacer más atractivo el juego a desarrollar.

Por último, se investigará la integración del juego desarrollado con el sistema IDEAS. Para esto se estudiarán diferentes estrategias de integración (*web services*, desarrollo de *APIs*, etc.) para posibilitar la comunicación entre ambas aplicaciones. De esta manera se buscará favorecer el seguimiento del alumno a través de IDEAS y la construcción de consignas que el juego pueda tomar de dicho entorno, para generar desafíos que posibiliten la autoevaluación del alumno.

1.4. Resultados esperados

Del corriente trabajo se esperan obtener varios resultados:

- Realizar un estudio sobre la temática de juegos serios aplicados a la educación.
- Realizar un análisis comparativo sobre las diferentes herramientas existentes para poder realizar el desarrollo de un juego serio.
- Desarrollar como alternativa a la autoevaluación de los alumnos un juego serio que esté integrado al entorno virtual de enseñanza y aprendizaje IDEAS.

1.5. Estructura del trabajo

A continuación se detalla la estructura de esta tesina.

Capítulo 2. Se analizará el estado del arte en relación a los juegos serios orientados a escenarios educativos. Se comenzará definiendo el concepto de juego serio y su relación con la educación, particularizando los casos de juegos serios móviles orientados a la evaluación. Se finalizará el capítulo con un análisis de algunas experiencias y los resultados obtenidos de éstas.

Capítulo 3. Se analizarán distintos motores de juego. Se empezará por definir el concepto de motor de juego junto con su importancia en el desarrollo de videojuegos y su evolución a lo largo de la historia. Se proseguirá con una definición de requisitos deseables en función de las necesidades específicas del juego propuesto, y se continuará con el análisis de seis diferentes motores en base a estos requisitos. Finalmente se fundamentará la elección del motor elegido para el desarrollo propuesto en esta tesina.

Capítulo 4. Se dedicará este capítulo a describir el diseño del juego propuesto. Se comenzará introduciendo la finalidad con que se realiza el desarrollo del juego y se continuará con una descripción general de éste. Cada descripción de la funcionalidad del juego estará acompañada de imágenes a modo de ilustración.

Capítulo 5. Este capítulo estará destinado a la implementación del juego. Se comenzará explicando los aspectos técnicos de la implementación del juego. Se introducirán las descripciones pertinentes a la arquitectura usada, además de los aspectos de integración y la tecnología subyacente utilizada para el desarrollo.

Capítulo 6. Se expondrá el proceso de evaluación llevado a cabo así como también los resultados obtenidos tanto en las sesiones con alumnos como con docentes

Capítulo 7. Finalmente, en este capítulo, se presentarán conclusiones y trabajos futuros a realizar, en relación a la temática de la tesina.

CAPÍTULO 2

JUEGOS SERIOS Y EDUCACIÓN

2.1. Introducción

En este capítulo se especificará el estado del arte del estudio de juegos serios y su relación con la educación. Se empezará por introducir el concepto de juego serio y como éste fue evolucionando. Se continuará por explicar como los juegos serios han sido usados principalmente en el ámbito educativo y las ventajas que esta aplicación ha tenido según diferentes autores.

En la siguientes dos secciones se particularizará por un lado en los juegos serios que fueron creados con fines evaluativos y por otro en los juegos serios diseñados específicamente para dispositivos móviles.

Para finalizar el capítulo se analizarán diferentes experiencias educativas con juegos serios y se tendrán en cuenta aspectos tales como el contexto educativo, el tipo de software involucrado y los resultados de dichas experiencias. Estos análisis servirá como base para el diseño del juego propuesto en esta tesina.

2.2. Juegos Serios

La función principal de los juegos a lo largo de la historia siempre fue el entretenimiento. Es esta función lo que hace a los juegos tan atractivos mas allá del tipo de juego que se hable, o del público al que esté dirigido. Con esto no es raro encontrar áreas como la psicología que creen ambientes de juego para lograr mayor confianza en niños, como puede verse en la aplicación del juego de garabato de Fernández-Manchón García (2012). Partiendo desde el componente lúdico, se busca lograr un objetivo concreto diferente del entretenimiento.

Estas experiencias pueden catalogarse como juegos serios teniendo en cuenta la definición dada en Michael y Chen (2005) donde se dice que un juego serio es aquel que no tiene al entretenimiento como su objetivo principal.

Algo muy importante que diferentes autores fueron notando a lo largo de los años, es la estructura que suele tener un juego. Todos los juegos disponen de un conjunto de reglas que limitan las acciones del jugador a un grupo finito, las cuales solo podrá usar en un determinado momento y contexto. Los jugadores deben aprenderse estas reglas y combinarlas con los recursos con los que dispone en el momento para poder tomar la decisión más acertada y de esta manera ir superando las distintas etapas del juego. Se puede notar que en este punto, el jugador realiza un proceso de aprendizaje de las reglas para poder lograr este objetivo. No es de sorprender que con este contexto, muchos autores relacionan a los juegos serios con un fin educativo, como notan también Michael y Chen (2005) al dar otra posible definición de juego serio. Esto no es una sorpresa si también se tiene en cuenta la acuñación del término.

En 1970 el investigador Clark Abt publica el libro *Serious Games* en donde se usa por primera vez este término. En este libro, Abt cuenta diferentes experiencias que fue realizando a lo largo de años en distintos colegios. Estas experiencias consistían en realizar juegos con fines educativos para comprobar si existía una mejora en el aprendizaje de los alumnos. En este libro Abt ya define a los juegos serios como aquellos que poseen un propósito educacional explícito y cuidadosamente pensado, y no han sido concebidos para ser jugados principalmente como modo de entretenimiento.

Como se puede ver el concepto de juegos serios existe desde hace mas de 4 décadas. Un ejemplo de esto puede verse en el estudio hecho por Marcano Lárez (2014) en donde nota que los juegos han constituido una herramienta para el aprendizaje del comportamiento y de la actitud, y ha sido utilizado en la educación desde hace años. Es de importancia notar que hasta el momento es la primera vez que se utiliza el término videojuego. Los juegos serios abarcan cualquier tipo de juego. No obstante, con el enorme crecimiento que tuvieron los videojuegos desde sus inicios, se encuentra un buen lugar para la creación de juegos serios.

Pero ¿es el entretenimiento la única ventaja que traen los juegos serios? En Boyle, Connoly, y Hainey (2011) se exploran varias de los impactos positivos de los videojuegos mas allá del entretenimiento. Uno de estos impactos es la motivación, donde cita a Przybylski, Ryan, y Rigby (2009) para notar que los juegos cubren necesidades de competencia y autonomía que tienen las personas. Otro impacto notado por Boyle et al. (2011) es el de la satisfacción que traen consigo la inmersión y el flujo provistos por los juegos. En este punto, cita el trabajo realizado por Csikszentmihalyi (1990) en donde se muestra la importancia del flujo como parte de la sensación de agrado obtenida en la actividad de juego. El flujo es un concepto acuñado por el mismo Csíkszentmihályi y se refiere al estado en el cual se encuentra una persona cuando esta completamente inmersa en la actividad que realiza. Por

ultimo Boyle et al. (2011) dice que los videojuegos también tienen impactos cognitivos y perceptivos. La mayor evidencia acerca de la obtención de habilidades por jugar videojuegos viene de la investigación que sugiere que los juegos de acción tienen un impacto positivo en la coordinación visomotora, la representación espacial y en la atención visual.

Marcano Lárez (2014) consolida la hipótesis de estos impactos cuando cita las conclusiones de los últimos avances de las ciencias cognitivas. El segundo impacto se apoya al afirmar que estos avances apoyan a los principios del aprendizaje que los juegos incorporan. Destacan el carácter activo y el rol protagónico que adquiere el alumno en la resolución de problemas en tiempo real y con un *feedback* inmediato. Explican que los videojuegos siguen estrategias de diseño para generar ambientes virtuales que atrapen las capacidades perceptivas del usuario, que les produzcan gratificación sensorial (citando a Crawford, 2003) y que generen la sensación de inmersión en la que el jugador "sienta" que puede "participar". Por último reafirma el ultimo impacto al decir que es esta interactividad la que logra la atracción, la inmersión, la emoción y motivación que son factores fundamentales en el proceso de aprendizaje de una persona.

Sin embargo, los videojuegos también pueden tener impactos negativo. En Sung y Hwang (2013) se citan varios estudios que notan que el gran desafío es el de proveer soporte y guía a los alumnos mientras se busca un balance entre el aprendizaje y el juego y entre el desafío y las habilidades individuales del alumno. Boyle et al. (2011) lista algunos de estos impactos negativos, los cuales son: la video violencia, la estereotipación, y la adicción que pueden generar algunos videojuegos. Esto se condice con los estudios anteriomente nombrados donde concluyen que sin un diseño apropiado el uso de juegos serios podría tener efectos negativos.

2.3. Juegos serios orientados a la evaluación

En la sección 2.2 se pudieron ver cuáles son las ventajas que resultan atractivas de aplicar los juegos serios en escenarios educativos, y cómo estos se encuentran relacionados desde los inicios del primero. En esta sección se analizarán en particular aquellos juegos serios pensados para ayudar como instrumentos de evaluación.

El objetivo principal de la educación es que el alumno aprenda ya sea un concepto, una habilidad, una actitud, un valor, algo que le haya dado el significado al curso que se haya dado. Siempre es importante saber si los alumnos han pasado por este proceso de aprendizaje satisfactoriamente o si, por el contrario, el resultado no fue el esperado. Las investigaciones en el campo de la didáctica han llevado a considerar la evaluación como parte integrante del proceso de enseñanza, no como una etapa final o última, que implica para los estudiantes una toma de conciencia de los aprendizajes adquiridos, mientras que para los docentes es fuente de conocimiento, y conlleva la posibilidad de generar conclusiones válidas. Es posible conceptualizarla, en principio, como una práctica

sistemática, planificada e integrada a un proyecto educativo, que tiene como finalidad apreciar, conocer y comprender un aspecto de la realidad educativa. La evaluación debe generar propuestas de cambio y mejora de los aspectos u objetos evaluados (Cian, Sanz, y Zangara, 2009). La evaluación puede brindar datos suficientes para detectar en que conceptos se esta fallando y de que forma, y así reforzar la enseñanza en estos mismos o cambiar los métodos que se utilizan. No solo los datos sirven para medir el aprendizaje de los alumnos, también sirven para evaluar la eficacia que tienen distintos métodos aplicados en diferentes contextos con el fin de seguir mejorando permanentemente la educación.

Es claro que la evaluación constituye una parte importante de la planificación de una propuesta educativa. Ya en la década de 1970, una gran parte de investigación demostró que lo que más influía aprendizaje no era la enseñanza, sino la evaluación(Bezanilla et al. 2014 citando a Boud y Falchikov 2006). Bezanilla et al. (2014) también afirma que los alumnos enfocan su atención y organizan su forma de estudiar, adaptándola al tipo de evaluación. Por lo tanto, la evaluación debe ser diseñada cuidadosamente para que se convierta en una herramienta precisa que contribuya a la consecución de un aprendizaje profundo.

Hasta este punto solo se han relacionado a los juegos serios con el proceso de aprendizaje. La mayoría de los estudios se enfocan en este punto a pesar de lo dicho en el anterior párrafo La evaluación es normalmente una de las principales fuentes de insatisfacción entre los estudiantes universitarios(Ferrel, 2012). El enfrentar una evaluación es una situación estresante para cualquier persona. Sentirse probado y saber que el resultado de una propuesta educativa formal depende de esa nota puede ser contraproducente para el alumno y terminar teniendo un desempeño menor al que en realidad debería tener. Teniendo en cuenta esto, ¿por qué no aprovechar la satisfacción brindada por los juegos serios como estrategia para analizar el desempeño de un alumno o la comprensión de determinadas temáticas abordadas? Esta satisfacción podría disminuir en gran medida el estrés producido por una evaluación y mejorar el desempeño de un alumno. Bezanilla et al. (2014) es una de las investigaciones que trata de aprovechar esto. En este artículo se muestra la creación de un juego serio con el objetivo de que los alumnos adquieran algunas competencias, a la vez que aplica distintos métodos de evaluación según el escenario y el momento en que se encuentre el jugador.

Si bien la aplicación de los juegos serios en la evaluación no se ha extendido aún y se han encontrado pocos estudios que abordan esta temática, las investigaciones encontradas resultan prometedoras.

2.4. Juegos serios para dispositivos móviles

Los juegos serios pueden aparecer en distintas formas. Puede ser juegos de mesa, juegos a desarrollar entre los alumnos en el aula, o videojuegos. Pero incluso esta última categoría se puede dividir aún más si se tienen en cuenta las distintas plataformas y soportes con las

que se cuentan hoy en día.

Sin duda la opción mas popular como soporte para los juegos es la de la computadora. Entre los dispositivos que se pueden utilizar para los videojuegos, es sin duda la de mas fácil acceso. Hoy en día no es raro ver una o varias computadoras en los hogares. Y en el caso de que esto no sea posible, por cualquier cuestión que ocurra, es bastante factible encontrar el acceso a una a través del establecimiento educativo al que el alumno asiste o a través de una biblioteca cercana. Si bien todavía existen espacios que no tienen acceso a estos dispositivos, se visualizan en general políticas estatales o institucionales que buscan acortar estas brechas relacionadas con el acceso a las tecnologías digitales. Pero las computadoras no son el único medio sobre el cual pensar a la hora de hablar de los juegos serios.

Otra plataforma muy popular para desarrollar juegos serios es la consola de *Microsoft*, la *Xbox 360*. Esta consola posee un dispositivo adicional y opcional llamado *Kinect*. Este es un dispositivo de captura de movimiento creado especialmente para interactuar con la consola de juegos en tiempo real y sin necesidad de ningún equipamiento o vestimenta especial. Dependiendo de la aplicación, ciertos movimientos activarán diferentes acciones mediante la cual la persona interactuará con la aplicación. (Fernández-Baena, Susín, y Lligadas, 2012). Pastor, Hayes, y Bamberg (2012) dice que el dispositivo tiene no tiene una buena precisión si se lo compara con otros dispositivos de captura de movimiento, pero que esta precisión es muy buena teniendo en cuenta el precio al que se vende. Además el tamaño es ideal para tener en una casa, lo que lo hace una opción muy viable para ciertos trabajos de rehabilitación hogareños, estando en concordancia con lo que se propone en el trabajo de Fernández-Baena et al. (2012). Estos estudios son solo unos pocos ejemplos del uso de *Kinect* para juegos serios y sin duda el tema de la rehabilitación física es el predominante en esta plataforma a la hora de hablar de juegos serios.

Una plataforma que está surgiendo en los últimos tiempos para la creación de juegos, es la de los dispositivos móviles, en particular los celulares(Gross Salvat, 2009). En los últimos años el uso de celulares ha crecido exponencialmente. Según datos oficiales de la AFTIC (Autoridad Federal de Tecnologías de la Información y las Comunicaciones), en Argentina en el 2003, había alrededor de 8 millones de abonos telefónicos celulares. Este número se vio incrementado en un 500% para el 2008, llegando a 46 millones de abonos. Este increíble incremento fue mermando lentamente los siguientes años se vio reducido año a año, incrementándose solo un 4% en los últimos 3 años según datos de ENACOM (Ente Nacional de Comunicaciones), pero logrando aún así una penetración del 146%, es decir, casi un celular y medio por persona en el país (ENACOM, 2016). Sin embargo, a pesar de estos enormes números, es importante notar que no es fácil desarrollar una aplicación para estos dispositivos. Antes de la aparición de los *smartphones* las diferentes características de los dispositivos móviles, y los diferentes sistemas operativos imposibilitaban el desarrollo pensando en un mercado general, y obligaban al desarrollador a pensar en unos pocos dispositivos.

A partir de la aparición de los *smartphones* las variables se vieron reducidas enormemente. Los sistemas operativos se vieron reducido a un número muy pequeño. Según estudios como los de la compañía IDC, en el tercer cuatrimestre del 2016 un 99 % del mercado de móviles era abarcado por los tres sistemas operativos principales: *Android, IOS* y *Windows Phone*(IDC, 2016). Si a esto se le suma la aparición la aparición de tecnologías multiplataforma como *Ionic* o *React*, el desarrollador puede tener un alcance mucho mayor con un sólo desarrollo. Aún teniendo esta facilidad es importante notar que los *smartphones* no representan un número mayoritario entre los celulares. Según estudios de la consultora *eMarketer*, en Argentina en el 2016, aproximadamente un 37 % de las personas poseen un *smartphone* y estiman que para el 2020 este porcentaje se incrementará a un 70 % estando en concordancia con la tendencia mundial (eMarketer, 2016). Si bien no resulta el mayor mercado, queda claro que apostar por este sector pensando en el futuro es algo seguro.

A estos datos hay que sumarle la portabilidad que generan los juegos móviles. Teniendo estos datos en cuenta no resulta raro que varios investigadores hayan volcado sus trabajos hacia el sector móvil. En Sanchez, Mendoza, y Salinas (2009) se afirma que los dispositivos móviles brindan la posibilidad de aprender en cualquier lugar: mientras se camina, en la calle, en el transporte o en la escuela, creando de esta forma una evolución en el aprendizaje potenciado por la tecnología, gracias a la continuidad de distintas experiencias de aprendizaje en o pensando en un aprendizaje contextualizado (por ejemplo, una actividad educativa in situ relevando datos reales vinculados al contexto bajo estudio). En Gross Salvat (2009), citando a Klopfer (2009), se afirma que el uso de los juegos a través del móvil tiene un gran potencial educativo y formativo porque permite la creación de situaciones de juego en el aula muy flexibles y cambiantes. Se puede promover la capacidad de adaptar los juegos a un número de diferentes estilos tales como la competencia y la colaboración, crear situaciones en las que los jugadores aprendan a combinar juego y estrategias de comunicación, es posible jugar en equipo, etc.

2.5. Análisis de juegos serios relevados en la bibliografía

En las secciones anteriores se analizaron los juegos serios particularizando especialmente en aquellos que se orientan a la evaluación y aquellos diseñados para los dispositivos móviles. En esta sección se analizarán algunos juegos serios que sirvan de ejemplo de lo expresado previamente a lo largo de este capítulo. Para ello se presentan aquí algunos ejes de análisis para cada juego a revisar, de manera tal que todos sean considerados bajo la misma mirada.

- **Nombre del juego:** se informará en el título de la subsección el nombre del juego realizado. En caso de no contar con un título se intentará ilustrar el mismo mediante la idea principal con la que fue creado.

- **Título del artículo:** es importante notar en cada experiencia cuál es la fuente de donde se extrae la información a relevar.
- **Autores:** se listarán todos los autores del artículo analizado.
- **URL:** se pondrá un enlace al artículo correspondiente.
- **País:** es importante tener en cuenta el país en que se realizó dicha experiencia, ya que el contexto educativo difiere mucho entre países. Algo aplicable en un país puede no serlo en otro.
- **Contexto educativo:** también es importante considerar el contexto educativo de la experiencia. Esto es, el nivel sobre el que se aplicó, cantidad de alumnos y demás datos que sean pertinentes.
- **Tipo de *software* involucrado:** en este punto se revisará el tipo de software utilizado para la realización del juego aplicado.
- **Descripción:** se describirá brevemente el objetivo perseguido con el juego, junto con una descripción de su funcionamiento.
- **Resultados:** como toda experiencia, es importante nombrar los resultados obtenidos de ésta, para poder confirmar si se cumplieron o no los objetivos.

A continuación se presenta una subsección por cada juego a analizar.

2.5.1. Juego serio inmersivo

- **Título del artículo:** *Assessing Knowledge Retention of an Immersive Serious Game vs. a Traditional Education Method in Aviation Safety.*
- **Autores:** Chittaro, Luca y Buttussi, Fabio.
- **URL:** `https://www.computer.org/csdl/trans/tg/preprint/07014255.pdf`
- **País:** Italia.
- **Contexto educativo:** esta experiencia no se realizó en ningún ámbito educativo formal. Los participantes fueron voluntarios con estudios de diferentes niveles universitarios con edades entre 18 y 38 años.
- **Tipo de *software* involucrado:** se realizó la experiencia con un juego desarrollado específicamente para utilizar *hardware* de realidad virtual junto con un adaptador *bluetooth* para poder conectar el control de Nintendo *Nunchuck*.

- **Descripción:** el objetivo de esta experiencia es la de enseñar sobre las medidas de seguridad a tomar en un accidente de avión y un aterrizaje forzoso y lograr que este aprendizaje sea más duradero en el tiempo. Los participantes fueron divididos en 2 grupos, donde uno utiliza un método tradicional de aprendizaje sobre la temática, mientras que el otro grupo utilizó el juego serio de realidad virtual. Este juego pone al jugador en el papel de un pasajero de un vuelo que sufre un accidente aéreo y tiene que realizar un aterrizaje de emergencia. Diferentes acciones para realizar se le van ofreciendo al jugador en diferentes momentos del juego. Si alguna de estas acciones produce un error irreversible en la vida real, o este error ocurre por la omisión de una acción correcta, el juego mostrara un *feedback* negativo y volverá a una etapa previa a este error. Si por el contrario, una acción produce un error que puede ser revertido, el juego dejará proseguir durante 10 segundos, en los cuales los demás pasajeros le informarán de esto al jugador. Si éste decide no hacer caso de estos consejos, entonces el juego tratará esto de la misma forma que los errores irreversibles.

- **Resultados:** los resultados de la experiencia mostraron que en el corto plazo no había una gran diferencia entre lo aprendido por los 2 grupos. Si bien el grupo que utilizó el juego mostró resultados mejores, estos no fueron significativos. Con las pruebas realizadas una semana después de realizado el experimento se empezaron a notar las diferencias. Mientras que en el grupo que utilizó el juego no hubo diferencias en lo aprendido, en el otro grupo se evidenció un menor desempeño en este punto. Otras comparaciones realizadas fueron la del miedo a volar sentido a través de la experiencia y del compromiso de ambos grupos. Estos resultados mostraron que el grupo que utilizó el juego sintió un miedo mayor al del otro grupo, y a su vez mostró un mayor compromiso con la experiencia.

2.5.2. Juego educativo colaborativo con herramienta de ayuda mental

- **Título del artículo:** *A collaborative game-based learning approach to improving students learning performance in science courses.*

- **Autores:** Sung, Han-Yu y Hwang, Gwo-Jen.

- **URL:** `http://www.sciencedirect.com/science/article/pii/S0360131512002849`

- **País:** Taiwan.

- **Contexto educativo:** se trata de una experiencia en tres clases de sexto grado en una escuela primaria en el sur de Taiwan, con la participación de 93 alumnos.

- **Tipo de *software* involucrado:** en esta experiencia se utilizaron dos tipos de software diferentes. Por un lado, se utilizó un juego desarrollado mediante el sistema *RPG Maker*, el cual es una herramienta de desarrollo de juegos de rol publicada por

Enterbrain Incorporation. Por otro lado se utilizó un sistema de construcción del conocimiento colaborativo desarrollado con *Google Sites*, el cual es una herramienta para el desarrollo de sistemas estructurados colaborativos creado por *Google*

- **Descripción:** en este experimento todas las clases participantes hicieron uso del juego. La diferencia estuvo en las herramientas de apoyo y el uso de este mismo juego. El primer grupo hizo uso del juego en forma grupal y contó con el apoyo de una grilla de ayuda mental para ir ordenando lo aprendido. El segundo grupo también hizo uso del juego de forma grupal pero sin la ayuda de la grilla. Por último, el tercer grupo utilizó el juego de forma individual pero cada alumno tenía acceso a la grilla de ayuda.

  Este juego consistió en una aventura de rol situada en un reino antigüo donde la gente se comienza a enfermar por culpa del envenenamiento de un río. Después de estudiar unos libros médicos, el rey concluye que algunas plantas pueden ser la cura a esta enfermedad, y decide ir en busca de ellas. Estas plantas son las que los alumnos tienen que identificar y diferenciar.

  Todos los grupos tomaron un curso de dos semanas para obtener conocimiento sobres las plantas que posee la escuela a la que asisten. Luego de esto, se realizaron una serie de cuestionarios sobre sus actitudes hacia el aprendizaje, la motivación que poseen, y la autoeficacia de aprendizaje en grupo. A su vez, los alumnos hicieron un *pre-test* sobre los conocimientos adquiridos en clase. A continuación realizaron la experiencia según lo correspondiente a cada grupo, para la cual tuvieron un tiempo límite de 100 minutos. Al finalizar la experiencia los alumnos realizaron un post-test para medir sus logros de aprendizaje y si hubo algún cambio con respecto al cuestionarios realizado anteriormente. Además los alumnos también completaron una encuesta para poder medir la carga cognitiva que les significaba el juego serio.

- **Resultados:** para los resultados de esta experiencia se compararon los datos obtenidos a partir de los cuestionarios iniciales de actitud y motivación y *pre-test* y los post-tests y en base a esto se observó el avance de cada uno de los grupos. El primer grupo, el cual pudieron usar el juego de manera grupal y con la ayuda de la grilla fue el que tuvo el mayor avance con respecto a los demás. Este grupo supero significativamente a los otros en todos los puntos (actitud, motivación, conocimientos y autoeficacia). De esta forma los investigadores concluyeron que una combinación bien planificada de juego serio con herramientas de ayuda mental puede potenciar los beneficios dados por ambas estrategias. Si bien, ambas estrategias tienen sus ventajas, bien combinadas pueden traer todavía mas beneficios.

2.5.3. Gem-Game

- **Título del artículo:** *Enjoy and learn with educational games: Examining factors affecting learning performance.*

- **Autores:** Giannakos, Michail.
- **URL:** `http://www.sciencedirect.com/science/article/pii/S0360131513001565`
- **País:** Grecia.
- **Contexto educativo:** esta experiencia se divide en dos estudios. El primer estudio se realizó en una escuela estatal en Grecia y se dividió a los alumnos de un grado en dos grupos teniendo un total de 41 alumnos. El segundo estudio se realizó en otra escuela estatal de Grecia y esta vez no se dividieron a los alumnos. Este estudio se realizó con un total de 46 alumnos intervinientes. Todos los alumnos que realizaron ambos estudios tenían 13 años de edad.
- **Tipo de *software* involucrado:** se utilizó un juego desarrollado por los mismos investigadores. El *software* utilizado fue realizado mediante *Flash*.
- **Descripción:** para el primer estudio se tomaron a los alumnos de dos clases de una escuela, totalizando 41 alumnos. A cada alumno se le realizó un *pre-test* confeccionado con la ayuda de los docentes de ambas clases. Estos resultados fueron utilizados para dividir de forma pareja a los alumnos en dos grupos de similar desempeño. Un grupo fue el experimental y el que iba a utilizar el juego serio, y el otro grupo realizó el aprendizaje de forma tradicional. Para minimizar el entusiasmo de los alumnos, ambos grupos utilizaron juegos educativos durante las 4 semanas anteriores a la experiencia. Estos juegos contaban con la misma tecnología y las mismas mecánicas que *Gem-Game* pero sin tener objetivos matemáticos. La experiencia se realizó a lo largo de 2 semanas. Este estudio tuvo el objetivo de verificar la eficacia del aprendizaje mediante el uso de juegos serios con respecto a la enseñanza tradicional.

  En el segundo estudio se tomaron a todos los alumnos de dos clases de otra escuela, totalizando 46 alumnos. Este estudio se realizó con el objetivo de verificar la actitud de los alumnos hacia el aprendizaje mediante el uso de juegos serios. En este caso se analizaron 4 características: el disfrute, la intención de usar el juego, la felicidad y el rendimiento en el mismo. A su vez se elaboraron estadísticas para comprobar la influencia de cada una de estas características en las demás. La experiencia se realizó durante una hora en la que los alumnos podían usar el juego y luego se desarrolló una encuesta en la cual los alumnos calificaban sus experiencias personales en el juego en una escala de 1 a 5, siendo 5 el puntaje máximo. En el caso del rendimiento se utilizó una prueba confeccionada con los docentes. Al finalizar se realizaron entrevistas con los alumnos y los profesores para preguntarles su opinión sobre la experiencia.

  El juego desarrollado cuenta la historia de Peter, un chico que se encuentra buscando a su perro Lucky, el cual ha sido secuestrado. Un hada le ofrece ayuda a Peter para poder encontrar a su perro, pero para eso deberá recolectar 30 diamantes a lo largo de 3 niveles. Cada nivel se presenta con diferentes alturas y para pasar de una altura

a otra el jugador deberá realizar una cuenta que lo lleve de la altura actual, a la que se desea ir. Es decir, si Peter se encuentra en la altura 6, el jugador deberá escribir -5 si desea llegar a la altura 1 y de esta forma Peter podrá recolectar el diamante que se encuentra en dicha altura. Cada nivel se completa una vez que se hayan recolectado 10 diamantes. El primer nivel solo cuenta con números enteros positivos. El segundo nivel solo posee números enteros negativos. Y el último nivel posee ambos tipos de números.

- **Resultados:** en el primer estudio se realizó un examen a ambos grupos posterior a la realización de la experiencia. Los resultados de ambos grupos no tuvieron mayores diferencias entre sí, con lo cual se concluyó que la eficiencia con la utilización de juegos serios no varía de aquella con un método de enseñanza tradicional.

  En el segundo estudio los resultados arrojaron que el rendimiento de los alumnos iba en aumento a medida que el disfrute de ellos aumentaba. Por otro lado la felicidad iba decreciendo a medida que el rendimiento aumentaba aunque los datos arrojados dieron muestra de que este decremento podía considerarse insignificante. Por lo tanto de los datos obtenidos, solo el disfrute se consideró que tenía la suficiente importancia en el rendimiento como para tenerlo en cuenta. En la entrevista, los alumnos señalaron que la experiencia fue positiva y que les gustaría contar con otras experiencias del mismo tipo en asignaturas diferentes. Por el lado de los docentes, ellos llegaron a la misma conclusión de los investigadores a través de los datos. Al mismo tiempo, se mostraron sorprendidos en relación a la motivación de los alumnos respecto de las clases tradicionales

2.6. Conclusiones del capítulo

En este capítulo se han resumido aspectos centrales en relación a los juegos serios orientados al escenario educativo, a partir de la revisión de diferentes artículos de revistas académicas y otras fuentes bibliográficas.

El análisis permite resaltar la vinculación existente entre los procesos educativos y los juegos, la conceptualización de juegos serios y sus posibilidades para la educación, y al mismo tiempo, se vislumbra la oportunidad que este tipo de juegos pueden ofrecer para los procesos de evaluación de los alumnos, mejorando las situaciones de estrés que provoca evaluaciones más tradicionales.

Finalmente, se han relevado 3 juegos utilizados con fines educativos. Si bien son pocos los juegos analizados, en todos los casos los juegos han dado resultados positivos para alguna de la variables de medición. Esto y los antecedentes revisados en los artículos consultados permiten augurar potencialidades de los juegos serios para los procesos de enseñar y aprender. Principalmente, se han observado mejoras en la motivación, la actitud, y la durabilidad de los conocimientos/habilidades adquiridas. Los juegos analizados

mayormente consistían en desarrollos propios y *ad-hoc*. Se han revisado experiencias de Italia y Grecia (dos países europeos) y en Taiwán (un país asiático). No se han visto aspectos relacionados con el contexto que resulten significativos para esta tesina.

CAPÍTULO 3

CARACTERIZACIÓN Y ANÁLISIS DE MOTORES DE JUEGO.

3.1. Introducción

En este capítulo, se introducirá el concepto de motor de juego y su importancia a la hora del desarrollo de un videojuego. Al mismo tiempo se presentarán sus características y componentes principales. Se proseguirá haciendo un análisis de diferentes motores seleccionados a partir de una serie de requisitos definidos y que se vinculan con la necesidad que origina el juego propuesto en este trabajo. A su vez se plantean algunos criterios deseables pero no excluyentes para la selección que se tendrán en cuenta a la hora del análisis.

En la siguiente sección, se fundamentará la elección de un motor de juegos específico para el desarrollo del juego propuesto aquí y se finalizará el capítulo con un resumen de éste.

Este capítulo se ha planteado de forma que el lector pueda entender las diferentes características que pueden poseer los motores de juego y la necesidad de realizar un análisis a la hora de tener que elegir un motor que se ajuste a las necesidades del contexto al momento de crear un juego o juego serio.

3.2. Definición

Decir que es un motor de juego no es algo sencillo. A través de los años se ha discutido mucho y todavía no se ha podido llegar a un consenso sobre esto. Bethke (2003) define al motor de juego como una serie de rutinas que permiten la representación de todos los elementos del juego. Es importante resaltar la parte de representación de los elementos. El motor solo se encarga de definir cómo los elementos serán visualizados y el comportamiento que tendrán los mismos en su entorno. La definición provista por Lewis y Jacobson (2002) extiende este concepto al decir que los motores de juego son una colección de módulos de código de simulación, que no especifican directamente el comportamiento del juego (lógica del juego) ni el entorno del juego (información de niveles). Es el desarrollador del juego quien se encargará de crear y especificar estos aspectos.

Los motores presentan una serie de componentes que los caracterizan, las posibilidades de cada uno de estos componentes son los que hacen la diferencia entre un motor y otro. En este punto también hay disenso entre los diferentes autores pero Pereira (2014) presenta una clasificación posible de estos componentes: 1) el motor gráfico, 2) el motor de sonido, 3) el motor de físicas, 4) el gestor de Inteligencia Artificial (IA) y 5) el control de interacción. A continuación se describe cada uno de ellos:

1. Motor Gráfico: el motor gráfico es quien se encarga de realizar la comunicación con el hardware gráfico para poder crear el mundo visual presentado al jugador. La potencia de los motores gráficos es muy amplia y van del rango de crear mundos simples 2D, hasta poder crear mundos muy complejos con visuales 3D que cada vez sorprenden más por el realismo logrado. Es importante elegir bien qué tipo de mundo se desea y qué tan real se verá este. Mientras más real se vea el mundo se necesitará más procesamiento, lo cual podría dejar muchos dispositivos y jugadores fuera de la posibilidad de utilizar el juego.

2. Motor de sonido: el motor de sonido cumple una función muy parecida. Es quien se comunica con el hardware de audio para poder reproducir los diferentes audios con que cuenta el juego. Además brinda facilidades para que el desarrollador gestione este audio generalmente vinculado con los sonidos y la música del juego.

3. Motor de físicas: el motor de físicas es un componente muy particular. Es el encargado de simular la física del mundo real en el mundo creado por el motor. Como el motor gráfico, la potencia de este motor puede variar en un rango tan amplio como simular la gravedad o unas colisiones simples, hasta simular el movimiento del agua en arroyos, ríos, etc. También hay que tener en cuenta que el procesamiento realizado por este motor suele ser de las más demandantes. Los motores de físicas suelen ser tan complejos que pueden requerir un desarrollo entero aparte, a tal punto que el propio

motor de desarrollo del videojuego muchas veces puede hacer uso de diferentes motores de físicas desarrollados por terceros.

4. Gestor de Inteligencia Artificial: el gestor de IA es el encargado de brindar las facilidades para que el desarrollador pueda crear la inteligencia artificial que pueden llegar a tener los diferentes elementos del juego. También es el encargado de interpretar esta información y lograr que el elemento actúe de acuerdo a ella.

5. Control de interacción: por último se encuentra el control de interacción, el cual es el encargado de brindar la interfaz mediante la cual el jugador se comunica con el juego. Cabe aclarar que la interfaz no solo se refiere a la forma de realizar las diferentes acciones propias del juego a través de una interfaz gráfica, sino también de poder comunicarse con diferentes dispositivos de entrada como pueden ser el *mouse*, el *touchpad*, el *joystick* o el teclado.

3.3. Evolución de los motores

En los primeros videojuegos las empresas desarrolladoras tenían que empezar cada proyecto desde cero. A medida que crecieron estos tipos de desarrollos, se fue trabajando en separar cierta lógica en *scripts* para facilitar el trabajo, además de poder reutilizarla para futuros proyectos y evitar tener que reinventar la rueda cada vez. Estos *scripts* fueron los primeros motores de juego.

Un ejemplo de esto fue *Script Creation Utility for Maniac Mansion* o *SCUMM*, creado por *Lucas Arts*. Como lo dice su nombre, *SCUMM* fue creado para la aventura gráfica *Maniac Mansion*, videojuego que vio la luz en 1987. A pesar de esto, el *script* continuó siendo utilizado hasta 1998 y en otros 11 proyectos distintos como puede verse en el libro de Smith (2008).

En 1993 *ID Software* creó el primer motor de juego propiamente dicho, el *Doom Engine*, posteriormente rebautizado como *id Tech 1* para seguir una lógica de numeración con los posteriores motores de la empresa. Este motor fue el primero en brindar los elementos con los que cuentan los motores de hoy en día. Algo para notar es que si bien, los elementos de este motor parecían ser 3D, en realidad todos los juegos eran 2D y el mismo motor simulaba esa visual. Éste solo fue utilizado por la misma empresa, algo bastante común en la época, y no fue hasta 1999, momento en que el motor fue lanzado bajo la licencia GNU, que otras empresas o personas pudieron hacer uso del motor.

En 1998 la empresa de videojuegos *Epic Games*, crea uno de los motores más utilizados hasta el día de hoy: el *Unreal Engine*, para el juego homónimo. Este fue uno de los primeros motores en ser utilizado por otras empresas lo que, junto con su poder y facilidad de uso, consiguió que sea utilizado para más de 400 juegos y de esta forma lograr el record *Guinnes* del Motor de juego más exitoso de la historia. En las primeras dos versiones el motor contaba

con una licencia propietaria que solo permitía usarlo a aquellos que podían pagarlo. Sin embargo, la aparición de *Unity* hizo que esto empezara a cambiar en la versión 3, y lo haga definitivamente en la versión 4.

Unity 3D apareció en 2005 y fue concebido con la idea de crear un motor profesional y accesible en términos económicos para desarrolladores *amateurs*. Esto le valió crecer rápidamente en popularidad entre la comunidad independiente y académica. *Unity* cuenta con diferentes tipos de licencias, contando siempre con una versión gratis que libera el uso del motor para cualquier persona, pero limita otros aspectos no necesarios como acceso al código fuente, control de estadísticas, o un soporte más rápido y especializado para resolver diferentes problemas.

Este tipo de licencia a medida del desarrollador y la popularidad que esto produjo, ocasionó que otros motores tomaran medidas similares. *Unreal Engine* en su tercer versión decidió generar una opción gratuita llamada *Unreal Development Kit*, que al igual que *Unity* limitaba ciertos aspectos, aunque en una mayor medida, ya que condicionaba opciones como poder generar el videojuego para ciertas plataformas. Ya en la versión 4 de *Epic Games* cambia totalmente la licencia. Bajo el lema "*we succeed when you succeed*" (nosotros triunfamos cuando vos triunfas), el motor pasa a ser gratuito, solo teniendo que pagar un porcentaje de regalías dispuesto por la empresa cuando se excede un monto fijo, también dispuesto por la empresa.

Los motores de juego han sufrido grandes cambios a lo largo de su historia. Siendo originalmente desarrollados para simplificar el desarrollo de las empresas pasaron a ser productos libres para todo el mundo debido en parte al auge que empezaron a tener en el mercado y en el mundo académico.

3.4. Definición de requisitos para la selección de un motor específico para el juego propuesto

Actualmente la cantidad de motores disponibles es muy grande, cada uno contando con sus propias licencias, potencia, facilidad de uso y demás características. Actualmente en *GitHub* hay 24 repositorios correspondientes a un motor cada uno. 2 de estos repositorios corresponden a uno de los 23 distintos motores HTML5 que existen hoy en día. Otro aspecto a tener en cuenta es que si bien la mayoría de los motores permite crear cualquier tipo de juegos, algunos están mejor preparados y/o presentan facilidades para ciertos géneros de juego (Nomdedeu, Díaz, y Fava, 2015). Con toda esta oferta es necesario definir requisitos que permitan realizar un análisis adecuado del motor de juegos acorde a los requerimientos específicos de la propuesta aquí realizada.

Para este trabajo se definieron los siguientes requisitos deseables en función de características propias de este tipo de herramientas de desarrollo y las necesidades

específicas del juego propuesto:

- **Facilidad de uso:** se requiere que el motor de juego sea fácil de usar, para reducir el tiempo requerido para poder lograr el funcionamiento del juego. En general, se busca un equilibrio entre facilidad de uso para el desarrollador y la flexibilidad y potencia de desarrollo que ofrece el motor. Este requisito será considerado para la selección del motor de juegos a utilizar.

- **Portabilidad a dispositivos móviles:** como el juego fue pensado para usar en dispositivos móviles, es necesario que ofrezca la posibilidad de crear el juego para dispositivos móviles y en particular interesa que funcione en *Android*. Es deseable que también el juego pueda ser utilizado en dispositivos *IOS*.

- **Curva de aprendizaje:** es deseable que el motor tenga una curva de aprendizaje buena, para poder lograr avances desde el principio, y no cuando recién se entienda completamente el funcionamiento del motor.

- **Interprete de código HTML:** debido a que el juego se integrará con las autoevaluaciones de un EVEA específico y las preguntas y respuestas de dichas evaluaciones suelen estar guardadas en código html, se definieron 2 alternativas para resolver esta complicación: realizar un *parser* para quitar todas las etiquetas HTML del código o que el motor permita interpretar el código HTML recibido para evitar modificaciones de éste. Para evitar realizar el *parser*, es una característica deseable, pero no excluyente, que el motor tenga esta característica.

- **Consumo de recursos del sistema:** el motor debe tener un funcionamiento fluido y no consumir una gran cantidad de recursos ya que se destinará a un juego con dispositivos móviles. Estos dispositivos cuentan con capacidades limitadas en cuanto a memoria y poder gráfico. También esto será importante para no dificultar el desarrollo del juego.

- **Motor de gráficos con vista 3D:** es deseable que el motor de gráficos posea la capacidad de crear mundos 3D para brindar una mejor experiencia al usuario, y una innovación con respecto a otros juegos de preguntas y respuestas. En caso de no contar con esta característica es requerido que el motor posea por lo menos la habilidad de crear mundos 2D con vista isotrópica.

- **Licencia gratuita:** se requiere que el motor posea al menos algún tipo de licencia gratuita sin limitar ningún tipo de elemento que se relacione con el desarrollo del juego.

3.5. Análisis de diferentes alternativas de motores de juegos

Con los requisitos ya definidos se analizaron los siguientes motores como alternativas a seleccionar para el desarrollo propuesto en esta tesina.

3.5.1. Phaser[1]

Es un motor HTML5 nacido en octubre del 2013. Entre los motores HTML5 se encuentra entre los de mayor reputación y entre lo más descargados en *GitHub*. Actualmente se encuentra en la versión 2.6, mientras que la versión 3 del motor se encuentra en desarrollo.

- **Facilidad de uso:** es un motor de fácil uso con el cual se pueden lograr resultados muy rápidamente. Cuenta con una gran cantidad de documentación, entre la que se encuentra tutoriales propios y de terceros (entre ellos el sitio de desarrolladores de Mozilla). Cuenta con un editor online propio que facilita el desarrollo, ya que permite visualizar en tiempo real la interpretación del código, aunque su uso no es obligatorio ya que se puede realizar el desarrollo con cualquier editor de texto.
- **Portabilidad a dispositivos móviles:** el motor no cuenta con una funcionalidad propia para exportar el juego a dispositivos móviles. Sin embargo, debido a la forma de desarrollar el juego como una página web, el motor posee flexibilidad para usar cualquier herramienta que sirva para exportar un sitio web como aplicación nativa o híbrida. Algunos ejemplos de las herramientas que se pueden usar son: *Intel XDK, Apache Cordova, Mighty Editor, Ionic* y *CocoonJS*.
- **Curva de aprendizaje:** debido a la rapidez con que se pueden lograr resultados, el motor posee una curva de aprendizaje aceptable para aprender lo básico. Al ser un motor puramente HTML5 y JS, dependerá de la habilidad del programador el poder construir juegos de gran complejidad.
- **Interprete de código HTML:** el motor, a pesar de ser HTML5, no cuenta con ningún tipo de herramienta o visualizador para interpretar código HTML.
- **Buen funcionamiento:** el motor es muy liviano y 2D, lo que hacen que sea muy rápido en cuanto a funcionamiento.
- **Motor de gráficos con vista 3D:** el motor no cuenta con la capacidad para renderizar elementos 3D. Sin embargo, cumple con el requisito de poder realizar juegos 2D con vista isotrópica.
- **Licencia gratuita:** el motor cuenta con una única licencia de tipo MIT (Massachusetts Institute of Technology).

[1] `http://phaser.io/`

3.5.2. Construct[2]

Es un motor HTML5 creado por la empresa *Scirra* en 2011. Actualmente, se encuentra en su segunda versión con el *release* r239, que permite crear mundos en vistas 2D. Es presentado como un motor que se puede usar sin necesidad de saber de programación, y tampoco se necesita escribir una línea de código. Se maneja con eventos y estas características hacer que sea muy fácil de usar, y que presente una curva de aprendizaje muy amigable para el desarrollador.

- **Facilidad de uso:** el motor se encuentra entre los más fáciles de usar en el mercado. Su enfoque está en poder crear juegos sin la necesidad de escribir una línea de código. Se maneja con eventos que ocurren en la pantalla general o en los diferentes elementos con los que cuenta el juego.

- **Portabilidad a dispositivos móviles:** este motor cuenta con unas posibilidades parecidas a *Phaser* aunque más reducidas. No cuenta con la misma libertad de *Phaser* de poder usar cualquier herramienta, pero es compatible con las más populares como *Apache Cordova, Intel XDK* y *CocoonJS*.

- **Curva de aprendizaje:** el hecho de estar apuntado a un público general sin conocimientos de programación hace que el motor tenga una curva de aprendizaje muy aceptable con el usuario, permitiendo dominarlo rápidamente.

- **Interprete de código HTML:** el motor no cuenta actualmente con un intérprete propio de código HTML. Sin embargo, posee un *plugin* para este propósito. Este *plugin* puede mostrar el contenido de una página que se accede a través de una URL, o puede contener un código HTML suelto el cual interpretará. Tiene el problema de que el código HTML debe ser ingresado entre comillas dobles, pudiendo generar problemas con algunos *tags* que posean estas comillas. Para evitar problemas es necesario realizar un *parser* que reemplace todas las comillas dobles por simples.

- **Buen funcionamiento:** el motor es muy liviano, y al formar su propio código en base a los eventos programados por el desarrollador, éste se encuentra siempre de forma optimizada.

- **Motor de gráficos con vista 3D:** al igual que *Phaser* el motor solo puede *renderizar* vistas 2D. Sin embargo, cumple con la condición de la vista isotrópica.

- **Licencia gratuita:** el motor cuenta con tres tipos de licencias, de las cuales dos son pagas. La versión gratuita limita la cantidad de elementos que el juego puede tener.

[2] `https://www.scirra.com/construct2`

3.5.3. PlayCanvas[3]

Es un motor HTML5 creado en 2011. Cuenta con la particularidad de no realizar ninguna descarga ya que todo el desarrollo se realiza en el navegador. Cuenta con proyectos de tipo públicos y privados.

- **Facilidad de uso:** el motor cuenta con una gran cantidad de tutoriales propios que lo hacen fácil de usar para cualquier persona que ya haya usado anteriormente un motor de juego.
- **Portabilidad a dispositivos móviles:** el motor no cuenta con funcionalidad propia para exportar el juego a dispositivos móviles, y no cuenta con las mismas libertades con las que cuentan *Phaser* y *Construct*. Sin embargo, existe la posibilidad, recomendada por el sitio del motor, de usar *CocoonJS* para esto.
- **Curva de aprendizaje:** la curva de aprendizaje del motor resulta costosa para desarrolladores principiantes de videojuegos. Sin embargo, es muy similar a otros motores 3D por lo cual para desarrolladores más avanzados puede resultar más fácil de aprender a usar.
- **Interprete de código HTML:** al igual que *Construct* el sistema no cuenta con un intérprete propio de código HTML, pero si con una librería que permite crear un *widget* para este propósito. Esta librería es muy completa y no se encontró ningún problema con su uso.
- **Buen funcionamiento:** al ser un motor que corre exclusivamente en el navegador, este tiende a ser lento cuando se encuentran muchos elementos en pantalla. En navegadores pesados en consumo de RAM como *Chrome*, esto puede ser un problema que dificulte el desarrollo de cualquier juego.
- **Motor de gráficos con vista 3D:** el motor es uno de los pocos motores HTML5 que posee la capacidad de generar vistas en 3D.
- **Licencia gratuita:** el motor, al igual que *Construct*, cuenta con tres tipos de licencias, de las cuales dos son pagas. Sin embargo, la licencia gratuita no limita ningún elemento del desarrollo más allá de la imposibilidad de crear proyectos privados.

3.5.4. eAdventure[4]

Es un motor creado en 2006 en la Universidad Complutense de Madrid. Sanz-Troyano, Torrente, Moreno-Ger, y Fernández-Manjón (2010) expresan que su objetivo principal es el de facilitar el desarrollo de videojuegos para personas sin conocimientos técnicos,

[3] `https://playcanvas.com/`
[4] `http://e-adventure.e-ucm.es/`

especialmente alumnos y profesores. Para reducir el tiempo de desarrollo, y la complejidad de la herramienta, el motor hace foco en los juegos correspondientes al género de *point-and-click*, el cual a su vez, es un subgénero de los videojuegos de aventura. Algo muy importante es que el motor agrega un valor educativo al realizar una evaluación automática, o un informe de lo realizado por el alumno, con el objetivo de generar nueva información para el docente. Por último, es importante destacar que los juegos realizados por el motor pueden ser encapsulados como un paquete estandarizado en diferentes formatos[5], lo que permite su integración con diferentes EVEA, así como su almacenamiento en repositorios de contenidos.

- **Facilidad de uso:** al estar enfocado en permitir crear juegos sin la necesidad de tener un conocimiento técnico, este motor se puede comparar con *Construct*. Al igual que ese, este motor es muy fácil de usar y se maneja principalmente por eventos.

- **Portabilidad a dispositivos móviles:** el motor no cuenta actualmente con una forma de portar los juegos a dispositivos móviles. Lo más cercano es exportar el juego para ser usado vía web y que se acceda a la URL del juego mediante el dispositivo móvil.

- **Curva de aprendizaje:** este es otro punto donde el motor se puede comparar con *Construct*. Su curva de aprendizaje es muy amigable, y permite crear juegos completos fácilmente. Al estar limitado solo al género *point-and-click* hace que también se eliminen objetos que no son necesarios, lo cual simplifica aún más el desarrollo. Un punto en contra, es que al estar el motor tan enfocado en el aspecto académico, éste no es muy conocido, por lo que no existe prácticamente documentación fuera del sitio oficial.

- **Interprete de código HTML:** el motor cuenta con la posibilidad de agregar documentos HTML de forma nativa. Incluso esto es promocionado como uno de sus puntos más atractivos para el docente.

- **Buen funcionamiento:** al ser un motor que se basa en imágenes fijas éste no consume muchos recursos, por lo cual todos los juegos funcionan de forma fluida.

- **Motor de gráficos con vista 3D:** debido al género al cual está limitado, el motor cuenta solo con gráficas 2D, sin vista isotrópica y sin posibilidad de hacer un mundo 3D.

- **Licencia gratuita:** al ser un motor dedicado al ámbito educativo,posee una licencia gratuita de tipo GNU LGPL (GNU *Lesser General Public License*).

[5]IMS Content Packaging, SCORM v1.2, SCORM 2004,y permite también empaquetarlo para publicarlo en WebCT y AGREGA.

3.5.5. Unreal Engine[6]

Es uno de los motores más usados y con mayor trayectoria en el mercado. Creado en 1998 por *Epic Games*, el motor cuenta con una gran comunidad de desarrolladores, dentro de la cual se pueden encontrar grandes empresas o desarrolladores independientes. Presenta una excelente documentación propia y de terceros para facilitar el desarrollo. También posee, en forma experimental, un *widget* de interprete HTML, el cual puede interpretar una URL o porciones de código HTML.

- **Facilidad de uso:** el motor cuenta con dos formas de desarrollar. Una forma es como *Construct* pero más avanzada. Los eventos y las acciones se van encadenando de una forma más intuitiva que con *Construct*. La otra forma le agrega a la anterior la posibilidad de incorporar código propio del desarrollador para poder tener mayor flexibilidad. Su facilidad de uso, hace que muchos desarrolladores lo consideren el más cómodo del mercado.

- **Portabilidad a dispositivos móviles:** el motor cuenta con la capacidad propia para exportar cualquier proyecto a una aplicación móvil nativa.

- **Curva de aprendizaje:** el motor cuenta con el mismo problema que *PlayCanvas*. La curva de aprendizaje para desarrolladores avanzados es aceptable, y los desarrolladores se acostumbran muy fácilmente a este motor, pero es desafiante para los desarrolladores principiantes que recién se inician en la implementación de videojuegos.

- **Interprete de código HTML:** el motor cuenta actualmente con un *plugin* propio experimental para generar *widgets* que interpreten ya sea URLs o código HTML. No se encontró ningún problema a la hora de usar este *plugin* a pesar de encontrarse en un estadío experimental.

- **Buen funcionamiento:** a pesar del gran poder con el que cuenta el motor, éste tiene un funcionamiento muy bueno, el cual hace fluído tanto el trabajo de desarrollo como el resultado final.

- **Motor de gráficos con vista 3D:** el motor gráfico de *Unreal* es 3D y es uno de los más poderosos motores en el mercado siendo utilizado frecuentemente para animaciones en todo tipo de ámbito.

- **Licencia gratuita:** el motor cuenta con un solo tipo de licencia la cual se basa en regalías. La empresa cobra un porcentaje de lo recaudado por el juego cuando se excede un monto de ganancias establecido por la empresa.

[6] `https://www.unrealengine.com/`

3.5.6. Unity[7]

Unity es actualmente el motor más utilizado en el mundo. Fue creado en el año 2005 con la idea de brindar un motor potente para el uso de todo el mundo. Al estar enfocado al público general desde su creación, la comunidad que utiliza el motor fue creciendo rápidamente, con lo cual, cuenta actualmente con la comunidad más grande entre los motores de juego. Prueba de esto son las 33000 preguntas que se tienen sobre el motor en *stackoverflow* contra las 1500 que posee *Unreal Engine*.

- **Facilidad de uso:** la gran cantidad de tutoriales propios, y de terceros, junto con la enorme documentación y resolución de problemas que se encuentra a través de toda Internet, hace de este uno de los motores más fáciles de usar que hay en el mercado.

- **Portabilidad a dispositivos móviles:** al igual que *Unreal* cuenta con la capacidad propia para ya exportar cualquier proyecto a una aplicación móvil nativa.

- **Curva de aprendizaje:** como es un motor pensado para el público general, cuenta con una curva de aprendizaje adecuada para los principiantes. Sin embargo, ser experto en el uso del motor puede ser algo que puede llevar mucho tiempo.

- **Interprete de código HTML:** el motor no cuenta actualmente con una forma propia de interpretar código HTML. Hasta mediados del 2016 existía un *plugin* de terceros muy completo el cual fue descontinuado y su descarga fue eliminada. Posee otros *plugins* que intentan solucionar este problema pero ninguno lo hace de forma apropiada, ya que solo poseen la capacidad de interpretar URLs o se ven limitados a un número reducido de dispositivos que no cubren los requeridos para este proyecto.

- **Buen funcionamiento:** el motor cuenta con un buen funcionamiento a la hora de del desarrollo. Dado que permite migrar el proyecto a muchas plataformas, es el mismo motor el que se encarga de optimizar el juego para la plataforma seleccionada.

- **Motor de gráficos con vista 3D:** al igual que *Unreal*, este motor cuenta con la capacidad de poder generar vistas 3D. Si bien el motor de *Unity* no es tan potente como el de *Unreal*, es un buen motor de gráficos y permite en general alcanzar los objetivos del desarrollo propuesto aquí.

- **Licencia gratuita:** el motor cuenta actualmente con cuatro tipos de licencias, de las cuales tres son pagas. En la licencia gratuita no se limita ningún elemento que pueda dificultar el desarrollo del juego propuesto aquí.

[7] `https://unity3d.com/`

3.6. Selección de un motor de juegos para el juego Desafiate

Como se pudo ver en la sección anterior, entre los motores seleccionados, ninguno cumple con todos los requisitos anteriormente definidos. Es por eso que se analizó cada caso para ver cuál era el que mayores ventajas ofrecía.

Los primeros dos motores descartados fueron *Phaser* y *Construct*. Ambos motores contaban con la problemática de ser motores 2D y tener problemas con el intérprete HTML. Además dependen de herramientas de terceros para poder crear la aplicación para dispositivos móviles, lo cual puede generar problemas futuros en la exportación. Otro punto en contra de *Construct* puede verse en el análisis hecho por (Nomdedeu et al., 2015), en el cual se afirma que es ideal para la creación de prototipos y juegos simples pero para la creación de juegos grandes presenta algunas dificultades.

El siguiente motor descartado fue *PlayCanvas*. Este motor no fue considerado como candidato principalmente por la gran cantidad de recursos que consume a la hora de *renderizar* los elementos. Esto provocaba un funcionamiento lento del equipo que interfería principalmente a la hora del desarrollo, haciendo que éste se atrase de forma innecesaria. A eso se le suma el problema que tienen los otros dos motores HTML5 vinculado con la exportación del juego a dispositivos móviles.

El motor eAdventure a simple vista parecía un motor ideal para este desarrollo. Las características de integración a EVEAs, la creación automática de informes, junto con su facilidad de uso lo hacían muy prometedor. Pero el hecho de no poseer una forma de exportar el juego a dispositivos móviles, junto con la limitación del género fueron motivos suficientes para descartar este motor.

La elección final estuvo entre *Unreal Engine* y *Unity*. El primero posee el problema de la de una curva de aprendizaje más lenta para desarrolladores independientes. El segundo cuenta con la imposibilidad de poder interpretar código HTML. Con este escenario, se resolvió que era más fácil e iba a consumir menos tiempo realizar un *parser* intermedio para eliminar todas las etiquetas HTML con las que cuentan las preguntas y respuestas de la autoevaluación, salteando así la necesidad de contar con el intérprete HTML.

3.7. Conclusiones del capítulo

En este capítulo se ha analizado la importancia que tienen los motores de juegos a la hora del desarrollo de videojuegos. A pesar de no haber un consenso en lo que es un motor, todos los autores coinciden en su importancia. Además se vio como los motores evolucionaron de ser sistemas creados por las empresas para uso propio, a ser sistemas cuyo desarrollo es pensado ya para el público general de desarrolladores.

Con este contexto no es raro que la oferta haya aumentado considerablemente a través

Cuadro 3.1: *Tabla comparativa de criterios en diferentes motores*

Criterios	Phaser	Construct	PC	eAdventure	UE	Unity
Facilidad de uso	5	5	4	4	5	5
Portabilidad	3	2	2	1	5	5
Curva de aprendizaje	4	5	3	5	2	5
HTML	1	2	5	5	5	2
Funcionamiento	5	5	2	5	5	5
Motor 3D	✗	✗	✓	✗	✓	✓
Licencia gratuita	✓	✓	✓	✓	✓	✓

de los años. Hoy en día existen una gran cantidad de motores, cada uno con sus ventajas y enfocados a construir diferentes estilos de juegos. Por eso ante esta gran oferta es necesario tener requisitos bien definidos a la hora de elegir el motor, ya que una mala elección puede perjudicar al desarrollo.

Es por eso que se han definido diferentes requisitos deseables para poder elegir el motor adecuado a partir de una selección de seis motores diferentes. Como ningún motor cumplía perfectamente con los requisitos planteados, se vio la necesidad de priorizar algunos de los criterios analizados sobre otros para llegar a la elección. Es así que se decidió no tener en cuenta el criterio de que el motor posea un intérprete HTML, y se decidió trabajar con Unity.

A modo de síntesis, se presenta en el cuadro 3.1 una tabla en la que se compara el grado de satisfacción de cada uno de los requisitos definidos a partir del análisis de los criterios tomados en cuenta para los seis motores analizados. Para los primeros cinco criterios se utilizó una escala numérica de 1 a 5 para describir en qué grado el motor responde a este criterio. En la escala 1 representa que no cumple el criterio, y 5 representa cumple con el criterio completamente. Para los otros dos criterios solo se tiene en cuenta si cumple o no con él, señalando con una tilde en el caso de tener esta funcionalidad y con una cruz en caso de no disponer de ella.

CAPÍTULO 4

DISEÑO DE LA PROPUESTA DEL JUEGO DESAFIATE

4.1. Introducción

En este capítulo se introducirá el juego que propone la presente tesina, que se ha denominado Desafiate, ya que incluye las preguntas de la cada autoevaluación como una serie de desafíos a resolver por parte del alumno. En la primera sección se hará un análisis general de los motivos por los cuales se realizó este juego en el contexto de los juegos serios móviles y se describirán las razones para su integración con el EVEA IDEAS.

Luego se introducirá el diseño propuesto para el juego y se mostrarán capturas de pantalla del resultado final. Esta sección se dividirá en diferentes subsecciones, las cuales corresponden a diferentes partes del juego.

4.2. Motivación

Como se ha analizado en el Capítulo 2 de este trabajo, existe una línea de investigación incipiente en relación a la utilización de juegos para los procesos de evaluación educativa. En el trabajo de Bezanilla et al. (2014) se observa que los juegos serios pueden ayudar a evitar el estrés propio de una evaluación, pero permiten dejar rastros del desempeño del alumno y la comprensión de los temas trabajados, y se constituyen en un instrumento más de evaluación, que permite al docente recuperar esta información. Por otra parte, existe una gran utilización de herramientas de autoevaluación que permiten la generación de pruebas cerradas con consignas del tipo verdadero o falso y *multiple choice*, en los entornos virtuales

de enseñanza y aprendizaje. Esto resulta atractivo para aprovechar la funcionalidad que ya poseen y generar desafíos con nuevas dinámicas para los alumnos, por ejemplo su inclusión en juegos. Otro aspecto a resaltar es la variedad de juegos disponibles para móviles con formato de tipo preguntas y respuestas que son consumidos por cantidad de jóvenes y niños, y despiertan su motivación. Para ejemplificar esto podemos tomar al juego argentino Preguntados, el cuál llegó a ser una de las aplicaciones más descargadas del mundo, y en el 28 de febrero del 2017 se encontraba en el sexto lugar de juegos mas descargados con más de 100 millones de descargas en la tienda móvil de *Google*, *PlayStore*. Geography es otro juego de preguntas muy popular. Tiene la particularidad de tener una visual 3D y de enfocarse en preguntas sobre Geografía. Cuenta actualmente con más de 500 mil descargas.

Todos estos aspectos han resultado de motivación para el diseño de Desafiate que tomará estas tres líneas mencionadas con anterioridad:

- Lograr una dinámica de autoevaluación para los alumnos basada en componentes propios de los juegos de manera tal de disminuir el estrés propio de las instancias de evaluación.
- Ofrecer alternativas para la autoevaluación de los alumnos con formatos más cercanos a los lenguajes y hábitos de las nuevas generaciones.
- Aprovechar la motivación que ya despiertan las dinámicas de los juegos tipo preguntas y respuestas para ser integradas a las funcionalidades de los EVEA, de forma tal de facilitar el seguimiento del alumno.

Es así que con estas metas en mente se inició el camino para el diseño de Desafiate.

4.3. Descripción general de Desafiate

Se propone la realización de un juego serio para dispositivos móviles orientado a la autoevaluación el cual ha sido llamado Desafiate. Este juego será de preguntas y respuestas el cual tendrá la particularidad de tomar dichas preguntas de las autoevaluaciones disponibles para el alumno en el EVEA IDEAS. Cada autoevaluación es una aventura nueva, en la cual el alumno personifica a un pirata en busca de tesoros. La historia cuenta que el pirata encontró unos mapas que le señalan diferentes tesoros encontrados en un archipiélago de islas. Cada isla tendrá la particularidad de que está habitada por diferentes tipo de personajes los cuales encontraron los tesoros deseados por el protagonista y sólo a cambio de información cederán el preciado bien. En el puerto de cada isla, un habitante le hará una pregunta a nuestro protagonista, y solo le dará el cofre si éste responde bien a la pregunta. El juego estará situado en el Caribe, lugar común donde estuvieron los piratas durante toda su historia.

4.3.1. Inicio

El inicio del juego muestra una imagen de una isla tropical desierta. Toda la vista está completamente llena de árboles para dar la sensación de encontrarse en una selva. Se agregarán detalles pertinentes al juego como algunos cocos, o un cofre del tesoro cerrado, para introducir al jugador en la temática propuesta. Lo principal de esta pantalla es el menú. Este muestra el nombre del juego y proporciona los *inputs* necesarios para poder iniciar sesión en el sistema. Por el momento el juego solo iniciará sesión con los datos de usuario del sistema IDEAS, pero el juego debe pensarse para poder iniciar sesión con otros EVEAs en el futuro. Un diseño preliminar de este menú se puede ver en la figura 4.1.

Figura 4.1: *Diseño preliminar del menú de inicio de sesión.*

Cuando el usuario ingresa los datos, en el tiempo que se realiza la comunicación y se recibe confirmación del servidor acerca de si los datos están bien, se visualiza una pantalla de carga para informar al usuario que debe esperar. Esta pantalla estará presente cada vez que haya que realizar una comunicación con el servidor. Esto puede verse en la figura 4.2. En caso de hubiese algún error en los datos o en la comunicación, se abrirá una ventana modal para comunicarle esta situación al usuario.

Figura 4.2: *Diseño final de la pantalla modal de carga.*

Figura 4.3: *Diseño final del menú de inicio de sesión.*

El diseño del menú ha sido creado para adaptarse a la temática del juego. Es por esto que los bordes han sido diseñados como cañas de azúcar atadas con hilos. Atrás de este menú se pueden visualizar unas hojas que actúan de forma meramente decorativas para hacer más contextualizada la vista. Esta visual se mantiene a lo largo de todas las pantallas de interfaz de usuario que se utilicen en el juego. El resultado final puede verse en la figura 4.3. Una vez que el usuario haya iniciado sesión correctamente, entonces el juego pasa al menú principal.

4.3.2. Menú principal

El menú principal está pensado para mostrar dos partes diferentes del juego: el perfil de usuario, mediante el cual se pueden ver los datos del usuario y su avatar, y un listado de las autoevaluaciones disponibles (aventuras del juego). El perfil del usuario se accede mediante un botón que propone un diseño icónico. De fondo se visualiza una playa con un muelle en el cual se encuentra el barco con el que se realizará la aventura. También se permite cerrar el juego mediante un botón iconográfico.

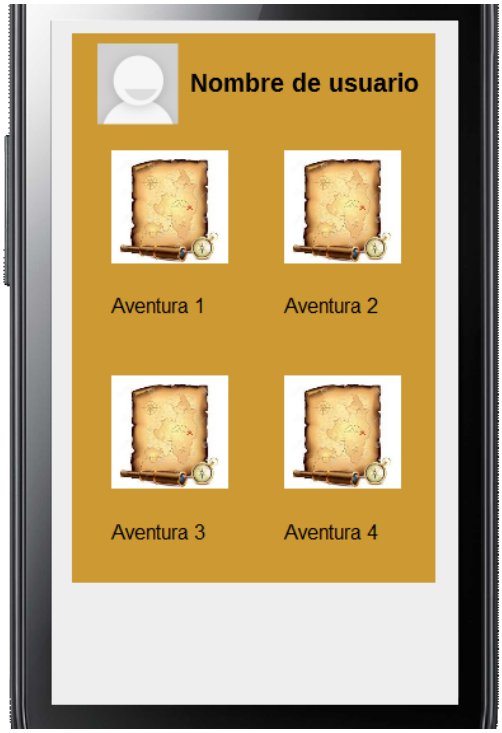

Figura 4.4: *Diseño preliminar del menú principal con el listado.*

El listado de autoevaluaciones será proporcionado por el servidor del EVEA. El orden de este listado será establecido por el mismo servidor, al igual que cuáles son las autoevaluaciones que podrán accederse. El juego solo mostrará lo proporcionado por el EVEA. Cada autoevaluación se representa con un mapa que indica que se trata de una aventura diferente. En cada mapa aparece la información pertinente a cada autoevaluación la cual es: nombre de la autoevaluación, la cantidad de preguntas que la componen

(desafíos), el estado de ésta (si ya fue realizada), y la nota si corresponde. En el marco del juego, cada mapa es una nueva aventura, que tiene un nombre, una cantidad de desafíos y se indica si la aventura fue completada o no y cuántas monedas fueron ganadas durante el recorrido. Este diseño preliminar se puede ver en la figura 4.4, y el diseño final se muestra en la figura 4.5. Es importante aclarar que en el diseño final del prototipo todavía no se cuenta con un acceso al perfil. Esta sección fue dejada como trabajo a futuro.

Figura 4.5: *Diseño final del menú principal con solo el listado.*

Figura 4.6: *Diseño final de los datos de la autoevaluación.*

Al hacer clic en cualquiera de las aventuras del listado, se abrirá una pantalla modal que extenderá los datos de la misma. Además de la información mostrada anteriormente en el listado, se agregan los datos de los objetivos y de los temas relacionados. A esta información se le suman dos botones. Un botón cumple la función de volver al listado cerrando el modal. El siguiente botón cierra todas las ventanas, y muestra el inicio de la aventura, la cual consiste en contar una breve historia y mostrar como el barco se empieza a mover

lentamente. Mientras sucede esto, el sistema internamente pide todas las preguntas de la autoevaluación con sus respectivas respuestas y guarda esta información para ser utilizada más tarde. La figura 4.6 muestra la ventana modal abierta con la información de la aventura (tomada de la autoevaluación en el EVEA), mientras que la figura 4.7 muestra el inicio de una de las aventuras.

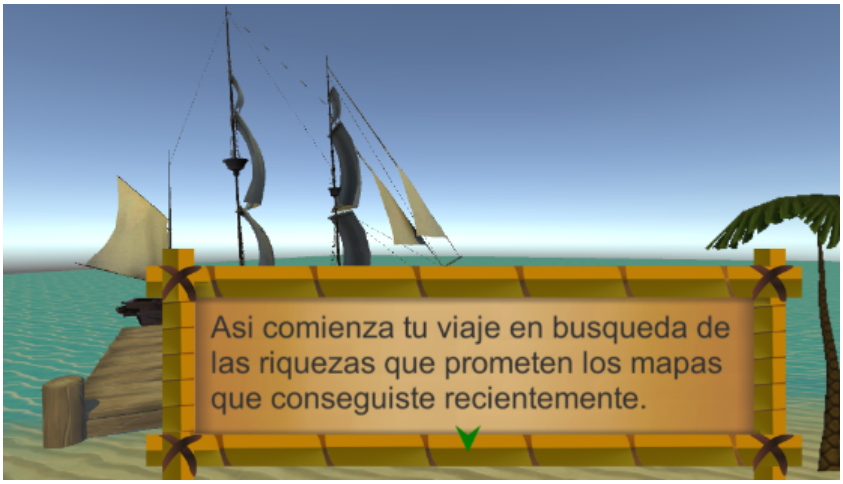

Figura 4.7: *Inicio de la aventura.*

4.3.3. Perfil

La modalidad del perfil es tal vez una de las más interesantes para el jugador, pero a su vez una de las más difíciles de diseñar y desarrollar. En esta ventana, que se puede ver en la figura 4.8, se muestran diferentes datos relevantes al jugador, como su nombre de usuario, la cantidad de aventuras realizadas mediante el juego, la cantidad de monedas que posee y la cantidad que lleva gastadas en el juego. Las monedas se irán obteniendo a medida que el jugador resuelva aventuras y se darán en base al resultado de ella.

En la parte inferior de la pantalla se puede ver el avatar que posee el jugador. Este empezará como un típico pirata con vestimenta de la época, barba y gorro con el símbolo pirata en él. Este avatar será personalizable para ajustarse a los gustos del jugador. Se tendrán algunas opciones base entre las cuales elegir, pero habrá más opciones las cuales podrán ser compradas gastando las monedas obtenidas anteriormente. Este avatar luego será utilizado para mostrarlo dirigiendo el barco del jugador, y también eventualmente en algún diálogo que pueda tener con personajes de las diferentes islas.

4.3.4. Responder preguntas

Una vez iniciada la aventura, nuestro personaje irá recorriendo en barco una serie de islas del archipiélago. Cada isla parte de un modelo creado anteriormente y que se selecciona en base al orden que le corresponde a la pregunta. En la versión actual del juego, estos modelos se van repitiendo en caso de que la cantidad de preguntas sea mayor a la cantidad

Figura 4.8: *Diseño propuesto para el perfil del usuario.*

de modelos definidos, sin embargo se ha diseñado que a futuro se utilice alguna fórmula que devuelva un número aleatorio para seleccionar de esa forma cuál es el modelo de isla que se verá. También es posible investigar sobre la generación procedural para crear cada uno de los escenarios. Esto traería ventajas como más aleatoriedad y una disminución significativa en el peso final del juego, pero aumentaría a su vez considerablemente la complejidad algorítmica. Estos aspectos se trabajarán en la segunda versión del juego.

Cada isla se corresponde con cada una de las preguntas pertenecientes a la autoevaluación, es decir en un desafío. La ambientación y estética de la isla varían entre cada uno de los escenarios. Cada isla puede tener diferentes tipos de árboles, y tiene una serie de elementos que la hacen única con respecto a las demás. Por ejemplo, una isla puede ser un interesante sitio arqueológico o ser hogar de un naufragio ocurrido hace poco tiempo. Las historias fueron diseñadas actualmente por el tesista, pero se ha estado trabajando en sesiones con alumnos que aportaron ideas para futuras historias en cada isla. Esto se detallará en el capítulo destinado a describir las sesiones de prueba.

El escenario muestra la llegada del personaje en su barco a la zona de la isla. El barco se dirige al puerto del lugar donde es recibido por alguien que actualmente está habitando la isla. Cuando el personaje avatar se encuentra con el habitante que lo recibe, éste le contará una historia que relatará como es que esa persona se encuentra en la isla en ese momento. Esta historia debe tener una relación importante con la ambientación ya generada.

Una vez contada la historia del habitante, éste le dará una explicación que justifique el

porqué se hace la pregunta. A cambio de la información dada por nuestro avatar, el habitante prometerá dar a cambio un cofre de tesoros encontrado en esa isla previamente, que suma monedas para el avatar. Un ejemplo del arribo del personaje a una isla ya desarrollada puede verse en la figura 4.9.

Figura 4.9: *El personaje llega a la isla.*

A continuación se le muestra al jugador una ventana con la pregunta y sus respuestas y cantidad de monedas que incluirá el cofre al responderla correctamente. En esta ventana el jugador debe seleccionar la o las respuestas que considere correctas y luego seleccionar el botón de responder. Una vez realizado este paso, se le mostrará un breve *feedback* al jugador donde las respuestas correctas seleccionadas se muestran en verde, y las respuestas incorrectas seleccionadas lo harán de rojo. Además se visualizará la cantidad de monedas obtenidas en la pregunta sobre el total posible. La visualización de la pregunta puede verse en la figura 4.10.

Es importante destacar en este punto que el sistema solo admite en un principio dos tipos de pregunta: *multiple choice* y verdadero o falso. La pregunta *multiple choice* permite seleccionar la cantidad de respuestas que el jugador considere, mientras que la pregunta de verdadero o falso solo permite seleccionar una opción a la vez. Sin embargo el juego debe prepararse para aceptar más tipos de consignas, o realizar cambios en los tipos ya existentes fácilmente. La confección de las preguntas y respuestas quedará a cargo total y exclusivamente del docente en el EVEA, el juego solo cambia la forma de visualizar las mismas.

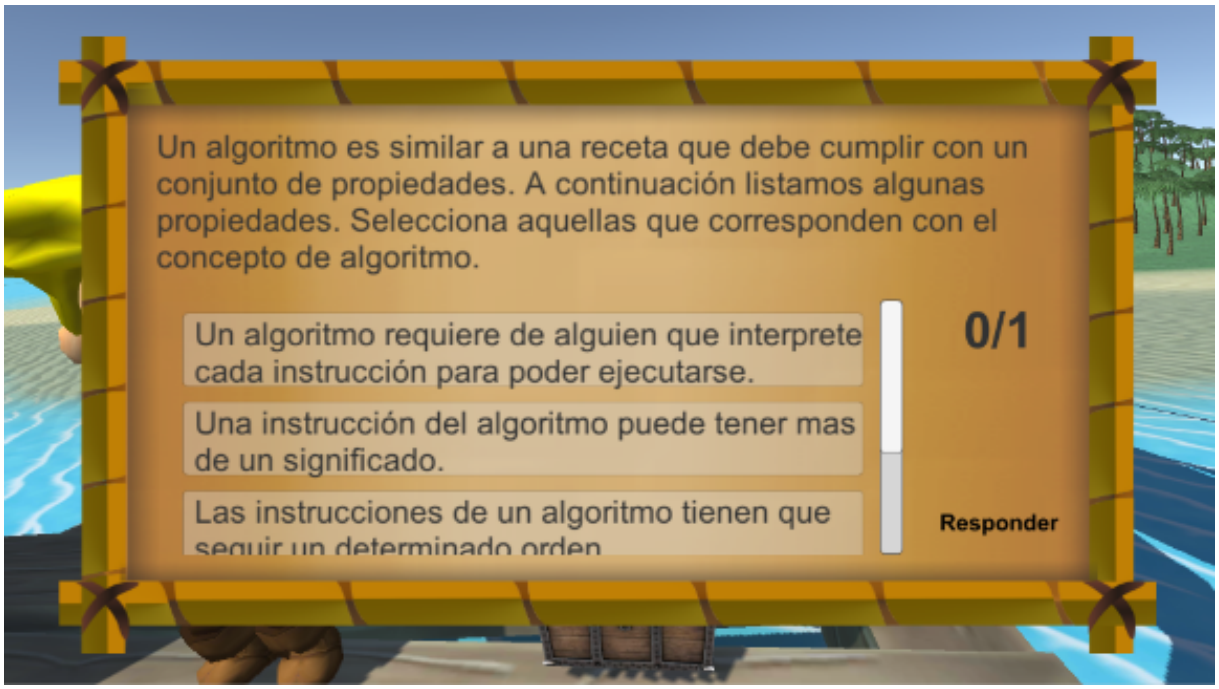

Figura 4.10: *El personaje llega a la isla.*

Para finalizar, una vez resuelta la pregunta y mostrado el *feedback*, el cofre se abre y el barco comienza a alejarse del puerto. En este punto se aprovecha para guardar los datos correspondientes a la pregunta y las respuestas seleccionadas. Una vez hecho esto y llegado a un punto específico del mapa, el juego avanza al siguiente desafío en una isla diferente. En caso de que se haya realizado la última pregunta el juego cambia a la escena de los resultados. Esto es explicado en la próxima sección.

4.3.5. Resultados

Cuando el jugador ya ha respondido todas las preguntas, más allá del resultado de éstas, se pasa a la escena de fin de la aventura donde se muestra a nuestro protagonista llegar al puerto donde empezó todo. Una vez llegado a destino, se puede ver un cofre del tesoro que simboliza las ganancias obtenidas a lo largo de la aventura. Por último se puede ver un listado de todas las preguntas incluidas como desafíos que se le fueron haciendo al protagonista, en el orden que las fue contestando. Para cada pregunta se lista el orden propio de ella, hasta 30 caracteres de la consigna, la cantidad de monedas que el jugador ha obtenido y el máximo que se podía obtener. Se decidió limitar lo mostrado de cada consigna a 30 caracteres ya que pueden llegar a ser muy extensas, o poseer una estructura que generen una visual muy alargada, lo que dificultaría poder realizar el objetivo de esta pantalla, que es el de mostrar un resumen de la aventura. Como única acción permitida hasta el momento, se visualiza un botón que permitirá regresar al menú principal. Un ejemplo de esta pantalla se puede ver en la figura 4.11

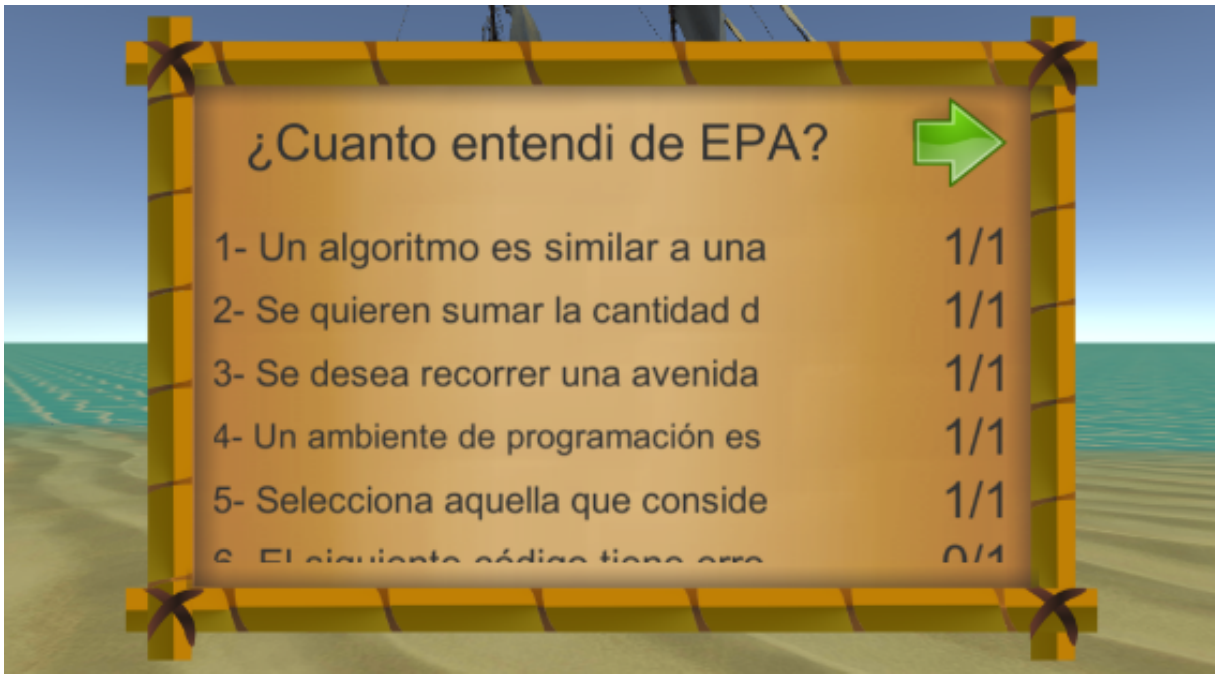

Figura 4.11: *Diseño final de la pantalla de visualización de resultados.*

4.4. Conclusiones del capítulo

En este capítulo se realizó una introducción de Desafiate junto con la mecánica de juego que posee. Se explicaron las diferentes secciones que componen el juego, desde el momento en que el jugador ingresa e inicia sesión, hasta el momento en que recibe los resultados de su aventura. Cada una de estas secciones fue acompañada con imágenes que muestran algunos diseños preliminares y el resultado final producto del desarrollo del juego.

En síntesis este capítulo resume la funcionalidad de Desafiate y los aspectos centrales de su diseño

CAPÍTULO 5

ARQUITECTURA E IMPLEMENTACIÓN DE LA PROPUESTA DEL JUEGO DESAFIATE

5.1. Introducción

En este capítulo se describe la arquitectura adoptada para el desarrollo de Desafiate y aspectos relacionados con su implementación.

El capítulo se divide en 4 partes. La primera parte presenta una descripción general de los aspectos claves de la arquitectura de Desafiate, se ilustra esto gráficamente con un esquema para hacer más representativa la descripción y se muestran allí las tres capas que componen la lógica de Desafiate. La segunda parte presenta un detalle sobre *Unity* que es el motor de juegos utilizado, rescatando los aspectos de este motor utilizados para el desarrollo de Desafiate, junto con los diferentes elementos que componen la capa del juego. La tercera sección presenta la capa de comunicación entre Desafiate e IDEAS. Por último, en la cuarta sección se describe la tecnología utilizada por IDEAS y cómo el docente crea una aventura (una autoevaluación) para ser incorporada en Desafiate. Este es un capítulo central del trabajo ya que presenta parte del trabajo.

5.2. Descripción general de la Arquitectura de Desafiate

Desafiate es un juego móvil desarrollado en el motor de juego *Unity 3D*. Debido a que el juego está pensado para usarse en conjunto con un EVEA que puede ser intercambiable según el requisito, se decidió dividir la arquitectura del juego en diferentes capas, para de esta forma, poder cambiar o reemplazar estas capas sin que las demás se vean afectadas.

Cada una de estas capas es explicada a continuación, y se acompañará esto con la figura 5.1 donde se podrá ver esquemáticamente las 3 capas. En esta figura, se toma la capa del EVEA con el ejemplo de IDEAS, ya que éste, es el EVEA utilizado para el prototipo desarrollado para esta tesina.

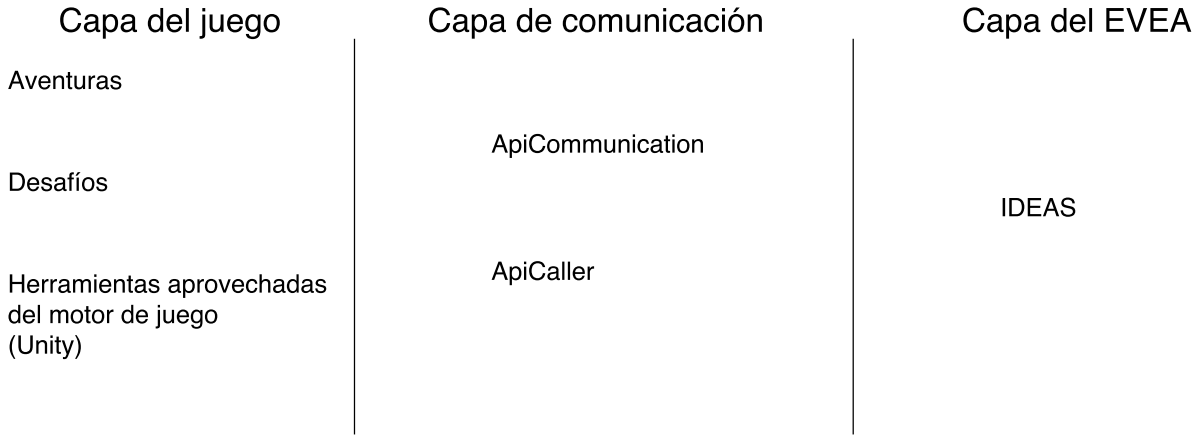

Figura 5.1: *Las diferentes capas de la arquitectura y sus componentes.*

- **Capa del juego:** esta capa está compuesta por todos los elementos que se utilizan para el funcionamiento interno del juego. Abarca el motor de juego elegido, los *assets* externos a utilizar, y los componentes desarrollados por el tesista como la jerarquía de ventanas modales, y los patrones aplicados a la resolución de los desafíos. Los componentes de esta capa serán explicados en las secciones 5.3, 5.4, 5.5, 5.6 y 5.7.
- **Capa del EVEA:** esta capa hace referencia al EVEA relacionado con el que se hará toda la comunicación y del cual se obtendrán los datos necesarios para la realización de las aventuras. Todo lo referente al funcionamiento de esta capa excede del control del desarrollo de Desafiate, pero interactuará con la capa de comunicación. Esta capa será analizada en la sección 5.9
- **Capa de comunicación:** esta capa es la que se encarga de poder establecer la comunicación entre el EVEA y Desafiate. Fue desarrollada especialmente a partir de las clases *ApiCommunication* y *ApiCaller*, las cuales han sido creadas con la idea de poder extenderse fácilmente en el futuro. Esta capa se explorará en la sección 5.8.

5.3. Unity, aspectos destacados y que ayudaron a la implementación del juego

Como se pudo ver en el capítulo 3, en el cual se analizaron varias alternativas de motores de juegos para realizar el desarrollo propuesto en esta tesina, el motor que resultó elegido fue *Unity 3D*. El uso de este motor resulto sencillo y adecuado para el desarrollo realizado.

Apenas inicia el motor se muestra una interfaz dividida en partes comúnmente usadas por los desarrolladores. Cabe destacar que esta interfaz es totalmente modificable y adaptable a los requerimientos que el desarrollador tenga. Una muestra de la interfaz de *Unity* puede verse en la figura 5.2

En la parte central de la pantalla se visualiza la zona espacial de una escena en donde se pueden ubicar los diferentes objetos pertenecientes a dicha escena mientras que en la parte izquierda se pueden visualizar las escenas abiertas, y la jerarquía de objetos pertenecientes a la misma. La pantalla de la zona espacial es probablemente la más importante para el desarrollo. En ella se puede explorar libremente toda la escena, e ir seleccionando los diferentes objetos. Se proveen herramientas para manipular el objeto en el espacio zonal para de esta forma obtener la posición, rotación y tamaño deseados. La parte central cuenta con otras 2 pestañas para cambiar la visualización según se requiera. Una de estas pestañas muestra la vista que se tendrá al correr el juego, mientras que la otra pestaña es un navegador que permite acceder a la tienda de *assets* del motor.

En la parte inferior de la pantalla se puede ver la jerarquía de carpetas del proyecto que se encuentra abierto. Esta visual es muy parecida a la mayoría de los exploradores de archivos donde se puede ver a la izquierda la jerarquía de carpetas y en la parte derecha un listado de los elementos que componen la carpeta seleccionada. Esta sección cuenta a su vez con una pestaña para poder visualizar la consola del sistema.

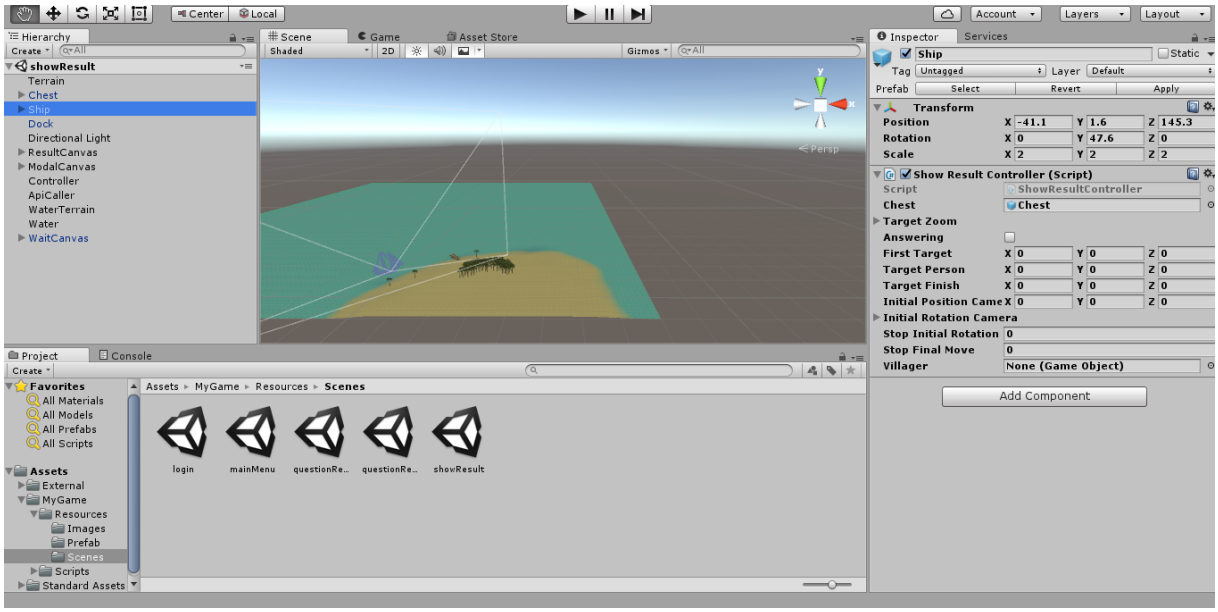

Figura 5.2: *Pantalla del entorno de desarrollo de Unity 3D.*

Por último, en la parte derecha de la pantalla se encuentra el inspector de elementos. En este se muestran las diferentes características y elementos que componen el objeto seleccionado en ese momento. Estas características pueden ser modificadas para poder

lograr que el objeto tenga el comportamiento deseado, y tener una mayor precisión a la hora de lograr la ubicación, rotación y tamaño del objeto con respecto a la vista de la escena.

A continuación se listarán características propias de la arquitectura y uso de *Unity* que se consideraron interesantes para comprender mejor este motor y que influyeron en el desarrollo del juego serio creado.

5.3.1. Zona espacial

Al comienzo de esta sección se ha dado un breve análisis de cómo se compone la zona espacial de una escena. Los objetos que se ubican aquí suelen ser muy diversos. Toda zona espacial cuenta con un objeto base especial: el terreno. Este objeto suele ser el primero en crearse, y es el objeto que mantendrá a los demás que se posicionen sobre él. En la subsección 5.3.2 se analizará en más profundidad este objeto.

Todos los objetos que forman parte de una escena pueden ser a su vez contenedores de varios objetos hijos formando así una jerarquía, es decir, se pueden combinar objetos más chicos y menos complejos para lograr un objeto de una complejidad mayor y agruparlos en éste. Una de las ventajas de esto es que se pueden habilitar y deshabilitar los objetos agrupados muy fácilmente ya que solo hace falta cambiar el estado al objeto principal. Otra ventaja es que, al editar la posición de un objeto padre, todos los objetos pertenecientes a él se moverán en conjunto y de esta forma mantendrán la relación y la forma del objeto principal.

Si bien anteriormente se dijo que se pueden manipular la ubicación, rotación y tamaños de los objetos con la vista de la zona espacial de la escena, o con el inspector del objeto, hay que tener en cuenta algo muy importante: los objetos tienen una ubicación, rotación y tamaños absolutos y relativos. Las características absolutas de los objetos con respecto a la zona espacial, se refieren justamente a su relación en el espacio olvidándose de la jerarquía a la que pertenecen. Las características relativas se refieren a su relación con el objeto padre que los contiene. Es decir, si un objeto contenedor es rotado, sus hijos mantendrán la misma rotación relativa, pero su rotación absoluta cambiará.

Una particularidad de la zona espacial es que no solo acepta objetos 3D, sino que también permite utilizar objetos 2D. Estos objetos son normalmente visualizados en un área que se extiende por el eje X y el eje Y. Esta área representa la pantalla donde se visualizará el juego cuando éste se ejecute y se compone por elementos de interfaz gráfica típicos de cualquier sistema. En la subsección 5.3.3 se analizará más a fondo esta funcionalidad.

Por último es importante destacar que en el momento de edición de la zona espacial, los objetos se encuentran fijos y pueden superponerse con otros. Solamente en el momento de la ejecución es que se utiliza la física y el sistema aplicado de colisiones que materializa los objetos para que estos no se superpongan.

5.3.2. Creación de terrenos

Unity provee una herramienta para la generación de terrenos muy poderosa y fácil de usar. Cada terreno empieza siendo un área ubicada en la posición 0 del eje Y, que se extiende por el eje X y el eje Z. Mediante el inspector del terreno, se nos brindan las herramientas necesarias para poder editar el terreno y ajustarlo a nuestro gusto de manera sencilla. Las diferentes acciones del editor se pueden ver en la figura 5.3.

Las primeras 3 acciones disponibles son las que permiten realizar la modificación de la altura del terreno, es decir, modificarlo a lo largo del eje Y. Si bien pueden parecer que 3 acciones para modificar el terreno son excesivas, cada una brinda una funcionalidad diferente y muy útil que complementa a las demás. La primera acción es la que permite cambiar la altura de una sección del terreno, ya sea aumentando o disminuyendo la altura. Hay que ser muy cuidadoso con esta herramienta ya que es muy sensible y suele tener cambios drásticos si no se la usa con cuidado y paciencia. La segunda acción permite ajustar una sección del terreno a una altura determinada. Esta altura puede ajustarse manualmente mediante el inspector, o puede seleccionarse copiando la altura que posea una sección determinada. Esta acción también provee una opción llamada *flatten*, mediante la cual se puede acomodar todo el terreno a la misma altura. La tercera y última acción es la que permite acomodar el terreno para que los diferentes cambios de altura entre una sección y otra no sean tan bruscos y se vean de una forma más suave y natural.

Cada terreno empieza sin textura, como un lienzo en blanco. Es la cuarta acción la que permite modificar la textura del terreno. Las texturas son imágenes que se utilizan sobre la superficie de un terreno, o sobre un objeto para darle la apariencia y el diseño que se verá luego en el juego. Es normal que en el terreno se utilice una textura de tamaño chico y que esta se repita muchas veces para cubrir una gran parte del terreno, el cual, no tiene límites en la cantidad de texturas que puede poseer. De esta forma, con dos simples imágenes se puede crear un terreno que contenga paso y arena al mismo tiempo. Para llenar el terreno con las diferentes texturas, *Unity* provee una serie de pinceles de diferente tipo, a los cuales se les puede ajustar el tamaño para, de esta forma, poder realizar un terreno de la manera deseada. Estos pinceles también son utilizados en las acciones de ajuste de la altura del terreno y pueden verse en la figura 5.4.

Figura 5.3: *Barra de acciones para la modificación del terreno.*

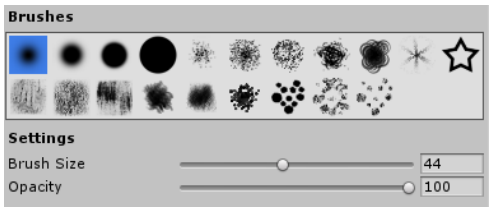

Figura 5.4: *Los diferentes pinceles disponibles para modificar el terreno.*

La quinta acción de la herramienta de generación de terreno es la de edición de árboles. En esta herramienta se pueden agregar diferentes modelos que posea el proyecto y una vez seleccionado uno de estos modelos, se puede utilizar un único pincel de forma redonda para ubicar varios árboles del modelo seleccionados al mismo tiempo. Se pueden configurar diferentes opciones para realizar esta acción. Se puede editar el tamaño del pincel, la densidad de los árboles que habrá al crearlos, la altura y ancho entre los que pueden variar los árboles y una variación en la sombra para cambiar el color de cada árbol. De esta forma se puede crear fácilmente un bosque con diferentes tipos de árboles, densidades, colores y tamaños. Por último, esta acción tiene la posibilidad de plantar de manera totalmente aleatoria una cantidad deseada de árboles a lo largo de todo el terreno.

Así como la acción recientemente descrita permite agregar árboles de forma fácil y variada, la sexta acción permite agregar el pasto al terreno de una forma similar. Esta acción no ha sido requerida para el proyecto con lo cual solo se la nombra al ser parte de la edición de terreno y no se extenderá en su descripción en esta tesina.

La última y séptima acción es la que permite modificar la configuración del terreno. Con ella se pueden editar aspectos como el tamaño del plano del terreno, la ubicación del punto de partida de éste, el viento que poseerá el terreno y otros aspectos no usados para el desarrollo de esta tesina.

Para finalizar con la descripción de la herramienta de generación de terreno hay que notar algunas cuestiones útiles pero que escapan a las acciones anteriormente descritas. Una vez creado un terreno, éste se asociara al proyecto y podrá ser usado en múltiples escenas. Este terreno permanecerá único a pesar de esta múltiple asociación, con lo cual, una modificación del terreno en una escena, lo modificará también en las demás escenas asociadas. Las escenas no están limitadas a un único terreno, sino que pueden hacer uso de múltiples terrenos ya creados para formar la visual necesaria.

5.3.3. Interfaz gráfica

La creación de interfaces gráficas es otro de los puntos fuertes con los que cuenta *Unity*. Como se dijo anteriormente en la subsección 5.3.1, la interfaz gráfica 2D puede verse en la zona espacial donde se modifican los objetos de la escena. Ésta interfaz aparece como un plano que representa a la pantalla del jugador y que se extiende a lo largo del eje X e Y. En el podrán utilizarse diferentes objetos básicos de interfaz gráfica, cada uno de los cuales poseen características diferentes.

El elemento más importante de la interfaz es el *canvas*. En él se agruparán todos los elementos que aparecerán como parte de la interfaz y limitará el área total accesible. El *canvas* es el área de la interfaz gráfica. Sin él, no existiría la interfaz.

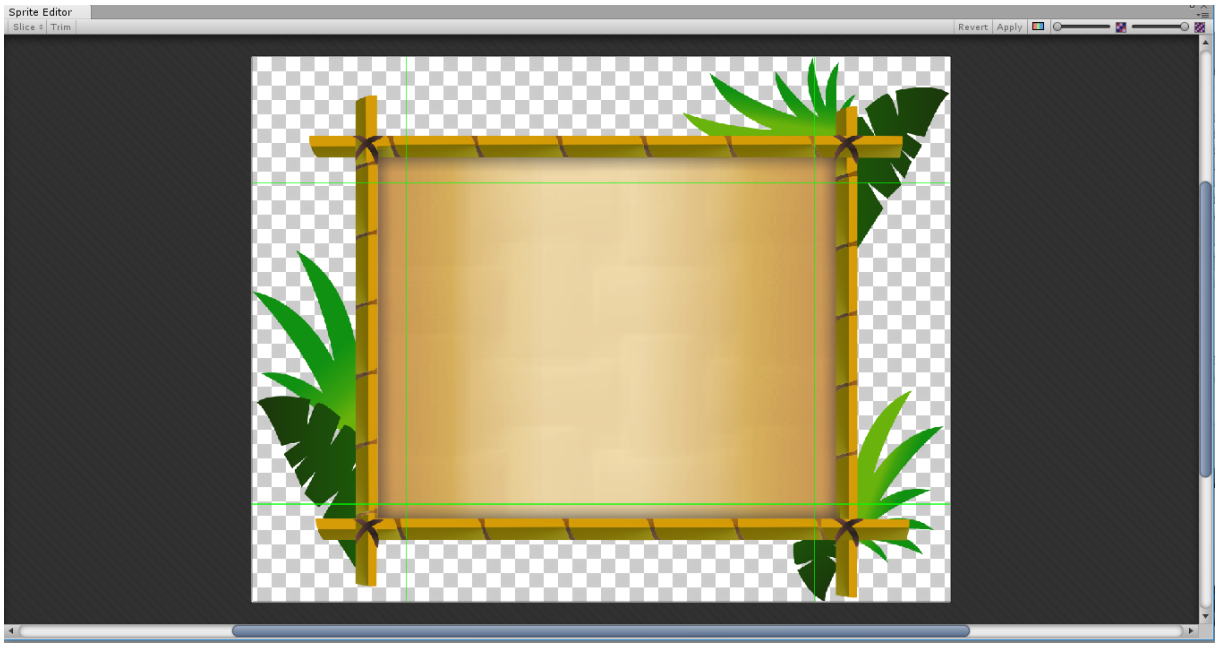

Figura 5.5: *Editor de sprite provisto por Unity.*

Otro elemento muy usado es el panel. Este elemento también suele cumplir la función de agrupar otros elementos, además de brindar la posibilidad de agregarle una imagen de fondo para el diseño. Si bien su función puede parecer similar a la del *canvas*, el panel solo cumple la función de agrupar elementos en una forma lógica mientras que el *canvas* es la interfaz. La opción de poder tener una imagen de fondo es uno de los mayores atractivos de *Unity* a la hora de crear interfaces, ya que la herramienta que provee para esto es increíblemente poderosa. Las imágenes asociadas al proyecto pueden ser de diferentes tipos, entre los cuales, se encuentra el tipo *sprite*. Los *sprites* pueden editarse de una forma muy sencilla mediante el editor que provee *Unity*. Entre otras opciones que tiene este editor, tiene la opción de preparar cualquier *sprite* para ser una imagen de fondo. Lo único que hay que hacer es seleccionar mediante unas guías, las cuales serán los bordes, las esquinas y la parte central del panel. De esta forma, al asociar el *sprite* al panel, *Unity* se encarga automáticamente de ajustar la imagen al tamaño correspondiente, y asociando los bordes seleccionados de la imagen con los bordes del panel. El editor de *sprite* puede verse en la figura 5.5.

A los elementos básicos de cualquier interfaz gráfica, como los textos, los botones, las imágenes, los *dropdowns* y los campos de entrada de datos, también llamados *inputs*, se le suman algunos muy útiles para crear interfaces de juegos como el elemento *slider*, que permite seleccionar diferentes valores dentro de un rango específico, o el elemento *scrollbar*, el cual permite manipular el desplazamiento del contenido de un elemento en caso de que este exceda el tamaño de su contenedor.

Algunos de estos elementos pueden tener asociados diferentes eventos con los que se

asocian funciones pertenecientes a algún *script*. Existe un elemento encargado de hacer dicha asociación, y es el *event system*. Este elemento se ocupa de recoger los diferentes eventos que ocurren con los medios de interacción que posee el usuario como el *mouse*, el teclado y demás dispositivos. Cuando ocurre alguno de estos eventos, el *event system* se encarga de ver sobre qué elementos ocurren estos eventos y ver si corresponde activar la invocación a una función de algún *script*.

5.3.4. *Scripting*

Hasta ahora todo lo analizado con el motor han sido elementos que tienen que ver con la parte visual y la creación del mundo con la que interactuará el jugador. En la subsección 5.3.3 se comentó que algunos elementos pueden tener asociados eventos que activan la llamada a una función de un *script*. En la última versión de Unity los *scripts* pueden escribirse en 2 lenguajes de programación: *C#* y *JavaScript*. El desarrollador puede elegir el lenguaje que prefiera y en el mismo proyector pueden convivir *scripts* de ambos lenguajes. Todos los *scripts* deben estar asociados a un objeto y se encargarán de darle funcionalidad a dicho objeto. Sin embargo, mediante diferentes funciones se podrán ubicar otros objetos asociados a la jerarquía a la que pertenece o incluso los que no se encuentran dentro de ésta, aunque esto último no es recomendable por lo lento que resulta en ejecución.

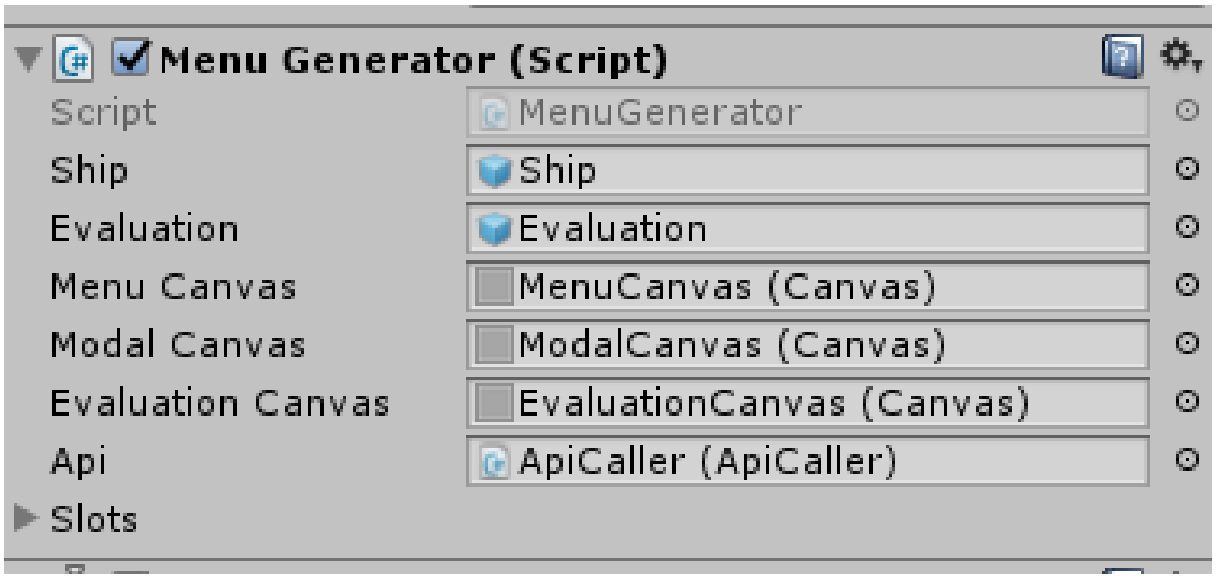

Figura 5.6: *Vista del inspector de un objeto donde se puede asignar los valores a las variables publicas del script asociado.*

A lo largo del desarrollo propuesto para esta tesina, se utilizó *C#* y es por esto que el análisis se centrará en este lenguaje. Cuando se crea un *script* nuevo, el sistema automáticamente le asocia una clase nueva y que hereda de la clase *MonoBehaviour*, creando también los métodos *start* y *update*. Estos dos métodos son la base de las

modificaciones que se desean hacer durante la ejecución del juego. *Start* es un método que es llamado para todos los objetos apenas inicia la escena. Su función es la de preparar todas las variables y los atributos que necesita el *script* y el objeto para poder funcionar a lo largo de la escena. *Update* es un método que es llamado en cada uno de los *frames* de la ejecución y su función es, básicamente, la de realizar una o varias acciones, como mover un objeto, rotarlo, armar un manejador de eventos propio y más complejo que el aportado por *Unity* o cualquier acción que necesite hacerse periódicamente. *MonoBehaviour* posee otros métodos muy útiles, especialmente cuando se trata del tratamiento de ciertos eventos de la física, pero no serán analizados por no ser utilizados en el desarrollo actual.

Si bien el sistema hace heredar a cualquier clase nueva de *MonoBehaviour*, el desarrollador puede cambiar esto en cualquier momento y de la forma que desee. El sistema no limita al lenguaje de ninguna forma, y la elección de heredar automáticamente de la clase *MonoBehaviour* se debe simplemente a una elección de facilitar la tarea del desarrollador ya que se estima que esa será la clase que más se heredará. A su vez, es posible heredar de esta clase sin la necesidad de sobrescribir los métodos *start* y *update*. Otro aspecto importante es que un *script* puede estar asociado a varios objetos en una escena o durante el juego y también puede no estar asociado a ningún objeto en particular, pero de esta forma el *script* no tendría sentido al no poder proveer funcionalidad a nada en particular.

Como cualquier clase en *C#*, las creadas en *Unity* pueden tener variables de diferentes tipos y alcance. Si bien, en este trabajo no se explorarán estos conceptos por escapar a la temática, es importante nombrarlos por una facilidad que brinda *Unity* con respecto a las variables públicas. Cuando el *script* se asocia a un objeto de juego, a todas las variables públicas se les puede asignar un valor inicial directamente desde el inspector del objeto como puede verse en la figura 5.6. Para asignar un valor a una variable de tipo *GameObject* o de alguna de los elementos de interfaz gráfica, solo hace falta arrastrar el objeto desde la jerarquía de objetos de la parte izquierda de la pantalla, hasta el inspector en el campo correspondiente a la variable a la que se desea asignar el valor. Esta funcionalidad resulta útil, especialmente cuando las variables que posee la clase son otros objetos del juego que se necesitan para realizar diferentes acciones.

La clase *MonoBehaviour* no es la única provista por *Unity*. El sistema provee muchas clases que cumplen diferentes funciones y facilitan la tarea del desarrollador. Algunas de estas clases serán analizadas en las siguientes secciones donde se las necesitará para explicar de una mejor forma diferentes cuestiones del desarrollo propuesto para esta tesina. Para finalizar con el análisis de esta funcionalidad, hay que destacar que en caso de que lo provisto por el motor no sea suficiente, se pueden importar librerías externas de una manera muy fácil. Solo hace falta agregar la librería a la carpeta del proyecto y luego solo hace falta referenciarla en el *script* deseado.

5.3.5. Cámara

La cámara es uno de los elementos más importantes y es, tal vez, dejado de lado por muchos desarrolladores. La cámara es quien muestra y define qué es lo que el jugador verá a la hora de utilizar un juego. Como cualquier objeto, ésta se encuentra en la zona espacial de una escena, y dependiendo de su rotación y ubicación es lo que enfocará y mostrará al jugador. En la figura 5.7 se puede ver un ejemplo de las herramientas que brinda *Unity* para facilitar la utilización de la cámara. Para esto, el motor administra una guía que señala hacia donde apunta y que es lo que abarcaría, además de mostrar una imagen pequeña ubicada en la esquina inferior derecha, y que muestra que es lo que se vería en tiempo de ejecución.

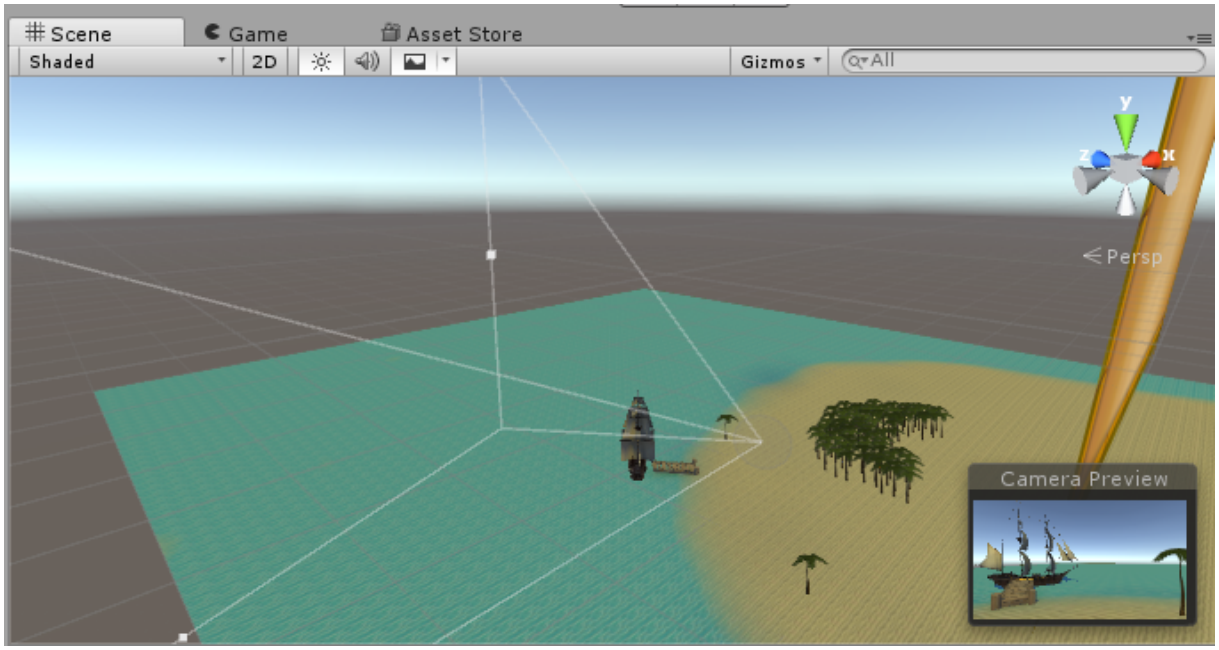

Figura 5.7: *Ejemplo de las facilidades que brinda Unity a la hora de seleccionar y manipular la cámara en la zona espacial.*

En la subsección 5.3.1 se dijo que los objetos pertenecientes a una escena pueden encontrarse en la jerarquía de padres e hijos de otros objetos. La cámara no es una excepción a esto, lo cual brinda una gran ventaja. Como todos los objetos relacionados entre sí a través de un único objeto se moverán en conjunto al ocurrir un desplazamiento del objeto padre, con esto se puede lograr que al agregar a la cámara a la jerarquía de un objeto, ésta se mueva con su objeto padre. De esta forma, se simula que la cámara siga y enfoque siempre a un objeto en particular, algo especialmente útil cuando se trata de seguir a un personaje o vehículo que representa al jugador.

5.4. *Assets* externos

Entre las diferentes funcionalidades que brinda *Unity*, existe la tienda de *Assets*. En este lugar *Unity* le da la posibilidad a los desarrolladores y diseñadores de subir sus creaciones las cuales pueden ser de tipos diferentes como por ejemplo animaciones, modelos 3D, sonidos, *scripts*, texturas y más. El autor de cada uno de estos elementos será el que defina el precio a pagar por ellos, el cual puede ser gratis. La tienda posee un sistema de clasificación de los elementos así como una visualización previa para de esta forma poder evaluar bien el producto que se está adquiriendo.

Para facilitar y agilizar el desarrollo propuesto en esta tesina se decidió hacer uso de la tienda de *Assets* para poder adquirir diferentes modelos 3D y de esta forma evitar tener que realizar los modelos 3D necesarios. El modelado de elementos 3D es una de las partes más importantes en el desarrollo de cualquier juego pero, a su vez, también lo es en términos de dificultad y consumo de tiempo. La creación de un modelo, con un nivel de detalle específico y adecuado para el juego en el que se va a utilizar, la creación de la textura y las animaciones que puede realizar el modelo creado, son trabajos importantes y que requieren una atención a los detalles, a tal punto, que normalmente este trabajo es realizado por una persona o un grupo de personas dentro del desarrollo.

A continuación se listarán los elementos utilizados en el desarrollo propuesto de esta tesina y se justificará por qué han sido elegidos. Cada uno de estos elementos será presentado en una subsección propia para facilitar el análisis y la compresión de los mismos. Además se presentan aquí algunos ejes de análisis para cada *asset* a revisar, de manera tal que todos sean considerados bajo la misma mirada. Todos los elementos analizados son gratuitos pero es posible que en el futuro sean reemplazados por modelos realizados por el autor de esta tesina o con ayuda de terceros.

- **Nombre del *asset*:** se informará en el título de la subsección el nombre del *asset* seleccionado.
- **Tipo del *asset*:** es importante nombrar de qué tipo y categoría dentro de la tienda es cada uno de los *assets* utilizados para comenzar a entender su propósito.
- **Autor:** se nombrará al autor del *asset* utilizado y de ser posible se dará una pequeña descripción de él mismo.
- **URL:** se pondrá un enlace al lugar de la tienda correspondiente.
- **Descripción:** se describirá brevemente el *asset* seleccionado junto con todos sus componentes.
- **Razón de la inclusión:** el punto más importante de este análisis son las razones por las cuales el *asset* ha sido seleccionado sobre los demás.

- **Imagen:** cada uno de las subsecciones serán acompañadas, de ser posible, por una imagen donde se podrá ver el elemento escogido y su integración en Desafiate.

5.4.1. *Island assets*

Figura 5.8: *Visual de la escena de ejemplo provista por Island assets.*

- **Tipo del *asset*:** Modelo 3D/Ambientación.
- **Autor:** *Lylek Games* es la empresa creadora de este *asset*. Se trata de una pequeña empresa que desarrolla juegos para PC, dispositivos *Android* y *assets* para la tienda de *Unity*.
- **URL:** `https://www.assetstore.unity3d.com/en//\#!/content/56989`.
- **Descripción:** se trata de un conjunto de *assets* que brinda los elementos básicos que se necesitan para dotar a una escena con la temática de una isla del caribe. Entre los modelos 3D que provee este conjunto los utilizados fueron los modelos de un cofre del tesoro lleno de monedas, un muelle de madera sencillo, un coco y dos tipos diferentes de árboles. El muelle se ha utilizado en casi todas las escenas en las que aparece el barco del *asset* de la subsección 5.4.3. El cofre y el coco han sido utilizados para decorar la pantalla de inicio de sesión. El cofre ha sido utilizado además en la pantalla de vistas de resultados simbolizando las ganancias del personaje del juego.

  En el aspecto de las texturas, el conjunto provee todas las necesarias para los modelos anteriormente descriptos junto con las texturas para el terreno. Entre estas últimas se encuentran las texturas del agua, la arena y el pasto. Todas estas texturas fueron utilizadas en Desafiate.

- **Razón de la inclusión:** este *asset* ha sido elegido sobre lo demás por lo completo que es. Brinda todos los elementos necesarios para crear una escena con una vista caribeña, mientras que las otras opciones solo proveían una parte y cada una tenía un estilo propio, lo que hacía que al combinar estas opciones, el diseño final quedará muy dispar.

5.4.2. *Medieval Toon Character*

Figura 5.9: *Visual de diferentes versiones del personaje que se pueden logra gracias a Medieval Toon Character.*

- **Tipo del *asset*:** Modelo 3D/Personajes/Caricatura.
- **Autor:** *Night Forest* es la empresa que creó un conjunto de *assets* para poder crear una escena con la temática de una aldea medieval.
- **URL:** `https://www.assetstore.unity3d.com/en/\#!/content/19641`.
- **Descripción:** este *asset* brinda un único modelo 3D que representa a una versión caricaturizada de un aldeano de una aldea medieval. Si bien un modelo solo puede parecer que es insuficiente, este modelo viene con tres texturas diferentes para poder cambiar el estilo del aldeano. Estas texturas varían en el color del pelo y de la ropa del personaje. Además se pueden variar también el estilo de los colores para representar diferentes expresiones y emociones.
- **Razón de la inclusión:** si bien puede parecer que un aldeano medieval no es una buena opción para una temática de piratas, el estilo del diseño de las ropas del personaje coinciden con el entorno creado en Desafiate. Este estilo puede utilizarse sin problemas con el de un isleño o un náufrago que habita la isla. Entre todas las

opciones encontradas, ésta era la única que se podía combinar con el entorno y la temática.

5.4.3. *Brig Sloop Sailing Ship*

- **Tipo del *asset*:** Modelo 3D/Vehículo/Mar.
- **Autor:** *Joe Censored Games* es una empresa que se dedica al desarrollo de videojuegos para PC. Ha aportado a la tienda de *assets* con diferentes modelos 3D de barcos antiguos a vela.
- **URL:** `https://www.assetstore.unity3d.com/en/\#!/content/77862`.
- **Descripción:** este *asset* se compone de un solo barco de vela el cual es parte de una gran colección de 40 barcos. Viene con el modelo del barco y la textura de éste. No provee ningún tipo de animación.
- **Razón de la inclusión:** se incluyó este modelo ya que era el único de los modelos disponibles gratuitamente que cumplía con la estética del juego. Era la única opción de un barco a vela antiguo.

Figura 5.10: *Visual de ejemplo del barco provisto por Brig Sloop Sailing Ship.*

5.4.4. *Inside the Island*

- **Tipo del *asset*:** Modelo 3D/Ambientación.
- **Autor:** Matheus Oliveira es un diseñador 3D que realizó diferentes tipos de modelos para crear ambientaciones en *Unity*.
- **URL:** `https://www.assetstore.unity3d.com/en/\#!/content/62362`.

- **Descripción:** al igual que el *asset* analizado en la subsección 5.4.1, éste es un conjunto de modelos diseñados para crear la ambientación de una isla caribeña. Entre los modelos que provee se encuentran un árbol, una planta, una cabaña y una jarra.

Figura 5.11: *Visual de ejemplo de los diferentes elementos de Inside the Island.*

- **Razón de la inclusión:** al tener ya prácticamente todo lo que aporta este conjunto con lo aportado por el analizado en la subseccion 5.4.1, no fue necesario utilizar la mayoría de los recursos provistos por éste. Sin embargo se utilizó el modelo de la cabaña para ambientar una de las islas como hogar de los habitantes de ésta.

5.4.5. *Bronze figures*

- **Tipo del *asset*:** Modelo 3D/Personajes/Humanoides/Fantasía.

- **Autor:** Endre Baráth es un diseñador que ha creado modelos y texturas para crear ambientaciones medievales y fantásticas.

- **URL:** `https://www.assetstore.unity3d.com/en/\#!/content/8751`.

- **Descripción:** este *asset* se compone de un conjunto de 3 cabezas de bronce gigantes con el objetivo de crear ambientaciones fantásticas. Las cabezas simulan ser figuras de enanos, reyes y reinas.

- **Razón de la inclusión:** estos modelos fueron incluidos para generar la ambientación de una de las islas y crear una historia en torno a ellas donde el personaje se encuentra ahí, ya que es un sitio arqueológico muy interesante por la naturaleza extraña de las cabezas.

Figura 5.12: *Visual de ejemplo de las 3 figuras provistas por Bronze figures.*

5.4.6. *Low Poly Dancing Rabbit*

- **Tipo del *asset*:** Modelo 3D/Personajes/Caricatura.

- **Autor:** *wHiteRabbiT sTudio* es la empresa creadora de este modelo y se trata de una empresa ubicada en Francia, cuyo objetivo es recuperar un cierto nivel de desafío en los videojuegos manteniendo la mayor originalidad posible y maximizar la diversión aportada por sus videojuegos.

- **URL:** `https://www.assetstore.unity3d.com/en/\#!/content/22788`.

- **Descripción:** este *asset* se compone de un solo modelo 3D y las texturas y animaciones correspondientes. Se trata de un conejo del tamaño de una persona cuyas animaciones le permiten bailar.

- **Razón de la inclusión:** este modelo se incluyó simplemente con la idea de ser una gracia y algo interesante para el jugador. Algo como para aumentar el disfrute solo por lo curioso que resulta.

Figura 5.13: *Visual de ejemplo del conejo bailarín provisto por Low Poly Dancing Rabbit.*

5.5. Modales

Al ser Desafiate un juego que se basa fuertemente en la interfaz de usuario, éste posee una gran cantidad de ventanas modales, de diferentes tipos, aunque con algunas características comunes. Si bien *Unity* provee la posibilidad de cambiar el estado de visualización de un *Canvas* o panel de forma sencilla, es necesario contar con una forma fácil para poder manejar los eventos de las pantallas modales, así como los datos que éstas poseen. Para esto, se creó la jerarquía de clases que se puede ver en la figura 5.14.

Los objetos creados a partir de las clases pertenecientes a la jerarquía, serán los encargados de la funcionalidad de manejar los modales, y de completar sus contenidos de acuerdo a las exigencias de la situación. La clase *Modal* no es una clase abstracta, ya que posee un funcionamiento, el cual es utilizado en varias partes en las clases herederas.

A continuación se explicarán las diferentes clases de esta jerarquía, en una subsección para cada una.

5.5.1. *Modal*

La clase *Modal* es la base sobre la que se sostiene toda la funcionalidad de ventanas modales. Es quien define cómo se realizan diferentes acciones, las cuales serán heredadas por las demás clases. Si bien, es quien tal vez tenga la mayor cantidad de funcionalidad brindada, es una clase que nunca se usa directamente en ningún *script*.

La clase posee una única variable pública que es el *Canvas* a manipular. Esta variable es inicializada en el constructor de la clase, al cual debe pasarse el *Canvas* que tendrá asociado.

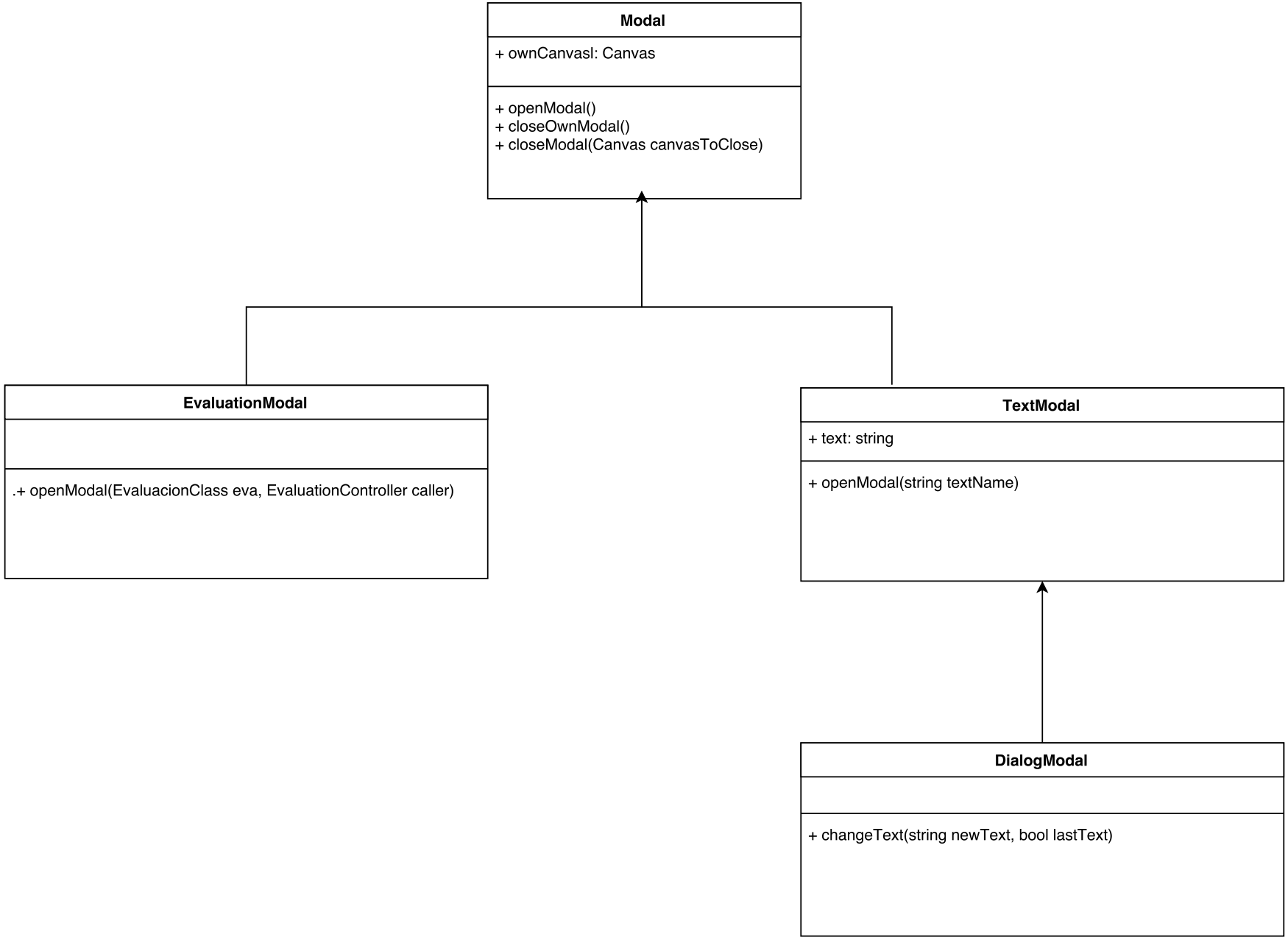

Figura 5.14: *Diagrama de clases de la jerarquía creada para la manipulación de ventanas modales.*

En este punto es importante nombrar que cada objeto de la clase Modal, o sus herederas, tendrá asociado un solo *Canvas* a utilizar, aunque se permitirá cerrar otros no pertenecientes al objeto.

Una vez creado el objeto se podrán utilizar 2 métodos destinados a manipular la visibilidad del *Canvas* asociado. El método *openCanvas*() es quien se encarga de cambiar el estado de visibilidad de éste a visible, mientras que el método *closeOwnCanvas*(), es quien se encarga de cambiar su visibilidad a oculta. Como se dijo anteriormente, la clase permite cerrar otros *Canvas* no asociados a ella. Esto se logra mediante el método *closeCanvas(Canvas)*, el cual, cambia el estado de visibilidad del objeto pasado a oculto. Es importante notar la diferencia sintáctica de los elementos de cerrado. El método *closeOwnCanvas* cierra el objeto asociado a él, mientras que el método *closeCanvas* lo hace con un objeto externo pasado como parámetro.

5.5.2. *EvaluationModal*

En la pantalla del listado de todas las aventuras disponibles para el jugador, es la clase *EvaluationModal* quien se encarga de abrir la ventana que contiene la información de la autoevaluación correspondiente a la aventura seleccionada, así como de llenar toda esta información. Para hacerlo, crea un nuevo método *openModal* que recibe como parámetro una instancia de la clase *EvaluationClass* y una instancia de la clase *EvaluationController*.

La instancia de *EvaluationClass* es quien contiene todos los datos necesarios de la autoevaluación. Se utilizan los datos provistos para esta clase para poder completar los textos donde se mostrarán la información al usuario.

La instancia de la clase *EvaluationController* es utilizada para poder configurar el botón de inicio de la aventura. Cada instancia de esta clase, posee un método que, junto con los datos que ya contiene, utiliza la interfaz de comunicación explicada en la sección 5.8, para poder obtener todos los desafíos necesarios para realizar la aventura seleccionada.

Una vez completadas todas las acciones para la ventana modal, se llamará al método *openModal* de la clase padre. Los métodos para cerrar las ventanas modales no necesitaron cambios.

5.5.3. *TextModal*

La parte de la jerarquía correspondiente a la clase *TextModal* es sin lugar a dudas la más utilizada a lo largo de todo Desafiate. Esta clase es quien se ocupa de las ventanas modales simples que solo poseen un texto relacionado. Es utilizada mayormente para comunicar sobre diferentes sucesos al jugador, como si algo salió mal en la comunicación con el servidor, o también es utilizado como *feedback* para informarle que la petición HTTP está siendo realizada y tiene que esperar una respuesta.

Para realizar esto, la clase define un constructor propio el cual, además del *Canvas* a manipular, recibe un texto que será mostrado dentro del *Canvas*. Luego define un nuevo método *openModal*, el cual recibe como parámetro el nombre del campo texto que contendrá el texto anteriormente definido en el constructor. De esta forma, al llamar al método base *openModal*, el texto ya aparecerá como parte de la ventana modal una vez abierta.

5.5.4. *DialogModal*

La clase *DialogModal* es la única que no hereda directamente de *Modal*, sino que lo hace a través de su clase padre *TextModal*. Esta clase está dedicada a manejar los cuadros de diálogos que tienen los personajes a largo de las islas recorridas en las aventuras, así como también de la historia contada apenas inicia una aventura. La decisión de heredar de la clase descrita en la subsección 5.5.3, se debe a las similitudes entre ambos funcionamientos. Se puede ver a los cuadros de diálogos como ventanas modales que van cambiando el texto a mostrar y solo se cerrarán una vez que haya mostrado el último texto del dialogo. Es por esto que esta clase no redefine ningún método o ninguna variable de su clase padre.

Para poder mostrar los cambios de los diálogos se creó un nuevo método llamado *changeText*, el cual recibe un nuevo texto a mostrar, y una variable de tipo booleano para saber si es el último texto a mostrar. Esto último es importante, ya que la ventana posee una imagen de una flecha que sirve para comunicarle al jugador que todavía quedan más textos para mostrar en ese dialogo.

5.6. Resolución de desafíos

La resolución de desafíos es otra de las partes centrales de Desafiate. Como se pudo ver en la sección 4.3.4, cada isla tiene un desafío, el cual será presentado por el habitante de la isla en forma de una historia. El jugador resolverá el desafío, y el protagonista se irá de la misma en su barco. La historia, ambientación de la isla, ubicación del muelle, posición inicial y final del barco son todos elementos que varían de una isla a otra. Sin embargo realizar la programación necesaria para cada isla sería un trabajo tedioso y largo. Es por eso que se buscaron puntos en común para lograr generar una estructura genérica a la que solo haga falta cambiar algunos atributos y de esa forma simplificar la creación de las diferentes islas.

Para lograr lo anteriormente dicho, se creó una jerarquía que representa el comportamiento del barco en las diferentes etapas en las que incurre: llegar al muelle de la isla, enfocar al isleño, resolver el desafío, irse de la isla y guardar los datos pertinentes. Incluso algunas de estas etapas coinciden con las llevadas a cabo por la escena de visualización de resultados, con lo que se resolvió agregar el comportamiento propio de esta escena a la jerarquía.

A lo largo de la resolución de esta solución se han utilizado diferentes patrones de

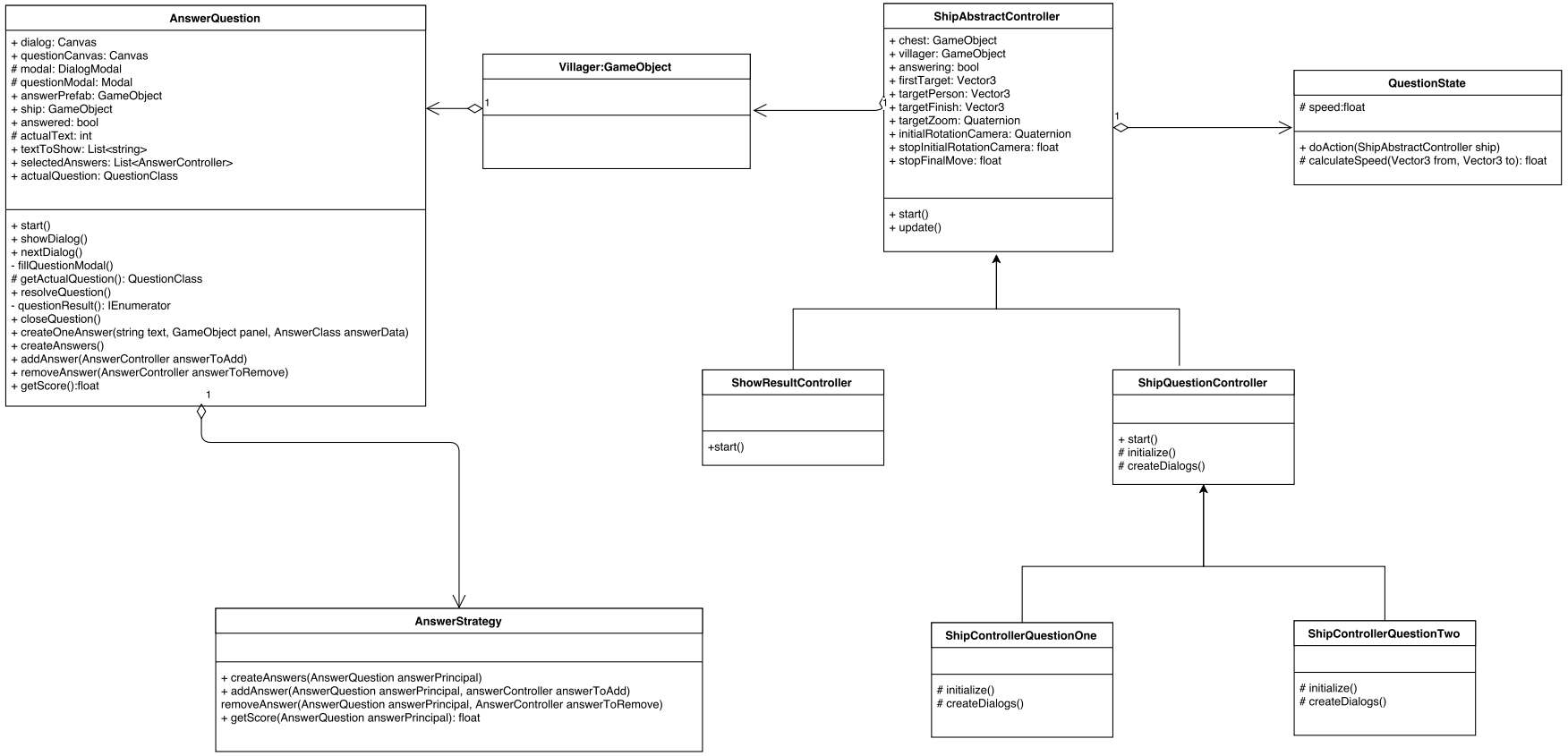

Figura 5.15: *Diagrama de clases de la jerarquía creada para manejar el funcionamiento por etapas de la resolución de desafíos.*

diseño para facilitar el desarrollo. A continuación se detallará la jerarquía mostrada en la figura 5.15 y se explicarán los patrones utilizados a medida que avanza el análisis siguiente. Tanto la clase *AnswerStrategy*, como la clase *QuestionState*, representan jerarquías propias que implementan cada una un patrón de diseño específico. Sus diagramas de clases serán mostrados más adelante a la hora de especificarlos. No se ha agregado esta información en la figura 5.15 para facilitar su comprensión.

5.6.1. *ShipAbstractController*

Esta clase es la clase central en lo que respecta al control del barco en las diferentes etapas por las que pasa tanto en las escenas de las islas, como en la escena para mostrar el resultado.Es una clase que detalla todos los datos necesarios que cada uno de sus herederos tendrán que sobrescribir para poder ir realizando todas las acciones. Las diferentes variables que posee de tipo *Vector3* y *Quaternion* son las que indicarán los diferentes puntos a donde se tendrá que mover el barco y su cámara asignada a lo largo de las etapas. El resto de las variables, exceptuando *chest, villager* y *answering* serán utilizadas para recordar ciertos datos iniciales para poder restablecer algunas posiciones luego de realizar algunos movimientos y rotaciones con respecto a la cámara. La variable *villager* será utilizada para acceder a la clase que se encarga de mostrar el dialogo de la historia y resolver el desafío. Esto será explicado con más profundidad en la subsección 5.6.3.

Todo el funcionamiento de la clase se limita al método *update*, donde lo único que hace es llamar al método *doAction* del objeto *QuestionState* que tenga relacionado en ese momento, siempre y cuando el jugador no se encuentre en un momento donde no se deba realizar ninguna acción, como por ejemplo, mientras está resolviendo algún desafío. En este caso se utilizó el patrón de diseño *State*, donde cada uno de los estados posibles representa una de las diferentes etapas por las que se pasa a la hora de resolver los desafíos, o de mostrar los resultados. Cada uno de los estados sabrá, con los datos aportados por el objeto de la clase *ShipAbstractController*, como realizar su acción y cuando terminarla y pasar al siguiente estado. La utilización de este patrón además de permitir resolver fácilmente cada una de las etapas por las que pasa el barco, también permite extender esta funcionalidad de una forma sencilla ya que solo hace falta agregar una clase a la jerarquía y después modificar el orden en que son llamados cada uno de los estados.

5.6.2. *QuestionState*

Esta clase es la base de todo el patrón *state*. Es una clase abstracta que define el método básico *doAction* que utilizarán todas las clases de la jerarquía. Además define la variable protegida *speed*, la cual es utilizada para saber la velocidad a la que se tendrá que mover el barco. Esta variable toma normalmente su valor del otro método definido en esta clase, el cual es *calculateSpeed*. En la figura 5.16 puede verse el diagrama de clase con todas las

clases pertenecientes a esta jerarquía. Cada clase será explicada a lo largo de la presente subsección.

La clase *InitialShipMovement* es la encargada de mover el barco hacia el muelle destino de cada isla. Tomará los datos de la posición inicial, y de la posición del muelle que posee la clase *ShipAbstractController* para calcular la distancia, el trayecto y la velocidad a moverse. Esta etapa también puede encontrarse en la visualización del resultado de la aventura, en donde el barco también deberá llegar al muelle destino. Es por esto que se transformó dicha clase en una jerarquía que implementa el patrón *template method*. El método *doAction*, actuará siempre de la misma forma, hasta llegar a destino, momento en el cual ejecutará el método *nextStep*. Este método seleccionará el siguiente estado a realizar según el objeto que lo ejecute, así como guardará la posición y la rotación relativas de la cámara con respecto al barco.

La clase *InitialCameraMovement* será la encargada de posicionar la cámara una vez que el barco haya llegado al muelle. Actúa de una forma parecida a la clase *InitialShipMovement*. Tomará los datos de la clase *ShipAbstractController* para poder ubicar la cámara en su lugar correspondiente. Esto variará de acuerdo a la escena en que se encuentre. En las islas donde se resuelven los desafíos la cámara terminará apuntando al habitante de la isla que presentará el desafío. En la escena de visualización de los resultados, la cámara terminará apuntando al cofre que simbolizará los tesoros encontrados en la aventura. Es por esto que también se transformó esta clase en una jerarquía que actuará según la escena donde se encuentre. Hace uso del patrón *template method* de la misma forma que la clase *InitialShipMovement*, y serán las clases herederas quienes se encargarán de implementar el método *nextStep*. Es en este método donde las clases herederas difieren bastante en su comportamiento. Mientras que la clase *CameraResultMovement* solo pasará al estado donde se muestran los resultados de la aventura, la clase *CameraQuestionMove* frenará toda acción de los estados, pasará al estado de *FinishMove*, el cual esperará que se termine de resolver el desafío, e iniciará el dialogo con el habitante que se encuentra en el muelle.

La clase *OpenResultModal* es quien se encargará de mostrar todos los resultados de cada uno de los desafíos resueltos en la aventura. Como su acción debe solo realizarse una vez, se encarga de activar la variable de la clase *ShipAbstractController* que frena las ejecuciones de las acciones de los estados.

La clase *FinishMove* es quien se encarga de realizar los movimientos del barco necesarios una vez terminada la resolución del desafío en la isla. Esto lo hace tomando la información necesaria hacia donde tiene que ir el barco para finalizar la isla. Además toma los datos guardados durante la ejecución de la clase *InitialShipMovement* para restablecer la cámara a su posición relativa original. Esto lo hace a través de los 2 métodos declarados para esas acciones. Una vez que llega a destino se pasará al último estado.

Este último estado es manejado por la clase *SaveQuestion*. Su función es la de guardar en

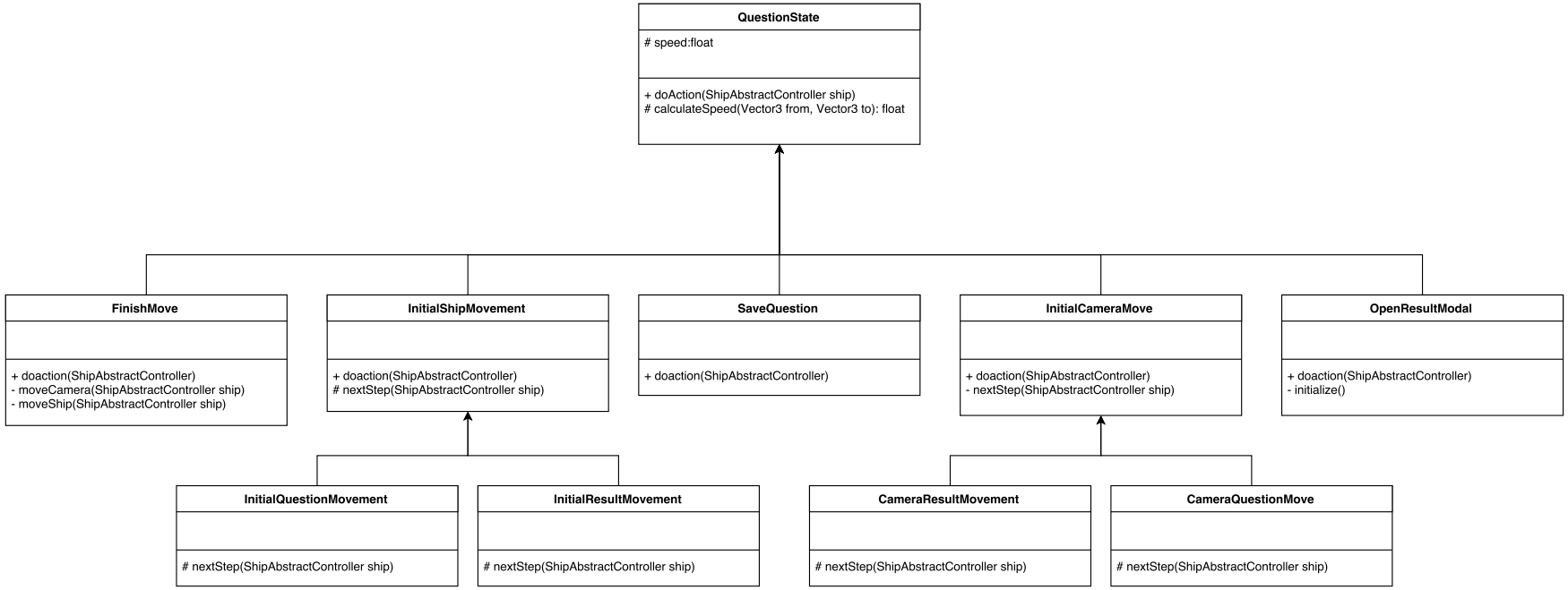

Figura 5.16: *Diagrama de clases de la jerarquía del patrón state.*

el dispositivo los datos de la resolución del desafío de la isla, y una vez realizado esto pasar al siguiente desafío o a la visualización de los resultados según corresponda.

5.6.3. *AnswerQuestion*

Una vez ya analizados todos las etapas por las que pasa el barco para cumplir su cometido, es tiempo de analizar la última: la resolución del desafío. La encargada de realizar y controlar toda esta etapa es la clase que da nombre a esta sección. Su ejecución comienza cuando el estado de *CameraQuestionMove* abre el dialogo con el habitante de la isla. La clase controlará este dialogo a través del método *nextDialog*. Este método irá tomando los textos guardados en la lista de *strings*, la cual fue llenada previamente por la instancia de la clase *ShipQuestionController*, y los pasará a la instancia de la clase *DialogModal* asociada con el *Canvas* de diálogo de la escena. También será la encargada de controlar cuando se trata del último dialogo, momento en el cual abrirá la ventana modal encargada de mostrar el desafío planteado al jugador.

Es en este punto donde entra en acción el patrón *strategy*. La resolución de cada desafío será diferente según el tipo de pregunta que haya creado el docente. Estos tipos de pregunta en principio solo son dos: *multiple choice* y verdadero y falso. Estos dos tipos varían mucho en la forma de manejar los distintos aspectos de la resolución. Mientras que el tipo de pregunta *multiple choice* puede admitir una o más opciones como respuesta, el verdadero y falso solo admite una única opción, con lo cual, no tiene sentido dejarle al jugador elegir las dos opciones que le son mostradas. Esto también tiene implicancia en la forma de calcular el resultado final. Mientras que en el verdadero o falso solo hace falta ver si eligió la respuesta correcta, en el *multiple choice* hay que recorrer todas las respuestas seleccionadas y verificar si son ciertas. En caso de encontrarse una respuesta incorrecta el resultado final será 0, y en caso de solo haber seleccionado respuestas correctas pero no todas las posibles que existían, el resultado final será un porcentaje del puntaje total. Es por esto que se resolvió utilizar el patrón *strategy* para poder pasar la resolución de cada uno de los tipos a una clase en particular que manejará la estrategia que tiene asociada a ella. El diagrama de clases de este patrón puede verse en la figura 5.17. Una vez que resuelto el desafío y que el objeto del patrón *strategy* calculó el resultado, el objeto de la clase de la presente subsección volverá a ceder el control al estado *FinishMove*.

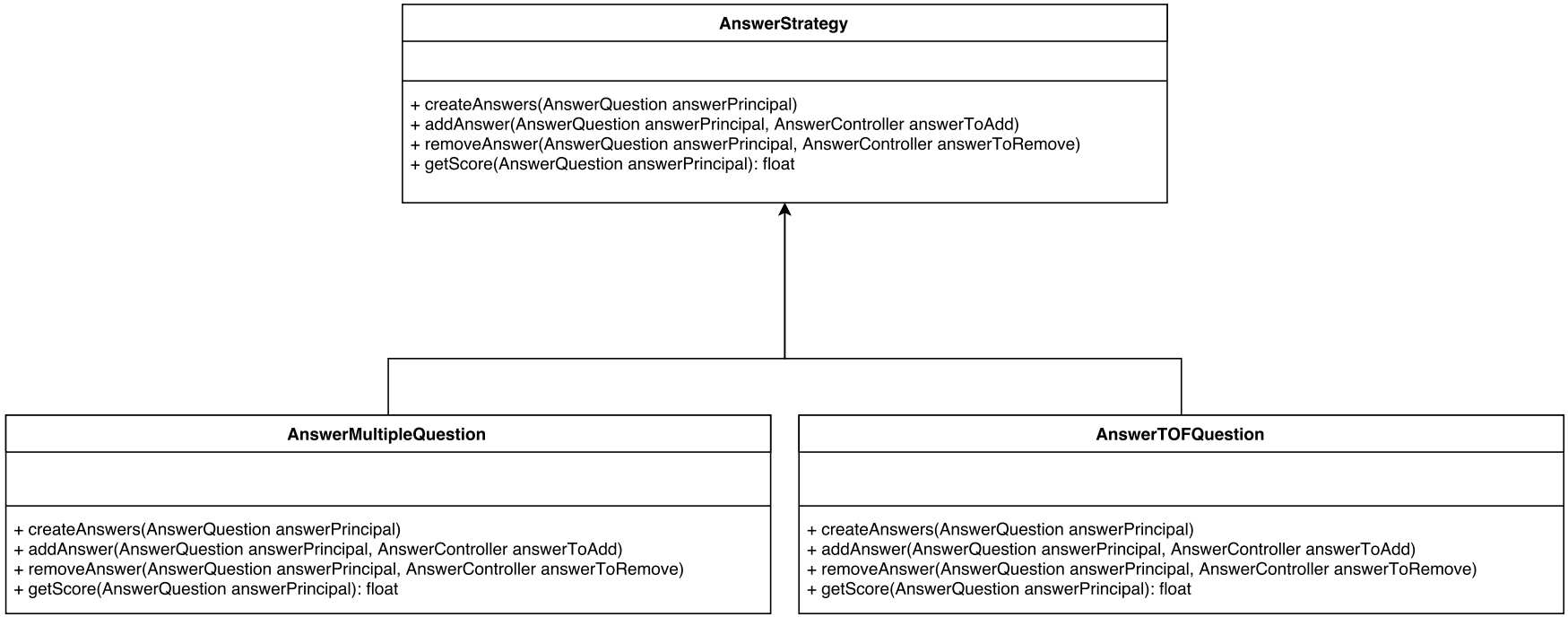

Figura 5.17: *Diagrama de clases de la jerarquía del patrón strategy.*

5.7. Datos a guardar

Para finalizar con la capa del juego, solo queda analizar la forma en que Desafiate guarda los datos para poder utilizarlos más tarde cuando los necesite. Esto lo hace utilizando las clases básicas que ya provee *Unity* para el manejo de *JSON* y para el guardado de datos en archivos. Si bien estas clases poseen diferentes limitaciones que serán explicadas más adelante, la funcionalidad que ambas proveen resultó adecuada para el funcionamiento que posee por el momento Desafiate.

El manejo de *JSON* es una parte importante que complementa a la comunicación realizada con el servidor de IDEAS. Todo lo que el servidor responde llega en formato *JSON*, y cuando hay que enviarle algo al servidor que está compuesto de varios datos, como cuando se le manda la resolución de la aventura, se envía en el mismo formato. Es por esto que es muy importante contar con una interfaz que se encargue de manejar estos datos, para poder utilizarlos fácilmente. La clase base de *Unity*, *JsonUtility* resuelve esto de una forma sencilla pero con algunas limitaciones. Su funcionamiento es el siguiente. Posee dos métodos principales, *FromJson* y *ToJson*. El primer método se encarga de crear un objeto y completar todos sus datos en base a un *JSON* que se pasa como parámetro. En caso de no encontrar relación entre un dato que se relacione con una variable, éste será descartado y no se utilizará para este propósito. En el caso contrario, cuando es la variable quien no encuentra relación, entonces se dejará con el valor por defecto que se haya asignado. El segundo método de interés hace justamente la función inversa. Se encarga de armar una cadena en formato *JSON* en base a la clase que recibe como parámetro.

Hasta acá la clase *JsonUtility* parece ideal, pero tiene algunas limitaciones. Solo puede manipular los *JSON* cuando se utiliza una clase, no es posible utilizar el *JSON* de forma de obtener un arreglo con diferentes datos. Lo que sí resulta posible es completar un arreglo dentro de una clase si este encuentra una relación con el *JSON* utilizado. Este arreglo puede ser contener objetos tanto de tipos básicos como objetos complejos de clases creadas por el usuario. Otra cuestión a tener en cuenta es que es necesario que las clases utilizadas sean marcada como "serializables", sino estas serán ignoradas por lo métodos utilizados. Esto incluye también las clases relacionadas a través de las variables de la clase a convertir. Sus objetos solo verán sus datos completados si su clase es "serializable". Estas cuestiones no resultaron una complicación a la hora del desarrollo, ya que todos los datos se pudieron pasar a sus clases correspondientes que luego fueron utilizadas a lo largo de todo el juego. Sin embargo, podría llegar un momento donde la imposibilidad de utilizar un arreglo, y tener que crear una clase que solo contenga el arreglo se vuelva una complicación importante.

La utilización de *JSON* también fue muy útil a la hora de utilizar la clase de *Unity* que se encarga de guardar los datos en la dirección que corresponde a cada dispositivo. *PlayerPrefs* es una clase encargada de resolver todas las cuestiones del guardado de datos. Reconoce automáticamente el dispositivo en el que corre el juego y en base a eso define

la dirección por defecto donde se guardarán los datos. El manejo que otorga es parecido al de un diccionario, obtiene y guarda los datos a través de una clave otorgada al momento de realizar la acción, lo cual resulta sencillo a la hora de usar la clase. Sin embargo, posee una limitación con respecto a los datos que puede guarda, ya que solo puede manipular tres tipos de datos básicos: cadenas de texto, enteros y números en punto flotante. Es por esto que resulto vital la utilización del formato JSON para poder guardar datos complejos como objetos enteros que incluían arreglos de otros objetos. Para simplificar todavía más la labor se decidió implementar una clase que haga la función de interfaz, que será la encargada de guardar los datos que le pasen. Esta interfaz está diseñada para que el día de mañana se encargue de cualquier tipo de dato y pueda resolver cualquier problema del desarrollador. El diagrama de clase de esta interfaz se puede ver en la figura 5.18.

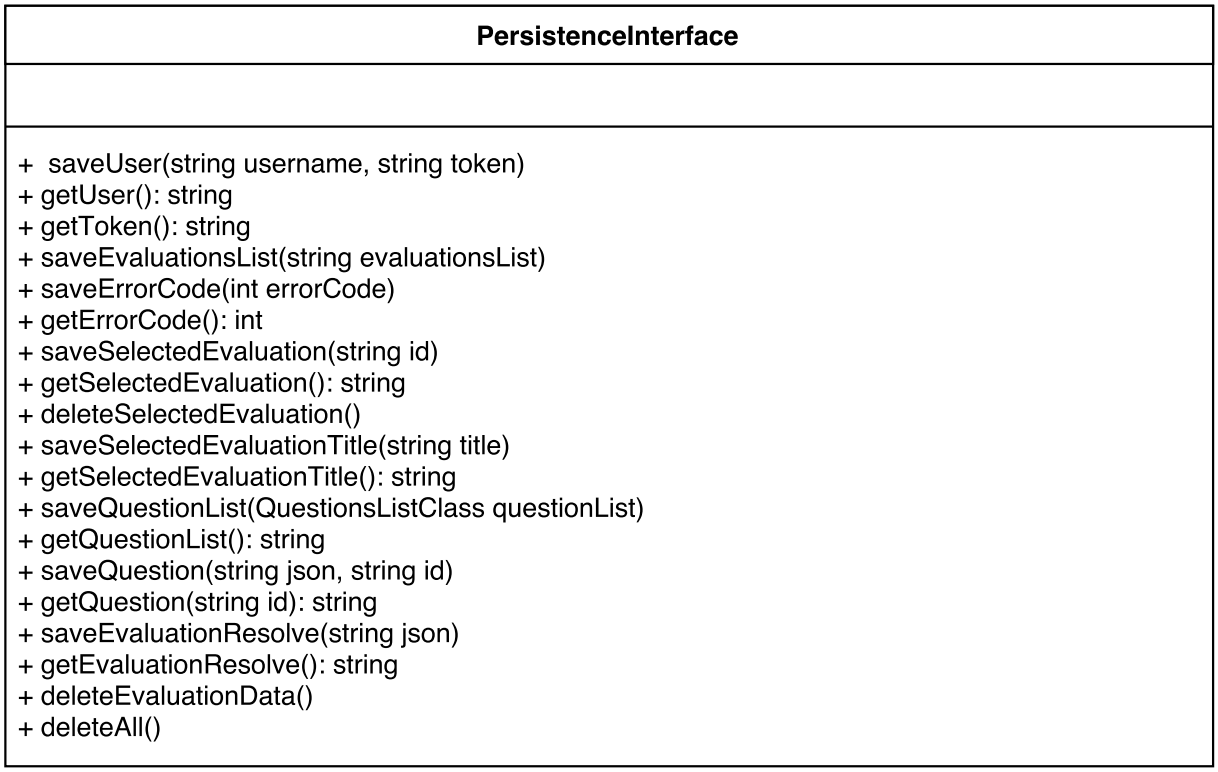

Figura 5.18: *Diagrama de clases de la interfaz de persistencia.*

5.8. Comunicación e integración con el EVEA IDEAS

Una de las partes centrales de esta tesina es la comunicación entre Desafiate y el sistema IDEAS. Es éste EVEA quien va a nutrir al juego de las diferentes aventuras y la composición de ellas. Es por ello que la arquitectura de la comunicación entre estos 2 sistemas tiene que estar bien planeada y aceitada para evitar problemas y poder extenderla en el futuro. También hay que considerar que tanto IDEAS como Desafiate son sistemas nuevos y es posible que ambos vayan evolucionando en el futuro. Por último también es importante que Desafiate

no se encuentre atado a IDEAS, ya que tiene como objetivo ser una herramienta de apoyo para la educación, por lo tanto es importante que en el desarrollo se tenga en cuenta que es posible que en el futuro Desafiate obtenga sus desafíos de otros EVEAs también.

Teniendo en cuenta estas cuestiones se decidió realizar una capa con una clase que cumple la función de una interfaz encargada de realizar todas las comunicaciones necesarias. Esta clase se asocia a un objeto en la escena, lo cual se hace con el objetivo de poder relacionar la clase con algunos objetos necesarios en la escena. Además posee métodos, en los cuáles, cada uno se encargará de realizar la comunicación con IDEAS, conseguirá los datos y los manipulará según sea necesario. La clase, llamada *ApiCaller* (ver figura 5.19), posee tres variables, una de las cuales es pública y las otras dos privadas. La variable pública hace referencia al objeto de tipo *Canvas* ubicado en la escena, y cuya función es la de mostrar el cartel modal de espera, el cual es una de las variables privadas, para dar un *feedback* al jugador de que se está realizando una acción que puede tomar un momento. Al comienzo de cada comunicación se activará este *feedback* y se cerrará una vez esta haya concluido. Este *Canvas* operará con la mecánica de las ventanas modales explicadas en la sección 5.5.

La otra variable privada es en realidad donde ocurren todas las modificaciones. Esta variable es una instancia de la clase abstracta *ApiCommunication*, la cual define todos los métodos necesarios para la comunicación. Al ser una clase abstracta la idea es que sean las clases herederas quienes implementen la funcionalidad de los métodos declarados en ella. Esto es así porque al cambiar de EVEA es muy probable que la funcionalidad implementada en un método para IDEAS, por ejemplo, no sirva para otro EVEA. Es por esto que se creó de esta forma para poder fácilmente adaptar el sistema a cualquier EVEA. Esta jerarquía y su relación con la clase *ApiCaller*, puede verse en el diagrama de clases mostrado en la figura 5.19.

Para realizar la comunicación con IDEAS se utiliza el concepto de *API REST*. *RESTful* aparece por primera vez en Fielding (2000). En este trabajo de doctorado se analizaron diferentes estilos de arquitectura de comunicación y Fielding propuso crear un nuevo estilo basándose en el protocolo *HTTP*. Cada vez que se necesita un dato, el cliente deberá hacer una petición *HTTP* al servidor que contiene el servicio *RESTful*, el cual le responderá normalmente en un formato *XML* o *JSON*. En caso de haber un error, el cliente se guiará por el número de estado de la respuesta provista por el servidor. Fielding también propuso una serie de principios a seguir. En Navarro y da Silva (2016) puede verse un buen resumen de estos principios, y es en este artículo en el que se basa el siguiente análisis de ellos:

- Los servicios *RESTful* no poseen estado. Esto significa que cada petición que se realice al servidor será única, aportará toda la información necesaria, y no podrá tomar ventaja de ningún contexto guardado en el servidor.
- Los servicios *RESTful* tienen una interfaz uniforme. Este principio se creó para que

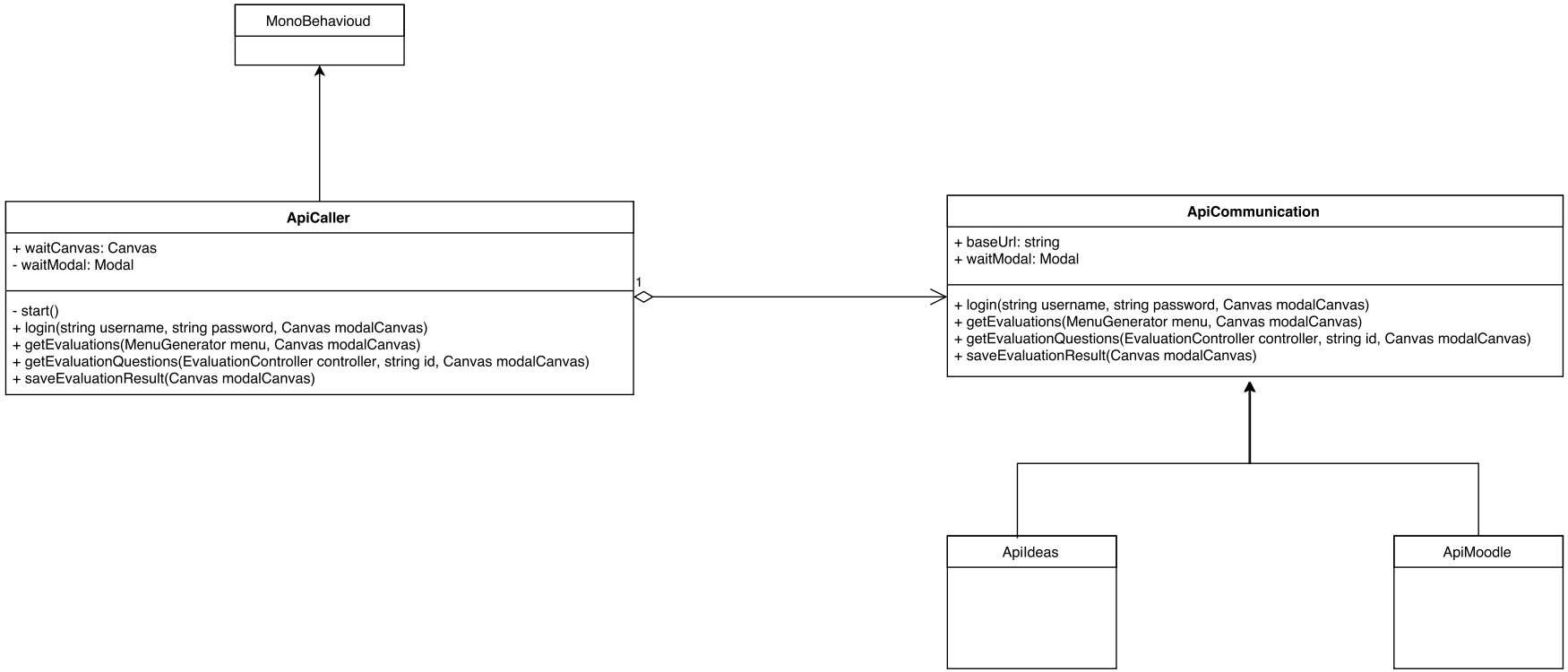

Figura 5.19: *Diagrama de clases de la interfaz de comunicación. La clase ApiMoodle no ha sido implementada pero se agrego para mostrar de una forma mejor la jerarquía existente.*

solo se utilicen los métodos *HTTP* de *GET, POST, DELETE*, y *PUT*. Cada uno de estos métodos puede encontrar una relación con los encontrados en el concepto *CRUD* de base de datos.

- Las arquitecturas basadas en *REST* se construyen a partir de recursos que tienen una identificación única mediante *URIs*.
- Los componentes *REST* manipulan los recursos para poder intercambiar representaciones de ellos. Para representar a los recursos suelen utilizarse representaciones en *XML* o *JSON*.

Con lo anteriormente dicho queda claro que para poder utilizar *REST* para la comunicación entre Desafiate e IDEAS, hay que utilizar peticiones *HTTP*. Para poder realizar esto en *Unity* la jerarquía de comunicación hace uso de una clase provista por el mismo motor llamada *WWW*.Esta clase provee los mecanismos necesarios para hacer peticiones usando los métodos *GET* y *POST*, aunque no soporta ni *PUT* ni *DELETE*. Esto no es de gran importancia en el desarrollo del juego ya que éste solo se encargará de pedir datos mediante *GET*, y enviar los resultados y datos de inicio de sesión mediante *POST*. Otra particularidad muy útil de esta clase es que permite usar otros protocolos comúnmente usados en web como *FTP* y *FILE*, aunque estos protocolos no son necesarios para el desarrollo de Desafiate, por el momento.

Como es de esperar de cualquier petición *HTTP*, estas son realizadas de forma asincrónica, de manera que pueden realizarse otras acciones en la espera de una respuesta. Sin embargo, estas comunicaciones suelen ocurrir por una acción realizada por el jugador. Es por esto que es importante utilizar un *Canvas* y una ventana modal para informar al usuario de que la petición se está realizando que, como se dijo anteriormente, será más explorado en la sección 5.5.

Todas las peticiones que se hacen a la *API REST* devolverán una respuesta compuesta por un texto con formato *JSON*. La forma de convertir este texto para que pueda ser utilizado por Desafiate se vio ya en la sección 5.7. Todas las peticiones, exceptuando la que se realiza para el inicio de sesión, poseen un dato especial en su cabecera. Este dato especial, llamado *token* de autenticación, es el encargado de identificar al usuario conectado en IDEAS. Cuando el jugador realiza el inicio de sesión, y los datos son correctos, el servidor devuelve este *token* codificado, el cual luego debe ser utilizado siempre que se desean obtener datos relacionados al usuario. Este *token* tiene un tiempo de vencimiento, con lo que es necesario renovarlo cada cierto tiempo. De esta forma se mantiene la seguridad en el sistema y no se permite a un usuario obtener datos de otro.

En este punto alguien podría preguntarse entonces cuál es el objetivo de la clase *ApiCaller*, si la encargada de toda la funcionalidad es la jerarquía de la clase *ApiCommunication*. La respuesta a esto es simple. Si bien, al crear la jerarquía, todos los hijos entienden automá-

ticamente los mismos métodos, estos actúan de forma diferente según las necesidades del servidor. Al cambiar un servidor, será necesario también modificar el tipo de la clase creada en cada una de las invocaciones de estos métodos. Al agregar a la clase *ApiCaller* como intermediaria, este cambio solo debe realizarse en un solo lugar (método *start* que posee), en vez de realizar cambios a lo largo de todas las clases de todas las escenas.

5.9. IDEAS

En esta sección se detallará brevemente la herramienta que provee el EVEA IDEAS para la creación de las autoevaluaciones de los alumnos. Este EVEA es una evolución surgida del proceso de reingeniería al que fue expuesto el EVEA WebUnlp, el cual fue desarrollado en la Facultad de Informática de la Universidad Nacional de La Plata, al igual que IDEAS. Se lanzó su primera versión al público general en Febrero de 2017.

Cada curso de IDEAS tiene la posibilidad de contar con la herramienta de autoevaluaciones, y es el docente a cargo quien toma la decisión de incluirla. En caso de que se desee utilizar la herramienta en su curso, los docentes pueden crear múltiples autoevaluaciones. El proceso de creación de una autoevaluación se compone de 4 pasos los cuales se detallarán a continuación.

En el primer paso, el docente determina la información básica de la autoevaluación. Esto es, se elige el título, los objetivos que se persiguen, y los temas relacionados que abarca.

En el segundo paso, se realizan las configuraciones básicas de la autoevaluación. Estas configuraciones incluyen el tipo de puntaje que utilizará la autoevaluación, pudiendo ser numérico o una escala de valores previamente definida. Además se puede elegir la forma en la que el alumno podrá responder las preguntas, aunque esto no es tenido en cuenta en Desafiate por lo que no se detallará más en este punto.

El tercer paso es el más importante de todo el proceso ya que es cuando se crean las preguntas que responderá el alumno. Aquí el docente puede crear una pregunta nueva entre los tipos *multiple choice* y verdadero y falso, le asigna un valor a cada pregunta, y configura las respuestas en caso de que se haya elegido el primer tipo mencionado. El porcentaje de puntos asociado al conjunto de respuestas no puede superar el 100 % del puntaje total de la pregunta, así como el puntaje total de las preguntas no puede sobrepasar el máximo puntaje configurado para la autoevaluación en el paso anterior. También se permite reutilizar preguntas que se hayan creado previamente en autoevaluaciones de cursos en los que el docente es parte. Una vez que todas las preguntas están completas y se llegó al puntaje necesario se puede pasar el último paso.

El paso cuatro no posee funcionalidad alguna con respecto a la modificación de la autoevaluación. Su función es la de ofrecer un resumen para que el docente pueda verificar el resultado del proceso de creación y defina si hay que realizar algún cambio.

Una vez que la autoevaluación fue creada exitosamente, ésta no se muestra automáticamente a los alumnos. El docente debe publicar la autoevaluación a partir de una fecha en la que será visible para los alumnos. De manera opcional el docente también puede marcar una fecha de fin, momento en el cual la autoevaluación ya no estará disponible para responder.

Desafiate tomará como aventuras aquellas autoevaluaciones publicadas. Más adelante también se dará la opción al docente que decida si quiere incorporarla como una aventura de Desafiate o no.

5.10. Conclusiones del capítulo

En este capítulo se ha presentado tanto la arquitectura planificada en capas para el juego Desafiate y su integración con un EVEA, como los aspectos de implementación vinculados al uso del motor *Unity*.

Unos de los aspectos a destacar es que se ha buscado flexibilidad en la arquitectura para que la capa del EVEA pueda ser a futuro instanciada por cualquier otro EVEA atendiendo a las particulares que requiera cada uno. Al mismo tiempo los detalles de implementación dan cuenta de la utilización de patrones de diseño y la inclusión de clases *ad-hoc* para lograr alcanzar los objetivos planteados para el juego a partir de un diseño flexible.

Se cree que si bien ha llegado a desarrollar un prototipo completamente funcional pero inicial, la arquitectura y las decisiones de la implementación permitirán evolucionarlo fácilmente a partir de los datos recogidos en las experiencias de prueba que se detallan en el próximo capítulo.

CAPÍTULO 6

EXPERIENCIAS CON EL USO DE DESAFIATE

6.1. Introducción

En el capítulo 2 se pudo ver como una de las ventajas que provee el uso de juegos serios en la educación es que el alumno sienta una mayor motivación con respecto al proceso de aprendizaje. En el caso particular de los juegos serios con orientación a la evaluación, se dijo que estos juegos tienen la posibilidad de reducir el estrés causado por el proceso en el que el alumno se encuentra.

Es por esto que resulta necesario realizar pruebas para poder comprobar que Desafiate esté cumpliendo con sus objetivos de mejorar la motivación de los alumnos, y a su vez que éstos vean reducidos su nivel de estrés al momento de realizar una autoevaluación. Estas pruebas también son importantes para comprobar que aspectos de Desafiate le resultan interesantes al alumno y cuáles son los aspectos que le gustaría mejorar o agregar para que la experiencia sea más disfrutable. Las experiencias realizadas han aportado también a la mejora de la usabilidad del juego ya que se recogieron datos sobre cómo las dificultades encontradas a lo largo de las sesiones. A su vez es necesario contar con una opinión proveniente de los docentes para comprobar si les resulta interesante contar con esta herramientacomo complemento para la autoevaluación de sus alumnos y cuál es su opinión sobre incluir juegos serios en procesos educativos, esto ayuda a visualizar la predisposición de los docentes para utilizar este tipo de juegos.

En la primera sección de este capítulo se explicará cómo se realizaron las pruebas realizadas con el prototipo desarrollado de Desafiate tanto con alumnos como con docentes del curso de ingreso de la Facultad de Informática de la Universidad Nacional de La Plata. En las siguientes secciones se detallará cada experiencia en particular junto con los resultados

que se obtuvieron.

6.2. Metodología de las experiencias realizadas

Para evaluar Desafiate, se realizaron dos experiencias con el prototipo de Desafiate, ambas dentro de un mismo contexto educativo pero con objetivos y participantes diferentes. Estas pruebas se hicieron aprovechando que durante el transcurso de esta tesina se estaba realizando el curso de ingreso a las carreras de la Facultad de Informática de la Universidad Nacional de La Plata, durante el transcurso del mes de Febrero. Durante este tiempo los alumnos tienen que cursar tres módulos diferentes donde se les dan contenidos básicos que se articulan con las primeras materias de la Facultad. Los tres módulos que se realizan a lo largo del curso son:

- **Expresión de problemas y algoritmos - EPA:** en este módulo el objetivo es brindar una metodología básica para la resolución de problemas utilizando una computadora. Se comienza analizando el problema a resolver, se propone una especificación clara de la manera de solucionarlo y finalmente se expresa esa solución en un lenguaje de programación.
- **Conceptos de organización de computadoras - COC:** este módulo esta pensado para analizar los aspectos básicos de las arquitecturas físicas de las computadoras, sus periféricos y los mecanismos de comunicación CPU-Memoria-Periféricos.
- **Matemáticas 0 - MAT0:** en este módulo los alumnos repasan los conceptos matemáticos básicos vistos en el nivel medio con el fin de lograr una nivelación de conocimientos.

Como se puede ver, cada módulo tiene objetivos distintos, por lo que los contenidos dados son diferentes en cada uno. En este contexto, se decidió realizar la prueba con los contenidos de uno solo de los módulos, de manera de facilitar la logística para llevar adelante dichas experiencias. Por esto se determinó utilizar el módulo de EPA, ya que es uno de los módulos que los alumnos tienen menos andamiaje.

Como para las pruebas a realizar era necesario contar con un servidor con el cual comunicarse, se configuró un servidor de prueba en el cual se alojó una versión en desarrollo del sistema IDEAS, el cual permitía la comunicación con Desafiate. Dentro de este servidor se utilizó un curso nuevo en el EVEA IDEAS creado específicamente para las experiencias realizadas. Dentro de este curso se creó una autoevaluación teniendo en cuenta los contenidos vistos hasta el momento de la realización de la autoevaluación con Desafiate. De esta forma, los alumnos podían evaluar sus propios conocimientos para analizar qué conceptos habían logrado aprender, y cuáles necesitaban reforzar. La

autoevaluación se creó en conjunto con los coordinadores de EPA, lo cual, trajo las ventajas de realizar la autoevaluación teniendo en cuenta las necesidades del contexto específico. La autoevaluación creada para la prueba se puede encontrar en el anexo 1.

Una vez creada la autoevaluación se realizaron experiencias en formato de sesiones de prueba de Desafiate. Los objetivos propuestos para estas sesiones se vincularon con conocer la opinión de los alumnos y docentes sobre el uso de juegos serios para el escenario educativo, en particular para procesos de autoevaluación, también saber su opinión sobre la usabilidad de Desafiate, aspectos que se consideran adecuados y aquellos a mejorar. Al mismo tiempo se pidió opinión para construir las historias del juego, personajes, tipos de islas que les gustaría encontrar, de manera tal que la próxima versión del prototipo considere estas opiniones para aumentar la motivación y la jugabilidad de la propuesta.

Se realizaron dos sesiones de pruebas con Desafiate, una realizada con alumnos invitados de diferentes comisiones del módulo de EPA, y otra realizada con diferentes docentes (profesores, coordinadores, ayudantes). Durante las sesiones se realizó registro mediante la observación participante por parte del tesista. Al mismo tiempo, luego de cada sesión se administró una encuesta. En el caso de los alumnos en papel y en el caso de los docentes a través de un cuestionario en GoogleForm.

Cada una de estas sesiones de pruebas tendrá dedicada una sección a continuación en donde se detallarán los datos más importantes correspondientes a cada una.

6.3. Experiencia con alumnos

Para esta experiencia se invitó a alumnos de diferentes comisiones y turnos del módulo de EPA para que participen de esta experiencia. Para no dificultar su concurrencia, ni complicar los horarios de los participantes, se decidió utilizar un horario donde todos puedan concurrir sin que se superponga con los horarios de clases. El único horario donde los alumnos de todas las comisiones no tenían que cursar era el horario de las 14:00 hs en cualquiera de los días de la semana. Los alumnos fueron invitados a partir del miércoles 15 de Febrero, con una invitación formal explicándoles el objetivo de la experiencia y las ventajas que podría tener ésta para ellos. Se eligió el día viernes 24 de Febrero, para poder invitar a la mayor cantidad de alumnos de las distintas comisiones.

Para poder realizar la experiencia de forma más rápida, se configuró el servidor para que los alumnos puedan bajar el *APK* del juego mediante una *URL* anteriormente definida y de esta forma poder evitar realizar la descarga en el momento de la prueba. Además se prepararon 7 *tablets* con el juego instalado y con la carga completa para que los alumnos que no posean un dispositivo *Android* puedan realizar la experiencia. A continuación se detallarán los datos más importantes a tener en cuenta de la experiencia realizada.

La experiencia duró aproximadamente una hora y se presentaron 6 alumnos que

accedieron a realizarla en forma voluntaria. Se inició la experiencia explicando acerca de Desafiate, el marco en el que fue creado, los objetivos que se persiguen y la etapa en la que se encuentra actualmente. Es importante aclarar que se les detalló que se trata de un prototipo en evolución y que interesaba su opinión para continuar evolucionando el juego, invitándolos a que sean lo más críticos y honestos posibles ya que se desea que esta sea una herramienta más para su proceso de aprendizaje. El prototipo al momento de las pruebas mostraba sólo dos tipos de islas y tres historias que se repetían, por lo que varios resultados han sido sesgados a esta situación. Se pueden observar algunas fotos obtenidas de esta experiencia en las figuras 6.1 y 6.2

Figura 6.1: *Uno de los participantes utilizando Desafiate.*

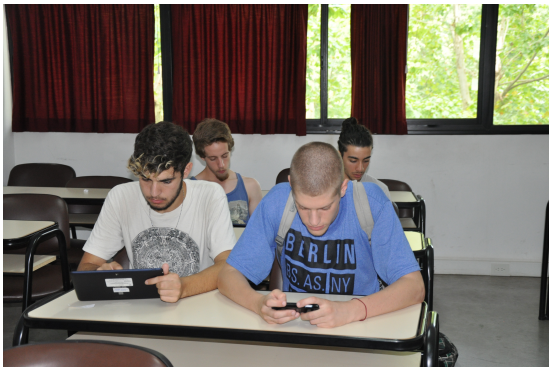

Figura 6.2: *Algunos de los participantes mientras realizaban la prueba.*

Una vez terminada la charla, se les ofreció a los participantes a elegir entre algunas de las *tablets* ya preparadas, o el celular en caso de que ya hayan realizado la descarga del juego previamente. En este punto, los alumnos que ya poseían el juego prefirieron utilizar sus dispositivos móviles, mientras que los demás participantes prefirieron utilizar las *tablets* ofrecidas. En esta etapa se pudo observar que la interfaz de juego no se adaptaba bien a dispositivos móviles de bajas resoluciones, lo que en algunos momentos dificultaba el uso correcto del juego. Este aspecto está siendo subsanado, pero de todas formas, la sesión pudo llevarse adelante aún con estos dispositivos.

Una vez que se terminó esta etapa los participantes realizaron un cuestionario, el cual se encuentra en el Anexo 2, para poder obtener sus opiniones acorde a los objetivos planteados para la experiencias con anterioridad. El cuestionario cuenta además con preguntas para poder conocer algunos aspectos del perfil de los participantes, así como también, datos sobre qué dispositivos informáticos poseen y su uso como apoyo en el estudio.

De la experiencia se obtuvieron los siguientes resultados:

- **Perfil de los participantes:** todos los participantes fueron personas de sexo masculino de entre 17 y 22 años de edad, todos de nacionalidad argentina y siendo una mayoría,

5 de 6 participantes, procedentes de colegios públicos.

- **Uso de dispositivos móviles:** se les preguntó sobre diferentes datos para evaluar el uso que hacen de diferentes dispositivos informáticos y de recursos digitales para su aprendizaje. Esto permite conocer si son alumnos ya vinculados al uso de tecnologías para aprender. Todos los participantes poseen por lo menos 2 dispositivos informáticos (por ejemplo smartphone y pc), y 4 de ellos tienen 3 dispositivos diferentes. El dispositivo que más comúnmente poseen es el *smartphone*, todos los participantes, estando en segundo lugar la *notebook*, la cual poseen 5 de 6 de los participantes. Este uso de dispositivos puede verse en la figura 6.3.

  Todos los participantes tienen acceso al uso de internet en sus hogares, y utilizan recursos digitales como apoyo para el estudio. En este punto, resulta interesante notar que todos hacen uso de páginas web y de la plataforma de videos *online Youtube* como complemento para su formación. También es importante notar el uso que hacen 3 de 6 de los participantes de distintos foros de ayuda. Esta información puede verse en la figura 6.4.

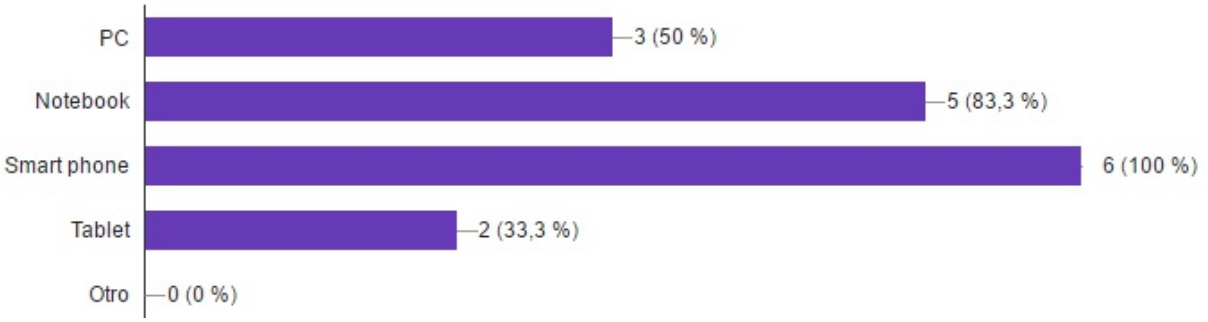

Figura 6.3: *Gráfico de las estadísticas sobre el uso de dispositivos móviles.*

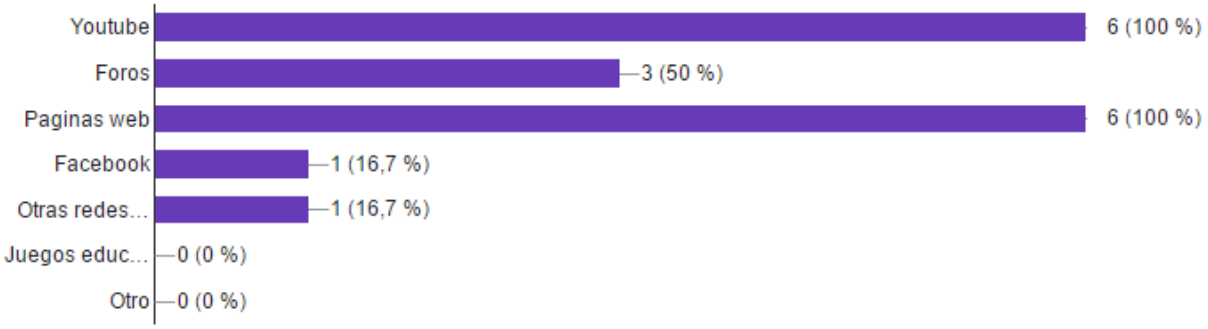

Figura 6.4: *Gráfico de las estadísticas sobre el uso de recursos digitales.*

- **Motivación:** en este apartado se les consulto a los participantes qué tan de acuerdo se encontraban con respecto a diferentes afirmaciones relacionadas a la motivación al trabajar con Desafiate y los juegos serios y su utilización en escenarios educativos, en

particular para su autoevaluación. La escala utilizada era: Muy de acuerdo, De acuerdo, Indeciso, En desacuerdo, Muy en desacuerdo. Los datos resultaron ser favorables al objetivo de esta tesina siendo que en todos los tópicos preguntados los participantes estaban en su mayoría de acuerdo. A continuación se listan los diferentes tópicos consultados y los resultados de las respuestas obtenidas.

- **El uso de juegos me resulta un complemento interesante para mi aprendizaje.** En este punto los resultados marcaron que 5 de 6 de los participantes estaban **de acuerdo**, siendo que la única respuesta diferente el participante estaba **muy de acuerdo**.
- **El uso de juegos para autoevaluarme me resulta motivante.** En este punto se obtuvieron datos dispares. 3 de 6 participantes observaron que estaba **muy de acuerdo** con este punto, mientras que 2 de los participantes estaba **indeciso** en cuanto a su postura.
- **El uso de juegos en mi proceso de aprendizaje me parece que puede mejorar mi actitud a una más positiva.** De este punto se obtuvieron resultados más dispares ya que la mitad de los participantes estaba **muy de acuerdo** con esto, y la mitad de los participantes estaba **indeciso** con respecto a su posición en este inciso.
- **El uso de Desafiate me sirvió para repasar los conceptos de Expresión de Problemas y Algoritmos (EPA).** Se observa en este punto que el 83 % de los participantes, es decir 5, estaban por lo menos **de acuerdo** en este punto, y además que 4 de ellos estaban **muy de acuerdo** en ese punto.
- **El uso de Desafiate me sirvió para medir mi nivel de conocimiento respecto a los conceptos de EPA.** Todos los participantes se encontraban por lo menos **de acuerdo** en este punto, y 2 personas estaban **muy de acuerdo** al respecto.
- **El uso de Desafiate resultó estimulante para continuar aprendiendo sobre la temática de EPA.** Aquí la mitad de los participantes estaba **de acuerdo** con la afirmación, un participante estaba **muy de acuerdo**, uno se encontraba **indeciso**, mientras que otro marco que estaba **en desacuerdo**.
- **Desafiate puede utilizarse para autoevaluarme en otros tópicos en caso de incluir nuevas islas con otros interrogantes.** Todos los participantes estuvieron **de acuerdo**o **muy de acuerdo** con esta afirmación. 4 de 6 (66 %) indicaron estar **muy de acuerdo**.
- **El uso de Desafiate me resultó menos estresante que una autoevaluación normal.** De este punto se obtuvieron resultados iguales al segundo punto analizado. 3 de 6 participantes estuvo **de acuerdo** con la afirmación, y 2 de 6 **indeciso** en cuanto a su postura.

En general, se visualizan resultados positivos, que al ser ampliados en diálogo con los alumnos durante las sesiones, muestran un fuerte interés en la propuesta, remarcan

por ejemplo, que les hubiera gustado contar con una aventura similar en relación a los contenidos del módulo de COC. Al mismo tiempo, proponen mejoras respecto de la jugabilidad y la interfaz que serán desarrollados luego de exponer el resto de los resultados.

- **Opinión sobre Desafiate:** en este apartado se les consultó a los participantes su opinión sobre diferentes cuestiones de Desafiate, como por ejemplo, qué aspectos les resultaron atractivos, que cambiarían y si volverían a utilizar el juego, y diferentes preguntas para que puedan explayarse y dar su opinión.

  Cuando se les consulto a los participantes sobre cuáles fueron los elementos que más y menos les gustaron, todos estuvieron de acuerdo en que la idea de usar un juego para autoevaluarse resulta interesante. El 83 % de los participantes también opinaron que el uso de los dispositivos móviles es otro punto a favor que posee el juego. Por el lado contrario, la mitad de los participantes opinó que la dinámica de juego es algo a ser mejorado a partir de la inclusión de nuevas historias en las islas. Estos datos pueden verse en las figuras 6.5 y 6.6.

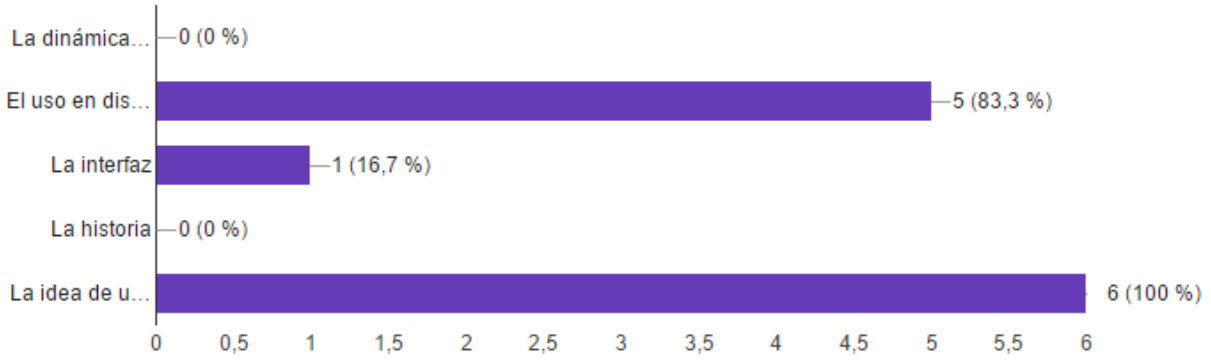

Figura 6.5: *Gráfico de las estadísticas sobre lo que más gusto de Desafiate.*

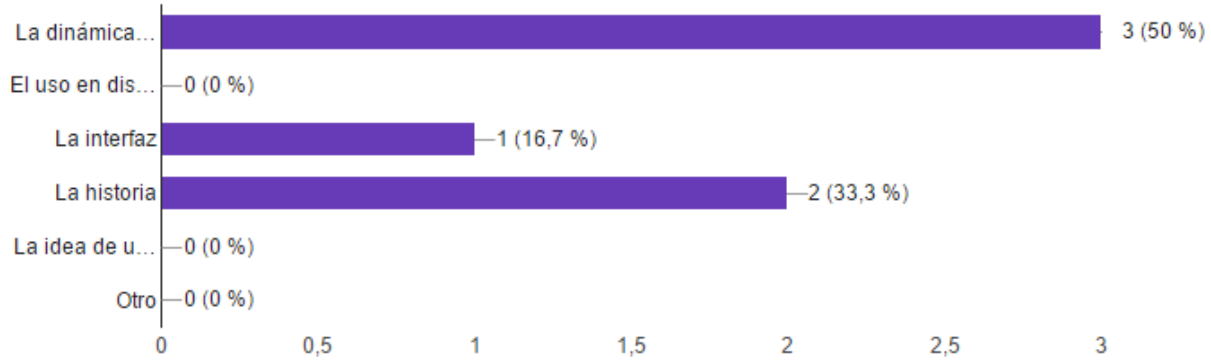

Figura 6.6: *Gráfico de las estadísticas sobre lo que menos gusto de Desafiate.*

Sobre las preguntas de si volverían a usar Desafiate o algún otro juego para complementar el proceso de aprendizaje, y si recomendarían a sus amigos el uso de estos, todos los participantes estuvieron de acuerdo en que volverían a usar Desafiate y que les gustaría utilizar juegos serios en la educación universitaria, y 5 de 6 participantes recomendaría Desafiate o cualquier otro juego serio. Hubo un caso particular, el cual no recomendaría el uso de juegos serios o de Desafiate a sus amigos, pero si está de acuerdo en volver a utilizar Desafiate, o cualquier otro juego serio para su uso personal. Una posible conclusión es que el alumno está de acuerdo con el uso de juegos para sí mismo, pero no está seguro de si es algo que utilizarían otros estudiantes.

Cuando se les consulto a los participantes sobre qué cuestiones les gustaría ver en Desafiate en sus próximas versiones, 5 de 6 de ellos aportó que la idea de poder desafiar a un compañero sería interesante en el futuro. A su vez, otras cuestiones en las que 4 de 6 participantes estuvieron de acuerdo en incorporar son: el agregado de sonido y música al juego, el recibir un mejor *feedback* a la hora de contestar las preguntas, y poder puntuar la dificultad de las preguntas.

Cuando se les preguntó a los alumnos sobre qué cambiarían de Desafiate, algunos estuvieron de acuerdo en que le agregarían una mayor velocidad a la hora de responder las preguntas, y un participante se mostró muy interesado en mejorar la interfaz gráfica (el alumno que utilizó un dispositivo móvil de baja resolución), ya que se le dificulto el poder resolver los desafíos.

Esta experiencia sirvió para poder confirmar las cuestiones analizadas en el capítulo 2 en cuanto a la motivación acerca del uso de juegos serios en educación y en particular en lo referente a la autoevaluación. Esto marca que el desarrollo de Desafiate puede ser útil y beneficioso para los alumnos. Además las diferentes opiniones brindadas sirvieron para poder marcar el camino a seguir en el futuro. Se abordarán en profundidad estas cuestiones en el capítulo de Conclusiones y trabajos futuros.

6.4. Experiencia con docentes

Para esta experiencia, se invitó a diferentes docentes que conforman el plantel del módulo de EPA. Se eligió un horario y día donde se estimaba que podían concurrir la mayor cantidad de docentes. Esto se debe a que cada uno de los docentes tiene horarios muy distintos debido a sus diferentes responsabilidades, con lo cual la coordinación de horario se torna una labor muy difícil. Como la mayoría de los docentes suelen concurrir a la mañana, se eligió el horario de las 10:00 hs y un día en el que no se dictan clases prácticas del módulo.

En este caso se utilizó el mismo servidor que en la anterior experiencia, y todos los participantes hicieron uso de las 4 *tablets* que fueron preparadas para esta experiencia, la

cual duró aproximadamente 1 hora. Se presentaron 4 docentes a realizarla entre los que se encontraban 2 auxiliares, 1 profesor de teoría y 1 coordinador. La experiencia empezó de una forma similar a la realizada con los alumnos. Se les comentó a los participantes el marco en el que fue creado Desafiate, y cuáles son los objetivos que se persiguen y que el juego a utilizar se trata de un prototipo en evolución y que interesaba su opinión para continuar desarrollando el juego, invitándolos a que sean lo más críticos y honestos posibles ya que se desea que ésta sea una herramienta de apoyo más para el proceso de enseñanza. En este punto los participantes empezaron a hacer uso del juego.

Cuando los participantes terminaron la aventura disponible en Desafiate, realizaron un cuestionario, el cual se encuentra en el Anexo 3, para poder obtener sus opiniones acorde a los objetivos planteados para las experiencias con anterioridad. El cuestionario cuenta además con preguntas para poder conocer algunos aspectos del perfil de los docentes participantes, así como también, datos sobre qué materias dictan. Al mismo tiempo se llevó un registro a partir de la técnica de observación participante. Se consideraron los pensamientos en voz alta, y consultas que fueron realizando los docentes.

De la experiencia se obtuvieron los siguientes resultados:

- **Perfil de los participantes:** los datos arrojaron que la mitad de los participantes son docentes en materias de primer año de la facultad, mientras que la otra mitad participa en materias de segundo año.
- **Juegos serios:** el formato de este apartado es el mismo que se utilizó para la parte de motivación en el cuestionario de la experiencia de alumnos. Se le presentaron diferentes afirmaciones a los participantes y ellos contestaron que tan de acuerdo estaban con estas afirmaciones. La escala fue la misma utilizada para el cuestionado de los alumnos: Muy de acuerdo, De acuerdo, Indeciso, En desacuerdo, Muy en desacuerdo. A continuación se listan las cuatro afirmaciones realizadas y los resultados obtenidos.
    - **El uso de juegos serios me resulta un complemento interesante para el proceso de enseñar.** En este punto todos los participantes coincidieron al estar **muy de acuerdo** con la afirmación.
    - **El uso de juegos serios para que los alumnos se autoevalúen me resulta una idea innovadora.** En este punto se obtuvieron los mismos resultados que el punto anterior ya que todos los participantes volvieron a estar **muy de acuerdo**.
    - **El uso de juegos serios para que los alumnos se autoevalúen me resulta una idea innovadora.** Este es otro punto en donde los participantes volvieron a estar **muy de acuerdo**.
    - **El trabajo con juegos serios sobre dispositivos móviles es factible de llevar a cabo con mis alumnos.** Este punto es el único que marca una mínima diferencia

con respecto a los anteriores. En este caso 2 de 4 participantes estuvieron **muy de acuerdo** con respecto a la afirmación, mientras que el resto de los participantes estuvo **de acuerdo**. Es interesante notar que los participantes que estuvieron **de acuerdo** con la afirmación, fueron los auxiliares que dictan las clases prácticas y que son los que suelen estar más en contacto con los alumnos. Será interesante como trabajo futuro continuar indagando esta situación.

Esta primera indagación arrojó resultados positivos en cuanto al uso de juegos serios estrategia para propuestas de enseñanza en las materias de los docentes. Lo que demarca una actitud positiva hacia la experiencia, y una apertura por parte de los docentes para poder incluir Desafiate más adelante en sus prácticas docentes.

- **Opinión sobre Desafiate:** en este apartado se les pregunto a los participantes sobre las cuestiones que más les gustaron de Desafiate, y cuáles fueron las que menos les gustaron. En el primer punto, los resultados estuvieron en concordancia con los arrojados por la experiencia con alumnos, ya que todos los participantes afirmaron que la idea de utilizar un juego para la autoevaluación de los alumnos fue lo que más le gustó. 2 de los 4 participantes contestaron que les gustó la dinámica de juego, y solo 1 participante afirmó que le gustó el uso en dispositivos móviles. En el otro punto, 3 de 4 participantes contestaron que lo que menos les gustó fue el tiempo que hay entre pregunta y pregunta, esta opción no estaba entre las respuestas posibles y fue agregada por los propios docentes. Por lo que será tenida en cuenta como un aspecto de mejora. Al mismo tiempo, un docente al igual que lo hizo un alumno, sugirió tener más de un desafío por isla, es decir contar con más de una pregunta por isla.

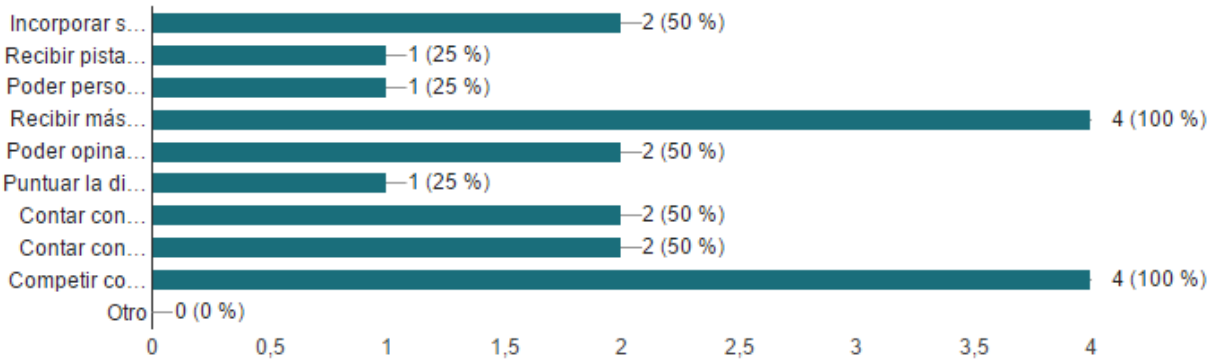

Figura 6.7: *Gráfico de las estadísticas sobre lo que espera cada participante docente en Desafiate en el futuro.*

Cuando se les preguntó a los participantes si usarían Desafiate en las materias que dictan, todos los participantes afirmaron que lo harían, además de recomendar el juego a otros docentes. En el punto sobre los elementos que les gustarían ver a futuro, los resultados fueron dispares como pueden verse en la figura 6.7, pero resaltó el deseo

de todos los participantes de incorporar el desafío entre jugadores y la posibilidad de mejorar el *feedback* que se obtiene de las respuestas.

Esta experiencia sirvió para evaluar la posibilidad de incorporar juegos serios, y en particular Desafiate, en materias de la Facultad de Informática de la Universidad Nacional de La Plata. Esto marca que los docentes mostraron actitudes positivas ante estas posibilidades. Al mismo tiempo se consideraron cuestiones de usabilidad y surgieron aspectos de mejora como los de mejorar el timing de la animación del barco entre isla e isla, la mejora del slider en las preguntas, entre otros. Los resultados sirven para poder evaluar cuáles son las cuestiones que se deberían incorporar o mejorar en el futuro en Desafiate desde el punto de vista de un docente.

6.5. Conclusiones del capítulo

En este capítulo se detallaron las experiencias realizadas con Desafiate, tanto con alumnos como con docentes. Ambas experiencias arrojaron resultados positivos en cuanto a la posibilidad de la incorporación de juegos serios en educación, estando estos datos en concordancia con lo analizado en el capítulo 2.

De la experiencia con alumnos se observó a partir de los resultados de la encuesta, así como la observación realizada por el tesista en el momento de la experiencia, que los participantes se motivaron con la posibilidad de poder usar un juego serio en su proceso de aprendizaje. Por el lado de los docentes, estos se mostraron abiertos a la posibilidad de poder incorporar los juegos serios como una estrategia que complemente su propuesta de enseñanza.

Con respecto a Desafiate, la opinión general en ambas experiencias fue buena. Ambas experiencias muestran que uno de los factores que más gustó fue la posibilidad de usar un juego como estrategia de autoevaluación. También se observó que la mayoría de los alumnos opinaba que el uso de dispositivos móviles es un aspecto llamativo, mientras que del lado del docente este aspecto no fue considerado entre los más seleccionados como desatacados de la experiencia. Sin embargo, estas opiniones también arrojaron resultados vinculadas a la usabilidad y dinámica del juego, que se irán analizando en el futuro, como una mejora en la interfaz, y la velocidad con que transcurre el juego. Varias de las opiniones obtenidas en ambas experiencias servirán como guía de trabajo para la evolución de Desafiate.

Para finalizar, la mayoría de los participantes de ambas pruebas afirmó que recomendaría Desafiate a sus pares como parte del proceso de enseñanza y aprendizaje.

En el próximo capítulo se expondrán las conclusiones del trabajo y las líneas de trabajo futuro.

CAPÍTULO 7

CONCLUSIONES Y TRABAJOS FUTUROS

7.1. Introducción

En este capítulo se resumen los aportes del trabajo y se presentan las conclusiones y líneas de trabajo futuro.

7.2. Aportes y conclusiones del trabajo

En esta tesina de grado se ha realizado una revisión bibliográfica sobre juegos serios orientados a escenarios educativos, en particular se han buscado trabajos relacionados con su uso para procesos de autoevaluación. Se encontraron pocos antecedentes al respecto, pero que mostraron resultados interesantes tomados como base para el trabajo aquí realizado.

Al mismo tiempo, se realizó una recopilación de motores de juego y se analizaron respecto de una serie de aspectos deseables en función de las necesidades del juego a desarrollar para la tesina. Se eligió un motor en base a este análisis para llevar adelante el desarrollo.

Finalmente, se realizó el diseño y desarrollo de Desafiate un juego serio con el fin de acompañar a los alumnos en procesos de autoevaluación, de una manera diferente. El juego ha sido diseñado para que se pueda integrar con un EVEA, se diseñaron 3 capas para lograr flexibilidad en la implementación: una orientada al juego en sí misma, otra para la comunicación con el EVEA y otra que representa al EVEA en cuestión. En este caso se trabajó con IDEAS, un entorno virtual de enseñanza y aprendizaje que es una evolución de WebUNLP.

Se realizaron experiencias con docentes y alumnos que permitieron analizar la opinión sobre los juegos serios en escenarios educativos, específicamente para la autoevaluación, los aspectos positivos de Desafiate y aquellos por mejorar, en relación a su usabilidad, dinámica del juego, e historias incluidas.

Todo este proceso permite arribar a las siguientes conclusiones:

1. Tanto los docentes como los alumnos participantes de las experiencias muestran una actitud positiva frente a la incorporación de juegos serios en procesos de enseñar y aprender. Consideran que incluirlos como estrategia de evaluación es una idea innovadora y que les ha gustado.

2. Desafiate ha tenido buena aceptación, rescatándose por parte de los alumnos el uso de dispositivos móviles como un aspecto interesante, al igual que la idea de utilizar un juego para su autoevaluación. Los docentes han valorado también este último aspecto. En general la tendencia en ambos casos, ha sido la de indicar que utilizarían Desafiate y lo recomendarían a otros compañeros o colegas, lo que motiva a continuar evolucionando este juego. En general los resultados obtenidos vislumbran coincidencia con los relevados en la bibliografía.

3. Se han recomendado varios aspectos de mejora y propuestas para la evolución del juego que han resultado muy enriquecedoras. Estos aspectos incluyen temas de usabilidad tales como la barra para hacer *scroll*, el *timing* de algunas animaciones, el tamaño de algunos textos, la visualización en celulares de baja resolución, entre otros. Al mismo tiempo, se han propuesto ideas para la dinámica del juego, las historias, y los tipos de islas que corresponden a mejorar al jugabilidad y la motivación durante la experiencia.

En lo personal el trabajo aquí realizado me ha permitido ganar experiencia en el desarrollo de una aplicación para móviles, profundizar en el conocimiento sobre motores de juegos y aprender sobre estrategias de integración entre diferentes aplicaciones. Ha sido una experiencia provechosa e interesante.

7.3. Líneas de Trabajo Futuro

Como se deja entrever en la sección previa, aún queda mucho trabajo por realizar, algunas de las líneas que se abren a partir de este trabajo son:

- Evolucionar Desafiate considerando aspectos de usabilidad y jugabilidad recogidos de las experiencias ya realizadas. Además, se continuarán realizando pruebas con alumnos y docentes para que resulte un diseño centrado en las opiniones de los

usuarios finales. Se incluirán también docentes y alumnos de otros ámbitos ya que por ejemplo, se ha visto el alto perfil tecnológico de los alumnos que participaron, por lo que sería interesante ver qué ocurre en otras disciplinas.

- Incluir en IDEAS posibilidades para mostrar la cantidad de monedas que se ha ganado un alumno desarrollando aventuras en Desafiate y puedan competir con compañeros.
- Avanzar en la integración de Desafiate con otros EVEAs.
- Profundizar la investigación sobre juegos serios para procesos de autoevaluación. Si bien en esta primera revisión bibliográfica se han encontrado pocos trabajos al respecto, se cree que este tema avanzará en los próximos años y se deberán considerar los resultados y antecedentes que vayan surgiendo.

Finalmente, se espera realizar publicaciones en eventos científicos y tecnológicos para difundir los trabajos aquí realizados.

REFERENCIAS

Abt, C. (1970). *Serious games.* Viking Press.

Bethke, E. (2003). *Game development and production.* Wordware Pub.

Bezanilla, M. J., Arranz, S., Rayón, A., Rubio, I., Menchaca, I., Guenaga, M., y Aguilar, E. (2014). Propuesta de evaluación de competencias genéricas mediante un juego serio. *NEW APPROACHES IN EDUCATIONAL RESEARCH*, *3*(1), 44-54.

Boud, D., y Falchikov, N. (2006, Agosto). Aligning assessment with long-term learning. special issue: Learning-oriented assessment: Principles and practice. *Assessment & Evaluation in Higher Education*, *31*(4), 399-413.

Boyle, E., Connoly, T. M., y Hainey, T. (2011). The role of psychology in understanding the impact of computer games. *Entertainment Computing*, *2*, 69-74.

Cian, T., Sanz, C. V., y Zangara, A. (2009). *La evaluación de los aprendizajes en una propuesta mediada.* Universidad Nacional de La Plata.

Crawford, C. (2003). *On game design.* New Riders.

Csikszentmihalyi, M. (1990). *The psychology of optimal experience.* Harper and Row.

De Benito Crosetti, B., y Salinas Ibañez, J. M. (2016). La investigación basada en diseño en tecnología educativa. *Revista Interuniversitaria de Investigación en Tecnología Educativa.*

eMarketer. (2016). *Smartphone uptake is on the rise in argentina.* Descargado 2016-08-16, de `https://www.emarketer.com/Article/Smartphone-Uptake-on-Rise-Argentina/1014300`

ENACOM. (2016). *Penetración de celulares en argentina en el 2016.* Descargado 2017-02-20, de `http://www.enacom.gob.ar/multimedia/noticias/archivos/201612/archivo\_20161212113553\_7094.pdf`

Fernández-Baena, A., Susín, A., y Lligadas, X. (2012, Sept). Biomechanical validation of upper-body and lower-body joint movements of kinect motion capture data for rehabilitation treatments. En *2012 fourth international conference on intelligent networking and collaborative systems* (p. 656-661).

Fernández-Manchón García, A. (2012). Aportaciones del juego del garabato de winnicott en la psicoterapia de adolescentes. *Cuadernos de Psiquiatría y Psicoterapia del Niño y del Adolescente, 54*.

Ferrel, G. (2012). A view of the assessment and feedback landscape: baseline analysis of policy and practice from the jisc assessment & feedback programme. *JISC*.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*. UNIVERSITY OF CALIFORNIA, IRVINE.

Garcia Aretio, L., Ruiz Corbella, M., y Domínguez Figaredo, D. (2007). *De la educación a distancia a la educación virtual*. Editorial Ariel.

Giannakos, M. (2013). Enjoy and learn with educational games: Examining factors affecting learning performance. *Computers & Education, 68*, 429 - 439.

Gross Salvat, B. (2009). Certezas e interogantes acerca del uso de los videojuegos para el aprendizaje. *Comunicación: revista Internacional de Comunicación Audiovisual, Publicidad y Estudios Culturales*, 251-264.

Herrero, D., Del Castillo, H., Monjelat, N., García-Varela, A. B., Checa, M., y Gómez, P. (2014, Enero). La teoría de la evolución y la selección natural: aprender a través del juego y la reflexión. En *New approaches in educational research* (Vol. 3, p. 28-35).

IDC. (2016). *Smartphone so in q3 2016*. Descargado 2017-02-20, de `http://www.idc.com/promo/smartphone-market-share/os`

Klopfer, E. (2009). *Augmented learning research and design for mobile educational games*. MIT press.

Lewis, M., y Jacobson, J. (2002). Game engines in scientific research. *Communications of the Association for Computing Machinery, 45*(1), 27-48.

Marcano Lárez, B. E. (2014). *Factores emocionales en el diseño y la ejecución de videojuegos y su valor formativo en la sociedad digital.: El caso de los videojuegos bélicos* (1.ª ed.). Ediciones Universidad de Salamanca.

Michael, D., y Chen, S. (2005). *Serious games: Games that educate, train, and inform*. Thomson Course Technology.

Mora Carreño, A., Riera, D., Gonzales, C. S., y Arnedo-Moreno, J. (2015). A literature review of gamification design frameworks. En *7th international conference on games and virtual worlds for serious applications (vs-games)*.

Navarro, A., y da Silva, A. (2016). A metamodel-based definition of a conversion mechanism between {SOAP} and {RESTful} web services. *Computer Standards & Interfaces, 48*, 49 - 70.

Nilsson, E., y Jakobsson, A. (2010). Simulated sustainable societies: Students' reflections on creating future cities in computer games. En *Journal of science education and technology* (p. 33-50).

Nomdedeu, L. J., Díaz, F. J., y Fava, L. (2015). *Raíces: un juego serio social para revalorizar las culturas originarias*. Universidad Nacional de La Plata.

Pastor, I., Hayes, H. A., y Bamberg, S. J. M. (2012, Aug). A feasibility study of an upper limb rehabilitation system using kinect and computer games. En *2012 annual international conference of the ieee engineering in medicine and biology society* (p. 1286-1289).

Pereira, A. M. M. (2014). El proceso productivo del videojuego: fases de producción. *Historia y Comunicación Social*, *19*(0).

Przybylski, A. K., Ryan, R. M., y Rigby, C. S. (2009). The motivating role of violence in video games. *Personality and Social Psychology Bulletin*, *35*(2), 243-259.

Sanchez, J., Mendoza, C., y Salinas, A. (2009, Febrero). Mobile serious games for collaborative problem solving. *Studies in health technology and informatics*, 193-197.

Sanz, C. V. (2015, Junio). Los objetos de aprendizaje, un debate abierto y necesario. *Bit & Byte*, *1*(1), 33-35.

Sanz-Troyano, E., Torrente, J., Moreno-Ger, P., y Fernández-Manjón, B. (2010, Septiembre). Introduciendo criterios de accesibilidad en una herramienta de juegos educativos: <e-adventure>. En *In proceedings of xi simposio nacional de tecnologías de la información y las comunicaciones en la educación (sintice)*.

Smith, R. (2008). *Rogue leaders: The story of lucasarts*. Chronicle Books.

Sung, H.-Y., y Hwang, G.-J. (2013). A collaborative game-based learning approach to improving students' learning performance in science courses. En *Computers & education* (Vol. 63, p. 43-51).

ÍNDICE DE FIGURAS

Anexo 1

Autoevaluación creada para las experiencias.

Pregunta 1:

Un algoritmo es similar a una receta que debe cumplir con un conjunto de propiedades. A continuación listamos algunas propiedades. Selecciona aquellas que corresponden con el concepto de algoritmo.

1 - Una instrucción del algoritmo puede tener mas de un significado.

2* - Las instrucciones de un algoritmo tienen que seguir un determinado orden. (50%)

3 - Un algoritmo escrito siempre en un lenguaje de programación.

4 - Un algoritmo puede ejecutarse indefinidamente.

5* - Un algoritmo requiere de alguien que interprete cada instrucción para poder ejecutarse. (50%)

Pregunta 2:

Se quieren sumar la cantidad de flores que hay en la calle 1. ¿Qué estructura de control utilizarías para hacer el recorrido de la calle?

1* - Repetir. (100%)

2 - Mientras.

3 - Si.

Pregunta 3:

Se desea recorrer una avenida de la ciudad hasta encontrar una esquina con exactamente 5 flores. ¿Qué estructura de control utilizarías para contar las flores en una esquina?

1 - Repetir.

2* - Mientras. (100%)

3 - Si.

Pregunta 4:

Un ambiente de programación es la habitación donde se desarrolla el programa.

1 - Verdadero.

2* - Falso. (100%)

Pregunta 5:

Selecciona aquella que consideres una ventaja de diseñar un algoritmo con módulos, respecto de uno totalmente secuencial.

1* - Evita escribir el mismo código repetidas veces. (100%)

2 - Mejora la eficiencia de la solución.

3 - Permite escribir algoritmos más seguros.

Pregunta 6:

Selecciona la opción que indica lo que hace el siguiente código.

```
Proceso cuadrado (E lado: numero)
Comenzar
  lado := lado + 1
  repetir 4
    repetir lado
      mover
    derecha
fin

variables
  largo : numero
comenzar
  largo := 1
  repetir 4
    cuadrado(largo)
    pos(posAv+5,1)
fin
```

1 - Realiza 4 cuadrados de 2, 3, 4 y 5 cuadras de largo.

2* - Realiza 4 cuadrados de 2 cuadras de largo.

3 - Realiza un cuadrado de 2 cuadras de largo.

Pregunta 7:

Los parámetros de entrada deben recibir si o si el valor de una variable

1 - Verdadero.

2* - Falso. (100%)

Pregunta 8:

Al declarar es una variable es necesario especificar el tipo de dato al que pertenece. Esto permitirá saber:

1 - El valor inicial que tendrá la variable

2 - Definir si puede ser modificada por un módulo

3* - Definir cuales son las operaciones que se pueden aplicar sobre la variable. (100%)

Pregunta 9:

Selecciona la opción que indica lo que hace el siguiente código.

```
variables
  condicion :boolean
comenzar
  condicion := HayFlorEnLaEsquina;
  mientras(condicion)
    tomarFlor
fin
```

1 - En cada iteración del mientras preguntará si hay flor en la esquina

2 - Da error ya que no se puede usar un booleano en el mientras

3* - Si hay flor en la esquina, el mientras se ejecutará infinitamente(100%)

Pregunta 10:

Cualquier cambio ocurrido en un módulo sobre un parámetro de entrada se verá reflejado en el programa principal.

1 - Verdadero.

2* - Falso. (100%)

Anexo 2

Cuestionario de la experiencia con alumnos.

Parte 1:

1) Nombre y apellido:
2) Carrera en la que te inscribiste:
3) Edad: 4) Sexo:
5) Nacionalidad:
6) Colegio de procedencia:

☐Gestión pública ☐Gestión privada ☐Otro

Parte 2: Sobre la utilización de recursos digitales.

1. ¿Disponés de algún dispositivo informático? ☐ Sí ☐ No

2. En caso afirmativo indicá con cuál o cuáles de los siguientes dispositivos contás:

☐ PC ☐ Smart phone
☐ Notebook ☐ Tablet
☐ Otros Indicar cuál: ……

3. ¿Tenés conexión a internet en tu casa? ☐ Sí ☐ No

4. ¿Usás recursos digitales como apoyo para el estudio? ☐ Sí ☐ No

5. En caso afirmativo, indicá cual o cuales de los siguientes recursos utilizás cuando estudias:

❏ Youtube, ❏ Foros
❏ Páginas web, ❏ Facebook,
❏ Juegos educativos ❏ Otras redes sociales
❏ Otro: _____

Parte 3: Motivación.

INSTRUCCIONES

Te solicitamos que contestes lo más sinceramente posible a los datos que se piden. Leé atentamente las diversas cuestiones y selecciona la opción de respuesta que te resulte más próxima o que mejor se ajuste a tu situación. Tené en cuenta que no hay respuestas correctas ni incorrectas.

Señala con una cruz el recuadro correspondiente a la respuesta elegida considerando tu grado de acuerdo con la frase:

MA: Muy de acuerdo - A: De acuerdo - I: Indeciso- D: En desacuerdo - MD: Muy en desacuerdo

	MA	A	I	D	MD
El uso de juegos me resulta un complemento interesante para mi aprendizaje.					
El uso de juegos para autoevaluarme me resulta motivante.					
El uso de juegos en mi proceso de aprendizaje me parece que puede mejorar mi actitud a una más positiva.					
El uso de Desafiate me sirvió para repasar los conceptos de Expresión de Problemas y Algoritmos (EPA).					
El uso de Desafiate me sirvió para medir mi nivel de conocimiento respecto a los conceptos de EPA					
El uso de Desafiate resultó estimulante para continuar aprendiendo sobre la temática de EPA					
Desafiate puede utilizarse para autoevaluarme en otros tópicos en caso de incluir nuevas islas con otros interrogantes					
El uso de Desafiate me resultó menos estresante que una autoevaluación normal					

Parte 4: Tu opinión sobre Desafiate.

1. ¿Qué fue lo que más te gustó de Desafiate? Marca la/s opción/es que consideres que refleja/n tu opinión:

   ❏ La dinámica del juego
   ❏ El uso en dispositivos móviles
   ❏ La idea de usar un juego para autoevaluarme
   ❏ La historia
   ❏ La interfaz
   ❏ Otro: _____

2. ¿Qué fue lo que menos te gustó de Desafiate? Marca la/s opción/es que consideres que refleja/n tu opinión:

   ❏ La dinámica del juego
   ❏ El uso en dispositivos móviles
   ❏ La idea de usar un juego para autoevaluarme
   ❏ La historia
   ❏ La interfaz
   ❏ Otro: _____

3. ¿Te gustaría usar Desafiate en diferentes materias como complemento para autoevaluarte y desafiarte? ☐ Sí ☐ No

4. ¿Recomendarías Desafiate a tus amigos? ☐ Sí ☐ No

5. ¿Recomendarías el uso de juegos como complemento en la educación universitaria en general? ☐ Sí ☐ No

6. ¿Recomendarías el uso de juegos como estrategia de autoevaluación a otros compañeros?

☐ Sí ☐ No

7. ¿Cuáles de estos agregados te gustaría ver en Desafiate?

- ❑ Incorporar sonidos a la dinámica del juego
- ❑ Recibir pistas o ayudas en las preguntas
- ❑ Poder personalizar mi avatar
- ❑ Recibir más feedback de las respuestas
- ❑ Poder opinar sobre las preguntas
- ❑ Puntuar la dificultad de las preguntas
- ❑ Contar con más tipos de islas
- ❑ Contar con más tipos de historias
- ❑ Competir con otros participantes
- ❑ Otro: _____

8. ¿Qué tipo de historias te gustaría ver en Desafiate?

9. ¿Qué tipo de islas te gustaría ver en las islas en Desafiate?

10. ¿Qué le cambiarías a Desafiate?

Gracias por participar de esta sesión. Tu opinión es muy importante.

Dejanos cualquier otro comentario que quieras hacer y no fue incluido en las preguntas anteriores.

Anexo 3

Cuestionario de la experiencia con docentes.

Parte 1:

1) Nombre y apellido:
2) Materias en las que participa como docente:

Parte 2: Tu opinión sobre el uso de juegos serios en el proceso de enseñanza.

INSTRUCCIONES

Te solicitamos que contestes lo más sinceramente posible a los datos que se piden. Leé atentamente las diversas cuestiones y selecciona la opción de respuesta que te resulte más próxima o que mejor se ajuste a tu situación. Tené en cuenta que no hay respuestas correctas ni incorrectas.

Señala con una cruz el recuadro correspondiente a la respuesta elegida considerando tu grado de acuerdo con la frase:

MA: Muy de acuerdo - A: De acuerdo - I: Indeciso- D: En desacuerdo - MD: Muy en desacuerdo

	MA	A	I	D	MD
El uso de juegos serios me resulta un complemento interesante para el proceso de enseñar.					
El uso de juegos serios para que los alumnos se autoevalúen me resulta una idea innovadora.					
Utilizaría juegos serios como parte de mi estrategia de enseñanza.					
El trabajo con juegos serios sobre dispositivos móviles es factible de llevar a cabo con mis alumnos.					

Parte 4: Tu opinión sobre Desafiate.

1. ¿Qué fue lo que más te gustó de Desafiate? Marca la/s opción/es que consideres que refleja/n tu opinión:

   ❑ La dinámica del juego
   ❑ El uso en dispositivos móviles
   ❑ La idea de usar un juego para la autoevaluación de los alumnos
   ❑ La historia
   ❑ La interfaz
   ❑ Otro: _____

2. ¿Qué fue lo que menos te gustó de Desafiate? Marca la/s opción/es que consideres que refleja/n tu opinión:

   ❑ La dinámica del juego
   ❑ El uso en dispositivos móviles
   ❑ La idea de usar un juego para la autoevaluación de los alumnos
   ❑ La historia
   ❑ La interfaz
   ❑ Otro: _____

3. ¿Te gustaría incorporar Desafiate en algunas de las materias en las que participas como docente? ☐ Sí ☐ No

   En caso de contestar sí, ¿en cuáles? _____

   En caso de contestar no, justifica tu respuesta

   _____

4. ¿Recomendarías Desafiate a otros docentes? ☐ Sí ☐ No

5. ¿Cuáles de estos agregados te gustaría ver en Desafiate?

   ❑ Incorporar sonidos a la dinámica del juego

   ❑ Recibir pistas o ayudas en las preguntas

   ❑ Poder personalizar mi avatar

   ❑ Recibir más feedback de las respuestas

   ❑ Poder opinar sobre las preguntas

   ❑ Puntuar la dificultad de las preguntas

   ❑ Contar con más tipos de islas

   ❑ Contar con más tipos de historias

   ❑ Competir con otros participantes

   ❑ Otro: _____

6. ¿Qué tipo de historias te gustaría ver en Desafiate?

7. ¿Qué tipo de islas te gustaría ver en las islas en Desafiate?

8. ¿Qué le cambiarías a Desafiate?

Gracias por participar de esta sesión. Tu opinión es muy importante.

Dejanos cualquier otro comentario que quieras hacer y no fue incluido en las preguntas anteriores.