Universitat Oberta de Catalunya
uoc.edu

Hipatia y los guardianes aritméticos

Jonatan Fúnez González
Grado en Ingeniería Informática
75.640 - Videojuego

Guillermo NWDD García Romero
Joan Arnedo Moreno

06/2021

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Hipatia y los guardianes aritméticos*
Nombre del autor:	*Jonatan Fúnez González*
Nombre del consultor/a:	*Guillermo NWDD García Romero*
Nombre del PRA:	*Joan Arnedo Moreno*
Fecha de entrega (mm/aaaa):	06/2021
Titulación:	*Grado de Ingeniería Informática*
Área del Trabajo Final:	*75.640 - Videojuegos*
Idioma del trabajo:	Castellano
Palabras clave	*RPG, Turn-Based, 2D*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

La principal finalidad de este proyecto es el desarrollo de un videojuego para dispositivos táctiles con sistema operativo Android. Adicionalmente, se trata de una buena oportunidad de poner en práctica todos los conocimientos adquiridos a lo largo de los estudios de esta titulación.

Para la realización de este proyecto se ha empleado una metodología iterativa, basada en el desarrollo de un primer prototipo para evaluar la jugabilidad. De esta forma, a través de las diferentes iteraciones, se ha ido mejorando y ampliando esta primera versión inicial hasta alcanzar el resultado deseado.

El producto final ha sido satisfactorio, ya que se han podido plasmar las ideas planteadas al inicio del desarrollo, así como el cumplimiento con todos los objetivos planteados a lo largo del proyecto. Para ello, ha sido necesaria una correcta planificación temporal, tanto a nivel de requisitos como de objetivos a alcanzar.

La conclusión alcanzada durante este proyecto es que el desarrollo de un juego no es una tarea trivial, ya que requiere de una correcta planificación. Además, es necesario profundizar en distintos tipos de tareas y realizar varias funciones para lograr un buen acabado final, tanto a nivel técnico como artístico.

Abstract (in English, 250 words or less):

The main purpose of this project is a videogame development for touch screen devices with Android operative system. In addition to this, it is a good opportunity to practice the recently acquired knowledge throughout time of this degree.

For the implementation of this project has use an iterative methodology, based on a first development prototype for evaluate the gameplay. This way, through the several iterations, is has been improving and increasing this initial version to reach the desired result.

The final product has been satisfactory, because the ideas raised at the beginning of the development have been captures, as well as the fulfillment of the objectives set during the project. A correct planning has been necessary for this, both at the level of requisites and objectives to be achieved.

The conclusion reached during this project is that a videogame development is not a trivial task, because it requires a correct planning. Furthermore, it has to go in depth in the different types of tasks and functions to achieve a good final result, both on a technical and artistic level.

Agradecimientos

En primer lugar, quiero dar las gracias a toda mi familia, por su incondicional apoyo durante todo este tiempo.

A mi amiga Maricarmen, por compartir tantos momentos durante estos estudios y por todas esas tardes en la sala de estudio.

A Encarni y Ramón, por su apoyo y por probar el juego hasta la saciedad ayudando a encontrar errores.

A mi tutor de proyecto, Guillermo NWDD García Romero, por resolver todas mis dudas, por sus recomendaciones a la hora de encarar este proyecto y sobre todo por compartir sus conocimientos de diseño de videojuegos.

Por último, a todas las personas con las que me he encontrado en los diferentes sitios y situaciones durante estos estudios. Gracias al intercambio de diferentes opiniones y puntos de vista, he podido ampliar mis horizontes.

A todos vosotros, de nuevo, GRACIAS.

Índice

Lista de figuras

Lista de tablas

1. Introducción

1.1. Contexto y justificación del Trabajo

Hoy en día, la industria del entretenimiento se encuentra más viva que nunca gracias a la expansión de los dispositivos portátiles y el acceso a Internet desde cualquier lugar. Esta expansión ha propiciado que toda persona que disponga de un terminal móvil pueda acceder a una amplia diversidad de contenidos en todo tipo de emplazamientos, desde el propio hogar o el transporte público, hasta prácticamente cualquier sitio en el que cuente con cobertura.

Uno de los más grandes representantes de estos contenidos se puede encontrar en el mundo de los videojuegos, que en 2019 facturó 14 billones de euros en el mercado europeo [1]. Este tipo de aplicaciones no sólo se encuentran en las pioneras consolas sino también en los imprescindibles móviles inteligentes. Actualmente, sólo hacen faltan un par de toques para poder estar disfrutando de un juego dónde y cuándo sea.

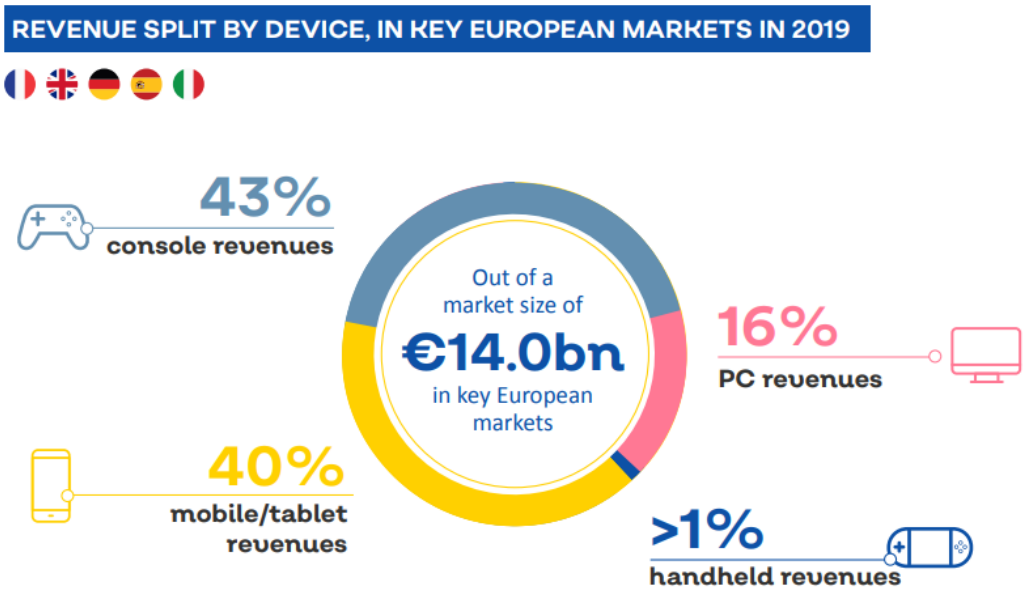


Figura 1.1 – Ingresos de videojuegos en mercados europeos en 2019 [2].

Este cambio de panorama ha influido directamente en la industria del videojuego y ha propiciado que un alto porcentaje de las ventas se produzcan en los dispositivos móviles. Según la Interactive Software Federation of Europe (ISFE), un 40% de los beneficios se focalizan en este tipo de dispositivos y hasta un 59% de los jugadores hacen uso de ellos como su plataforma principal [1].

Aunque el principal objetivo de los videojuegos es el entretenimiento, existen estudios que los relacionan con diversos desarrollos personales

en sus usuarios, como el aumento de empatía, la competitividad o la deportividad.

Así, en las competiciones profesionales de videojuegos (conocidas como *e-sports*) se fomenta el trabajo en equipo, el cumplimiento de las reglas establecidas y el compañerismo a través del respeto entre adversarios de diferentes equipos en la competición [3].

Además, se ha demostrado que los videojuegos pueden ayudar a desarrollar la empatía y la realización de una autocrítica sobre juicios morales cuando los jugadores actúan como personajes de dudosa moral dentro del juego [4].

Por todo ello, el desarrollo de un videojuego puede llegar a implicar muchos aspectos y objetivos a tener en cuenta, no sólo en el plano técnico o el posible rendimiento económico. De esta forma, existen muchos enfoques diferentes a la hora de abordar un desarrollo.

La principal finalidad del desarrollo de este proyecto es poder aportar un juego accesible a todo tipo de público y que pueda ser jugado por cualquier usuario que disponga de un dispositivo táctil con sistema operativo Android.

Aparte del propio objetivo principal de un videojuego, que es el hecho de entretener, se pretende aportar alguna pequeña novedad en el género del juego. Así, se han implementado algunas mecánicas de resolución de operaciones aritméticas, que fomenta la atención del jugador y pueden ayudar a los más jóvenes a interesarse por las matemáticas, obteniendo algunos incentivos dentro del juego.

1.2. Objetivos del Trabajo.

El principal objetivo de este Trabajo Fin de Grado es el desarrollo, diseño e implementación de un videojuego para dispositivos con sistema operativo Android. Se pretende realizar un juego de corte clásico adaptado a la actualidad, que pueda hacer uso de pantallas táctiles de una forma adecuada.

Además, existe la posibilidad de añadir algunas mecánicas de juego orientadas al uso de la pantalla táctil, sin necesidad de alterar los aspectos más puros de la jugabilidad y del género del juego. El hecho de añadir nuevas mecánicas siempre supone un reto, ya que es difícil identificar si éstas van a encajar bien con el estilo de juego original. No obstante, contar con una mecánica diferenciadora aportará al juego un punto diferenciador con su competencia y puede ser la forma de llamar la atención a su público objetivo.

Los objetivos del trabajo se han dividido en dos secciones diferenciadas: objetivos teóricos y objetivos prácticos. Se listan a continuación con una breve descripción de cada uno de ellos.

1.2.1. Objetivos teóricos.

Los objetivos teóricos que se quieren alcanzar realizando este desarrollo son los siguientes:

- Estudio de tecnologías y metodologías para poder llevar a cabo el desarrollo de un videojuego cuya plataforma de destino son los diferentes dispositivos móviles.

- Estudio de diseño de videojuegos, desde su conceptualización, pasando por el planteamiento de los diferentes niveles, hasta la elección de los posibles tipos de jugabilidad disponibles según el género elegido para desarrollar.

- Evaluación de la jugabilidad, de la percepción y la forma de interacción entre el juego y el jugador final, atendiendo a los motivos por los cuales un usuario querría jugar a este juego.

- Evaluación y elección de los diferentes motores de desarrollo, programas y herramientas disponibles que permitan desarrollar videojuegos.

- Investigación en términos de usabilidad y diseño de interfaces para su correcta y adecuada implementación en dispositivos cuya principal interfaz es la pantalla táctil.

- Estudio de las diferentes opciones y configuraciones disponibles que permitan la adaptación del juego a diferentes tipos de dispositivos. Todo ello incluye pantallas de diferentes tamaños, resoluciones y relaciones de aspecto.

1.2.2. Objetivos prácticos.

Los objetivos prácticos que se quieren alcanzar realizando este desarrollo son los siguientes:

- Diseño e implementación desde cero de un juego para dispositivos con pantalla táctil.

- Realización de un análisis de requisitos para definir las necesidades y funcionalidades a desarrollar para el sistema.

- Aplicación de la metodología elegida para llevar a cabo el desarrollo, a través de sus diferentes fases e iteraciones a realizar.

- Aplicaciones de criterios y técnicas de usabilidad, adaptabilidad y compatibilidad para interfaces táctiles, indicadas para los dispositivos objetivos, principalmente móviles inteligentes.

- Diseño de diferentes prototipos que se irán mejorando, iterando y ampliando hasta alcanzar el producto final.

- Pruebas y testeo del producto obtenido en diferentes dispositivos reales.

- Realización de pruebas con usuarios finales y mejora/ampliación de las funcionalidades del producto final a través de la retroalimentación aportada por ellos.

1.3. Enfoque y método seguido.

Existen multitud de metodologías y enfoques a la hora de desarrollar un determinado software que ayudan a evitar posibles problemas durante su creación. De esta manera, en el desarrollo de videojuegos, que es el caso que ocupa este proyecto, pueden llegar a producirse problemas que son bastante recurrentes en muchos proyectos que se llevan a cabo en esta industria.

Algunos de estos problemas son principalmente de alcance, producidos por no establecer correctamente los objetivos del proyecto, su complejidad y dimensión. Por otro lado, surgen problemas a la hora de cumplir fechas de entrega, lo que desemboca en otro conflicto derivado muy conocido en la industria, conocido *crunch time* [5].

Por todo lo anterior, elegir una correcta metodología de trabajo resulta vital para llevar a buen término un proyecto de desarrollo de software de videojuegos.

Actualmente, se suelen utilizar metodologías ágiles, ya que el desarrollo de videojuegos resulta de un proceso iterativo e incremental a lo largo del tiempo. Así, algunas de las metodologías que se pueden utilizar son *Scrum*, *Extreme Programming* (XP) o *Kanban* [6]. También se pueden emplear otros tipos que son combinación de varios enfoques, como *Game Unified Process*, que unifica modelos y técnicas de *Rational Unified Process* (RUP) y *XP* [7]. Este tipo de metodología se ideó para resolver problemas derivados del desarrollo en cascada.

Para el proyecto que nos ocupa, se ha optado por utilizar una combinación de dos metodologías: *Design Thinking* y *Scrum* [8]. Las razones principales es que estos modelos permiten alcanzar un consenso y encarar diferentes puntos de vista de los integrantes y participantes del proyecto. Esto resulta muy útil a la hora de aportar soluciones a los diferentes desafíos que irán surgiendo a lo largo del desarrollo.

Por otro lado, el enfoque de la metodología ágil *Scrum*, permitirá trabajar bajo un modelo iterativo e incremental a través de los diferentes *sprints*. También se podrá llevar un mejor seguimiento del estado de las tareas y del cumplimiento de los objetivos a alcanzar a través del tiempo de desarrollo del proyecto.

De esta manera, el primer paso es asumir el enfoque de *Design Thinking*. Se trata de un modelo cuyos usuarios presentan unas características de empatía (disponer de diferentes puntos de vista), pensamiento integrador (no basarse sólo en procesos analíticos, sino poder ver más allá del problema), optimismo ante cualquier problema, experimentación y colaboración (entre diferentes personas con experiencia en temas de diferentes campos y disciplinas) [9].

Por tanto, esta metodología se basa en disponer de diferentes perspectivas de un mismo problema y que, al discutir y enfrentar dichos puntos de vista, se pueda alcanzar una solución. Todo ello se logra gracias a un equipo especializado en diferentes disciplinas, lo que permite contar y valorar una amplia variedad de opiniones, que a su vez desemboca en formas de resolver problemas y encontrar mejores soluciones.

Figura 1.2 – Fases de la metodología *Design Thinking*[1].

Este modelo cuenta con una serie de fases, que son inspiración, constitución de la idea e implementación [9]. Durante la primera etapa de inspiración, se observa el entorno y se definen los requisitos que debe cumplir o las funcionalidades a cubrir en el proyecto.

[1] Imagen propia, basada en BROWN, 2008 [9].

A continuación, se realiza la constitución de la idea, que trata de llevar a la práctica todo lo planteado en la primera fase, a través de diferentes técnicas que desembocan en la creación de diferentes prototipos.

Por último, en la fase de implementación, se llevan a cabo el propio desarrollo en sí de las ideas y soluciones planteadas en las etapas previas, atendiendo a los requisitos, funcionalidades y objetivos a cumplir.

Por otro lado, se encuentra el enfoque de la metodología *Scrum*. Este modelo pertenece al desarrollo de proyectos de forma ágil y está basado en realizar iteraciones, conocidas como *sprints*, a lo largo de todo el desarrollo del proyecto [10].

Estos *sprints* disponen de una lista de objetivos que tienen que estar cubiertos a la hora de finalizar cada una de estas iteraciones. Además, se realizan reuniones diarias para realizar un seguimiento del progreso y poder exponer los diferentes problemas que hayan podido ir surgiendo y buscar una solución.

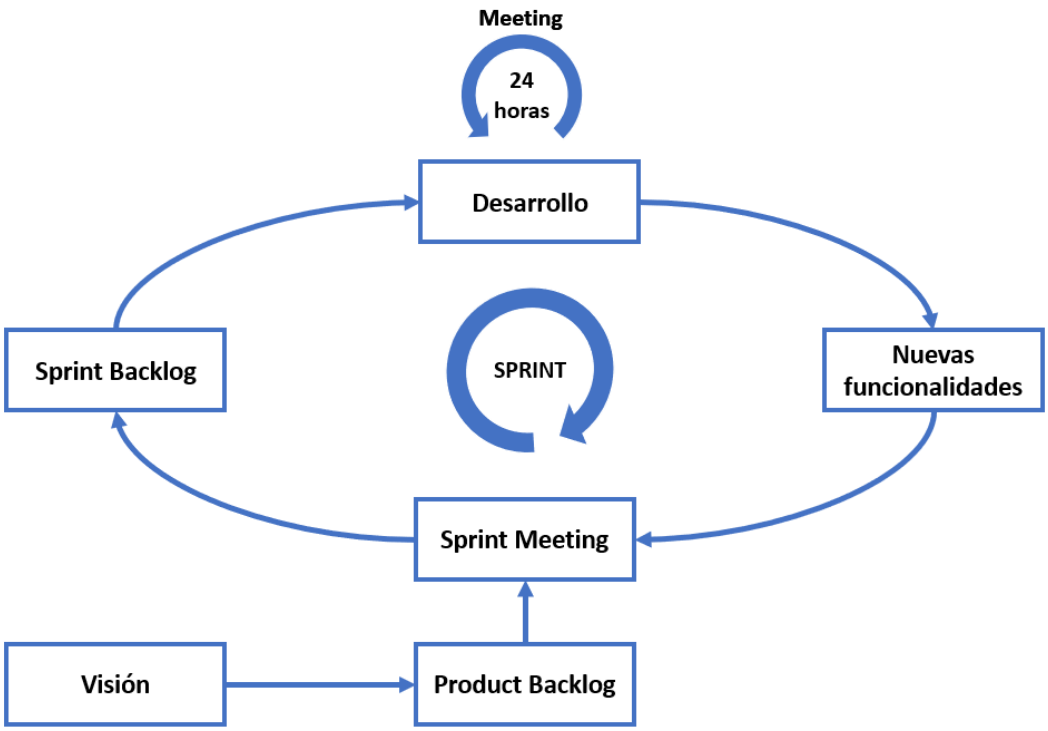


Figura 1.3 – Fases de la metodología Scrum[2].

A la hora de encajar esta metodología con el desarrollo de videojuegos, se puede llegar a hablar del concepto de *Game-Scrum*, que es una adaptación del modelo Scrum a proyectos software cuyo objetivo sea la creación de juegos [11].

[2] Imagen propia, basada en SCHWABER, 2004 [10].

Así, este modelo se basa en un proceso dividido en tres grandes fases, que son preproducción, producción y postproducción. La fase inicial del proyecto se conoce como preproducción y consiste en definir los objetivos y requisitos que se quieren alcanzar en el desarrollo del proyecto. También se definen algunos aspectos básicos, pero de gran trascendencia en el proyecto, como la elección de plataformas objetivo y el software de desarrollo que se va a utilizar en su concepción.

Además, en el caso que ocupa este proyecto, se debe realizar una conceptualización del videojuego y definir las mecánicas básicas de las que debe disponer. Esto no es una tarea sencilla, ya que en muchas ocasiones es difícil saber si una determinada mecánica encajará con el tipo de juego que se quiere desarrollar. De esta manera, se deben proponer diferentes enfoques de la jugabilidad y realizar varios prototipos que ayuden a decidir si las ideas y la conceptualización del juego encajan de una forma adecuada.

La mayoría de los aspectos descritos anteriormente se pueden plasmar en lo que se conoce como *Game Design Document*. Este documento debe determinar qué aspectos hacen que el videojuego sea disfrutable por parte del usuario o jugador final [12]. También tiene que definir las mecánicas básicas del juego, el diseño de los niveles, el tipo de arte gráfico que se quiere emplear, etc.

No obstante, se debe poner especial atención en la concepción de este documento. Su uso procede de las metodologías de desarrollo en cascada (Waterfall) en las cuales la redacción de dicho documento se considera un paso más [13]. Es decir, se da por hecho que todos los requisitos y funcionalidades quedan definidos y son conocidos de antemano y esto no es adecuado en el desarrollo de videojuegos. Esto se debe a que el desarrollo de un juego es un proceso iterativo y que algunas de las mecánicas ideadas inicialmente, pueden llegar a cambiar en el transcurso del desarrollo por otras que se demuestren que encajan mejor en la jugabilidad.

Otro punto de vista a la hora de definir este documento es tratar de evitar que se obtenga un fichero muy denso o difícil de consultar a lo largo del desarrollo. Un enfoque diferente es realizar “diseños de una página” (*One-Page Designs*), propuesto por Stone Librande, director creativo de EA/Maxis [14]. Este modelo implica realizar diseños de una página que puedan ser consultados fácilmente a lo largo del desarrollo. Así, es posible encontrar todo el diseño en un mismo lugar y que el propio documento sea el acto mismo de diseñar el juego. También presenta algunas desventajas, cómo que es difícil de escalar y de actualizar.

Siguiendo con las fases que componen la metodología *Scrum*, después de la preproducción se encuentra la producción. Así, una vez que se han decidido y formalizado los requisitos y las funcionalidades que se deben cubrir en el proyecto y la conceptualización del juego se puede pasar a

dividir en pequeñas tareas que serán cubiertas por diferentes especialistas a lo largo del tiempo. De esta manera, se empieza a definir lo que será cada una de las iteraciones que se llevarán a cabo durante todo el desarrollo del proyecto.

Es necesario puntualizar que las iteraciones o *sprints* que se vayan realizando no son tareas completamente cerradas. Al ser un proceso iterativo, incremental y revisable a lo largo del tiempo, permite la posibilidad de que algunas tareas, sobre todo las que dependen de la finalización de otras, puedan ser ampliadas o corregidas en iteraciones posteriores. De esta forma, si una tarea no puede continuar a la espera de otras, el desarrollador puede ponerse con otra y la pendiente revisarla en una iteración posterior. El objetivo de esto es mejorar el flujo de tareas y que permitan una cierta maleabilidad.

Por último, después de producción se alcanza la fase de postproducción. En este punto el desarrollo del juego se ha completado y se puede realizar una fase de testeo para localizar y corregir posibles fallos. Además, se debe prestar atención a toda la retroalimentación recibida a lo largo del proyecto para crear el documento de *postmortem* [12].

La finalidad de este documento es identificar las fortalezas y debilidades de todo el proceso de desarrollo, los problemas que hayan surgido a lo largo del proyecto y sugerencias de posibles mejoras. Todo esto ayuda a mejorar las estimaciones de futuros proyectos y realizar ajustes al proceso. Este documento también puede recoger otros datos como el número de desarrolladores que han participado en el desarrollo, tiempo de finalización del proyecto, fecha de lanzamiento del producto, etc.

A la hora de enfocar el presente proyecto, se realiza una combinación de ambas metodologías, partiendo de la de *Design Thinking* y finalizando con *Scrum*. Así, se toma de base el modelo de *Design Thinking* para definir la conceptualización del proyecto y la fase de preproducción. A continuación, se enlaza con la metodología ágil de Scrum, lo que llevaría a la fase de producción del juego. Asimismo, se seguirá utilizando *Scrum* para el resto de fases del proyecto, que incluirá la fase de postproducción (cabe destacar que la metodología se aplicará “a pequeña escala”, ya que el presente proyecto ha sido desarrollado por una única persona y estos tipos de metodologías están ideados para equipos de trabajo).

Es decir, en primer lugar se llevarán a cabo la fases propias de *Design Thinking*. Así, el inicio del proyecto estará marcado por la fase de inspiración donde se elegirán las ideas generales del juego que se quiere desarrollar, sus mecánicas, etc. [8].

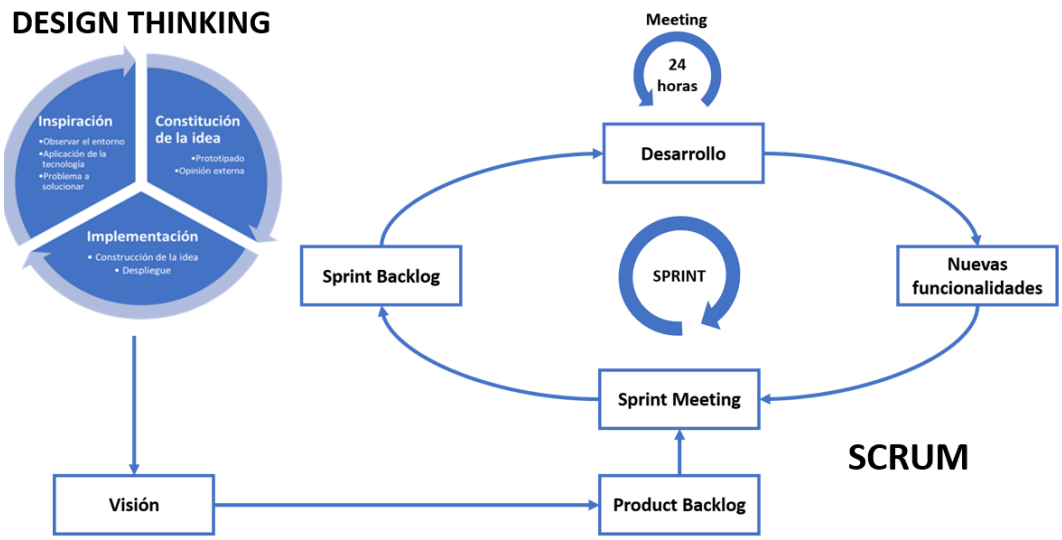


Figura 1.4 – Fases resultantes de la combinación de las metodologías de *Design Thinking* y *Scrum*[3].

A continuación, se pasará a la fase de constitución de la idea. En este momento del desarrollo se enfrentan todas las propuestas sugeridas en la fase anterior y se realiza una selección de entre todas ellas a través del consenso de los miembros del equipo.

Seguidamente, se alcanza la fase de implementación, que será previa a enlazar con la metodología ágil Scrum. En este momento los desarrolladores experimentan con las propuestas alcanzadas en pasos previos y pueden probar cómo funcionan estas ideas a través de la implementación de prototipos. Por último, estos prototipos son probados para la validación de la conceptualización del juego y de las mecánicas implementadas.

De esta forma, se alcanza una fase intermedia entre ambas metodologías. Así, se pasa a definir lo que se conoce como *Product Backlog* en el modelo *Scrum* [15]. Este *Product Backlog* no es sino una lista de las necesidades y requisitos que se deben cumplir a lo largo de las iteraciones o *sprints* del desarrollo.

Así, pasamos a la metodología ágil que ofrece Scrum. En primer lugar, se realiza la *iteration meeting*, que consiste en definir qué partes del proyecto se van a desarrollar a lo largo de una iteración. Tras esta reunión, se obtiene el *Sprint Backlog*, que contiene el listado de requisitos, funcionalidades y tareas a realizar a lo largo del sprint.

A lo largo del *sprint* se van realizando reuniones diarias para realizar un seguimiento del progreso de las tareas. En este momento, los integrantes del equipo indican en qué estado se encuentra su tarea asignada y también se pone de manifiesto si ha surgido alguna dificultad

[3] Imagen propia, basada en HIGUCHI y NAKANO, 2017 [8]; en BROWN, 2008 [9] y SCHWABER, 2004 [10].

o problema en el transcurso del desarrollo y se sugieren posibles soluciones.

Al final de cada iteración se realiza una revisión de la misma, analizando si se han cumplido las tareas y objetivos marcados y una visión en general de todo el conjunto del proyecto. Además, se discute acerca de las cosas que fueron bien durante la iteración y qué cosas son susceptibles de mejora.

En resumen, se hace uso de dos metodologías de forma prácticamente secuencial. En primer lugar, se emplea el modelo de *Design Thinking*, orientado a la conceptualización de la idea del proyecto y la planificación inicial. Posteriormente, se migra a un enfoque ágil, haciendo uso de *Scrum* orientado y adaptando ciertos procesos al desarrollo objetivo de este proyecto, los videojuegos.

1.4. Planificación del Trabajo.

La planificación se dividirá principalmente en cuatro espacios de tiempo correspondientes a cada una de las entregas parciales. De esta manera, los diferentes objetivos y tareas se asignarán a lo largo de estos períodos. Adicionalmente, se aplicará la metodología *Design Thinking* y *Scrum* según corresponda, en cada una de las fases temporales del proyecto.

A continuación, se listan los diferentes objetivos, distribuidos en diferentes grupos:

Instalación del entorno de trabajo y búsqueda de información.

- Instalación del motor Unity [16], incluyendo la creación de cuenta en Unity Store (si procede).
- Instalación de Visual Studio [17].
- Instalación de *plugins* adicionales, incluyendo pruebas de control de versiones y de exportación a Android.
- Búsqueda de información general para el uso de las distintas herramientas y establecer una metodología de trabajo.

Análisis de requisitos y diseño del sistema (prototipo inicial, iteración inicial).

- Análisis de requisitos y funcionalidades del sistema.
- Diseño del sistema teniendo en cuenta los requisitos analizados en el punto anterior.

Implementación y pruebas (prototipo inicial, iteración inicial).

- Implementación y codificación del diseño.

*Para cada iteración y los cambios sugeridos que aporte la retroalimentación de los usuarios, se repetirán las dos anteriores fases teniendo en cuenta el alcance de los nuevos cambios/funcionalidades.

La última iteración estará centrada en la corrección de errores incluyendo:

- Pruebas de verificación de cumplimientos de los requisitos y funcionalidades.
- Corrección de errores y comportamientos no esperados.

Recopilación de recursos (*assets*).

- Búsqueda de recursos gráficos (incluye personajes, animaciones, fondos, *tilemaps*, etc.).
- Búsqueda de recursos sonoros (música/melodías y diferentes efectos de sonido).
- Pruebas e implementación de dichos recursos.

Documentación del proyecto.

- Redacción capítulo 1, introducción.
- Redacción capítulo 2, estado del arte.
- Redacción capítulo 3, definición del juego (mejora y/o ampliación de la primera entrega).
- Redacción capítulo 4, diseño técnico.
- Redacción capítulo 5, diseño de niveles.
- Redacción capítulo 6, manual de usuario.
- Redacción capítulo 7, conclusiones.
- Redacción de glosario y otros apartados o capítulos adicionales.
- Revisión final para entregar al consultor para su revisión, previa a la entrega final.
- Revisión final tras la revisión del consultor de cara a la entrega final.

Material multimedia.

- Vídeo explicación funcionamiento del juego (entrega 2).
- Vídeo cambios realizados (entrega 3).
- Vídeo presentación oral (entrega final).
- Vídeo tráiler final (entrega final).
- Ejecutable del juego (disponible para cada entrega).

A continuación, se especifica la planificación temporal de las tareas a lo largo de los cuatro períodos de tiempo, indicando las fechas límite de cada uno de ellos y con una descripción general sobre las tareas realizadas durante cada fase.

1.4.1. Período 1. Conceptualización y planificación inicial.

Esta entrega cubre la etapa de tiempo comprendida entre el inicio del proyecto hasta el día 28 de febrero.

En esta primera fase se debe planificar y conceptualizar el proyecto. Se trata de establecer una primera aproximación de los requisitos básicos, conceptualización y una planificación temporal de todo el proyecto.

En esta primera fase se aplica principalmente la metodología de *Design Thinking* orientada a analizar las ideas y realizar diferentes propuestas para abordar el proyecto. Para la planificación temporal, se tiene en cuenta la metodología de *Scrum*, ya que será la utilizada a la hora de realizar cada una de las iteraciones a lo largo del desarrollo del proyecto.

De esta forma, se elaborará un pequeño informe con las conclusiones, requisitos, conceptos, ideas y planificación temporal inicial en un pequeño documento que será entregado al consultor antes del día 28 de febrero de 2021.

1.4.2. Período 2. Versión parcial, prototipo inicial.

Esta entrega cubre la etapa de tiempo comprendida entre el 1 de marzo y el 4 de abril.

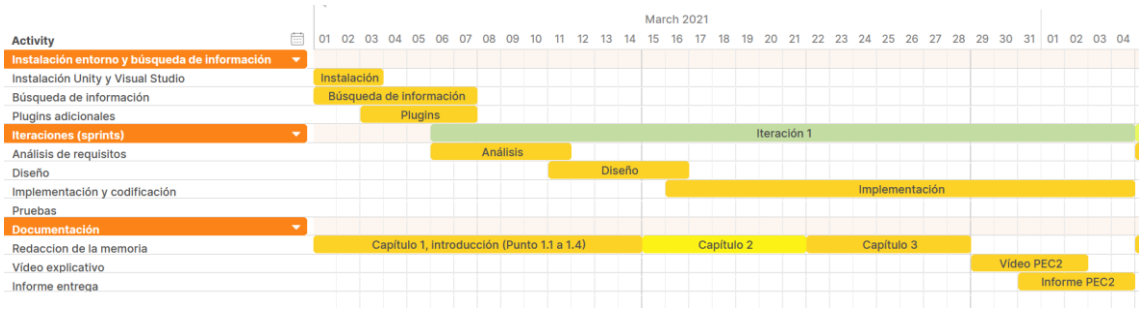


Figura 1.5 – Detalle de la planificación para el segundo período de desarrollo del proyecto[4].

En primer lugar, durante este período se realizará la instalación del entorno y herramientas necesarias para el desarrollo del proyecto. A continuación, se comenzará con la primera iteración o *sprint* del proyecto, que constará con una fase de análisis, de diseño y de implementación. Esta primera iteración se centrará en la constitución de un primer prototipo que represente principalmente la jugabilidad general del juego a desarrollar.

De forma paralela se irán redactando los primeros capítulos de la documentación del proyecto, correspondientes a la introducción, estado del arte y definición del juego. También se realizará el vídeo y el informe del estado del proyecto para entregar antes del último día de este período, correspondiente al día 4 de abril.

1.4.3. Período 3. Versión jugable.

Esta entrega cubre la etapa de tiempo comprendida entre el 5 de abril y el 9 de mayo.

[4] Imagen propia, generada con la aplicación web Tom´s Planner [18].

Figura 1.6 – Detalle de la planificación para el tercer período de desarrollo del proyecto[5].

Durante este período se realizarán varias iteraciones mejorando y ampliando el prototipo inicial. El desarrollo se centrará en cubrir la mayoría de los requisitos y objetivos previstos, pulir el funcionamiento del prototipo y ampliar sus funcionalidades.

Adicionalmente, se irá avanzando en la redacción de diferentes partes de la documentación. Durante este período se centrarán los esfuerzos para finalizar los capítulos de diseño técnico, diseño de niveles y el manual de usuario. También se realizará el vídeo y el informe del estado del proyecto para entregar al finalizar este período, el día 9 de mayo.

1.4.4. Período 4. Versión final.

Esta entrega cubre la etapa de tiempo comprendida entre el 10 de mayo y el 6 de junio.

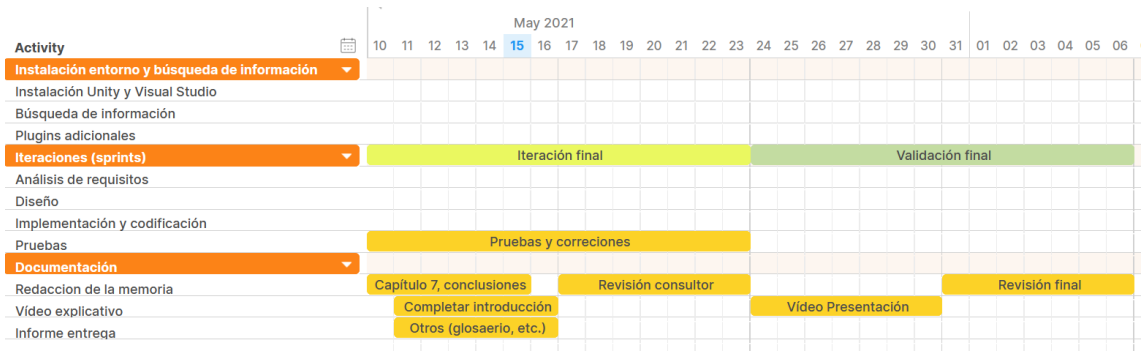


Figura 1.7 – Detalle de la planificación para el cuarto período de desarrollo del proyecto[5].

Durante este período los esfuerzos se centrarán en la iteración final, que consistirá en la identificación de errores y comportamientos no deseables del juego. Se realizará un repaso de todos los requerimientos iniciales y se analizará si se han cubierto todas las funcionalidades planificadas.

Adicionalmente, se revisará toda la documentación y se completarán las partes que hayan quedado pendientes a lo largo del desarrollo. También

[5] Imagen propia, generada con la aplicación web Tom´s Planner [18].

se realizará el vídeo de presentación del proyecto, así como de un pequeño tráiler del juego en su versión final.

1.5. Breve sumario de productos obtenidos.

El principal producto obtenido es el archivo ejecutable y/o instalador del videojuego para dispositivos con sistema operativo Android, mediante un archivo con extensión .apk.

1.6. Breve descripción de los otros capítulos de la memoria.

En el capítulo 2 se realiza un análisis del estado del arte. Se habla sobre el género perteneciente al juego desarrollado y se hace un repaso de las tecnologías y plataformas de desarrollo disponibles para su concepción.

Durante el capítulo 3 se abordará la definición del juego. Se hará un repaso de la idea principal y los objetivos planteados por el juego, así como del subgénero al que pertenece y se describirán otros elementos como la interacción de los diferentes actores del juego, incluyendo al propio jugador.

El capítulo 4 abordará todo lo referente al diseño técnico. Se comentarán las diferentes herramientas de desarrollo empleadas, así como la representación de los diferentes diagramas, incluyendo los de casos de uso, de clases y de secuencia.

En el capítulo 5 se podrá ver en detalle el diseño de niveles. Se comentará el planteamiento principal del juego, así como los componentes de los diferentes elementos que forman el nivel principal del juego.

El capítulo 6 está constituido por un manual de usuario. En esta sección se describe toda la funcionalidad a la que tiene acceso el jugador de forma directa y se proporciona una explicación de los diferentes menús del juego.

En el capítulo 7 se puede encontrar toda la información referente al tema de testeo del juego con usuarios reales. Se describen varias fases durante este período de pruebas y se comentan algunas sensaciones de los usuarios finales.

El capítulo 8 consiste en una reflexión final sobre el proyecto realizado durante el trabajo de fin de Grado, incluyendo una serie de conclusiones y unas posibles mejoras futuras que no se hayan podido explorar durante el presente desarrollo.

2. Estado del arte.

2.1. Género del juego.

A lo largo de los años, han ido surgiendo multitud de géneros en el mundo de los videojuegos. Desde el clásico plataformas protagonizado por el famoso fontanero, hasta juegos completamente inmersivos gracias a la realidad virtual, que provocarán una experiencia totalmente diferente a sus usuarios.

El consumo de videojuegos y su forma de jugar se vieron ampliados gracias a la aparición de dispositivos inteligentes, especialmente los conocidos *smartphones*. Esto hizo que otros sistemas, como las consolas o los ordenadores tomarán una dirección distinta a este nuevo estilo de juego táctil.

Estos sistemas dotados de una pantalla táctil, provocó la aparición de una nueva forma de jugar sin necesidad de controles físicos. Este hecho fue uno de los principales reclamos de la consola portátil Nintendo DS, que sin duda fue la pionera en establecer una nueva jugabilidad gracias al uso que se hizo de una de sus pantallas táctiles.

Otro factor característico de la nueva era de dispositivos móviles fue el aumento de potencia que sufrieron en poco tiempo. Gracias a ello, hoy día no existen grandes impedimentos para desarrollar casi cualquier tipo de juego en este tipo de dispositivos. Es más, este aumento de prestaciones en tan poco tiempo provoca que los desarrollos tengan que adaptarse a multitud de versiones de los dispositivos disponibles en el mercado, así como lidiar con todo tipo de resoluciones y diferentes formatos de pantalla.

Uno de los géneros más tradicionales se puede encontrar en los denominados juegos de rol. Estos juegos no son exclusivos del mundo de los videojuegos, ya que existen muchas vertientes pioneras, desde juegos de mesa (*pen and paper, table-top*) hasta juegos de rol de acción real (*live-action*) [19].

En el caso que nos ocupa, la denominación común utilizada en la industria es *Role Playing Game* (RPG). Este género de juegos se puede definir como aquellos dónde un jugador controla a uno o varios personajes y los guía a través de una serie de desafíos o misiones hasta alcanzar su objetivo. Además, los personajes pueden ir mejorando sus habilidades y se ofrece la posibilidad de exploración y resolución de puzles [20].

Más allá de esta definición podemos encontrar una amplia variedad de características que provocan que hayan surgido subgéneros dentro de esta clasificación. De esta manera, se pueden encontrar los denominados *Japanese Role Playing Game* (JRPG), *Action Role Playing Game* (ARPG), *Tactical Role Playing Game* (TRPG) e incluso mezcla

con otros tipos de juegos, como los *Massively Multiplayer Online* (MMO), originando lo que se conoce como *Massively Multiplayer Online Role Playing Game* (MMORPG).

En el caso del proyecto que nos ocupa, nos centraremos en el género de la vertiente japonesa, JRPG. A pesar de esta denominación popular, este tipo de género no sólo es desarrollado en Japón. La razón por la que se le conoce popularmente por este nombre probablemente sea por la nostalgia y asociación temporal de los jugadores anteriores a la convergencia tecnológica de consolas y ordenadores [21]. Otra posibilidad es que el término se use para referirse a la época dorada de finales de los 80 y los 90, cuando los juegos desarrollados en Japón eran muy populares.

La razón para elegir este tipo de rol es poder aprovechar el nicho de mercado que ofrece en las plataformas móviles. Otro motivo importante es la alta competitividad y saturación del mercado en otros tipos de géneros, en los que en muchas ocasiones se abusa del modelo *free-to-play*, provocando que sea muy difícil la entrada a estudios noveles.

No obstante, esto no quiere decir que no exista una fuerte competencia en este campo. La veterana empresa japonesa Square Enix dispone de un gran listado de este tipo de juegos, ayudándose de su largo recorrido en la industria. Sin embargo, existen algunos ejemplos que demuestran que es posible competir y hacerse un hueco en el mercado a pesar de estos gigantes de la industria, siendo un buen ejemplo las entregas de juegos como “Cat Quest” o “Temtem”.

2.2. Tecnologías y plataformas de desarrollo actuales.

En la actualidad existen muchas soluciones y herramientas orientadas al desarrollo de videojuegos. De esta manera, es posible encontrar todo tipo de aplicaciones, desde las más básicas en las que no es necesario contar con conocimiento técnicos hasta soluciones todoterreno que permiten el desarrollo de prácticamente cualquier tipo de juego.

De esta manera, se pueden encontrar en el mercado programas como Stencyl, que permite desarrollar juegos sin necesidad de escribir código y resulta muy versátil, especialmente si se desea crear un juego de plataformas.

Otro programa enfocado al desarrollo de un determinado género sería RPG Maker. Tal como indica su nombre, se trata de una herramienta centrada en el desarrollo de juegos de género RPG. Dispone de muchas herramientas que facilitan la creación de las diferentes mecánicas para este género de juegos.

Por otro lado, se encuentran los motores de juegos que ofrecen posibilidades casi ilimitadas a la hora de crear los mismos. En este caso se encuentran soluciones como Unity o Godot, que son herramientas

con una gran flexibilidad, ya que permiten el desarrollo de juegos en 2D y 3D, así como la implementación de casi cualquier tipo de jugabilidad o mecánica.

También es posible optar por el uso de Unreal Engine. Este motor de juegos es muy popular a la hora de diseñar juegos bajo entornos 3D y ha demostrado su versatilidad en la concepción de juegos de una amplia gama de géneros.

A pesar de contar con todo este tipo de herramientas, existe la posibilidad de ir más allá de ellas creando un motor de juego personalizado. Es decir, es posible crear una herramienta que se ajuste mejor a las necesidades del proyecto que se quiera realizar. No obstante, no resulta una tarea fácil, ya que se requieren muchos conocimientos técnicos y el desarrollo de este tipo de herramientas suelen tener un alto coste y una gran inversión de tiempo.

En resumen, existen muchas posibilidades a la hora de decantarse por una herramienta u otra. Por tanto, se debe escoger la aplicación adecuada según el tipo de juego que se quiera desarrollar, teniendo en cuenta sus mecánicas jugables. Otro aspecto a tener en cuenta será la experiencia previa en el uso de las distintas soluciones y los conocimientos técnicos de los diferentes lenguajes de programación entre los que se pueden elegir dependiendo de la herramienta.

3. Definición del juego.

3.1. Idea del juego.

3.1.1. Descripción del juego.

El juego se basará en una jugabilidad clásica de combates por turnos, con un estilo artístico en dos dimensiones, empleando el conocido pixel art. El jugador podrá avanzar a través de diferentes niveles y lugares que podrá ir desbloqueando según avance en la historia principal.

La idea principal del juego es profundizar en la investigación de un antiguo poder, explorando diferentes zonas y hablando con distintos personajes. Los protagonistas podrán mejorar sus habilidades combatiendo contra diferentes enemigos y tendrán que enfrentarse a diferentes jefes de zona durante su aventura.

3.1.2. Subgénero y referencias a videojuegos existentes.

El subgénero principal del juego es el de rol o JRPG. Estos juegos se caracterizan por contar una historia que suele profundizar en los personajes protagonistas, los cuales suelen compartir un objetivo común.

Así, el enfoque principal del juego será contar una historia avanzando por una serie de niveles y/o misiones hasta alcanzar el destino final. De esta forma, el jugador podrá explorar e investigar en las diferentes localizaciones y profundizar en la historia del juego.

Además, a lo largo de los diferentes emplazamientos será posible enfrentarse a varios enemigos para fortalecer a los héroes protagonistas. Para ello, se ha elegido un sistema de juego clásico basado en el combate por turnos.

Una de las principales referencias pioneras en este género y que lo representa perfectamente sería la saga de juegos “Final Fantasy”, principalmente las entregas de la primera a la sexta, entre otros *spin-offs*. Otro ejemplo actual sería la franquicia “Pokémon”, cuya base jugable se mantiene intacta (aunque con mayor profundidad y cantidad de interacciones).

También es posible encontrar referencias a este género en algunos juegos *indie* como “Battle Chasers: Nigthwar”, “Darkest Dungeon” o “Epic Battle Fantasy”, entre otros.

3.1.3. Tipo de interacción juego-jugador.

El juego cuenta con la premisa principal de avanzar en la historia y fortalecer a los personajes para que puedan ganar los combates contra sus enemigos.

De esta forma, es posible interactuar con el entorno de diferentes formas. Por ejemplo, a lo largo del nivel principal es posible hablar con diferentes *Non Playable Characters* (NPCs) para conseguir nuevas pistas o ahondar en el mundo del juego, descubriendo diferentes aspectos acerca de los personajes, lugares visitados, etc.

Por otro parte, otra de las principales interacciones se encuentra en el enfrentamiento contra diferentes criaturas y enemigos. Así, el jugador podrá dar diferentes órdenes a sus personajes para que éstos salgan airosos de los enfrentamientos y puedan mejorar sus habilidades.

Además, el jugador podrá interactuar en ciertos momentos cuando sus personajes estén realizando las acciones solicitadas. Así, cuando un personaje recibe la orden de ejecutar un hechizo, el jugador podrá ayudarle a través de la resolución de un pequeño reto matemático.

De esta manera, el jugador debe estar atento al combate, ya que la correcta resolución de estos retos propiciará un incremento en la fuerza del hechizo y se logrará infringir mayores daños a los enemigos.

3.1.4. Plataforma de destino.

La plataforma de destino principal son dispositivos móviles, concretamente con sistema operativo Android. No se cierra la posibilidad de hacer algunas pruebas con otras plataformas de destino como el ordenador/*desktop*, ya que el motor Unity es bastante flexible a la hora de cubrir diferentes plataformas en su exportación.

La elección de la plataforma también está basada en el tipo de juego. La mecánica se basa en ofrecer niveles en los que se puede participar en combates de corta duración, que resultan ideales para sesiones ligeras de juego. Además, es posible retomar la partida desde el último combate participado o desde algún punto de control, como las diferentes aldeas y ciudades. También es posible guardar la partida en cualquier momento en el nivel principal de juego.

De esta forma, el usuario puede jugar una partida rápida simplemente abriendo la aplicación en su móvil en cualquier parte, tanto si dispone de poco tiempo o de varias horas.

3.2. Conceptualización.

3.2.1. Historia y ambientación.

La historia tiene lugar en un mundo que quedó arrasado en el pasado debido al descontrol de poder que llevaron a cabo sus habitantes. Este poder se basaba en el conocimiento, que desembocó en el descubrimiento de la magia, una ciencia que acabó fuera de control con unas consecuencias muy graves.

La acción se sitúa 1200 años después de la gran catástrofe que dejó al mundo en ruinas. En la época actual, los habitantes apenas utilizan la magia, ya que es una ciencia olvidada y temida. Sin embargo, nuestros protagonistas tendrán que hacer uso de este poder tan controvertido si quieren alcanzar su meta, que no es otra que salvar al propio mundo de su desconocimiento.

Deberán descubrir qué se esconde en el propio origen de la magia y encontrar al hechicero primigenio, aquel que se dice fue el mismísimo descubridor de la magia pura. El camino no será fácil, porque según la leyenda, el espíritu de este hechicero se encuentra en el antiguo castillo situado en las tierras prohibidas y custodiado por cuatro antiguos guardianes. Además, el acceso a dichos parajes puede que también tenga un precio…

3.2.2. Definición de los personajes y elementos de juego.

En principio, el juego está concebido para contar con un total de cuatro héroes. El presente proyecto se constituye con una misión o nivel intermedio de la trama, por lo que los cuatro héroes protagonistas ya se han reunido. No obstante, en una ampliación del juego, se escalaría la historia poco a poco, empezando la aventura con uno o dos personajes protagonistas y el tercer y cuarto héroe se unirían más tarde, de forma justificada en la historia.

El número final de personajes se estableció a partir de las pruebas realizadas para pulir la jugabilidad, ya que este número afectaba directamente al ritmo de juego. Es decir, contar con pocos personajes puede hacer que el juego mejore su ritmo, pero puede aumentar su dificultad. No obstante, disponer de demasiados aliados puede hacer que las batallas sean lentas, tediosas e incluso que se pierda el interés del jugador.

Por otro lado, también se deberá prestar atención al número de enemigos y la dificultad que representan. Se pretende ir escalando el nivel de reto a medida que se vayan superando y avanzando a través de las diferentes zonas, dando un pequeño respiro al jugador cada cierto número de combates.

Otro aspecto a tener en cuenta será el equilibrio con que debe contar el minijuego de preguntas aritméticas. Este reto se presenta cada vez que

un personaje usa un hechizo y su objetivo es que el jugador ayude al héroe potenciando su ataque contestando de forma correcta a la pregunta planteada.

La relación entre la dificultad de la pregunta y el tiempo resulta vital para lograr un equilibrio en el reto. Además, se debe tener en cuenta otros aspectos como los números involucrados y el tipo de operaciones implicadas en el cálculo, ya que podrían llegar a dificultar demasiado las preguntas y desalentar al jugador si falla demasiadas veces.

Por todo esto y para poder acotar la jugabilidad, se experimentará con diferentes enfoques y ajustes en las iteraciones que se llevarán a cabo en los prototipos. Estas iteraciones se centrarán en buscar un equilibrio de la mecánica de juego y la satisfacción del jugador, tanto a nivel de objetivos, como de ritmo de juego.

3.2.3. Interacción entre los actores del juego.

La principal interacción se produce en el nivel principal, donde es posible poder hablar con otros personajes o bien enfrentarse a distintos enemigos. Por un lado, es posible hablar con otros NPCs para buscar información y avanzar en la historia y por otro, se puede participar en enfrentamientos para subir de nivel a los héroes protagonistas.

Otra interacción sería en el propio combate en sí. De esta forma, los héroes van recibiendo órdenes por parte del jugador y realizan sus distintos ataques contra los enemigos. De la misma manera, los enemigos pueden ir devolviendo los ataques e intentar debilitar al grupo.

Conforme ambos bandos vayan realizando sus ataques, la vida de cada personaje irá disminuyendo y si se agota, será eliminado. El combate finaliza cuando todos los integrantes de uno de los grupos sean derrotados.

3.2.4. Objetivos planteados al jugador.

Uno de los objetivos principales del jugador es avanzar a través de la historia del juego. Así, deberá ir hablando con distintos personajes y desvelar los misterios del mundo del juego. También deberá encontrar a los personajes adecuados para poder avanzar en la trama y acceder a nuevas zonas y localizaciones.

Otro objetivo del jugador es demostrar su habilidad en los combates. Así, tendrá que decidir qué órdenes dará a sus héroes y tendrá que definir una estrategia en cada combate. Por ejemplo, el jugador debe pensar si resulta más beneficioso centrar los ataques en un mismo enemigo o atacar a varios contrarios a la vez.

Por tanto, al superar cada batalla sus personajes ganarán experiencia y podrán subir de nivel, fortaleciendo sus habilidades y abriendo la posibilidad de enfrentarse con más soltura a enemigos más fuertes.

3.2.5. Concept art.

El estilo artístico elegido será el denominado *pixel art*, por tanto, se deberá prestar atención a la forma de usar estilo sobre todo en el plano técnico, ya que una mala manipulación puede provocar diferentes errores gráficos.

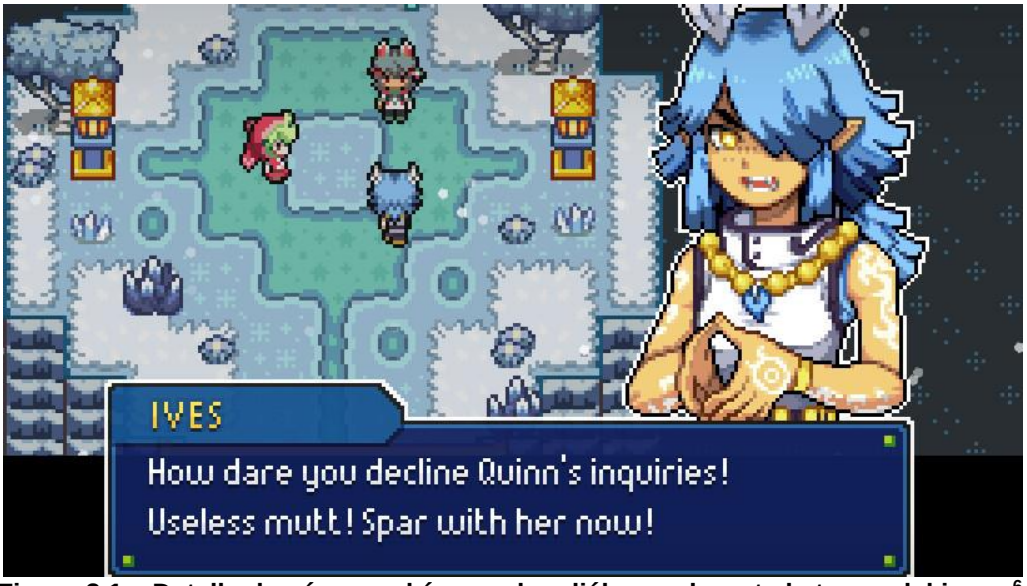


Figura 3.1 – Detalle de cómo podrían ser los diálogos durante la trama del juego[6].

La historia se irá contando a través de diferentes personajes, tanto principales como secundarios. En ocasiones se hará uso de breves escenas centradas en la historia, especialmente al principio de zonas nuevas o cuando suceda algún acontecimiento importante en la trama.

Por otro lado, la forma de enfrentarse a los enemigos será a través del sistema de combates por turnos. Así, cada grupo podrá decidir sus acciones en cada uno de los turnos y éstas serán ejecutadas de forma ordenada según diferentes parámetros, como la agilidad de los diferentes combatientes.

[6] Imagen perteneciente al juego “Heartbeat”, 2018 [22].

Figura 3.2 – Detalle de cómo podría ser el diseño de los combates[7].

3.2.6. *Edutainment* y *serious games*.

A pesar de que el presente proyecto no se trata de un juego orientado al sector educativo, se quiso experimentar con una determinada parte de la jugabilidad, concretamente el minijuego de preguntas matemáticas.

De esta forma, es una manera de animar al jugador a mejorar sus habilidades aritméticas y una forma de dotar al juego de una pincelada de la denominación de *edutainment*, o incluso *serious game*.

Este hecho también ofrece la posibilidad de expandir este tipo de jugabilidad en una posible línea de desarrollo futura, ampliando el minijuego con otro tipo de cuestiones, como por ejemplo, preguntas de hechos históricos.

3.3. Definición de requisitos.

Los requisitos abarcan la visión de la que dispone el usuario final frente al comportamiento del sistema y la visión del desarrollador desde el prisma de las características internas con las que debe contar el software. Todo ello incluye la gestión del comportamiento del sistema bajo determinadas condiciones, lo que desemboca que la aplicación final sea adecuada para su uso por parte de los usuarios finales [24].

3.3.1. Requisitos funcionales.

Los requisitos funcionales de un sistema describen las funcionalidades que debe cubrir el software a desarrollar. De esta manera, se deben definir las características y necesidades del usuario que vaya a utilizar este software.

[7] Imagen perteneciente al juego “Orangeblood”, 2020 [23].

En el juego, el principal rol es el propio jugador. Este jugador tendrá acceso por completo a todas las funcionalidades de la aplicación y podrá configurar y/o personalizar ciertos parámetros a sus necesidades.

A continuación, se listan los requisitos funcionales del sistema, acompañados de una breve descripción y un código de identificación de cada uno de ellos:

Identificador	Requisito Funcional	Descripción
RF01	Iniciar partida	El jugador puede iniciar el juego desde el principio o cargar una partida anterior.
RF02	Mostrar tutorial	Será posible mostrar un pequeño tutorial en el inicio del juego. También se podrá acceder a este tutorial desde el menú principal del título.
RF03	Mostrar configuración	El usuario podrá modificar diferentes parámetros de su partida (volumen música, efectos, etc.) desde un submenú en la pantalla de título.
RF04	Mostrar créditos	Se podrá consultar la información referente al desarrollador del juego y los autores de los diferentes recursos.
RF05	Desplazamiento del personaje	El jugador podrá desplazar a su personaje a través del nivel principal e interactuar con el mundo del juego.
RF06	Combatir	El jugador podrá participar en combates y dar órdenes a sus personajes para salir victorioso y poder avanzar en la historia.
RF07	Gestionar inventario	Se podrán usar y se tendrá acceso a los objetos tanto en combate como en el nivel principal.

Tabla 3.1 – Requisitos funcionales del proyecto.

De esta manera quedan listados todos los requisitos funcionales, los cuales se centran en proporcionar al usuario (jugador) las funcionalidades necesarias para poder interactuar con el juego y poder realizar las acciones necesarias para lograr las metas planteadas por la aplicación.

3.3.2. Requisitos no funcionales.

Los requisitos no funcionales son una serie de restricciones que vienen impuestas por los propios requisitos funcionales definidos en el punto anterior. En otras palabras, por sí mismos no definen un determinado comportamiento o funcionalidad de sistema.

A continuación, se listan los requisitos no funcionales del sistema, acompañados de un código de identificación de cada uno de ellos.

Identificador	Descripción del requisito
RNF01	El juego fomentará dar un objetivo y reto al jugador, otorgando sensación de progreso.
RNF02	Sistema de guardado que permitirá al usuario poder guardar su progreso, con posibilidad de continuar un nivel en el punto en que lo dejó en su anterior partida.
RNF03	Generación aleatoria de enemigos, que permitirá que los oponentes sean diferentes cada vez que se accede a una zona o al propio combate en sí.
RNF04	Sistema de avisos e información al jugador, que permitirá informar al jugador cuando una determinada acción no pueda llevarse a cabo y la razón.
RNF05	El sistema debe permitir el almacenamiento de un fichero con la información de guardado de la partida del jugador.
RNF06	El diseño de la aplicación debe cumplir los criterios de usabilidad de pantallas táctiles de dispositivos móviles, ya que es la principal plataforma de destino.
RNF07	Compatibilidad con las últimas versiones del sistema operativo Android.
RNF08	Compatibilidad con diferentes resoluciones y formatos de pantalla.

Tabla 3.2 – Requisitos no funcionales del proyecto.

De esta manera quedan definidos todos los requisitos no funcionales, los cuales se centran en aportar características que debe cumplir el juego, así como la compatibilidad de la aplicación en los diferentes dispositivos y versiones.

4. Diseño técnico.

4.1. Elección del entorno y herramientas.

En primer lugar, se debe identificar a grandes rasgos los requisitos principales con los que cuenta este proyecto. El objetivo principal es el desarrollo de un juego en dos dimensiones, con preferencia en el uso de *pixel art* como medio artístico.

El tipo o género del juego también puede llegar a influir en la decisión de la herramienta a utilizar. En el caso particular de este proyecto, el género principal se centrará en el rol por turnos. De esta manera, al contar con unos requisitos generales, es posible barajar diferentes opciones de herramientas que puedan encajar para el desarrollo del proyecto.

A pesar de la amplia variedad de motores disponibles, la decisión final se tomó a partir de una selección de tres de ellos. De esta manera, la lista de posibles candidatos se vio reducida, lo que facilitó la comparación de sus principales características y ayudó a poder tomar una decisión lo más adecuada posible ante este proyecto de desarrollo.

Finalmente, se barajaron como principales candidatos los siguientes motores de desarrollo:

- Unity [16].
- RPG Maker [25].
- Godot [26].

La primera opción, Unity, se escogió por su amplia comunidad activa y su gran versatilidad, tanto en entornos de dos como de tres dimensiones. Otra ventaja adicional es que permite exportar a una amplia gama de plataformas.

La segunda elección, RPG Maker, se tuvo en cuenta porque es un motor centrado en el tipo de juego que se pretendía desarrollar, concretamente en el género de rol (JRPG).

Figura 4.1 – Logo [27] del motor Godot, una de las opciones que se barajó para usar en el proyecto.

La tercera opción, Godot, fue barajada por ser otra herramienta todoterreno y que permitía poder trabajar con distintos estilos gráficos. Por tanto, esta característica resultaba ideal para el posible tipo de arte gráfico que se pensaba utilizar en el proyecto, el denominado *pixel art*.

Finalmente se optó por emplear el motor Unity. Para empezar, una de sus ventajas es su rápida curva de aprendizaje y la posibilidad de encontrar mucha información disponible en su comunidad oficial. Además, permite trabajar con una amplia variedad de tipos de archivo, lo que facilita la compatibilidad de los recursos a utilizar.

Otra ventaja es que permite instalar los archivos de desarrollo de Android de forma casi transparente y es posible realizar una exportación a dicha plataforma sin requerir de ninguna herramienta o recurso adicional, permitiendo ahorrar mucho tiempo e incompatibilidades.

Las otras dos opciones fueron descartadas por diversos motivos. En el caso de RPG Maker fue suprimido principalmente por la cantidad de versiones existentes y por su modelo económico. Así, a pesar de que ofrece una prueba gratuita del programa y de todas sus versiones, es difícil establecer cuál de ellas sería la que mejor podría llegar a encajar con este proyecto.

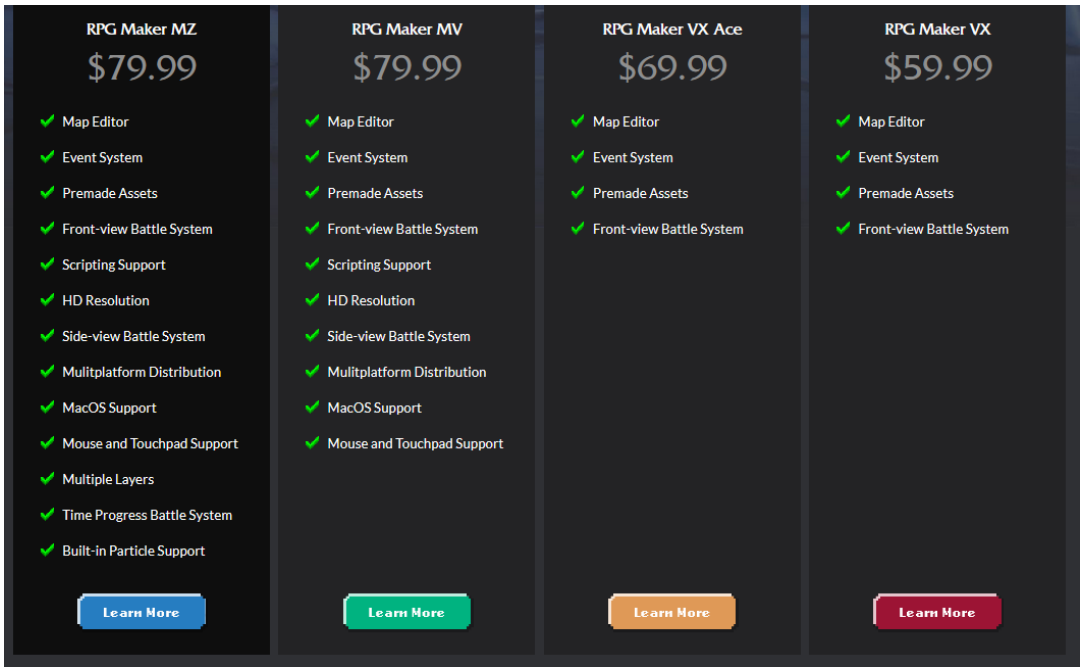


Figura 4.2 – Comparativa de las diferentes versiones de RPG Maker [28].

Godot fue otra de las opciones descartadas. A pesar de que su flujo de trabajo era una buena opción para el proyecto, el hecho de que no dispusiera de una comunidad tan activa como las otras dos opciones, suponía una gran desventaja.

Por otro lado, era necesario elegir un sistema de control de versiones para el código. Las opciones que se barajaron fueron Subversion [29], Git [30] y Perforce [31]. El objetivo era elegir el sistema que resultara más sencillo de usar para un integrante, ya que el proyecto es desarrollado por una única persona. Por esta razón, Perforce fue la primera opción descartada debido a que era un sistema demasiado

sobredimensionado y complejo para ser empleado por una única persona.

Figura 4.3 – Logo [32] del sistema de control de versiones Git.

Las dos opciones restantes, Git y Subversion, resultaban bastante acertadas para este proyecto. Finalmente se decidió usar Subversion por varios motivos. En primer lugar, emplear Git para ficheros pesados puede requerir algo de formación, ya que para el desarrollo de juegos debe configurarse con cuidado, mientras que Subversion resultaba menos restrictivo en el tamaño de los ficheros.

En segundo lugar, existe un plugin de integración de Subversion para Unity con muy buenas críticas, mientras que el de Git no era tan popular.

Por último, aunque menos importante, el desarrollador del proyecto tiene más experiencia en el uso de Subversion, por lo que suponía una pequeña ventaja adicional para elegir este sistema.

4.1.1. Plataforma de desarrollo escogida y requisitos.

La plataforma de desarrollo elegida es un ordenador con sistema Windows 10 Pro. Se ha elegido esta plataforma porque es el sistema principal y personal del desarrollador de este proyecto.

Así, los requisitos para la ejecución del entorno y motor Unity son los siguientes:

- Versión del sistema operativo: Windows 7 (SP1+) y Windows 10, sólo versiones de 64-bit.
- CPU: arquitectura X64 con soporte de instrucciones SSE2.
- API gráfica: GPUs con soporte DX10, DX11 y DX12.
- Requisitos adicionales: drivers adicionales de terceros/fabricantes del dispositivo de uso.
- Para el desarrollo con plataforma final Android, es necesario al menos instalar el Android SDK (10/API 29), Android NDK (r19) y OpenJDK, que son instalados por defecto con la herramienta Unity Hub.

Toda esta información procede de la documentación oficial de Unity [33].

4.1.2. Plataforma de destino escogida y requisitos.

Se ha escogido como plataforma de destino dispositivos con sistema operativo Android. La razón principal es porque se trata de un sistema muy extendido y las aplicaciones son fáciles de instalar sin necesidad de que estén alojadas en la tienda oficial de la compañía.

Los requisitos con los que debe contar la plataforma de destino o ejecución del juego final son los siguientes:

- Versión del sistema operativo: Android 4.4 (API 19)+.
- CPU: ARMv7 con soporte Neon (32-bit) o ARM64.
- API gráfica: OpenGL ES 2.0+, OpenGL ES 3.0+, Vulkan.
- Requisitos adicionales: 1GB+ de memoria RAM. Los dispositivos deben disponer al menos de los requisitos de compatibilidad definidos por Google [34], limitados a los tipos de dispositivos handheld (sección 2.2), televisión (sección 2.3) y tablets (sección 2.6).
- El dispositivo debe soportar nativamente el sistema operativo Android.

De nuevo, toda esta información se puede encontrar en la documentación oficial de Unity [33].

La intención de este proyecto era ofrecer soporte a todo tipo de dispositivos y lograr el mayor grado de compatibilidad entre los diferentes dispositivos del mercado.

Durante las pruebas, se pudo confirmar que los modelos de media y alta gama que cumplían los requisitos mínimos, no tuvieron ningún tipo de problema para ejecutar el juego y poder completarlo hasta el final. Además, el rendimiento fue adecuado durante la ejecución del juego.

4.1.3. Listado de herramientas utilizadas.

Balsamiq Wireframes [35]. Se trata de una herramienta que permite realizar bocetos de interfaces de usuario (también conocidos como prototipos de baja fidelidad). Resulta muy útil para realizar un prototipo rápido y representativo del tipo de interfaz que se busca desarrollar. Además, soporta bocetos de dispositivos móviles, por lo que resulta muy conveniente para este tipo de proyecto, ya que permite apreciar cómo quedará la interfaz en el dispositivo final.

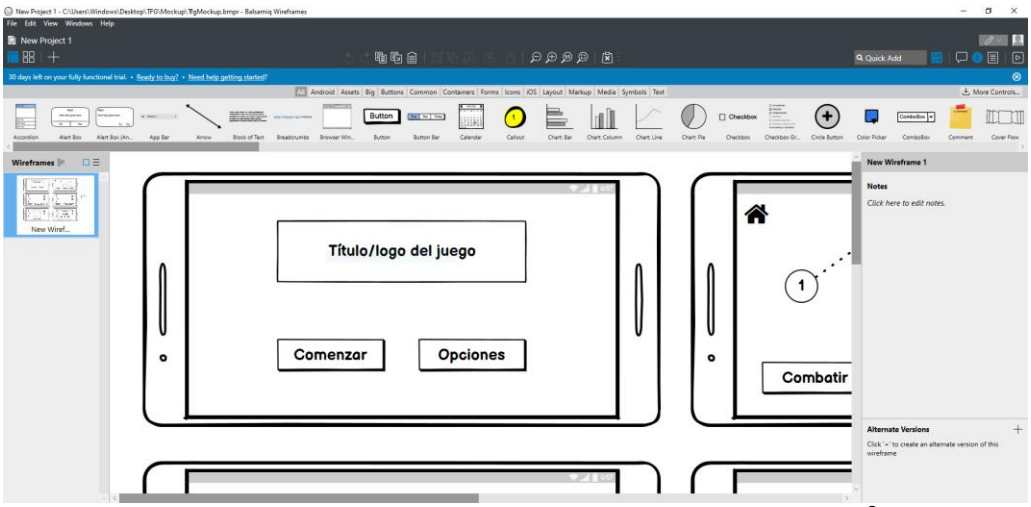


Figura 4.4 – Detalle de la interfaz de Balsamiq Wireframes[8].

Visual Paradigm [36]. Es una herramienta *Computer Aided Software Engineering* (CASE) que permite modelado *Unified Modeling Language* (UML). Esta herramienta es muy útil en varias de las fases de desarrollo del software del proyecto, ya que permite modelar diferentes tipos de diagrama. Por ejemplo, tras la fase de definición de requisitos resulta muy útil para modelar los diferentes diagramas de casos de uso de los que debe disponer el juego final. También se utiliza en otras fases y modelado de diagramas, como los de clase y de secuencia.

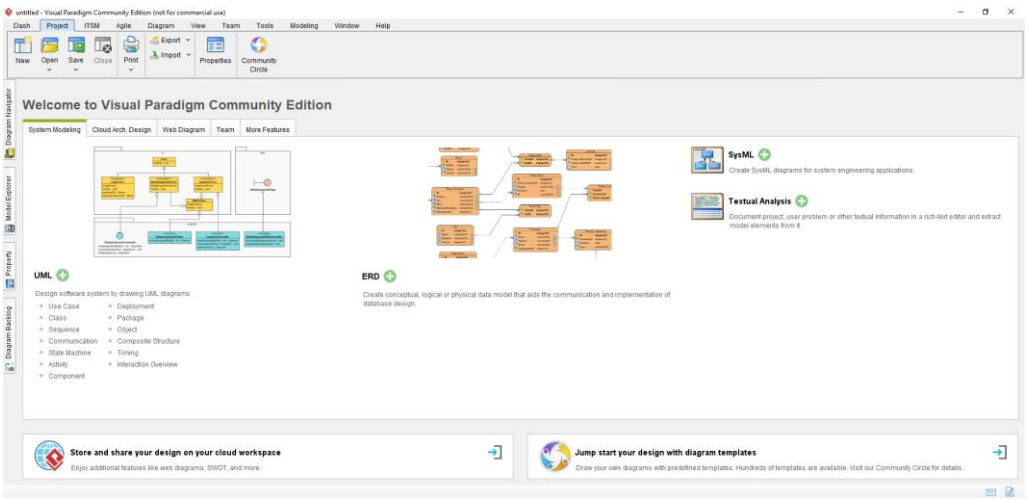


Figura 4.5 – Detalle de la interfaz de Visual Paradigm[9].

Unity [16]. Esta herramienta es el eje principal del proyecto, ya que es la base para poder realizar su desarrollo una vez se han superado las fases iniciales del desarrollo, como las de conceptualización y definición de requisitos. Esta potente herramienta permite poder desarrollar el juego en sí, y aporta diferentes flujos de trabajo para los diferentes recursos a utilizar en el proyecto. Así, permite administrar

[8] Captura de pantalla propia de la herramienta Balsamiq Wireframes [35].
[9] Captura de pantalla propia de la herramienta Visual Paradigm [36].

todo tipo de elementos gráficos, pasando por los sonoros, así como a nivel de interfaz y gestión de las diferentes clases a nivel de código.

Visual Studio [17]. Se trata de un *Integrated Development Environment* (IDE) que permite codificar en diferentes lenguajes de programación, incluido C#, que es el lenguaje usado por el motor Unity. Así, esta herramienta permite agilizar la creación de código y permite realizar depuración en tiempo real vinculando el código a la instancia de Unity, haciendo más fácil la detección de errores en tiempo de ejecución.

TortoiseSVN [37]. Esta sencilla pero potente herramienta permite administrar todas las opciones de los repositorios del sistema de versiones del código. Posteriormente se integrará en Unity a través de un *plugin* de terceros (especificado en el apartado siguiente).

Audacity [38]. Este programa de edición de audio se ha utilizado para normalizar las diferentes melodías del juego, así como los efectos de sonido. De esta forma, se ha logrado que todo el apartado sonoro fuera uniforme.

4.1.4. *Plugins* adicionales.

Wise SVN [39]. Se trata de un plugin de la Unity Asset Store que permite integrar SVN como gestor de control de versiones dentro del propio motor Unity. Facilita la tarea de control de código, además de agilizar algunas tareas a la hora de sincronizar ficheros, como los archivos de extensión .meta.

Joystick Pack [40]. Se trata de un plugin de la Unity Asset Store que permite utilizar un joystick virtual para controlar a los personajes. Este plugin resulta ideal para ser utilizado en una pantalla táctil.

4.1.5. Listado de recursos.

Los recursos utilizados han sido recopilados de diferentes servicios y fuentes, como por ejemplo, Humble Bundle, itch.io y Patreon.
A continuación, se especifican cada uno de los recursos utilizados y su autor original.

- Los fondos de los combates, de la pantalla de historia y de la de título son obra de: Tyler Warren [41].
- Los personajes y sus animaciones son obra de: finalbossblues [42] [43].
- El fondo del nivel principal (*overworld, tilemap*) es obra de: Megatiles [44].
- La banda sonora es obra de: Andrey Sitkov (Muz Station Productions) [45].
- Los efectos de sonido proceden de: https://www.zapsplat.com [46].

4.2. Diagramas de caso de uso.

En este punto se realizará un modelado del sistema a desarrollar mediante diagramas de casos de uso. Estos diagramas constan de una representación del sistema (a la que se le suele referir como sujeto), los propios usuarios (o actores) y las diferentes funcionalidades o tareas que son representadas mediante los casos de uso.

También se definen las relaciones y posibles referencias que hay entre todas las partes implicadas en el sistema, ya sean actores o los propios casos de uso.

Asimismo, cada diagrama contará con una pequeña tabla explicativa en la cual se indicará el nombre del diagrama, los actores implicados, así como de unas determinadas condiciones de entrada y de salida. De forma adicional, también se indicarán posibles escenarios alternativos.

Todos los siguientes diagramas son de producción propia utilizando la herramienta Visual Paradigm [36].

4.2.1. Diagrama general.

A continuación, se detalla el diagrama de casos de uso general de la aplicación. A partir de este diagrama, se detallarán cada uno de los casos de uso, ampliándolos si fuera necesario con un diagrama adicional para mostrar de forma correcta la estructura a seguir.

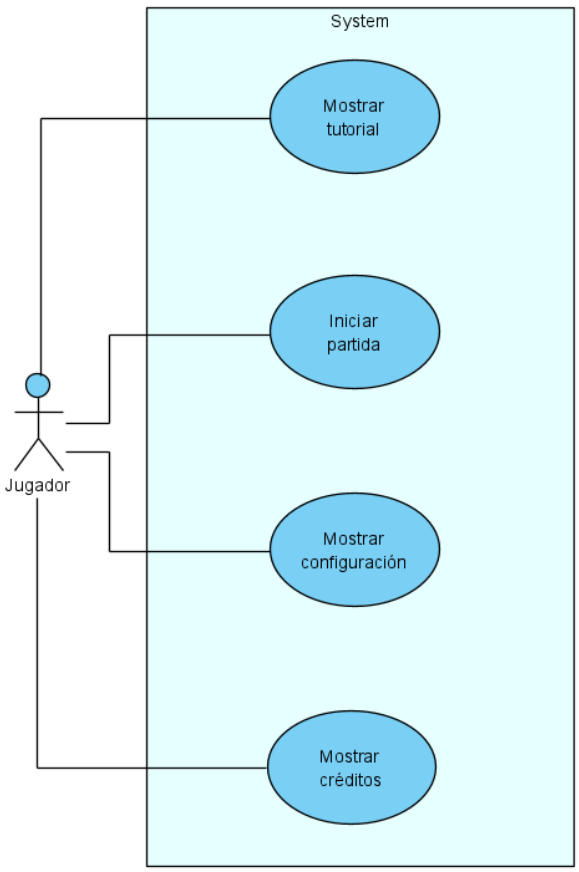


Figura 4.6 – Diagrama de casos de uso general de la aplicación.

4.2.2. Diagrama de caso de uso “iniciar partida”.

En la figura 4.7 se encuentra detallado el diagrama del caso de uso “iniciar partida” y el flujo general del sistema de juego. Las condiciones, actores y escenarios están especificados en la tabla 4.1.

Identificador CU01. Iniciar partida.	
Nombre	Iniciar partida.
Descripción	El jugador podrá iniciar la partida e interactuar con el juego.
Actor	El jugador.
Condiciones de entrada	El jugador inicia la partida. Puede existir o no una partida guardada anteriormente.
Condiciones de salida	Superar combate final o perder combate.
Escenario	1. El usuario presiona el botón “Iniciar partida”. 2. (Opcional) Se cargan los datos guardados. 3. El usuario puede desplazar al personaje a través del nivel. 4. (Opcional) El usuario habla con NPCs. 5. (Opcional) El usuario consulta las estadísticas de sus personajes. 6. (Opcional) El usuario muestra (y puede utilizar) los objetos disponibles. 7. (Opcional) El usuario toca a un enemigo y sus personajes luchan contra él. 8. (Opcional) El usuario pierde el combate y puede salir o cargar un punto de guardado anterior. 9. El usuario accede al combate final y sus personajes luchan contra el enemigo final. 9.1. El usuario gana el combate, supera el nivel. 9.2. El usuario pierde el combate, se ofrece la opción de cargar un punto de guardado anterior o salir del juego.

Tabla 4.1 - Detalle del caso de uso “iniciar partida”.

4.2.3. Diagrama de caso de uso “combatir”.

En la figura 4.8 se encuentra detallado el diagrama del caso de uso “combatir”, que deriva del anterior caso de uso de “iniciar partida”. Se realiza este diagrama para detallar el funcionamiento del combate, ya que supone una de las partes fundamentales del desarrollo del juego. Las condiciones, actores y escenarios están indicados en la tabla 4.2.

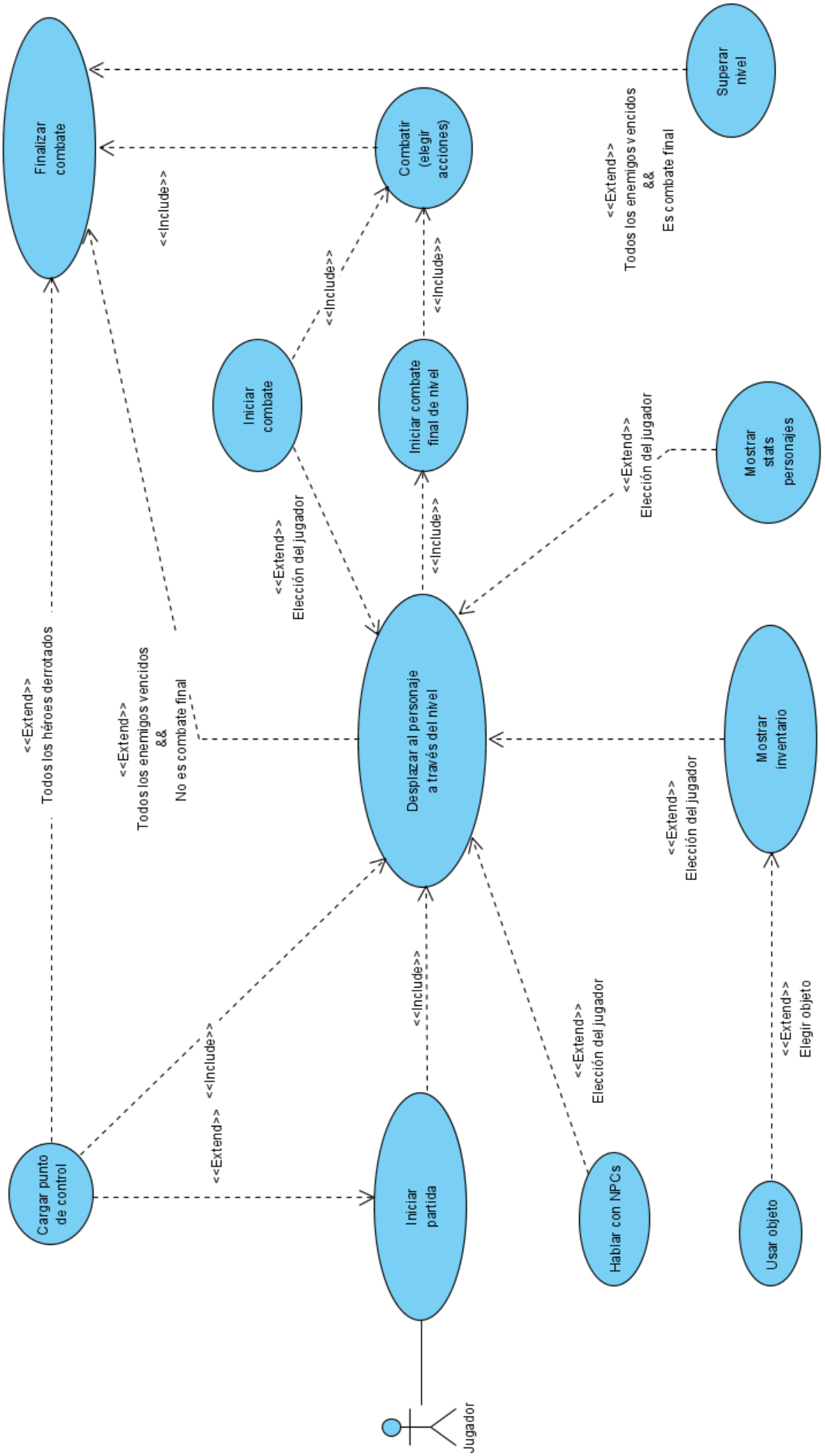


Figura 4.7 – Diagrama del caso de uso “iniciar partida”.

Identificador CU02. Combatir.	
Nombre	Combatir.
Descripción	El jugador podrá iniciar un combate y tomar decisiones interactuando con los personajes y enemigos.
Actor	El jugador.
Condiciones de entrada	El jugador toca a un enemigo o alcanza el combate final.
Condiciones de salida	Todos los enemigos son derrotados (victoria) o todos los héroes son vencidos (derrota).
Escenario	1. El usuario selecciona una opción del menú: 1.1. El usuario selecciona “ataque normal”, se salta a paso 4. 1.2. El usuario selecciona “ataque mágico”, se salta a paso 2. 1.3. El usuario selecciona “objeto”, se salta a paso 3. 1.4. El usuario selecciona “huida”, se pasa a paso 5. 2. Se selecciona un ataque mágico de la lista, se salta a paso 4. 3. Se selecciona un objeto de la lista. 4. Se selecciona un objetivo. 5. Se gestiona el combate. 6. Se almacenan las acciones del jugador. 7. Se generan los ataques enemigos. 8. Se procesan todos los ataques. 8.1. (Opcional) Si es ataque mágico, se genera el minijuego *challenge*. 9. Se calculan los daños. 10. Se comprueba la condición de victoria/derrota. 10.1. Si todos los enemigos son vencidos, es victoria, se vuelve al caso de uso “Desplazar al personaje a través del nivel”. 10.2. Si todos los héroes son vencidos, es derrota, se vuelve al caso de uso “Cargar punto de control”. 10.3. Si queda algún héroe y algún enemigo con vida, continúa el combate, se pasa al caso de uso inicial “Combatir”.

Tabla 4.2 – Detalle del caso de uso “combatir”.

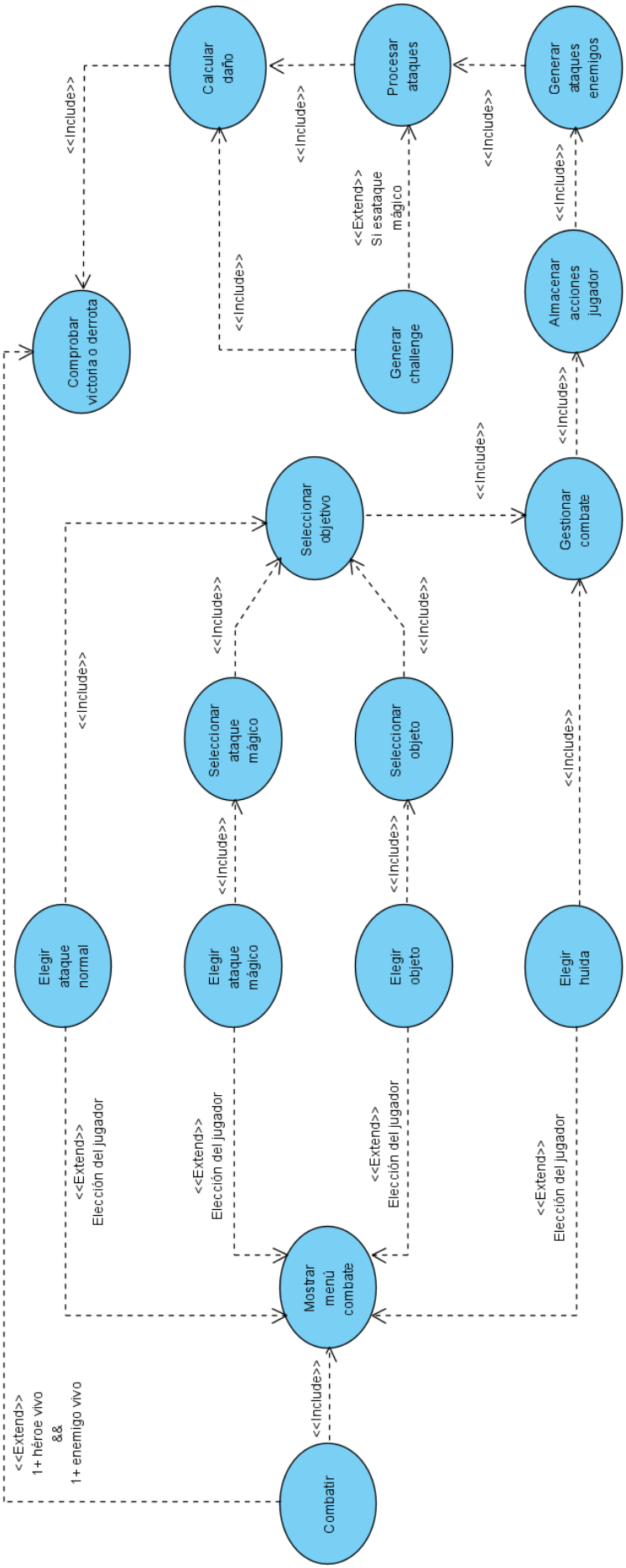


Figura 4.8 – Diagrama del caso de uso “combatir”.

4.2.4. Diagrama de caso de uso “Mostrar tutorial”.

En la figura 4.9 se encuentra detallado el diagrama del caso de uso “mostrar tutorial”. La función de este caso de uso es bastante sencilla, ya que sólo debe mostrar un breve tutorial del juego al usuario. Las condiciones, actores y escenarios están indicados en la tabla 4.3.

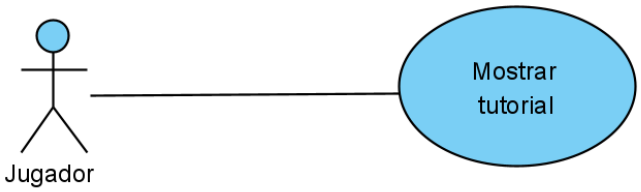


Figura 4.9 – Diagrama del caso de uso “mostrar tutorial”.

Identificador CU03. Mostrar tutorial.	
Nombre	Mostrar tutorial.
Descripción	El jugador puede acceder a un tutorial que explica cómo jugar.
Actor	El jugador.
Condiciones de entrada	El jugador pulsa el botón “Tutorial” del menú principal del título.
Condiciones de salida	El jugador pulsa sobre el botón “Finalizar tutorial”.
Escenario	1. El jugador pulsa el botón “Tutorial” del menú principal. 2. El jugador navega a través del tutorial. 3. El jugador puede ir hacia adelante o atrás en el tutorial. 4. El jugador pulsa el botón “Finalizar tutorial” y vuelve al menú principal.

Tabla 4.3 – Detalle del caso de uso “mostrar tutorial”.

4.2.5. Diagrama de caso de uso “Mostrar créditos”.

En la figura 4.10 se detalla el diagrama del caso de uso “mostrar créditos”. El cometido este caso de uso es mostrar la información del creador del juego, así como de indicar la autoría de los diferentes recursos utilizados en el proyecto. Las condiciones, actores y escenarios están indicados en la tabla 4.4.

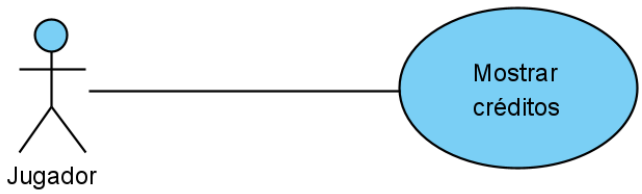


Figura 4.10 – Diagrama del caso de uso “mostrar créditos”.

Identificador CU04. Mostrar créditos.	
Nombre	Mostrar créditos.
Descripción	El jugador puede acceder a la información que muestra el nombre del desarrollador y autores de los diferentes recursos.
Actor	El jugador.
Condiciones de entrada	El jugador pulsa el botón de “Créditos” del menú principal del título.
Condiciones de salida	El jugador pulsa sobre el botón de “atrás”.
Escenario	1. El jugador pulsa el botón “Créditos” del menú principal. 2. El jugador pulsa el botón “atrás” y vuelve al menú principal.

Tabla 4.4 – Detalle del caso de uso “mostrar créditos”.

4.2.6. Diagrama del caso de uso “mostrar configuración”.

En la figura 4.11 se detalla el diagrama del caso de uso “mostrar configuración”. La función de este caso de uso es permitir al jugador poder visualizar y alterar diferentes parámetros de configuración de su partida. Las condiciones, actores y escenarios están indicados en la tabla 4.5.

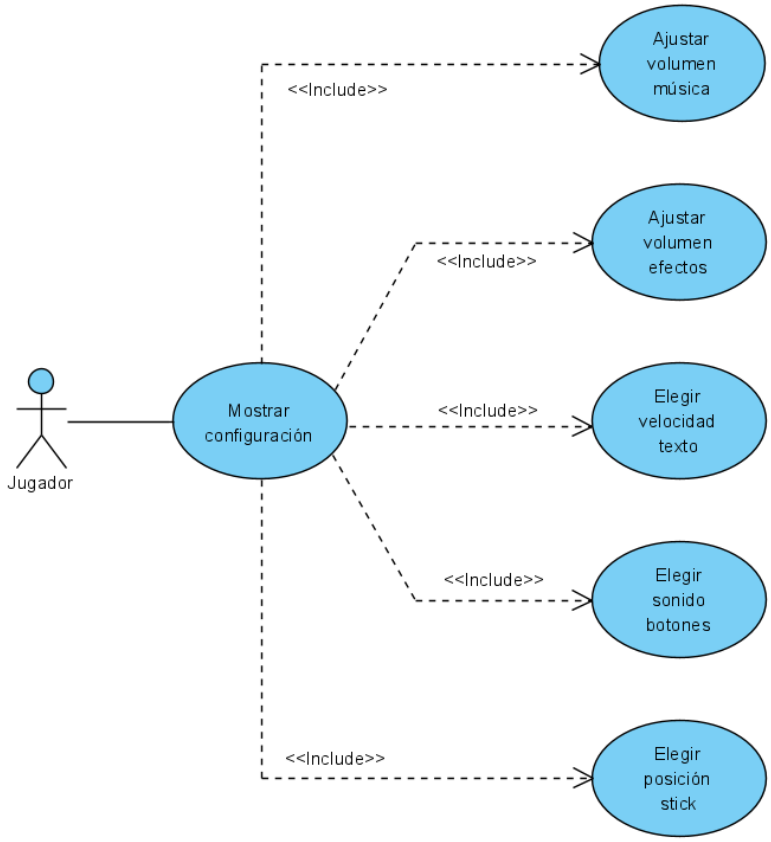


Figura 4.11 – Diagrama del caso de uso “mostrar configuración”.

Identificador CU05. Mostrar configuración.	
Nombre	Mostrar configuración.
Descripción	El jugador puede configurar algunos parámetros de la partida.
Actor	El jugador.
Condiciones de entrada	El jugador pulsa el botón “Configuración” del menú principal del título.
Condiciones de salida	El jugador pulsa sobre el botón “atrás”.
Escenario	1. El jugador pulsa sobre el botón “Configuración” del menú principal. 2. (Opcional) El jugador ajusta el volumen de la música. 3. (Opcional) El jugador cambia el volumen de los efectos. 4. (Opcional) El jugador elige la velocidad del texto. 5. (Opcional) El jugador escoge si desea escuchar el sonido al pulsar cada botón. 6. (Opcional) El jugador puede elegir la posición del *joystick* de desplazamiento del personaje. 7. El jugador pulsa el botón “atrás” y vuelve al menú principal.

Tabla 4.5 – Detalle del caso de uso “mostrar configuración”.

4.3. Diagramas de clases.

En este apartado se encuentran los diferentes diagramas de clases, que especifican los atributos, métodos y relaciones entre las diferentes entidades que conforman el proyecto.

Este desarrollo emplea la arquitectura basada en componentes, que permite su reutilización y aporta cierta independencia entre los diferentes componentes que forman el sistema. Otra particularidad de Unity, es que las clases suelen heredar de “MonoBehaviour”, permitiendo su uso como “componentes” y su integración directa en los *GameObjects*.

Para empezar, varias de las clases principales siguen el patrón *singleton*. De esta forma, este tipo de entidades sólo podrán instanciarse una única vez y serán accesibles por el resto de los componentes.

Este patrón resulta muy útil para mantener información entre las diferentes escenas y agilizar su tratamiento y acceso por parte de diferentes clases.

De esta manera, una de las clases principales y con mayor utilidad en diferentes funcionalidades del proyecto es la denominada como

"GameController". Esta clase es dependiente de varias, ya que almacena diferente información y parámetros de la partida y facilita la carga y transmisión de información entre escenas. También se encarga de gestionar los datos de la partida del jugador, que se guardarán en memoria a través de las clases "PlayerData" y "SaveController".

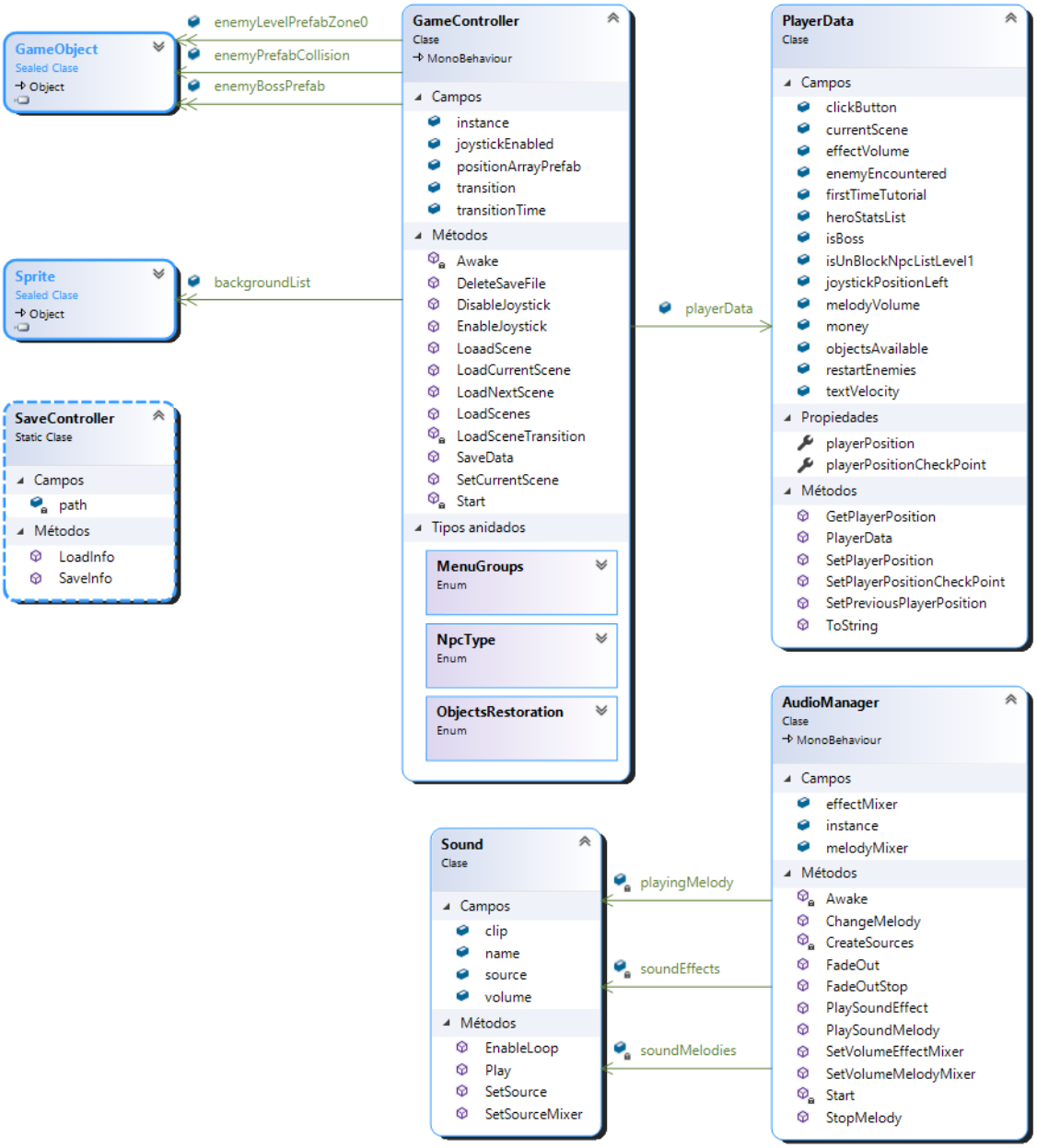


Figura 4.12 – Detalle de clases con patrón *singleton* y sus dependencias.

Otra de las clases que aprovechan el patrón *singleton* es la encargada de administrar y gestionar el audio, denominada "AudioManager". Esta clase es dependiente de "Sound", ya que todas las melodías y efectos deben estar contenidos en un objeto de esta clase. "AudioManager" se encargará de ofrecer disponibilidad a través de todo el proyecto para poder reproducir sonido de forma fácil y rápida.

Por otro lado, se encuentran las clases que administran los diálogos de los personajes. Estos objetos se instancian y son gestionados por la clase “DialogueHistoryController”, que permite visualizar las diferentes líneas de diálogos.
La clase “MainMenuController” se encarga de administrar y permitir la navegación en el menú principal. Adicionalmente permite guardar la configuración del jugador a través de la clase “GameController”.

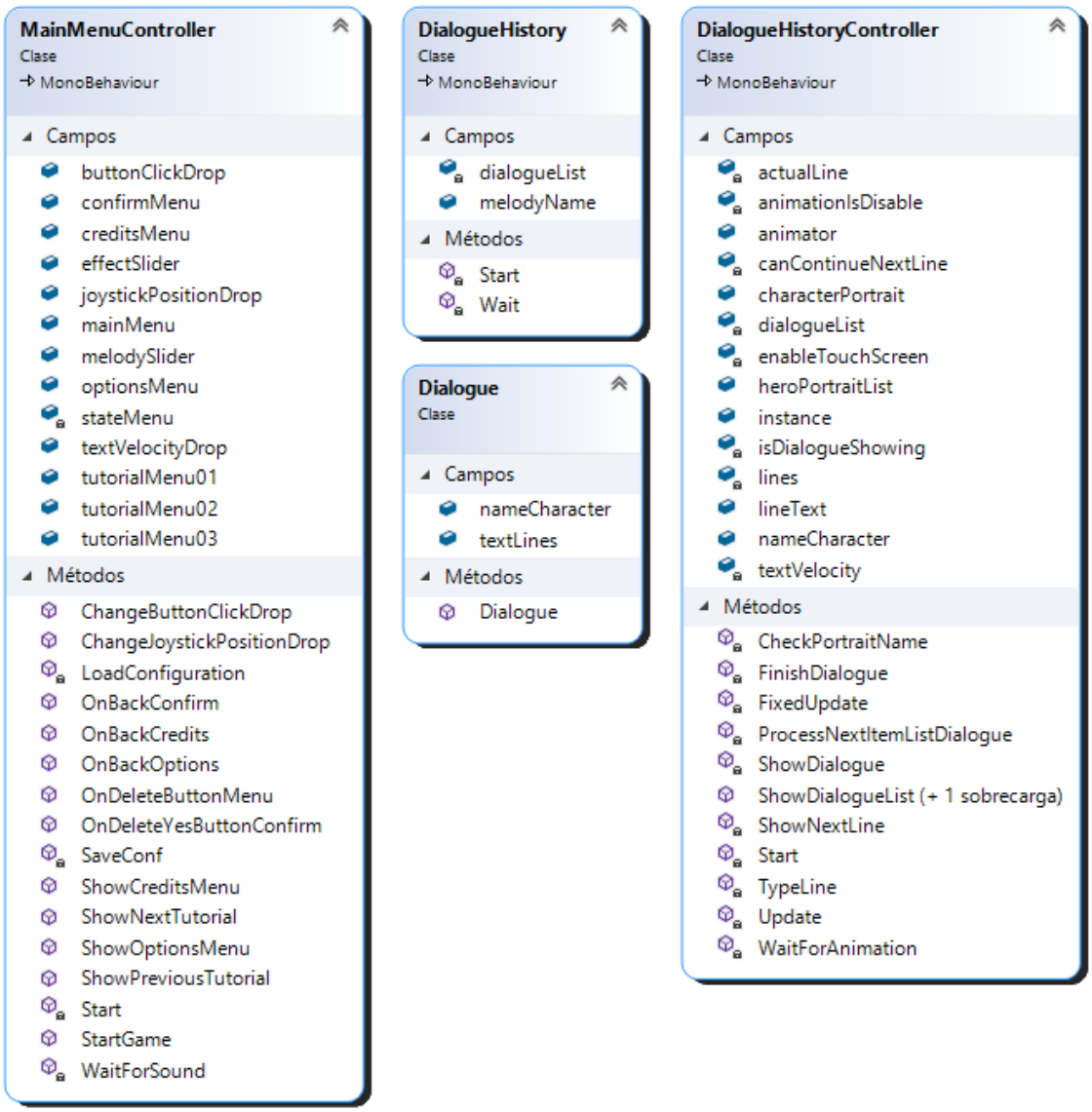


Figura 4.13 – Detalle de la clase de menú principal y de gestión de diálogos.

También hay una serie de clases que se encargarán de gestionar al personaje principal, los enemigos, la cámara y los diferentes personajes no controlables por el jugador. Además, se cuenta con algunas clases encargadas de generar diferentes partes del menú de combate, como los botones de selección de enemigo, hechizos y objetos.

“LevelController” es la clase que se encarga de gestionar el nivel principal. Entre otras funciones, se encarga de generar los enemigos presentes en el mapa principal, así como de comprobar el progreso del

jugador y desbloquear las diferentes zonas o personajes que permitan progresar en el juego.

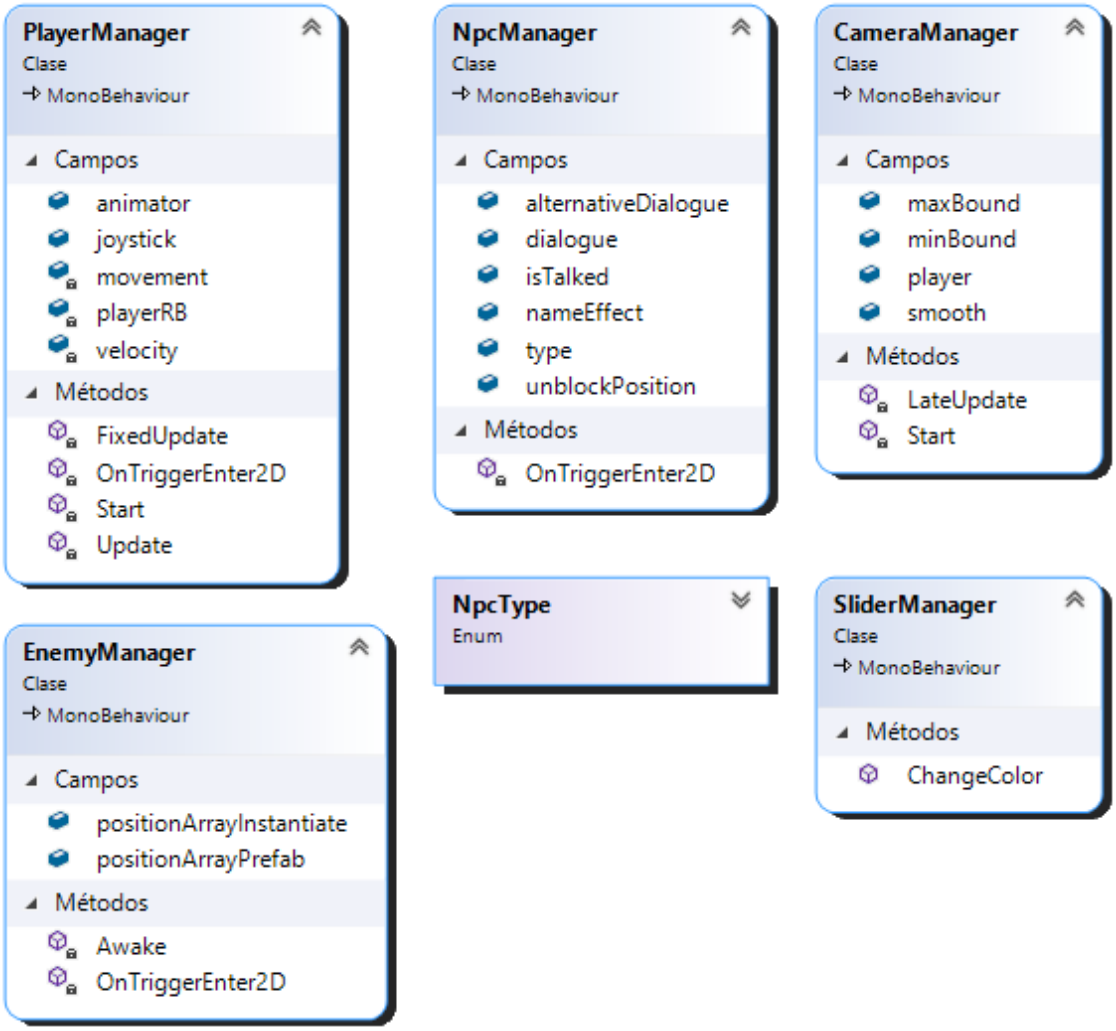


Figura 4.14 – Detalle de clases de gestión de personajes y otros elementos.

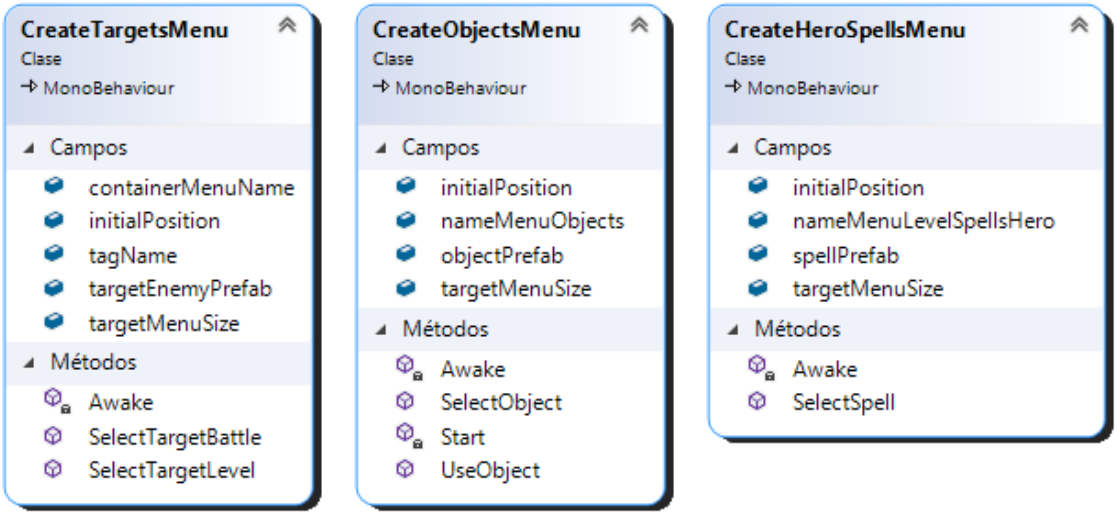


Figura 4.15 – Detalle de clases de creación de diferentes secciones del menú de combate.

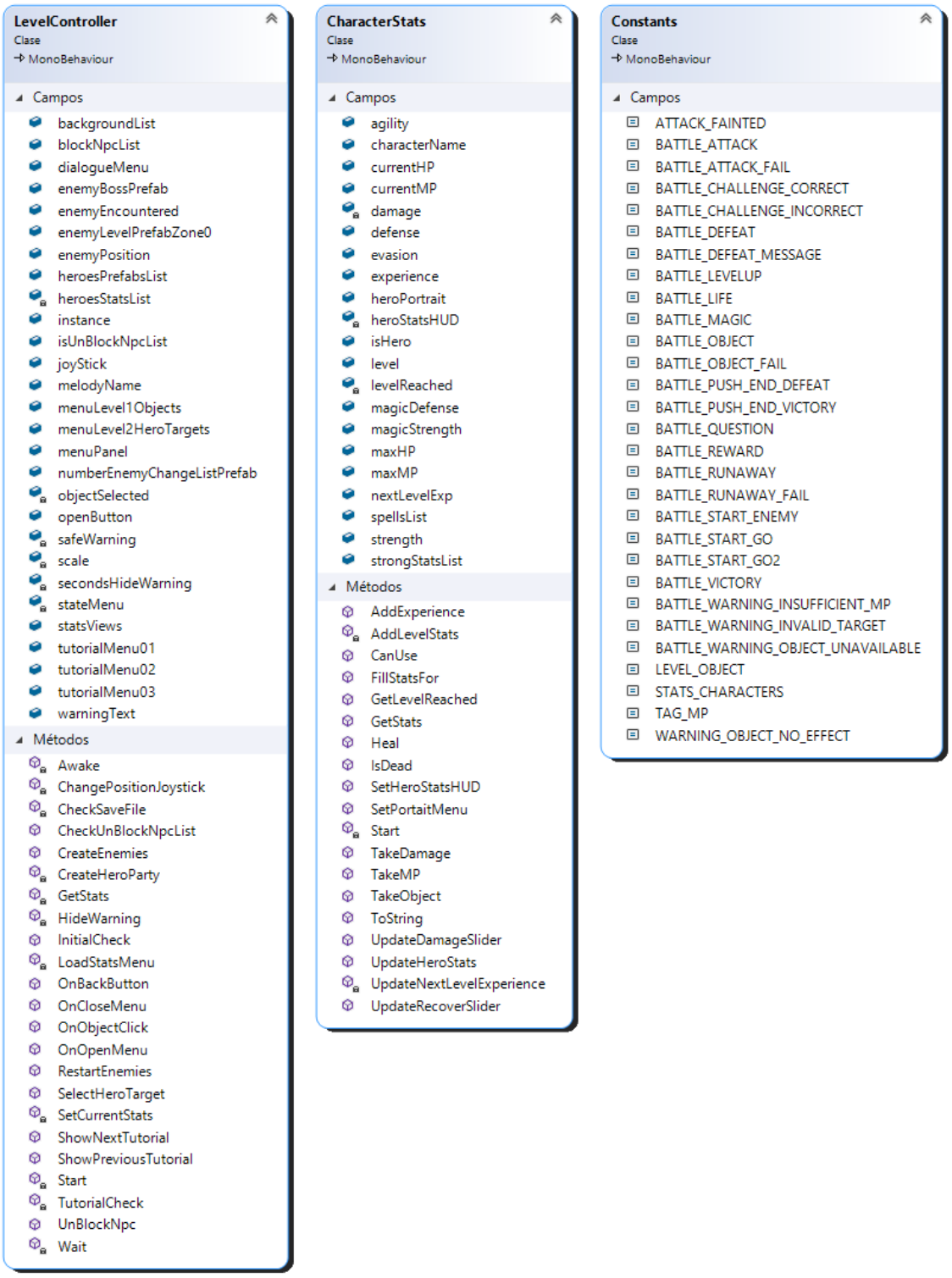


Figura 4.16 – Detalle de clases de gestión nivel principal y estadísticas de los personajes.

Otra de las clases con mayor funcionalidad es “BattleManager”. Esta clase es dependiente de varias, ya que se encarga de gestionar todos los objetos involucrados en el combate. Algunas de las clases involucradas son “Attack” (que se encarga de los cálculos de daño),

"BattleHUD" (que gestiona los diferentes componentes del menú), "Challenge" (para la gestión del minijuego) y "Results" (para el resumen de la batalla).

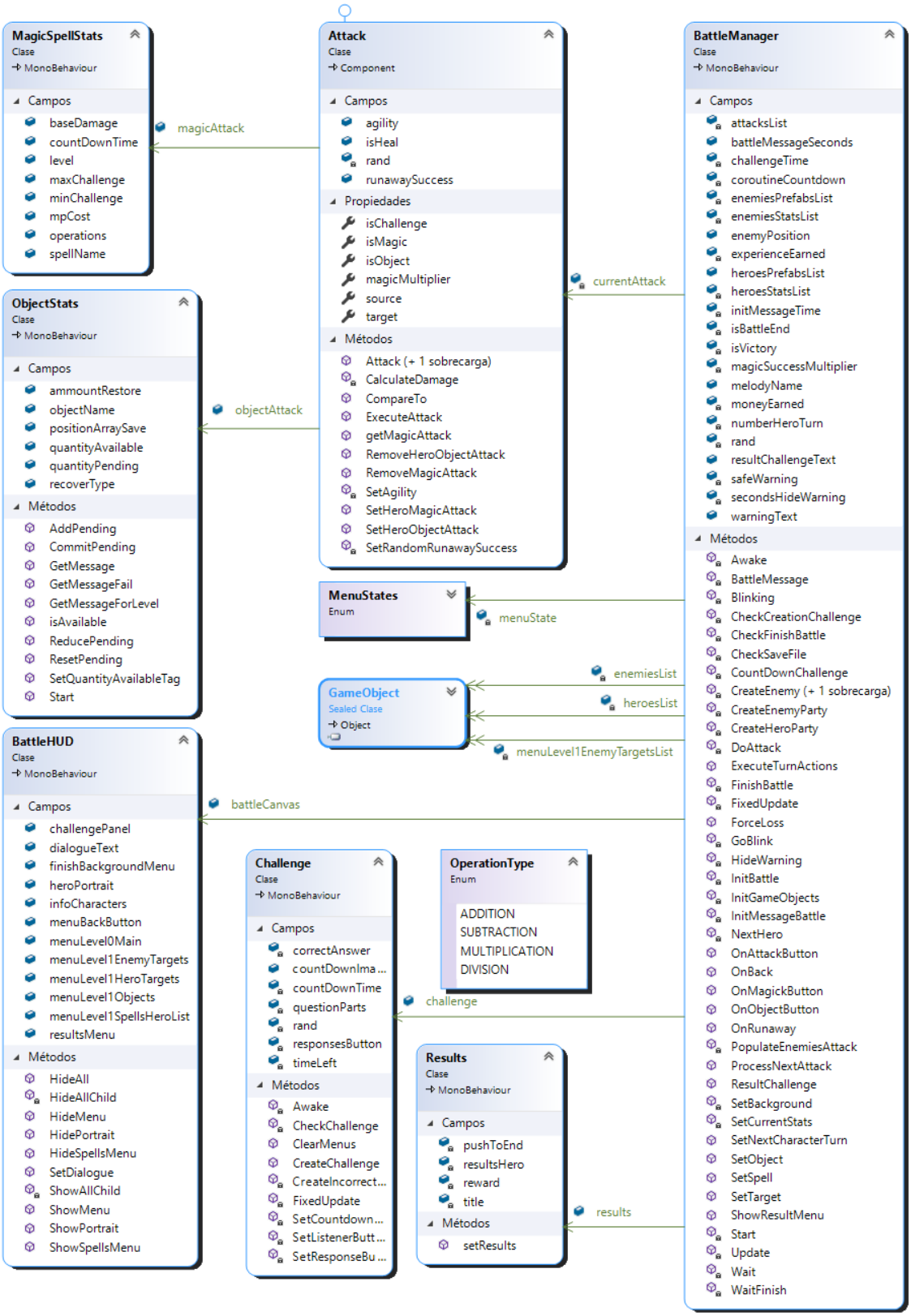


Figura 4.17 – Detalle de la clases gestora de los combates y sus dependencias.

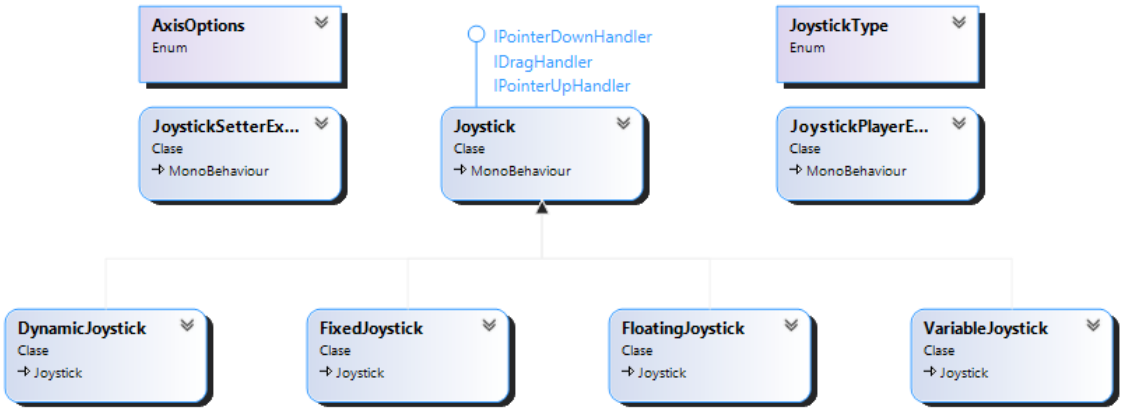


Figura 4.18 – Detalle de las clases del plugin externo "Joystick Pack" [40].

Por último, también se especifican las clases relativas al *plugin* "Joystick Pack" [40], simplemente a modo ilustrativo, ya que no se ha realizado ningún cambio o adaptación en el código.

4.4. Diagramas de secuencia.

En esta sección se pueden encontrar los diferentes diagramas de secuencia. Estos diagramas muestran la comunicación e intercambio de diferentes mensajes entre las distintas clases del sistema y se pueden apreciar las clases implicadas.

4.4.1. Diagrama de secuencia para el caso de uso "Mostrar tutorial".

En la figura 4.19 se puede apreciar el diagrama de secuencia del caso de uso "Iniciar partida". En este diagrama se muestra la secuencia de mensajes que se producen cuando el comienza el juego y en el trascurso del nivel principal.

4.4.2. Diagrama de secuencia para el caso de uso "Combatir".

En la figura 4.20 se puede encontrar el diagrama de secuencia del caso de uso "Combatir". En este diagrama se muestra la secuencia de mensajes que se producen cuando el usuario accede a un combate.

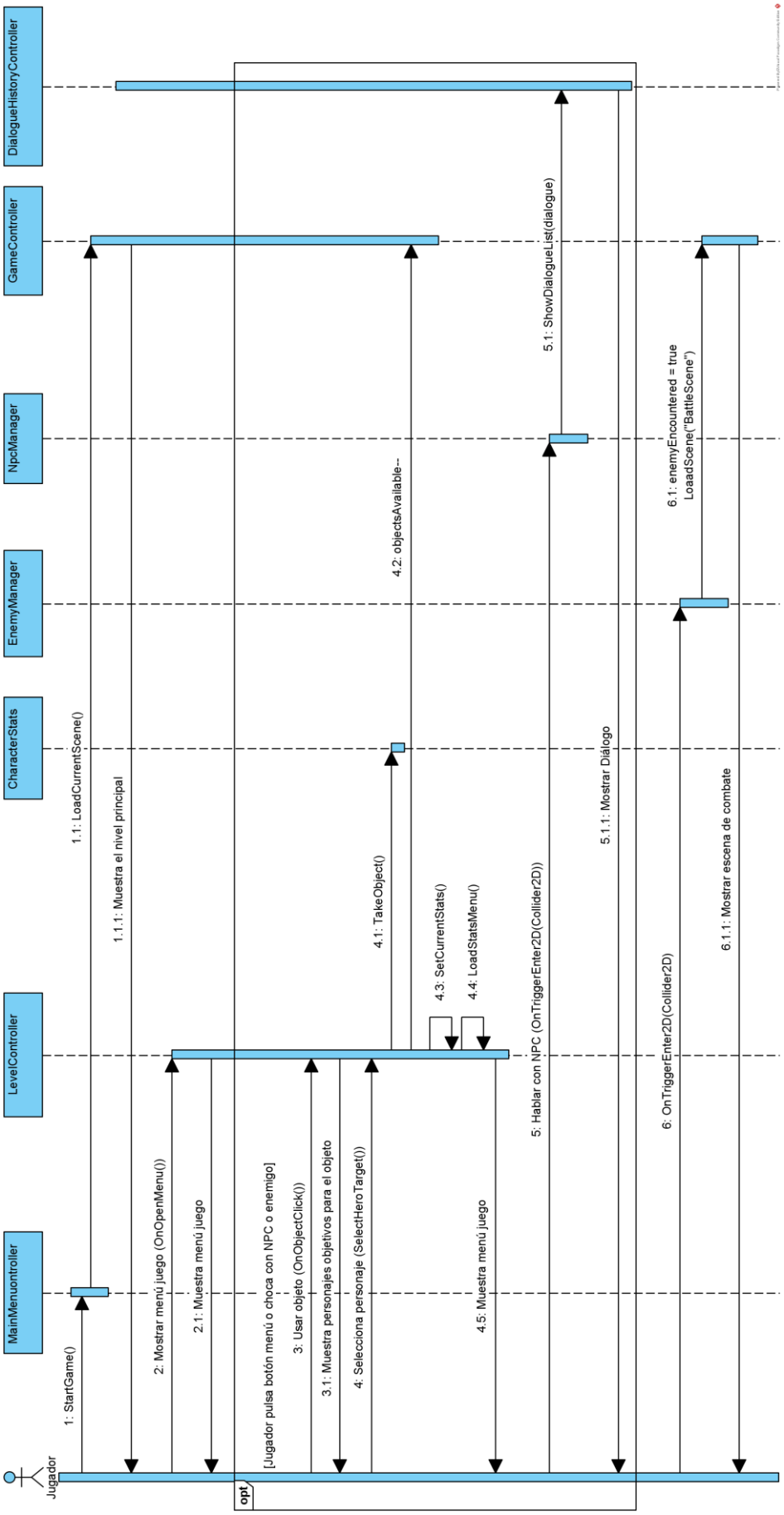


Figura 4.19 – Diagrama de secuencia para el caso de uso “iniciar partida”.

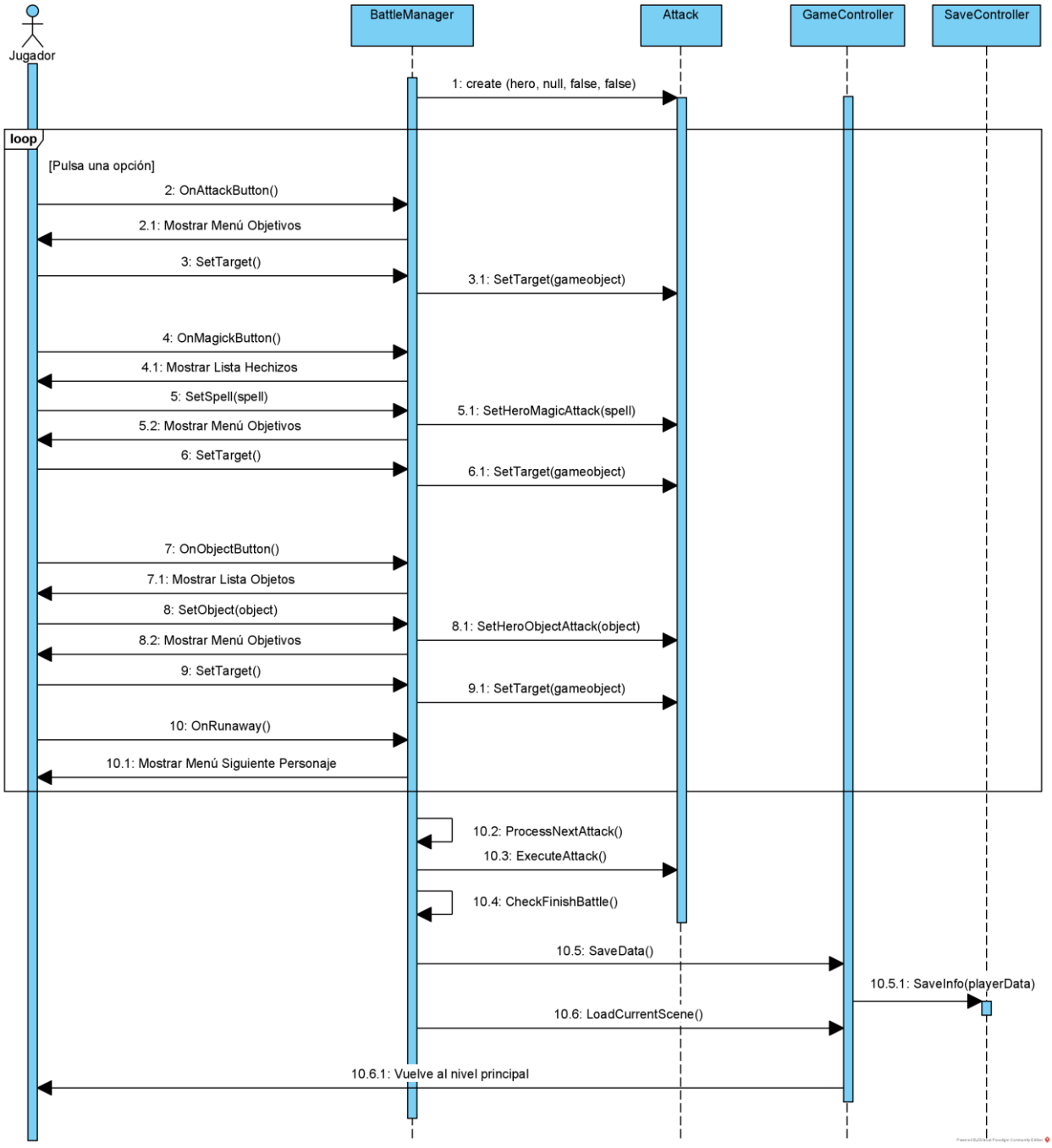


Figura 4.20 - Diagrama de secuencia para el caso de uso “combatir”.

4.4.3. Diagrama de secuencia para el caso de uso “Mostrar tutorial”.

En la figura 4.21 se puede apreciar el diagrama de secuencia del caso de uso “Mostrar tutorial”. En este diagrama se muestra la secuencia de mensajes que se producen cuando el usuario desea visualizar el tutorial de juego.

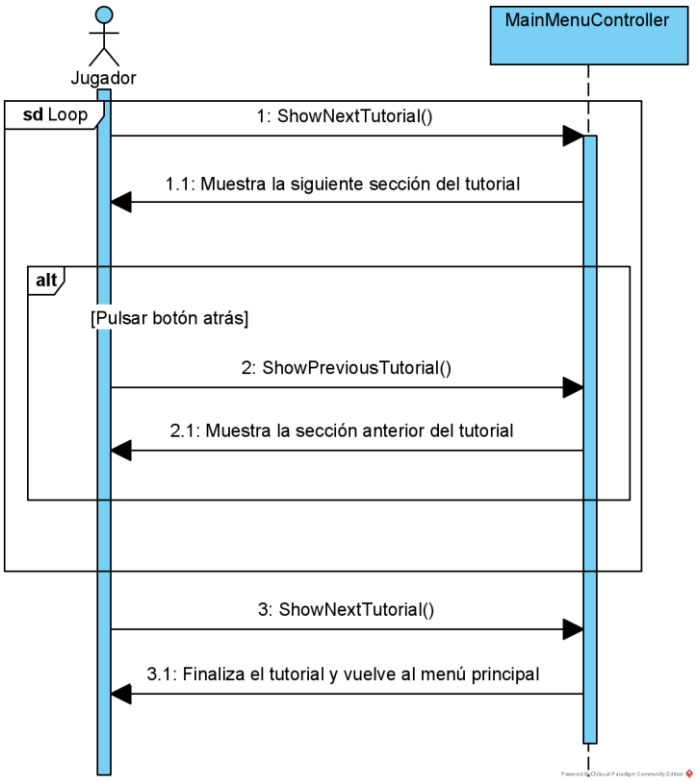


Figura 4.21 – Diagrama de secuencia para el caso de uso “Mostrar tutorial”.

4.4.4. Diagrama de secuencia para el caso de uso “Mostrar créditos”.

En la figura 4.22 se puede encontrar el diagrama de secuencia del caso de uso “Mostrar créditos”. En este diagrama se muestra la secuencia de mensajes que se producen cuando el usuario desea ver ña información de los autores del juego.

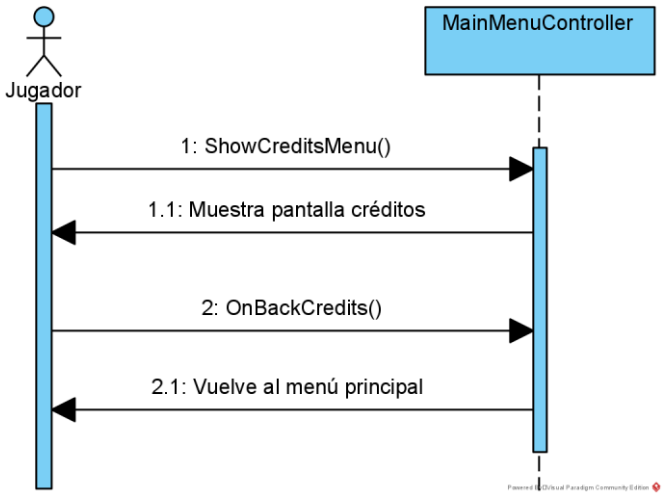


Figura 4.22 – Diagrama de secuencia para el caso de uso “Mostrar créditos”.

4.4.5. Diagrama de secuencia para el caso de uso “Mostrar configuración”.

En la figura 4.23 se puede consultar el diagrama de secuencia del caso de uso “Mostrar configuración”. En este diagrama se muestra la secuencia de mensajes que se producen cuando el usuario desea cambiar alguna opción de la configuración del juego.

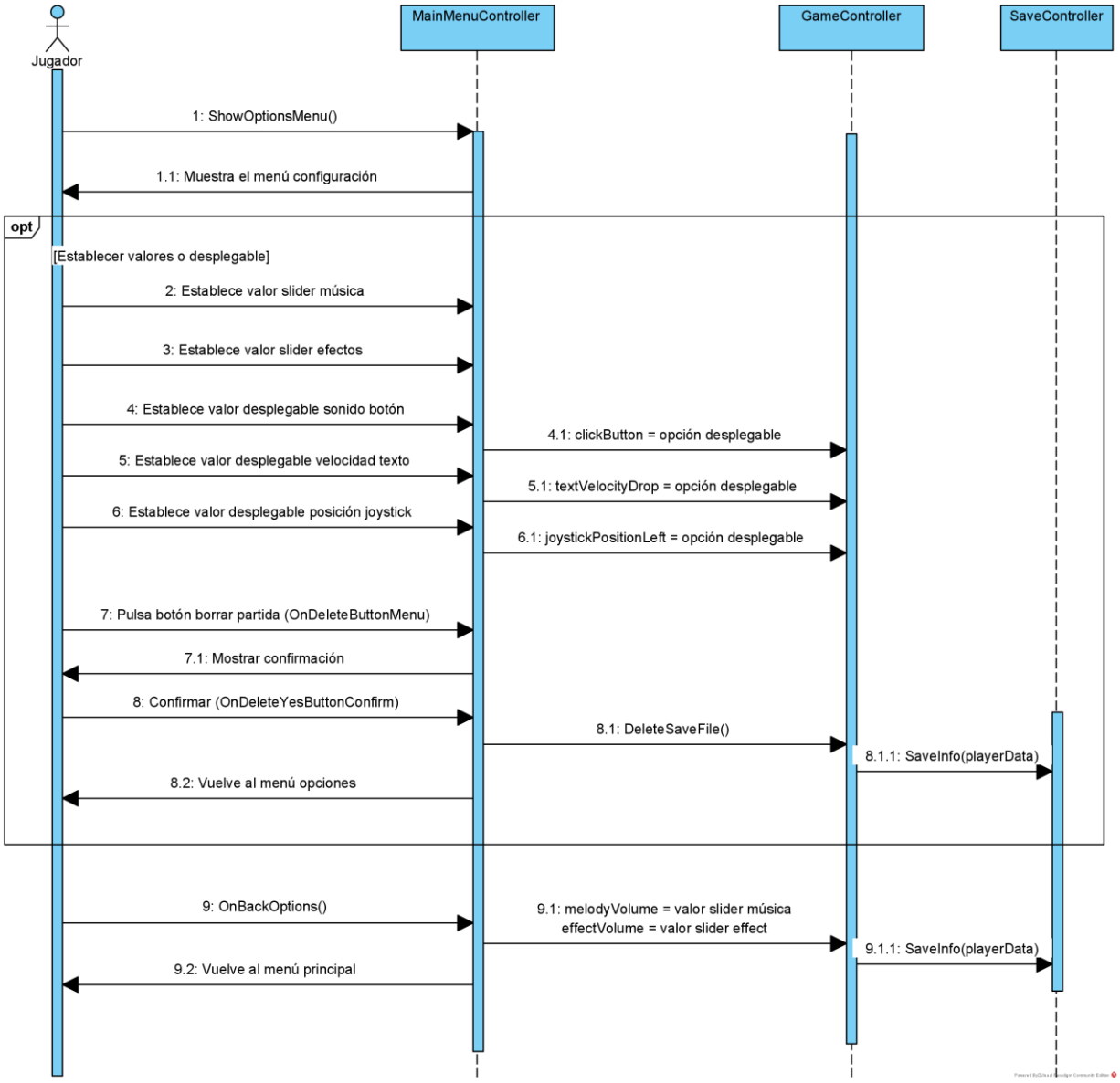


Figura 4.23 – Diagrama de secuencia para el caso de uso “Mostrar configuración”.

4.5. Diseño de la interfaz.

4.5.1. Metáforas.

Una metáfora es una figura retórica que cambia el sentido de una determinada palabra o frase, pero manteniendo una relación de analogía o semejanza.

Este concepto aplicado a las interfaces de las aplicaciones o juegos, tiene el cometido de que el usuario pueda identificar el funcionamiento de diferentes botones de forma completamente visual [47].

Figura 4.24 – Captura del menú de combate.

Este tipo de recurso se ha utilizado principalmente en el menú de combate. De esta manera, los iconos son los suficientemente descriptivos para que el usuario pueda intuir su funcionalidad. Otro ejemplo sería a la hora de navegar por las diferentes partes del tutorial, utilizando los botones de flechas para avanzar o retroceder.

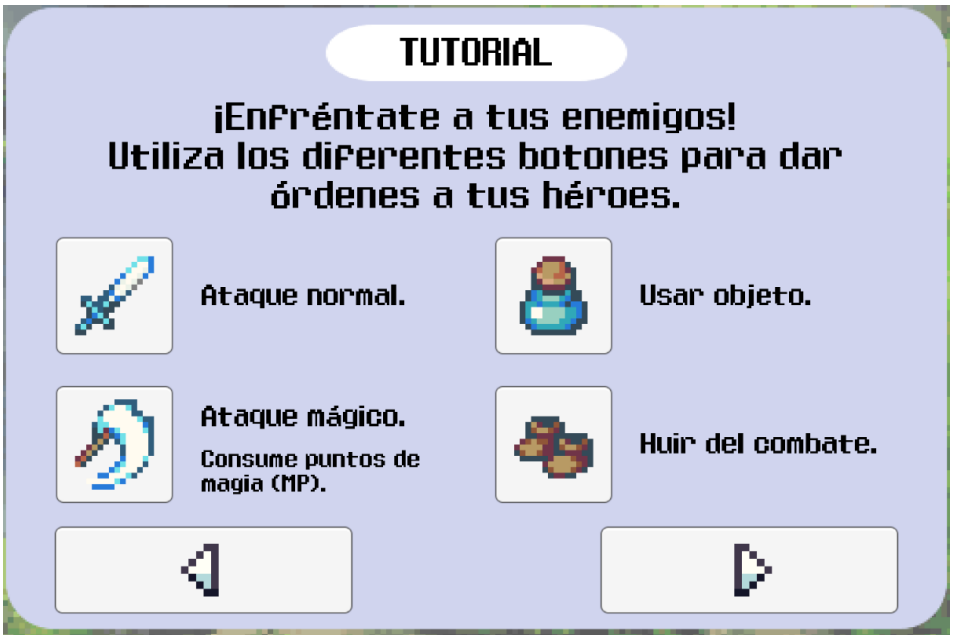


Figura 4.25 – Captura de pantalla del tutorial.

4.5.2. Bocetos y prototipos de la interfaz.

Para realizar una primera aproximación al diseño de la interfaz de las diferentes partes del juego, la cual se puede apreciar en la figura 4.26, se utilizó la herramienta Balsamiq Wireframes [35]. Esta aplicación resulta muy útil para modelar prototipos de interfaces de usuario y al contar con elementos enfocados a los dispositivos móviles, es posible hacerse una idea bastante aproximada de cómo lucirán los elementos en pantalla.

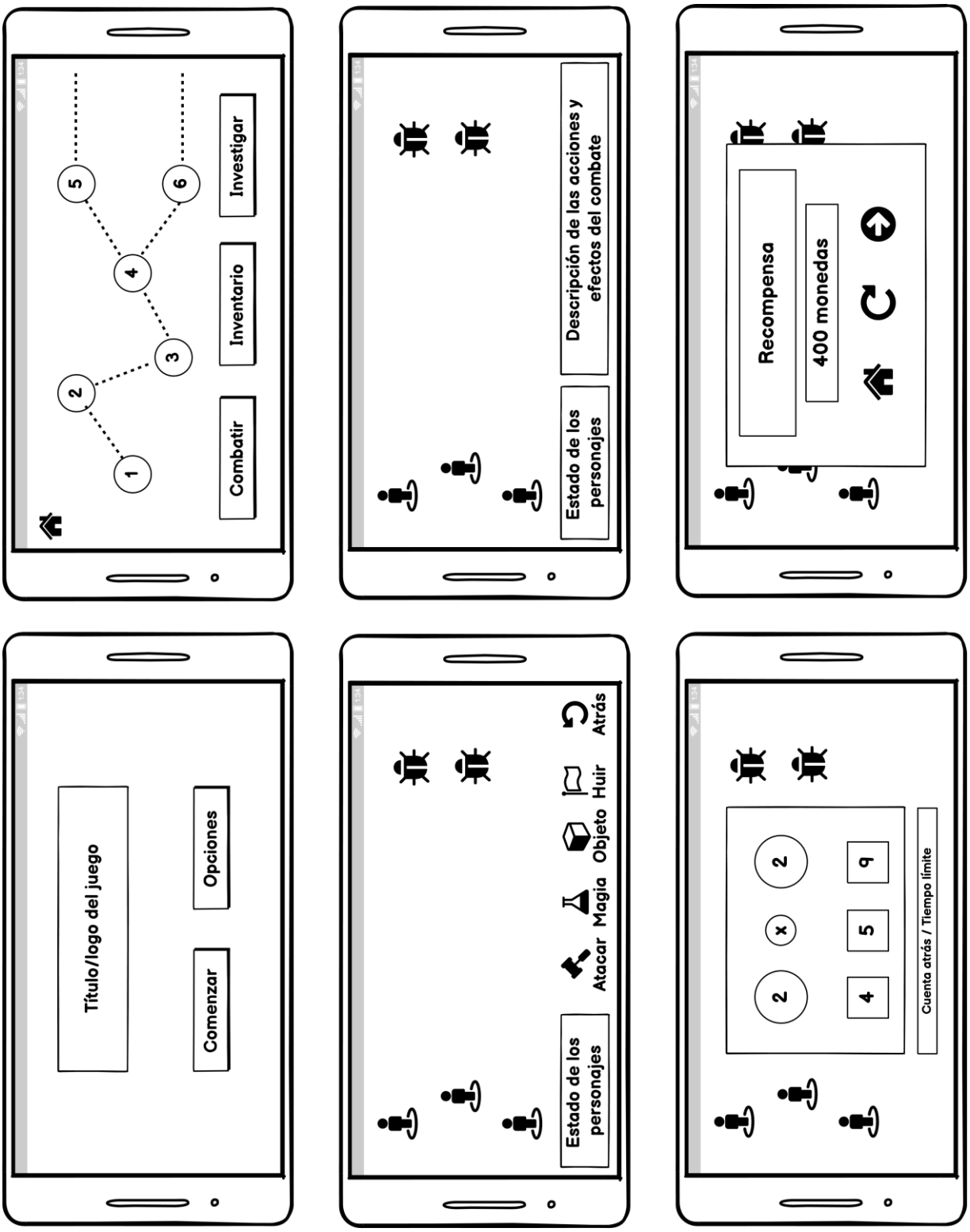


Figura 4.26 – Prototipos y bocetos iniciales de la posible interfaz de juego.

4.6. Implementación y aspectos destacables.

Durante el desarrollo de este proyecto ha sido necesario enfrentarse a ciertas cuestiones, tanto a nivel técnico como jugable, derivadas de algunas decisiones de diseño y de los objetivos que se pretendían llevar a cabo.

4.6.1. Resolución y *pixel art*.

El estilo artístico escogido para el juego fue el denominado pixel art. Se debe prestar especial atención al correcto uso que se realice de los *sprites*, ya que aunque permiten ser escalados fácilmente, se pueden producir algunos defectos que estropean el recurso base.

En primer lugar, se debe prestar atención a que el tamaño de los píxeles es homogéneo en la totalidad del *sprite* que se esté utilizando, así como de una correcta configuración para evitar posibles problemas como el *blur* (difuminado).

También se tuvo en cuenta la configuración a la hora de escalar la escena de juego. El método utilizado es el de escalar según el tamaño de pantalla. La decisión de esta configuración fue debida a la plataforma final de juego, los dispositivos Android, que cuentan con una amplia gama de tamaños de pantalla, así como de distintas resoluciones y relaciones de aspecto.

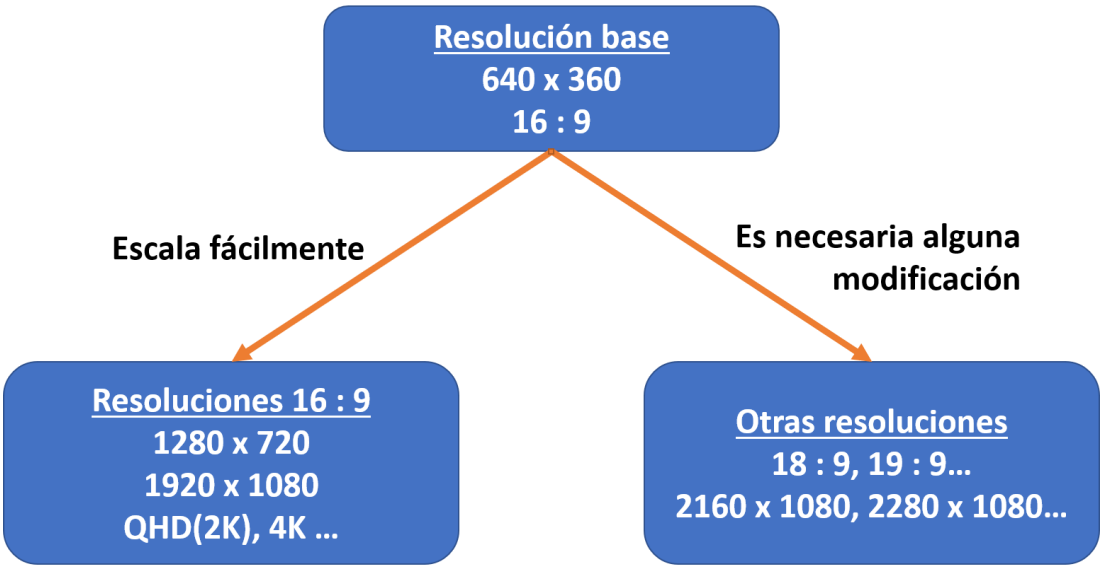


Figura 4.27 – Diagrama de resoluciones y posibilidades a la hora de escalar.

Se escogió una resolución base de 640 x 360, porque permitía escalar de forma sencilla a otras resoluciones mayores con relación de aspecto 16:9. No obstante, se tuvo que tener en cuenta otra serie de resoluciones más panorámicas resultantes de relaciones de aspecto tales como 19:9 o 19:8, que se han vuelto bastante populares en los últimos años entre diferentes fabricantes de dispositivos.

De esta forma, para evitar que se produjeran algunos defectos gráficos, como las conocidas “bandas negras” a los lados de la pantalla, se optó por modificar los límites definidos por la cámara que sigue al jugador.

Así, en el caso de que se use un dispositivo con este tipo de resoluciones más panorámicas, no habrá ningún problema porque la cámara no llegará hasta el límite final del *canvas*, aprovechando de esta forma todo el ancho de la pantalla.

4.6.2. Patrón de diseño *singleton*.

Como ya se ha comentado en el apartado anterior de diagramas de clases, se ha hecho uso del patrón de diseño *singleton* para la constitución de algunos de los objetos presentes en el proyecto.

Este patrón de diseño se basa en tener siempre disponible una instancia de la clase donde se implemente y mientras dure la ejecución de la aplicación. Este hecho es especialmente útil para ciertas tareas y persistencia de la información, así como el paso de información entre diferentes entidades.

El uso que se ha hecho de este patrón en el proyecto ha recaído principalmente en la clases “GameController” y “AudioManager”.

Por un lado, “GameController” se encarga de almacenar el estado del jugador, sus estadísticas y configuración. También se encarga de gestionar la transición entre las diferentes escenas del proyecto y del sistema de guardado, tanto la lectura como la escritura del fichero de guardado.

Por otro lado, la clase “AudioManager” puede ser llamada por cualquier otra clase u objeto y mediante el paso de un atributo *string* o cadena, se encargará de buscar el recurso (melodía o efecto) y reproducirlo sin tener que realizar ninguna acción adicional.

4.6.3. Uso de corrutinas.

Las corrutinas se pueden emplear de muchas maneras diferentes según las necesidades que se quieran cubrir. Permiten ejecutar una parte del código y que el resto se reanude de forma posterior, bien porque tenga que esperar a otra función o simplemente realizar una espera determinada.

Una de estas funciones es empleada a la hora de lograr que alguna acción en el código sea visible en tiempo real de juego, es decir, que se pueda apreciar en los distintos *frames*. Por ejemplo, en el uso de animaciones, como el *fade* (o efecto de desvanecimiento) entre las diferentes escenas del juego [48].

El uso de corrutinas se ha empleado principalmente en la gestión del combate por turnos. La forma de gestionar cada turno se realiza a partir de una lista de ataques, que se ordenan según el parámetro de agilidad de cada personaje. A continuación, se recorre la lista ejecutando cada ataque. En lugar de usar una corrutina general para el proceso, se lanza la misma para la gestión de cada ataque. De esta manera, se puede disponer de tiempo real para que el jugador pueda ser informado de las acciones a través de mensajes y avisos por pantalla y también para disponer del tiempo necesario para visualizar las diferentes animaciones de combate.

También se han empleado a la hora de gestionar la cuenta atrás del *challenge* o minijuego de preguntas dentro del combate. De esta forma, el tiempo de cuenta atrás viene definido por el tipo de hechizo que el jugador haya elegido y la corrutina implementa la espera de ese tiempo límite si el jugador no responde a la pregunta. En el caso de que el usuario responda a la pregunta antes de que acabe esta cuenta atrás, la corrutina se cancela, ya que se fuerza la comprobación del resultado del desafío cuando el usuario ha pulsado una de las opciones, independientemente de si es correcta o no.

5. Diseño de niveles.

5.1. Planteamiento general.

El proyecto se ha centrado en la creación de un nivel plenamente funcional y que puede ser completado de principio a fin. El objetivo elemental del juego se lleva a cabo a través de la exploración de un nivel principal.

De esta manera, el jugador puede desplazar a su personaje a través del nivel, en el que podrá visitar diferentes emplazamientos y hablar con otros personajes para progresar en la historia. Además, otra parte importante de la jugabilidad radica en la posibilidad de entablar combate contra enemigos mientras se realiza la exploración del nivel principal. En los siguientes puntos de profundiza en estos dos sistemas de jugabilidad.

Figura 5.1 – Vista general del mapa del nivel principal.

5.2. Nivel principal.

En primer lugar, se cuenta con un nivel principal donde el jugador puede desplazar a su personaje a través de diferentes zonas. A lo largo de las distintas localizaciones, el usuario se encontrará con diferentes enemigos con los que podrá entablar un combate si el personaje que controla entra en contacto con ellos.

Adicionalmente, existe la posibilidad de poder investigar la zona y encontrar pistas acerca del objetivo principal a cumplir hablando con los distintos personajes. También se deben cumplir unos determinados objetivos para poder acceder a nuevos lugares y poder progresar en la historia principal.

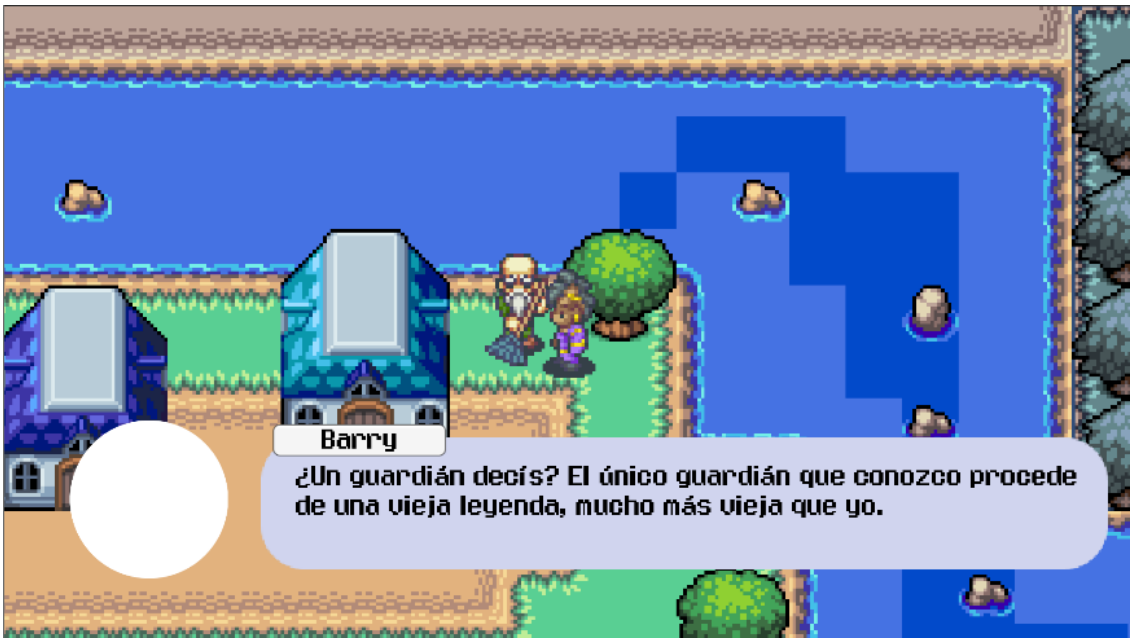


Figura 5.2 – Hablar con los habitantes será crucial para avanzar en la historia.

De forma general, al final de cada misión o nivel se debe hacer frente a un enfrentamiento contra un jefe de zona. Este enemigo tendrá una fuerza superior al resto de contrincantes del lugar, por lo que será necesario que los héroes dispongan de un buen nivel para poder hacerle frente con soltura.

Figura 5.3 – Vista del menú de estadísticas, uso de objetos y opciones de partida.

Por último, se ofrece la posibilidad de acceder a un menú que permite consultar las estadísticas y el estado de los personajes. Además, se podrá hacer uso de los diferentes objetos de recuperación, guardar la partida en cualquier momento o bien volver al menú principal del juego.

5.3. Combates.

El combate contra diferentes enemigos es otra parte fundamental de la jugabilidad y desarrollo de este proyecto. De esta manera, se permite al jugador poder fortalecer a sus personajes mediante el aumento de su nivel y estadísticas. Un aspecto a destacar es que todos los combates son aleatorios y cada enfrentamiento implica a un mínimo de un enemigo y a un máximo de tres al mismo tiempo en un mismo combate.

Otro detalle que se ha tenido en cuenta es la distribución de los diferentes enfrentamientos, así como de los enemigos que pueden hacer acto de presencia. De esta manera, dependiendo de la zona donde se encuentre el jugador, es posible que aparezcan enemigos más o menos fuertes, a pesar del factor de aleatoriedad. Es decir, los enemigos se generan al azar, pero teniendo en cuenta unos determinados parámetros, generalmente atendiendo a la zona donde se encuentre el personaje principal.

Por consiguiente, este sistema ofrece una forma de ir escalando el desafío propuesto al jugador y que el juego no acabe siendo monótono. Además, es una forma de mostrar progreso al jugador, ya que al luchar contra varios enemigos, podrá subir de nivel a sus personajes, mejorar sus habilidades, hacer frente a mayores desafíos o incluso, enfrentarse al jefe final del nivel.

Figura 5.4 – Vista de la pantalla de combate.

Los combates se desarrollan a través de un sistema de turnos. Así, el jugador dará una serie de órdenes a sus personajes, las cuales se llevarán a cabo en un determinado orden junto a las acciones de los enemigos, dependiendo de ciertos factores. Asimismo, a través del atributo de agilidad de los héroes y los enemigos, se estima el orden de consecución de los ataques.

El enfrentamiento puede llegar a su fin de tres formas diferentes. Por un lado, el jugador puede salir victorioso si logra vencer a todos los enemigos. Por otro lado, si los contrincantes logran derrotar a todos los héroes, el combate finalizará como una derrota. Por último, se ofrece la posibilidad de poder realizar una huida del combate, lo que permite al jugador regresar al nivel principal sin necesidad de derrotar al enemigo.

Figura 5.5 – Vista de la pantalla de resumen del combate.

Asimismo, a la hora de realizar ciertas acciones durante el combate, se invita al jugador a participar de forma activa. De esta manera, cuando el jugador ha dado las órdenes a sus personajes y comienzan a sucederse los ataques, es posible que tenga que realizar una acción adicional dependiendo de la orden elegida.

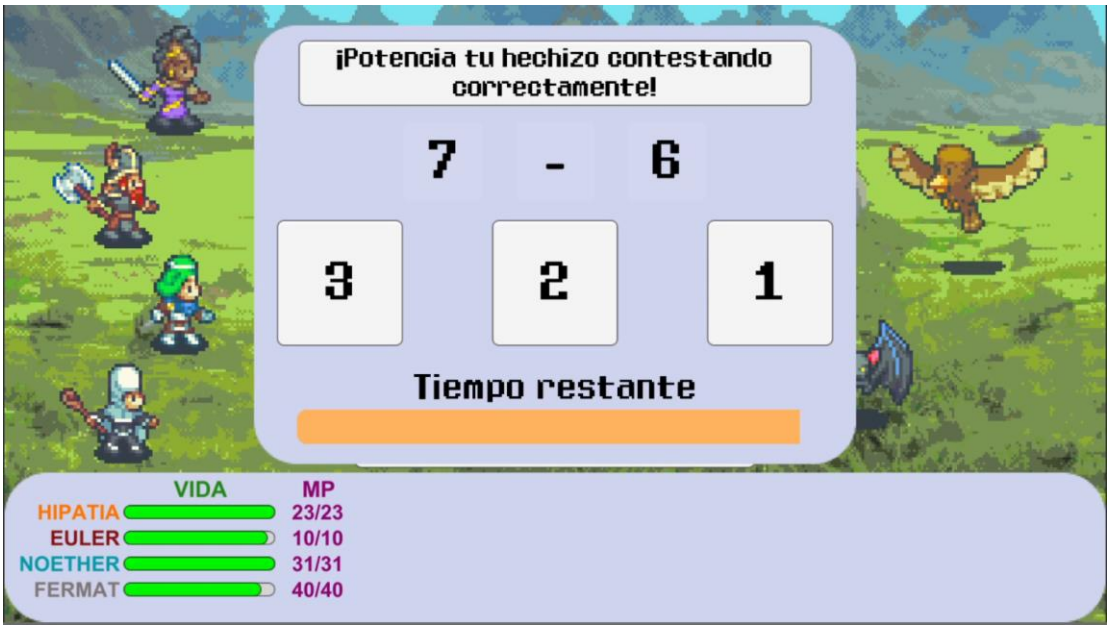


Figura 5.6 – Vista de la pantalla del minijuego durante el combate.

Así, si el jugador ha escogido la opción de lanzar un hechizo o magia, debe permanecer atento al combate, ya que a la hora realizar el ataque, tendrá que resolver un pequeño minijuego. Este reto consiste en contestar correctamente a una operación aritmética en un determinado tiempo límite.

De esta manera, si el usuario contesta adecuadamente, el hechizo lanzado será más potente y hará más daño al enemigo. En caso de contestar incorrectamente, el hechizo se lanzará, pero perderá parte de su poder y efectividad.

5.4. Subida de nivel.

El aumento del nivel y de las estadísticas de los personajes se logra a través de la obtención de puntos de experiencia venciendo a los enemigos en combate. Conforme se avanza en esta progresión, los puntos de experiencia necesarios para alcanza un nuevo nivel aumentan de forma lineal siguiendo una determinada fórmula.

$$XP = (4 * nivel) + 1$$

De esta manera, se puede calcular la curva de subida de nivel. Los puntos de experiencia aumentan de forma constante por cada nivel alcanzado [49]. En la tabla 5.1 se puede ver la progresión necesaria para los primeros diez niveles.

Nivel	Puntos necesarios para alcanzar el siguiente nivel
1	XP = 9
2	XP = 13
3	XP = 17
4	XP = 21
5	XP = 25
6	XP = 29
7	XP = 33
8	XP = 37
9	XP = 41
10	XP = 45

Tabla 5.1 – Progresión y puntos de experiencia necesarios para la subida de niveles.

6. Manual de usuario.

6.1. Introducción.

Nota: para este capítulo se ha empleado un registro de lenguaje más informal. La razón es que el objetivo de este manual es el propio jugador y utilizar un lenguaje más cercano resulta más adecuado.

"Hipatia y los guardianes aritméticos" es un juego en el que tendrás que recorrer y visitar diferentes localizaciones en busca del secreto que encierran los antiguos guardianes sobre un poder olvidado que asoló el mundo hace mucho tiempo.

¡Acompaña a Hipatia y a su grupo en una trepidante aventura! Podrás visitar todo tipo de lugares y buscar pistas del paradero de los guardianes. Habla con todos los personajes que veas para poder profundizar en la historia del juego y conocer sus inquietudes.

Pero ten cuidado durante tu viaje, ya que existen muchos peligros ahí fuera. Deberás enfrentarte a todo tipo de criaturas que intentarán detener tu avance.

¡Fortalece las habilidades de tus héroes para que puedan salir airosos de cualquier enfrentamiento!

6.2. Menú principal.

Al iniciar el juego accederás al menú inicial del título. Aquí podrás encontrar diferentes opciones.

Figura 6.1 – Menú inicial del juego.

Con el botón de “comenzar” podrás iniciar la partida. Si es la primera vez que juegas, comenzarás el juego desde el principio. En caso de tener una partida guardada, los datos se cargarán automáticamente.

Si pulsas en la opción “tutorial”, tendrás acceso a un pequeño manual con instrucciones acerca del juego y algunos consejos que te serán muy útiles durante tu aventura.

A través del botón de “opciones” podrás personalizar algunos aspectos de tu partida, como el volumen de la música o la posición del joystick, entre otras opciones. En el siguiente apartado se especifica toda la configuración disponible.

Gracias al botón de “créditos” podrás acceder a la información del creador del juego y a los autores de los diferentes recursos que se han utilizado para su creación.

6.2.1. Submenú opciones.

En este submenú tendrás la posibilidad de personalizar diferentes aspectos de tu partida de juego.

Figura 6.2 – Submenú de opciones del menú principal.

El botón de “borrar partida” te permite borrar los datos guardados de una sesión anterior. Ten cuidado al usar esta opción, una vez confirmes el borrado de la información, no será posible su recuperación. Es decir, se

trata de un proceso irreversible y no podrás recuperar tu partida una vez se haya borrado.

El volumen de la música y de los efectos se puede ajustar de forma independiente, desplazando los cursores hacia la izquierda (menor volumen) o la derecha (mayor volumen).

También es posible configurar la posición del *joystick*, pudiendo elegir entre izquierda (por defecto) y derecha.

La velocidad del texto determina cómo de rápido se visualizarán los diferentes diálogos de la historia y las conversaciones con el resto de personajes. Puedes elegir entre tres velocidades diferentes.

Por último, tienes la opción de activar o desactivar el sonido que se reproduce al pulsar cualquier botón de la pantalla.

6.3. Nivel principal.

En el nivel principal del juego podrás explorar diferentes lugares, simplemente moviendo a tu personaje deslizando el *joystick* de control táctil. Cuánto más lo inclines hacia una determinada dirección, mayor será la velocidad de tu personaje.

Figura 6.3 – Captura de pantalla del nivel principal.

Pero no tengas demasiada prisa y tómate tu tiempo para entablar conversación con los diferentes habitantes, ya que te proporcionarán pistas y te ayudarán a progresar en tu aventura. En otras ocasiones, tendrás que encontrar a algún personaje clave para poder continuar.

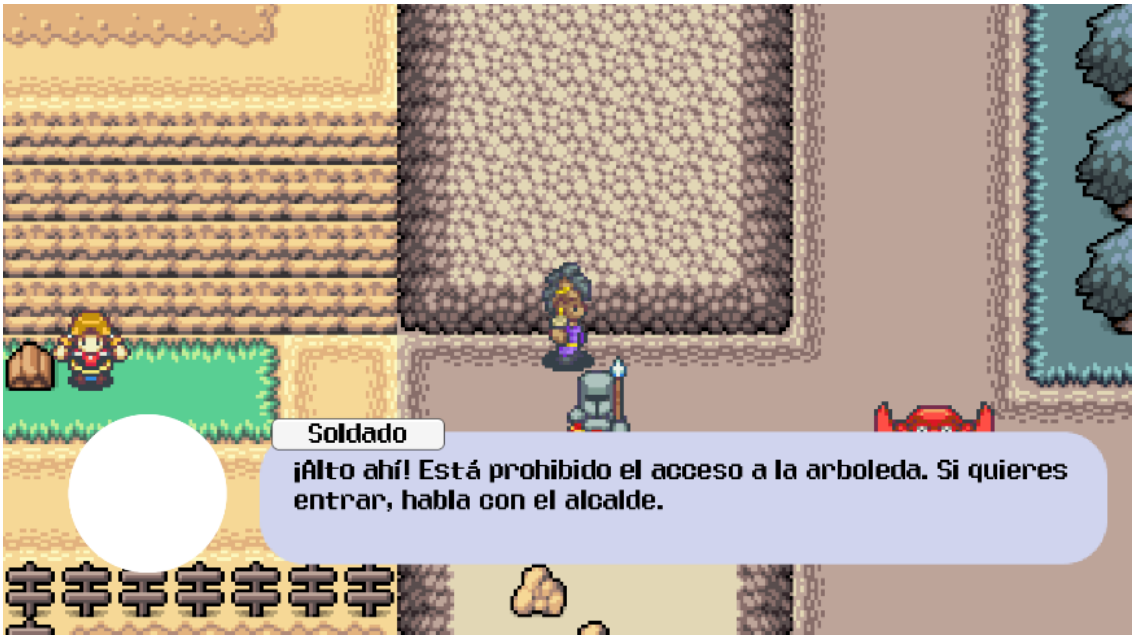


Figura 6.4 – En ocasiones tendrás que cumplir ciertas condiciones para poder avanzar.

6.3.1. Menú principal.

Si pulsas el icono de la parte superior derecha podrás acceder al menú principal. La información que se te mostrará en primer lugar son las diferentes estadísticas de cada uno de tus personajes.

Además, si pulsas en los iconos de los objetos, podrás usarlos directamente sobre uno de tus héroes.

1 Estadísticas de los personajes
2 Listado de objetos
3 Botón de guardar partida
4 Botón de volver a pantalla de título

Figura 6.5 – Vista del menú principal.

6.4. Combate.

A lo largo de la aventura te toparás con multitud de enemigos que intentarán detener tu avance y derrotar a tus héroes. Piensa bien tu estrategia y dirige a tus compañeros hacia la victoria.

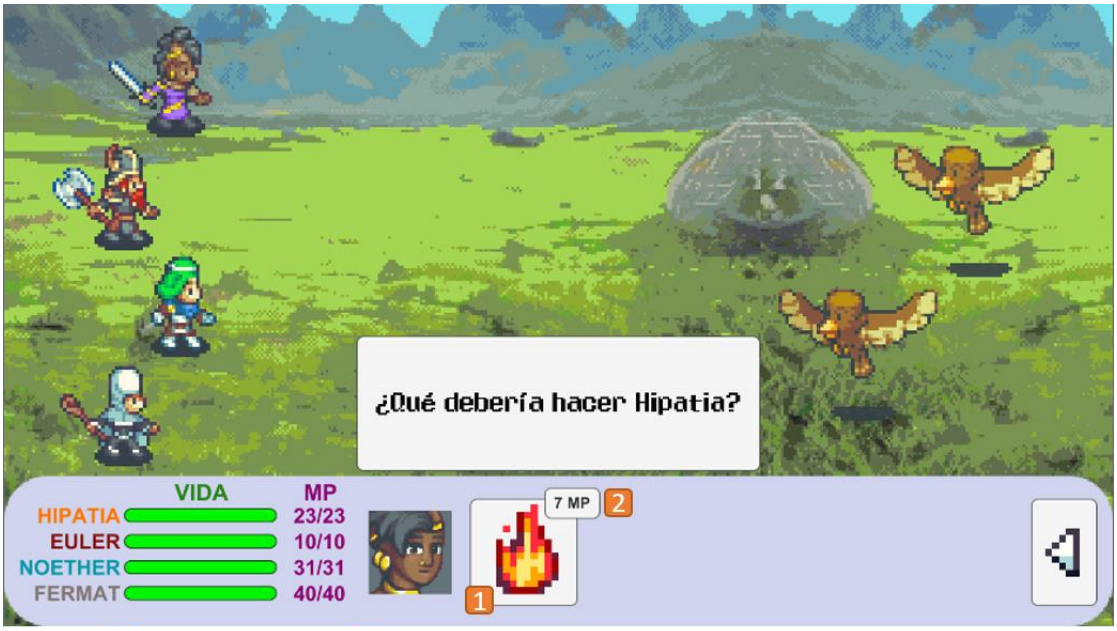
Figura 6.6 – Captura del menú inicial de combate.

En el menú inicial puedes elegir entre cuatro acciones: ataque normal, ataque mágico, usar objeto o huida.

Si seleccionas ataque, a continuación tendrás que elegir a un objetivo. Puedes hacerlo tocando directamente sobre el enemigo o bien pulsando el botón con su imagen. ¡Tú decides!

Figura 6.7 – Menú de selección de objetivo.

Si seleccionas ataque mágico, a continuación tendrás que elegir el tipo de magia. En la parte superior derecha del icono de hechizo se encuentra su coste de puntos mágicos (MP). Si no tienes suficientes MP, no podrás lanzar el hechizo. Seguidamente, debes elegir al objetivo.



1 Botón de hechizo 2 Cantidad de puntos de magia (MP) para lanzar el hechizo

Figura 6.8 – Menú de selección de hechizo.

Si optas por usar un objeto, accederás a la lista de objetos disponibles. El número de la parte superior derecha indica la cantidad disponible del objeto en cuestión. Para acabar, tendrás que elegir a un objetivo aliado tocándolo o pulsando el botón con su imagen para usar el objeto.

1 Listado de objetos 2 Cantidad disponible del objeto

Figura 6.9 – Menú de selección de objeto.

Figura 6.10 – Menú de selección de objetivo del objeto.

Por último, si pulsas sobre la huida, tu personaje intentará huir del combate. Esta acción la pueden repetir varios de ellos para aumentar las posibilidades de huida. Una huida a tiempo, es una victoria.

Una vez que todos tengan su función asignada, comienza el intercambio de ataques. Pero no te relajes mucho, ya que deberás ayudar a tus personajes a lanzar sus ataques mágicos. Si contestas bien a la pregunta planteada, tus ataques mágicos serán más efectivos.

Figura 6.11 – Vista del minijuego dentro del combate.

¡Presta atención a la pregunta! Si aciertas, potenciarás el hechizo lanzado por tu personaje. Simplemente pulsa el botón de la opción que creas que es correcta. Recuerda que hay límite de tiempo para contestar a la pregunta y que variará según el hechizo utilizado. Ten en cuenta que la dificultad de la pregunta también cambia según el hechizo que se haya escogido.

6.5. Uso de objetos.

Durante tu aventura podrás utilizar objetos para reponer la fuerza de tus héroes. Puedes utilizarlos en mitad de un combate o bien desde el menú principal del juego.

Utiliza las pociones verdes para restaurar salud y las rojas para restablecer los puntos mágicos (MP) de tus personajes.

¡No te olvides de curar a tus compañeros antes de la siguiente batalla!

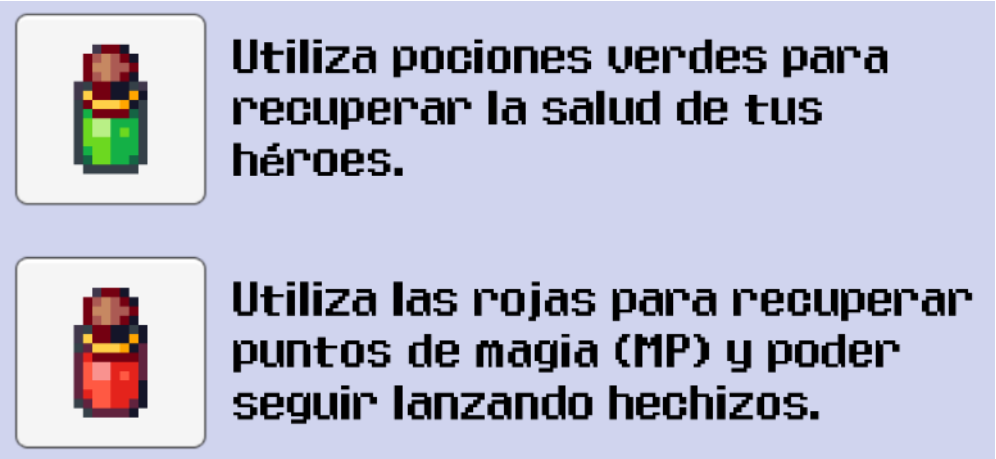


Figura 6.12 – Captura del tutorial dónde se explica el efecto de los objetos.

7. Testeo y pruebas con usuarios reales.

7.1. Conceptualización y prototipo inicial.

Inicialmente la jugabilidad del proyecto se iba a centrar en contar una historia y en los combates por turnos. Después de pedir opinión a diferentes compañeros y amigos, mucho de ellos comentaron que el hecho de centrarse en los combates podría llegar a ser repetitivo.

Por esta razón se añadió a la jugabilidad la posibilidad de explorar un nivel principal y poder hablar con diferentes personajes. De forma paralela, el jugador también podría participar en combates por turnos. De esta manera, los enfrentamientos contra enemigos en los denominados combates por turnos seguían teniendo un gran peso a nivel jugable.

Todo esto se podría considerar una primera iteración justo antes de la concepción del primer prototipo inicial. Una vez realizado dicho prototipo, fue posible probar la jugabilidad y probar si los diferentes elementos encajaban de forma correcta en el juego.

7.2. Pruebas de la versión jugable.

Durante las pruebas de la versión jugable, los usuarios pudieron probar un juego que ya tenía definida por completo la jugabilidad. Uno de los objetivos de esta versión era comprobar las sensaciones que se producían en los usuarios finales, especialmente a la hora de interactuar con los diferentes elementos del juego.

Durante esta fase, el juego pudo ser probado por tres usuarios, que aportaron una retroalimentación acerca de su opinión del juego. Así, se realizaron comentarios acerca de los elementos que les habían convencido, así como de otros que echaban en falta. De esta manera, se pudieron catalogar una serie de mejoras en referencia al *feel* por parte de los jugadores de varios de los elementos de juego.

7.3. Pruebas finales.

La pruebas finales se realizaron justo antes de la entrega final. Estas pruebas fueron realizadas por tres usuarios reales, que pudieron probar el juego en sus propios dispositivos.

Durante esta fase se realizaron algunos añadidos finales, como por ejemplo, la adición de un pequeño tutorial dentro del juego, ya que fue una funcionalidad que pidieron varios de los usuarios de las pruebas.

De esta manera, se pudieron localizar diferentes errores y comportamientos no deseados, que pudieron corregirse antes de pasar a la denominada versión *Gold*. En resumen, el hecho de realizar estas pruebas con usuarios finales en distintas etapas del desarrollo, desembocó en el logro de un producto más pulido y probado de antemano antes de su posible lanzamiento y entrega final.

8. Conclusiones.

8.1. Conclusiones finales del proyecto.

A lo largo del desarrollo de este proyecto se ha podido experimentar con múltiples aspectos relacionados con el desarrollo de videojuegos, tanto a nivel conceptual, técnico y artístico.

Este desarrollo ha supuesto una oportunidad perfecta para poder poner en práctica todos los conocimientos obtenidos en las diferentes asignaturas de esta titulación. De esta forma, se han empleado conocimientos sobre programación, diseño de interfaces, estructuras de datos, uso de diferentes diagramas, etc.

Uno de los logros obtenidos durante la concepción de este proyecto ha sido poder experimentar el desarrollo de un videojuego completo. Todo esto ha supuesto poder profundizar en el uso de distintas herramientas, principalmente el motor de juego Unity.

También ha resultado muy satisfactorio poder profundizar en la conceptualización y diseño de videojuegos, realizando una experimentación y refinamiento de las soluciones a través de un desarrollo iterativo basado en el diseño de un prototipo inicial y su mejora continua a lo largo del tiempo. Un punto a destacar ha sido el cuidado de varios aspectos a la hora de diseñar el juego, especialmente las sensaciones y percepción por parte del jugador a la hora de jugar.

Los objetivos planteados durante las primeras fases de desarrollo, así como su planificación has resultado en general satisfactorias. Sin embargo, la redacción de la documentación sufrió algunos retrasos debido al propio desarrollo. Además, algunos capítulos de esta memoria supusieron un tiempo mayor del estimado inicialmente.

El hecho de haber profundizado en el uso de las diferentes herramientas ha provocado que algunas de las últimas tareas pudieran llevarse a cabo en menos tiempo, lo que ha permitido compensar los retrasos e imprevistos surgidos en algunas de las tareas anteriores.

La metodología empleada ha permitido poder identificar y prevenir diferentes problemas a lo largo del desarrollo. Esto ha sido posible debido a la mejora continua en cada una de las iteraciones, que ha permitido pulir la jugabilidad y ampliar las diferentes funcionalidades del juego.

En general, la planificación y consecución de los diferentes objetivos ha sido satisfactoria y se han podido plasmar las ideas iniciales en el producto final, cumpliendo con las funcionalidades planteadas durante el inicio del proyecto.

8.2. Posibles líneas de trabajo futuro.

En primer lugar, una de las posibles mejoras de este proyecto consiste en aumentar la escala de su contenido. De esta manera, la creación de nuevo niveles dotará de mayor duración al título y será posible poder contar una historia completa de principio a fin.

Otro de los aspectos que no se han podido abordar es el soporte de varios idiomas. Ante un hipotético lanzamiento global, sería necesario contar al menos con el idioma inglés, ya que ampliaría enormemente el público objetivo.

Por otro lado, una de las funciones que hubiera sido interesante implementar sería un gestor más amplio de los objetos que pueden usar los personajes. Es decir, podría ampliarse la variedad de objetos a poder utilizar, e incluso diseñar un posible sistema que permita equipar diferentes piezas de equipo, como armas y accesorios que repercutieran en las estadísticas de los personajes. O incluso que les aportara algún tipo de nueva habilidad.

Adicionalmente, los combates pueden aumentar su profundidad jugable. Un buen comienzo sería la implementación de diferentes estados que pudieran beneficiar o perjudicar a los personajes. Estos estados tendrían su efecto directamente en las estadísticas de los personajes. Por ejemplo, sería posible envenenar a un personaje y que éste fuera perdiendo puntos de vida tras cada turno. Todo ello abre la posibilidad de implementar nuevos objetos (como un antídoto) o bien nuevos hechizos de sanación.

Por último, otro aspecto bastante interesante a tratar sería el tema de monetización. Al hacer uso de la plataforma Android como principal objetivo, serían posibles varias formas de monetización del juego. Por ejemplo, se podría basar en micropagos o bien dando la posibilidad de visualización de pequeños vídeos publicitarios a cambio de determinados objetos dentro del juego. También se podría ofrecer la opción de realizar un único pago y acceder a todo el contenido del juego.

En resumen, este género de juegos puede llegar a alcanzar una alta profundidad jugable ya que dispone de muchos tipos de estadísticas que pueden ser alteradas. Sin embargo, uno de los grandes inconvenientes es el tiempo que puede llegar a consumir toda su implementación, así como el aumento de dificultad a la hora de constituir todas las opciones y explicarlas al jugador.

9. Glosario

Canvas.
: El *canvas* es el área dónde se sitúan todos los elementos de la interfaz en una escena de Unity. Todos los elementos de la interfaz deben ser hijos de dicho *canvas*.

Escena.
: En Unity se trata de la entidad que contiene todos los elementos y objetos del juego, como los personajes, interfaz, etc.

Extensión .meta.
: Es una extensión de archivos especiales que permiten a Unity identificar cada uno de los recursos que se utilizan en el juego. Por ejemplo, los directorios, ficheros de imagen, sonido, etc.

Frame.
: Se trata básicamente de una imagen. La consecución de estas imágenes da lugar al *frame rate*, que es la frecuencia de estas imágenes que son mostradas por la pantalla.

Indie.
: Este término, referido al mundo de los videojuegos, indica que el juego es de carácter independiente y se ha desarrollado por una única persona o pequeño grupo sin contar con un gran editor (*publisher*).

Japanese Role Playing Game (JRPG).
: Término por el que se conoce a los juegos de rol popularizados en Japón a finales de los 80 y 90. Este término se sigue utilizando popularmente incluso con juegos que siguen la misma estética japonesa, aunque no hayan sido desarrollados en Japón.

Non Playable Characters (NPCs).
: Se trata de personajes que aparecen en el juego, pero no pueden ser controlados por el jugador. Suelen aportar pistas o tener algún objetivo principal en la trama del juego.

Role Playing Game (RPG).
: Es un género de juegos dónde el jugador controla a uno o varios personajes a través de una serie de misiones hasta alcanzar su destino final.

Singleton.
: Se trata de un patrón de diseño, que permite que una clase se instancie una única vez durante toda la ejecución de un determinado programa o aplicación.

Spin-off.

Se trata de una obra derivada de otra principal. Suelen estar protagonizadas por personajes importantes de la obra principal, añadiendo nuevas tramas o acontecimientos.

Sprint.

Se trata de cada ciclo o iteración que se realiza en un proyecto bajo la metodología *Scrum*.

Sprite.

Se trata de una imagen en dos dimensiones que puede formar parte o ser un conjunto de otras. Se suele utilizar en el denominado pixel art y puede representar modelos de personajes o fondos de pantalla, por ejemplo.

Unity Asset Store.

Se trata de una página que aloja recursos, tanto gratuitos como de pago, compatibles con el motor de juegos Unity.

Versión *Gold*.

Suele ser la versión final de un determinado software. Esta versión es la que será distribuida a los usuarios finales.

10. Bibliografía

[1] *ISFE Key Facts – ISFE* [en línea] [Consulta: 4 de abril de 2021]. Disponible en: https://www.isfe.eu/isfe-key-facts/

[2] ISFE Key Facts.2019. *Revenue-split* [Gráfico]. Disponible en: https://www.isfe.eu/isfe-key-facts/

[3] MUÑOZ, J.; MARTÍNEZ TORRES, D.; Liga de Videojuegos Profesional. Guía legal sobre esports. Presente y futuro de la regulación de los esports en España. En: *Guía legal sobre esports* [en línea]. 2018. [Consulta: 14 de abril de 2021]. Disponible en: https://es.ontier.net/ia/guialegalesports-2018web.pdf

[4] GRIZZARD, Matthew; TAMBORINI, Ron; LEWIS, Robert; WANG, Lu; PRABHU, Sujay. Being Bad in a Video Game Can Make Us Morally Sensitive. En: *Cyberpsychology, behavior and social networking* [en línea]. 2014. Vol. 10, nº 10. DOI: 10.1089/cyber.2013.0658. [Consulta: 15 de abril de 2021]. Disponible en: https://www.researchgate.net/publication/263296879_Being_Bad_in_a_Video_Game_Can_Make_Us_Morally_Sensitive

[5] PETRILLO, Fábio, et al. Houston, we have a problem... a survey of actual problems in computer games development. En: *Proceedings of the 2008 ACM symposium on Applied computing* [en línea]. 2008. p. 707-711. [Consulta: 15 de abril de 2021]. Disponible en: https://dl.acm.org/doi/pdf/10.1145/1363686.1363854

[6] STARLOOP STUDIOS. Best Agile Practices in Game Development [en línea]. *Starloop Studios*. (25 de agosto de 2020). [Consulta: 16 de abril de 2021]. Disponible en: https://starloopstudios.com/best-agile-practices-in-game-development/

[7] FLOOD, Kevin. Game Unified Process [en línea]. *Gamedev*. (14 de mayo de 2003). [Consulta: 16 de abril de 2021]. Disponible en: https://www.gamedev.net/tutorials/programming/general-and-gameplay-programming/game-unified-process-r1940/

[8] HIGUCHI, Marcelo Makoto; NAKANO, Davi Noboru. Agile design: A combined model based on design thinking and agile methodologies for digital games projects. En: *Revista de Gestão e Projetos* [en línea]. 2017. Vol. 8, nº 2, p. 109-126. [Consulta: 18 de abril de 2021]. Disponible en: https://periodicos.uninove.br/gep/article/view/9670

[9] BROWN, T. Design thinking. En: *Harvard Business Review* [en línea]. 2008. Vol. 86, nº 63, págs. 84-92,141. [Consulta 18 de abril de 2021]. Disponible en: http://search.ebscohost.com/login.aspx?direct=true&db=edselc&AN=eds

elc.2-52.0-46749127039&authtype=sso&custid=uoc&site=eds-live&scope=site

[10] SCHWABER, Ken. *Agile project management with Scrum* [en línea]. Microsoft press, 2004. [Consulta: 20 de abril de 2021]. Disponible en: https://books.google.es/books?hl=es&lr=&id=6pZCAwAAQBAJ&oi=fnd&pg=PT9&dq=Schwaber+scrum&ots=kcpWP_6ukX&sig=cz8gt1RuZzZlJD51XuxC4zqb60g#v=onepage&q=Schwaber%20scrum&f=false

[11] GODOY, André; BARBOSA, Ellen F. Game-Scrum: An approach to agile game development. En: *Proceedings of SBGames* [en línea]. 2010. p. 292-295. [Consulta: 20 de abril de 2021]. Disponible en: http://www.sbgames.org/sbgames2010/proceedings/computing/short/Computing_short19.pdf

[12] SANSONE, Anthony T. Game design documents: Changing production models, changing demands. En: *Computer games and technical communication: Critical methods and applications at the intersection* [en línea]. 2014, p. 109-124. [Consulta: 21 de abril de 2021]. Disponible en: https://books.google.es/books?hl=es&lr=&id=0s0WBgAAQBAJ&oi=fnd&pg=PA109&dq=Game+Design+Document&ots=FYqsWa6XCu&sig=gD_MJXWCNyAB0XX_IOgGv7Nt690#v=onepage&q=Game%20Design%20Document&f=false

[13] DORMANS, Joris, et al. Engineering emergence: applied theory for game design [en línea]. Universiteit van Amsterdam [Host]. 2012. [Consulta: 21 de abril de 2021]. Disponible en: https://pure.uva.nl/ws/files/1167833/102090_08.pdf

[14] mediaXstanford (charla de LIBRANDE, Stone). (11 de mayo de 2017). *One-Page Designs*. [vídeo en línea]. https://www.youtube.com/watch?v=GXmsxYm0Mk0

[15] SCHWABER, Ken; SUTHERLAND, Jeff. The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game. En: *The Scrum Guide* [en línea]. 2020. [Consulta: 23 de abril de 2021]. Disponible en: https://scrumguides.org/scrum-guide.html

[16] UNITY TECHNOLOGIES. *Unity* [software]. Versión 2020.1.1f1. Android Build Support. 2020.

[17] MICROSOFT. *Visual Studio* [software]. Versión Community, 16.8.4.2020.

[18] TOM'S PLANNER NV. *Tom´s Planner* [software]. 2021.

[19] HITCHENS, Michael, et al. The many faces of role-playing games. En: *International journal of role-playing* [en línea]. 2009. Vol. 1, nº 1, p. 3-21. [Consulta: 25 de abril de 2021]. Disponible en:

https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.468.2599&rep=rep1&type=pdf

[20] ADAMS, Ernest. *Fundamentals of Role-Playing Game Design* [en línea]. New Riders, 2014. [Consulta: 25 de abril de 2021]. Disponible en: https://books.google.es/books?id=1vQqAwAAQBAJ&printsec=frontcover&dq=Adams+ernest+2014+Fundamentals+Game+Design&hl=es&sa=X&ved=2ahUKEwjTyfODmJrwAhVz6uAKHW8EDhoQ6AEwB3oECAkQAg#v=onepage&q&f=false

[21] MALLINDINE, Jayme Dale. *Ghost in the Cartridge: Nostalgia and the Construction of the JRPG Genre* [en línea]. Gamevironments, 2016. [Consulta: 26 de abril de 2021]. Disponible en: https://media.suub.uni-bremen.de/handle/elib/3283

[22] CHUMBOSOFT LLC.(2018). Captura del juego "Heartbeat" [imagen]. Steam. Disponible en: https://cdn.cloudflare.steamstatic.com/steam/apps/984560/ss_b2e67901dcbd05fccd980a8a454089660f624fad.1920x1080.jpg?t=1545874354

[23] GRAYFAX SOFTWARE. (2020). Captura del juego "Orangeblood" [imagen]. Steam. Disponible en: https://cdn.cloudflare.steamstatic.com/steam/apps/1042670/ss_36640a8d60d8d8c08e59d75416b433e0c91f9d7b.1920x1080.jpg?t=1585020355

[24] WIEGERS, Karl; BEATTY, Joy. *Software requirements* [en línea]. Pearson Education, 2013. [Consulta: 28 de abril de 2021]. Disponible en: https://books.google.es/books?hl=es&lr=&id=nbpCAwAAQBAJ&oi=fnd&pg=PT32&dq=software+requirements&ots=9pK0FZ7BSq&sig=iQZ6a8Oxpvq3NUsBpNQJ2AYt1ww#v=onepage&q&f=false

[25] *Make Your Own Game with RPG Maker* [en línea] [fecha de consulta: 29 de abril de 2021]. Disponible en: https://www.rpgmakerweb.com/

[26] Godot Engine - Free and open source 2D and 3D game engine [en línea] [fecha de consulta: 29 de abril de 2021]. Disponible en: https://godotengine.org/

[27] CALABRÓ, Andrea. 2017. *Godot Engine Logo*, licensed under CC-BY-4.0 International (https://creativecommons.org/licenses/by/4.0/). Disponible en: https://godotengine.org/themes/godotengine/assets/press/logo_large_color_light.png

[28] RPG Maker. *Tabla comparativa de versiones* [Gráfico]. Disponible en: https://rpgmaker.webflow.io/products

[29] *Apache Subversion* [en línea] [fecha de consulta: 3 de mayo de 2021]. Disponible en: https://subversion.apache.org/

[30] *Git* [en línea] [fecha de consulta: 3 de mayo de 2021]. Disponible en: https://git-scm.com/

[31] *Perforce Software | Development Tools For Innovation at Scale* [en línea] [fecha de consulta: 3 de mayo de 2021]. Disponible en: https://www.perforce.com/

[32] LONG, Jason. *Git Logo Color*, licensed under CC-BY-3.0 Unreported (https://creativecommons.org/licenses/by/3.0/). Disponible en: https://git-scm.com/images/logos/downloads/Git-Logo-2Color.png

[33] *Unity – Manual: System requirements for Unity 2020.1* [en línea] [fecha de consulta: 3 de mayo de 2021]. Disponible en: https://docs.unity3d.com/2020.1/Documentation/Manual/system-requirements.html

[34] *Definición de compatibilidad con Android 9* [en línea] [fecha de consulta: 3 de mayo de 2021]. Disponible en: https://source.android.com/compatibility/9/android-9-cdd.html

[35] BALSAMIQ STUDIOS. *Wireframes for Desktop* [software]. Versión 4.2.1. 2021.

[36] VISUAL PARADIGM. *Visual Paradigm* [software]. Community Edition, versión 16.2. 2021.

[37] THE TORTOISE TEAM. *TortoiseSVN* [software]. Versión 1.14.1. 2021.

[38] AUDACITY TEAM. *Audacity* [software]. Versión 3.0.2. 2021.

[39] SLAVOV, Filip. *Wise SVN* [software]. Versión 1.3.5. 2021.

[40] FENERAX STUDIOS. *Joystick Pack* [software]. Versión 2.1. 2019.

[41] *Tyler Warren RPG Battlers – itch.io* [en línea] [fecha de consulta: 4 de abril de 2021]. Disponible en: https://tylerjwarren.itch.io/

[42] *finalbossblues está creando Pixel Art Game Assets – Patreon* [en línea] [fecha de consulta: 4 de abril de 2021]. Disponible en: https://www.patreon.com/finalbossblues

[43] *finalbossblues – itch.io* [en línea] [fecha de consulta: 4 de abril de 2021]. Disponible en: https://finalbossblues.itch.io/

[44] *Mega Tiles – itch.io* [en línea] [fecha de consulta: 4 de abril de 2021]. Disponible en: https://megatiles.itch.io/

[45] *Muz Station Productions - Asset Store* [en línea] [fecha de consulta: 4 de abril de 2021]. Disponible en: https://assetstore.unity.com/publishers/4078

[46] *ZapSplat - Download Free Sound Effects & Royalty Free Music* [en línea] [fecha de consulta: 5 de mayo de 2021]. Disponible en: https://www.zapsplat.com/

[47] SHNEIDERMAN, Ben, et al. *Designing the user interface: strategies for effective human-computer interaction* [en línea]. Pearson, 2016. [Consulta: 15 de mayo de 2021]. Disponible en: https://nsuworks.nova.edu/gscis_facbooks/18/

[48] *Unity - Manual: Coroutines* [en línea] [fecha de consulta: 18 de mayo de 2021]. Disponible en: https://docs.unity3d.com/2020.1/Documentation/Manual/Coroutines.html

[49] PAV. Level systems and character growth in RPG games [en línea]. *Pav Creations*. (12 de octubre de 2020). [Consulta: 2 de mayo de 2021]. Disponible en: https://pavcreations.com/level-systems-and-character-growth-in-rpg-games/