UNIVERSIDAD
DE GRANADA

Desarrollo de un juego RPG adaptativo con Unity

Grado en Ingeniería Informática

Trabajo Fin de Grado

Autor:

Jorge Juan Ñíguez Fernández

Tutor:

Alejandro José León Salas

Índice

Abstract

Videogame's world has been an entity in continuous evolution since the beginning. A being that has grown to become one of the most important entertainment industries, with RPGs (Role Playable Game) being one of the most played genres in the world and also one of the oldest.

Although the genre has changed a lot since its beginning and divided into two main side, the WRPG (Western RPG) and the JRPG (Oriental RPG, mainly from Japan) each one with very defined characteristics and visual styles. We live a few years of stagnation and little evolution in this genre, as in video games in general, if we ignore the evident graphic jump that all these have suffered.

The objective that we intend to present here is an alternative to one of the oldest mechanics and that less change has suffered since its creation, the difficulty. We understand the difficulty as the resistance that the game offers to the player who tries to overcome his challenges. A greater difficulty will offer greater challenges that will require greater ability to be overcome while on the opposite side we will find a game easier to overcome. Our goal will be to create a personalized experience for each player and thus achieve challenges that are not too simple or too difficult to create a satisfying experience. The project presents a novel enrichment of the gameplay associated with the RPG in particular but which aims to be extensible to other genres.

1. Introducción

Los videojuegos hoy día abarcan un sinfín de géneros, estilos artísticos y narrativos, pero sobre todo de experiencias. Sin embargo, en su enorme mayoría solo enfocan un tema tan importante como es la dificultad o el reto para los jugadores desde dos visiones muy marcadas casi desde la concepción misma de los videojuegos abiertos al público en 1958. En seis décadas de incontables avances, los más palpables en el aspecto visual, la dificultad y la experiencia adaptativa siguen sin haber visto grandes avances. Entendemos como dificultad la resistencia que el juego ofrece al jugador que intenta superar sus retos. Una dificultad mayor ofrecerá retos mayores que requerirán de mayor habilidad para ser superados mientras que en el lado opuesto el juego será más amigable y sencillo de superar. Con una dificultad adaptativa se pretende que el juego analice al jugador y establezca cada reto de manera personalizada a cada jugador para evitar situaciones excesivamente complejas o sencillas y lograr así una experiencia satisfactoria.

Analizando estas dos vertientes vemos que podemos clasificar en una o en otra prácticamente la totalidad de los juegos existentes. La primera y más sencilla de las visiones es la de que el reto sea el mismo para todos más o menos fácil pero que cada jugador tenga exactamente la misma experiencia de juego. Juegos tan famosos como Mario Bros de Nintendo o Dark Souls de From Software serían dos de los principales estandartes de esta modalidad de juego siendo el primero muy accesible para el público y el segundo más enfocado en jugadores con gran experiencia. Este enfoque tiene grandes desventajas en muchos aspectos, pero centrándonos en el tema que nos atañe si nos fijamos en la dificultad de estos dos juegos vemos que se cierran a una clase de público en concreto y puede llegar a generar frustración en el público en el que se centra el otro juego. Podemos ejemplificar esto de manera sencilla, un jugador poco experimentado de primeras tendrá alguna dificultad para enfrentarse a un juego como Super Mario, pero o bien el juego le asistirá de primeras para que se haga con el manejo del juego lo más rápido posible o bien el jugador se habrá hecho finalmente con sus sencillas mecánicas y podrá disfrutar del resto del título. Sin embargo, un jugador ampliamente experimentado no tardará en hacerse con el control del juego y sentirá estas ayudas como una molestia y una vez este acabe podría perder el interés en el juego debido al poco o nulo reto que se encuentre. En el caso contrario, un jugador que entre en el mundo de Dark Souls sentirá instantáneamente una enorme frustración al ver que la dificultad no es comparable con su habilidad y que no puede cambiarla, debe aprender o dejarlo.

Figura 1.- Imágenes del juego Super Mario Odyssey y Dark Souls III

El segundo enfoque que se le da a los videojuegos es la elección de la dificultad por parte del jugador. En este enfoque el diseñador le da una serie de opciones desde la más accesible hasta el mayor reto para que sea el usuario quien decida como quiere afrontar el juego en función de sus habilidades. Este sería el caso de juegos como DOOM de id Software o los juegos deportivos de EA como FIFA. Si bien es cierto que este enfoque es más adecuado para que cada jugador decida como quiere hacer frente al juego en función de sus habilidades o de la experiencia que desea encontrar no deja de estar carente de inconvenientes. La elección suele hacerse nada más empezar el juego sin haber visto nada del juego y en muchos casos sin la posibilidad de cambiarla si no es empezando de nuevo. Esto ya de por sí puede ser una molestia para el jugador y en años recientes se ha ido solucionando ofreciendo la opción de cambiarla en cualquier momento del transcurso del juego. Sin embargo, esto no termina de solucionar nuestro problema pues existe la posibilidad que los niveles ofertados no sean suficientes o que no sean equiparables unos con otros y deje al jugador insatisfecho en todos ellos y generen una sensación de mal diseño.

Figura 2.- Portadas FIFA 19 y DOOM para PS4

En ambos casos, en mayor o menor medida, es el diseñador quien determina como debe ser el juego y como va a ser la experiencia. Esto relega al jugador a una posición secundaria y hace que la experiencia vivida siempre sea igual lo juegues las veces que lo juegues. ¿Por qué no hacer que cada jugador realmente decida su propia aventura y la dificultad se adapte a su progresión?

Figura 3.- Menú de selección de dificultad

Para terminar este capítulo quisiera hacer una pequeña introducción sobre el segundo punto que nos atañe en este proyecto que es la experiencia adaptativa.

Ciertos juegos han hecho acercamientos a este tipo de experiencia adaptando su historia a según qué respuestas diera un jugador en un evento dentro del juego, cambiando ciertos enemigos dependiendo de la habilidad del jugador o adaptando progresivamente la dificultad. Juegos como Omega Boost de PlayStation tenía un sistema de progresión que respondía al jugador, añadiendo nuevos enemigos más fuertes o eligiendo una ruta u otra dependiendo de si el jugador era más o menos habilidoso. Otro ejemplo sería Hellblade. Cuando iniciamos el juego, contamos con una dificultad que es automática, en la que se adaptará a la forma en la que jugamos, al inicio se nos aclara que la muerte permanente tiene relación en la historia, y ocurre que si el jugador muere en repetidas ocasiones el juego borra nuestro progreso y debemos volver a comenzar. El sistema automático de Hellblade pretende obligar al jugador a no acostumbrarse a un estilo de combate, sino que debe ser capaz de evolucionar al ritmo que le marca el sistema para evitar la repetición.

Nosotros pretendemos darle una vuelta más a estas mecánicas. Pretendemos que historia, jugabilidad y escenarios varíen para cada jugador. Que cada acción tenga su recompensa o su castigo. Esto creará una enorme reutilización del producto que añadirá horas de contenido al juego y experiencias nuevas con cada repetición a modo de recompensa al jugador que busque mejorar sus habilidades o enfrentar los retos de manera distinta.

2. Contexto

En este apartado se va a hacer un pequeño resumen sobre las características básicas que definen a un RPG como género y cuáles son las referencias que se han tomado para realizar este juego.

2.1. Videojuegos RPG

Los RPGs como género son aquellos en el que el jugador encarna a uno o varios personajes que deben superar una serie de retos y en las cuales el jugador tiene cierto poder de decisión sobre como solventarlas para hacer avanzar la historia, al contrario de juegos como los juegos de plataformas o los de lucha donde el transcurso está muy marcado y no suele variar de un jugador a otro.

La característica más reconocible dentro del género viene dada por la adaptación de elementos de los juegos de rol clásicos de mesa y es el desarrollo estadístico del personaje. El desarrollo estadístico se refiere la evolución de las principales características del personaje como serían ataque, defensa o velocidad la cual varía dependiendo del nivel, objetos, etc. Esta característica ha sido adoptada por otros géneros siendo muy popular actualmente en los shooter lo que ha dado lugar a su vez a un subgénero llamado RPS (Role Playing Shooter) destacando juegos tan exitosos como Fallout, BioSchok o Borderlands.

Sin embargo, en este trabajo vamos a centrarnos en los JRPG clásicos basados en la ya mencionada evolución estadística de los personajes, los combates por turnos y la toma de decisiones para construirnos un perfil y adaptar el juego a ese perfil que hemos establecido sobre el jugador.

Otros rasgos característicos de este género son el hecho que se suelen desarrollar en entornos de fantasía o futuristas, que los personajes poseen poderes sobrenaturales que se van desarrollando según el jugador utilice a estos personajes que controla, la gestión de recursos económicos y el uso de un inventario que ofrezca diversas combinaciones y vienes que usar para superar los distintos retos que se ha de enfrentar a lo largo del juego, los cuales se compran, se obtienen como resultado de batallas o se encuentran de manera fortuita.

2.2. Ejemplos de videojuegos del género RPG

Para terminar de visualizar este contexto mostraré algunos ejemplos de juegos que he empleado como referencia ya sea a nivel de mecánicas, gestión de los recursos, narrativa o arte, para desarrollar mi propio proyecto.

2.2.1. Pokémon

Figura 4.- Logo principal de Pokémon Company

Pokémon es sin duda una de las sagas más famosa del mundo de los videojuegos. Esta franquicia comenzó en febrero de 1996 en Japón y desde entonces se ha convertido en una de las franquicias más exitosas de la historia. 30 títulos de la saga principal, más 2 nuevos programados para final del año 2019, en torno a otros 40 secundarios, más aquellos en los que aparecen algunos de sus personajes y 4 aplicaciones para móviles entre los cuales destaca sin lugar a dudas Pokémon Go, juego que se convirtió en todo un fenómeno de masas sin precedentes con su lanzamiento en 2016. A esto hay que añadir series, películas, juegos de mesas, coleccionables, merchandising de todo tipo… Tal es el afán por este juego que en Japón podemos llegar a encontrar hasta alcantarillas públicas con las imágenes de algunos de sus personajes o un tren decorado íntegramente con su personaje más famoso, Pikachu. De este juego hemos extraído elementos de interfaz y del sistema de combate, así como de la gestión del inventario.

Figura 5.- Alcantarilla decorada en Japón con un personaje de la saga Pokémon

2.2.2. Final Fantasy

Figura 6.- Logo de Final Fantasy XV

La que fuera la saga más importante del género durante las décadas de los 80 y los 90 es recordada por muchos como una obra maestra llena de historias apasionantes, personajes carismáticos y mecánicas muy emuladas por otros juegos que le sucedieron en el tiempo. Tras ciertas entregas de dudosa calidad y diversos problemas económicos parece haber tenido un pequeño resurgimiento con su 15ª entrega numerada. Cuenta con más de 40 títulos en su historia. Su narrativa y gestión del equipo de los personajes han sido usados como ejemplo para el proyecto.

2.2.3. Octopath Traveler

Figura 7.- Imagen promocional de Octopath Traveler

Una de las últimas novedades dentro del género y que ha sabido entender muy bien a los fans del género. Tal fue su éxito que la propia Square Enix, empresa creadora del juego, tuvo que disculparse por las escasas unidades que habían llegado a las tiendas cuando fue lanzado. En este juego al final de cada episodio nos enfrentamos a una mazmorra con un jefe final especialmente poderoso que se ha usado como referencia para ciertas fases del juego llamadas "exámenes" las cuales explicaremos más adelante.

Figura 8.- Recorte de la noticia de la página Nintenderos

3. Motores para el desarrollo de videojuegos

El término motor de juego, hace referencia a una serie de rutinas de programación que permiten el diseño, la creación y el funcionamiento de un videojuego. Actualmente existe una gran variedad de estos, tanto gratuitos, completa o parcialmente, de pago, de código abierto y cerrado o aquellos propios de las empresas desarrolladoras que solo son utilizados en esas empresas.

Los pilares básicos para manejar uno de estos motores son los siguientes:

- Programa de juego principal:
  - La lógica del juego real tiene que ser implementada por algunos algoritmos. Es distinto de cualquier trabajo de renderizado, sonido o entrada.
- Motor de renderizado
  - El motor de renderizado genera gráficos 3D animados mediante cualquiera de varios métodos (rasterización, trazado de rayos, etc.). En lugar de programarse y compilarse para ejecutarse en la CPU o GPU directamente, la mayoría de las veces los motores de renderización se basan en una o varias interfaces de programación de aplicaciones (API), como Direct3D, OpenGL o Vulkan, que

proporcionan una abstracción del software de la GPU. Las bibliotecas de bajo nivel como DirectX, Simple DirectMedia Layer (SDL) y OpenGL también se usan comúnmente en juegos, ya que proporcionan acceso independiente del hardware a otro hardware como dispositivos de entrada (ratón, teclado y joystick), tarjetas de red, y tarjetas de sonido.

- Motor de audio
    - El motor de audio es el componente que consiste en algoritmos relacionados con la carga, modificación y salida de sonido a través del sistema de altavoces del usuario. Como mínimo, debe poder cargar, descomprimir y reproducir archivos de sonido. Los motores de audio más avanzados pueden calcular y producir cosas tales como efectos Doppler, ecos, ajustes de tono/amplitud, oscilación, etc.
- Motor de física
    - El motor de física es responsable de emular las leyes de la física de manera realista dentro de la aplicación. Específicamente, proporciona un conjunto de funciones para simular fuerzas físicas y colisiones, actuando sobre los diversos objetos dentro del juego en tiempo de ejecución.
- Inteligencia artificial
    - La inteligencia artificial es quien provee de estímulo al videojuego. Su elaboración es crítica a la hora de lograr un sistema de juego pulido y que entretenga. Puede tornarse muy compleja y es necesario tener en cuenta ciertas variables, tales como crear comportamientos programados, delimitar su visión del mundo, su interacción en él, la toma de decisiones y con ello lograr una consistencia lógica y coherente en la que el jugador debe responder de una manera esperada.

En este proyecto hemos empleado Unity ya que nos ofrece todas estas herramientas además de un entorno grafico sencillo, aunque cualquiera de los otros que vamos a nombrar podría haber servido para este propósito.

3.1. Unity

Figura 9.- Logo Unity

Unity es un motor de videojuego multiplataforma creado por Unity Technologies. Unity está disponible como plataforma de desarrollo para Microsoft Windows, OS X, Linux. La plataforma de desarrollo tiene soporte de compilación con diferentes tipos de plataformas (Véase la sección Plataformas objetivo). A partir de su versión 5.4.0 ya no soporta el desarrollo de contenido para navegador a través de su plugin web, en su lugar se utiliza WebGL.

Unity tiene dos versiones: Unity Professional (pro) y Unity Personal.

3.2. Unreal Engine

Figura 10.- Logo Unreal

Unreal Engine es un motor de juego de PC y consolas creado por la compañía Epic Games, demostrado inicialmente en el shooter en primera persona Unreal en 1998. Aunque se desarrolló principalmente para los shooters en primera persona, se ha utilizado con éxito en una variedad de otros géneros, incluyendo los videojuegos de sigilo, lucha, MMORPG y otros RPG.

Con su código escrito en C++, el Unreal Engine presenta un alto grado de portabilidad y es una herramienta utilizada actualmente por muchos desarrolladores de juegos.

3.3. GameMaker

Figura 11.- Logo GameMaker

GameMaker: Studio es una plataforma basada en un lenguaje de programación interpretado y un kit de desarrollo de software (SDK) para desarrollar videojuegos, creado por el profesor Mark Overmars en el lenguaje de programación Delphi, y orientado a usuarios novatos o con pocas nociones de programación. Posee una interfaz visual en la que hacer juegos sin programar directamente el código parecido a Scratch.

El programa es gratuito, aunque existe una versión comercial ampliada con características adicionales.

3.4. Godot

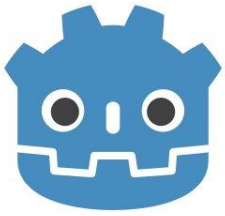


Figura 12.- Logo Godot

Godot es un motor de videojuegos 2D y 3D multiplataforma, de código abierto publicado bajo la Licencia MIT y desarrollado por la comunidad de Godot. Los videojuegos en Godot son codificados en el lenguaje de programación C#, o en el lenguaje GDScript. GDScript, es un lenguaje de programación de alto nivel, muy similar a Python que fue creado especialmente para Godot, por lo que añade funcionalidades y optimización.

4. Elecciones tecnológicas y por qué modificar el gameplay del género RPG

Como comentábamos inicialmente en la introducción de este documento nuestro objetivo es crear una experiencia que se adapte a cada usuario de manera única y para ello vamos a desarrollar un entorno, en este caso un RPG 2D, en el que probar su efectividad usando para ello a jugadores con distintos perfiles y experiencias en el mundo de los videojuegos.

Tomando como muestra los primeros niveles de juego que el usuario resuelva, con una dificultad predefinida, podremos hacernos una primera impresión de las habilidades del jugador. Una vez pasado este punto podremos determinar un grado de habilidad y no solo estableceremos la dificultad de los próximos enfrentamientos que el usuario deba resolver, sino que adaptaremos el propio entorno (generando mazmorras más o menos complejas) que cambiarán el aspecto en su totalidad del juego y la recompensa que recibirá al completar el reto para que sea acorde al reto enfrentado. Sin dejar de tomar medidas iremos reajustando esta dificultad que hemos propuesto para evolucionar al ritmo que el jugador lo hace y que la curva de dificultad no se desplome ni crezca demasiado rápido. Así mantendremos equilibrada la línea entre recompensa y frustración y que el juego se mantenga atractivo durante toda su duración.

El objetivo es, por una parte, realizar una adaptación dinámica tanto de la estructura de los niveles como de los retos planteados en estos a partir de una estimación inicial de las habilidades del jugador, y por otra, evaluar la experiencia del jugador y jugabilidad de este durante la consecución de los niveles para alcanzar dinámicamente un compromiso entre el grado de recompensa y grado de frustración.

El resultado que esperamos obtener es un juego que pueda ser jugado por un espectro más amplio de usuarios ya que durante el proceso de juego se adapta a cualquiera de ellos en función de sus habilidades e intereses. Con ello no excluimos a ningún perfil de jugador y creamos entornos que el jugador pueda sentir como suyo y que la aventura que enfrentan como algo único.

El videojuego se construirá con el motor Unity 5. Se desarrollarán una serie de funcionalidades para esta clase de juegos entre las que destacan:

- Clases de personajes diferenciadas: existirán distintos personajes con habilidades únicas al resto de clases.
- Sistema de combate: ciertos retos serán solo superables mediante un combate por equipos organizado por turnos.
- Sistema de evolución de las estadísticas basada en niveles: el nivel del personaje nos servirá para hacernos una idea de cuan fuerte es un personaje con respecto a otro.
- Inventario, objetos y economía: existirán objetos que podremos almacenar en nuestro inventario para usar en momentos concretos de la aventura. Estos podrán comprarse y venderse libremente en su mayoría y el modo de hacerlo será con la moneda del juego.

- Diversas habilidades: nuestros personajes contarán con habilidades que deberán desarrollarse y mejorarse para superar los retos que vayamos enfrentando.
- Historia contada a través de misiones: las misiones que nos asignen otros personajes dentro del juego serán los que hagan progresar la historia.
- Animaciones.

4.1. ¿Por qué Unity 5?

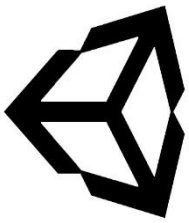


Figura 13.- Logo Unity

Se utilizará el motor Unity 5 dado que es una de las herramientas más extendidas en el mundo de los videojuegos. Es una herramienta potente, comercial y con versión gratuita muy completa. Además, posee una comunidad muy activa que aporta mucha información y soluciones a problemas que pueden surgir durante el aprendizaje del uso de esta herramienta.

A nivel de programación Unity permite hacer scripts en dos lenguajes de programación diferentes como son JavaScript y C#, siendo este último el usado en este proyecto.

A pesar de no poder competir a nivel gráfico con otros motores como Unreal, tras la introducción de la versión 5, este motor que vamos a emplear ha dado un notable salto de calidad en cuanto a potencia. Si a esto le sumamos que el juego que aquí se pretende hacer no tiene una gran carga en ese sentido y que Unity resulta más accesible para comenzar a aprender cómo desarrollar videojuegos todo parece indicar que esta es la opción más adecuada.

4.2. ¿Por qué un RPG?

Los RPGs nos proporcionan un escenario amplio con una gran toma de decisiones por parte del usuario que podemos aprovechar para analizarle en detalle y establecer un perfil detallado con el transcurso de la partida. A esto hay que sumarle que nos permite adaptar a ese perfil ciertos escenarios conocidos como “Mazmorras” en los cuales el jugador debe superar una serie de retos y posteriormente recibe algún tipo de premio, bien sea un objeto, dinero o que la historia continúe. Este premio también podemos adaptarlo dinámicamente con este sistema en función de lo difícil que haya sido superarlo logrando que la relación dificultad premio sea lo más justa posible.

Este amplio espectro de elecciones, combates y escenarios variados nos permite encontrar un sistema rico en información que aprovechar para usar como campo de pruebas del concepto que deseamos desarrollar. Por ello considero que es el género perfecto desde el que establecer las bases para implementarlo en otros géneros del mundo de los videojuegos.

5. Desarrollo del proyecto

5.0. Planificación, Metodología y Presupuesto

En este apartado pasaremos a exponer el desarrollo que se ha realizado dividiéndolo en tres partes:

- Primero explicaremos el documento de diseño del juego (GDD), documento ampliamente utilizado en el mundo de los videojuegos para mostrar los aspectos de diseño a tener en cuenta a la hora de crear un juego.
- La segunda parte de este punto se usará para explicar el desarrollo e implementación del videojuego. En ella explicaremos entre otras cosas las herramientas empleadas y cómo funcionan.
- Por último, haremos una pequeña explicación de la estimación de costes para un proyecto de este tipo.

5.1. Documento de diseño del juego (GDD)

En este apartado se podrán encontrar todo lo relacionado con el diseño teórico del videojuego a nivel de mecánicas, arte e historia. Este apartado es de los más importantes en la industria del videojuego ya que establece las pautas fundamentales sobre las que se basará el resto del desarrollo.

5.1.1. Información sobre el juego

University RPG es un juego de rol inicialmente pensado para ser jugado en PC. En él encarnamos a un personaje que inicia su vida en la universidad y debe ser capaz de sobrevivir a ello. Nuestro personaje se enrola en una organización de estudiantes llamada “La Resistencia”, comandada por cuatro estudiantes, para enfrentar a un malvado grupo llamado “La Cúpula”, el cual se ha hecho con el control de la Universidad del imperio de Ancia, y que desea acabar con el libre albedrío en las universidades de todo el mundo.

Este juego pretende hacer un especial énfasis en la personalización de la aventura para cada jugador, y en que cada uno de ellos viva una experiencia única y satisfactoria.

5.1.2. Visión general

University RPG como ya hemos dicho es un juego de rol. En él emplearemos mecánicas que ya son familiares en estos tipos de como serían la exploración de los distintos escenarios, mejorar

tanto a nuestro personaje principal, como aquellos que consiga reunir para acompañarle, ya sea comprando u obteniendo mejor equipamiento o subiéndolos de nivel para que sus habilidades mejoren, y la de completar misiones o bien principales, las cuales irán avanzando la historia y determinando el rumbo que tomará esta, o bien secundaria, que nos otorgarán recompensas adicionales ya sean en forma de experiencia, lograr objetos nuevos, dinero o incluso algún aliado adicional para el resto de la aventura.

El juego se desarrollará en un entorno de fantasía de corte medieval, como en juegos tales como los primeros Final Fantasy de Squaresoft, los cuales ya hemos introducido previamente, con poderes mágicos, espadas o seres mitológicos, pero con matices futuristas referentes a tecnología actual como podrían ser robots, armas de fuego, ordenadores, laboratorios…

Con el avance del juego deberemos decidir qué clase de personaje queremos que sea nuestro protagonista ya que comienza siendo un personaje “Neutro” ajeno a las demás clases que presentaremos a continuación. Las 5 clases son:

- Dormilón
- Fiestero
- Friki
- Responsable
- Tirano

Además de esto podemos contar la clase “Neutro” como una sexta clase a la que solo nuestro personaje puede pertenecer. Decidiendo una de las 6 clases que se ofrecen, con sus consecuentes beneficios y defectos, determinaremos un estilo de combate que explicaremos en más detalle en el apartado “Clases”.

El corazón de University RPG se basará en la toma de decisiones para que el jugador pueda determinar el rumbo de la aventura a su gusto y ayudar de este modo a la personalización del juego además de con el sistema de análisis que implementaremos. Esto generará una personalización a dos niveles siendo la más visible la que determine la aventura y la segunda menos perceptible para el jugador la que establezca la dificultad y con ello algunos eventos dentro del juego, las recompensas, enfrentamientos y diseño de las mazmorras.

5.1.3. Historia

Como ya dijimos nuestro protagonista encarna a un estudiante que acaba de comenzar la universidad. Él se imaginaba un lugar idílico donde aprender y vivir experiencias nuevas. Sin embargo, nada más poner un pie allí se encuentra un escenario completamente distinto, es una zona de guerra, un lugar gris y lleno de sufrimiento. Es a causa de este escenario que decide unirse a un grupo de estudiantes llamados La Resistencia los cuales le cuentan a que se debe esta situación de conflicto dentro de la universidad. Un malvado grupo llamado La Cúpula se ha hecho con el control de la universidad y pretende acabar con el libre albedrío en el mundo empezando por las universidades. Su plan maligno es crear un ejército de estudiantes zombis que no sientan pasión ni interés por nada y se dediquen a hacer lo que se les dice.

Ahora estará en nuestras manos decidir que queremos hacer, ayudar a La Resistencia en su misión eligiendo a que facción dentro de esta queremos pertenecer (habrá 4 facciones

disponibles, cada una con sus propias misiones, recompensas y nivel de dificultad), dudar de la palabra de este grupo y conocer cuáles son las verdaderas intenciones de La Cúpula y quizás hasta aliarnos con ellos o bien ir por libre y decidir nosotros mismos el devenir del mundo.

5.1.4. Lugares importantes

Habrá una serie de puntos clave para los jugadores. El primero será la universidad, lugar donde transcurrirá la mayor parte de la historia. Este es territorio de la Cúpula el cual será mayoritariamente hostil y que contará con múltiples secciones a explorar y que deberemos enfrentar para avanzar en nuestro camino.

El segundo más importante será la base de La Resistencia. Este será un lugar amistoso, siempre que nos unamos a este grupo o nos mantengamos neutrales. En él se nos encargarán misiones, podremos comprar objetos, curar nuestras heridas entre otras acciones que estarán disponibles.

A parte de estos habrá una serie de lugares que completarán la geografía del juego, tales como ciudades o pueblos conectados por rutas, cuevas o alguna otra clase de elemento. En estas rutas y algunos lugares hostiles habrá las denominadas zonas de monstruos, lugares por los cuales si entramos en ellos podrían aparecer enemigos que nos ataquen y que debemos enfrentar con el sistema de combate que explicaremos en el próximo punto.

También existirán tiendas donde hacernos con una serie de objetos, enfermerías y posadas donde curarnos de nuestras heridas y tabernas donde cambiar los integrantes de nuestro equipo.

Existe un último punto de interés para los jugadores que serán los puntos de guardado. Estos estarán dispersados por todo el mapa y será el lugar donde el jugador puede guardar la partida.

5.1.5. Sistema de combate

El sistema de combate se basa en enfrentamientos por equipos pudiendo constar ambos de uno a tres miembros. Esta norma puede cambiar dependiendo del tipo de personaje que sea nuestro enemigo diferenciando 3 tipos:

- Enemigos pequeños, los cuales seguirían la norma de un equipo compuesto de 1 a tres miembros.
- Enemigos grandes, que pueden formar equipo de hasta dos miembros siendo el segundo miembro otro enemigo grande o uno pequeño.
- Enemigos gigantes, solo podrá formar el equipo un único miembro de esta clase.

Los combates se dividen en 3 fases:

- Decidir acción: el jugador decidirá entre 3 opciones
    - Atacar

        - Se decidirá que ataques lanzarán los miembros de su equipo y cuáles son los objetivos de estos.
    - Usar objeto
        - Se mostrará la lista de objetos en nuestro inventario y podremos elegir cual usar y con quien usar ese objeto.
    - Huir
        - Esta opción no estará disponible en todos los combates y nos dará la opción de abandonar dicho combate.

Figura 14.- Escena de la Fase 1 del combate

- Combate
    - Si se ha decidido usar un objeto se aplica los efectos de este.
    - Después se sucederán los ataques de los personajes que los lancen, primero los de nuestro equipo y después los del enemigo.

Figura 15.- Escena de la Fase 2 del combate

Figura 16.- Escena de selección de objetivo

Figura 17.- Escena de daño

Figura 18.- Daño débil

Dependiendo del tipo de ataque el daño será mayor o menor además de tener en cuenta otros factores como el ataque, la defensa y la velocidad. Si la combinación de elementos es positiva el daño se representará en rojo, si es normal en blanco y si es débil en azul. También se mostrará un x1.5 si la velocidad es más del doble o x2 si es más del triple.

- Fase final
    - De haber sido derrotado uno de los equipos se terminará el combate. Si hemos ganado se nos dará una recompensa en forma de dinero, experiencia para los personajes que han participado y la posibilidad de obtener algún objeto que usar en nuestra aventura, además reapareceremos donde lo dejamos. Si por el contrario hemos sido derrotados se nos impondrá una penalización y deberemos volver al último punto de control. Si la batalla es

contra un jefe, fase llamada “Examen final”, tendremos que pasar por una fase llamada “Examen de recuperación”. Ambas fases las explicaremos en un punto posterior.

- Si el combate no ha finalizado volveremos a la fase uno.

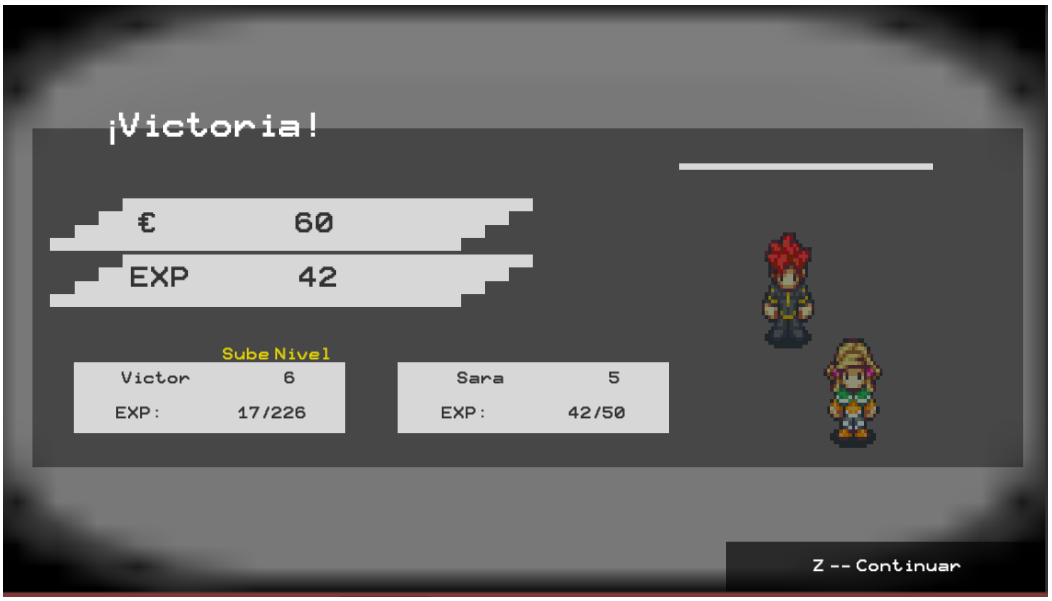


Figura 19.- Escena de victoria

En esta escena se muestra la recompensa lograda, el primer dato que vemos es el dinero que se logra tras la batalla, después la experiencia que recibirá cada personaje. En la parte inferior se muestra el nivel, la experiencia actual y la experiencia necesaria para subir de nivel. Además, cuando suba de nivel un personaje aparecerá un pequeño mensaje. Cuando logremos objetos durante el combate aparecerán en el lado derecho del menú.

5.1.6. Estadísticas de los personajes

Los personajes tienen 7 estadísticas claves para determinar su valor de combate. Estas son:

- Vida: determina cuanto daño podemos aguantar
- Ataque físico: utilizado para calcular cuánto daño generan las habilidades físicas
- Ataque mágico: utilizado para calcular cuánto daño generan las habilidades mágicas
- Defensa física: utilizado para reducir el daño físico que recibimos
- Defensa mágica: utilizado para reducir el daño mágico que recibimos
- Velocidad: determina el número de veces que los ataques golpearán. Esto dependerá de la diferencia de velocidad entre el personaje atacante y el defensor, siendo el mínimo un solo golpe y el máximo cuatro.
- Evasión: probabilidad de evitar un ataque

Estas se pueden mejorar durante el transcurso de la batalla con habilidades de apoyo u objetos además de con el equipo que llevemos equipado al combate.

Figura 20.- Primer prototipo de ficha protagonista

Figura 21.- Ficha del personaje protagonista final

5.1.7. Clases

Como dijimos anteriormente nos encontramos con 6 clases distintas. En este apartado nos encargaremos de definir qué beneficios y perjuicios nos otorgan cada uno:

- **Neutro**:
  - Fuerte contra: -
  - Pros:
    - Puede aprender una mayor variedad de habilidades.
    - Añade una capa extra de dificultad si buscamos un reto mayor.
  - Contras:

        - Es la clase con más debilidades.
        - No es especialmente fuerte contra ningún otro tipo.
- **Dormilón**:
    - Fuerte contra: Fiestero y Tirano.
    - Pros:
        - Mayor capacidad de recuperación de vida.
        - Posee las habilidades más difíciles de contrarrestar.
    - Contras:
        - Personaje complejo de jugar.
        - Ralentiza el estilo de las batallas.
        - Necesita más experiencia para subir de nivel y mostrar su potencial.
- **Fiestero**:
    - Fuerte contra: Neutro, Tirano y Responsable.
    - Pros:
        - Buenas estadísticas de ataque físico.
        - Personaje simple de manejar.
    - Contras:
        - Pocas habilidades de apoyo.
        - Sus ataques pueden causarles perjuicios a si mismo.
- **Friki**:
    - Fuerte contra: Fiestero y Dormilón.
    - Pros:
        - Buenas estadísticas mágicas.
        - Personaje sin estadísticas especialmente débiles en ningún punto.
    - Contras:
        - Estadísticas físicas (ataque físico y defensa física) no son destacables, aunque tampoco son muy malas.
- **Responsable**:
    - Fuerte contra: Neutro, Friki y Dormilón.
    - Pros:
        - Requiere menos experiencia para subir de nivel.
        - Estadísticas equilibradas.
        - Fuerte contra más tipos.
    - Contras:
        - Habilidades menos potentes.
- **Tirano**:
    - Fuerte contra: Neutro, Friki y Responsable
    - Pros
        - Fuerte ataque y defensa
        - Las habilidades más destructivas
    - Contras:
        - El que más experiencia requiere para subir de nivel
        - Malas habilidades de apoyo.

5.1.8. Economía y Objetos

Como ya hemos comentado anteriormente en el apartado “Sistema de Combate” si salimos victorioso de nuestros enfrentamientos contra distintos enemigos lograremos una serie de recompensas, en este caso nos centraremos en dos de ellos, los objetos y el dinero. El dinero será utilizado principalmente en los distintos tipos distintos de tiendas. Estas son un total de tres tipos distintos, una por cada tipo de objeto existente. La primera es la de objetos consumibles, los cuales se usan una única vez y darán alguna clase de beneficio al personaje que lo use de manera temporal. La segunda será para los objetos de equipo, que son aquellos que darán beneficios permanentes a aquellos personajes que tengan equipado ese objeto, estos pueden ser para la cabeza, para el cuerpo, botas, un complemento, un arma y un escudo. La tercera es la de objetos de ataque, aquellos objetos que enseñan una habilidad en concreto a un personaje que sean compatibles según su tipo.

Figura 22.- Tiendas del juego

Las imágenes mostradas encima corresponden a unos ejemplos de cómo lucirían las distintas tiendas, aunque estas pueden ser distintas dependiendo del escenario en el que nos encontremos.

Figura 23.- Menú de compra de objetos

En el menú de la tienda los objetos que sean equipables nos mostrarán información sobre como alteran las estadísticas de nuestros personajes antes de comprarlos o venderlos, así como la cantidad de elementos de ese mismo objeto que tenemos en el inventario ya o cuál es su coste. En el lado izquierdo veremos una pequeña descripción que explicará que hace dicho objeto para que el jugador pueda decidir con todos los datos posibles si le conviene o no comprarlo.

Los objetos también se podrán vender en estas tiendas con un valor que dependerá de su rareza y los beneficios que ofrezca, y de la tienda en la que se venda. Si el objeto es del mismo tipo que el de la tienda en donde se vende esto dará un plus al valor.

El dinero también podrá ser utilizado para acceder o superar algún aspecto en ciertas misiones o para las llamadas “Clases particulares” que serán misiones secundarias que ayudarán a nuestro personaje a entrenar y mejorar sus habilidades si no es capaz de superar ciertos retos.

5.1.9. Enfermerías

Estos lugares nos permitirán curar a los personajes que se encuentren en nuestro equipo en ese momento a cambio de una pequeña cantidad de dinero.

5.1.10. Misiones

Figura 24.- Imagen del menú de misiones

Las misiones son el punto clave para hacer avanzar la historia pues es completando estas que el jugador puede ver cómo evoluciona el juego. Cada misión tiene un nombre identificativo, una descripción, la ubicación de donde se nos asignó, uno o varios objetivos y una o varias

recompensas. La recompensa dependerá de la clase a la que pertenezca. Estas se dividen en tres clases distintas:

- Principal: hacen avanzar la historia y su cumplimiento es obligatorio.
- Secundarias: no son obligatorias y no harán avanzar la historia principal, pero si nos pueden contar historias paralelas o concedernos objetos adicionales además de alargar la vida del juego.
- Reclutamiento: estas misiones no son obligatorias, pero si muy recomendables dado que es el principal método para añadir personajes extras a nuestro grupo lo cual puede ayudar mucho a completar el juego.

5.1.11. Objetos

Los objetos pueden ser logrados de varias formas. La primera y la más directa de ellas sería comprándolas en las tiendas como hemos explicado previamente. Después de manera más indirecta sería gracias a la exploración de los mapas donde pueden haber escondidos distintas clases de cofres los cuales al abrirlos nos otorgarán estos. Para poder abrir estos cofres es necesario encontrar una llave para cada clase de estos. Las clases de los cofres de menor a mayor valor son:

- Común
- Raro
- Mítico
- Legendario

En cuanto encontremos la llave de cada uno de estos podremos abrir todos los que encontremos de esa clase.

La tercera manera de lograr objetos será como resultado de las misiones que como ya hemos comentado se nos otorgarán recompensas al completarlas.

La última será en los combates. Si salimos victoriosos de una existe la posibilidad de lograr algún objeto.

5.1.12. Sistema de diálogos

Se ha implementado un sistema para poder hablar con todos los NPCs (non-player character) los cuales nos podrán dar misiones, objetos, información sobre el juego o conversaciones sobre la narrativa. Estas conversaciones con ciertos personajes también serán empleadas para crear el ya mencionado perfil del jugador.

Figura 25.- Dialogo durante la primera animación del juego

Figura 26.- Dialogo con un NPC que nos ofrece una misión secundaria

5.1.13. Evolución de las estadísticas

La siguiente tabla se corresponde al crecimiento de cada una de las estadísticas por nivel de las diferentes características de los personajes.

RANGO	ESTADÍSTICA
S	x4
A	x3
B	x2
C	x1

5.1.14. Evolución de las clases

- Dormilón

ESTADÍSTICAS	CRECIMIENTO
Vida	S
Ataque Físico	C
Ataque Mágico	B
Defensa Física	B
Defensa Mágica	A
Velocidad	C

- Fiestero

ESTADÍSTICAS	CRECIMIENTO
Vida	C
Ataque Físico	S
Ataque Mágico	C
Defensa Física	B
Defensa Mágica	B
Velocidad	A

- Friki

ESTADÍSTICAS	CRECIMIENTO
Vida	B
Ataque Físico	C
Ataque Mágico	S
Defensa Física	C
Defensa Mágica	A
Velocidad	B

- Neutro

ESTADÍSTICAS	CRECIMIENTO
Vida	B
Ataque Físico	B
Ataque Mágico	B

Defensa Física	B
Defensa Mágica	B
Velocidad	B

- Responsable

ESTADÍSTICAS	CRECIMIENTO
Vida	S
Ataque Físico	B
Ataque Mágico	A
Defensa Física	C
Defensa Mágica	B
Velocidad	A

- Tirano

ESTADÍSTICAS	CRECIMIENTO
Vida	B
Ataque Físico	S
Ataque Mágico	A
Defensa Física	A
Defensa Mágica	A
Velocidad	C

5.1.15. Protagonista

Figura 27.- Protagonista del juego

Explicación del personaje: nuestro protagonista representa a un estudiante que comenzará su andadura en la vida universitaria. Será nuestro avatar durante toda la aventura y será la misión de cada jugador decidir las acciones que tomará para definir su historia.

Estadísticas:

ESTADÍSTICAS	VALOR BASE
Vida	50
Ataque Físico	15
Ataque Mágico	15
Defensa Física	14
Defensa Mágica	14
Velocidad	12

5.1.16. Examen final

Esta fase es la más importante para desarrollar la historia del juego. Nuestro personaje deberá pasar un número determinado, basado en el desarrollo del juego por cada usuario, con un máximo de diez exámenes. Estos propondrán un reto distinto a los jugadores que deberán superar para enfrentarse al jefe final con el sistema de combate que hemos explicado previamente. Cuando entremos a una de estas fases no podremos guardar hasta terminarla dado que si no somos capaces de superar uno de los dos retos habremos suspendido el examen y se nos enviará al examen de recuperación. En esta fase se nos devolverá al último punto de control para prepararnos de nuevo para intentar superar las fases, la primera de ellas puede haber cambiado mientras que la segunda seguirá siendo un combate. Hasta que no enfrentemos la fase de recuperación el resto de misiones quedarán bloqueadas y solo podremos acceder a las zonas de combate o a las “Clases particulares”. En caso que fallemos una de las pruebas habremos perdido el juego y la única manera de volver a empezar será cargar la última partida cargada.

5.1.17. Interfaz

Para hablar de la interfaz del juego haremos un repaso de las diferentes vistas que tiene el juego explicando brevemente cada una de ellas.

Figura 28.- Escena de selección de idiomas

Figura 29.- Escena de selección de idiomas en inglés

Lo primero que veremos al iniciar el juego será la selección de idiomas para los textos tanto de la interfaz como para los diálogos de los personajes. En la parte inferior podemos ver cuáles son los controles que se pueden usar y que funciones realizan. Esto se extenderá a otras vistas del juego. Tras una escena con el título del juego se mostrará el menú inicial del juego.

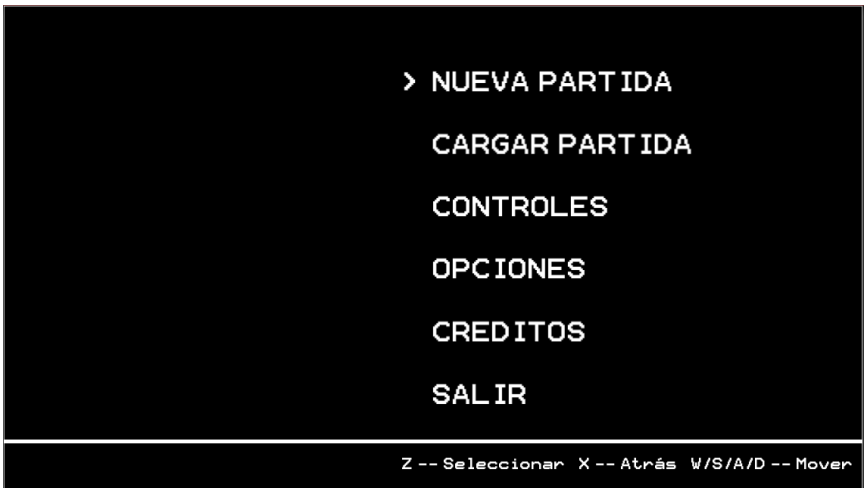


Figura 30.- Menú principal

En este menú podremos elegir entre comenzar una nueva partida, cargar la última partida si existe, ver los controles que se usarán para el juego, la configuración del juego, los créditos del juego y la opción de salir del juego.

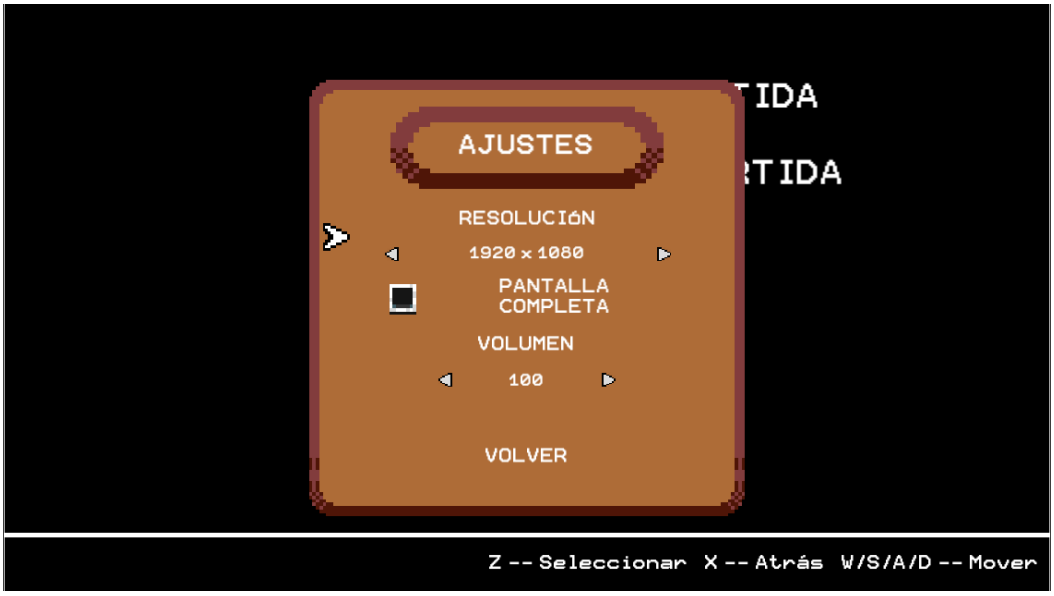


Figura 31.- Menú configuración

En el menú de Configuración podremos ajustar la resolución de la pantalla, si lo queremos en pantalla completa o no y el volumen de la música y los efectos de sonido.

Para la vista principal del juego se han probado diversos diseños que han ido cambiando según se definía el estilo del juego.

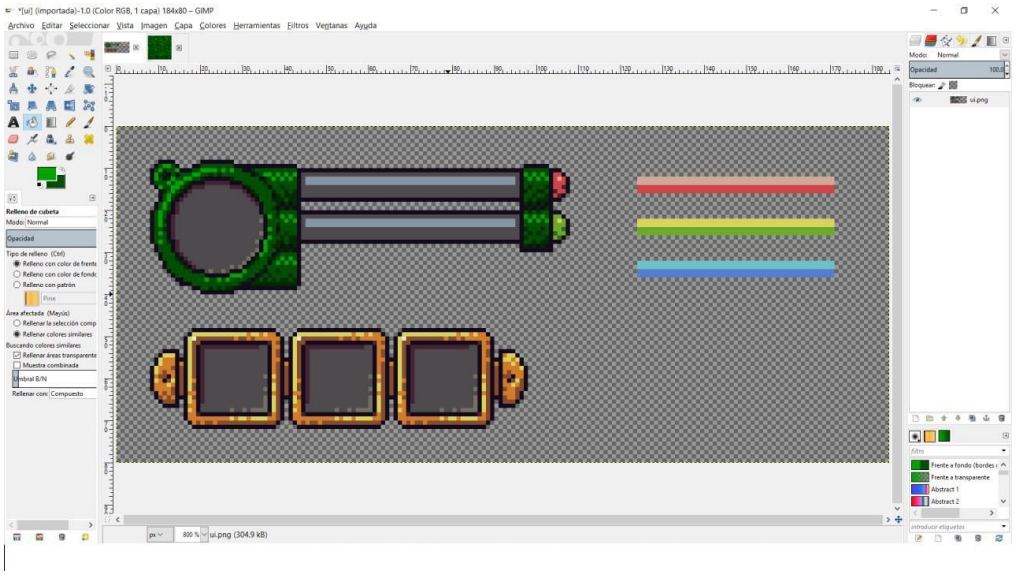


Figura 32.- Prototipo de elementos de interfaz

Como de primeras se pensó en que solo se controlara al personaje protagonista diseñamos unos marcadores para la vida y la experiencia además de objetos clave que irían en la parte superior de la pantalla. Sin embargo, esto se desechó al incluir a varios personajes jugables a la vez lo que saturaba la pantalla de información.

Figura 33.- Muestra del escenario de exploración

Finalmente decidimos que en el escenario de exploración tendríamos una vista limpia para que el jugador disfrute del entorno sin elementos que lo obstaculicen y ocultando la información de la vida u otros elementos de interés al menú de pausa.

Figura 34.- Muestra de información al cambiar de escenario

El único elemento que aparecerá en la escena principal a modo interfaz será un pequeño texto que se desvanece a los pocos segundos cuando entramos en una zona peligrosa del mapa para mostrarnos el nombre de la zona en la que nos encontramos y el nivel mínimo recomendado para enfrentarse a la zona.

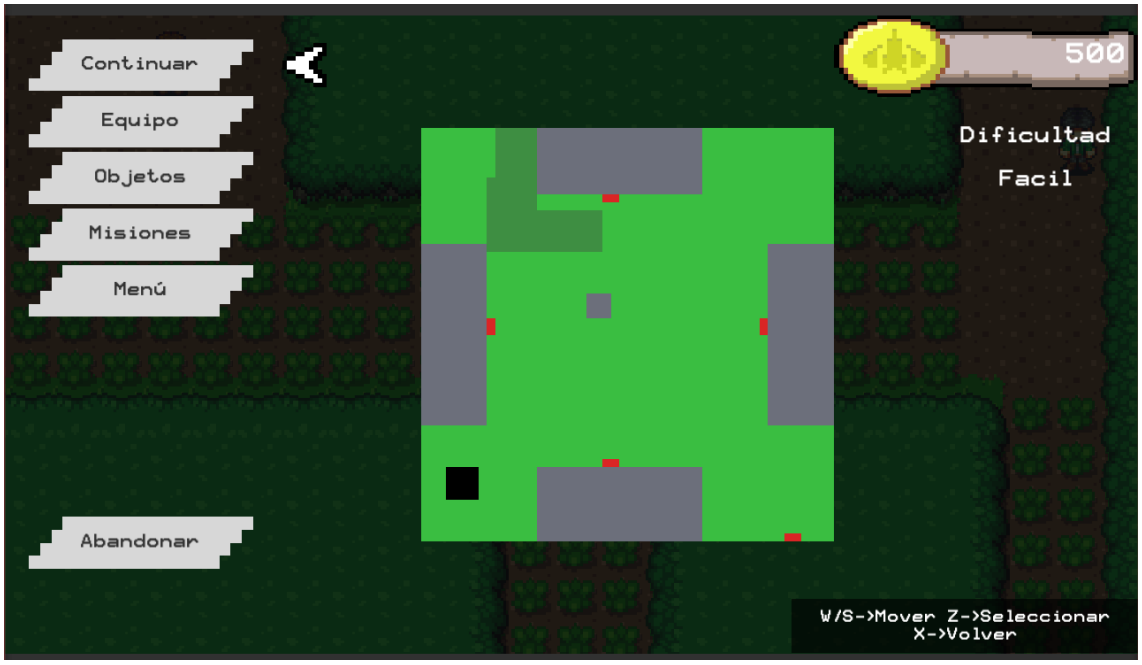


Figura 35.- Muestra del menú de pausa

En el menú de pausa lo primero que vemos son 6 opciones en el lado de izquierdo los cuales sirven para continuar la partida, los miembros de nuestro equipo y sus características que veremos en una imagen posteriormente, el inventario de objetos, el listado de misiones que vimos en la figura 22, una opción para salir al menú principal del juego y finalmente la opción para cerrar el juego. En el centro de la imagen vemos un mini mapa el cual indicará nuestra posición actual y nos dará una primera idea de cómo es el entorno. A la derecha podemos ver cuánto dinero poseemos y la dificultad de juego actual.

Figura 36.- Muestra del menú de equipo

En el menú de equipo podremos ver los miembros que nos acompañan en la aventura, así como sus estadísticas y nos dará acceso a la ficha de cada personaje la cual se mostró previamente en la figura 20.

Finalmente, para terminar por esta muestra de los elementos de la interfaz explicaremos cómo funciona el menú de inventario el cual es uno de los más usados en los juegos de genero RPG.

Figura 37.- Muestra del menú de inventario

En este menú podemos ver cuatro clases de objetos distintos. La primera clase son los objetos Consumibles los cuales se usan una única vez y su efecto es inmediato, en esta clase se incluirían objetos tales como los curativos o los que potencien las habilidades de algún modo. Esta clase de objetos son los únicos que se pueden usar durante una batalla. La segunda clase es la de los objetos de Equipo. Estos son aquellos que se les puede equipar a los personajes para potenciar sus habilidades. La tercera clase es la de objetos de Ataque los cuales se emplean para aprender ataques nuevos que usar durante los combates. Y finalmente la cuarta clase se trata de los Objetos Clave. Estos objetos se irán desbloqueando en el transcurso de la historia y nos permitirá abrir cofres de mayor nivel, desbloquear nuevos escenarios u hacer avanzar la historia en algunos puntos.

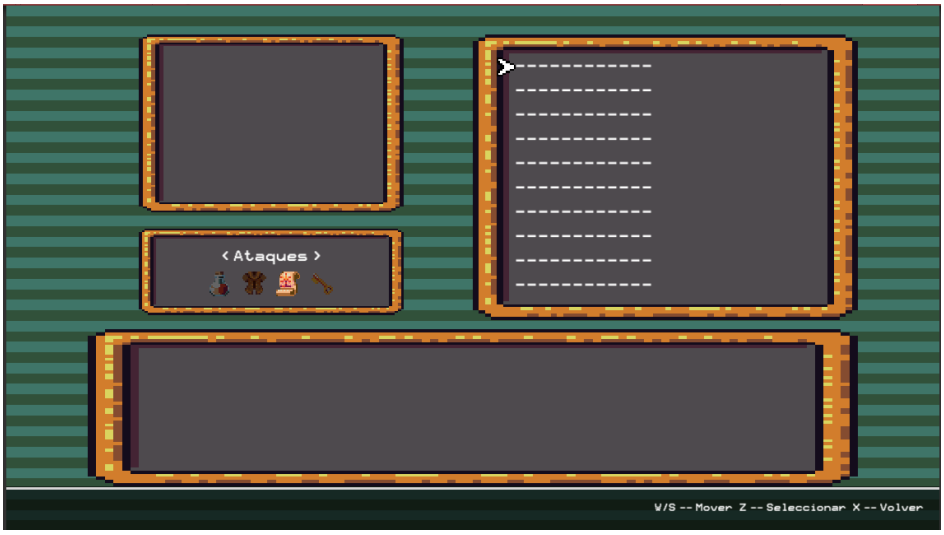


Figura 38.- Muestra del menú de inventario sin objetos

En caso de que no tengamos ningún objeto de alguna de las clases no se verá ni la imagen, ni la descripción de ningún objeto.

5.1.18. Inteligencia Artificial

Este es uno de los puntos más complejos de desarrollar y calibrar ya no solo en este proyecto debido a su enfoque adaptativo sino a cualquier videojuego que requiera de la interacción con otros agentes autónomos. Esto se hace aún más delicado si cabe en el género que nos encontramos dada la gran importancia que tienen los combates contra personajes controlados por la IA durante el transcurso de todo el juego.

Para este proyecto la IA debe ser capaz de decidir qué ataque emplear cada turno que dure el combate y contra que objetivo. Los ataques puedes ser ofensivos o de apoyo bien sea para mejorar sus estadísticas o la de algún aliado si lo tuviera o bien reducir las estadísticas de alguno de los personajes de nuestro equipo. Para lograr este objetivo, en el turno del enemigo se clasifican los objetivos a los que se le va a lanzar el ataque de cada miembro del equipo enemigo atendiendo a tres prioridades siendo la prioridad 1 la más eficaz contra el objetivo que se va a lanzar y la prioridad 3 la menos eficaz. Para organizar estas prioridades se le da la prioridad 1 al enemigo cuyo tipo es débil contra el tipo del lanzador, prioridad 2 si es neutra y 3 si es fuerte contra él lanzador. El resto de personajes del equipo enemigo, si los hay, se les dará prioridad 2 si el lanzador tiene algún ataque de apoyo positivo, en caso contrario no se incluirán en estas prioridades. Posteriormente se clasificarán los ataques que el lanzador puede usar dividiéndolas en el mismo sistema de prioridades. Los ataques de apoyo siempre se les dará prioridad 2.

Una vez tenemos todos los elementos clasificados se generará un número aleatorio de 0 a 100 para determinar a qué prioridad se va a atender para elegir el objetivo. La posibilidad de elegir una de las tres prioridades cambiará en función de la dificultad de la partida, dentro de esta se elegirá el elemento que haya en esa prioridad si es único o se volverá a generar un nuevo número aleatorio de 0 al número de elementos que haya con esa prioridad. Por poner un ejemplo un jugador que llegue al combate en dificultad Fácil tendrá más posibilidades de que la IA decida elegir como objetivo a un personaje cuyos ataques no serán tan eficaces o que emplee un ataque de apoyo contra él o alguno de los miembros del equipo enemigo, mientras que alguien que llegue a ese mismo combate en dificultad Titán (la más alta del juego), verá como los objetivos son elegidos de manera más agresiva. Esto mismo ocurrirá con el ataque empleado, se elegirá primero la prioridad y después el elemento a escoger dentro de la misma.

Con ello se pretende lograr un combate lo más balanceado posible dependiendo del nivel de dificultad en el que nos encontremos y permitir una experiencia satisfactoria para cada jugador que enfrente ese desafío.

5.2. Requisitos

En este apartado se describirán los requisitos funcionales, no funcionales del juego y de información. Con el fin de facilitar la tarea del lector primeramente se va a explicar brevemente la estructura de cada uno de los requisitos, así como los campos que lo componen.

Cada requisito está representado con una tabla como la siguiente:

Identificador	
Título	
Descripción	
Prioridad	☐Alta ☐Media ☐Baja
Necesidad	☐Esencial ☐Deseable ☐Opcional

Donde cada campo significa lo siguiente:

- Identificador: Código que identifica de forma única y unívoca cada uno de los requisitos.
- Título: Texto descriptivo del requisito.
- Descripción: Explicación breve del requisito que se está especificando.
- Prioridad: Orden de cumplimiento de un requisito. A mayor prioridad, mayor urgencia para realizarlo.
- Necesidad: Interés de los usuarios en la realización de un requisito.

5.2.1 Requisitos funcionales

Las siguientes tablas se corresponden con los requisitos funcionales que se han extraído del videojuego desarrollado:

Identificación: RSF-01	
Título	Comenzar nueva partida
Descripción	El sistema deberá permitir iniciar una nueva partida siempre que el usuario así lo desee desde el menú principal.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-02	
Título	Continuar partida guardada
Descripción	El juego deberá ofrecer la posibilidad al usuario de continuar con una partida guardada de existir esta. Al cargar se debe mostrar en: - Posición en el mapa del jugador. - Aliados desbloqueados. - Desarrollo de los personajes desbloqueados - Objetos logrados. - Cofres abiertos. - Misiones completadas. - Dificultad. - Dinero.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-03	
Título	Comenzar nueva partida
Descripción	El sistema deberá permitir iniciar una nueva partida siempre que el usuario así lo desee desde el menú principal.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-04	
Título	Mostrar controles juego
Descripción	El sistema deberá permitir que el jugador pueda consultar cuales son los controles del mismo.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-05	
Título	Ajustar configuración del juego
Descripción	El sistema deberá permitir ajustar resolución, modo de visualización y volumen de la música.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-06	
Título	Mostrar créditos
Descripción	El sistema deberá permitir mostrar los créditos del juego cuando el usuario lo desee consultar.
Prioridad	☐Alta ☐Media ☒Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-07	
Título	Cerrar el juego
Descripción	El sistema deberá permitir ser cerrado en cualquier momento que el usuario ya no desee continuar jugando.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-08	
Título	Guardar partida
Descripción	El sistema deberá permitir al jugador guardar su progreso para que la partida pueda ser posteriormente cargada desde ese punto.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-09	
Título	Sistema de dialogo
Descripción	El sistema deberá permitir que el jugador pueda interactuar con los NPCs presentes en el juego.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-10	
Título	Aceptar/Rechazar misiones
Descripción	El sistema deberá permitir al jugador decidir aceptar o no una misión para poder tener retos.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-11	
Título	Pausar el juego
Descripción	El sistema deberá permitir pausar el juego cuando nos encontremos en la escena de exploración para mostrar el menú.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-12	
Título	Mostrar equipo
Descripción	El sistema deberá permitir mostrar el menú de equipo con la información de cada uno de los miembros de nuestro equipo. En él se debe ver: - Vida del personaje. - Experiencia del personaje - Ataque físico y mágico. - Defensa física y mágica. - Velocidad. - Equipo que lleva cada personaje.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-13	
Título	Ficha del personaje
Descripción	El sistema deberá permitir mostrar la ficha de cada personaje aliado con: - Nombre del personaje. - Estadísticas del personaje. - Objetos que se llevan equipados. - Ataques del personaje.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-14	
Título	Inventario
Descripción	El sistema deberá permitir mostrar y usar los objetos que se han ido logrando a lo largo de la aventura, así como la cantidad de esos objetos o una breve descripción.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-15	
Título	Listado de misiones
Descripción	El sistema deberá permitir mostrar el listado de misiones pendientes y completadas para poder tener un registro del transcurso de la historia.
Prioridad	☐Alta ☐Media ☒Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-16	
Título	Abrir cofres
Descripción	El sistema deberá permitir al jugador abrir cofres de distinto tipo para obtener objetos.
Prioridad	☐Alta ☐Media ☒Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-17	
Título	Iniciar combate
Descripción	El sistema deberá permitir iniciar combates ya sea por estar en una zona de combate o bien por ser un momento de la historia donde haya un combate.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-18	
Título	Obtener recompensa combate
Descripción	El sistema deberá otorgar una recompensa en modo de dinero, experiencia y objetos.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-19	
Título	Calcular dificultad
Descripción	El sistema deberá ser capaz de calcular la dificultad en el que se desarrollará el juego.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-20	
Título	Ejecutar tienda
Descripción	El sistema deberá permitir iniciar la tienda dentro del juego para poder comprar y vender objetos.
Prioridad	☐Alta ☐Media ☒Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-21	
Título	Comprar/Vender objetos
Descripción	El sistema deberá ser capaz de ofrecer la posibilidad de comprar/vender objetos en la tienda cuando el jugador así lo desee.
Prioridad	☐Alta ☐Media ☒Baja
Necesidad	☐Esencial ☐Deseable ☒Opcional

Identificación: RSF-22	
Título	Cambio de escenario
Descripción	El sistema deberá ser capaz de cambiar de escenario cuando se llegue al extremo de un mapa o a la entrada/salida de un edificio.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-23	
Título	Adaptar mapas
Descripción	El sistema deberá ser capaz de cambiar las mazmorras disponibles dependiendo del nivel de dificultad del propio juego.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-24	
Título	Adaptar mapas
Descripción	El sistema deberá ser capaz de cambiar las mazmorras disponibles dependiendo del nivel de dificultad del propio juego.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSF-25	
Título	Reproducir animaciones
Descripción	El sistema deberá ser capaz de reproducir animaciones de los siguientes tipos: - Fragmentos de historia. - Movimiento de los personajes. - Combates.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-26	
Título	Reproducir sonido
Descripción	El sistema deberá ser capaz de reproducir diferentes tipos de música dependiendo de la zona en la en que se encuentre y sonidos varios al interactuar con ciertas clases de elementos.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSF-27	
Título	Subir nivel
Descripción	El sistema deberá ser capaz de subir de nivel a los personajes que hayan logrado la cantidad de experiencia necesaria para ello y actualizar las estadísticas de dicho personaje.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

5.2.2 Requisitos no funcionales

Las siguientes tablas se corresponden con los requisitos no funcionales que se han extraído del videojuego desarrollado:

Identificación: RSNF-01	
Título	Compatibilidad con distintos Sistemas Operativos (SO)
Descripción	El sistema deberá ser compatible con varios SO y varias versiones del mismo.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSNF-02	
Título	Compatibilidad con resoluciones
Descripción	El sistema deberá ser compatible con distintas resoluciones de pantalla y de diferenciar entre modo Pantalla Completa y modo en Ventana.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSNF-03	
Título	Compatibilidad con formatos de audio
Descripción	El sistema deberá ser compatible con distintos formatos de audio como mp3 o WAV.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSNF-04	
Título	Mostrar textos legibles
Descripción	El sistema se adaptará a la resolución de las pantallas y mostrará los textos de manera legible para el usuario.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSNF-05	
Título	Diferentes idiomas
Descripción	El sistema deberá ofrecer en que idioma deseamos jugar el juego.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

Identificación: RSNF-05	
Título	Interfaz auto explicativa
Descripción	La interfaz debe de ser capaz de explicarle al usuario como funciona a través de ella misma.
Prioridad	☐Alta ☒Media ☐Baja
Necesidad	☐Esencial ☒Deseable ☐Opcional

5.2.3 Requisitos de información

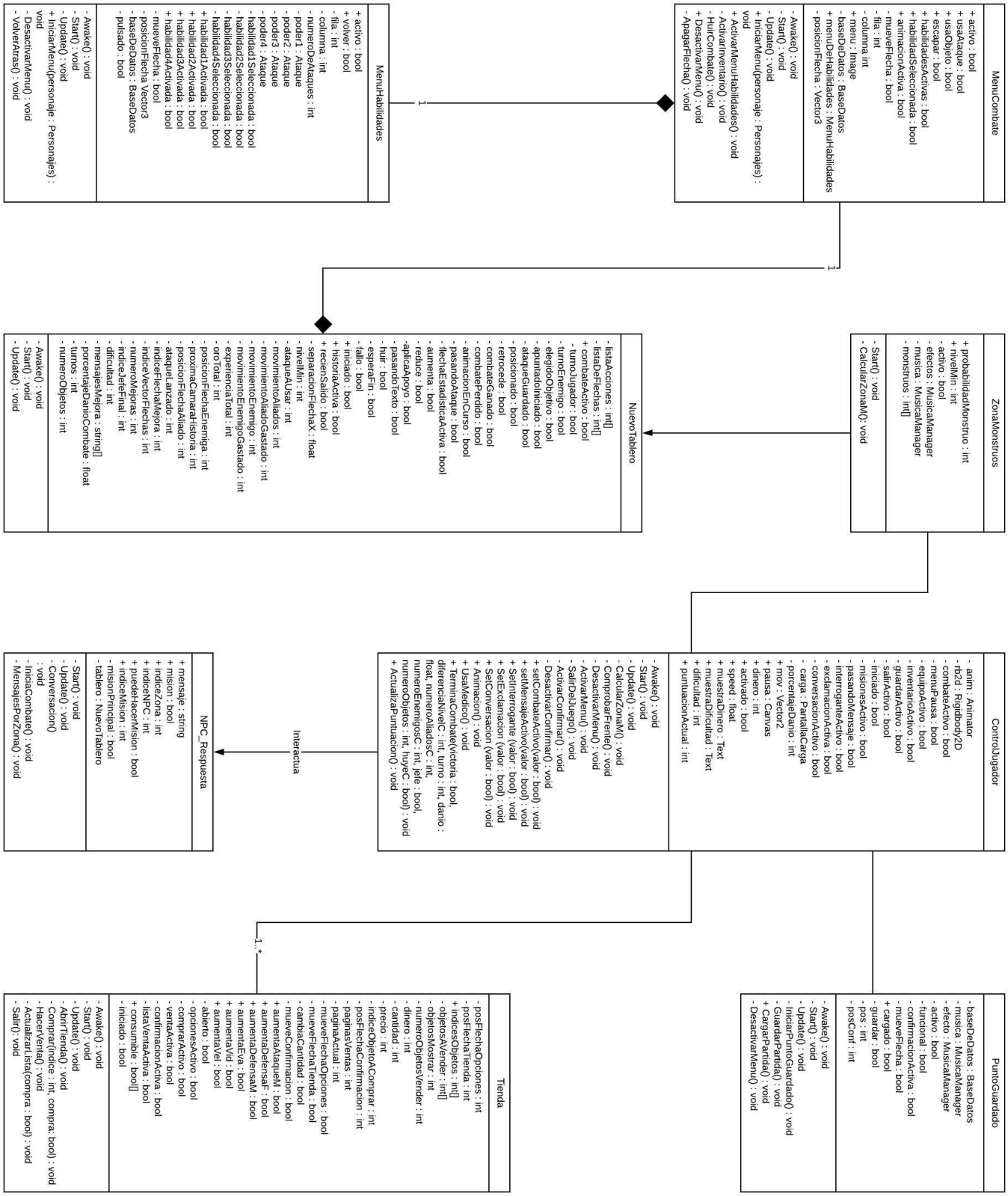
Las siguientes tablas se corresponden con los requisitos de información que se han extraído del videojuego desarrollado:

Identificación: RSI-01	
Título	Almacenamiento de partida guardada
Descripción	El sistema deberá ofrecer guardar la partida. Para ello será necesario.
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

Identificación: RSI-02	
Título	Perfil del jugador
Descripción	El sistema deberá ofrecer el perfil de dificultad para el jugador que ya ha iniciado una partida
Prioridad	☒Alta ☐Media ☐Baja
Necesidad	☒Esencial ☐Deseable ☐Opcional

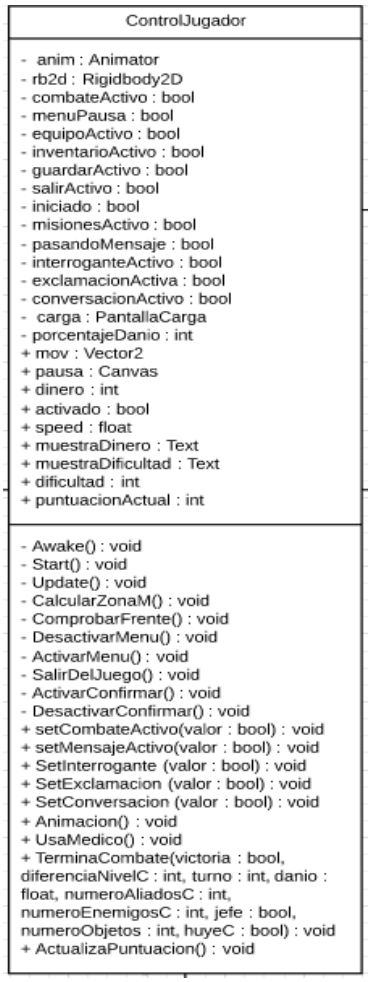
5.3. Diagrama de clases

En este apartado se muestra el diagrama de clases para el control del jugador y otros elementos con los que interactúa.



En los siguientes apartados iremos explicando una a una cada clase para su mejor comprensión.

5.3.1. Control Jugador

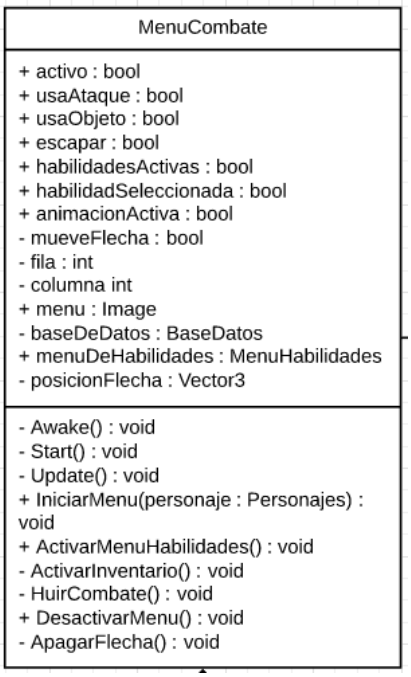


Esta clase se encarga del control del personaje protagonista. En él tenemos las animaciones de movimiento y algunas otras como la interacción con otros elementos como los cofres o los NPCs.

Esta clase también gestiona el menú “Pause” y da acceso a todas sus opciones entre ellas la de cerrar el juego o mostrar el dinero que tenemos en ese momento como ya vimos anteriormente.

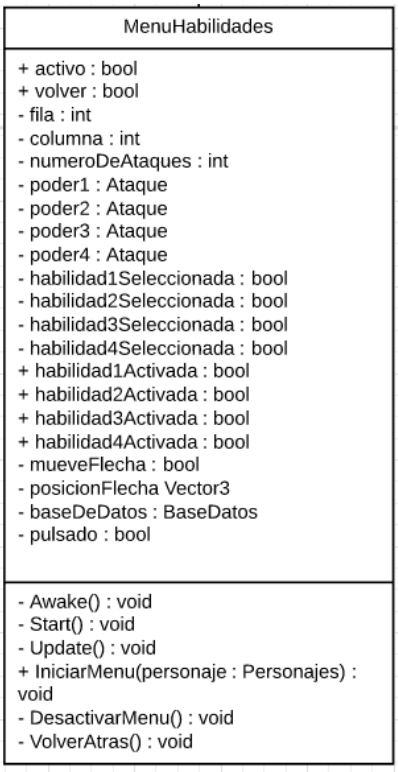
Finalmente, también realiza las comprobaciones de si se inicia un combate cuando nos encontramos en una de las zonas habilitadas para ello. En caso afirmativo bloquea al personaje, cambia la cámara y se da inicio al combate.

5.3.2. Menu Combate



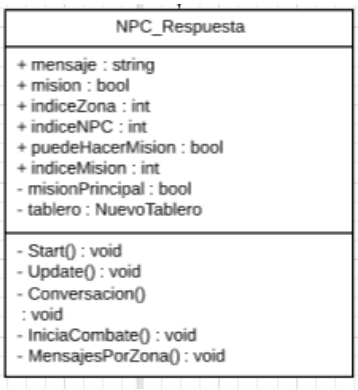
Esta clase se encarga de gestionar las opciones de combate. La opción “Atacar” abre el menú de habilidades, “Objetos” abre el inventario de objetos y “Huir” comprueba si es posible escapar del combate en el que nos encontramos en ese momento.

5.3.3. Menu Habilidades



Esta clase se encarga de la elección de la habilidad que vamos a emplear en este turno. Su labor es mostrarnos primero las habilidades del personaje que va a lanzar el ataque, así como información de la energía y el tipo y posteriormente darle la información a la clase NuevoTablero sobre que ataque se ha seleccionado.

5.3.4. NPC

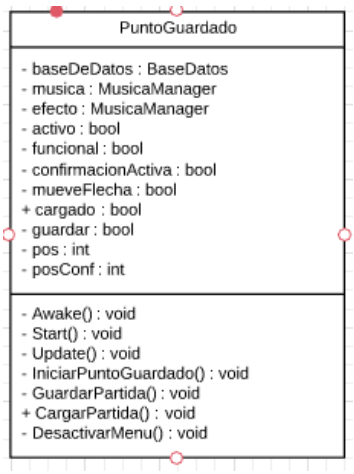


Esta clase se encarga de gestionar las conversaciones con los NPCs. Muestra el cuadro de texto como el contenido de este mostrando una animación de escritura letra a letra. También activa la animación del personaje del jugador en la que se muestra una exclamación si el mismo está cerca del NPC.

5.3.5. Nuevo Tablero

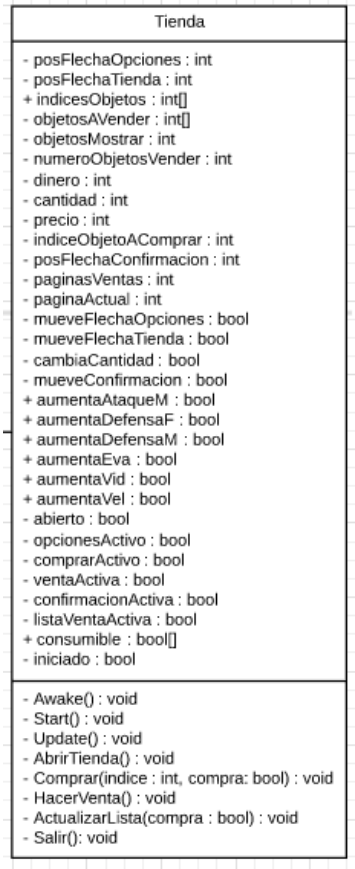
Esta clase es la encargada de gestionar todos los aspectos del combate. Guarda la información de los personajes que participan en el combate, gestiona los turnos y las animaciones de combate, otorga las recompensas si se logra superar o termina el juego si por el contrario salimos derrotados.

5.3.6. Punto Guardado



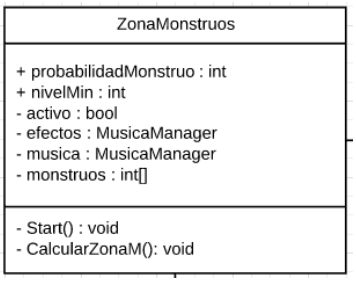
Esta clase se encargará de guardar la partida, almacenando los datos esenciales para que se pueda continuar desde ese punto. También permitirá cargar una partida anterior siempre que se desee.

5.3.7. Tienda



La clase Tienda gestiona la compra/venta de objetos, quita/incluye el dinero cuando se realiza una transacción, muestra el nombre y la información de cada objeto en la interfaz y muestra las estadísticas en cada personaje en comparación con el ya equipado en caso de haberlo.

5.3.8. ZonaMonstruos



La clase ZonaMonstruos se encarga de calcular si al movernos por una de las zonas de aparición de monstruos se iniciará un combate o no. En caso afirmativo NuevoTablero iniciará su funcionamiento.

5.4. Metodología empleada

En este proyecto se ha optado por uno de los sistemas más empleados dentro del marco de las metodologías ágiles, el Scrum. En periodos de 3-4 semanas se establecía uno o varios objetivos que se debían completar o dejar parcialmente establecidos en caso de que dependieran a su vez de elementos que se desarrollaran en ciclos posteriores. Una vez realizado el Sprint se probaba que estos elementos funcionaran correctamente con el resto del sistema ya desarrollado. En caso de que todo funcionara correctamente se establecían nuevos objetivos y se comenzaba de nuevo el ciclo. En caso contrario se trataba de solucionar los errores encontrados durante la prueba antes de comenzar una nueva iteración.

Gracias a esta metodología podíamos organizar temporalmente el desarrollo del proyecto, así como llevar un control del estado del desarrollo en comparación con lo planificado. Además, nos permite asegurarnos que el producto en el momento de iniciar una nueva iteración funciona correctamente lo cual facilita la tarea de solucionar fallos que se puedan surgir en iteraciones posteriores.

5.5. Presupuesto

En este apartado haremos una breve estimación de los costes que tendría un proyecto como el que hemos planteado. Lo dividiremos en coste de personal, coste de software, coste de hardware y finalmente presentaremos los costes agrupados en un resumen para ver el gasto total. Se ha estimado que el proyecto ha ocupado unas 1080 horas (9 meses trabajando durante 4 horas diarias).

5.5.1 Coste de personal

Para este proyecto se ha supuesto necesario un programador para el desarrollo del código en C# y realizar las tareas necesarias en Unity. Un diseñador para establecer la disposición de los niveles, apariencia del entorno y mecánicas del juego. Un dibujante para implementar la visión

del diseñador con un estilo Pixel Art. Y finalmente el encargado de la documentación para establecer la documentación final del producto.

Puesto	Número de horas	Coste hora (€)	Total (€)
Programador	800	12,5	10.000
Diseñador	70	10	700
Dibujante	100	10	1.000
Documentación	30	8	240
Musico	80	9	720
			12.660

5.5.2 Coste de software

Para el desarrollo del producto ha sido necesario disponer de diferentes herramientas software. Se ha optado por emplear principalmente herramientas con versión gratuita, pero algunos otros se han optado por la versión de pago aun habiendo otras similares gratuitas por mayor facilidad o mejores prestaciones.

Producto	Coste (€)
Windows 10 Home	145
Microsoft Office 365 Home	99
GIMP 2.10.8	0
Unity Personal	0
Visual Studio 2017	0
Audacity	0
Lucidchart	0
	244

5.5.3 Coste de hardware

Producto	Coste (€)
Monitor ordenador	65
Ratón ordenador	10
Teclado	10
Ordenador Portátil	1.000
Cable HDMI	5
Auriculares	20
	1110

5.5.4 Gasto total

Elemento	Coste (€)
Personal	12.660
Software	244
Hardware	1.110
	14.014

Teniendo en cuenta el 21% de I.V.A. obtenemos la siguiente tabla:

Descripción	Coste (€)
Total sin I.V.A.	14.014
21% I.V.A.	2942,94
	16956,94

6. Implementación

A continuación, se presentarán algunos elementos del código que se ha implementado para este proyecto. Dada su gran extensión solo se mostrarán ciertos fragmentos del mismo de algunas de las clases principales.

- **ControlJugador**

```
void Update()
{
    if (!carga.activo)
    {
        digitalX = Input.GetAxis("D-Horizontal");
        digitalY = Input.GetAxis("D-Vertical");

        mov = new Vector2(0, 0);
        if (!TextBox.on && !combateActivo && !menuPausa && !pasandoMensaje)
        {
            speed = 2f;

            if (Input.GetKey(KeyCode.RightShift) || Input.GetKey(KeyCode.LeftShift) || Input.GetButton("X"))
            {
                speed *= 1.3f;
            }

            if (Input.GetKeyUp(KeyCode.RightShift) || Input.GetKeyUp(KeyCode.LeftShift) || Input.GetButtonUp("X"))
            {
                speed /= 1.3f;
            }


            if (Input.GetKey(KeyCode.W) || Input.GetKey(KeyCode.UpArrow) || digitalY > 0)
            {
                mov = new Vector2(0, speed);
                mover = movimiento.ARRIBA;
            }
            else if (Input.GetKey(KeyCode.S) || Input.GetKey(KeyCode.DownArrow) || digitalY < 0)
            {
                mov = new Vector2(0, -speed);
                mover = movimiento.ABAJO;
            }
            else if (Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.LeftArrow) || digitalX < 0)
            {
                mov = new Vector2(-speed, 0);
                mover = movimiento.IZQUIERDA;
            }
```

```
        }
        else if (Input.GetKey(KeyCode.D) || Input.GetKey(KeyCode.RightArrow) || digitalX > 0)
        {
            mov = new Vector2(speed, 0);
            mover = movimiento.DERECHA;
        }
        else if (Input.GetKeyDown(KeyCode.Z) || Input.GetButtonUp("A"))
        {
            ComprobarFrente();
        }
        else if (Input.GetKeyDown(KeyCode.Escape) || Input.GetButtonUp("Start"))
        {
            muestraDinero.text = "" + dinero;

            if (dificultad == 1)
            {
                muestraDificultad.text = "Facil";
            }
            else if (dificultad == 2)...
            else if (dificultad == 3)...
            else if (dificultad == 4)...
            else if (dificultad == 5)...
            else if (dificultad == 6)...
            else if (dificultad == 7)...

            this.ActivarMenu();
        }
    }
    else if (menuPausa)
    {
        if (pulsado)
        {
            if (digitalY == 0 && digitalX == 0)
            {
                pulsado = false;
            }
        }

        if (salirActivo)
```

```
{
    if (Input.GetKeyDown(KeyCode.A) || Input.GetKeyDown(KeyCode.LeftArrow) || (digitalX < 0 && !pulsado))
    {
        pulsado = true;
        musica.ProduceEfecto(11);
        posFlechaConf--;
        if (posFlechaConf < 0)
        {
            posFlechaConf = 1;
        }
    }
    else if (Input.GetKeyDown(KeyCode.D) || Input.GetKeyDown(KeyCode.RightArrow) || (digitalX > 0 && !pulsa

    if (posFlechaConf == 0)
    {
        flechaConf.transform.position = posConf1.transform.position;
    }
    else...

    if (Input.GetKeyDown(KeyCode.Z) || Input.GetButtonUp("A"))
    {
        musica.ProduceEfecto(10);
        if (posFlechaConf == 0)
        {
            if (posFlecha == 4)
            {
                SceneManager.LoadScene("Portada");
            }
            else
            {
                SalirDelJuego();
            }
        }
        else if (posFlechaConf == 1)
        {
            DesactivarConfirmar();
        }
    }
    else if (Input.GetKeyDown(KeyCode.X) || Input.GetButtonUp("B"))...
}
```

```
    }
    else if (equipoActivo)
    {
        if (!equipo.activo && !iniciado)
        {
            equipo.ActivarMenu();
            iniciado = true;
            pausa.gameObject.SetActive(false);
        }
        else if (!equipo.activo && iniciado)
        {
            iniciado = false;
            equipoActivo = false;
            pausa.gameObject.SetActive(true);
        }
    }
    else if (misionesActivo)
    {
        if (!misiones.activo && !iniciado)
        {
            misiones.IniciaMenu();
            iniciado = true;
            pausa.gameObject.SetActive(false);
        }
        else if (!misiones.activo && iniciado)
        {
            iniciado = false;
            misionesActivo = false;
            pausa.gameObject.SetActive(true);
        }
    }
    else if (inventarioActivo)
    {
        if (!inventario.activo && !iniciado)
        {
            inventario.IniciarMenu(false);
            iniciado = true;
            pausa.gameObject.SetActive(false);
        }
        else if (!inventario.activo && iniciado)
```

```
    else if (!inventario.activo && iniciado)
    {
        iniciado = false;
        inventarioActivo = false;
        pausa.gameObject.SetActive(true);
    }
}
else
{
    if (pulsado)
    {
        if(digitalY == 0 && digitalX == 0)
        {
            pulsado = false;
        }
    }

    if (Input.GetKeyDown(KeyCode.W) || Input.GetKeyDown(KeyCode.UpArrow) || (!pulsado && digitalY > 0))
    {
        pulsado = true;
        musica.ProduceEfecto(11);
        posFlecha--;
        if (posFlecha < 0)
        {
            posFlecha = 5;
        }
    }
    else if (Input.GetKeyDown(KeyCode.S) || Input.GetKeyDown(KeyCode.DownArrow) || (!pulsado && digitalY < 0))...

    if (Input.GetKeyDown(KeyCode.X) || Input.GetKeyDown(KeyCode.Escape) || Input.GetButtonUp("B") || Input.GetButtonUp("Start"))
    {
        musica.ProduceEfecto(12);
        this.DesactivarMenu();
        posFlecha = 0;
    }
    else if (Input.GetKeyDown(KeyCode.Z) || Input.GetButtonUp("A"))
    {
        musica.ProduceEfecto(10);
        if (posFlecha == 0)
        {
```

```
                {
                    DesactivarMenu();
                }
                else if (posFlecha == 1)
                {
                    equipoActivo = true;
                }
                else if (posFlecha == 2)
                {
                    inventarioActivo = true;
                }
                else if (posFlecha == 3)
                {
                    misionesActivo = true;
                }
                else if (posFlecha == 4 || posFlecha == 5)
                {
                    ActivarConfirmar();
                }
            }

            if (posFlecha == 0)
            {
                flecha.transform.position = pos1.transform.position;
            }
            else if (posFlecha == 1)...
            else if (posFlecha == 2)...
            else if (posFlecha == 3)...
            else if (posFlecha == 4)...
            else if (posFlecha == 5)...
        }
```

Figura 39.- Método Update clase ControlJugador

- **NPC_Respuesta**

```
private void Update()
{
    if(Input.GetKeyDown(KeyCode.Z) || Input.GetButtonUp("A"))
    {
        if (TextBox.ocultar)
        {
            if (!mision)
            {
                switch (indiceMision)
                {
                    case 1:
                        IniciaCombate();
                        break;
                    case 9:
                        IniciaCombate();
                        break;
                }
            }
        }

    }
}
```

Figura 40.- Método Update clase NPC_Respuesta

```
void Conversacion(){
    if (!TextBox.on)
    {
        if(baseDeDatos.idioma == 0)
        {
            MensajesPorZona();
        }
        else if(baseDeDatos.idioma == 1)
        {
            MensajesPorZonaIngles();
        }


        if (mision)
        {
            TextBox.MuestraTextoConMision(mensaje, false, true, misionPrincipal, indiceMision);
        }
        else
        {
            TextBox.MuestraTexto(mensaje, false);
        }
    }
}
```

Figura 41.- Método Conversación clase NPC_Respuesta

- **ZonaM**

```
void CalcularZonaM()
{
    if (activo)
    {
        if (!tablero.recienSalido)
        {
            int aux = Random.Range(0, 500);

            if (aux <= probabilidadMonstruo)
            {
                Personajes enemigo1, enemigo2, enemigo3;
                aux = Random.Range(0, 100);
                int aux2 = Random.Range(0, monstruos.Length);

                int numeroEnemigos = 1;

                enemigo1 = baseDeDatos.BuscarPersonajeIndice(monstruos[aux2], false);
                enemigo2 = null;
                enemigo3 = null;

                if (aux > 30)
                {
                    aux2 = Random.Range(0, monstruos.Length);
                    enemigo2 = baseDeDatos.BuscarPersonajeIndice(monstruos[aux2], false);
                    numeroEnemigos = 2;
                }

                if (aux > 80)
                {
                    aux2 = Random.Range(0, monstruos.Length);
                    enemigo3 = baseDeDatos.BuscarPersonajeIndice(monstruos[aux2], false);
                    numeroEnemigos = 3;
                }


                AnimacionCombate.IniciaCombate();
                efectos.ProduceEfecto(1);
                tablero.IniciarCombate(enemigo1, enemigo2, enemigo3, numeroEnemigos, nivelMin, false, -1, -1);
                musica.CambiaCancion(11);
```

```
                musica.CambiaCancion(11);
            }
            else
            {
                StartCoroutine(Espera());
            }
        }
        else
        {
            StartCoroutine(EsperaSalida());
        }
    }
```

Figura 42.- Método Update clase ZonaM

- **NuevoTablero**

```
void TerminaCombate(bool victoria, bool huir)
{
    int nivelesEnemigos = 0;
    int nivelesAliados = 0;
    int auxVida = 0;

    if (controlSecundarias.misionPedroActiva)...

    if (controlSecundarias.misionGamezActiva)
    {
        if (victoria && indiceJefeFinal == 1)
        {
            baseDeDatos.listaMisionesReclutamiento[controlSecundarias.posicionMisionGamez].subMisionCompletada[0] = true;
            baseDeDatos.listaMisiones[controlSecundarias.posicionGeneralMisionGamez].subMisionCompletada[0] = true;
            controlSecundarias.Ladron.SetActive(false);
            controlJugador.SetConversacion(false);
        }
    }


    for (int i = 0; i < movimientoAliados; i++)
    {
        nivelesAliados += datosDelPersonaje[i].nivel;
        auxVida += datosDelPersonaje[i].vidaModificada;
    }

    for(int i = 0; i < movimientoEnemigo; i++)
    {
        nivelesEnemigos += datosDelPersonaje[3 + i].nivel;
    }

    int diferenciaNivel = nivelesEnemigos - nivelesAliados;

    porcentajeDanioCombate = (porcentajeDanioCombate / auxVida) * 100;

    if (historiaActiva)
    {
        if (victoria)
        {
            controlJugador.TerminaCombate(true, diferenciaNivel, turnos, porcentajeDanioCombate, movimientoAliados, movimientoEnemigo, true, numeroObjetos, false);

            GanaCombate();
        }
        else
        {
            controlJugador.TerminaCombate(false, diferenciaNivel, turnos, porcentajeDanioCombate, movimientoAliados, movimientoEnemigo, true, numeroObjetos, false);

            StartCoroutine(EsperaGameOver());
        }
    }
    else
    {
        if (victoria)
        {
            controlJugador.TerminaCombate(true, diferenciaNivel, turnos, porcentajeDanioCombate, movimientoAliados, movimientoEnemigo, false, numeroObjetos, false);

            GanaCombate();
        }
        else if (huir)
        {
            StartCoroutine(EsperaFinCombate());
            controlJugador.TerminaCombate(false, diferenciaNivel, turnos, porcentajeDanioCombate, movimientoAliados, movimientoEnemigo, false, numeroObjetos, false);
        }
        else
        {
            controlJugador.TerminaCombate(false, diferenciaNivel, turnos, porcentajeDanioCombate, movimientoAliados, movimientoEnemigo, false, numeroObjetos, false);

            StartCoroutine(EsperaGameOver());
        }
    }

    TextBox.OcultaTextoFinCombate();
}
```

Figura 43.- Método TerminaCombate clase NuevoTablero

7. Pruebas

Para la realización de este proyecto se realizaron pruebas con usuarios en el momento que se tuvo un producto jugable. Las pruebas se hicieron con 13 personas diferentes de manera simultánea. Los usuarios estuvieron jugando de 2 a 4 horas en los que no se les dio información alguna para comprobar que la interfaz y los controles eran entendibles si la intermediación de agentes externos. Con esta prueba se pretendían conseguir varios objetivos.

- Detección y corrección de errores.
- Comprobar la comprensión de las mecánicas del juego.
- Evaluar si los jugadores se sentían cómodos con el reto que el juego les proponía.
- Comprobar si los controles sugeridos eran cómodos para el jugador.
- Recibir comentarios o sugerencias para mejorar la experiencia.

Cuando esta prueba se completó se les pasó una encuesta para comprobar cuales habían sido los resultados de esta prueba. Las preguntas realizadas con sus respuestas fueron las siguientes:

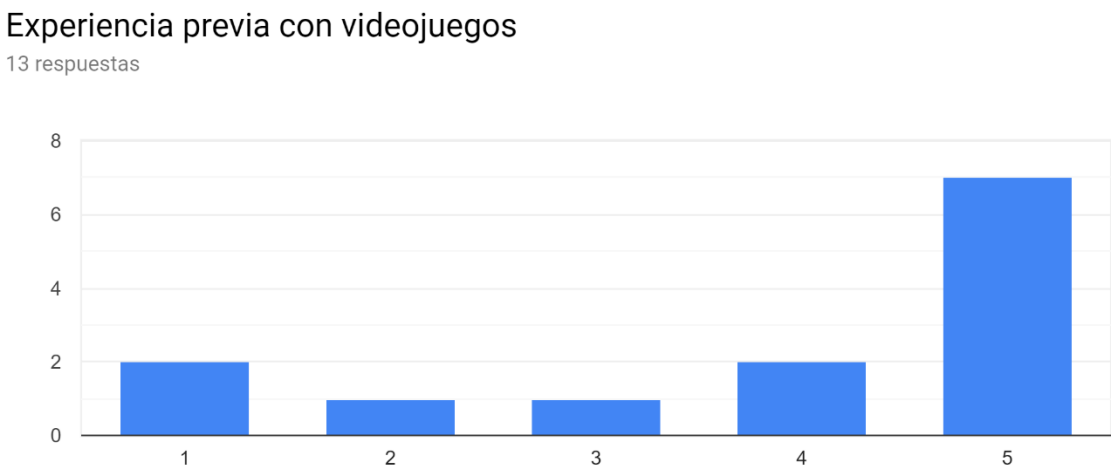


Figura 44.- Gráfica sobre experiencia de juego

Figura 45.- Gráfica sobre experiencia con juegos similares

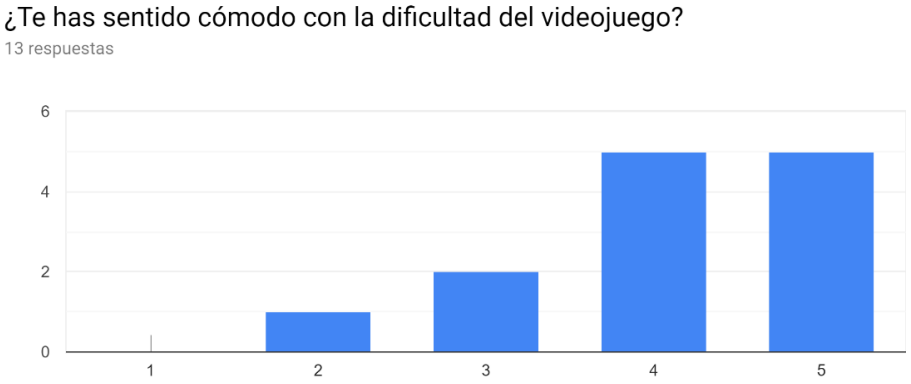


Figura 46.- Gráfica comodidad con la dificultad

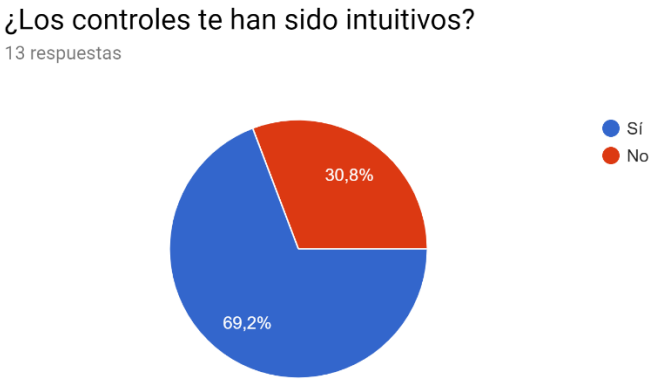


Figura 47.- Gráfica comodidad con los controles

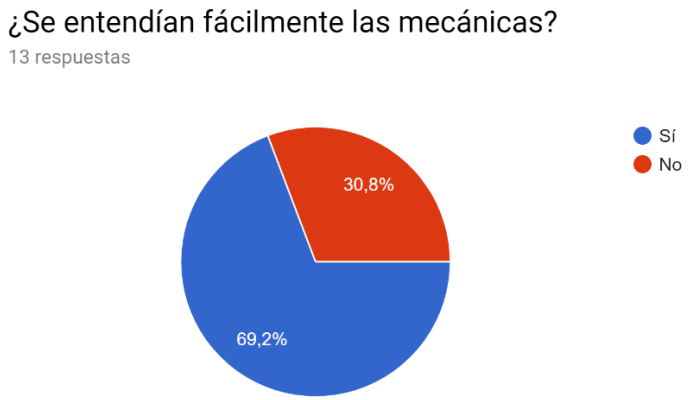


Figura 48.- Gráfica entendimiento de las mecánicas

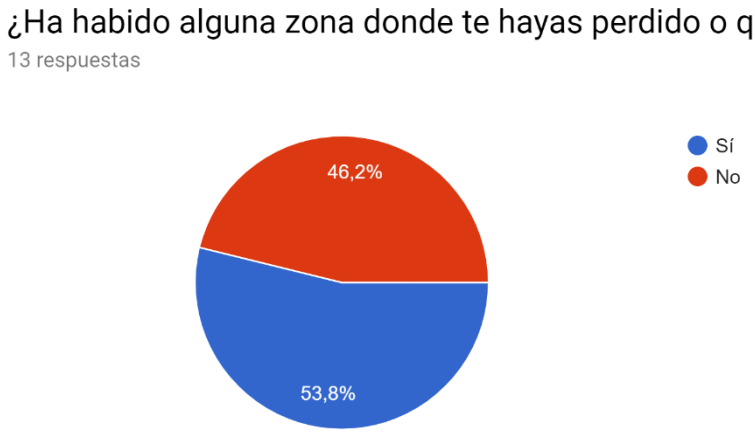


Figura 49.- Gráfica sobre entendimiento de objetivos

Además de estas preguntas también se les preguntó sobre qué cambios harían en los apartados con los que no se sentían cómodos y se pudo comprobar que los problemas encontrados eran generalmente los mismos con lo que logramos detectar dicho problema y solucionarlo. Con estos resultados concluimos que el proyecto lograba de manera bastante satisfactoria los objetivos marcados de ofrecer un reto a medida de distintos perfiles de usuario y con los comentarios de los usuarios de prueba considero que se ha logrado un mayor equilibrio que aumentaría la respuesta positiva.

8. Conclusiones y Trabajos Futuros

Este proyecto ha supuesto una enorme dedicación por mi parte y un enorme sacrificio de tiempo y situaciones estresantes donde todo parecía que se iba a caer de un momento a otro. Sin embargo, la constancia y esfuerzo han dado frutos y estoy muy orgulloso del proyecto presentado y todo lo aprendido con ello.

A pesar de ello lo aquí presentado no es más que un primer paso, un proyecto en una edad muy temprana y que aún requiere de mucho esfuerzo en varios aspectos que me gustaría comentar para finalizar este documento.

Lo primero es el contenido del mismo. A pesar de poder dar, dependiendo del jugador, entre 1 y 3 horas de juego, el contenido que ofrece el juego es aún muy escaso en comparación con lo que se tiene previsto para el que son más de 40 horas con la posibilidad de repetirlo desde el principio obteniendo una experiencia completamente nueva dada la gran cantidad de opciones que ofrecerá el juego.

El segundo elemento que se debe mejorar en el futuro es el sistema de combate. En esta etapa del proyecto se ha mostrado una versión bastante básica sin apenas opciones. El objetivo que se marca este proyecto es lograr un combate con mayor variedad de estrategias y una IA aún más capacitada para responder a estas nuevas opciones. Con ello se pretende lograr un juego aún más interesante en todas sus facetas.

Otro elemento que se debe mejorar es el elemento audiovisual, así como añadir nuevos idiomas para que sea más atractivo y más gente pueda acceder a él de manera sencilla. Tanto la calidad de los dibujos, elementos con los que interactuar, animaciones y sonido debe ser mejorado para lograr una versión más pulida.

Finalmente se debe mejorar el análisis del jugador y la capacidad de adaptarse al él. Este era el tema principal de nuestro proyecto y que a pesar de haber logrado un nivel aceptable aún se debe afinar más y realizar más pruebas con más jugadores de distintos perfiles.

Es de este modo que se logrará alcanzar un punto que pueda ofrecer una versión satisfactoria y que pueda llegar a ser comercializada en las principales plataformas de distribución digital para PC como serían Steam o GOG.

9. Bibliografía

Unity Documentation [Online]. Available at:

https://docs.unity3d.com/Manual/Unity2D.html

Juegos de rol [Online]. Available at:

https://es.wikipedia.org/wiki/Juego_de_rol

Videojuegos de rol [Online]. Available at:

https://es.wikipedia.org/wiki/Videojuego_de_rol#Consolas

Jowser (2018). La dificultad en los videojuegos, su integración y su importancia. Available at:

https://www.sonyers.com/articulo-la-dificultad-en-los-videojuegos-su-integracion-y-su-importancia/

MARTINPIXEL (2017). Hellblade, el videojuego que puede borrar tu partida si mueres varias veces. Available at:

https://www.xataka.com.mx/videojuegos/hellblade-el-videojuego-que-puede-borrar-tu-partida-si-mueres-varias-veces

Omega Boost [Online]. Available at:

https://es.wikipedia.org/wiki/Omega_Boost

The Pokémon Company [Online]. Available at:

https://www.pokemon.com/es/videojuegos-pokemon/

Pokémon [Online]. Available at:

https://es.wikipedia.org/wiki/Pok%C3%A9mon

Wiki Pokémon [Online]. Available at:

https://pokemon.fandom.com/es/wiki/WikiDex

Final Fantasy [Online]. Available at:

https://es.wikipedia.org/wiki/Final_Fantasy_(franquicia)

Octopath Traveller [Online]. Available at:

https://es.wikipedia.org/wiki/Octopath_Traveler

Eder32 (2018). Square Enix pide perdón por subestimar la enorme demanda de Octopath Traveler. Available at:

https://www.nintenderos.com/2018/07/square-enix-pide-perdon-por-subestimar-la-enorme-demanda-de-octopath-traveler/

Unity Engine [Online]. Available at:

https://unity3d.com/es/unity?_ga=2.65882414.1422029669.1560599627-795306594.1544210142

Unity [Online]. Available at:

https://es.wikipedia.org/wiki/Unity_(motor_de_juego)

Unreal Engine Webpage [Online]. Available at:

https://www.unrealengine.com/en-US/

Unreal Engine [Online]. Available at:

https://es.wikipedia.org/wiki/Unreal_Engine

Yoyo Games [Online]. Available at:

https://www.yoyogames.com/gamemaker

GameMaker Studio [Online]. Available at:

https://es.wikipedia.org/wiki/GameMaker_Studio

Godot Engine [Online]. Available at:

https://godotengine.org/

Godot [Online]. Available at:

https://es.wikipedia.org/wiki/Godot

Diablo, Game Concept by Condor, Inc. [Document]. Available at:

https://www.graybeardgames.com/download/diablo_pitch.pdf

Documento de diseño de videojuegos [Online]. Available at:

https://es.wikipedia.org/wiki/Documento_de_dise%C3%B1o_de_videojuegos

Scrum (desarrollo de software) [Online]. Available at:
https://es.wikipedia.org/wiki/Scrum_(desarrollo_de_software)

Clinton Keith. Scrum for Video Game Development [Online]. Available at:

https://es.slideshare.net/clintonnkeith/scrum-for-video-game-development-41126492

Fundamentos de Ingeniería del Software, profesor Miguel Vega López. Tema 2.1 “Introducción a la Ingeniería de requisitos”.

Finalbossblues (2018) [Image]. Available at:

https://www.gamedevmarket.net/member/finalbossblues/

ArMM1998 (2017) [Image]. Available at:

https://opengameart.org/content/zelda-like-tilesets-and-sprites

Komiku (2018) [Music]. Available at:

http://freemusicarchive.org/music/Komiku/

Anexo I. Ejecución Juego

Anexo I.1. Windows

1. Descomprimir el fichero “PruebaURPG”.
2. Abrir la carpeta “PruebaURPG”.
3. Ejecutar el elemento “University RPG”.

Anexo I.2. Linux

1. Descomprimir el fichero “PruebaURPGLinux”.
2. Abrir la carpeta “PruebaURPGLinux”.
3. Ejecutar el elemento “URPG.x86_64”.

Anexo II. Manual de Usuario

1. Seleccionamos el idioma en el que deseamos jugar. Para esta versión solo es funcional la versión en español.

Figura 50.- Imagen de la selección de idioma

2. Seleccionamos en el menú la opción que nos interese.

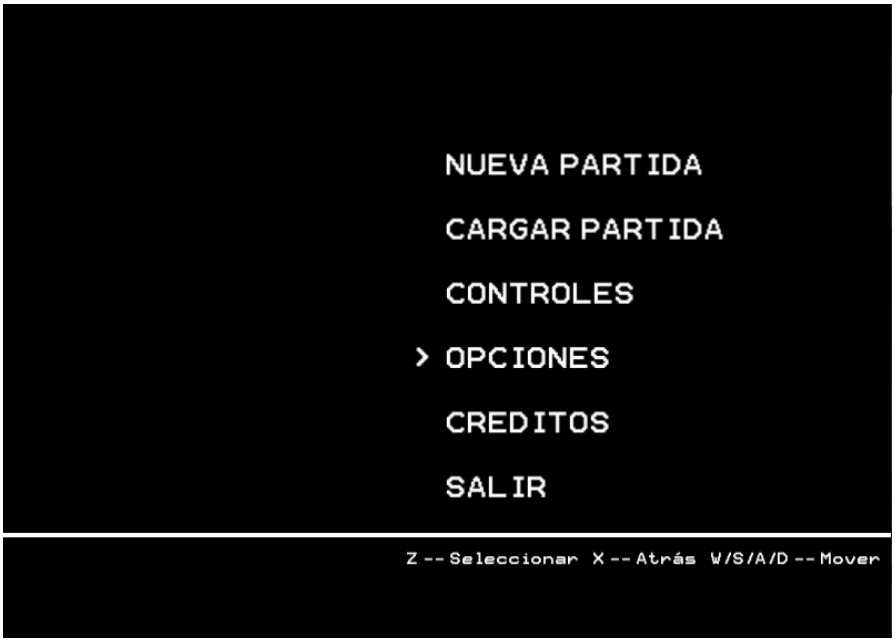


Figura 51.- Imagen del menú inicial

a. Nueva Partida: Inicia un nuevo juego desde cero.
b. Cargar Partida: Recupera la partida que hemos iniciado anteriormente.
c. Controles: Nos muestra los controles del juego.
    i. Controles teclado:
        1. Z → Seleccionar, interactuar y confirmar.
        2. X → Cancelar, rechazar, correr (manteniendo pulsado).
        3. W/A/S/D o flechas de dirección → Mover personaje y flecha de selección.
        4. Esc → Pause.
    ii. Control mando XBOX
        1. A → Seleccionar, interactuar y confirmar.
        2. B → Cancelar, rechazar, correr (manteniendo pulsado).
        3. Cruceta → Mover personaje y flecha de selección.
        4. Start → Pause.

3. Podemos interactuar con aquellos personajes en los que aparezca el siguiente cuadro pulsando el botón de interacción.

Figura 52.- Imagen cuadro de texto

4. En las zonas con hierba como las de la imagen siguiente podrán aparecer monstruos lo cual dará inicio a un combate.

Figura 53.- Imagen de la zona de monstruos