



ugr | Universidad
de **Granada**

TRABAJO FIN DE GRADO
GRADO EN INGENIERÍA INFORMÁTICA

Desarrollo y despliegue de un juego geolocalizado para dispositivos android

Autor

Blas Varela López

Directores

Luis Francisco Gutiérrez Vela
Patricia Paderewski Rodríguez



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y DE
TELECOMUNICACIÓN

—
Granada, mes de 201

Desarrollo y despliegue de un juego geolocalizado para dispositivos android

Blas Varela López

Palabras clave: palabra_clave1, palabra_clave2, palabra_clave3,

Resumen

Poner aquí el resumen.

Project Title: Project Subtitle

First name, Family name (student)

Keywords: Keyword1, Keyword2, Keyword3,

Abstract

Write here the abstract in English.

Yo, **Blas Varela López**, alumno de la titulación de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI XXXXXXXXXX, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Blas Varela López

Granada a X de mes de 201 .

D. **Luis Francisco Gutiérrez Vela**, Profesor del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Granada.

Dña. **Patricia Paderewski Rodríguez**, Profesora del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Granada.

Informan:

Que el presente trabajo, titulado ***Desarrollo y despliegue de un juego geolocalizado para dispositivos android***, ha sido realizado bajo su supervisión por **Blas Varela López**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a X de mes de 201 .

Los directores:

Luis Francisco Gutiérrez Vela Patricia Paderewski Rodríguez

Agradecimientos

Poner aquí agradecimientos...

Índice general

1. Introducción	1
1.1. Definición del proyecto	1
1.2. Motivación	1
1.3. Objetivos principales	1
1.4. Estructura del proyecto	2
2. Estado del arte	5
2.1. Evolución de los RPG	6
2.1.1. El inicio del género	6
2.1.2. Los primeros RPG comerciales	7
2.1.3. Los RPGs en Japón	8
2.1.4. Los RPGs multijugador	8
2.2. Juegos geolocalizados en el mercado actual	9
2.2.1. Pokémon Go	10
2.2.2. Ingress	10
2.2.3. Orna: A Geo-RPG	10
2.3. Geolocalización	11
2.4. Conclusión	12
3. Análisis inicial del problema	13
3.1. Concepto inicial	13
3.1.1. Narrativa	13
3.1.2. Combate	14
3.1.3. Facciones	15
3.1.4. Clases	16
3.1.5. Objetos	16
3.1.6. Misiones	16
3.1.7. Elementos del mapa	17
4. Tecnología a usar	19
4.1. Elección del motor gráfico	19
4.1.1. Unity	19
4.1.2. Unreal Engine	20

4.1.3. Godot Engine	21
4.1.4. CryEngine	21
4.1.5. CoronaSDK	22
4.1.6. Conclusión	23
4.2. Slim	23
4.3. Mapbox	24
4.4. Arquitectura inicial	25
5. Metodologías a usar en el proyecto	27
5.1. Diseño centrado en el usuario	27
5.1.1. Principios del diseño centrado en el usuario	27
5.2. Modelo de prototipos	29
5.3. Metodologías ágiles	29
5.4. Aplicación de las metodologías en el proyecto	29
6. Plan de entregas	31
6.1. Breve descripción del alcance del sistema	31
6.2. Descripción de las entregas	32
6.3. Historias de usuario	32
7. Desarrollo. Entregas e iteraciones	35
7.1. Entrega 1	35
7.1.1. Bocetos iniciales	35
7.1.2. Caracterización del jugador	38
7.2. Entrega 2	40
7.2.1. Mapbox	40
7.2.2. Mapa del mundo	42
7.2.3. Estructura	45
7.2.4. Pruebas	45
7.3. Entrega 3	46
7.3.1. Tienda	46
7.3.2. Gremio	48
7.3.3. Menú	50
7.4. Entrega 4	55
7.4.1. Ejemplo de la actualización de las coordenadas	59
7.5. Entrega 5	60
7.5.1. Combate	60
7.5.2. Chat	66
8. Conclusiones y trabajo futuro	67
Bibliografía	67
Apéndices	70
Documento de diseño de videojuegos	71

Índice de figuras

2.1. Interfaz del RPG Moria	7
2.2. Gráficos del juego Advanced Dungeons & Dragons	8
2.3. Combate en Final Fantasy	9
2.4. Mapa de Ingress	11
4.1. Logo de Unity	20
4.2. Logo de Unreal Engine	21
4.3. Logo de Godot Engine	21
4.4. Logo de Cry Engine	22
4.5. Logo de CoronaSDK	22
4.6. Logo de Slim	24
4.7. Logo de Mapbox	24
4.8. Mapbox Studio	25
4.9. Arquitectura inicial	26
5.1. Estándar ISO 9241	28
7.1. Boceto de la creación de la cuenta	36
7.2. Boceto de la creación del personaje	36
7.3. Boceto del mapa de juego	37
7.4. Boceto del combate	37
7.5. Creación del personaje	39
7.6. GameObjects de Mapbox	40
7.7. Opciones del mapa	42
7.8. Mapa del mundo	44
7.9. Ventana de filtros	44
7.10. Escena de la tienda	46
7.11. Compra en la tienda	47
7.12. Creación de objetos	48
7.13. Catálogo de objetos	48
7.14. Misión no aceptada	49
7.15. Misión completada	49
7.16. Inventario	51
7.17. Inventario con objeto usable seleccionado	51

7.18. Inventario con objeto equipable seleccionado	52
7.19. Misiones aceptadas	53
7.20. Habilidades aprendidas	54
7.21. Creación de habilidades	54
7.22. Catálogo de habilidades	55
7.23. Base de datos	56
7.24. API REST	56
7.25. Método PUT	58
7.26. Método GET	58
7.27. Información necesaria coordenadas	59
7.28. Actualización de las coordenadas en el servidor	60
7.29. Captura de la información obtenida del servidor	61
7.30. Actualización de los elementos mostrados en el mapa	61
7.31. Escena de combate	62
7.32. Log combate	63
7.33. Calculo probabilidad de encuentro	63
7.34. Jefes en el mapa	64
7.35. Calculo de la distancia entre dos coordenadas	64
7.36. Recompensas combate	65
7.37. Escena de chat	66

Índice de cuadros

6.1.	Plan de entregas	32
6.2.	Historia de usuario: Definir un personaje	33
6.3.	Historia de usuario: Realizar una misión	33
7.1.	Elementos del mundo	43

Capítulo 1

Introducción

1.1. Definición del proyecto

En este proyecto se desarrollara un videojuego geolocalizado del género RPG utilizando en su construcción el motor gráfico Unity.

Como juego geolocalizado se construirá un sistema en el que se muestra una representación de la realidad en el mapa del juego de modo que al movernos en la realidad también lo haremos en el juego. Al ser un RPG contara con las principales características del género como niveles, objetos, habilidades, combates contra enemigos, tiendas y misiones.

1.2. Motivación

En los últimos años se ha producido un crecimiento de la popularidad de los videojuegos geolocalizados motivado principalmente por *Pokemon Go*, esto ha provocado que gente ajena a los videojuegos acabara acercándose a los mismos.

Debido a esto creo que es buena idea fusionar un género como los RPG, que permite una progresión del personaje, con los juegos geolocalizados, consiguiendo así un estilo de juego diferenciado y una progresión que motiva a seguir jugando tal y como lo hace *Pokemon Go*.

Este proyecto me permitirá por lo tanto aprender a desarrollar con un motor gráfico, mejorar mis habilidades programando y aumentar mis conocimientos a la hora de desarrollar un servidor gracias a la construcción del sistema multijugador del juego.

1.3. Objetivos principales

Los principales objetivos en el desarrollo del proyecto son:

- Investigar el tipo de juego a desarrollar y reunir sus características principales.
- Definir las principales mecánicas que se utilizaran en el videojuego y la narrativa en la que estará basado.
- Investigar y seleccionar las herramientas que se utilizaran en el desarrollo del proyecto.
- Construir los bocetos de las interfaces que se desarrollaran.
- Desarrollar el videojuego de forma local en Unity.
- Desarrollar el servidor y la interacción con el videojuego.
- Conseguir un prototipo y evaluarlo.

1.4. Estructura del proyecto

El contenido del proyecto listado por capítulos es el siguiente:

- El capítulo 1 (**Introducción**) muestra una texto con la descripción inicial del proyecto y sus objetivos principales.
- El capítulo 2 (**Estado del arte**) informa de los antecedentes teóricos del tipo de juego propuesto y un estudio de los principales videojuegos geolocalizados en el mercado actual incluyendo aquellos que ya han fusionado la geolocalización y el género RPG.
- El capítulo 3 (**Análisis inicial del problema**) muestra el concepto de juego y las principales características con las que contara el mismo.
- El capítulo 4 (**Tecnología a usar**) enseña las principales herramientas utilizadas para el desarrollo del proyecto y una arquitectura inicial del mismo.
- El capítulo 5 (**Metodologías a usar**) muestra las metodologías de desarrollo utilizadas en el proyecto y una justificación del uso de las mismas.
- El capítulo 6 (**Plan de entregas**) incluye el alcance del sistema, la tabla con las entregas y las historias de usuario.
- El capítulo 7 (**Desarrollo. Entregas e iteraciones**) presenta lo obtenido en las entregas propuestas en el plan de entregas y un diagrama de clases de la aplicación generada en Unity.

- El capítulo 8 (**Conclusiones y trabajo futuro**) reflexiona sobre el desarrollo del proyecto y lo obtenido del mismo y presenta posibles mejoras del mismo en el futuro.
- En la sección de **Apéndices** se muestra el Documento de diseño de videojuegos (**GDD**) el cual detalla el concepto de juego y sus características.

Capítulo 2

Estado del arte

Los juegos geolocalizados han presentado un auge en los últimos años liderado por Pokémon Go, estos han definido una nueva forma de juego en la que la localización del usuario en el mundo pasa a ser algo esencial dentro del juego puesto que influirá en el estado del mismo, permitiendo realizar unas acciones u otras dependiendo del lugar donde se encuentre el jugador.

Este concepto hace que los jugadores tengan que jugar al aire libre fomentando por lo tanto la actividad física al tener que desplazarse continuamente para poder cumplir los objetivos. Sin embargo esto mismo a provocado algunos problemas al distraer a los jugadores en plena calle, ocasionando estos incidentes como robos.

Existe por lo tanto un componente social en el que varios jugadores se reúnen en facciones y necesitan de trabajo en equipo para poder afrontar ciertos retos, influyendo los mismos además en el estado del mundo y generando este hecho una competitividad entre las facciones.

Por otro lado el género RPG (*role-playing games*) es un tipo de videojuego que comparte características con los juegos de rol tradicionales, este género ha evolucionado mucho a lo largo del tiempo apareciendo además muchas variantes, pero en general suelen compartir una serie de características.

En los juegos RPG [1] el jugador asume el rol de uno o varios personajes que irán evolucionando a lo largo de la aventura, esta evolución suele realizarse al ganar diferentes tipos de puntos en los combates los cuales son gastados en habilidades o estadísticas y al subir de nivel. Suele existir un amplio abanico de artículos y accesorios que ayudarán en las diferentes etapas del juego. Por último suelen existir misiones mediante las cuales se avanza en la historia y por las que se reciben recompensas.

2.1. Evolución de los RPG

Este género de videojuegos nació basándose en los juegos de rol tradicionales y adaptando la mayoría de sus características como la creación de personajes, armamento, niveles, misiones y otras mecánicas.

Los primeros videojuegos RPG aparecieron en los 80 y han ido evolucionando hasta la actualidad cambiando o añadiendo mecánicas para adaptarse a los nuevos jugadores, durante su evolución también han ido apareciendo diferentes subgéneros como resultado de las nuevas mecánicas o de la fusión con otros géneros.

2.1.1. El inicio del género

En 1974 [2] salió al mercado *Dungeons & Dragons*, el que fue el primer juego de rol de mesa y en el que se empezarían a basar más tarde los RPGs. Al mismo tiempo en las universidades se había estado desarrollando el sistema PLATO, una intranet con foros, chats y entre otros juegos.

Los primeros títulos en aparecer en esta intranet fueron algunos como Dungeon, pedit5 y dnd, todos estos eran *Dungeon Crawlers* (juegos de exploración de mazmorras) donde se avanzaba mediante comandos de texto por la mazmorra en donde se debía derrotar a los diferentes enemigos que iban apareciendo, ganando así equipo o características para el personaje.

Debido al código abierto estos videojuegos pudieron evolucionar de forma que adelantaron lo que más tarde aparecería en la industria comercial, incluso aparecieron títulos multijugador algo que más tarde se perdería con la irrupción de las consolas que no disponían de una conexión entre ellas.

En 1975 apareció *Moria* que fue el primero en ofrecer una vista en primera persona y un sistema de gremios. Más tarde en 1977 *Oubliette* fue más allá implementando un sistema de hechizos y añadiendo clases, razas y una gran cantidad de monstruos y piezas de equipo. Finalmente de todos los RPGs que aparecieron en la intranet el de mayor éxito fue Avatar en 1979, este título implementó el aspecto económico-social con chat, comercio y otras características.

Tras su comienzo en las universidades en 1979 apareció el primer RPG comercial titulado *Temple of Apshai*, salió para ordenadores como el Apple II, el IBM, o Commodore 64 y también para la consola de Atari. El combate en este juego era una mezcla entre combate por turnos y tiempo real, a falta de impacto visual este título ofrecía amplias descripciones de sus zonas.

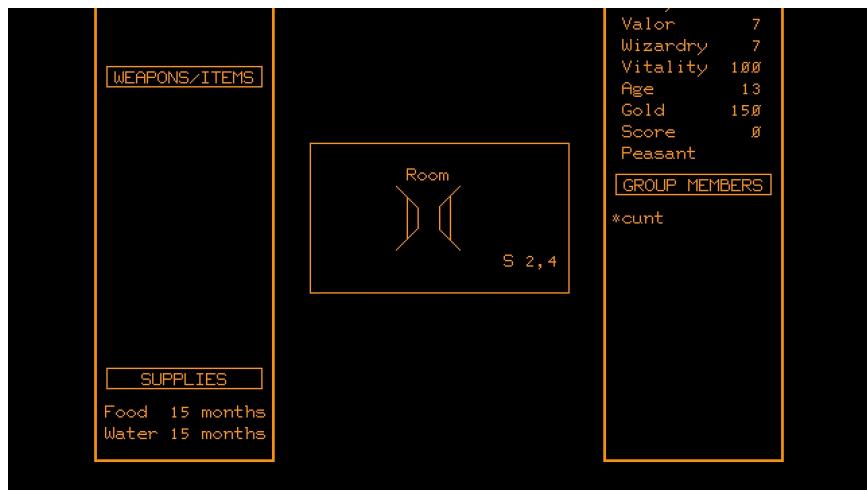


Figura 2.1: Interfaz del RPG Moria

2.1.2. Los primeros RPG comerciales

Entre 1980 y 1982 aparecieron tres títulos que serían de los más influyentes del género, uno de ellos fue Rogue en 1980, este título supondría la aparición más tarde del género *Roguelike*. Este juego podía funcionar en terminales que incluso no tenían gráficos puesto que las versiones originales estaban diseñadas con texto, también fue el primero en implementar un sistema de generación de mazmorras aleatorias de manera que cada partida era diferente de la anterior. Debido a la dificultad del título incluso empezaron a aparecer las primeras trampas en videojuegos mediante un programa externo que controlaba de manera perfecta el personaje.

Otro de los títulos en aparecer fue *Wizardry: Providing Grounds of the Mad Overlord* para el Apple II. Fue el primer RPG por turnos en el que manejábamos un grupo de héroes, esto sería un referente para títulos como *Dragon Quest* o *Final Fantasy*. La segunda parte de este juego sería el primer título en tomarse en serio el concepto de secuela permitiendo importar el grupo del juego anterior.

El último título influyente sería Ultima I para el Apple II, este juego introdujo un área de mapamundi que cambiaba a primera persona al entrar en las mazmorras.

En 1983 aparecieron los SRPG (Videojuego de rol táctico) mediante títulos como *Ultima III: Exodus* con su sistema de casillas o *Bokosuka Wars* donde se manejaba un ejercito. En este año también salió al mercado *Advanced Dungeons & Dragons: Treasure of Tarmin* que ofrecía un apartado visual muy superior al de su competencia.



Figura 2.2: Gráficos del juego Advanced Dungeons & Dragons

2.1.3. Los RPGs en Japón

En 1984 apareció la que sería la primera saga del subgénero de los JRPGs con *Dragon Slayer*, está mas tarde evolucionaría a las series *The Legend of Heroes* e *Ys* las cuales están todavía vigentes a día de hoy. Esta saga junto con *The Tower of Druaga* sentaron las bases del género de los ARPGs que combinaban elementos de acción y aventura de las máquinas recreativas con los elementos de los RPGs.

En 1986 inició la saga de *Dragon Quest* saliendo el primer título para la NES, al mismo tiempo la compañía *Square*, que estaba pasando por problemas financieros, al ver el éxito de *Dragon Quest* empezó a desarrollar un RPG con más profundidad en la historia y en la jugabilidad que sería bautizado como *Final Fantasy* y que vería la luz en 1987. Estas dos sagas terminarían siendo las más representativas del género de los JRPGs cuya una de sus características es tener una historia más lineal.

2.1.4. Los RPGs multijugador

Nacieron en los años noventa [3] con la proliferación de internet aunque sus inicios fueron con *Secret of Mana* y su multijugador cooperativo. Mas tarde saldría a la luz *Diablo* que permitía a los jugadores conectarse entre ellos para enfrentarse a monstruos y comprar objetos en linea.

Hasta el momento el género de los MMORPG (Videojuego de rol multijugador masivo en línea) que ya se había visto en parte reflejado en los juegos del sistema PLATO había dado algunos pasos con *Island of Kesmai* y finalmente aparecería el primer MMORPG gráfico, *Neverwinter Nights*,

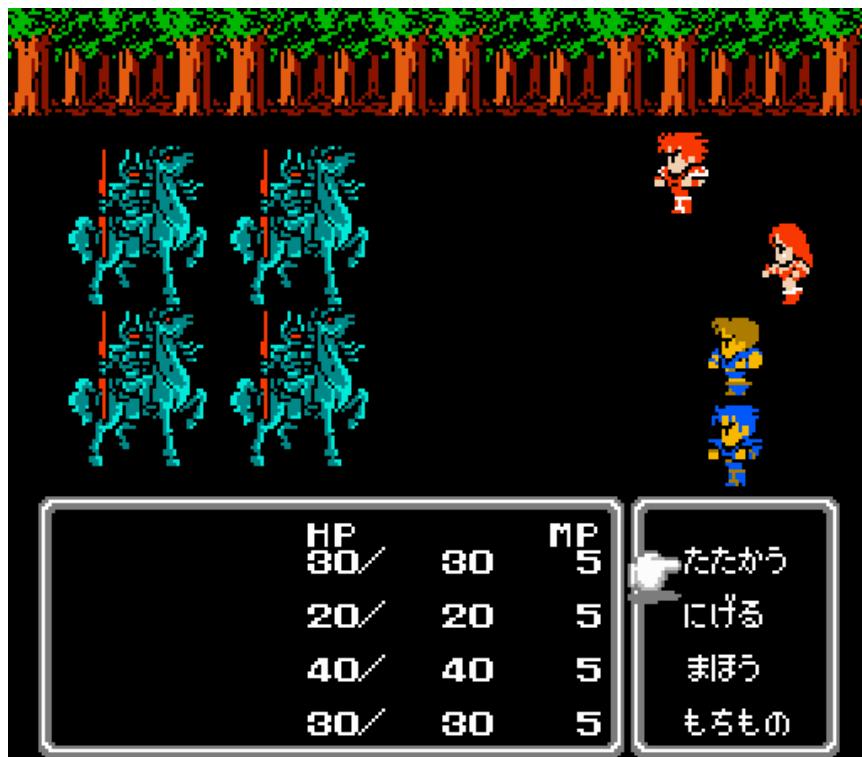


Figura 2.3: Combate en Final Fantasy

en 1991. Más adelante en 1997 vería la luz *Ultima Online* que sería uno de los títulos que más daría a conocer este género.

A partir del año 2000 salieron al mercado una gran cantidad de títulos MMORPG pero la mayoría fallaron en alcanzar el éxito. Finalmente en 2004 vio la luz *World of Warcraft* el cual tuvo un gran éxito, alcanzando niveles que ningún otro MMORPG o incluso videojuegos de otros géneros alcanzó. El éxito de *World of Warcraft* vino dado en parte a un buen diseño, siendo profundo y detallado pero al mismo tiempo fácil de aprender para nuevos jugadores.

2.2. Juegos geolocalizados en el mercado actual

En esta sección se van a describir algunos juegos geolocalizados existentes en el mercado actual, los juegos han sido elegidos por su popularidad y por su similitud con el juego que se va a desarrollar.

2.2.1. Pokémon Go

Es un videojuego geolocalizado y con realidad aumentada que salió al mercado en 2016. El jugador debe buscar los personajes de la saga Pokémon en el mundo real y capturarlos haciendo uso opcionalmente de realidad aumentada.

Existen determinadas localizaciones fijas donde los jugadores pueden reabastecerse de los objetos usados en el juego y otras las cuales contienen gimnasios donde los jugadores podrán enfrentarse a los equipos de otros consiguiendo ese gimnasio para su facción.

Por otro lado también existen incursiones en las que varios jugadores tendrán que colaborar para derrotar a un enemigo más poderoso.

Estos aspectos fomentan por lo tanto que los jugadores recorran las calles realizando actividad física y descubriendo nuevos lugares de su ciudad gracias a las localizaciones fijas. También fomenta el trabajo en equipo y la interacción en el mundo real entre los jugadores puesto que tendrán que estar en el mismo lugar al realizar las incursiones.

2.2.2. Ingress

Este juego geolocalizado [4] fue lanzado en 2013 para Android y más tarde en 2014 para iOS. En este título existen dos facciones enfrentadas que tendrán que aumentar su red mediante la conquista y enlace de portales localizados en puntos de interés.

Los portales ya conquistados pueden ser tomados por jugadores de la facción contraria rompiendo así los enlaces que ya habían sido creados, estos portales también pueden ser defendidos mediante diferentes mecanismos.

Este título promueve la comunicación entre los integrantes de una facción para decidir qué portales atacar y que redes de portales formar, para ello integra un chat entre los jugadores de un área en concreto.

Al igual que en Pokémon Go este juego fomenta que los jugadores recorran la ciudad y realicen actividad física, el aspecto de interacción entre jugadores es más fuerte en este título puesto que tendrán que realizarse estrategias a la hora de montar las redes de portales.

2.2.3. Orna: A Geo-RPG

Se trata de un RPG geolocalizado disponible para Android. En este videojuego el jugador manejará un personaje que irá subiendo de nivel al enfrentarse a los diferentes enemigos que van apareciendo. Los enemigos no

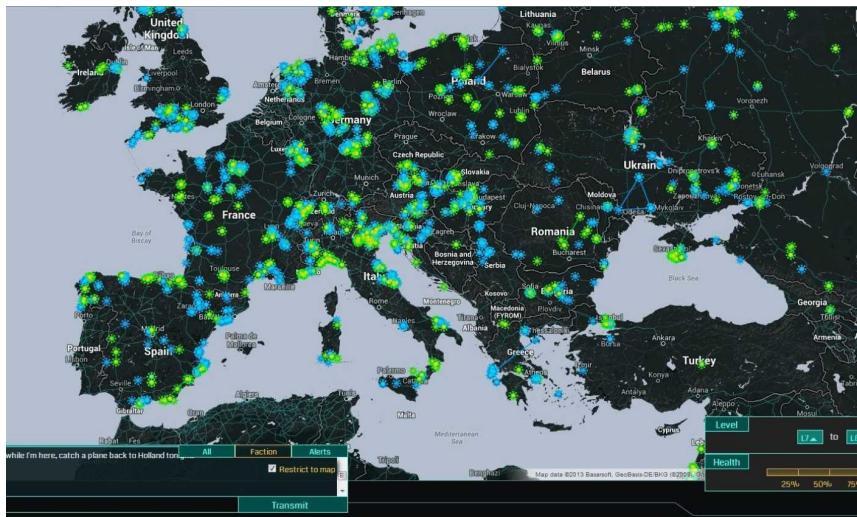


Figura 2.4: Mapa de Ingress

solo nos darán experiencia, también diferentes objetos que servirán para mejorar el personaje al equiparlos.

Los enemigos irán apareciendo en el mapa conforme se avance por el mundo real y para luchar contra ellos tan solo tendrá que pulsarse sobre su ícono. El combate consiste en un sistema por turnos clásico donde el personaje podrá atacar, defenderse o usar alguna habilidad mediante el uso de mana.

El juego permite elegir entre tres clases y cuatro facciones que permitirán poseer tener una especialización. También existen diferentes reinos formados por hasta cincuenta personas de la misma facción mediante los cuales es posible pelear contra miembros de otros reinos o contra jefes.

2.3. Geolocalización

La geolocalización es la capacidad para obtener la ubicación geográfica de un objeto, pudiendo ser este un radar, un dispositivo móvil, un ordenador u otros dispositivos. Actualmente existen 3 tipos [5] principales de geolocalización:

- **GPS:** Para poder hacer uso de esta tecnología es necesario que el dispositivo disponga de un receptor de GPS. Consiste en una red compuesta por varios satélites que transmiten señales que son recibidas e identificadas por los receptores de GPS, estos proporcionan su latitud, longitud, altura y hora local mediante los cuales se calculan sus coordenadas.

- **GSM:** Consiste en hacer uso del sistema global de comunicaciones móviles para calcular la ubicación, teniendo en cuenta la aproximación a las torres de telefonía, el tiempo que tarda en llegar la señal y la fuerza de la misma.
- **WIFI:** Conociendo la dirección MAC del router al que un dispositivo está conectado se puede conocer la ubicación del mismo. Este método de geolocalización es ideal para interiores o lugares a donde la señal GPS no puede llegar.

2.4. Conclusión

Como se ha podido ver los RPG han tenido una larga evolución y siempre han conseguido un hueco en el mercado. Los juegos geolocalizados por su parte han tenido un gran auge en los últimos años motivado en parte por el lanzamiento de Pokémon Go y aunque ningún otro ha conseguido aún lo mismo merece la pena intentar hacerse un hueco en este nuevo género pudiendo ser una buena forma de hacerlo el fusionarlo con otros géneros ya existentes como el de los RPG.

El videojuego que se desarrollara constara de muchas de las características vistas anteriormente, el combate sera similar a los clásicos por turno y al igual que en la mayoría de los RPG el jugador ira subiendo de nivel con los enfrentamientos y consiguiendo nuevas habilidades y objetos. En el aspecto de la geolocalización también tomara algunas características del RPG Orna puesto que es algo similar a lo que se quiere conseguir.

Capítulo 3

Análisis inicial del problema

3.1. Concepto inicial

El juego consiste en un RPG geolocalizado con una temática centrada en la Universidad. Al ser geolocalizado se debe avanzar por la ciudad para poder completar los retos que se plantean y realizar ciertas acciones dependiendo de la localización del jugador. Por otro lado al ser un RPG los principales retos consisten en pelear contra diferentes enemigos para conseguir objetos y subir de nivel.

3.1.1. Narrativa

Según la teoría de la relatividad la gravedad es una manifestación de la curvatura del tejido espacio-tiempo causada por las concentraciones de masa, produciendo ademas una deformación del tiempo que implica que cuanta mas gravedad exista mas lentamente transcurre el tiempo.

El juego supone la existencia de una segunda capa, la Capa de la Consciencia que tiene por función procesar todo lo que ocurre en el Universo quedando grabada en esta todo lo que ha sucedido y conteniendo una representación de lo mismo en todo momento.

La capacidad por unidad espacial de la Capa de la Consciencia es limitada y esto provoca que las concentraciones de sucesos de alto coste computacional puedan superar la capacidad de procesamiento de una zona. El pensamiento, sobretodo el de los humanos, es un proceso de alto coste computacional, por ello los lugares con una alta concentración de personas pensantes pueden sobrecargar la Capa de la Consciencia generando anomalías que pueden llegar a alterar la realidad.

Los lugares con un alto nivel de pensamiento como la Universidad han provocado a lo largo de los años residuos que han desembocado en la alteración de la realidad con la aparición de monstruos hostiles. Debido a este

debilitamiento en la capa los humanos también han empezado a ser capaces de alterar la realidad por si mismos siendo capaces de hacer uso de magia, un conjunto de leyes que han ido autoescribiéndose en la capa como consecuencia del pensamiento humano.

La aparición de esta nueva hostilidad ha provocado que para combatirla se vuelvan a hacer uso de las armas del pasado. Como defensa para estos enemigos las Universidades han evolucionado en instituciones aun mas relevantes encargadas de entrenar a los alumnos como combatientes capaces de derrotar a estos monstruos. Sin embargo cuanta mas enseñanza se efectúa mas se debilita la Capa de la Consciencia y los monstruos aparecen mas frecuentemente y con mas poder. Por otro lado se ha generado una competencia entre las diferentes Escuelas de la Universidad que buscan tener mas relevancia.

Este nuevo mundo ha introducido un dilema: Si las Escuelas dejaran de enseñar la Capa de la Consciencia se iría reparando y los monstruos dejarían de aparecer, pero esto provocaría que la humanidad dejara de avanzar.

3.1.2. Combate

El juego cuenta con un sistema de combate por turnos en el que todos los jugadores seleccionan su acción a la vez y se ejecutan en orden dependiendo de la velocidad de los personajes. Las posibles acciones son las siguientes:

- **Atacar:** Se ataca al enemigo seleccionado con el daño base.
- **Defender:** Se recibe menos daño en los siguientes ataques de ese turno, esta acción siempre es la primera en ejecutarse dentro del turno. Se recupera un poco de vida dependiendo del valor de la fuerza, un poco de mana dependiendo de la inteligencia y existe una probabilidad de realizar contraataques a los enemigos provocando el daño que fueran a realizar y defendiéndose por completo del mismo, esta probabilidad aumenta con la destreza.
- **Habilidad:** Se usa una facultad que gastara mana. Las facultades pueden producir daño, aumentar alguna estadística del personaje o disminuir las de los enemigos.
- **Objeto:** Se utiliza un objeto del inventario.
- **Huir:** Se intenta escapar del combate.

Los combates se dan entre el jugador y hasta un máximo de tres enemigos o entre dos jugadores de facciones diferentes. Todos los participantes poseen una serie de estadísticas comunes que son las siguientes:

- **Puntos de vida:** Vida máxima del personaje, si llega a cero en el jugador este debe desplazarse hasta un gremio para curarse o usar una piedra de recuperación, hasta ese momento no es posible realizar combates ni completar misiones del juego.
- **Puntos de mana:** Mana máximo del personaje, se usa para realizar habilidades.
- **Fuerza:** Aumenta el daño base al atacar con el arma. Al usar la acción de defender se recupera un porcentaje de vida dependiendo del valor de este atributo.
- **Destreza:** Aumenta la velocidad, la cual influye en el orden de los turnos, y las probabilidades de realizar un crítico. Al usar la acción de defender aumenta la probabilidad de realizar un contraataque.
- **Inteligencia:** Aumenta el daño realizado por las habilidades. Al usar la acción de defender recupera un porcentaje del mana dependiendo del valor de este atributo.
- **Probabilidad de crítico:** Como base se posee un 10% y aumenta con la destreza. Si se realiza un crítico se infinge el doble de daño.
- **Daño base:** El valor es dado por el arma y por defecto el personaje tiene un punto. Al atacar es aumentado por la fuerza.

Existe una probabilidad para que un combate inicie mientras el jugador camina, esta probabilidad aumenta mientras mas distancia haya recorrido el jugador sin que se haya producido ningún encuentro. Por otro lado para enfrentarse a los jefes es necesario localizar los mismos en el mapa y el jugador debe decidir si pelear con ellos o no.

Al terminar el combate el jugador es recompensado con experiencia y dinero, también existe la probabilidad de que este consiga objetos.

3.1.3. Facciones

Las facciones depende de la rama de estudios seleccionada, estas se llaman escuelas y son tres:

- **Escuela de las Artes y las Humanidades**
- **Escuela de la Ciencia**
- **Escuela de la Ingeniería**

Las facciones poseen una puntuación llamada reputación que aumenta cada vez que un jugador de la misma derrote a un jefe, en los gremios de su facción los jugadores pueden ver la reputación de todas.

3.1.4. Clases

Pueden seleccionarse tres clases independientemente de la facción elegida, estas son:

- **Guerrero:** Su atributo principal es la fuerza y se basa en hacer daño mediante ataques básicos. Sus habilidades consisten en mejorar alguna de sus estadísticas.
- **Espadachín:** Su atributo principal es la destreza y se basa en hacer daño mediante ataques básicos intentando realizar críticos. Sus habilidades consisten en empeorar alguna de las estadísticas de los enemigos.
- **Mago:** Su atributo principal es la inteligencia y se basa en hacer daño mediante habilidades. Sus habilidades consisten en realizar daño.

Al subir de nivel cada clase aumenta unos atributos determinados y en ciertos niveles aprende una habilidad.

3.1.5. Objetos

Pueden ser usables, en cuyo caso es posible utilizarlos en cualquier momento desde el inventario o durante un combate excepto sin ser piedras de recuperación, o equipables, los cuales aumentan una estadística al agregarlos al equipo del jugador. Los tipos son los siguientes:

- **Poción de vida:** Es un objeto usable, recupera una cantidad de puntos de vida.
- **Poción de mana:** Es un objeto usable, recupera una cantidad de puntos de mana.
- **Piedra de recuperación:** Es un objeto usable, permite que un jugador que se ha quedado sin puntos de vida se recupere.
- **Arma:** Es un objeto equipable, se asigna en su espacio correspondiente pudiendo aumentar estadísticas y proporcionando un daño base.
- **Accesorio:** Es un objeto equipable, se asigna en su espacio correspondiente y aumenta las estadísticas.

3.1.6. Misiones

Las misiones pueden ser aceptadas en los gremios y consisten en eliminar un número de enemigos los cuales pueden ser normales o jefes. Una vez finalizadas el jugador debe desplazarse hasta un gremio y como recompensa se obtiene dinero, experiencia y opcionalmente objetos.

3.1.7. Elementos del mapa

Dentro del mapa existen elementos estáticos y dinámicos, los estáticos siempre se encuentran en el mismo lugar mientras que los dinámicos cambian su localización al terminar el día.

Los elementos estáticos son las tiendas donde los jugadores pueden comprar objetos y los gremios de las escuelas que están localizados en las facultades correspondientes. En los gremios pueden aceptarse misiones y entregar las ya completadas, los jugadores también pueden curarse y recobrar su mana e incluso descansar para recuperar sus puntos de salud tras haber sido derrotados.

Los elementos dinámicos son los jefes que los jugadores tienen que derrotar, solo se puede combatir contra los mismos tras adquirir puntos de valor que son obtenidos al recorrer cierta distancia. También podemos encontrar a otros jugadores en el mapa con los que es posible iniciar un chat en caso de que sean de la misma facción o iniciar un combate si son de otra facción.

Capítulo 4

Tecnología a usar

En este capítulo se van a presentar las distintas tecnologías usadas en el desarrollo del proyecto y una justificación de su elección frente a otras alternativas.

4.1. Elección del motor gráfico

En esta sección se va a realizar una comparación entre varios motores gráficos para poder elegir el que mejor se adapte a las necesidades del proyecto.

4.1.1. Unity

Unity es un motor gráfico [6] desarrollado por Unity Technologies, es uno de los mas usados por los desarrolladores, utiliza herramientas de diseño 2D y 3D, cuenta con un modo *Play* que permite ejecutar el juego y editar sus parámetros rápidamente en tiempo de ejecución y posee un potente sistema de animaciones.

Unity es uno de los motores mas populares a la hora de desarrollar juegos para móviles y para el mercado VR. Algunas de las plataformas para las que se puede desarrollar en Unity son Windows, Mac, iOS, Android, Playstation, Xbox, Windows Phone, entre otras.

Algunos juegos desarrollados con este motor son Pokemon GO, Super Mario Run, Angry Birds 2, Wasteland 2 y otros muchos.

Por otro lado Unity dispone de tres tipos [7] de licencia:

- La licencia Pro por 125 \$ al mes que esta pensada para profesionales y estudios, incluye acceso prioritario a especialistas de Unity, *Success Advisor* y servicio al cliente.



Figura 4.1: Logo de Unity

- La licencia Plus por 25 \$ al mes si se paga todo el año y 35 \$ al mes en caso contrario, esta pensada para aficionados que quieran acelerar su aprendizaje y desarrollo.
- La licencia Personal, disponible gratis si los ingresos o fondos no superan los 100000 \$ por año.

Por último otras características [8] de Unity son que los lenguajes usados pueden ser C#, JavaScript o Boo, posee una tienda de Assets con muchos de ellos gratuitos, tiene un editor UI y otras muchas funcionalidades.

4.1.2. Unreal Engine

Unreal Engine es un motor desarrollado por Epic Games, seguramente sea la mejor opción si lo que se busca es un motor para el desarrollo de un juego sofisticado que requiera un potente motor gráfico en 3D, aunque también tiene soporte para 2D. El tamaño de los juegos generados serán mayores en este motor y los dispositivos necesarios para ejecutarlos tendrán que ser de gama alta.

Algunas de las plataformas para las que se puede desarrollar con Unreal Engine son Windows, Mac, Linux, iOS, Android, Playstation, Xbox, entre otras. Existe un gran número de juegos desarrollados con este motor, estos suelen requerir dispositivos mas exigentes para ejecutarlos, algunos de estos juegos son *Marvel Heroes*, *Batman: Arkham Origins*, *Infinity Blade 3* y otros muchos. Actualmente Unreal Engine es gratuito aunque se tendrán que pagar un 5 % de los ingresos generados con el juego a Epic si estos son superiores a 3000 \$.

Otras de las características presentes en este motor son que los proyectos pueden ser desarrollados con Blueprint, C++ o una combinación de ambos,



Figura 4.2: Logo de Unreal Engine

posee un modo *Instant game preview* para poder observar rápidamente los resultados de lo desarrollado, tiene un diseñador de UI, posee un *Marketplace*, tiene integración multijugador y otras muchas funcionalidades.

4.1.3. Godot Engine

Godot es un motor gráfico desarrollado por la comunidad, tiene soporte para 2D y 3D. Posee un amplio catálogo de herramientas como un editor, un sistema de escenas, un creador de interfaces y un diseñador de niveles y animaciones.



Figura 4.3: Logo de Godot Engine

El motor gráfico es totalmente gratuito y *open source*. Algunas de las plataformas para las que se puede desarrollar son Windows, Mac, Linux, iOS, Android, BlackBerry, HTML5, PlayStation, Nintendo, entre otras.

4.1.4. CryEngine

CryEngine es un motor gráfico desarrollado por Crytek, ofrece el código completo de forma gratuita. Es un potente motor aplaudido por sus gráficos generados mediante su colección de herramientas. CryEngine también ofrece FMod, una de las mejores herramientas de audio. Como desventaja a diferencia de los motores anteriores este no soporta desarrollo en 2D.

Las plataformas para las que se puede desarrollar con CryEngine son iOS, Android, Windows, Linux, Playstation, Xbox y Wii. Algunos de los juegos desarrollados con este motor son Far Cry, Crysis, Sniper: Ghost Warrior 2, entre otros.



Figura 4.4: Logo de Cry Engine

4.1.5. CoronaSDK

Es un motor gráfico destinado al desarrollo de juegos 2D. Esta pensado para poder empezar los desarrollos en el menor tiempo posible. *CoronaSDK* trabaja con una API en Lua que permite a los desarrolladores acceder rápidamente a sus características nativas. Este motor gráfico también aporta facilidades en la monetización de los juegos desarrollados ya que proporciona soporte nativo para compras dentro de la aplicación y publicidad.



Figura 4.5: Logo de CoronaSDK

Corona también permite *testear* los juegos mediante su emulador el cual simula el dispositivo al completo, incluyendo herramientas y depuración.

Las plataformas para las que se puede desarrollar son Mac, Windows, Android, iOS, Kindle, entre otras. Corona es gratuito pero dispone de una

versión de pago para empresas y pequeños negocios por 75 \$ al mes, esta integra *CoronaSDK Plus* y librerías nativas.

4.1.6. Conclusión

En primer lugar CryEngine parece que no seria una buena elección para el proyecto puesto que no dispone de herramientas para el desarrollo en 2D, además no sera necesario el nivel gráfico ofrecido por este motor. Con Unreal Engine sucede algo similar puesto que ofrece demasiada potencia gráfica para lo que exigirá el proyecto, además parece estar mas enfocado a grandes proyectos de empresas.

Por otro lado dos buenas opciones son *CoronaSDK* y *Godot* puesto que se pueden obtener de forma totalmente gratuita y poseen herramientas para el desarrollo 2D, sin embargo esto es algo que también ofrece Unity y este último parece tener una comunidad muy superior a los otros dos, además de tener una *Marketplace* mas completa.

Por último una búsqueda rápida de bibliotecas o recursos para el desarrollo con mapas en línea a resultado en varios recursos para Unity como por ejemplo Vuforia o Mapbox, sin embargo parece no existir ninguno para CoronaSDK o Godot. Por lo tanto la mejor opción parece ser Unity puesto que ya existen las herramientas necesarias para el desarrollo del proyecto y gracias a su gran comunidad sera mas fácil resolver los problemas que aparezcan durante el desarrollo.

4.2. Slim

El *framework* utilizado para el desarrollo del servidor es Slim el cual usa PHP. Las principales ventajas [9] de este *framework* son las siguientes:

- Es fácil de aprender.
- Es usada para desarrollar APIs RESTful y servicios web.
- Incluye enrutamiento, sesiones, encriptación de *cookies* y almacenamiento en caché HTTP del lado del cliente.
- Es perfecta para pequeñas aplicaciones web.

Debido a que se desarrollara una API RESTful el framework mencionado es ideal puesto que utiliza PHP, uno de los lenguajes mas usados en la carrera y gracias al fácil aprendizaje se podrá invertir mas tiempo en aprender Unity.



Figura 4.6: Logo de Slim

4.3. Mapbox

Mapbox es el kit de desarrollo utilizado para construir el mapa del juego. Además de para Unity Mapbox esta disponible para Android, iOS y Javascript.



Figura 4.7: Logo de Mapbox

Todo el código de Mapbox [10] es abierto y posee repositorios en Github desde donde se puede seguir su desarrollo e incluso participar en el mismo.

Mapbox cuenta con un editor de mapas en linea llamado *Mapbox Studio* desde el cual se puede personalizar el diseño del mapa que se mostrara en la aplicación.

Para poder utilizar Mapbox en la aplicación tan solo se debe crear una cuenta y generar un *token* de acceso, esto sera gratis hasta los primeros 50000 usuarios activos al mes, a partir de ese número se pagara cada cierta cantidad de usuarios. En el caso de Unity al cargar la librería nos pedirá el *token* de acceso.

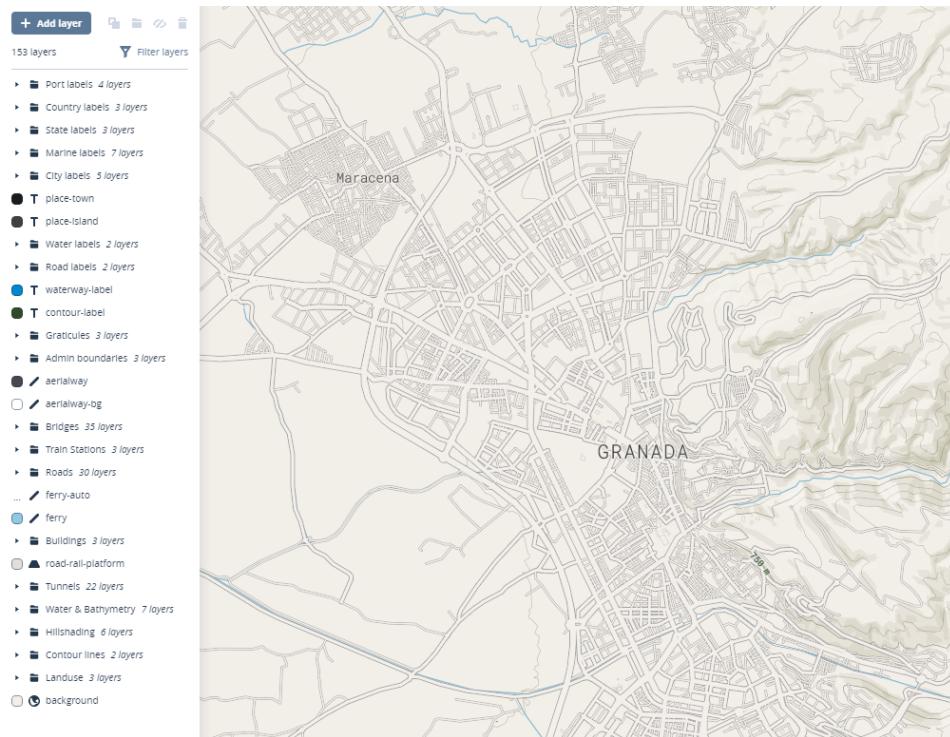


Figura 4.8: Mapbox Studio

4.4. Arquitectura inicial

En esta sección se muestra la arquitectura inicial de la aplicación y una breve descripción de la misma.

Como se puede observar en la figura 4.9, la aplicación esta compuesta por varios gestores y catálogos. Los catálogos contienen las listas de los objetos creados como enemigos o misiones, estos serán utilizados por los gestores para adquirir los objetos específicos que necesiten en cada momento.

El gestor de juego sera el encargado de guardar todos los gestores, alternar entre ellos y guardar la información del mundo proporcionandola a los gestores que la soliciten.

El gestor de conexión sera el encargado de generar las peticiones y enviarlas a la *API RESTful*, todos los gestores podran utilizarlo en cualquier momento.

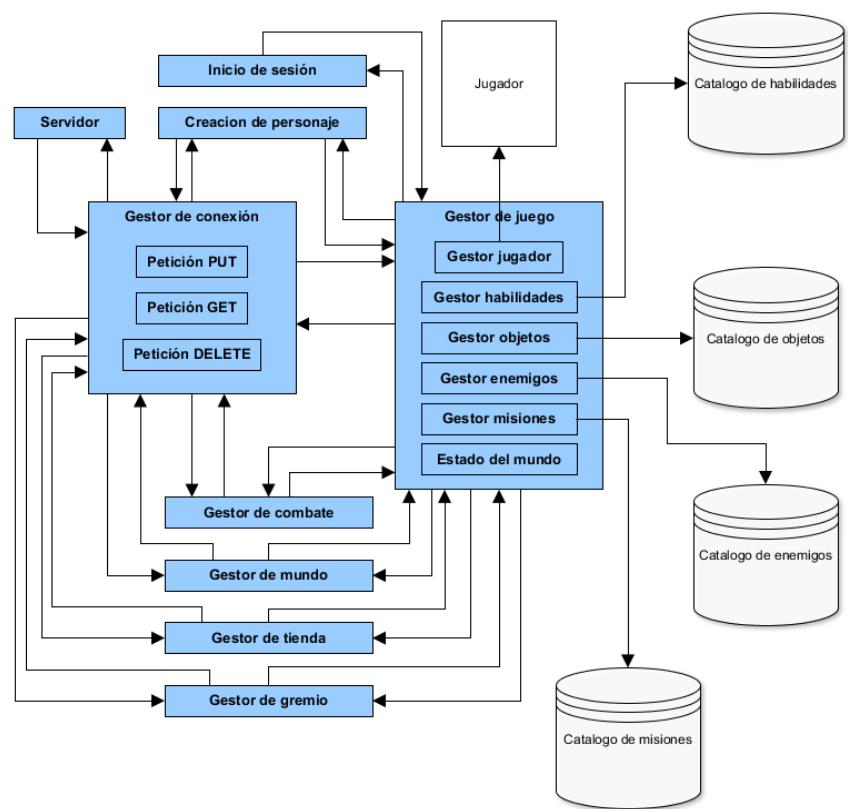


Figura 4.9: Arquitectura inicial

Capítulo 5

Metodologías a usar en el proyecto

En este capítulo se van a presentar las metodologías de desarrollo usadas en el proyecto que incluye diseño centrado en el usuario, metodologías ágiles y prototipado.

5.1. Diseño centrado en el usuario

El diseño centrado en el usuario (DCU) es una metodología que se centra [11] en el diseño de productos que respondan a las necesidades reales de los usuarios finales.

5.1.1. Principios del diseño centrado en el usuario

Existen varios estándares para adoptar esta metodología de diseño, entre ellos destaca el *ISO 9241-210:2010 – Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems*, en este se describen seis principios [12] que caracterizan un diseño centrado en el usuario los cuales son:

- El diseño esta basado en una compresión explícita de objetivos tareas y entornos.
- Los usuarios están involucrados durante el diseño y el desarrollo.
- El diseño está dirigido y refinado por evaluaciones centradas en los usuarios.
- El proceso es iterativo.
- El diseño está dirigido a toda la experiencia de usuario.



Figura 5.1: Estándar ISO 9241

- El equipo de diseño incluye habilidades y perspectivas multidisciplinares.

Esta formado por un conjunto de cuatro actividades que se van repitiendo de forma cíclica, estas se pueden ver en la figura 5.1.

Sin embargo como ya se ha mencionado existen varios estándares y aunque el anterior descrito es uno de los mas destacados se pueden destacar unos aspectos comunes a la mayoría de ellos que son los siguientes:

- El diseño esta orientado a los usuarios del producto y estos participan durante todo el proceso.
- Se aplica durante todas las fases de desarrollo, incluyendo por lo tanto planificación, diseño, desarrollo y evaluación.
- Es iterativo.
- Es multidisciplinar.
- El objetivo es obtener productos usables y satisfactorios para los usuarios.

5.2. Modelo de prototipos

Es un modelo de desarrollo [13] en el que el sistema se va construyendo mediante el refinamiento consecutivo de prototipos hasta que se cumple con todos los requisitos dados.

El modelo esta constituido por varias etapas que se van repitiendo de forma cíclica mientras se va refinando el prototipo, en primer lugar se realiza un plan rápido con el cliente sobre las funcionalidades ha desarrollar, de esto se realiza un modelado y diseño rápido que se centrara sobre todo en los aspectos software visibles para el usuario, a continuación se construye el prototipo que sera elaborado de forma rápida y económica, este constituirá un sistema funcional con los requisitos obtenidos del cliente, por último se entregaría el prototipo al cliente y se obtendría una retroalimentación del mismo.

Puesto que el cliente participa activamente proporcionando retroalimentación de los prototipos obtenidos, el producto generado sera mas seguro a la hora de satisfacer las necesidades del cliente, esto además ayudara a seguir un diseño centrado en el usuario puesto que sera mas fácil que cumpla las necesidades de los usuarios al haber estado el cliente involucrado en el desarrollo de los prototipos.

Este modelo se seguirá durante el desarrollo del proyecto puesto que las entregas obtenidas serán prototipos con las diferentes funcionalidades indicadas. Los clientes en este caso sera los tutores del proyecto puesto que irán comprobando los prototipos y aportando retroalimentación sobre los mismos.

5.3. Metodologías ágiles

Se utilizara un desarrollo iterativo e incremental [14] en la construcción del videojuego. En este tipo de desarrollo el proyecto se planifica en iteraciones que duran cierto tiempo. En cada una de las iteraciones se obtiene una parte del producto final, completándose cada requisito en una única iteración.

Por lo tanto en cada iteración el producto evoluciona a partir de los resultados obtenidos en iteraciones anteriores. En el caso de este proyecto la mayoría de iteraciones estarán constituidas por un prototipo de acuerdo al modelo de prototipos visto en la sección anterior.

5.4. Aplicación de las metodologías en el proyecto

Capítulo 6

Plan de entregas

6.1. Breve descripción del alcance del sistema

El proyecto consiste en la implementación de un juego geolocalizado, algunos de los objetivos mas importantes son:

- Permitir la creación de cuentas y personajes.
- Realizar la actualización de los personajes de acuerdo a los eventos que ocurren.
- Permitir interactuar sobre diferentes elementos del mapa dependiendo de la posición del jugador en el mundo.
- Permitir al jugador ver las estadísticas, misiones y habilidades de su personaje.
- Permitir al jugador aceptar y entregar misiones en los gremios.
- Permitir al jugador equipar y usar diferentes objetos.
- Permitir al jugador comprar objetos en la tienda.
- Realizar combates basados en turnos y obtener una recompensa al final de los mismos.
- Obtener puntos al derrotar a los jefes para las facciones correspondientes y poder hacer un seguimiento de los mismos desde los gremios.
- Se seguirá una arquitectura como la mostrada en el capítulo 4.

6.2. Descripción de las entregas

En la siguiente tabla se muestran las fechas de cada entrega con sus objetivos:

Entrega	Objetivo
1	Bocetos iniciales del concepto de juego y caracterización del jugador
2	Prototipo del mapa
3	Prototipo de las escenas de tienda, menú y gremio
4	Prototipo con conexión al servidor
5	Prototipo del combate y chat entre jugadores
6	Prototipo con misiones y combate entre jugadores
7	Evaluación de la aplicación

Cuadro 6.1: Plan de entregas.

6.3. Historias de usuario

Esta sección presenta algunas historias de usuario que ayudan a mostrar ciertos requisitos de la aplicación.

HU.1 Definir un personaje

Descripción: Un usuario puede crear un personaje escribiendo un nombre, seleccionando una clase y por último una facción.

Pruebas de aceptación:

- Intentar crear un personaje con algún campo sin llenar, el sistema impedirá continuar.
- Intentar crear un personaje con todos los datos correctos, el sistema entrara en el mapa del mundo con el personaje creado.

Observaciones: Para poder crear un personaje se tendrá que crear una cuenta. Al crear un personaje se enlazara a la cuenta, solo podrá tenerse un personaje por cuenta.

Cuadro 6.2: Historia de usuario: Definir un personaje

HU.2 Realizar una misión

Descripción: Un usuario puede realizar una misión derrotando a los enemigos indicados en la misma y entregándola en el gremio una vez completado el encargo.

Pruebas de aceptación:

- Intentar entregar la misión sin completarla, la misión no aparecerá en el gremio para entregar.
- Intentar entregar la misión una vez completada, la misión. El sistema lo permitirá y se aportara el dinero y la experiencias mostrados al jugador.

Observaciones: Para poder hacer la misión previamente se tendrá que aceptar en el gremio.

Cuadro 6.3: Historia de usuario: Realizar una misión

Capítulo 7

Desarrollo. Entregas e iteraciones

7.1. Entrega 1

En esta sección se muestran los bocetos iniciales del concepto de juego y el desarrollo de la caracterización del personaje, la cual incluye la creación de la interfaz gráfica y los datos que el personaje necesitará.

7.1.1. Bocetos iniciales

Los bocetos iniciales muestran la navegación del usuario desde que crea la cuenta hasta que entra en combate. En este proceso el usuario pasa por la creación de cuenta en la cual elige un email y una contraseña, la creación de personaje en la cual selecciona un nombre, facción y clase, el mapa del mundo donde existen diferentes elementos mediante los cuales se accede a otras secciones del juego y por último el combate al que se accede al caminar por el mapa del mundo o al pulsar sobre los jefes dentro del mismo.

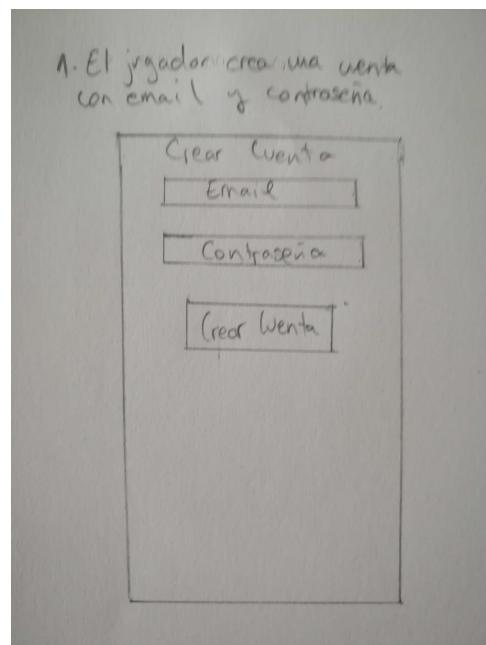


Figura 7.1: Boceto de la creación de la cuenta

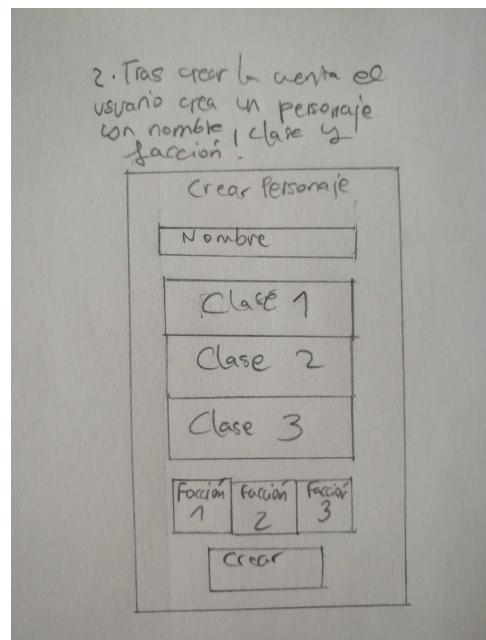


Figura 7.2: Boceto de la creación del personaje

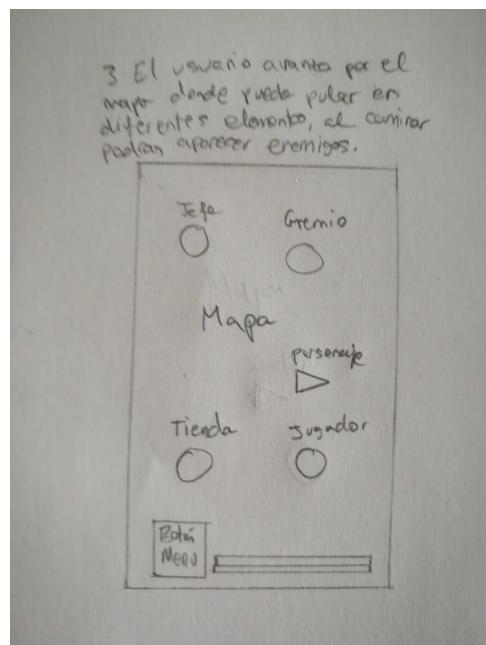


Figura 7.3: Boceto del mapa de juego

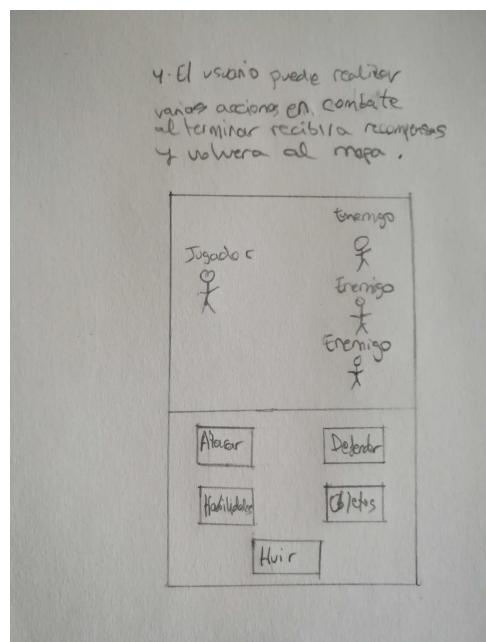


Figura 7.4: Boceto del combate

7.1.2. Caracterización del jugador

Al crear un personaje se debe elegir un nombre y seleccionar su clase y escuela, los atributos iniciales del personaje se pondrán dependiendo de la clase seleccionada y las habilidades se van aprendiendo al subir de nivel dependiendo también de la clase elegida.

Los datos del personaje son los siguientes:

- **Puntos de vida máximos:** Vida máxima del personaje.
- **Puntos de mana máximos:** Mana máximo del personaje.
- **Puntos de vida actuales:** Vida actual del jugador, no puede ser mayor que los puntos de vida máximos, si llega a cero en el jugador este debe desplazarse hasta un gremio para curarse o usar una piedra de recuperación, hasta ese momento no es posible realizar combates ni completar misiones del juego.
- **Puntos de mana actual:** Mana actual del personaje, no puede ser mayor que los puntos de mana máximos, se gasta al utilizar habilidades.
- **Fuerza:** Aumenta el daño base al atacar con el arma. Al usar la acción de defender se recupera un porcentaje de vida dependiendo del valor de este atributo. Es el atributo predominante en el guerrero.
- **Destreza:** Aumenta la velocidad, la cual influye en el orden de los turnos, y las probabilidades de realizar un crítico. Al usar la acción de defender aumenta la probabilidad de realizar un contraataque. Es el atributo predominante en el espadachín.
- **Inteligencia:** Aumenta el daño realizado por las habilidades. Al usar la acción de defender recupera un porcentaje del mana dependiendo del valor de este atributo. Es el atributo predominante en el mago.
- **Daño base:** El valor es dado por el arma y por defecto el personaje tiene un punto.
- **Experiencia:** Puntos de experiencia necesarios para llegar al siguiente nivel.
- **Experiencia actual:** Puntos de experiencia actuales, si igualan a la experiencia o la sobrepasan se avanza al siguiente nivel.
- **Inventario:** Objetos que posee el jugador.
- **Lista de habilidades:** Habilidades que ha aprendido el jugador.

- **Dinero:** Puede ser usado en la tiendas.
- **Arma:** Arma equipada, aumenta el daño base.
- **Accesorio:** Accesorio equipado, aumenta el alguna estadística.
- **Nivel:** Nivel actual del jugador, los enemigos aumentan sus estadísticas dependiendo del mismo.
- **Retrato:** Imagen del personaje, cambia dependiendo de su clase.
- **Puntos de valor:** Se ganan al caminar y se gastan al combatir contra jefes.
- **Lista de misiones:** Misiones aceptadas por el jugador, se recogen y completan en los gremios.



Figura 7.5: Creación del personaje

7.2. Entrega 2

Esta entrega tiene como objetivo conseguir un prototipo del mapa de juego, contara con elementos que se mostraran dependiendo de la posición del jugador y una interfaz que permitirá ver el estado del personaje, entrar al menú y poder filtrar los elementos del mapa que se muestran.

7.2.1. Mapbox

La librería utilizada para mostrar el mapa es Mapbox, esta ya aporta un ejemplo en Unity de su uso y el mapa obtenido se ha basado en el mismo. En la figura 7.6 se muestran los tres principales *Gameobjects* utilizados en el ejemplo.

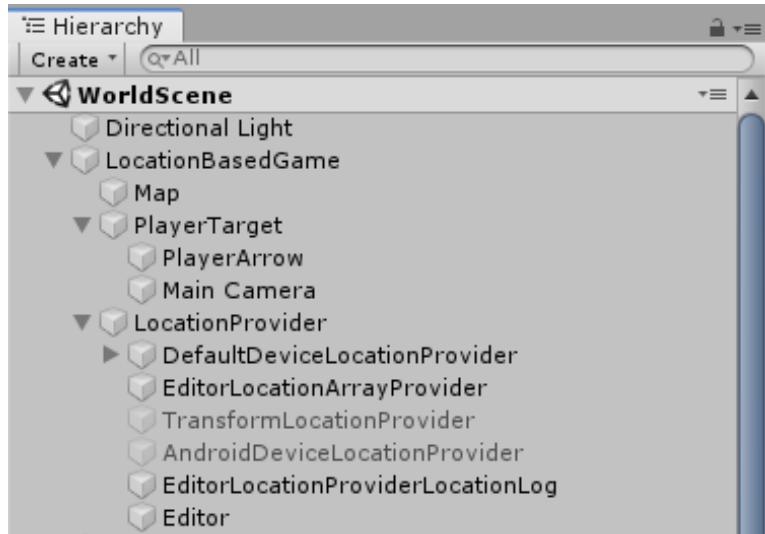


Figura 7.6: GameObjects de Mapbox

El *GameObject LocationProvider* mantiene una colección de *scripts* que sirven para determinar la localización del jugador de forma real o simulada desde el editor de Unity. Los *scripts* son los siguientes:

- **DeviceLocationProvider:** Usa los datos del GPS. El *GameObject PlayerTarget* se reubica en el mapa cada vez que las coordenadas del GPS se actualizan.
- **EditorLocationProvider:** Es similar a *DeviceLocationProvider* pero simulado desde el editor de Unity.
- **LocationArrayEditorLocationProvider:** Permite al jugador moverse entre diferentes coordenadas dadas en un vector.

- **TransformLocationProvider:** Permite al *Gameobject PlayerTarget* seguir la posición y rotación de otro *Gameobject*.
- **DeviceLocationProviderAndroidNative:** Usa datos GPS optimizados para dispositivos Android.

El *Gameobject PlayerTarget* representa la localización del jugador en el mapa, el jugador se mueve y rota junto con el *LocationProvider*, para ello hace uso de dos *scripts*, *ImmediatePositionWithLocationProvider* que mueve al jugador dependiendo de las coordenadas obtenidas por el *LocationProvider* y *RotateWithLocationProvider* que rota al jugador dependiendo de los datos obtenidos por el *LocationProvider*.

Por último el *Gameobject Map* es el encargado de generar el mapa con las opciones indicadas como se muestra en la figura 7.7.

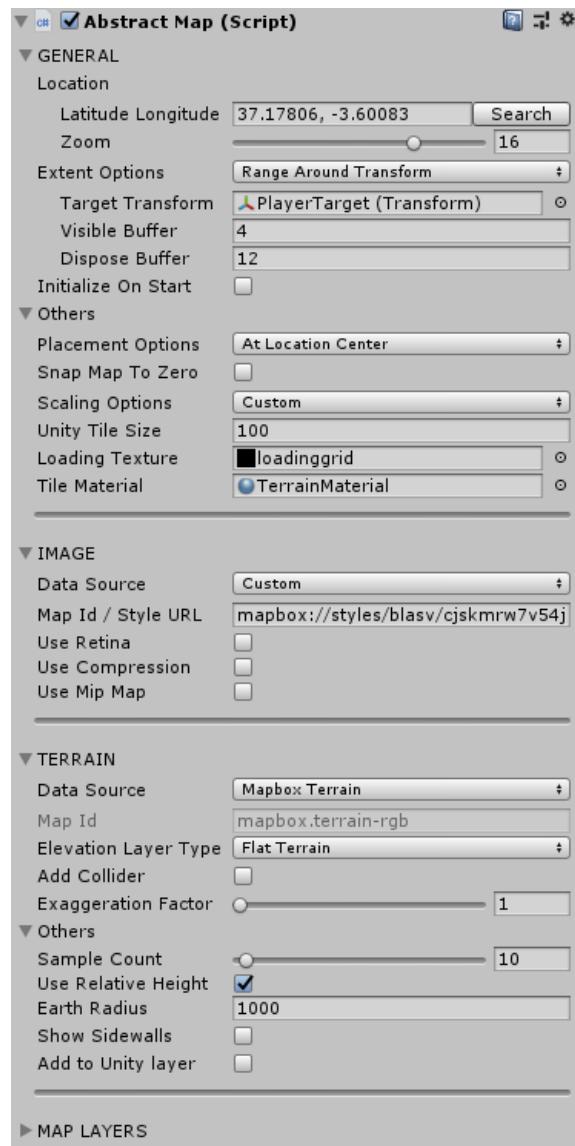


Figura 7.7: Opciones del mapa

7.2.2. Mapa del mundo

El mapa es una representación del mundo real por lo tanto cuando el jugador se mueva en la vida real también lo hará en el juego. Existen diferentes elementos que están posicionados en localizaciones del mundo real y son obtenidos del servidor, estos son tiendas, jefes, gremios y otros jugadores.

El jugador envía periódicamente su posición al servidor y este responde con los elementos que estén cerca de sus coordenadas incluyendo otros juga-

dores que serán rojos en caso de que sean de una facción contraria y verdes si son de la misma.

El jugador puede interactuar con los diferentes elementos del mundo, excepto sobre el ícono que lo representa al él, pulsando sobre los iconos correspondientes que se muestran en el cuadro 7.1.

Elemento	Icono	Descripción
Jugador		Muestra la posición del jugador.
Tienda		Entra en la interfaz de la tienda y se muestran los objetos específicos de la misma.
Gremio		Entra en la interfaz de gremio y se muestra la lista de misiones disponibles, también pueden entregarse las ya completadas.
Jugador Enemigo/Aliado		En caso de que sea un jugador enemigo entra en la escena de combate contra el jugador, si es un aliado entra en la interfaz de comercio donde se pueden intercambiar objetos.
Jefe		Entra en la escena de combate contra el jefe seleccionado.

Cuadro 7.1: Elementos del mundo.

En la figura 7.8 puede observarse el mapa del mundo con el jugador, dos tiendas cercanas y dos jugadores, un aliado y un enemigo.

Como se muestran en las figuras 7.8 y 7.9 desde el mapa de mundo también se pueden visualizar las barras de vida y de mana del jugador, el retrato del jugador, desde el cual se accede al menú y un ícono de filtro mediante el cual se puede abrir la ventana para filtrar los elementos que se muestran tal y como se presenta en la figura 7.9.

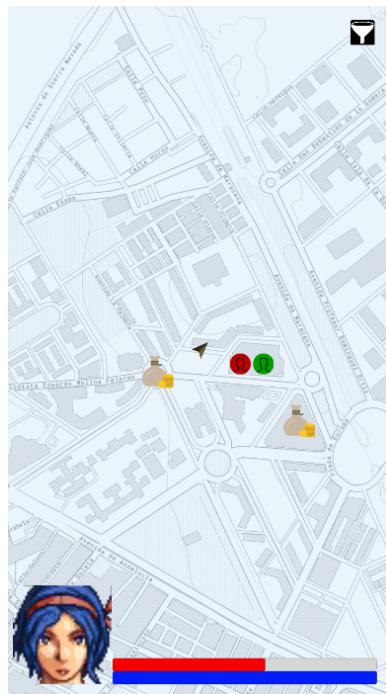


Figura 7.8: Mapa del mundo



Figura 7.9: Ventana de filtros

7.2.3. Estructura

En esta sección se van a presentar los principales *Gameobjects* usados en la escena *WorldScene* en la cual se encuentra el mapa del mundo.

Existen varios gestores que se encargan de mostrar los diferentes elementos del mapa, estos son:

- **ShopSpawnerManager:** Genera los *prefabs* de las tiendas con los datos obtenidos del servidor y los muestra en el mapa posicionandolos en las coordenadas dadas.
- **PlayerSpawnerManager:** Genera los *prefabs* de los jugadores aliados y enemigos con los datos obtenidos del servidor y los muestra en el mapa posicionandolos en las coordenadas dadas.
- **EnemyEncounterManager:** Mide los pasos dados por el jugador y genera encuentros aleatorios contra enemigos aumentando la probabilidad de los mismos al aumentar el número de pasos sin encuentros.
- **BossSpawnerManager:** Genera los *prefabs* de los jefes con los datos obtenidos del servidor y los muestra en el mapa posicionandolos en las coordenadas dadas.
- **GuildSpawnerManager:** Genera los *prefabs* de los gremios con los datos obtenidos del servidor y los muestra en el mapa posicionandolos en las coordenadas dadas.
- **FilterManager:** Guarda información de que objetos se mostraran y gestiona la apertura y el cierre de la ventana de filtros.

Las coordenadas del jugador son enviadas al servidor cada cinco segundos, esto es realizado desde el *Gameobject PlayerTarget* que también es el encargado de posicionar el icono del jugador en el mapa. Como respuesta al envío de las coordenadas el servidor responde con los datos de los elementos cercanos al jugador. El servidor solo responderá con los datos de los jugadores cercanos si estos han actualizado sus coordenadas hace al menos veinte segundos asegurando así que los jugadores mostrados están conectados.

7.2.4. Pruebas

Se han realizado varias pruebas colocando múltiples elementos en el mapa, los jugadores han sido emulados con una petición mediante el programa *Postman*. También se ha comprobado que los elementos aparecen solo a cierta distancia del jugador que en las pruebas ha sido de quinientos metros.

7.3. Entrega 3

Esta entrega tiene como objetivo conseguir un prototipo de las escenas de tienda, gremio y menú. Desde la tienda pueden comprarse objetos con el dinero recibido como recompensa de misiones y combates, desde el gremio pueden aceptarse nuevas misiones y entregar las ya completadas y desde el menú pueden verse las habilidades del personaje, las misiones activas, las estadísticas del personaje y los objetos del inventario que pueden usarse, equiparse o eliminarse.

7.3.1. Tienda

En la escena de la tienda se pueden adquirir objetos mediante el dinero conseguido al derrotar enemigos o entregar misiones.

Como se muestra en las figuras 7.10 y 7.11 la interfaz esta compuesta por un panel con desplazamiento para poder ver los objetos disponibles, cada objeto tiene dos botones uno para aumentar la cantidad a comprar y otro para disminuirla, en la parte superior existe un panel donde se muestra una descripción del objeto seleccionado. En la zona inferior existe un panel que muestra el dinero disponible del jugador y el dinero que se gastara en la compra actual.



Figura 7.10: Escena de la tienda



Figura 7.11: Compra en la tienda

Los objetos del juego son *ScriptableObject*s, esto permite que los objetos se puedan guardar fuera de tiempo de ejecución y cargarse mas tarde al ejecutar el juego, este tipo de objeto permite ademas crear editores en la interfaz de Unity para gestionarlos con mas facilidad tal y como se muestra en la figura 7.12.

Las variables incluidas en los objetos son las siguientes:

- **ItemName:** El nombre del objeto.
- **ItemType:** El tipo de objeto que puede ser un accesorio, un arma, una poción o una piedra recuperadora.
- **Values:** Guarda la lista de valores para cada estadística de *TargetStats*.
- **TargetStats:** Guarda una lista con las estadísticas a las que afecta el objeto al equiparlo o al usarlo.
- **Price:** El precio del objeto en la tienda.
- **Description:** La descripción del objeto.

Por ultimo existe un objeto llamado *ItemCatalog* que guarda la lista de objetos del juego tal y como se muestra en la figura 7.13.



Figura 7.12: Creación de objetos

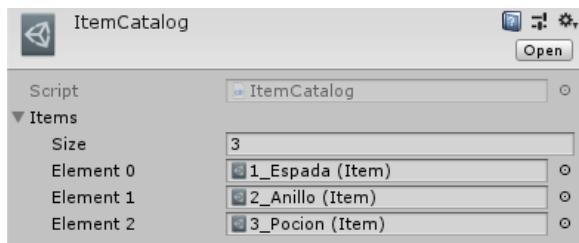


Figura 7.13: Catálogo de objetos

7.3.2. Gremio

En la escena del gremio se puede ver la lista de misiones disponibles para aceptarlas y las ya aceptadas con su progresión total. Una vez conseguida la progresión total se puede entregar la misión ganando el dinero y la experiencia mostrada.

Como se puede observar en las figuras en la parte superior se muestra el nombre del gremio que corresponde con una de las facultades de la Universidad, también aparecen los tres iconos de las facciones y la puntuación de las mismas la cual aumenta al derrotar jefes o entregar misiones.

En la parte inferior se encuentra el botón de sanar que recupera toda la vida y el mana del jugador, ademas junto a las piedras de recuperación es la única manera de recuperar la vida del jugador en caso de que esta llegue a cero. También existen dos botones que aparecen al pulsar sobre las misiones, uno para aceptarlas y otro para completarlas en caso de que se haya conseguido la progresión indicada.

Existe una clase *Quest* que es la que guarda la información de las misiones



Figura 7.14: Misión no aceptada



Figura 7.15: Misión completada

mostradas en la escena, las variables contenidas en la clase son las siguientes:

- **Name:** El nombre de la misión.
- **Experience:** La experiencia recibida al completar la misión.
- **Money:** El dinero recibido al completar la misión.
- **Description:** La descripción de la misión.
- **Progress:** El número de enemigos derrotados.
- **Number:** El número que hay que derrotar para completar la misión.
- **EnemyName:** El nombre del enemigo que hay que derrotar.
- **Status:** La cadena de texto que muestra el estado de la misión, esta cambia dependiendo del progreso de la misión o de si esta aceptada.

7.3.3. Menú

En la escena del menú se puede alternar entre tres paneles, uno con el inventario, otro con la lista de misiones y otro con la lista de habilidades.

En la parte superior del panel del inventario se muestran los diferentes datos del jugador y en la intermedia una lista con los objetos que posee. Como se muestra en las figuras 7.17 y 7.18 al seleccionar un objeto aparece su descripción en la parte inferior y una serie de botones junto a la lista. En caso de que sea un objeto usable aparece un botón para usarlo y si el objeto es equipable sera para equiparlo. En ambos casos también aparece un botón para eliminar el objeto.



Figura 7.16: Inventario



Figura 7.17: Inventario con objeto usable seleccionado



Figura 7.18: Inventario con objeto equipable seleccionado

En la pestaña de misiones se muestran las misiones aceptadas, los datos mostrados de las mismas son iguales a los presentados en la escena del gremio. En el centro del panel se muestra la lista de misiones y en la parte inferior la descripción de la misión seleccionada tal y como se muestra en la figura 7.19.



Figura 7.19: Misiones aceptadas

Por último en el panel de habilidades se muestra la lista de habilidades. Como se puede ver en la figura 7.20 la información mostrada incluye el nombre de la habilidad, la duración de su efecto en turnos, el coste de puntos de mana, si afecta a múltiples objetivos y una descripción de la habilidad seleccionada en la parte inferior.

Las variables incluidas en los objetos *Skill* pueden verse en la figura 7.21 y son las siguientes:

- **SkillName:** El nombre de la habilidad.
- **SkillType:** El tipo de habilidad, pudiendo servir para realizar daño, aumentar alguna estadística del jugador o disminuir la de un enemigo.
- **Value:** El valor que suma o resta a la estadística seleccionada según el tipo de habilidad.
- **MultiTarget:** Indica si afecta a múltiples objetivos.
- **TargetStat:** La estadística a la que afecta la habilidad.
- **ManaCost:** El coste de mana de la habilidad.
- **ActiveTurns:** El número de turnos en el que la habilidad esta activa.

- **Description:** Una descripción de la habilidad.

También existe un objeto llamado *SkillCatalog* que guarda la lista de habilidades del juego tal y como se muestra en la figura 7.22.



Figura 7.20: Habilidades aprendidas

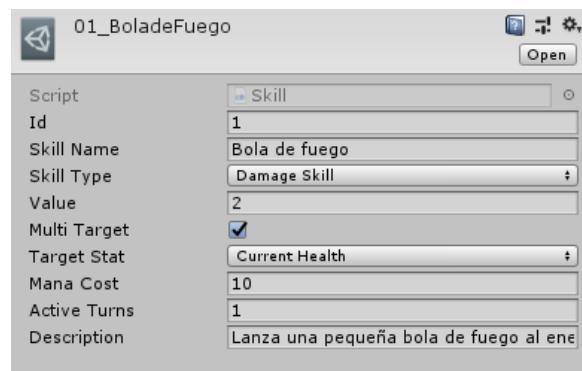


Figura 7.21: Creación de habilidades

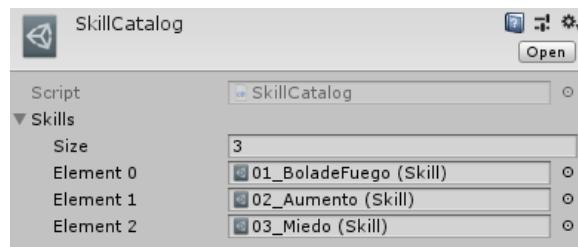


Figura 7.22: Catálogo de habilidades

7.4. Entrega 4

Esta entrega tiene como objetivo conseguir un prototipo con conexión al servidor de lo realizado hasta el momento, teniendo para ello que desarrollar la base de datos, la aplicación restful y una clase para realizar las peticiones desde Unity.

En la figura 7.23 se pueden observar las tablas utilizadas, su descripción se muestra a continuación:

- **accepted_quest**: Mantiene las misiones aceptadas por el jugador y su progreso.
- **delivered_quest**: Guarda las misiones que han sido completadas y entregadas por el jugador.
- **enemy**: Mantiene los enemigos del juego.
- **faction**: Guarda las facciones con su puntuación.
- **guild**: Guarda la posición de los gremios y sus nombres.
- **inventory**: Mantiene los objetos del jugador.
- **item**: Mantiene los objetos del juego.
- **player**: Guarda los datos de los jugadores.
- **player_skill**: Mantiene las habilidades de los jugadores.
- **quest**: Guarda las misiones disponibles.
- **shop**: Guarda la posición de las tiendas.
- **shop_item**: Mantiene los objetos de las tiendas.
- **skill**: Guarda las habilidades del juego.
- **user**: Mantiene los datos de los usuarios.

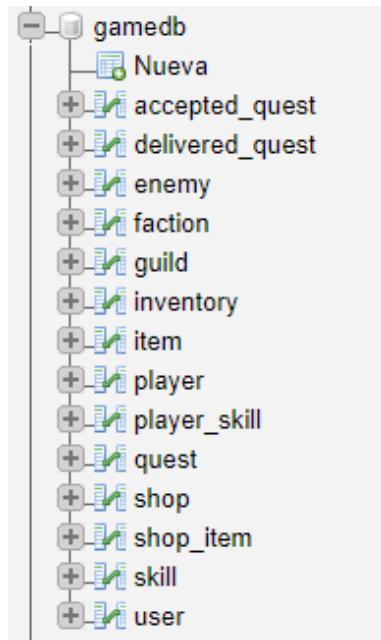


Figura 7.23: Base de datos

Para desarrollar el servidor se ha utilizado *Slim*, un *micro framework* para construir aplicaciones *Restful* en PHP. El principal archivo que hay que usar en *Slim* para desarrollar la aplicación es el de *routes.php*, en donde se incluyen las URLs que usara la API y los métodos asociados a las mismas.

```
// Routes
// Grupo de rutas para el API
$app->group('/api', function () use ($app) {
    $app->get('/user/{email}', 'getUser');
    $app->get('/shops/{email}', 'getUser');
    $app->get('/factions_points', 'getFactionsPoints');
    $app->put('/shops_coordinates', 'getShopsCoordinates');
    $app->put('/guilds_coordinates', 'getGuildsCoordinates');
    $app->get('/shop/{id}', 'getShop');
    $app->get('/user_player/{email}', 'getPlayer');
    $app->put('/create', 'createUser');
    $app->put('/create_player', 'createPlayer');
    $app->put('/update_coordinates', 'updateCoordinates');
    $app->put('/login', 'userLogin');
    $app->put('/delete_item', 'deleteItem');
    $app->put('/update_attributes', 'updateAttributes');
    $app->put('/add_item', 'addItem');
    $app->put('/equip_items', 'equipItems');
    $app->put('/quests', 'getQuests');
    $app->put('/accept_quest', 'acceptQuest');
    $app->put('/accepted_quests', 'getAcceptedQuests');
});
```

Figura 7.24: API REST

En la figura 7.24 se muestra la lista de URLs y sus métodos, a continuación se aporta una descripción de cada una de las URLs:

- **/factions_points**: Devuelve los puntos de las facciones.
- **/shops_coordinates**: Devuelve las coordenadas de las tiendas cercanas al usuario.
- **/guilds_coordinates**: Devuelve las coordenadas de los gremios cercanos al usuario.
- **/shop/{id}**: Devuelve los objetos de la tienda identificada por el id dado.
- **/user_player/{email}**: Devuelve los datos del jugador indicados por el email.
- **/create**: Crea un usuario con el email y contraseña dados siempre que el email no exista ya.
- **/create_player**: Crea un jugador asociado al usuario dado con el nombre, facción y clase aportados.
- **/update_coordinates**: Actualiza las coordenadas del usuario.
- **/login**: Comprueba si el usuario y la contraseña introducidos son válidos.
- **/delete_item**: Elimina un objeto del inventario de un jugador.
- **/update_attributes**: Actualiza las estadísticas de un jugador.
- **/add_item**: Añade un objeto al inventario de un jugador.
- **/equip_items**: Añade un arma y accesorio al jugador.
- **/quests**: Devuelve la lista de misiones que no hayan sido entregadas.
- **/accept_quest**: Añade una misión a la lista de misiones aceptadas del jugador.
- **/accepted_quests**: Devuelve la lista de misiones aceptadas del jugador.

Para realizar las peticiones en Unity se tiene que hacer uso de corutinas que es el nombre que reciben las funciones que tienen la capacidad de pausar su ejecución y devolver el control a Unity pudiendo continuar donde lo dejó en el siguiente *frame*. El tipo de retorno de este tipo de función es *IEnumerator*, también tiene que tener una instrucción de retorno *yield* que

marca donde se pausara la ejecución para reanudarse en el siguiente *frame*. La llamada a la corutina se realiza con la función *StartCoroutine(funcion Ienumerator)*.

```
private string url = "http://localhost/api/";

public IEnumerator postRequest(JSONObject json, string path, System.Action<JSONObject> callBack) {
    JSONObject response;
    using (UnityWebRequest www = UnityWebRequest.Put(url + path, json.Print()))
    {
        www.SetRequestHeader("Content-Type", "application/json");
        yield return www.SendWebRequest();

        if (www.isNetworkError || www.isHttpError)
        {
            Debug.Log(www.error);
        }
        else
        {
            response = new JSONObject(www.downloadHandler.text);
            if(callBack != null) {
                callBack(response);
            }
        }
    }
}
```

Figura 7.25: Método PUT

```
public IEnumerator getRequest(string path, System.Action<JSONObject> callBack) {
    JSONObject response;
    using (UnityWebRequest www = UnityWebRequest.Get(url + path))
    {
        www.SetRequestHeader("Content-Type", "application/json");
        yield return www.SendWebRequest();

        if (www.isNetworkError || www.isHttpError)
        {
            Debug.Log(www.error);
        }
        else
        {
            Debug.Log("Form upload complete!");
            response = new JSONObject(www.downloadHandler.text);
            Debug.Log(response.Print());
            callBack(response);
        }
    }
}
```

Figura 7.26: Método GET

El objeto *ServerConnection* es el encargado de realizar las peticiones al servidor, en las figuras 7.25 y 7.26 se pueden ver los métodos para realizar

las peticiones. Los métodos tienen como parámetro la URL, una función *callback* a la que se llamará cuando se reciba una respuesta y en el caso de la función *PUT* el *JSON* que se enviará al servidor, tan solo se han podido usar métodos *PUT* y *GET* y no *POST* ya que este último no funciona bien en Unity al enviar *JSON* puesto que este llega corrupto al servidor.

7.4.1. Ejemplo de la actualización de las coordenadas

A la hora de actualizar las coordenadas es usado un método *put* con la URL */update_coordinates*, es necesario enviar la información necesaria, que incluirá el usuario, la contraseña del usuario, la longitud y la latitud, todo ello en formato *Json*. El envío de esta información es mostrado en la figura 7.27.

```
JSONObject json = new JSONObject(JSONObject.Type.OBJECT);
json.AddField("latitude", position.x.ToString().Replace(",", "."));
json.AddField("longitude", position.y.ToString().Replace(",", "."));
json.AddField("email", gameManager.getUser().getEmail());
json.AddField("user_password", gameManager.getUser().getPassword());
StartCoroutine(gameManager.getServerConnection().postRequest(json, "update_coordinates", getResponse));
```

Figura 7.27: Información necesaria coordenadas

En el servidor se comprueba que el usuario y la contraseña introducidos son válidos y tras esto se actualizan las coordenadas de ese usuario en la base de datos, incluyendo también la fecha de la actualización puesto que más tarde se usará para determinar si ese usuario está conectado o desconectado basándose en la antigüedad de la actualización. El código de esta actualización en el servidor puede verse en la figura 7.28.

El método de la figura 7.28 también se usa para devolver la posición de los usuarios cercanos. Se puede observar que se devuelve su latitud, longitud, facción, id y nombre en formato *Json*.

Los datos del servidor son captados por el método mostrado en la figura 7.29 y tras obtener la información del *Json* los datos son pasados al método de la figura 7.30 donde los elementos, en este caso jugadores, del mapa son actualizados de acuerdo a la nueva información obtenida.

```

function updateCoordinates($request) {
    $data = json_decode($request->getBody());
    $user_id = getUserId($data->email, $data->user_password);

    if($user_id != 0) {
        $sql = "UPDATE user SET latitude = :latitude, longitude = :longitude, last_connection = NOW() WHERE email=:email";
        try {
            $db = getConnection();
            $stmt = $db->prepare($sql);
            $stmt->bindParam("latitude", floatval($data->latitude));
            $stmt->bindParam("longitude", floatval($data->longitude));
            $stmt->bindParam("email", $data->email);
            $stmt->execute();
            $db = null;
        }
        // Se comprueba si hay jugadores cerca cuya conexión no haya caducado y se devuelven las coordenadas
        $sql = "SELECT id, latitude, longitude, last_connection,
        ( 6371 * acos(cos(radians(latitude)) * cos(radians(longitude)) + sin(radians(latitude)) * sin(radians(longitude))) )
        AS distance FROM user WHERE email=:email AND last_connection > DATE_SUB(CURRENT_TIMESTAMP, INTERVAL 20 SECOND) HAVING distance < 0.5 ORDER BY distance";
        $sql2 = "SELECT id, faction, name FROM player WHERE user_id=:user_id";

        $db = getConnection();
        $stmt = $db->prepare($sql);
        $stmt->bindParam("email", $data->email);
        $coordinates = array();
        $players_id = array();
        $players_factions = array();
        $players_names = array();
        $stmt->execute();
        while ($row = $stmt->fetch())
        {
            if(!empty($row['latitude']) && !is_null($row['latitude'])) && !empty($row['last_connection']) && !is_null($row['last_connection']))
            {
                $data = strval($row['latitude']) . "," . strval($row['longitude']);
                $coordinates[] = array("coordinates" => $data);
                $user_id = $row['id'];
                $db = getConnection();
                $second_stmt = $db->prepare($sql2);
                $second_stmt->bindParam("user_id", $user_id);
                $second_stmt->execute();
                while($second_row = $second_stmt->fetch())
                {
                    $players_id[] = array('player_id' => $second_row['id']);
                    $players_factions[] = array('player_faction' => $second_row['faction']);
                    $players_names[] = array('player_name' => $second_row['name']);
                }
            }
        }
        $array = array("players_coordinates" => $coordinates, "players_id" => $players_id, "players_factions" => $players_factions, "players_names" => $players_names);
        echo json_encode($array);
    }
}

```

Figura 7.28: Actualización de las coordenadas en el servidor

7.5. Entrega 5

El objetivo de esta entrega es realizar un prototipo con el combate contra enemigos locales y con un chat entre jugadores aliados cercanos.

7.5.1. Combate

La escena de combate es mostrada en la figura 7.31. Como puede observarse las acciones posibles son atacar, defender, usar habilidades u objetos e intentar huir, estas acciones se encuentran descritas en la sección 3.1.2 del capítulo 3.

Existen dos objetos principales en la escena del combate el *BattleManager* y el *CharacterController*, tan solo hay uno del primer tipo, sin embargo del segundo hay uno por cada participante del combate.

El *BattleManager* es el encargado de coordinar a los participantes, espera a que todos ellos seleccionen una acción y tras ello ejecuta las acciones seleccionadas. La información necesaria para realizar las acciones viene incluida en un objeto *BattleAction* que guarda la estadística a la que afectara la acción, el valor numérico que se sumara o restara a esa estadística, el objetivo de la acción, el tipo de acción, que corresponde con las posibles

```

public void getResponse(JSONObject json) {
    List<string> coordinates = new List<string>();
    GameObject playerSpawnerManager = GameObject.Find("PlayerSpawnerManager");
    SpawnOnMap spawnOnMap = playerSpawnerManager.GetComponent<SpawnOnMap>();
    JSONObject array = json.GetField("players_coordinates");
    foreach(JSONObject j in array.list) {
        coordinates.Add(j.GetField("coordinates").str);
    }

    List<int> players_id = new List<int>();
    JSONObject id_array = json.GetField("players_id");
    foreach(JSONObject j in id_array.list) {
        players_id.Add(int.Parse(j.GetField("player_id").str));
    }

    List<string> players_factions = new List<string>();
    JSONObject factions_array = json.GetField("players_factions");
    foreach(JSONObject j in factions_array.list) {
        players_factions.Add(j.GetField("player_faction").str);
    }

    List<string> players_names = new List<string>();
    JSONObject names_array = json.GetField("players_names");
    foreach(JSONObject j in names_array.list) {
        players_names.Add(j.GetField("player_name").str);
    }

    spawnOnMap.setPlayersSpawnsCoordinates(coordinates, players_id, players_factions, players_names);
}

```

Figura 7.29: Captura de la información obtenida del servidor

```

public void setPlayersSpawnsCoordinates(List<string> coordinates, List<int> players, List<string> factions, List<string> names) {
    GameObject[] playersSpawns = GameObject.FindGameObjectsWithTag("PlayerSpawn");
    GameManager gameManager = GameManager.instance;
    // Se eliminan los spawns anteriores
    foreach(GameObject player in playersSpawns) {
        GameObject.Destroy(player);
        isReady = false;
    }
    if(filter.canShowPlayers()) {
        _locationStrings = new string[coordinates.Count];
        for(int i = 0; i < _locationStrings.Length; i++) {
            _locationStrings[i] = coordinates[i];
        }
        _locations = new Vector2d[_locationStrings.Length];
        _spawnedObjects = new List<GameObject>();
        for (int i = 0; i < _locationStrings.Length; i++)
        {
            var locationString = _locationStrings[i];
            _locations[i] = Conversions.StringToLocation(locationString);
            var instance = Instantiate(_markerPrefab);
            instance.GetComponent<PlayerSpawner>().setPlayer(players[i], factions[i], names[i]);
            if(instance.GetComponent<PlayerSpawner>().getFaction() == gameManager.getPlayer().getFaction()) {
                instance.GetComponent<SpriteRenderer>().color = Color.green;
            } else {
                instance.GetComponent<SpriteRenderer>().color = Color.red;
            }
            instance.transform.localPosition = _map.GeoToWorldPosition(_locations[i], true);
            instance.transform.localScale = new Vector3(_spawnScale, _spawnScale, _spawnScale);
            _spawnedObjects.Add(instance);
        }
        isReady = true;
    }
}

```

Figura 7.30: Actualización de los elementos mostrados en el mapa

acciones en combate, y una variable de tipo *bool* que indica si la acción ha



Figura 7.31: Escena de combate

sido seleccionada o no, esta última variable sera la que se comprobara para saber si todos han seleccionado su acción.

El *CharacterController* guarda las estadísticas del participante en combate y su *BattleAction*, el *CharacterController* es el encargado por lo tanto de llenar el *BattleAction* con la acción seleccionada y enviarlo al *BattleManager*. En la figura 7.32 se puede observar que tras ejecutar un turno se muestra en pantalla un texto con las acciones que cada jugador a realizado.

La probabilidad de encuentros contra enemigos aumenta mientras mas distancia se ha recorrido sin que se produzca un encuentro, el método que se ha seguido para ello puede observarse en la figura 7.33, el encuentro se producirá si un aleatorio entre uno y cien es mayor que un valor calculado de restar a cien la distancia recorrida sin encuentros por otro aleatorio, de esta manera mientras mayor sea la distancia mayor sera la probabilidad de iniciar el encuentro.

En esta entrega también se han agregado los combates contra jefes, los



Figura 7.32: Log combate

```
distanceWithoutEncounter += CalculateDistance((float)lastPosition.x, (float)lastPosition.y, (float)position.x, (float)position.y);
float rand1 = Random.Range(1, 100);
float rand2 = 100.0f - (distanceWithoutEncounter*Random.Range(0.6f, 3.0f));
Debug.Log(rand1 + " " + rand2);
if(rand1 > rand2) {
    distanceWithoutEncounter = 0;
    gameManager.changeScene("BattleScene");
}
```

Figura 7.33: Calculo probabilidad de encuentro

cuales se encuentran en una localización y se necesitan un número de puntos para poder enfrentarlos. Los puntos son llamados puntos de valentía y se obtienen al recorrer cien metros. En la figura 7.34 se puede observar que cuando no se tiene el número necesario de puntos las espadas aparecen rojas y el jugador no podrá enfrentarse al jefe.



Figura 7.34: Jefes en el mapa

```
private float CalculateDistance(float lat_1, float long_1, float lat_2, float long_2)
{
    int R = 6371;
    var lat_rad_1 = Mathf.Deg2Rad * lat_1;
    var lat_rad_2 = Mathf.Deg2Rad * lat_2;
    var d_lat_rad = Mathf.Deg2Rad * (lat_2 - lat_1);
    var d_long_rad = Mathf.Deg2Rad * (long_2 - long_1);
    var a = Mathf.Pow(Mathf.Sin(d_lat_rad / 2), 2) + (Mathf.Pow(Mathf.Sin(d_long_rad / 2), 2) * Mathf.Cos(lat_rad_1) * Mathf.Cos(lat_rad_2));
    var c = 2 * Mathf.Atan2(Mathf.Sqrt(a), Mathf.Sqrt(1 - a));
    var total_dist = R * c * 1000; // Se pasa a metros
    return total_dist;
}
```

Figura 7.35: Calculo de la distancia entre dos coordenadas

El cálculo de la distancia se realiza guardando las coordenadas cada cinco segundos y midiendo la distancia entre las nuevas coordenadas medidas y las de hace cinco segundos. El cálculo de la distancia entre estas dos coordenadas se realiza mediante la formula de Haversine cuyo código puede verse en la figura 7.35.

Al derrotar a los enemigos se obtienen recompensas de dinero y experiencia como puede observarse en 7.36. Para poder realizar lo anterior se han

añadido varias tablas a la base de datos que son:

- **enemy**: Guarda los jefes disponibles.
- **boss_enemy**: Guarda las coordenadas de los jefes, su enemigo asociado y los puntos necesarios para enfrentarlo.

Por último también se han añadido métodos a la *API Restful* para poder gestionar los puntos necesarios para enfrentar a los jefes y poder adquirir las recompensas, estas son:

- **/use_bravery_points**: Gasta los puntos de valor indicados.
- **/add_bravery_point**: Añade un punto de valor al jugador.
- **/reward**: Añade las recompensas al jugador dependiendo del número de enemigos derrotado.



Figura 7.36: Recompensas combate

7.5.2. Chat

La escena de chat es mostrada en la figura 7.37, esta puede iniciarse al encontrar un jugador cercano de la misma facción.

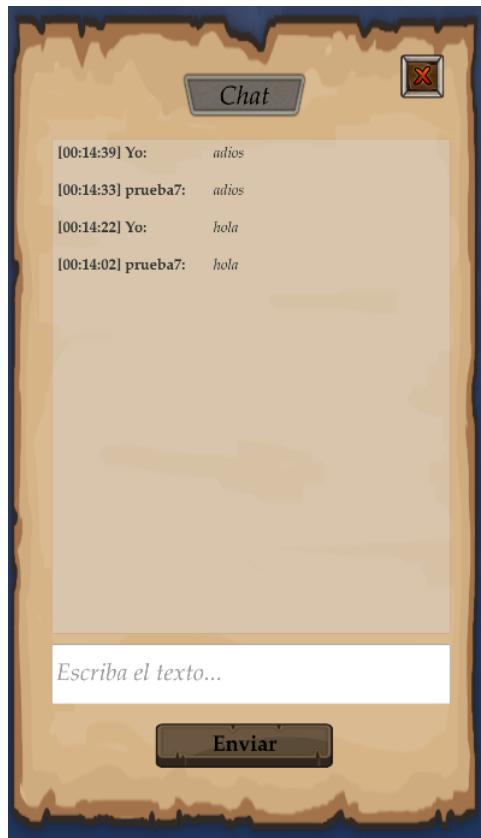


Figura 7.37: Escena de chat

En la escena existe un *GameObject* llamado *ChatManager* que es el que se encarga de realizar las peticiones al servidor y recibirlas, también es el encargado de cambiar las líneas de chat mostradas.

Se ha añadido una tabla llamada *chat_line* a la base de datos, esta incluye el identificador del propietario del mensaje, el identificador del jugador al que se le envió, el texto del mensaje y la fecha a la que se envió.

Por último también se han añadido dos URLs en la *API Restful* que son:

- **/set_chat_line:** Inserta una linea de chat con los datos indicados.
- **/get_chat:** Obtiene todos los mensajes de chat de los dos jugadores indicados.

Capítulo 8

Conclusiones y trabajo futuro

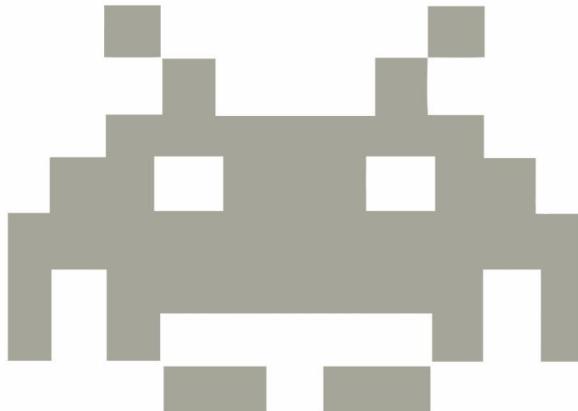
Bibliografía

- [1] “RPG”. Valentina Sancler. Recuperado en enero de 2019 de <https://www.euston96.com/rpg/>
- [2] “[RETRO GAMING] Un poco de la historia de los RPG — OPINIÓN”. Adell. 2015. Recuperado en enero de 2019 de <http://www.destinorpg.es/2015/08/historia-de-los-videojuegos-los.html>
- [3] “La Historia de juegos de rol online (MMORPG)”. Tomás Garcia. 30 julio 2009. Recuperado en enero de 2019 de <https://www.neoteo.com/la-historia-de-juegos-de-rol-online-mmorpg/>
- [4] “Ingress, cómo un juego de realidad virtual te cambia la vida”. Alkar. 26 Diciembre 2013. Recuperado en enero de 2019 de <https://www.xatakandroid.com/juegos-android/ingress-como-un-juego-de-realidad-virtual-te-cambia-la-vida>
- [5] “Geolocalización, qué es y cómo funciona”. KZgunea. 31 Marzo 2017. Recuperado en enero de 2019 de <http://kzgunea.blog.euskadi.eus/blog/2017/03/31/geolocalizacion-que-es/>
- [6] ‘Top Game Engines’. Hady ElHady. 7 Diciembre 2017. Recuperado en Marzo de 2019 <https://instabug.com/blog/game-engines/>
- [7] ‘Ya sea con un equipo profesional o como freelancer, aficionado o principiante, siempre habrá un plan de Unity para ti.’. Unity Technologies. Recuperado en Marzo de 2019 <https://store.unity.com/es>
- [8] ‘Game Engine Analysis’. Game Sparks Technologies. Recuperado en Marzo de 2019 <https://www.gamesparks.com/blog/game-engine-analysis/>
- [9] “10 Popular PHP frameworks in 2019”. Alice Njenga. 21 Noviembre 2018. Recuperado en Febrero de 2019 <https://raygun.com/blog/top-php-frameworks/>
- [10] “Mapbox: el SDK de mapas abierto”. Pablo Guardiola. 10 Octubre 2018. Recuperado en Febrero de 2019 <https://www.genbeta.com/desarrollo/mapbox-el-sdk-de-mapas-aberto>

- [11] “En busca del Diseño Centrado en el Usuario (DCU): definiciones, técnicas y una propuesta”. Jordi Sánchez. 5 Septiembre 2011. Recuperado en Marzo de 2019 http://www.nosolousabilidad.com/articulos/dcua.htm?utm_source=iNeZha.com&utm_medium=im_robot&utm_campaign=iNezha
- [12] “Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems”. ISO 2010. <https://www.sis.se/api/document/preview/912053/>
- [13] “Metodología de prototipos”. mariCh. 31 Marzo 2016. Recuperado en Marzo de 2019 <http://marich.blogspot.es/1459397526/metodologia-de-prototipos/>
- [14] “Desarrollo iterativo e incremental”. proyectosagiles.org. Recuperado en Marzo de 2019 <https://proyectosagiles.org/desarrollo-iterativo-incremental/>

Apéndices

Documento de diseño de videojuegos



Game Design Document

Geono RPG

Blas Varela López

Blas Varela López

Curso 2018/2019

Fecha: 18/06/2019

Versión: 1.0.0.

Game Design Document

Control de Versiones

Versión	Fecha	Cambios
1.0.0	13/01/2014	Creación de estructura del GDD ESCAV

Tabla de contenido

Control de Versiones	3
1 Información general	7
1.1 Resumen del juego	7
1.2 Objetivos a alcanzar por el juego	7
1.3 Justificación del Juego	7
1.4 Core gameplay	7
1.5 Características del juego	8
1.5.1 Género	8
1.5.2 Número de jugadores	9
1.5.3 Plataformas de destino	9
1.5.4 Estética y arte del juego	9
1.5.5 Resumen de Historia	9
1.6 Características del jugador	9
1.7 Recursos iniciales	10
2 Mecánicas	10
2.1. Elementos Juego: categorías	10
2.2 Elementos de juego: Núcleo principal	14
2.3 Reglas	15
2.3.1 Reglas de interacción	16
2.4 Elementos de registro y progreso	19
2.5 Elementos de Jugabilidad y experiencia del jugador	19
2.6 Otros elementos	19
2.7 Lista de recursos activos	19
3 Dinámica	19
3.1 Mundo de Juego. Universo Virtual	20
3.1.1 Detalles del Juego en temática	20
3.1.2 Descripción de misiones, niveles o capítulos.	20
3.1.3 Historia detallada	21
3.2 Misiones	22
3.2.1 Objetivos	22
3.2.1.1 Primarios	22
3.2.1.2 Secundarios	22
3.2.2 Recompensas	22
3.2.2.1 Implícitas	22

3.2.2.2 Explícitas	22
3.2.3 Desafíos	22
3.2.4 Otros elementos de Misiones / Contenidos / capítulos	22
3.2.5 Flujo de las Misiones / Contenidos / capítulos	23
3.2.6 Zonas especiales	23
3.2.6.7 Experiencia del jugador	23
3.3 Interfaz del juego	23
3.4 Controles de la interfaz	25
3.5 Aprendizaje del juego	25
3.6 Equilibrio Juego	25
4 Estética y Arte	26
4.1 Elementos básicos del juego	26
4.2 Elementos del mundo	26
4.3 Elementos de registro de progreso	26
4.4 Otros elementos visuales	26
4.5 Mundo del Juego	26
4.6 Misiones / niveles / capítulos	26
4.7 Áreas especiales	26
4.8 Interfaz del juego	26
5 Experiencia	27
5.1 Jugabilidad Intrínseca	27
5.2 Jugabilidad Mecánica	27
5.3 Jugabilidad Interactiva	27
5.4 Jugabilidad Artística	27
5.5 Jugabilidad Intrapersonal	27
5.6 Jugabilidad Interpersonal	28
6 Marketing y publicidad	28
6.1 Poster o portada/contraportada del juego	28
6.2 Anuncios	28
6.3 PEGI	28
7 Limitaciones y supuestos	28
6.1 Limitaciones técnicas	28
6.3 Restricciones comerciales	28
7 Información del documento	29
7.1 Definición, acrónimos y abreviaturas .	29
7.2 Referencias.	29

7.3 Ficheros adjuntos

29

1 Información general

Este es el documento de diseño de Geono RPG. Es un videojuego 2D para Android que utiliza Unity como motor gráfico. Este escrito tiene como objetivo plasmar los elementos que contiene Geono RPG.

1.1 Resumen del juego

Geono RPG es un videojuego geolocalizado en el que el jugador avanza por la ciudad enfrentándose a diferentes monstruos que han aparecido como consecuencia de una sobrecarga en la capa de la conciencia que procesa todo lo que ocurre en el universo y cuyas anomalías alteran la realidad.

Los monstruos proporcionan experiencia para aumentar el nivel, objetos y dinero. Los jugadores pueden usar el dinero conseguido al enfrentarse a los monstruos para comprar nuevos objetos en las tiendas.

Las diferentes facciones intentan conseguir la mayor reputación derrotando a los monstruos más poderosos y completando misiones. Los jugadores de diferentes facciones también pueden enfrentarse entre si para conseguir reputación y debilitar a la facción rival.

1.2 Objetivos a alcanzar por el juego

Los objetivos del juego son los siguientes:

- **Subir de nivel:** A mayor nivel los enemigos que aparecen son más poderosos y por tanto las misiones y los jefes derrotados aportan más reputación para la facción.
- **Ganar reputación:** Es el principal objetivo del juego, aporta competitividad entre los jugadores y por tanto incentiva que estos suban de nivel para conseguir una mayor reputación.

1.3 Justificación del Juego

Actualmente los juegos geolocalizados han aumentado su popularidad y siguen surgiendo nuevos intentos por innovar donde algunos consiguen triunfar, por ello hay que intentar probar nuevos géneros que funcionen con las técnicas que aportan este tipo de juegos y el género de los RPGs es uno con el que todavía no se ha intentado innovar del todo.

1.4 Core gameplay

En Geono RPG se ofrece un mapa, el cual corresponde con el mundo real, por el que el

jugador avanza enfrentándose contra diferentes monstruos que al ser derrotados ofrecen recompensas. El jugador también interactúa con varios elementos que se encuentran en el mapa los cuales son los siguientes:

- **Tiendas:** Se pueden comprar objetos con el dinero obtenido como recompensa al derrotar monstruos y al completar misiones.
- **Jefes:** Son los monstruos más poderosos, ofrecen mejores recompensas y reputación para la facción.
- **Gremios:** Son los puntos de encuentro de las facciones, en estos puede consultarse la reputación de las facciones, aceptar las misiones y completar las mismas.

El combate se realiza por turnos, todos los participantes eligen su próxima acción y estas se ejecutan por orden dependiendo de la velocidad de cada participante. En el combate pueden utilizarse ataques básicos o habilidades para hacer daño o influir en las características del enemigo o propias, también puede utilizarse objetos para recuperar vida o maná el cual es usado al utilizar habilidades.

Los jugadores pueden consultar en el menú sus habilidades, las misiones activas y completadas y los objetos de los que disponen. Pueden equiparse determinados objetos en los espacios correspondientes mejorando las características del jugador en combate.

1.5 Características del juego

Esta sección describe las características principales de Geono RPG.

1.5.1 Género

El juego es el resultado de la fusión de dos géneros, por un lado el género RPG que fue uno de los primeros géneros en aparecer y por otro lado el género de los juegos geolocalizados que han aparecido en los últimos años.

Al pertenecer al género RPG el juego posee una serie de características comunes al mismo, el jugador irá evolucionando a lo largo de su aventura mediante la obtención de puntos de experiencia en los combates y misiones que le permitirán subir de nivel, esto aumentará sus estadísticas y desbloqueara nuevas habilidades dependiendo de la clase seleccionada. El jugador también contará con una serie de artículos que le ayudarán a lo largo de su aventura, estos son obtenidos al derrotar enemigos, al completar misiones o comprandolos en las tiendas. Por último el juego cuenta con un sistema de combate por turnos clásico que es algo común en los RPG, en este sistema los participantes eligen su próxima acción y una vez todos la han elegido estas se ejecutan por orden dependiendo de su destreza.

Por otro lado al ser un juego geolocalizado también comparte una serie de características con los mismos, la localización en el mundo real pasa a ser algo esencial puesto que dependiendo de su posición un jugador puede o no realizar ciertas acciones como acceder a las tiendas, enfrentarse a jefes o adquirir y entregar misiones en los gremios. Los jugadores también pueden enfrentarse entre ellos si se encuentran

en localizaciones cercanas siempre que no pertenezcan a la misma facción, existe además cierta competitividad entre las facciones puesto que estas ganan puntos por las tareas completadas.

1.5.2 Número de jugadores

El juego cuenta con multijugador tanto competitivo como cooperativo. Los jugadores podrán librarse combates entre ellos y chatear para compartir información de la localización de los jefes en el mundo. El número de jugadores que estaría destinado a manejar sería el número de estudiantes de la Universidad.

1.5.3 Plataformas de destino

La plataforma de destino es Android 7.0 o versiones superiores.

1.5.4 Estética y arte del juego

El juego transcurre en la actualidad pero mezclada con temática de fantasía medieval puesto que los jugadores usarán objetos típicos de esta época y existirán otras características de este estilo como la magia.

1.5.5 Resumen de Historia

La Capa de la Consciencia ha ido alcanzando su límite en las zonas con una alta concentración de personas debido al pensamiento de las mismas, algo todavía más notable en lugares con un alto nivel de pensamiento como las Universidades. Esto ha desembocado en alteraciones en la realidad que han provocado la aparición de monstruos, pero gracias a estas alteraciones los humanos han ganado nuevas capacidades mediante las cuales son capaces de combatirlos. Las Escuelas de las Universidades se encuentran en un enfrentamiento por intentar ganar la mayor reputación en un mundo en el que han ganado aún más importancia.

Este mundo plantea un dilema en el que si las Universidades dejarán de enseñar la Capa de la Consciencia se iría reparando al bajar el nivel de pensamiento de esas zonas, pero esto impediría a la humanidad seguir avanzando.

1.6 Características del jugador

Se ha usado la técnica Personas para realizar tres perfiles de jugadores que son mostrados en la tabla inferior.

Nombre	Detalles	Objetivo
Ana	Tiene 21 años, estudia ingeniería civil y le	Buscar algún juego de

	gustan todo tipo de videojuegos, aunque los juegos de móviles no le acaban de convencer.	móvil que le guste.
Roberto	Estudia magisterio y tiene 18 años, principalmente juega en móvil y tiene consola pero apenas la usa, suele buscar los juegos nuevos que van apareciendo en móvil para probarlos.	Probar los nuevos juegos para móvil.
Raquel	Tiene 20 años, estudia farmacia y apenas juega videojuegos, tan solo en móvil y principalmente Pokemon Go.	Entretenerse en sus desplazamientos con algún juego nuevo en el móvil.

1.7 Recursos iniciales

El tiempo previsto es de cinco meses por lo que también será el tiempo de desarrollo disponible, tan solo se contará con un desarrollador y no se dispondrá de dinero para el proyecto.

2 Mecánicas

Esta sección describe los elementos de juego , sus atributos y sus reglas de interacción

2.1. Elementos Juego: categorías

En este sección se muestran los elementos del juego diferenciados por categorías, tal y como se muestra en la tabla inferior.

Nombre	Arte	Descripción	Historia
Jugador Mago		Su atributo principal es la inteligencia y se basa en hacer daño mediante poderosas habilidades.	Tras la aparición de la Capa de la Consciencia algunos estudiantes se centraron por completo en el estudio de la misma aprendiendo a alterar la realidad mejor que los demás y usando este conocimiento para crear poderosos ataques con elementos generados por ellos

			mismos.
Jugador Guerrero		Es una de las tres clases disponibles, su atributo principal es la fuerza y se basa en hacer daño mediante ataques básicos. Sus habilidades consisten en aumentar sus características.	Tras la aparición de la Capa de la Consciencia algunos estudiantes se centraron en aumentar su fuerza y alterar la realidad mejorando su propio cuerpo.
Jugador Espadachín		Su atributo principal es la destreza y se basa en hacer daño mediante ataques básicos con la realización de críticos. Sus habilidades consisten en disminuir las características del enemigo.	Tras la aparición de la Capa de la Consciencia algunos estudiantes se centraron en aumentar su destreza y en alterar la realidad empeorando el cuerpo de los rivales.
Facción Ciencia		Es una de las tres facciones, está constituida por los estudiantes de carreras de ciencias.	Tras la aparición de la Capa de la Consciencia las facultades cobraron gran importancia al ser las encargadas de entrenar a los estudiantes, con el tiempo empezaron a generarse disputas entre las distintas áreas de conocimiento, esto desembocó en la aparición de tres facciones enfrentadas siendo la de la ciencia una de ellas.
Facción Ingeniería		Es una de las tres facciones, está constituida por los estudiantes de carreras de ingeniería.	La facción de la ingeniería fue una de las que surgió de las disputas entre áreas de conocimiento.
Facción Arte y Humanidades		Es una de las tres facciones, está constituida por los estudiantes de arte y humanidades.	La facción de las artes y las humanidades fue una de las que surgió de las disputas entre áreas de conocimiento.

Poción		Es un tipo de objeto que recupera vida o mana.	Los estudios sobre la Capa de la Consciencia y la alteración de la realidad llevaron a la creación de líquidos que al ingerirlos eran capaces de curar las heridas y ayudar a recuperar la capacidad de alterar la realidad que es perdida tras un uso continuado.
Arma		Es un tipo de objeto que puede equiparse para aumentar las características y el daño base.	Tras la aparición de la Capa de la Consciencia se empezaron a utilizar de nuevo las armas del pasado, además gracias a los conocimientos adquiridos de la Capa de la Consciencia pudieron crearse de manera que alteraban la realidad cercana mejorando así las capacidades de quien la porte.
Accesorio		Es un tipo de objeto que puede equiparse para aumentar las características.	Tras la aparición de la Capa de la Consciencia gracias a los conocimientos adquiridos de la misma se pudieron crear objetos de manera que alteraban la realidad cercana mejorando así las capacidades de quien los porten.
Piedra de recuperación		Es un tipo de objeto que puede recuperar la vida de un jugador una vez que este ha sido debilitado por completo.	Tras la aparición de la Capa de la Consciencia aparecieron piedras cuyo contacto recuperaba por completo a aquellos que se encontraban al borde

			de la muerte, aunque su número es escaso.
Tienda		En un lugar que permite comprar objetos.	Algunas personas empezaron a vender los objetos que creaban con lo que empezaron a aparecer lugares destinados a este comercio.
Gremio		Es un lugar que permite aceptar misiones, entregarlas y recuperar al personaje.	Las facultades se convirtieron en centros que ofrecían misiones destinadas a proteger la ciudad ganando así prestigio. Los estudiantes heridos también pueden ser atendidos en estos lugares.
Enemigo		Aparecen al caminar, el jugador luchará contra ellos.	Tras las aparición de la Capa de la Consciencia aparecieron monstruos hostiles por la ciudad.
Jefe		Se encuentran en lugares fijos, son más poderosos que los enemigos normales, el jugador luchará contra ellos.	Algunos de los monstruos que aparecieron eran increíblemente poderosos con lo que los estudiantes tenían que estar muy bien preparados para poder derrotarlos.
Habilidad		Pueden ser utilizadas en los combates.	Los estudiantes desarrollaron su propia forma de alterar la realidad.
Misión		Son aceptadas y entregadas en los gremios, aportan experiencia y dinero como recompensa.	Los gremios empezaron a ofrecer misiones para proteger la ciudad ganando así prestigio. Los estudiantes que

			consiguen realizarlas reciben una compensación de dinero.
--	--	--	--

2.2 Elementos de juego: Núcleo principal

Esta sección describe los elementos del juego incluidos en alguna de las categorías de la sección anterior.

Nombre	Arte	Atributos	Categoría
Araña		Vida: 7 Mana: 7 Daño Base: 1 Fuerza: 6 Destreza: 7 Inteligencia: 1	Enemigo
Escorpión		Vida: 6 Mana: 5 Daño Base: 1 Fuerza: 7 Destreza: 5 Inteligencia: 1	Enemigo
Murciélagos		Vida: 5 Mana: 5 Daño Base: 2 Fuerza: 5 Destreza: 8 Inteligencia: 3	Enemigo
Rata		Vida: 8 Mana: 8 Daño Base: 4 Fuerza: 8 Destreza: 6 Inteligencia: 4	Enemigo
Slime		Vida: 11 Mana: 10 Daño Base: 1 Fuerza: 5	Enemigo

		Destreza: 2 Inteligencia: 2	
Esqueleto		Vida: 8 Mana: 8 Daño Base: 5 Fuerza: 8 Destreza: 4 Inteligencia: 5	Enemigo
Serpiente		Vida: 5 Mana: 5 Daño Base: 3 Fuerza: 8 Destreza: 8 Inteligencia: 2	Enemigo
Hombre Lagarto		Vida: 10 Mana: 10 Daño Base: 6 Fuerza: 9 Destreza: 7 Inteligencia: 7	Enemigo
Aurum		Vida: 22 Mana: 22 Daño Base: 18 Fuerza: 16 Destreza: 16 Inteligencia: 14	Jefe
Goliath		Vida: 16 Mana: 16 Daño Base: 12 Fuerza: 14 Destreza: 10 Inteligencia: 11	Jefe
Daemarbora		Vida: 14 Mana: 14 Daño Base: 10 Fuerza: 12 Destreza: 4 Inteligencia: 10	Jefe

2.3 Reglas

Esta sección describe las acciones válidas que el jugador puede realizar y la respuesta del juego a las mismas.

2.3.1 Reglas de interacción

Las reglas de interacción válidas se muestran en la tabla siguiente:

Pantalla	Reglas
Inicio de sesión	<ul style="list-style-type: none"> Al pulsar sobre el campo de texto email se puede escribir el email del usuario. Al pulsar sobre el campo de texto contraseña se puede escribir la contraseña del usuario. Al pulsar sobre el botón de conectar se cambia a la escena de mapa del mundo siempre que los datos introducidos sean correctos. El jugador se rellena con los datos asociados a ese usuario. Al pulsar sobre el botón de crear cuenta se cambia al panel de crear cuenta.
Creación de cuenta	<ul style="list-style-type: none"> Al pulsar sobre alguno de los campos de texto se podrá escribir de igual manera a como se hacía en el panel de iniciar sesión. Al pulsar sobre el botón de aceptar se crea una cuenta con los datos introducidos siempre que el email no esté en uso. Se cambiará al panel de crear jugador.
Creación de jugador	<ul style="list-style-type: none"> Al pulsar sobre el campo de texto de nombre se puede escribir un nombre para el jugador. Al pulsar sobre una de las secciones de clase se selecciona esa clase para el personaje. Al pulsar sobre una de las secciones de facción se selecciona esa facción para el personaje. Al pulsar sobre el botón de aceptar se creará el personaje con los datos introducidos y se cambia a la escena de mapa del mundo.
Mapa del mundo	<ul style="list-style-type: none"> Caminar en el mundo real hace que el personaje se mueva en el mundo virtual, al hacerlo se ganan puntos de valentía para enfrentar a los jefes y existe la posibilidad de que aparezca un enemigo.

	<ul style="list-style-type: none"> ● Al pulsar sobre el ícono de tienda se entra en la escena de tienda. ● Al pulsar sobre el ícono de gremio se entra en la escena de gremio. ● Al pulsar sobre el ícono de jugador aliado se abre la escena de chat con el jugador. ● Al pulsar sobre el ícono de jugador enemigo se inicia un combate contra el jugador. ● Al pulsar sobre el ícono de jefe se inicia un combate contra el mismo siempre que se tengan los puntos de valentía necesarios, al hacerlo se gastan los puntos. ● Al pulsar sobre el retrato del jugador se abre el menú. ● Al pulsar sobre el ícono de filtro se abre panel de filtros. ● Al pulsar en un ícono de la ventana de filtros dejará de mostrarse ese elemento del mapa si se estaba mostrando y pasará a mostrarse en caso contrario.
Menú	<ul style="list-style-type: none"> ● Se pueden seleccionar los objetos de la lista al pulsar sobre ellos, al hacerlo aparece una descripción del mismo y en caso de poder usarse o equiparse aparece su botón correspondiente. Aparece un botón de eliminar siempre que se seleccione un objeto. ● Al pulsar sobre el botón de equipar se sustituye el objeto equipado por el seleccionado y el que anteriormente estaba equipado se devuelve al inventario. ● Al pulsar sobre el botón de usar realiza la acción del objeto y se elimina del inventario. ● Al pulsar sobre el botón de eliminar se borra el objeto del inventario. ● Al pulsar sobre la pestaña de misiones se cambia a ese apartado del menú. ● En la pestaña de misiones al pulsar sobre una de ellas se muestra su descripción. ● Al pulsar sobre la pestaña de habilidades se cambia a ese apartado del menú. ● En la pestaña de habilidades al pulsar sobre una de ellas se muestra su descripción. ● Al pulsar sobre la pestaña de inventario se cambia a ese apartado del menú. ● Al pulsar sobre el ícono con la “x” se vuelve a la escena de mapa del mundo.
Combate	<ul style="list-style-type: none"> ● Al pulsar sobre un enemigo se seleccionara para recibir la próxima acción siempre que ésta no deba realizarse sobre el jugador.

	<ul style="list-style-type: none"> ● Al pulsar sobre el botón de ataque se realiza la acción siempre que se haya seleccionado un enemigo y todos hayan elegido ya acción. ● Al pulsar sobre el botón de ataque se realiza la acción siempre que se haya seleccionado un enemigo y todos hayan elegido ya acción. ● Al pulsar sobre el botón de defender se realiza la acción siempre que todos hayan elegido ya acción. ● Al pulsar sobre el botón de habilidad se abre una lista con las habilidades. ● Al pulsar sobre una habilidad en la lista de habilidades se selecciona y se muestra una descripción. ● Al pulsar sobre el botón de usar en la lista de habilidades se realiza la acción de usar habilidad siempre que se haya seleccionado un enemigo, una habilidad, se tenga mana suficiente para usarla y todos hayan elegido ya acción. Al terminar la acción se restará al mana actual del jugador el coste de mana de la habilidad. ● Al pulsar sobre el botón de objeto se abre una lista con los objetos disponibles. ● Al pulsar sobre un objeto en la lista de objetos se selecciona y se muestra una descripción. ● Al pulsar sobre el botón de usar en la lista de objetos se realiza la acción de usar objeto siempre que se haya seleccionado un objeto y todos hayan elegido ya acción. Al terminar la acción se eliminará el objeto del inventario. ● Al pulsar sobre el botón de huir se realiza la acción siempre que todos hayan elegido ya acción. Si se consigue huir se cambia a la escena de mapa del mundo.
Tienda	<ul style="list-style-type: none"> ● Se pueden seleccionar los objetos de la lista al pulsar sobre ellos, al hacerlo aparece una descripción del mismo. ● Se puede pulsar sobre el ícono con el símbolo de sumar para añadir un objeto de ese tipo al carrito. El gasto actual de la compra se actualiza con el nuevo objeto. ● Se puede pulsar sobre el ícono con el símbolo de restar para quitar un objeto de ese tipo del carrito. El gasto actual de la compra se actualiza con el nuevo objeto. ● Al pulsar sobre comprar se añadirán los objetos del carrito al inventario siempre que se tenga suficiente dinero. El dinero se restará del jugador. ● Al pulsar sobre el ícono con la “x” se vuelve a la escena de mapa del mundo.
Gremio	<ul style="list-style-type: none"> ● Se pueden seleccionar las misiones de la lista al pulsar

	<p>sobre ellas, al hacerlo aparece una descripción de la misión. Si la misión está aceptada y completada aparece un botón de entregar, si no está aceptada aparece un botón de aceptar.</p> <ul style="list-style-type: none"> ● Al pulsar sobre el botón de aceptar se añade la misión a la lista de misiones del jugador. ● Al pulsar sobre el botón de completar se añade la experiencia y dinero indicados al jugador, puntos a la facción y se elimina esa misión de la lista de aceptadas por el jugador. ● Al pulsar sobre el botón de sanar se recuperan la vida y el mana del jugador por completo. ● Al pulsar sobre el ícono con la “x” se vuelve a la escena de mapa del mundo.
Chat	<ul style="list-style-type: none"> ● Al pulsar sobre el campo de texto puede escribirse un mensaje. ● Al pulsar sobre el botón de enviar se actualiza la conversación con el texto introducido. ● Al pulsar sobre el ícono con la “x” se vuelve a la escena de mapa del mundo.

2.4 Elementos de registro y progreso

Esta sección describe los elementos que registran el progreso del jugador. Los principales elementos son la experiencia , el nivel, los objetos y el dinero.

Tanto la experiencia como el dinero se ganan al derrotar a los enemigos y completar misiones. Al llegar a cierta cantidad de experiencia se sube de nivel y con el dinero podrán adquirirse nuevos objetos.

Los objetos se consiguen al comprarlos en las tiendas y permiten mejorar las estadísticas del jugador.

Por último al subir de nivel se aprenden nuevas habilidades y se aumentan las estadísticas. Los enemigos que aparecen se harán más fuertes pero también darán más experiencia y dinero, con este dinero pueden comprarse mejores objetos que aumentan aún más las estadísticas.

2.5 Elementos de Jugabilidad y experiencia del jugador

Describir brevemente utilizando las atributos y propiedades de la jugabilidad elementos que marcarán cómo será la experiencia del jugador. Citar ejemplos

2.6 Otros elementos

Describa cualquier otro elemento que no puede ser clasificado en cualquiera de las otras clasificaciones de elementos.

2.7 Lista de recursos activos

Esta sección contiene la lista de todos los activos de juego que se deben crear para terminar el juego.

3 Dinámica

Esta sección describe el flujo del juego. Historia, niveles , capítulos , rompecabezas, retos y las interfaces (hardware y software). Esta sección está directamente relacionada con la sección de la mecánica desde la dinámica se construyen a partir de los elementos en la mecánica.

3.1 Mundo de Juego. Universo Virtual

Esta sección describe el universo donde se desarrolla Geono RPG.

3.1.1 Detalles del Juego en temática

El juego transcurre en un mundo similar al nuestro donde existe la Capa de la Consciencia en la cual se graba todo lo que ocurre en el universo. Las concentraciones de sucesos de alto coste computacional como el pensamiento humano pueden superar la capacidad de una zona de la Capa de la Consciencia. Si la capacidad de una zona es superada se producen anomalías que pueden llegar a alterar la realidad.

3.1.2 Descripción de misiones, niveles o capítulos.

El jugador avanza por el mundo virtual recorriendo el mundo real pero el espacio en el que el juego funciona está restringido a la ciudad de Granada.

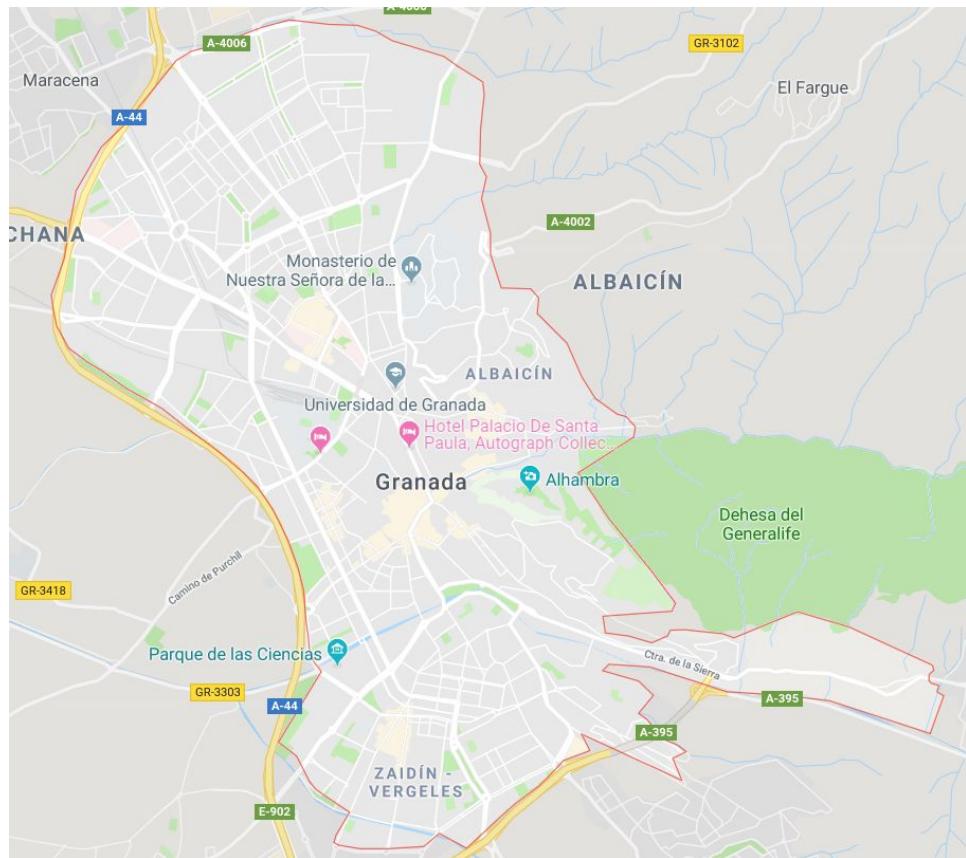


Figura 3.1: Mapa de Granada



Figura 3.2: Mapa del juego

3.1.3 Historia detallada

Los lugares con una alta concentración de personas provocan alteraciones en la Capa de la Consciencia al superar la capacidad de procesamiento de esa zona, esto ha provocado que los sitios con un alto nivel de pensamiento como las Universidades hayan provocado a lo largo de los años residuos que han desembocado en la alteración de la realidad y la aparición de monstruos hostiles.

Debido al debilitamiento de la capa los humanos también han empezado a ser capaces de alterar la realidad por sí mismos siendo capaces de hacer uso de magia.

La aparición de esta nueva hostilidad ha provocado que para poder combatirla se vuelvan a hacer uso de las armas del pasado. Para defenderse de estos nuevos enemigos las Universidades han evolucionado en instituciones aún más relevantes encargadas de entrenar a alumnos como combatientes capaces de enfrentarse a estos nuevos monstruos hostiles. Esto ha provocado una competencia entre las Escuelas de la Universidad que buscan tener más relevancia.

Las enseñanzas en las Universidades generan también un problema puesto que mientras más enseñanza se efectúa más se debilita la Capa de la Consciencia haciendo que los monstruos sean cada vez más poderosos, esto plantea un dilema puesto que si las Escuelas dejaran de enseñar la Capa de la Consciencia se iría reparando y los monstruos dejarán de aparecer, pero esto provocaría que la humanidad dejara de avanzar.

3.2 Misiones

En esta sección se van a describir las misiones que se pueden encontrar dentro del juego.

3.2.1 Objetivos

Los objetivos son derrotar un número de enemigos de un tipo determinado.

3.2.1.1 Primarios

Objetivos primarios están relacionados con el progreso del juego. ¿Qué objetivos se deben lograr para el avance en el juego o cuál es la condición de victoria u objetivo final necesario para ganar el juego?

3.2.1.2 Secundarios

Objetivos complementarios , pero no necesarias para avanzar en el juego.

3.2.2 Recompensas

Al completar misiones el jugador

3.2.2.1 *Implícitas*

Recompensas implícitas no impactan directamente en la capacidad de los jugadores, sino que están relacionados con la experiencia del jugador. Un ejemplo de recompensa implícita es rescatar a una princesa en un castillo y de cómo felicita a ella al personaje principal por su esfuerzo.

3.2.2.2 *Explícitas*

Recompensa explícita provoca un impacto directo en las capacidades de los jugadores. Algunos ejemplos de recompensa explícita son: una nueva arma , una vida extra o monedas de oro.

3.2.3 Desafíos

Desafíos retan a los jugadores a través del juego. Algunos ejemplos de retos son : una pelea, un rompecabezas, un salto por plataformas, un acertijo o una lucha del jefe.

3.2.4 Otros elementos de Misiones / Contenidos / capítulos

Agregar en esta sección otros elementos misiones / niveles / capítulos no incluidos en los apartados anteriores.

3.2.5 Flujo de las Misiones / Contenidos / capítulos

Describir el flujo de cada misión / nivel / capítulo del juego. Utilice los elementos descritos con anterioridad. Ilustrar el tamaño del área es, indicar su mapa, las instrucciones para superarlo y cómo se relaciona con la historia. Trate de cubrir todos los detalles que experimentará el jugador. Añadir conceptos de arte para todo el nivel y elementos representativos de este.

3.2.6 Zonas especiales

Las áreas disponibles son las siguientes:

- **Tiendas:** Se encuentran en puntos fijos dentro de la ciudad, en ellas se pueden comprar objetos con el dinero obtenido de los enfrentamientos y las misiones.
- **Gremios:** Se encuentran localizados en las facultades de la Universidad, en ellos se pueden consultar los puntos de las diferentes facciones, aceptar misiones y entregar las mismas una vez se han completado los objetivos que proponen.

3.2.67 Experiencia del jugador

Indicar mediante atributos y propiedades de la jugabilidad elementos concretos de la experiencia del jugador a desarrollar en cada uno de los niveles

3.3 Interfaz del juego

Existen diferentes pantallas con las que el jugador puede interaccionar:

- **Inicio de sesión:** Esta pantalla está compuesta por dos campos de entrada para el correo y la contraseña y dos botones, uno para iniciar sesión con los datos introducidos y otro para pasar a la pantalla de creación de cuenta. Si no se ha creado un personaje el botón de inicio de sesión lleva a la pantalla de creación de personaje.
- **Creación de cuenta:** Está compuesta por dos campos de entrada para el correo y la contraseña y un botón para crear la cuenta con los datos introducidos y acceder a la creación de personaje.
- **Creación de personaje:** Está compuesta por un campo de entrada para el nombre del personaje, dos secciones con botones para seleccionar la clase y fracción del personaje y un botón para crear el personaje con los datos introducidos.
- **Mapa del mundo:** En esta pantalla se muestra un mapa de la ciudad, el jugador avanza por el mismo moviéndose en la realidad. Sobre el mapa se muestra la vida y el maná actuales del jugador y un retrato del mismo mediante el cual se puede acceder al inventario. Sobre el mapa también aparecen elementos para acceder a las pantallas de tienda, gremio y combate.
- **Tienda:** Está compuesta por tres secciones, una con la lista de objetos, los cuales contienen dos botones para aumentar o disminuir la cantidad a comprar, otra sección que contiene la descripción del objeto seleccionado y una última que muestra el dinero del jugador y el dinero total a gastar con los objetos añadidos, por último existe un botón para comprar los objetos seleccionados.
- **Combate:** En una zona de la pantalla se muestran los participantes del combate, se puede cambiar el objetivo al que afectan las habilidades pulsando sobre el mismo. En otra zona se muestran varios botones para atacar, defender, usar una habilidad, usar un objeto y huir. En caso de pulsar los botones de habilidad u objeto la zona cambia por otra con una lista donde podemos seleccionar una habilidad u objeto a usar y dos botones, uno para usar lo seleccionado y otro para volver atrás.
- **Gremio:** Está compuesto por una sección donde se muestran los puntos de la facción y otra con una lista de misiones. Las misiones que no hayan sido aceptadas aparecen con un botón para aceptarlas, las que ya han sido aceptadas pero no están completadas no aparecen y las que ya han sido completadas aparecen con un botón para poder entregarlas.
- **Inventario:** Está compuesto por una sección que muestra la información del personaje, otra que incluye la lista de objetos y una última que muestra la descripción del objeto seleccionado. Existen dos botones, uno para usar o equipar el objeto seleccionado y otro para eliminarlo del inventario. Por último

es posible cambiar entre la lista de misiones y habilidades del jugador pulsando en las pestañas correspondientes.

- **Misiones del jugador:** Muestra una lista con las misiones y el estado de las mismas. Es posible cambiar entre el inventario y la lista de habilidades del jugador pulsando en las pestañas correspondientes.
- **Habilidades del jugador:** Muestra una lista con las habilidades y una descripción de las mismas. Es posible cambiar entre el inventario y la lista de misiones del jugador pulsando en las pestañas correspondientes.

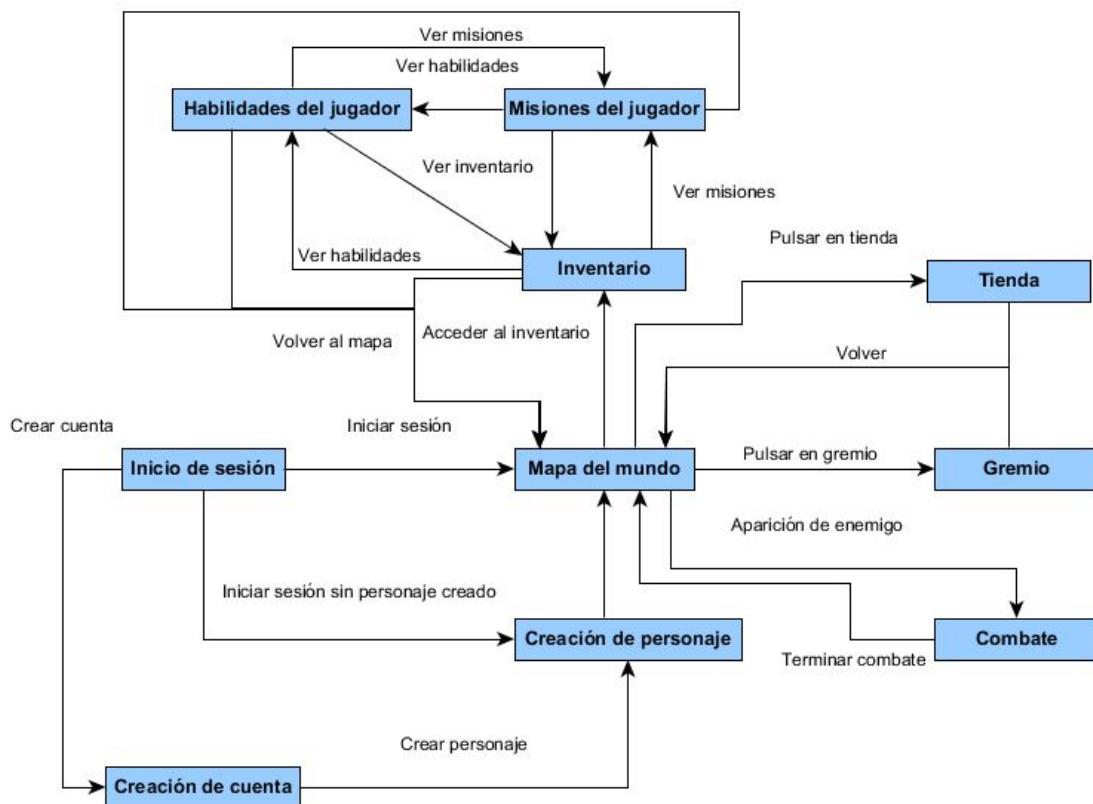


Figura 3.2: Diagrama de flujo

3.4 Controles de la interfaz

El jugador puede controlar el juego pulsando en los diferentes botones que aparecen en la pantalla o en los iconos en el caso de los elementos que aparecen en el mapa. En combate además el jugador puede pulsar sobre el enemigo para seleccionarlo como objetivo.

3.5 Equilibrio Juego

La dificultad del juego puede ser equilibrada mediante los cambios en las estadísticas base de los enemigos, estos adaptan sus estadísticas dependiendo del nivel del jugador por lo que podemos hacer que un enemigo sea mas o menos complicado modificando

las estadísticas base del mismo.

El número de enemigos en combate también puede ser modificado pudiendo por lo tanto aumentar o disminuir la dificultad de los encuentros.

4 Estética y Arte

Esta sección detalle lo que el jugador ve y oye . En esta sección se puede ampliar en caso de juegos de realidad aumentada, como la inclusión de olores o tacto. Esta parte es forma parte de la “Biblia de Arte y Conceptos”

4.1 Elementos básicos del juego

Describa todos los aspectos visuales/sonoros de los elementos del juego core. Incluir arte conceptual

4.2 Elementos del mundo

Describa todos los aspectos visuales/sonoros de los elementos del mundo del juego. Incluir arte conceptual

4.3 Elementos de registro de progreso

Describa todos los aspectos visuales/sonoros de los elementos de registro del juego. Incluir arte conceptual

4.4 Otros elementos visuales

Describa todos los aspectos visuales/sonoros incluidos en la otra sección elementos. Incluir arte conceptual

4.5 Mundo del Juego

Describa todo el aspecto visual/sonoro de mundo del juego. Incluir arte conceptual

4.6 Misiones / niveles / capítulos

Describa todos los aspectos visuales/sonoros de las misiones , los niveles y capítulos. Incluir arte conceptual

4.7 Áreas especiales

Describa todos los aspectos visuales/sonoros de las zonas especiales.

4.8 Interfaz del juego

Describa todos los aspectos visuales/sonoros de la interfaz del juego.

5 Experiencia

Esta sección explora la cuestión relevante desde el diseño del juego y la calidad que se espera de este problema. Esto ayudará a establecer lo que es importante para la prueba y qué esperar de la prueba de los prototipos y las futuras fases de desarrollo de juegos. Analizar aspectos claves de la experiencia del usuario usando facetas y en cada una objetivos claros usando cuestiones como la jugabilidad, usabilidad o accesibilidad.

5.1 Jugabilidad Intrínseca

Señale lo que es importante acerca de la dinámica y núcleo del juego : la dificultad de los desafíos, recompensas adecuadas, objetivos claros. Y describir lo que se espera de estos aspectos importantes. Citar ejemplos

5.2 Jugabilidad Mecánica

Señale lo que es importante acerca de los aspectos técnicos del juego: la calidad de las cinemáticas, la calidad del sonido o de las reacciones de colisión. Y describir lo que se espera de estos aspectos importantes. Citar ejemplos.

5.3 Jugabilidad Interactiva

Señale lo que es importante acerca de los aspectos de la interfaz del juego : las habilidades o dificultades de los movimientos, auto - pausa o la gestión de inventarios. Y describir lo que se espera de estos aspectos importantes. Citar ejemplos.

5.4 Jugabilidad Artística

Señale lo que es importante acerca de los aspectos estéticos del juego : Los enemigos “dan miedo”, música inspirar acciones épicas o historia divertida, si se corresponde con un mundo virtual o real conocido. Y describir lo que se espera de estos aspectos importantes. Citar ejemplos

5.5 Jugabilidad Intrapersonal

Señale lo que es importante acerca de la experiencia del jugador : jugador se siente ansioso y nervioso casi todo el tiempo, el jugador siente empatía por el personaje principal de la tragedia o el reproductor se vuelve adicto porque quiere conocer el nuevo poder que ser desbloqueados. Y describir lo que se espera de estos aspectos importantes. Citar ejemplos.

5.6 Jugabilidad Interpersonal

Señale lo que es importante acerca de la experiencia de los jugadores en el modo multijugador: jugador siente que está luchando contra otro jugador en la igualdad de terreno o reproductor sensación que tienen más posibilidades de lograr algo en el juego si se colabora con otro jugador. Y describir lo que se espera de estos aspectos importantes. Citar ejemplos

6 Limitaciones y supuestos

Esta sección describe la restricción de que el juego va a tener y debe ser tomada en consideración a la hora de tomar las decisiones de diseño de juegos.

6.1 Limitaciones técnicas

Estas limitaciones se derivan generalmente de la plataforma o las opciones de motor de juego. Describa lo que las limitaciones técnicas que el juego tiene. Algunos ejemplos de restricciones técnicas pueden ser: no hay memoria permanente o plataforma sólo tienen pantalla multi-touch.

6.2 Restricciones comerciales

Describir qué limitaciones negocio que el juego tiene. Algunos ejemplos de restricciones del negocio pueden ser: la clasificación del juego debe ser para toda la familia o juego deben publicando antes de 20 de diciembre.

7 Información del documento

7.1 Definición, acrónimos y abreviaturas .

Definir todos los conceptos, acrónimos y abreviaturas necesarios para la comprensión del documento.

7.2 Referencias.

Todos los documentos que se mencionan en el presente GDD y especificar dónde se

pueden encontrar.

7.3 Ficheros adjuntos

Añadir ficheros adjuntos relevantes al videojuego.

