Escuela Politécnica Superior de Jaén

Grado en Ingeniería Informática

UNIVERSIDAD DE JAÉN

Escuela Politécnica Superior

Trabajo Fin de Grado

Desarrollo de un Videojuego

Flame Knights Chronicles

Alumno: Alejandro Gómez López

Tutor: Prof. D. Juan Roberto Jiménez Pérez

Dpto: Departamento de Informática

Junio, 2016

Índice

1. INTRODUCCIÓN 5
2. MOTORES DE VIDEOJUEGOS 7
2.1. Unity 3D 8
2.1.1. Ventajas 9
2.1.2. Desventajas 9
2.2. Unreal Engine 4 10
2.3. CryEngine 3.8 12
2.4. Eligiendo un Motor 13
3. ESTILO 15
3.1. Género 15
3.2. Cámara 16
3.3. Gráficos 16
3.4. Predecesores 17
4. PLATAFORMA 21
5. PROPÓSITO Y AUDIENCIA 23
5.1. ¿Qué es PEGI? 23
5.2. PEGI 12 23
6. HISTORIA 25
6.1. Prólogo 25
6.2. Línea Argumental 26
7. MECÁNICAS DE JUEGO Y CARACTERÍSTICAS 29
7.1. Objetivos 29
7.2. Recompensas del jugador 29
7.3. Economía 30
7.4. Criterios de Éxito 30
7.5. Criterios de Fracaso 31

7.6. Controles básicos 31
7.7. Habilidades 31
7.8. Estadísticas y Sistema de puntos de Característica 33
7.9. Experiencia y Nivel 34
7.10. Objetos 35
7.11. Enemigos 36
7.11.1. Mecánicas de los Enemigos 36
7.11.2. Enemigos Comunes 37
7.11.3. Enemigos Élite (Jefes) 41
7.12. Funcionamiento 44
7.13. Flujo de la Experiencia de Juego 44
8. INTERFAZ GRÁFICA 45
8.1. La Interfaz Principal (GUI) 46
8.2. El Minimapa 47
8.3. Menús y Navegación 48
9. EFECTOS VISUALES 49
9.1. Habilidades Especiales del personaje 49
9.2. Proyectiles de los enemigos a distancia 51
9.3. Sangrado de los enemigos y el personaje principal 53
9.4. Efecto “Glow” (Brillo) en los enemigos 54
9.5. Efecto de relleno y vaciado de barras 54
9.6. Textos flotantes 55
9.7. Subida de nivel 56
9.8. Auras de los Jefes 57
10. EFECTOS DE SONIDO 61
11. DEFINICIÓN DEL ÁMBITO DEL PROYECTO 63
11.1. Alcance del proyecto (Extracción de Requisitos) 63

11.2. Estructura de Desglose de Requisitos 68
11.3. Ciclo de Vida de la Gestión del Proyecto 69
11.4. Declaración General Del Proyecto 70
12. PLANIFICACIÓN DEL PROYECTO 71
12.1. Estructura de Desglose del Trabajo 71
12.2. Estimación de la Duración 77
12.3. Estimación de Recursos 78
12.4. Estimación de Costes 80
12.5. Diagrama de Gantt 82
13. EJECUCIÓN DEL PROYECTO 87
13.1. Iteración 1 (del 1-02-2016 al 14-02-2016) 87
13.2. Iteración 2 (del 15-02-2016 al 28-02-2016) 99
13.3. Iteración 3 (del 29-02-2016 al 13-03-2016) 114
13.4. Iteración 4 (del 14-03-2016 al 27-03-2016) 122
13.5. Iteración 5 (del 28-03-2016 al 10-04-2016) 129
13.6. Iteración 6 (del 11-04-2016 al 24-04-2016) 144
13.7. Seguimiento y Control del Proyecto 167
14. FINALIZACIÓN DEL PROYECTO 173
14.1. Informe para el Equipo de Desarrollo 173
15. DIAGRAMAS REPRESENTATIVOS 177
15.1. Diagrama de Componentes: Jugador 177
15.2. Diagrama de Componentes: Enemigo cuerpo a cuerpo 178
15.3. Diagrama de Componentes: Enemigo a distancia 179
15.4. Diagrama de Componentes: Jefe 180
15.5. Diagrama de Componentes: Canvas 182
15.6. Diagrama de Componentes: Mapa/Misión 183
15.7. Diagrama Máquina de Estados: Flujo del Juego 184

15.8. Diagrama de Clases: Interfaz y Patrón Arquitectónico MVC 184
15.9. Diagrama Máquina de Estados: Enemigos 185
15.10. Diagrama de Secuencia: Combate Jugador – Enemigo 185
16. PATRONES USADOS 187
17. CONCLUSIONES: NOTAS DEL AUTOR 189
18. POR HACER (FUTURO) 191
19. BIBLIOGRAFÍA 193
20. FUENTES DE INFORMACIÓN USADAS 195
21. ENLACE DE DESCARGA 197
22. AGRADECIMIENTOS 197
23. ÍNDICE DE ILUSTRACIONES 199
24. ÍNDICE DE TABLAS 205
25. ANEXO I: GLOSARIO DE TÉRMINOS 207

1. INTRODUCCIÓN

La industria del videojuego mueve millones de euros a diario. Este hecho propicia la aparición de estudios que buscan un nicho de mercado en un sector emergente como este. Hoy en día, los videojuegos acompañan a las personas a lo largo de toda su vida. El objetivo de los videojuegos puede ser muy variado, desde simples mini-juegos que buscan “matar” el tiempo a simuladores de vuelo usados por la NASA para entrenar a sus pilotos.

Un videojuego es un producto software complejo, y como tal, su éxito o fracaso está estrechamente relacionado con la correcta aplicación de técnicas propias de la ingeniería del software para su definición, gestión, control y desarrollo. Los proyectos de esta naturaleza están compuestos por múltiples elementos muy dispares como el diseño gráfico, la animación, un guión de la historia, la inteligencia artificial, el diseño de mecánicas, el modelado de escenarios, los efectos de sonido, la música… Un proyecto software de estas características supone un intensivo trabajo de ingeniería, con equipos de desarrollo enormes. Como dato representativo, la firma Konami formó un equipo de más de 200 personas para desarrollar la afamada saga de videojuegos “Metal Gear Solid” (Hideo Kojima, 2016).

Estos proyectos poseen unas peculiaridades propias, tales como la alta interacción entre el usuario y la máquina, convirtiendo la pantalla y el mundo virtual en un conjunto de elementos interactivos para el jugador. También cabe destacar que son proyectos en los que la ingeniería y el arte (música, diseño 3D y 2D, efectos especiales…) van de la mano durante todo el proceso de desarrollo y la vida del producto.

Para afrontar este reto, se usarán como base los conocimientos adquiridos durante el periodo de aprendizaje en asignaturas como Desarrollo Ágil, Gestión de Proyectos Software y Desarrollo de Videojuegos. Seguir una fuerte metodología desde el comienzo para definir, planificar, gestionar, desarrollar y controlar el proyecto propiciarán el éxito del mismo. Estas tareas serán realizadas minuciosamente debido a su gran importancia.

Flame Knights Chronicles, el título de este videojuego, nos transportará a un oscuro y fantástico mundo medieval donde las fuerzas del bien y el mal luchan por su supremacía. El jugador, tomando el control del protagonista principal, deberá combatir contra criaturas malignas y legiones infernales para evitar que un poderoso brujo no-muerto esclavice a la humanidad.

El videojuego posee un gran componente de estrategia, que combinado con la frescura y dinamismo de la acción en tiempo real de los innumerables combates da una experiencia de juego adictiva, y ante todo divertida. Esta atractiva combinación de estrategia y acción crea un género de juego denominado “Hack ‘n Slash”.

Una parte importante de los esfuerzos serán dirigidos a obtener un prototipo que resulte atractivo y amigable al usuario final. Para conseguir este fin se dispondrá de un conjunto de usuarios que probará el producto en sus diferentes fases de desarrollo. Estos usuarios darán una valiosa visión al equipo de desarrollo acerca de la interacción del usuario final con el juego y sus opiniones personales.

El sector de los videojuegos y la informática gráfica en general tiene un conjunto de términos en inglés muy específicos que serán definidos en el capítulo *25 ANEXO I: GLOSARIO DE TÉRMINOS*. En este capítulo también se incluirán vocablos que puedan no resultar familiares al lector.

Este documento seguirá la siguiente estructura. En primer lugar se presenta una comparativa y análisis de los motores de videojuegos más representativos (capítulo *2*). A continuación una definición de las características más importantes del videojuego, como por ejemplo los enemigos y los objetivos del juego (capítulos del *3* al *10*). Después todo el proceso de planificación y desarrollo (capítulos del *11* al *14*), una colección de los diagramas más representativos (capítulo *15*), los patrones de diseño y arquitectónicos usados (capítulo *16*), las conclusiones del autor (capítulo *17*), las funcionalidades y características que han quedado fuera del proyecto (capítulo *18*), la bibliografía y las fuentes de información consultadas (capítulos *19* y *20*), el enlace de descarga (capítulo *21*), una sección de agradecimientos (capítulos *22*), los índices de ilustraciones y tablas (capítulos *23* y *24*) y un glosario de términos como anexo (capítulo *25*).

2. MOTORES DE VIDEOJUEGOS

Se define como motor gráfico al framework de software diseñado para crear y desarrollar videojuegos. Los desarrolladores pueden crear videojuegos para consolas, dispositivos móviles u ordenadores usando este software.

La funcionalidad básica de un motor es proveer al videojuego de un motor de renderizado para los gráficos 2D y 3D, motor físico o detector de colisiones, sonidos, scripting, animación, inteligencia artificial, redes, streaming, administración de memoria y un escenario gráfico.

Es indiscutible que la escena de videojuegos "indie" ha crecido de forma exponencial durante los últimos años. Cada vez son más las personas que se animan a crear su propio videojuego usando las infinitas herramientas y ayudas que se pueden encontrar en internet.

Ilustración 1 Motores Populares de Videojuegos

Lo más increíble es que motores de una altísima calidad como Unreal Engine 4 o CryEngine 3 estén actualmente al alcance de todo el mundo. Tenemos infinidad de opciones a la hora de elegir un motor gráfico. Ejemplos representativos son: Torque, CryEngine, Unity, Unreal Engine, Source Engine, Cocos2D-x, Shiva, Game Maker…

A continuación se analizarán tres de los motores más populares para decidir cuál se usará en este proyecto.

2.1. Unity 3D

Ilustración 2 Logo Unity 3D

Unity 3D ofrece una amplia gama de características y su interfaz es bastante fácil de entender. Se caracteriza por su **gran integración multiplataforma**, es decir, los juegos se pueden trasladar de forma rápida y fácilmente en Android, iOS, Windows Phone 8 y BlackBerry, por lo que es un gran motor de juego para el desarrollo de juegos para móviles. También tiene la capacidad de desarrollo para Playstation 3, Xbox 360, Wii U y navegadores web.

El motor de juego es **compatible con las principales aplicaciones 3D** como 3ds Max**,** Maya**,** Softimage**,** Cinema 4D**,** Blender y más, lo que significa que no hay restricciones reales al tipo de formatos de archivo que soporta. Implementa sprites de apoyo y la física en 2D, por lo que es un gran motor de juego que se utiliza para el desarrollo de juegos 2D (yeeply.com, 2014).

Unity **no posee características reales de modelado** o construcción más allá de unas pocas formas primitivas, de manera que los equipos de desarrollo deberán apoyarse en aplicaciones de terceros. Sin embargo, cuenta con una tienda de Assets (**Asset Store**) desde la que se puede descargar y usar infinidad de recursos gratuitamente o por poco dinero.

Actualmente la **versión Personal** de Unity 3D es gratuita con todas sus características disponibles excepto desarrollo para consolas y otros servicios adicionales como Unity Cloud Build (herramienta para la integración automática de cambios en red). La pega es que ,si se compila un juego en esta versión, tendremos una pantalla al inicio de la ejecución del videojuego con el logo de Unity que no se podrá eliminar. Se puede usar la versión Personal para producción si los ingresos brutos al año son menores de 100.000 $.

Existe una **versión Profesional** (desde 75$ al mes) la cual posee todas las características y servicios disponibles.

Unity 3D Engine es **perfecto para equipos pequeños** y el desarrollo para plataformas móviles. La curva de aprendizaje de Unity es la más fácil de los tres motores pero su motor gráfico es el menos potente (*Véase Tabla 1 Comparativa Motores*) . Es un motor de juegos de **propósito general**.

2.1.1. Ventajas

- Posee una **amplia documentación** y multitud de videos explicativos. Al llevar más tiempo accesible económicamente a muchos usuarios, la cantidad de información sobre scripting, plugins desarrollados por terceros y recursos es enorme.
- A la hora de implementar los Scripts, se puede elegir entre dos lenguajes de programación: **C#** y **JavaScript**. Anteriormente también se podía programar en **Boo**.
- Está **basado en componentes**, lo que invita a la reutilización y estructuración de las entidades usadas.
- Los requisitos técnicos para trabajar con este motor son medios-bajos.
- La **Asset Store** ofrece infinidad de soluciones ya creadas, modelos, animaciones… Dispone de muchos componentes gratuitos.
- Es ideal para la **creación prototipos**.
- El IDE que nos ofrece por defecto (**Monodevelop**) es muy fácil de usar. Permite utilizar otros IDE como Visual Studio con relativa facilidad.

2.1.2. Desventajas

- No está muy orientado a ser usado por diseñadores o modeladores.
- Su versatilidad para el desarrollo en diferentes plataformas es activada con **licencias de pago** que pueden elevar bastante el coste de producción.

2.2. Unreal Engine 4

Ilustración 3 Logo Unreal Engine

UE4 tiene algunas **capacidades gráficas increíbles**, incluyendo características como capacidades avanzadas de iluminación dinámica y un nuevo sistema de partículas que puede manejar hasta un millón de partículas en una escena a la vez. Unreal Engine ofrece varias herramientas adicionales de **gran ayuda para diseñadores** y artistas. La facilidad de uso del UE4 hace que sea mucho más atractivo para los nuevos desarrolladores de juegos.

El motor Unreal fué originalmente **desarrollado principalmente para los FPS** (First Person Shooter), y el primer juego que fue diseñado, como DEMO fue el aclamado Unreal, pero desde entonces ha sido utilizado para una gran variedad de juegos y géneros como los juegos de rol.

El lenguaje usado para scripting es **C++**, un lenguaje muy familiar para la mayoría de los desarrolladores.

Ofrece un sistema llamado **Blueprint**, el cual puede usarse para “programar” sin tener que escribir una sola línea de código. Los blueprints y el código C++ están pensados para complementarse. Todos los blueprints pueden exportarse y se transformarán en código C++ que podremos modificar carácter a carácter.

Cada mes, **Epic Games** lanza nuevas versiones de UE4 con suculentas mejoras. A nivel gráfico y estético el engine es inmejorable, y valga la redundancia, además mejora mes a mes. La **gestión de partículas** es la mejor del momento, ahora hacer superficies que reflejen en tiempo real no supone toparse con múltiples dificultades, gracias al **nuevo sistema de raycasting**, la gestión de transparencias ya no tiene un orden arbitrario.

UE4 viene con su propio gestor de menús 2D y 3D. Cuenta con un módulo (**Papers 2D**) increíblemente potente para gestionar juegos en 2D dimensiones de cualquier tipo: profundidad, colisiones 2D, físicas restringidas en Z, gestión de orden de elementos y menús.

Está más **orientado a equipos grandes** y el desarrollo de juegos de gran calidad (**juegos triple A**). Su motor gráfico es el más potente de la comparativa. Al igual que Unity, está **basado en componentes**.

Cabe señalar que Unreal Engine 4 está disponible **gratuitamente en su totalidad**, pero a partir de obtener unas **ganancias de 3.000 $** con un producto desarrollado sobre este motor se deberá pagar el **5% de los beneficios brutos** posteriores (royalties).

Unreal Engine ha sabido adaptarse a las demandas del sector mediante una estrategia agresiva y arriesgada de desarrollo abierto y esto ha hecho que el motor coja una carrerilla y una ventaja respecto a sus competidores que ahora lo tienen más difícil que nunca (zehngames.com, 2014).

2.3. CryEngine 3.8

Ilustración 4 Logo CryEngine

CryEngine es un motor muy potente diseñado por la empresa de desarrollo de Crytek que se introdujo en el primer juego de la saga Far Cry. Está diseñado para ser utilizado en plataformas móviles, PC y consolas, incluyendo PlayStation 4 y Xbox One.

Las capacidades gráficas de CryEngine están a la par con el Unreal Engine 4, con una **iluminación fantástica**, la **física realista**, **sistemas avanzados de animación** y mucho más. Similar a UE4, CryEngine tiene características intuitivas y potentes de diseño de niveles en el motor del juego (ozom.cl, 2014).

CryEngine es un motor de juego muy potente, no obstante su **curva de aprendizaje** es **un poco complicada** (*Véase Tabla 1 Comparativa Motores de Videojuegos*). Para empezar a utilizar el motor de juego de manera productiva, y puede ser más difícil de entender si no se tiene ninguna experiencia con motores de juegos.

Es especialmente sencilla la creación de escenarios exteriores con terrenos de gran amplitud. La **documentación** existente es **extensa** y hay foros oficiales para desarrolladores. Está diseñado para que equipos multidisciplinares trabajen conjuntamente sobre un mismo proyecto.

2.4. Eligiendo un Motor

Todos estos motores de juego son una gran opción para el proceso de desarrollo. Mientras que la Unity es ideal para juegos 2D y 3D móviles, Unreal Engine 4 te da la capacidad de crear juegos con gráficos fotorrealistas o simples scrollers laterales 2D y CryENGINE tiene unas capacidades gráficas increíbles para plataformas de nueva generación.

Ordenando los tres motores por dificultad de aprendizaje (de más fácil a más difícil), se tiene el siguiente orden. Unity, Unreal Engine y CryEngine (*Véase Tabla 1 Comparativa Motores*). CryEngine y Unreal Engine están orientados a **equipos grandes** y **multidisciplinares**, y al desarrollo de proyectos **gráficamente superiores** (los llamados juegos **Triple A**). Además, estos dos motores poseen muchas **formas de optimización y eficiencia** que son transparentes para el desarrollador y que mejoran mucho el producto final.

Por otra parte, Unity nos ofrece un **motor de videojuegos** bastante **completo**, **ligero** y que permite a desarrolladores con poca experiencia tomar contacto con este mundo de una manera **fácil** y **amigable**.

A continuación se muestra la tabla resumen que se ha creado para elegir el motor de juegos que más ventajas y herramientas aporte al proyecto.

Comparativa Motores de Videojuegos	Unreal Engine 4	CryEngine 3	Unity 3D
Curva de Aprendizaje	Media	Difícil	Fácil
Calidad Gráfica	Muy Alta	Alta	Media
Características de modelado 3D y construcción de escenas avanzada	Sí	Sí	No
Apto para equipos pequeños	No	No	Sí
Basado en componentes	Sí	No	Sí
Calidad del Motor de físicas	Muy Bueno	Excelente	Bueno
Gratis al inicio del proyecto	Sí	Sólo licencias para estudiantes	Sí
Gran biblioteca de recursos	No	No	Sí
Comunidad extensa	en Aumento	No	Sí
Documentación propia	Excelente	Buena	Excelente
Tutoriales de terceros	Sí	No	Sí, muchos

Tabla 1 Comparativa Motores de Videojuegos

Como opción para realizar este proyecto se ha optado por usar **Unity, versión 4.6.2**. Las razones son las siguientes. El tiempo para la ejecución es bastante limitado. Su **curva de aprendizaje** es la más **fácil**. Ell **equipo de desarrollo** es **extra-pequeño** (1 persona). Brinda **infinidad recursos a nuestra disposición**. El **sistema basado en componentes** ayuda a organizarse mejor a la hora de desarrollar y a reutilizar partes comunes. Está respaldado por una comunidad enorme. Existe muchísima información y documentación para su uso.

3. ESTILO

En este capítulo se definen tres de los rasgos que definen y dan forma al estilo y a la manera en la que el usuario percibirá el juego. Como añadido, se termina este capítulo con una sección en la que se citarán los predecesores más representativos del estilo en cuestión según la opinión del autor.

El estilo determina gran parte de la interactuación del usuario con el videojuego. También define elementos clave, como el tipo y posición de la cámara y el detalle gráfico.

3.1. Género

El género **RPG** (Role Playing Games), sumerge al jugador en un mundo ficticio, siguiendo el transcurso de una historia y tomando el control de un héroe (el cual puede personalizar completamente). Esta historia va proporcionando retos que el jugador deberá superar para continuar la aventura. Las ambientaciones de un juego RPG son tremendamente amplias (fantasía medieval, épocas futuristas, actualidad,…).

El género **"Hack and Slash"** se basa en combates rápidos, normalmente con armas cuerpo a cuerpo y contra hordas de enemigos. Ofrecen combates en tiempo real, a diferencia del género RPG. Se suele combinar con otros géneros como terror, aventuras,…

La mezcla de estos géneros dió lugar a otro llamado **Action RPG** (Action-Role Playing Games), adoptando éste las características más importantes de los géneros anteriormente mencionados.

Es un género apasionante que mezcla la profundidad en la trama que caracteriza a los videojuegos RPG y la acción desenfrenada de los de tipo "Hack and Slash".

3.2. Cámara

Se conoce **perspectiva Top-Down** (también llamada vista ojo de ave, vista de pájaro, vista elevada o vista de helicóptero) a un ángulo de cámara utilizado en los videojuegos que muestra al jugador y al área circundante desde arriba.

Los videojuegos que son referencias en el género Action RPG suelen usar esta perspectiva pero con una **proyección cónica** (similar nuestra vista), tal como la **proyección en perspectiva**, la cual da una mayor sensación de profundidad. En el proyecto se usará una cámara tipo Top-Down.

Las vistas top-down están implementadas típicamente usando proyecciones ortográficas o isométricas. La isométrica se hizo más familiar por los primeros juegos de la saga Sims (Electronic Arts) y varios juegos de estrategia al estilo de “Command and Conquer” (Westwood Studios). Esta proyección esencialmente hace que las líneas paralelas continúen al infinito. Hoy en día, las proyecciones isométricas se usan típicamente para diseñar juegos de estética retro.

3.3. Gráficos

La mayoría de videojuegos de este estilo anteponen la importancia de unas mecánicas atractivas y divertidas de usar a una calidad gráfica inmejorable, siempre dentro de lo aceptable. Se debe tener en cuenta que los personajes no se ven lo suficientemente cerca como para apreciar detalles demasiado sutiles.

No obstante se ha intentado cuidar al máximo el detalle en las animaciones y habilidades especiales del personaje, teniendo en cuenta que no se dispone de personal especializado en diseño gráfico y animación. Los recursos gráficos que se usarán serán gratuitos o generados por el propio autor.

3.4. Predecesores

Para poner al lector en situación, a continuación se mencionan los videojuegos que podrían ser los más representativos del género. Estos títulos han sido jugados y seguidos de cerca por el autor, aportando varias ideas tenidas en cuenta en la fase de diseño de este proyecto.

3.4.1. Path of Exile

Ilustración 5 Logo Path Of Exile

Path of Exile es un ARPG ambientado en un mundo de fantasía oscura. Lo desarrolla la compañía independiente neozelandesa Grinding Gear Games y se puede descargar y jugar gratis. El proyecto se mantiene económicamente gracias a "micropagos éticos".

El 23 de enero de 2013 se publicó una versión Beta abierta. En marzo de 2013 el juego había alcanzado los dos millones de subscriptores. El juego abandonó la fase Beta y se publicó finalmente tanto en Steam como en su propia web el 23 de octubre de 2013. Cabe mencionar que el juego es totalmente online.

Ilustración 7 Cámara en Path of Exile

Ilustración 6 Combate en Path of Exile

3.4.2. Torchlight 2

Ilustración 8 Logo Torchlight 2

Torchlight 2 es un ARPG desarrollado por Runic Games y distribuído por Microsoft Games en Septiembre de 2012.

Como su predecesor, las mazmorras son generadas aleatoriamente, además presenta más clases disponibles para el jugador (arquetipos) y un modo historia más largo. Las únicas críticas negativas que ha sufrido, es debido a la falta de un sistema multijugador online. Los jugadores pueden crear contenido (misiones, mazmorras,…) y compartirlo a través de Steam.

Ilustración 10 Cámara y Ambientación Torchlight 2

Ilustración 9 Combate en Torchlight 2

3.4.3. Diablo 3

Diablo III es un A-RPG desarrollado por Blizzard Entertainment. Ésta es la continuación de Diablo II y la tercera parte de la serie. Su temática es de fantasía oscura y terrorífica. Su aparición fue anunciada el 28 de junio de 2008 en el Blizzard Entertainment Worldwide Invitational en París, Francia.

Ilustración 11 Logo Diablo 3

Blizzard anunció que el lanzamiento se realizaría el 15 de mayo de 2012. Fue uno de los lanzamientos más importantes de un videojuego en la historia, vendiendo una cifra de 3.5 millones de copias en 24 horas y 6.3 millones en una semana.

Ilustración 12 Combate multijugador online en Diablo 3

Ilustración 13 Gráficos y Ambientación en Diablo 3

4. PLATAFORMA

La plataforma a la que va dirigida este videojuego es Windows 64 bits (versión 7 y superiores). Las razones que han llevado a elegir esta plataforma de entre las posibles son:

- Existencia de **mouse**, para facilitar el control y la jugabilidad.
- Disponer del dispositivo en cuestión para realizar las pruebas.
- El **rendimiento gráfico** es **mucho mayor** que en otros dispositivos.
- Los **ordenadores personales**, concretamente con sistema operativo Windows, están **ampliamente extendidos** por todo el mundo.
- Posibilidad de conseguir **futura financiación y distribución** del juego por Steam, con su sistema **Greenlight**.
- **No se requieren licencias especiales** de Unity si desarrollamos el videojuego para Windows.

Se han barajado otras opciones como desarrollarlo para Xbox One o PlayStation 4, pero no se dispone de estos dispositivos para la realización de las pruebas.

5. PROPÓSITO Y AUDIENCIA

El propósito de este videojuego es entretener y divertir al usuario proporcionándole retos desafiantes pero alcanzables. Es un juego intensivo, que trata de sumergir al jugador en un mundo fantástico-medieval.

5.1. ¿Qué es PEGI?

Para definir la audiencia a la que va dirigida este producto debemos mencionar el sistema de clasificación se usa en Europa por todos los desarrolladores de videojuegos.

El sistema de clasificación por edades establecido por la Información Paneuropea sobre Juegos (PEGI) se estableció con el objeto de ayudar a los progenitores europeos a tomar decisiones informadas a la hora de adquirir juegos de ordenador. Se estrenó en la primavera de 2003 y sustituyó a una serie de sistemas nacionales de clasificación por edades englobándolos en un único sistema que se utiliza ya en la mayor parte de Europa.

El sistema está respaldado por los principales fabricantes de consolas, incluidos Sony, Microsoft y Nintendo, así como por editores y desarrolladores de juegos interactivos de toda Europa. El sistema de clasificación por edades fue desarrollado por la Federación de Software Interactivo de Europa (ISFE).

5.2. PEGI 12

En esta categoría pueden incluirse los videojuegos que muestren violencia de una naturaleza algo más gráfica hacia personajes de fantasía y/o violencia no gráfica hacia personajes de aspecto humano o hacia animales reconocibles, así como los videojuegos que muestren desnudos de naturaleza algo más gráfica. El lenguaje soez debe ser suave y no debe contener palabrotas sexuales (pegi.info, 2016).

La audiencia a la que va dirigido el videojuego de este proyecto son personas mayores de 12 años, ya que contiene situaciones violentas y lenguaje soez según la clasificación PEGI.

Estos serán los iconos que deberán ser visibles a la hora de adquirir el videojuego, ya sea en un portal web o físicamente en su respectiva caja.

Ilustración 16 Icono PEGI +12

Ilustración 15 Icono PEGI Uso de lenguaje soez

Ilustración 14 Icono PEGI Violencia

El primer icono muestra la edad máxima recomendada. El segundo y el tercero informan de que el videojuego contiene leguaje soez y situaciones de violencia.

6. HISTORIA

A continuación se presenta el prólogo del videojuego y una breve descripción de la línea argumental. Partiendo de este guión se han seleccionado y creado los enemigos y parte del decorado.

6.1. Prólogo

En el reino de Xanria se respira paz. Después del fin de las guerras del este, todos sus habitantes viven en armonía y tranquilidad. Un reino antiguo y lleno de secretos. Pero algo maligno está surgiendo en el bosque de Talmoth. Los habitantes de una aldea cercana han visto cosas inexplicables. Criaturas sacadas de las peores pesadillas acechan en ese bosque, y muchos de los que se aventuran a explorar no han sido vistos nunca más.

Este hecho ha llegado a oídos del rey, quien ha convocado de nuevo a los caballeros de la llama, un gremio de guerreros que usan la magia para defender el reino de las garras de la oscuridad y el caos. Esta sociedad secreta permanece oculta entre la gente, velando por la prosperidad y custodiando reliquias antiguas que poseen un poder casi inimaginable.

Si algo corrupto y oscuro está gestándose en el bosque, ellos son los únicos capaces de dar fin a tal amenaza contra la paz. Han enviado a uno de sus más jóvenes y prometedores integrantes a investigar los extraños sucesos. ¿Podrá este novato enfrentarse a los peligros que acechan en las sombras y completar su misión con éxito?

6.2. Línea Argumental

La línea argumental se divide en actos. Estos actos, se subdividen en misiones llevadas a cabo por el jugador. El transcurso de la historia será lineal y la narrativa guiará al personaje, no es un videojuego de toma de decisiones.

6.2.1. Acto I: No hay descanso para los malvados (No Rest For The Wicked)

El joven caballero llega al bosque oscuro de Talmoth, descubre que los muertos están resucitando para sembrar el caos. Después de informar a sus superiores es enviado a investigar unas ruinas en la pequeña isla de Gildor para buscar información sobre el ancestral y prohibido arte de la nigromancia. Al llegar a las ruinas ve que están fuertemente custodiadas por no-muertos. Decide entrar a las ruinas por un pasadizo secreto con la ayuda de Nissa, una ladrona de tesoros. El caballero logra recuperar un antiguo libro pero Nissa se lo roba en un descuido y ésta desaparece.

Al llegar a informar al gremio es informado de que un ejército de esqueletos ha arrasado la aldea de Mirem y es enviado a recopilar información. Mirem ha sido devastada con una fuerza brutal. Uno de los 4 supervivientes le señala al caballero el camino que siguió el ejército de no-muertos tras el ataque y decide seguir esa pista. El joven llega al pantano de Xenroc, logra dar con el ejército y acabar con su líder, una criatura pestilente llamada MudFist, después de interrogarlo.

6.2.2. Acto II: El Vórtice (The Vortex)

Antes de morir, MudFist alardeó del ejército que se estaba agrupando en el castillo de Risco Rojo. Este castillo es un montón de ruinas abandonadas hace cientos de años. Al explorar el castillo más detenidamente, nuestro protagonista se da cuenta de que por la noche hay pequeños terremotos, lo que hace que investigue los posibles pasadizos subterráneos ocultos.

Decide ir a una aldea cercana, donde tras hablar con el tabernero descubre que existe un pasadizo secreto muy antiguo que une el sótano de la taberna con el castillo.

Una vez recorrido el túnel, nuestro héroe llega a una gran construcción mágica donde ve como un archimago no-muerto abre un pequeño portal de donde salen horribles criaturas nunca vistas. El caballero es descubierto y logra escapar por los pelos.

Al volver al castillo con refuerzos todos los accesos al piso inferior han sido derrumbados sin posibilidad de bajar.

Unos días más tarde, el joven es convocado a la cuidad de Solrem por el consejo de magia del reino.

6.2.3. Acto III: La Prisión del Espacio-Tiempo (Space-Time Prison)

Una vez en Solrem, el gran mago Sterlin le cuenta al héroe que el archimago no-muerto se llama Abramelin. Abramelin fue el líder del consejo de magos y asesinó al bisabuelo del rey actual mientras reinaba hace 150 años. Fue condenado a muerte pero su cuerpo desapareció mágicamente al exhalar su último aliento.

El protagonista también descubre que Nissa, la ladrona escurridiza, es la hija de Sterlin y hace prometer a Sterlin que su gremio estará al día de los descubrimientos que se hagan del antiguo libro de nigromancia. Sterlin predice que los ejércitos de Abramelin atacarán la ciudad costera de Shale, pues en las profundidades de esta ciudad se oculta un objeto mágico de valor incalculable.

El héroe llega a Shale, habla con Theon, el alcalde de la ciudad, y se dobla la vigilancia. Theon le cuenta que, según la leyenda, la cuidad de Shale es en realidad una caja de contención para un artefacto capaz de abrir grandes portales a otros mundos de manera fija. Ante el peligro de ese artefacto y la imposibilidad de destruirlo sellaron el objeto bajo acero y roca para evitar su uso.

A los 2 días comienza el asedio a Shale. El mar escupe miles de muertos y horrores del inframundo. A duras penas pueden defender la cuidad, pero Abramelin llega a la plaza central, donde tiene unas palabras con nuestro héroe. Acto seguido invoca a unos golems y trolls para que ataquen al joven caballero mientras él comienza un ritual. Tras el ritual, consigue extraer el artefacto de su prisión y se desvanece llevándolo consigo.

6.2.4. Acto IV: El ciclo de la vida y la muerte (Cycle of Life and Death)

La ciudad de Shale vuelve a la normalidad poco a poco. Una noche, una gran columna de luz roja empieza iluminar todo el horizonte. Nissa busca al héroe en Shale, le comenta que su padre desapareció hace 2 días junto con el libro de nigromancia. Juntos deciden ir al lejano haz de luz.

El espeluznante flujo lumínico proviene de una gran torre hecha de huesos. El colosal edificio está rodeado por una muralla que encierra en su interior un gigantesco laberinto. Mientras intentan llegar al centro del laberinto se enfrentan a centenares de enemigos y trampas. Una vez llegan al centro del laberinto, deben enfrentarse a Nightmare, la mascota de Abramelin. Nightmare es un insecto aberrante, ágil y muy agresivo. Nissa queda mal herida en el enfrentamiento y el caballero debe subir solo a la torre.

Cada piso de la torre transporta al caballero a un escenario diferente a través del tiempo y el espacio. Al llegar al último piso, encuentra a Abramelin terminando de activar el artefacto. Abramelin raptó a Sterlin, el cual se encuentra allí, inconsciente.

Abramelin le explica al caballero que fue sentenciado injustamente. Mató al rey porque planeaba diezmar la población ante la crisis económica para seguir enriqueciéndose. El mismo pueblo que él salvó lo condenó a morir quemado en aceite. Abramelin usó un hechizo para viajar en el tiempo justo antes de morir, pero fue tan poderoso que consumió todo su cuerpo y su alma dejando solamente sus huesos, dejando atrás toda su humanidad.

Abramelin activa el portal sacrificando a Sterlin. Es tal la cantidad de magia que necesita que portal que atrapa a Abramelin y lo desintegra mientras él muere entre carcajadas. El portal se encuentra inestable. Parece SkullCrusher, un campeón infernal. El caballero debe matar a la bestia antes de que el portal se estabilice e intentar destruirlo. Consigue matar a SkullCrusher y comienza a absorber desesperadamente energía del portal. Es tanta la energía que el caballero tiene que soportar, que el portal lo atrapa y se crea una gran explosión que destruye la torre por completo.

7. MECÁNICAS DE JUEGO Y CARACTERÍSTICAS

En este capítulo se describen las características del juego. Estas caracteríticas pertenecen al producto final, por lo que el prototipo no presentará todo su conjunto. Algunos apartados simplemente presentarán resultados finales, puesto que en el capítulo *13 EJECUCIÓN DEL PROYECTO* se describe con más detalle todo el proceso de diseño e implementación.

7.1. Objetivos

El objetivo del usuario será completar las misiones que componen la historia linealmente. Cada misión tendrá unos objetivos principales a completar y podrá tener objetivos secundarios opcionales. Cuando se completa una misión, se debloquea la siguiente si no es ésta la misión final.

Las misiones podrán volverse a jugar pudiendo mejorar la puntuación. Un usuario puede intentar batir su propio record en una misión un número infinito de veces.

Los objetivos para completar las misiones son muy variados. Pueden ser eliminar a un jefe final (un enemigo muy superior en vida y daño a los comunes), limpiar una determinada zona de enemigos, escoltar a un NPC (non-player character, personaje no jugable) hacia un destino, recoger una serie de objetos por el mapa o llegar a una localización dentro de un límite de tiempo.

7.2. Recompensas del jugador

Las recompensas de las misiones serán experiencia y una puntuación por misión que dependerá de los objetivos cumplidos, los enemigos asesinados y las veces que el jugador ha muerto intentando completar dicha misión.

7.3. Economía

La economía dentro del juego recae sobre la cantidad de vida, de maná y de experiencia del personaje principal (avatar controlado por el usuario).

Es necesario mantener los puntos de vida en un valor positivo o el personaje principal morirá, reiniciándose así su posición en el mapa y teniendo que volver a dirigirse a la zona en la que se encontraba.

Los puntos de maná (o puntos de magia) son necesarios para que el usuario pueda usar las habilidades especiales del personaje principal. Si bien es cierto que no es un requisito totalmente imprescindible para completar el juego, el uso de estas habilidades mejoran la dinámica de juego y ayudan al usuario a superar retos de una forma más cómoda y estratégica.

Al eliminar a cualquier enemigo se obtiene un cantidad variable de puntos de experiencia. Esta experiencia es acumulada hasta que el personaje sube de nivel. Cada vez que el personaje sube de nivel se produce un crecimiento (personalizable por el usuario) en las características pasivas del personaje principal haciéndolo más fuerte, más rápido o más resistente. Siempre será beneficioso para el usuario conseguir la cantidad más alta de experiencia en cada mapa, pues los retos futuros serán más asequibles.

7.4. Criterios de Éxito

Los criterios de éxito están estrechamente relacionados con el alcance de los objetivos propuestos por las misiones (*7.1 Objetivos*). Para superar una misión exitosamente el usuario debe completarlos. Cumplir todos los objetivos de una misión (principales y opcionales) dará al usuario la nota más alta en esa misión.

7.5. Criterios de Fracaso

No existen criterios de fracaso como tal. La muerte del personaje principal simplemente penaliza en tiempo al usuario al devolverlo a la posición de inicio en el mapa.

7.6. Controles básicos

La forma de mover al personaje se consigue a través de “clicks” sobre el mapa. Se usará el botón izquierdo del ratón sobre el terreno para desplazarse a ese punto siempre que la zona sea transitable y esté conectada con la posición actual. Si este botón se usa sobre un enemigo, el personaje principal se moverá hasta su posición.

Para atacar a un enemigo, hay que seleccionarlo primero. Se usará el botón derecho del ratón sobre un enemigo para seleccionarlo. Si el enemigo ya está seleccionado y si ya estamos en el rango adecuado, pulsando otra vez el botón derecho del ratón ejecutaremos un ataque básico.

Al cabo de un tiempo sin que ocurra un intercambio de daño entre el enemigo seleccionado y el personaje principal, el enemigo se deseleccionará automáticamente.

7.7. Habilidades

Las habilidades especiales tienen tiempo de reutilización (tiempo de enfriamiento) dependiendo de su poder. Estas habilidades se ejecutan por medio de teclas del teclado. Las teclas predefinidas serán Q, W, E y R, siendo totalmente configurables por el usuario.

La habilidad Q lanza una bola de fuego que hará daño a los enemigos que atraviese. La habilidad W teletransporta al jugador cierta distancia, dentro de un rango permitido. La habilidad E proporciona un escudo de hielo que mitiga un porcentaje de daño recibido durante un tiempo. La habilidad R desencadena una explosión alrededor del jugador ocasionando una gran cantidad de daño mágico y

físico. Para consultar los efectos visuales de cada habilidad, véase el capítulo *9 EFECTOS VISUALES*.

A continuación se muestra una tabla con las características detalladas de las habilidades especiales.

Características de las Habilidades Especiales del Personaje Principal						
Icono	**Nombre**	**Descripción**	**Coste de Maná**	**Tiempo de uso**	**Beneficio**	**Rango de Acción**
	**Bola de Fuego**	El personaje lanza una bola de llamas que atraviesa a los enemigos infligiendo daño mágico.	25	1,5	Inflige el daño mágico del jugador como daño.	Enfrente del personaje, el proyectil se destruye a los 6 segundos.
	**Teletransporte**	Teletransporta al personaje a la posición actual del ratón instantáneamente.	25	5	--	4 unidades como máximo.
	**Escudo de Hielo**	Otorga un escudo temporal que mitiga el daño recibido por los enemigos.	200	15	Reducción del 35% del daño recibido.	El propio jugador. Dura 15 segundos.
	**Ragnarök**	Desencadena una explosión alrededor del personaje ocasionando grandes daños físicos y mágicos.	85% del maná máximo	60	La suma del 60% del daño mágico de jugador y el 40% de su daño físico, por 4.	Hasta 8 unidades alrededor del personaje.

Tabla 2 Características de las Habilidades Especiales del Personaje Principal

7.8. Estadísticas y Sistema de puntos de Característica

El usuario podrá personalizar al personaje principal a través de 5 atributos. Estos atributos son Fuerza, Inteligencia, Vitalidad, Velocidad de Ataque y Velocidad de Movimiento. El valor máximo de puntos que pueden ser asignados a estos atributos es de 30. Inicialmente se empieza con 10 puntos a repartir libremente por el jugador. Cada subida de nivel se otorgan al jugador 2 puntos y cada 10 niveles 1 adicional.

A continuación se muestra una tabla con los valor base de cada atributo y los efectos por cada punto asignado.

Tabla de Efectos de Características		
Característica	**Efecto por cada punto asignado**	**Valor Base**
Fuerza	Suma 10 puntos de daño al ataque básico	20 puntos de daño
Inteligencia	Suma 13 puntos de daño mágico a las Habilidades	15 puntos de daño
	Suma 0,3% de regeneración de maná máximo por segundo	0.0% de regeneración de maná por segundo
	Suma 10 puntos de maná máximo	200 puntos de maná máximo
Vitalidad	Suma 15 puntos de vida máxima	300 de vida máxima
	Suma 0,5% de regeneración de vida máxima por segundo	0.0% de regeneración de vida por segundo
Velocidad de Ataque	Resta el 5% de tiempo en completar la animación	0,533 segundos en completar la animación
Velocidad de Movimiento	Suma el 5% del valor base	3,5 en el atributo “Speed” del componente “NavMeshAgent“

Tabla 3 Efectos de los Puntos de Característica

7.9. Experiencia y Nivel

Cada enemigo otorga cierta cantidad de experiencia (dependiendo de su tipo). Inicialmente el personaje principal tiene nivel 1.

Un nuevo nivel otorgará puntos de característica. Cada nivel necesitará más experiencia que el anterior. Como en la mayoría de los videojuegos de rol, se va a usar una fórmula cuadrática para el cálculo de la experiencia necesaria. En nuestro caso, se ha seleccionado la siguiente:

$$12.75x^2+56.4803x+60.5667$$

La incógnita es el nivel actual. El resultado será la experiencia necesaria para alcanzar el siguiente nivel. A continuación se muestra el gráfico de los puntos de experiencia requeridos para los primeros veinte niveles.

Ilustración 17 Experiencia necesaria por Nivel

7.10. Objetos

Los objetos que se pueden obtener en el juego son pociones de vida y pociones de maná. En un futuro se podría incluir cualquier objeto, como bonificaciones temporales de resistencia, velocidad, etc.

Al morir un enemigo se instanciará aleatoriamente 1 poción de vida y otra de maná. Para las pociones de vida se tendrá una probabilidad del 40% y para las de maná del 30%. Las pociones de vida restauran 50 puntos de salud y las de maná 50 puntos de magia al personaje principal. Al entrar en contacto con una poción del suelo se usará inmediatamente si puede restaurarse esa cantidad de salud o magia.

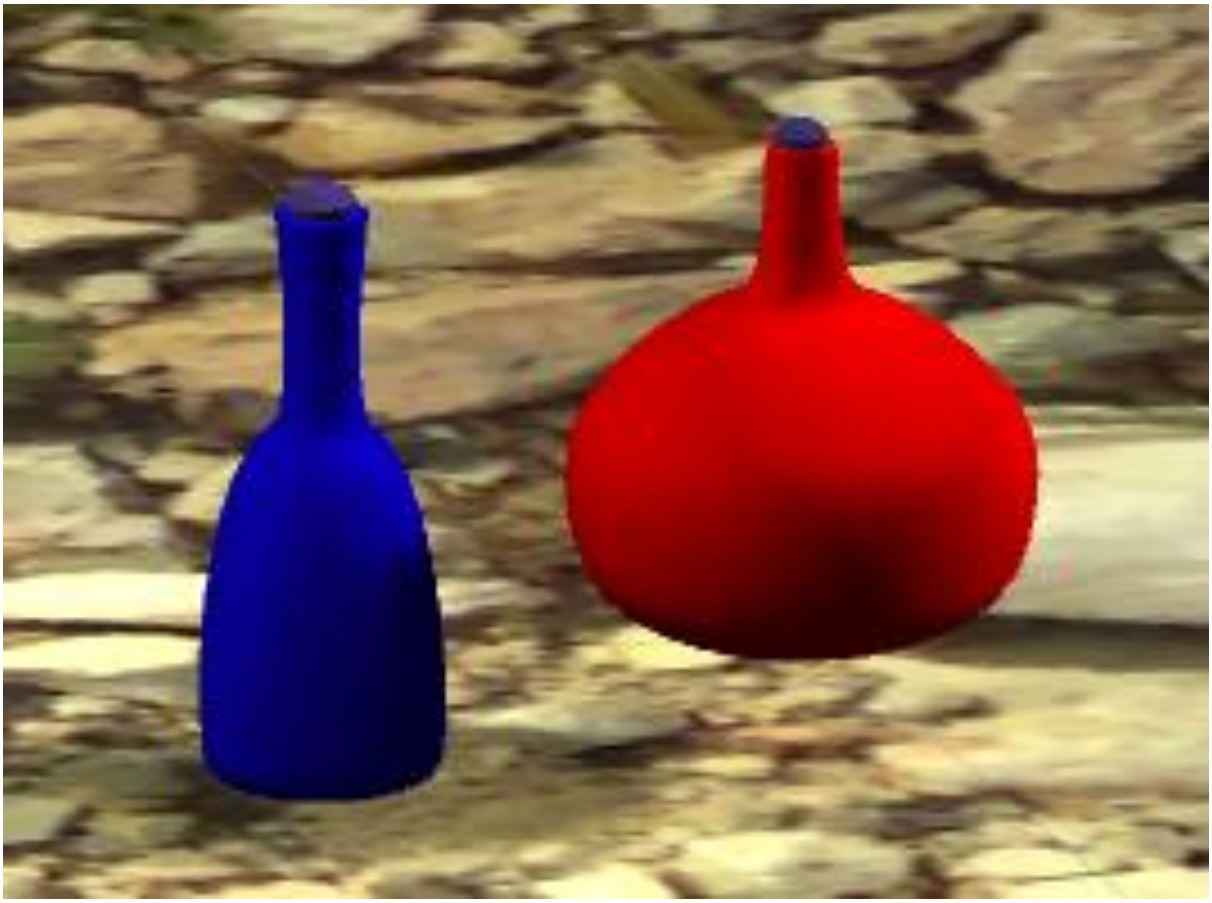


Ilustración 18 Una poción de maná (izquierda) y una poción de vida (derecha)

7.11. Enemigos

Los enemigos son una parte muy importante de este videojuego. Se han incluído 21 enemigos diferentes en el juego. Existen 15 enemigos son comunes y 6 enemigos son únicos (jefes).

Los enemigos únicos sólo aparecen una vez en el transcurso del juego. Estos enemigos son más difíciles y poseen mecánicas más variadas. La barra de vida de los jefes es ligeramente diferente a la de los enemigos comunes. Los jefes tienen un icono en el minimapa característico y tienen un aura distintiva, para que sean más reconocibles.

Los enemigos comunes se subdividen en 12 que atacan cuerpo a cuerpo y 3 que atacan a distancia.

El número de enemigos, sus tipos y su nivel varía dependiendo del mapa en cuestión.

7.11.1. Mecánicas de los Enemigos

El comportamiento de los enemigos consistirá en, una vez el personaje principal está en el rango del detección del enemigo, perseguir al jugador incesantemente para atacarlo hasta que uno de los dos agentes muera. Existe un diagrama de máquina de estados explicativo (*15.9 Diagrama Máquina de Estados: Enemigos*).

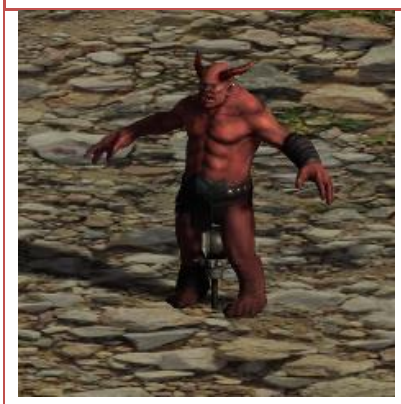
Evidentemente, los enemigos que atacan a distancia no necesitarán acercarse tanto al jugador para ejecutar sus ataque como los enemigos cuerpo a cuerpo.

7.11.2. Enemigos Comunes

Aquí se detallan los enemigos comunes y algunas peculiaridades sobre de su comportamiento.

Enemigos Comunes Cuerpo a Cuerpo	
Imagen	**Características**
	**Nombre:** Skeleton Lancer **Vida:** 200 **Ataque:** 10 **Comportamiento:** Cuando detecte al jugador lo perseguirá incesantemente hasta acabar con él cuerpo a cuerpo usando su lanza.
	**Nombre:** Skeleton Highlander **Vida:** 200 **Ataque:** 30 **Comportamiento:** Cuando detecte al jugador lo perseguirá incesantemente hasta acabar con él cuerpo a cuerpo usando sus dos espadas.
	**Nombre:** Skeleton Protector **Vida:** 400 **Ataque:** 20 **Comportamiento:** Cuando detecte al jugador lo perseguirá incesantemente hasta acabar con él cuerpo a cuerpo. Si el jugador tiene poco nivel puede suponer un gran reto.

	**Nombre:** Slime (Limo) **Vida:** 100 **Ataque:** 5 **Comportamiento:** Un ser repulsivo que habita en los pantanos. Se le suele encontrar en grupos numerosos. Ataca cuerpo a cuerpo. Es bastante débil.
	**Nombre:** Black Slime **Vida:** 300 **Ataque:** 25 **Comportamiento:** Tiene algo de más vida y ataque que los anteriores slimes. Puede parecer débil, pero nunca se debe menospreciar.
	**Nombre:** Mud Hunter **Vida:** 500 **Ataque:** 60 **Comportamiento:** Un cazador en estado puro. El Mud Hunter acecha a sus presas y espera el momento más oportuno para atacar por la espalda.
	**Nombre:** Mud Opressor **Vida:** 600 **Ataque:** 100 **Comportamiento:** Una versión más potente del Mud Hunter. Tiene más vida y casi el doble de ataque que éste último.

	**Nombre:** Goblin Snatcher **Vida:** 500 **Ataque:** 70 **Comportamiento:** En grupos numerosos son muy molestos, impidiendo el paso y asesinando a sus víctimas como si fuesen un enjambre. Cuando están solos la cosa cambia, suelen ser cobardes y huirán del combate.
	**Nombre:** Wolf **Vida:** 125 **Ataque:** 25 **Comportamiento:** Suelen atacar en manada, aunque no suponen un gran problema para el jugador experimentado. Habitan por todo el continente.
	**Nombre:** Troll **Vida:** 1200 **Ataque:** 160 **Comportamiento:** Una poderosa criatura que jamás rechazará una batalla. Suele ser lento, pero cuidado con sus ataques devastadores cuerpo a cuerpo. Tiene mucha vida. Se aconseja herirlo a distancia y rematarlo cuerpo a cuerpo.
	**Nombre:** Cyclops **Vida:** 1500 **Ataque:** 250 **Comportamiento:** Liberados del infierno, estas criaturas son usadas para proteger los tesoros de sus invocadores. No temen a la muerte y provocan un gran daño.

	**Nombre:** Flesh Golem **Vida:** 1700 **Ataque:** 90 **Comportamiento:** Otra criatura invocada del infierno. Son muy resistentes en combate al de tener mucha vida.
	**Nombre:** Earth Golem **Vida:** 1500 **Ataque:** 180 **Comportamiento:** Estos Golems doblan en ataque al Flesh Golem, teniendo casi la misma vida.

Tabla 4 Enemigos Comunes Cuerpo a Cuerpo

Enemigos Comunes a Distancia	
Imagen	**Características**
	**Nombre:** Skeleton Mage **Vida:** 170 **Ataque:** 20 **Comportamiento:** El mago atacará a distancia al jugador, lanzándole bolas de fuego. Su velocidad al lanzar hechizos es alta y no hay que tomarlo a la ligera, sobre todo se encuentra rodeado por otros enemigos.

	**Nombre:** Red Slime (Limo) **Vida:** 100 **Ataque:** 10 **Comportamiento:** Es un monstruo que prefiere atacar a distancia. Como los limos anteriores, se mueve en grandes grupos y si el jugador no tiene cuidado puede llegar a morir al enfrentarse a tantos enemigos a la vez.
	**Nombre:** CaveWorm **Vida:** 250 **Ataque:** 25 **Comportamiento:** Un híbrido gusano-araña que vive en las cuevas y grutas más oscuras. Escupe ácido a sus víctimas a gran velocidad.

Tabla 5 Enemigos Comunes a Distancia

7.11.3. Enemigos Élite (Jefes)

De forma análoga a los enemigos comunes se presentan los enemigos únicos del videojuego. Estos enemigos poseen un aura de color característica alrededor de ellos. Este efecto puede observarse en el capítulo *9 EFECTOS VISUALES*.

El daño y la vida de estos enemigos es muy superior a la de los enemigos comunes. Las mecánicas y comportamientos son mucho más complejas y espectaculares.

Enemigos Únicos (Jefes)	
Imagen	**Características**
	**Nombre:** Burp, Slime King **Vida:** 900 **Ataque:** 80 **Comportamiento:** Nunca se separa de su reina y suele engullir a quien ose entrar en sus dominios de la ciénaga. Ataca cuerpo a cuerpo.
	**Nombre:** Gulp, Slime Queen **Vida:** 700 **Ataque:** 90 **Comportamiento:** Ataca a distancia, siendo protegida por el rey Slime.
	**Nombre:** Sir Thomas II, Relentless Commander **Vida:** 350 **Ataque:** 40 **Comportamiento:** Un luchador, caído en la batalla hace siglos. Ha resurgido para conquistar el reino de Xanria y sembrar el caos y la destrucción. Ataca cuerpo a cuerpo e invoca espadas espectrales que lo protejan, eliminando a los enemigos que se acerquen demasiado.

	**Nombre:** Abramelin, The Dark Mage **Vida:** 5000 **Ataque:** 300 **Comportamiento:** Invoca esqueletos cada cierto tiempo y lanza enormes bolas de fuego infernal a sus enemigos. Es un duro rival si no se está preparado.
	**Nombre:** Nightmare **Vida:** 2000 **Ataque:** 170 **Comportamiento:** Ataca a distancia lanzando ácido a sus rivales. Es la mascota de Abramelin. Este engendro demoníaco protege la entrada a la torre del mago.
	**Nombre:** SkullCrusher **Vida:** 6000 **Ataque:** 400 **Comportamiento:** Invocado desde las mazmorras del infierno, esta enorme criatura aplasta a sus enemigos sin esfuerzo. Es muy resistente y se va curando vida en el tiempo. Cuenta con el ataque más alto jamás visto en Xanria.

Tabla 6 Enemigos Únicos (Jefes)

7.12. Funcionamiento

El juego consiste en ir completando diferentes mapas a lo largo del transcurso de la historia.

Cada mapa tiene unos diferentes objetivos a completar. Eliminar a todos los enemigos, eliminar a un jefe final o encontrar un objeto, entre otros, son los posibles objetivos.

Cuando un mapa sea superado, automáticamente se guardará la partida en un archivo binario en el directorio de los archivos del juego. Al finalizar una misión se obtendrá una calificación. Esta calificación depende del grado de completitud de los objetivos principales y secundarios.

Aunque un mapa haya sido superado, se podrá volver a jugar para subir de nivel, batir el record anterior o simplemente disfrutarlo.

Si el usuario falla a la hora de completar el mapa volverá inmediatamente al punto inicial, con intentos ilimitados.

Si un usuario cierra la aplicación en medio de una misión, su avance no se guardará hasta que no lo complete.

7.13. Flujo de la Experiencia de Juego

El juego está construido de tal forma que los mapas se ajusten a un rango de niveles concreto. Así, un usuario puede verse obligado a jugar con más calma o a repetir algún nivel anterior para ganar experiencia y hacerse más fuerte. También se premia así a los jugadores que disfrutan exprimiendo cada mapa. Si se completan todas las misiones al 100% se obtiene más experiencia y, por lo tanto, se afrontan los siguientes retos siendo más fuerte.

Los nuevos jugadores disfrutarán enfrentándose a verdaderos retos y los jugadores expertos encontrando nuevas formas de superar las misiones completándolas totalmente.

8. INTERFAZ GRÁFICA

Desde el menú principal (el punto de acceso al vieojuego) se puede acceder a los mapas que se tengan desbloqueados. Cuando un mapa sea superado, se desbloqueará el siguiente.

Como se ha desarrollado un prototipo, no se dispone de imágenes del menú principal. En su lugar se mostrará la pantalla de título inicial del prototipo.

Ilustración 19 Pantalla de Título Principal del Prototipo

Los elemento restantes pertenecientes a la interfaz gráfica son la interfaz gráfica de usuario (GUI, Graphics Interface User) dentro del juego en sí, el minimapa y los menús de opciones, pausa y demás.

8.1. La Interfaz Principal (GUI)

La interfaz gráfica de usuario trata de no ser invasiva en el juego, ocupando una pequeña parte de pantalla, en la zona inferior izquierda. Los menús podrán ser activados con atajos de teclado o haciendo click sobre su botón en particular.

Se ha tratado de hacer una interfaz limpia y que sea tan sencilla que no necesite explicación.

Ilustración 20 Interfaz Gráfica de Usuario (GUI) dentro del juego

La interfaz del juego muestra:

- Las habilidades especiales, sus teclas de uso rápido asignadas por el jugador (Q, W, E y R en este caso) y su disponibilidad (tiempo de reutilización restante, los segundos sobre las dos habilidades centrales).
- Un retrato del personaje controlado.
- El nivel del personaje (1 en este caso en la esquina inferior derecha del retrato del personaje).
- Las barras de vida (barra en rojo), maná (barra en azul) y experiencia (barra de color morado).
- Un conjunto de accesos a los diferentes menús.

8.2. El Minimapa

Existe un minimapa en la parte inferior derecha de la pantalla para orientar al jugador en el transcurso de las misiones. El jugador y los enemigos aparecen en el minimapa representados por sus iconos característicos.

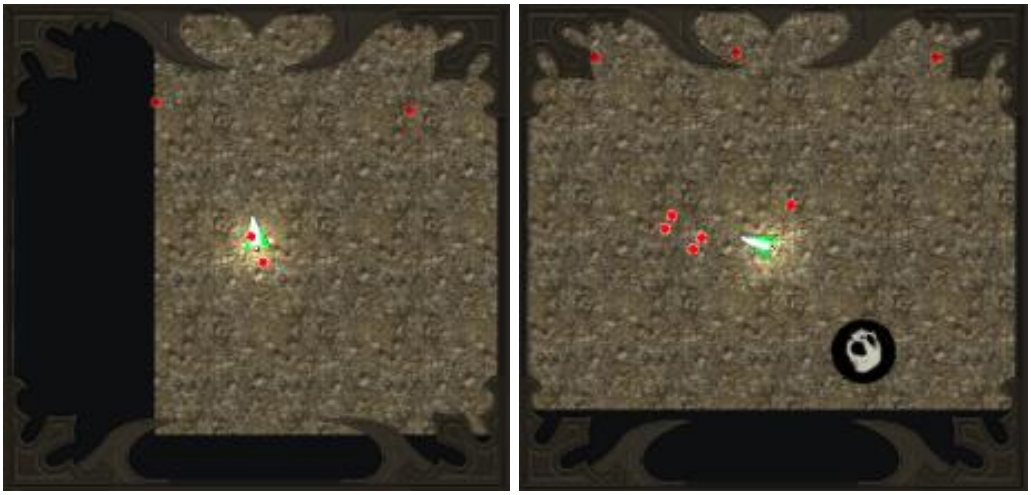


Ilustración 21 Ejemplos de Minimapa con iconos representando a los diferentes agentes

Cada agente (personaje principal, enemigo común y jefe) tiene un icono propio con el que diferenciarse del resto.

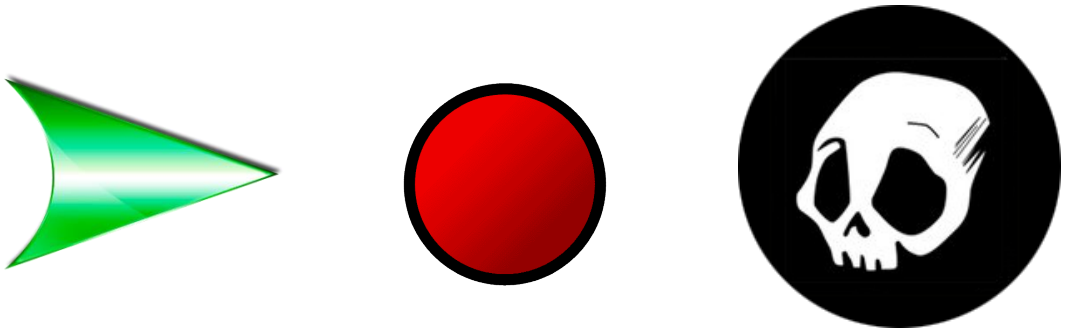


Ilustración 22 De izquierda a derecha, icono del personaje principal, icono de un enemigo común e icono de un jefe

8.3. Menús y Navegación

Los menús con los que cuenta el juego son:

- Menú de Estadísticas en el que, además de mostrar información, se pueden repartir los puntos de característica obtenidos al subir de nivel.
- Menú de Pausa, para pausar el juego.
- Menú de Opciones, en la que se puede modificar el detalle gráfico, el sonido o las teclas definidas por el usuario para el uso de las habilidades especiales.
- Menú de Abandonar Partida, en el que se puede volver al menú principal.

Exceptuando el menú de pausa, los demás menús no detienen el transcurso del juego, así que hay que tener cuidado a la hora de decidir abrir un menú o no.

Las teclas asignadas a la activación y desactivación de estos menús son “P” para el menú de Pausa, “O” para el menú de Opciones, “L” para el menú de Abandonar Partida y “S” para el menú de Estadísticas (características del personaje principal).

Las siguientes imágenes muestran los diferentes menús del videojuego.

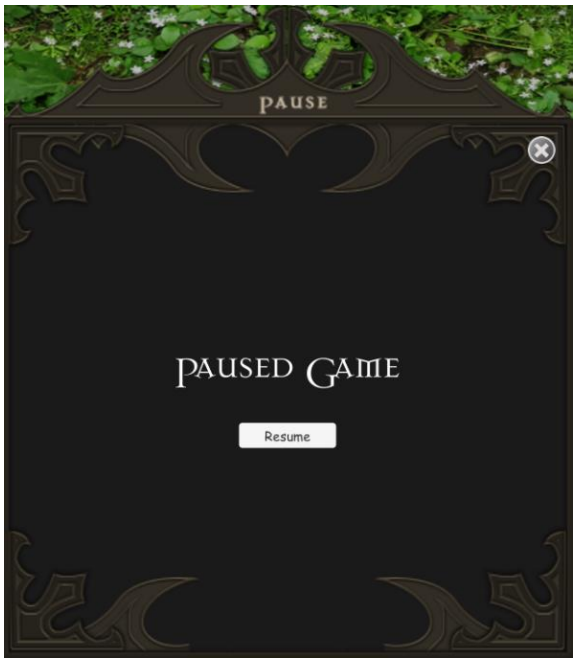


Ilustración 23 Menú Pausa

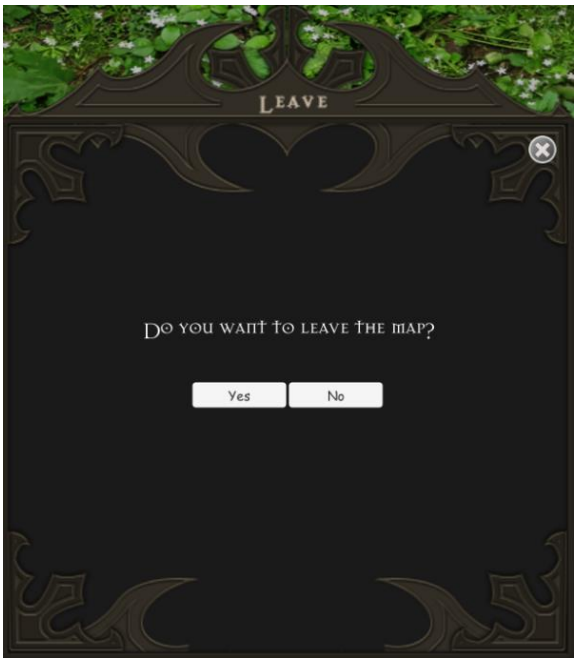


Ilustración 24 Menú Abandonar Partida

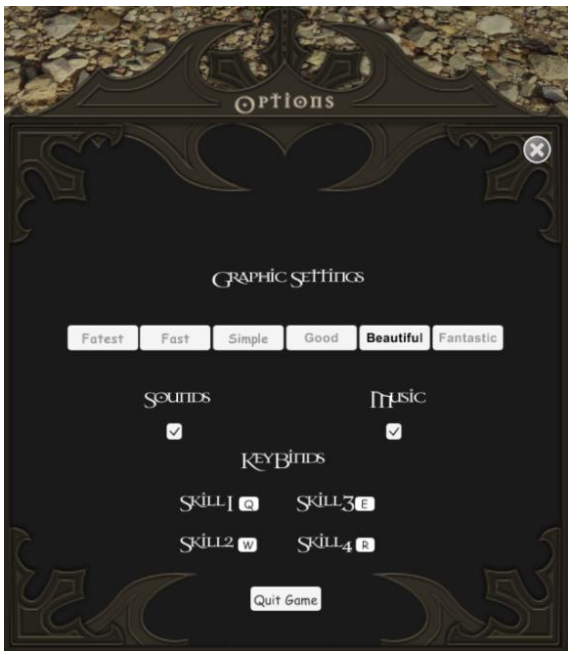


Ilustración 26 Menú Opciones

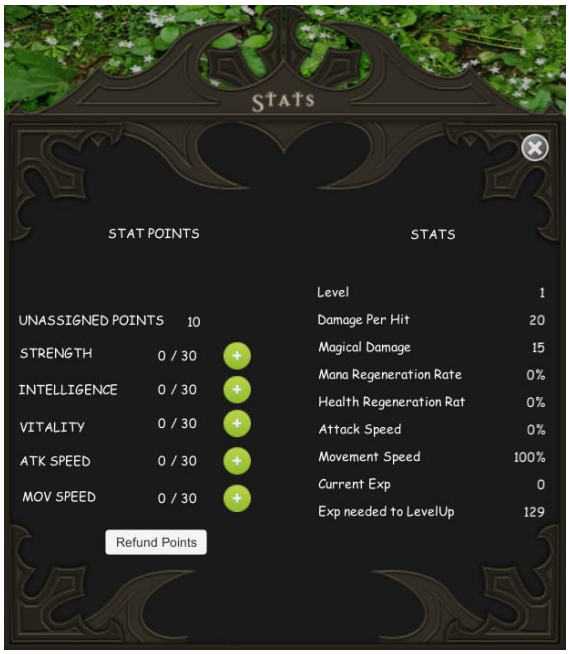


Ilustración 25 Menú Estadísticas

9. EFECTOS VISUALES

En este capítulo se recopilan las capturas de los efectos visuales más representativos con los que cuenta el juego.

9.1. Habilidades Especiales del personaje

Ilustración 27 Habilidad Especial "Bola de Fuego"

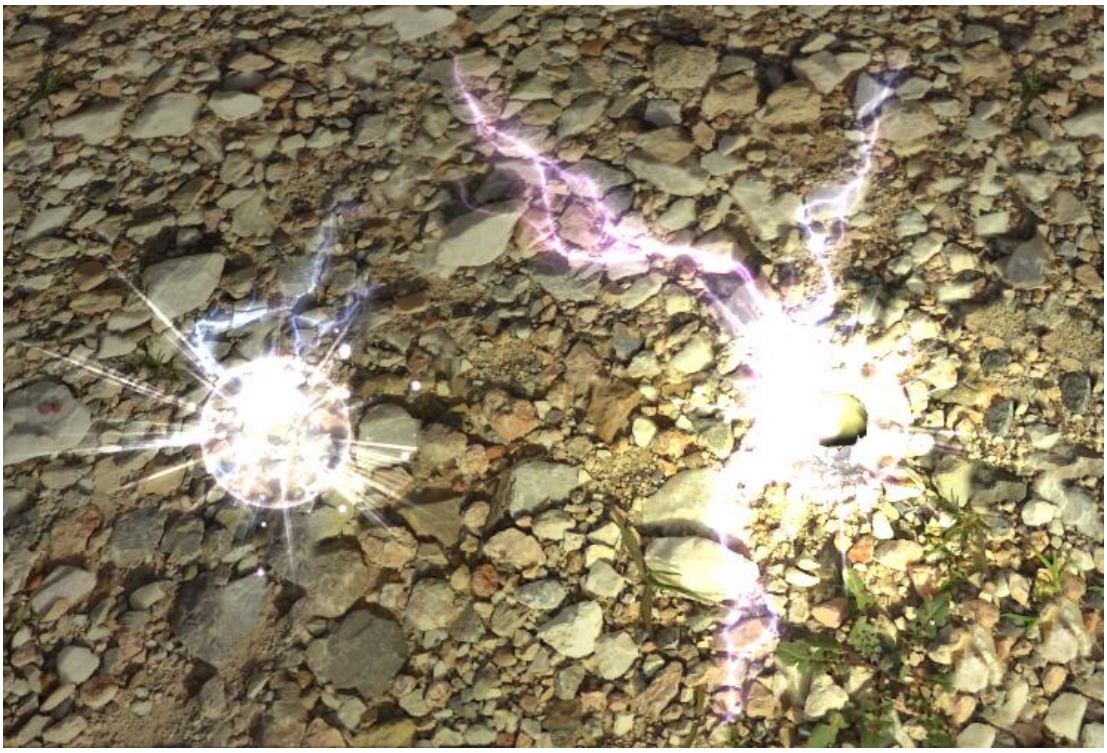


Ilustración 28 Habilidad Especial "Teletransporte"

Ilustración 29 Habilidad Especial "Escudo de Hielo"

Ilustración 30 Habilidad Especial "Ragnarök"

9.2. Proyectiles de los enemigos a distancia

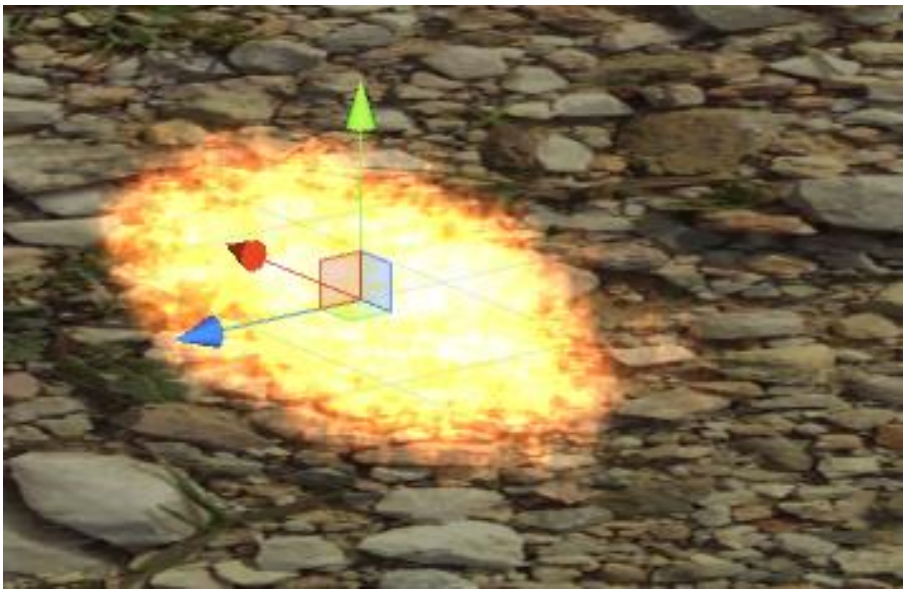


Ilustración 31 Proyectil del enemigo Skeleton Mage

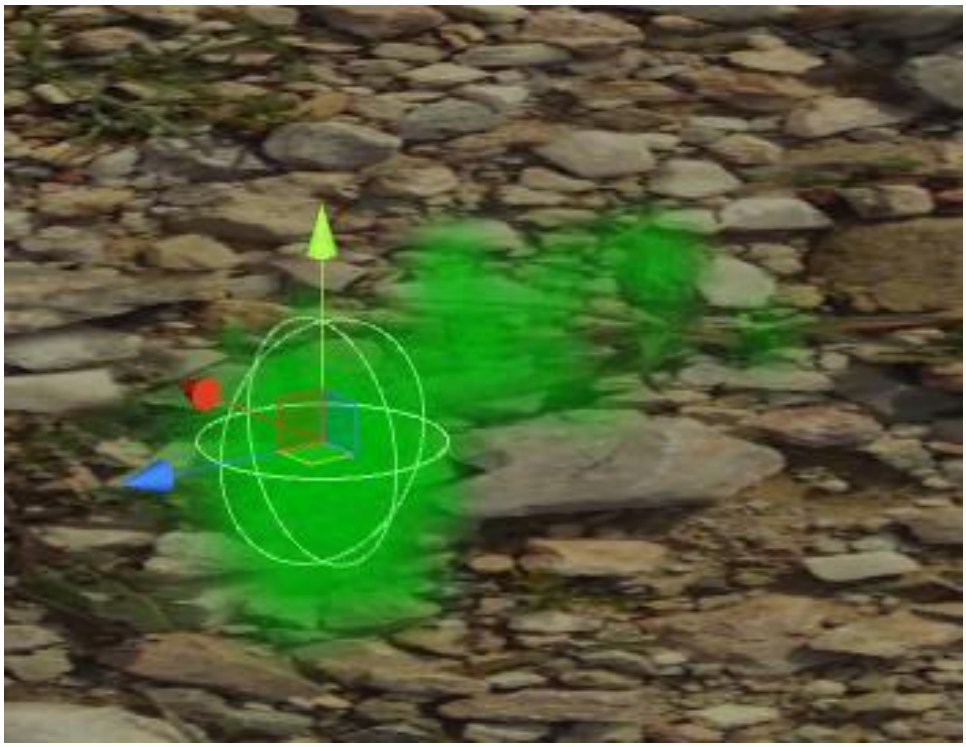


Ilustración 32 Proyectil del enemigo CaveWorm

Ilustración 33 Proyectil del enemigo Red Slime

Estos proyectiles se desplazan en la dirección del eje z positivo (flecha azul en las imágenes).

9.3. Sangrado de los enemigos y el personaje principal

Estos efectos de sangrado se aplican cuando un agente recibe daño. Cada agente posee su propio efecto de sangrado con color y tamaño personalizado.

Ilustración 34 Capturas de efecto de sangrado del personaje principal

Ilustración 35 Efecto de sangrado verde del enemigo Slime

9.4. Efecto “Glow” (Brillo) en los enemigos

Este efecto se aplica cuando el usuario selecciona a un enemigo.

Ilustración 36 Skeleton Lancer normal y con efecto Glow

9.5. Efecto de relleno y vaciado de barras

Las barras de vida, maná y experiencia del personaje actualizan su valor através de una interpolación lineal, lo que hace que parezca que se vayan rellenando o vaciando poco a poco. La barra de vida de los enemigos tiene el mismo efecto.

9.6. Textos flotantes

Para mejorar la retoralimentación del usuario se han incluído textos flotantes para el daño, la ganancia de vida y la ganancia de maná.

Ilustración 38 Texto flotante de ganancia de vida

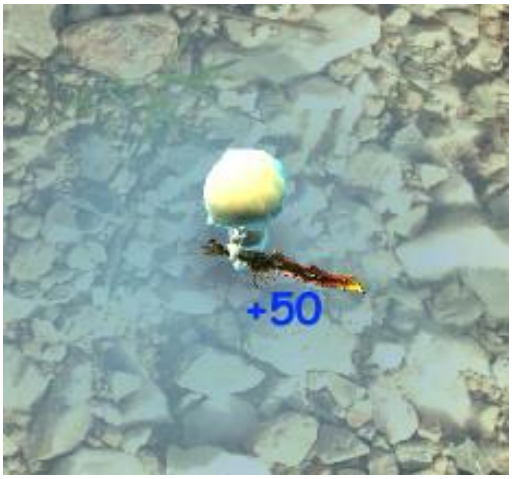


Ilustración 37 Texto flotante de ganancia de magia

Ilustración 39 Texto flotante de daño y efecto glow en Skeleton Lancer

9.7. Subida de nivel

Al subir de nivel el personaje principal será rodeado por un halo de fuego durante unos segundos, ganando un efecto de brillo naranja momentáneamente.

Ilustración 40 Efectos de subida de nivel del personaje principal

9.8. Auras de los Jefes

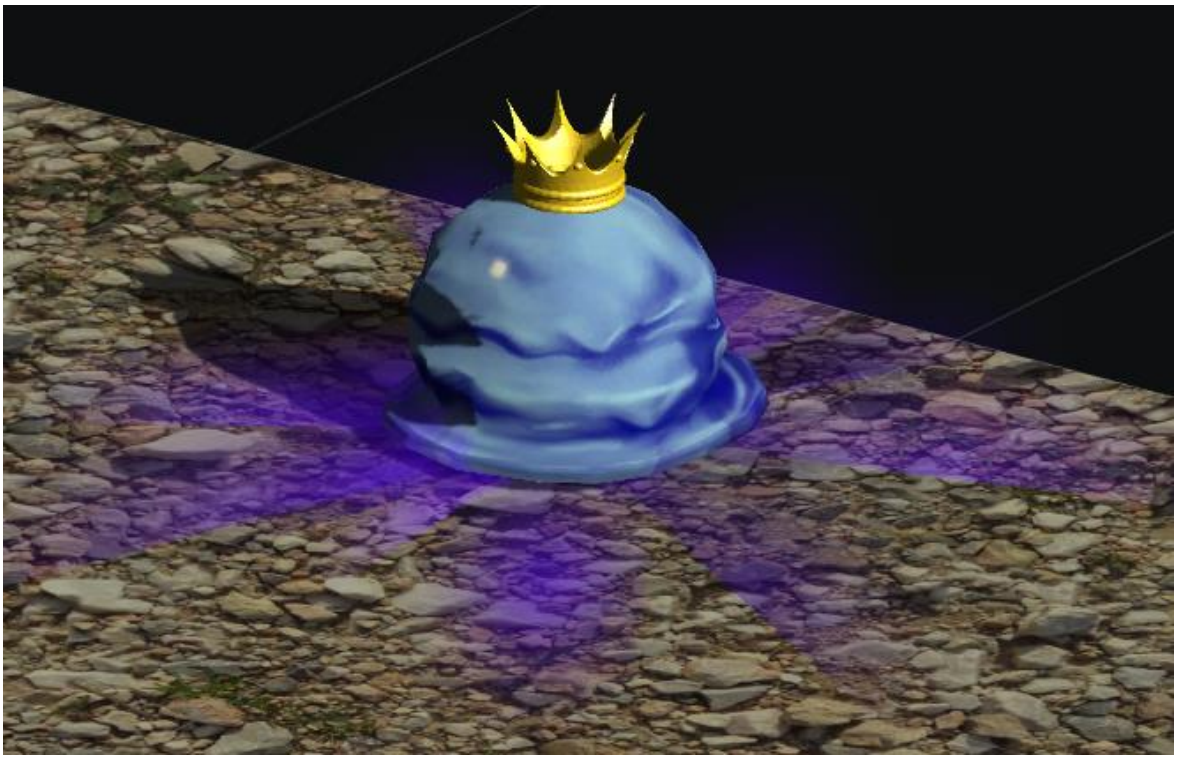


Ilustración 42 Jefe Burp con Aura

Ilustración 41 Jefe Sir Thomas II son Aura

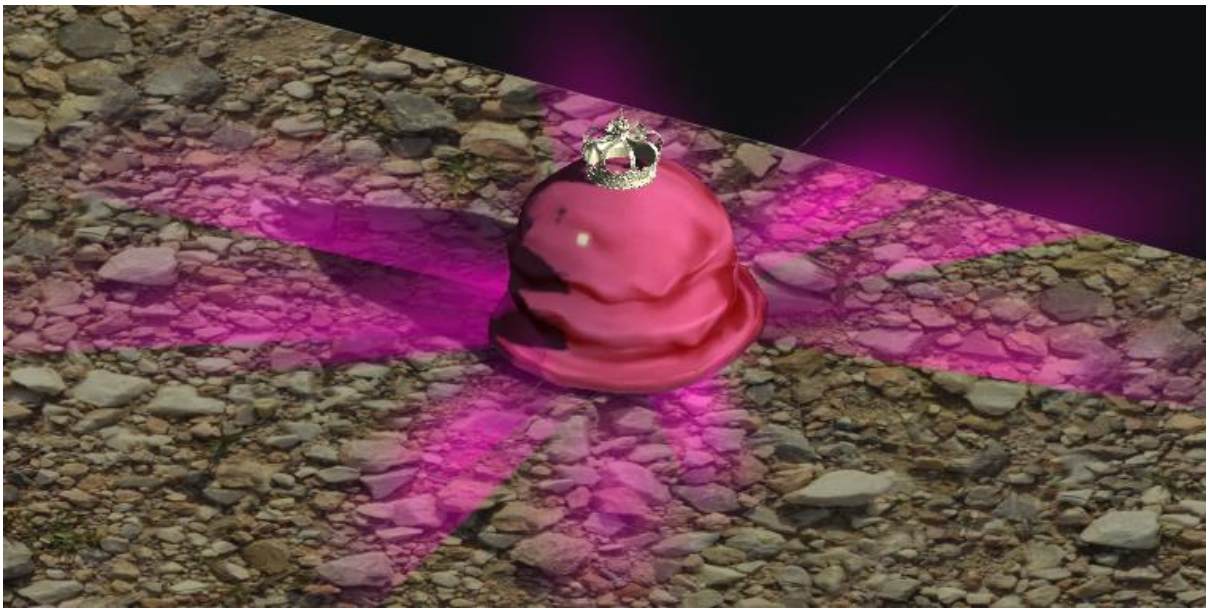


Ilustración 43 Jefe Burp con Aura

Ilustración 44 Jefe Nightmare con Aura

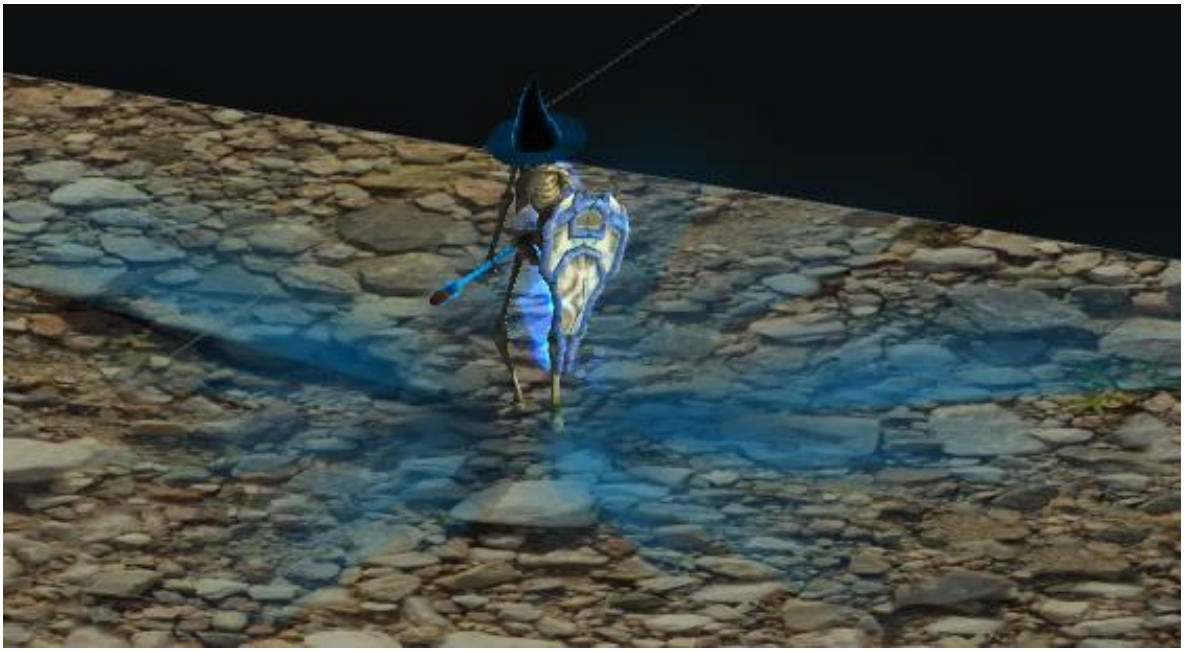


Ilustración 46 Jefe Abramelin con Aura

Ilustración 45 Jefe SkullCrusher con Aura

10. EFECTOS DE SONIDO

Este proyecto no dispone de efectos sonoros ya que estas características han quedado fuera del ámbito del proyecto. Al final de la fase de desarrollo se han incluído un par de pistas ambientales en la pantalla de título principal y la escena Demo para contribuir a crear la oscura atmósfera en la que se quiere zambullir al jugador.

11. DEFINICIÓN DEL ÁMBITO DEL PROYECTO

El autor de este trabajo ha tomado el rol de gestor del proyecto y desarrollador. Esta doble labor implica la realización simultánea de tareas de gestión, planificación, control y desarrollo.

La definición del alcance del proyecto es el punto de partida para una planificación, gestión y ejecución exitosa. Se va a tratar el proyecto de una manera profesional, como si se tratase de un encargo a una pequeña empresa de desarrollo. Con esta idea, en la etapa de planificación, se estimarán costes y recursos de una manera real. Esta definición será vital a la hora de elegir el ciclo de vida de la gestíon del proyecto.

11.1. Alcance del proyecto (Extracción de Requisitos)

Como no se dispone de un cliente real al que involucrar en la etapa de extracción de requisitos, se redactará una definición del proyecto que nos permitirá obtener los requisitos a cumplir por el producto. Después de analizar la definición, se obtendrán los principales requisitos del proyecto.

11.1.1. Definiendo el proyecto

Es un videojuego 3D, el cual mezcla aventura y acción. El objetivo de este videojuego es entretener al jugador. El usuario vivirá aventuras, avanzando linealmente por la historia, hasta acabar el juego.

El juego estará compuesto por una sucesión de misiones diferentes. Cada misión tendrá unos objetivos principales a completar y podrá tener objetivos secundarios opcionales. Las recompensas de las misiones serán experiencia y una puntuación por misión que dependerá de los objetivos cumplidos, los enemigos asesinados y las veces que el jugador ha muerto intentando completar dicha misión. Cuando se completa una misión, se debloquea la siguiente si no es ésta la misión final. Las misiones podrán volverse a jugar pudiendo mejorar la puntuación. Las misiones no deben ser excesivamente largas (preferiblemente de 25 a 40 minutos de duración), para que no se vuelvan tediosas.

Existirá una amplia gama de enemigos diferentes, con diferentes mecánicas y características (vida, daño y velocidad de ataque). Existirá un tipo especial de enemigos llamados Jefes, los cuales serán más fuertes y tendrán mecánicas más interesantes. Se requerirán unos 15 enemigos comunes y unos 6 Jefes.

Los Jefes sólo aparecerán una vez en el juego. La función de estos enemigos será bajar la vida del jugador a 0. Los enemigos tendrán un rango de visión con el que detectarán al jugador y empezarán a perseguirlo. Los ataques de los enemigos deben poder esquivarse. Al acabar con un enemigo, éste otorgará al jugador experiencia y, de forma aleatoria, pociones que restauren parte de su vida o maná.

Eventualmente, con la experiencia ganada, el jugador subirá de nivel. La cantidad de experiencia requerida para subir de nivel será cada vez más alta conforme el jugador tenga un nivel más elevado. La vida y el ataque de los enemigos escalará con el nivel del jugador para evitar que el juego deje de ser un reto. Al subir de nivel, se otorgará un punto de característica que podrá ser distribuído por el usuario entre las diferentes características de su avatar (fuerza, velocidad de ataque, vitalidad, inteligencia y velocidad de movimiento). Este sistema de características debe dar lugar a variadas personalizaciones del avatar. Cada característica tendrá un máximo posible. Estos puntos de característica deben poder redistribuirse en cualquier momento si el usuario lo desea.

El usuario desplazará a su avatar (representación de él mismo en el mundo virtual) usando el botón derecho del ratón en una posición del mapa. Usando el botón izquierdo del ratón sobre un enemigo se efectuará un ataque simple sobre éste.

El sistema de combate será en tiempo real. El jugador contará con 4 habilidades especiales únicas que costarán maná y tengan tiempos de enfriamiento. Estas habilidades especiales serán notorias visualmente y el buen uso de ellas recompensará al jugador con más probabilidades de victoria en sus combates.

Las teclas usadas para la ejecución de las habilidades especiales podrán ser definidas por el usuario.

Se usará una cámara Top-Down que seguirá automáticamente al personaje principal.

Las acciones de los enemigos y el personaje tendrán sonidos característicos y animaciones fluídas. La música usada será ambiental y variada.

El guardado y cargado de la partida será transparente al jugador. Se cargará la partida al iniciar el juego y se guardará cada vez que se finalice una misión.

La interfaz de usuario ocupará la mínima porción posible de la pantalla y ofrecerá información en tiempo real de los recursos más importantes del personaje (nivel, puntos de vida, maná y experiencia). Integrado en la GUI existirá un minimapa que alertará al usuario de enemigos en las cercanías y dará una visión cenital del mapa real. Desde la GUI se tendrán acceso a menús que controlarán los niveles de sonido, el detalle gráfico, la opción de pausa, la opción de abandonar partida y la distribución de puntos de característica.

La información proporcionada por la GUI debe ser clara y concisa. Se requieren barras de vida, maná y experiencia vistosas. Se requieren iconos en el minimapa que representen al jugador, a los enemigos simples y a los jefes.

Al combatir se mostrarán los números del daño causado o recibido, las ganancias de vida o maná y los puntos de característica sin asignar. Se utilizarán efectos visuales llamativos para alertar al usuario de eventos importantes como existencia de enemigos especiales (Jefes).

Se requiere que las transiciones entre escenas sean fluidas visualmente.

La ambientación general del juego será oscura y misteriosa. El juego debe ser simple para jugadores novatos y profundo para jugadores expertos.

11.1.2. Principales requisitos del proyecto y tiempo estimado por requisito

Esta es la lista de los principales requisitos extraídos de la anterior definición, ordenados por orden de prioridad. Se ha realizado una estimación inicial del tiempo necesario para satisfacer cada requisito principal.

Requisitos Principales	Estimación (Horas)
1. Usando el ratón, el usuario desplazará al personaje principal (botón derecho) y efectuará ataques básicos contra los enemigos (botón izquierdo). El personaje principal tendrá animaciones fluidas y cuidadas.	**8**
2. Los enemigos detectarán al jugador en un rango y lo perseguirán, atacándolo hasta que la vida de éste sea 0. Los enemigos dañarán la vida del jugador y viceversa. Los ataques de los enemigos se podrán esquivar.	**19**
3. Se implementará una cámara principal Top-Down que seguirá al personaje principal automáticamente y otra cámara cenital que generará un minimapa.	**6**
4. El personaje principal ganará experiencia de los enemigos derrotados, subirá de nivel y obtendrá puntos de característica.	**9,5**
5. El personaje principal tendrá un sistema de características personalizable que podrá ser modificado en cualquier momento.	**18**
6. Se tendrá una GUI no intrusiva, que reflejará la vida, maná, experiencia y nivel actual del personaje, además de permitir configurar el juego desde otros submenús.	**33,5**
7. El personaje principal tendrá 4 habilidades especiales que ayudarán al jugador a superar los retos propuestos por el juego. Estas habilidades serán notorias visualmente, su uso costará maná y tendrán un tiempo de enfriamiento. El uso de estas habilidades se reflejará en la GUI con un contador que informará al usuario de su disponibilidad.	**46**

8. Existirá una amplia gama de enemigos diferentes, con mecánicas diferentes. Existirá un tipo de enemigo especial llamado Jefe que contará con mecánicas más interesantes y más puntos de vida y ataque y su aparición en el juego será única. Se crearán unos 15 enemigos comunes y unos 6 Jefes. Estos enemigos otorgarán al jugador experiencia y, de forma aleatoria, pociones que restauren parte de su vida o maná. Los enemigos tendrán animaciones fluidas y cuidadas.	**94**
9. Las teclas usadas para la ejecución de las habilidades especiales podrán ser definidas por el usuario (Sistema de Keybinds).	**9**
10. Se utilizarán efectos visuales llamativos para alertar al usuario de eventos importantes: Pérdidas o ganancias de vida, pérdidas de maná, daño causado, puntos de característica sin asignar, subidas de nivel y existencia de enemigos especiales (Jefes). La información proporcionada por la GUI debe ser clara y concisa. Las barras de vida, maná y experiencia deben ser vistosas. Deben existir iconos en el minimapa que representen al jugador, a los enemigos y a los enemigos especiales (Jefes) unívocamente. Se requiere que las transiciones entre escenas sean fluidas visualmente.	**17**
11. El juego estará dividido en misiones que han de completarse de manera lineal. Una misión se desbloquea si se ha completado satisfactoriamente la anterior. Las misiones dan experiencia y una puntuación como recompensa. Cada misión tendrá unas objetivos principales a completar y podrá tener objetivos secundarios opcionales. Las misiones podrán volverse a jugar pudiendo mejorarse la puntuación. El guardado y cargado de la partida será transparente al jugador.	**185**
12. La música usada será ambiental y variada. Las acciones de los enemigos y el personaje tendrán sonidos característicos	**59,5**

Tabla 7 Requisitos Principales

11.2. Estructura de Desglose de Requisitos

El nivel superior del RBS (Requirement Breakdown Structure) está constituído por los principales requisitos. Una de las ventajas del RBS es que es una representación intuitiva y cercana al cliente. Muestra una visión clara del grado en que la solución está definida. Permite elegir el modelo de ciclo de vida de la gestión del proyecto por su grado decompletitud. Constituye la parte superior del WBS (Work Breakdown Structure), el cual será construído en la fase de planificación. Este documento sirve de referencia al cliente desde la fase de definición del ámbito del proyecto (José Ignacio Gómez Espínola, teoría Gestión de Proyectos Software, 2015).

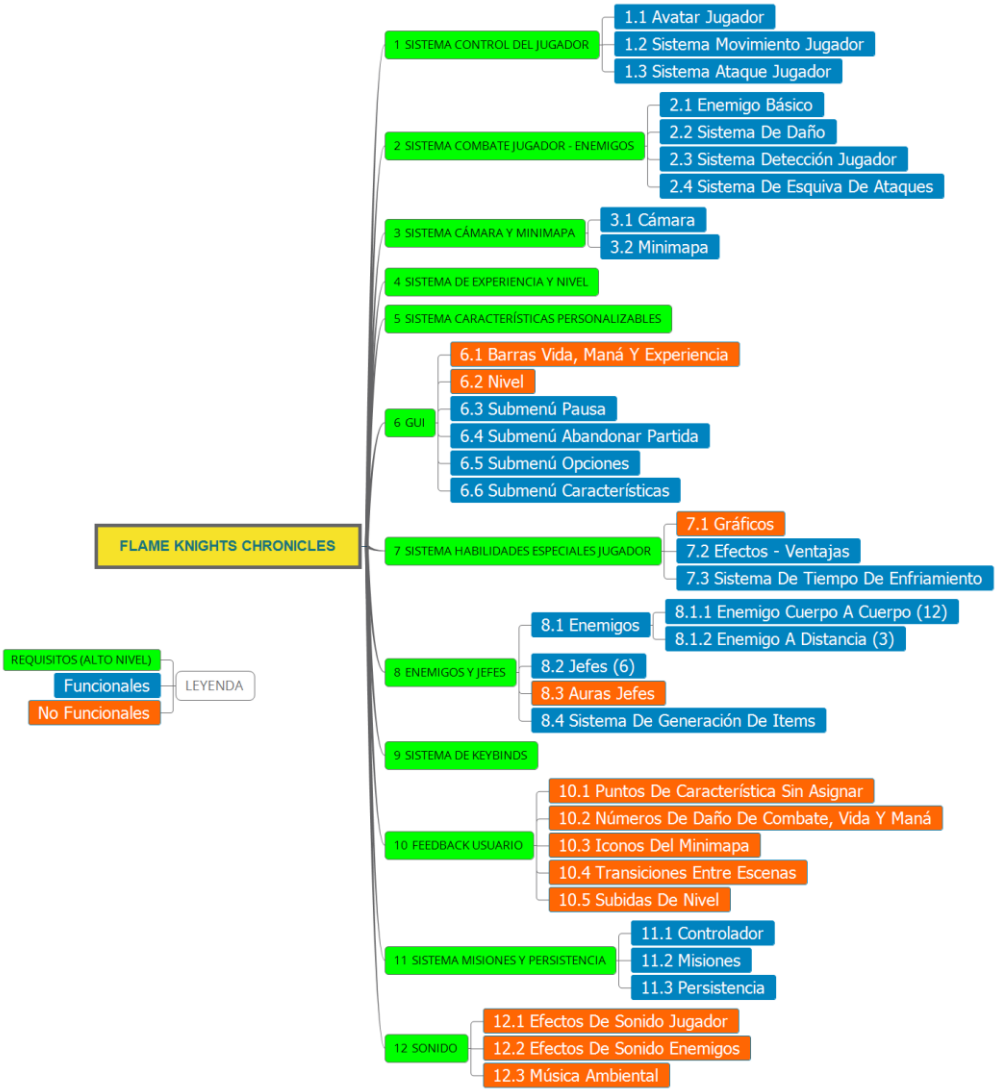


Ilustración 47 Diagrama RBS

11.3. Ciclo de Vida de la Gestión del Proyecto

Para la planificación y ejecución de las futuras tareas de desarrollo que se definirán en la etapa de planificación se ha optado por el uso de una metodología ágil (Scrum). La razón que lleva a esta decisión es que el diagrama RBS (*Véase Apartado 11.2*), aun siendo bastante completo, no está muy detallado y el equipo de desarrollo no tiene experiencia en este tipo de proyectos.

11.3.1. ¿Por qué Scrum?

Teniendo en cuenta la naturaleza del proyecto, una metodología ágil es totalmente aplicable. El proyecto tiene requisitos cambiantes, ya que estos están definidos vagamente y en cualquier momento se puede eliminar o modificar cualquier funcionalidad. El equipo de desarrollo no posee experiencia en un proyecto parecido. Se va a realizar un aprendizaje por descubrimiento, una característica propia de metodologías ágiles. Se requiere una metodología para afrontar un todo muy extenso poco a poco, con incrementos de funcionalidad, teniendo un producto funcional desde el primer momento.

11.3.2. Problemas a la hora de adaptar Scrum

La desventaja de desarrollar en solitario y usar Scrum es la imposibilidad de llevar a cabo las reuniones propuestas por la metodología, tanto de planificación como de retroalimentación, las cuales son muy importantes.

Una forma de paliar el efecto negativo de la falta de visión en el alcance ha sido mantener regularmente conversaciones con varias personas “expertas” en el mundo de los videojuegos, que han propiciado la inserción de nuevos requisitos. Con esta idea se ha involucrado colateralmente a 4 personas que han testeado el producto en las fases de pruebas de usuario, descubriendo anomalías que han sido corregidas y proporcionando una retroalimentación muy valiosa.

11.4. Declaración General Del Proyecto

Este documento es una declaración general del proyecto. Es un medio para obtener la aprobación y pasar a la fase de planificación. Debe ser aprobado por el cliente, el gestor, los responsables de recursos y el equipo de desarrollo.

<table>
<tr><td>PROJECT OVERVIEW STATEMENT</td><td>Project name
Flame Knights Chronicles</td><td>Project No.
001</td><td>Project Manager
Alejandro Gómez López</td></tr>
<tr><td colspan="4">Problem/Opportunity
La industria del entretenimiento digital es un sector en expansión, que genera al año más de 100.000 millones de dólares. Este hecho nos brinda una oportunidad de negocio muy rentable.</td></tr>
<tr><td colspan="4">Goal
Obtener al final de la fase de ejecución del proyecto un videojuego comercializable que satisfaga los requisitos expresados por el cliente en términos de calidad y funcionalidad.</td></tr>
<tr><td colspan="4">Objectives
Obtener un prototipo básico que contenga al menos:
El sistema de Control del personaje
El sistema de Combate
El sistema de Habilidades
El sistema de Experiencia
El sistema de Características
15 Enemigos comunes y 6 Jefes
El sistema de teclas de habilidades configurables por el usuario (Keybinds).
El sistema de misiones y persistencia
Efectos de sonido y música</td></tr>
<tr><td colspan="4">Success Criteria
Que un 30% de los clientes potenciales compren el juego.
Que el 80% de los usuarios que prueben el juego lo valoren positivamente.</td></tr>
<tr><td colspan="4">Assumptions, Risks, Obstacles
La posibilidad de que el equipo informático que usamos para desarrollar se estropee en un momento crucial del proyecto.
Disponer siempre de conexión a internet para buscar información.
Disponer del tiempo necesario para llevar a cabo el proyecto.
Aprender rápidamente lo necesario para seguir avanzando en el desarrollo.</td></tr>
<tr><td>Prepared by
Alejandro Gómez López
Autor TFG</td><td>Date
01/02/2016</td><td>Approved by
Juan Roberto Jiménez Pérez
Tutor TFG</td><td>Date
01/02/2016</td></tr>
</table>

Tabla 8 Documento POS

12. PLANIFICACIÓN DEL PROYECTO

Una buena planificación es importante porque incrementa la comprensión del proyecto y mejora sustancialmente la eficiencia en su ejecución (José Ignacio Gómez Espínola, teoría Gestión de Proyectos Software, 2015). Como el proyecto es de tamaño medio e iterativo, pero tiene un fuerte carácter lineal, es recomendable usar alguna herramienta software para la planificación. En este caso, se ha instalado Redmine (la versión "todo en un click" de Bitnami) en el propio portátil usado para desarrollar.

En esta fase del proyecto se descompondrá el RBS en tareas estimables en tiempo y esfuerzo. Estas tareas deben tener una duración aceptable, es decir, que no superen el tiempo de una iteración. También se estimará la duración, los costes y los recursos necesarios para llevar a cabo el proyecto.

12.1. Estructura de Desglose del Trabajo

Para generar el WBS (Work Breakdown Structure) se ha usado el programa "Xmind" y se ha exportado la lista de tareas a Excel. Desde Excel se ha generado un fichero CSV añadiendo campos como duración estimada, asignación o prioridad y se ha importado a un nuevo proyecto en Redmine.

Para facilitar la lectura de las tareas se ha decidido no incluir el mapa WBS generado en Xmind. En su lugar se ha creado esta lista. Los elementos en color rojo son los elementos superiores del RBS. Los elementos verdes son las tareas del piso inferior (elementos hijo del árbol) que se estimarán.

Flame Knights Chronicles	
TAREAS	**TIEMPO ESTIMADO (Horas)**
1 Gestión de la Arquitectura de la Aplicación	
1.1 Diseño	1
1.2 Construcción del prototipo	0,5

1.3 Construcción Sala de Pruebas	0,5
2 Sistema Control del Jugador	
2.1 Avatar jugador	
2.1.1 Diseño	1
2.1.2 Implementación	1
2.2 Sistema Movimiento Jugador	
2.2.1 Diseño	1
2.2.2 Implementación	1
2.2.3 Pruebas de Unidad	0,5
2.3 Sistema Ataque Jugador	
2.3.1 Diseño	1
2.3.2 Implementación	1
2.3.3 Pruebas de Unidad	0,5
2.4 Pruebas de Integridad	1
3 Sistema Combate Jugador - Enemigos	
3.1 Enemigo Básico	
3.1.1 Diseño	1
3.1.2 Implementación	0,5
3.1.3 Implementación IA	1
3.1.3 Pruebas de Unidad	0,5
3.2 Sistema de Daño	
3.2.1 Diseño	1
3.2.2 Implementación	2
3.2.3 Pruebas de Unidad	0,5
3.3 Sistema Detección Jugador	
3.3.1 Diseño	0,5
3.3.2 Implementación	0,5
3.3.3 Pruebas de Unidad	0,5
3.4 Sistema de Esquiva de Ataques	
3.4.1 Diseño	2
3.4.2 Implementación	1
3.4.3 Pruebas de Unidad	1
3.5 Pruebas	
3.5.1 Pruebas de Integridad	2
3.5.2 Hito: Pruebas de Usuario	5
4 Sistema Cámara y minimapa	
4.1 Cámara	
4.1.1 Diseño	1
4.1.2 Implementación	1
4.1.3 Pruebas de Unidad	0,5
4.2 Minimapa	
4.2.1 Diseño	1
4.2.2 Implementación	2
4.2.3 Pruebas de Unidad	0,5

5 Sistema de Experiencia y Nivel	
5.1 Diseño	1
5.2 Implementación	2
5.3 Pruebas de Unidad	0,5
6 Sistema Características Personalizables	
6.1 Diseño	3
6.2 Implementación	5
6.3 Pruebas de Unidad	2
7 GUI	
7.1 Barras Vida, Maná y Experiencia	
7.1.1 Diseño	2
7.1.2 Implementación	3
7.1.3 Pruebas de Unidad	1
7.2 Nivel	
7.2.1 Diseño	0,5
7.2.2 Implementación	0,5
7.2.3 Pruebas de Unidad	0,5
7.3 Submenú Pausa	
7.3.1 Diseño	2
7.3.2 Implementación	1
7.3.3 Pruebas de Unidad	0,5
7.4 Submenú Abandonar Partida	
7.4.1 Diseño	1
7.4.2 Implementación	1
7.4.3 Pruebas de Unidad	0,5
7.5 Submenú Opciones	
7.5.1 Diseño	3
7.5.2 Implementación	2
7.5.3 Pruebas de Unidad	1
7.6 Submenú Características	
7.6.1 Diseño	3
7.6.2 Implementación	2
7.6.3 Pruebas de Unidad	1
7.7 Pruebas	
7.7.1 Pruebas de Integridad	3
7.7.2 Hito: Pruebas de Usuario	5
8 Sistema Habilidades Especiales jugador	
8.1 Gráficos	
8.1.1 Diseño	8
8.1.2 Implementación	6
8.2 Efectos - Ventajas	
8.2.1 Diseño	2
8.2.2 Implementación	8
8.2.3 Pruebas de Unidad	5

8.3 Sistema de Tiempo de Enfriamiento	
8.3.1 Diseño	3
8.3.2 Implementación	4
8.3.3 Pruebas de Unidad	1
8.4 Pruebas	
8.4.1 Pruebas de Integridad	4
8.4.2 Hito: Pruebas de Usuario	5
9 Enemigos y Jefes	
9.1 Enemigos	
9.1.1 Enemigos Cuerpo a Cuerpo (12)	
9.1.1.1 Diseño	16
9.1.1.2 Implementación	16
9.1.1.3 Pruebas de Unidad	8
9.1.2 Enemigos a Distancia (3)	
9.1.2.1 Diseño	6
9.1.2.2 Implementación	3
9.1.2.3 Implementación IA	1
9.1.2.4 Pruebas de Unidad	3
9.2 Jefes (6)	
9.2.1 Diseño	4
9.2.2 Implementación	4
9.2.3 Implementación IA	5
9.2.4 Pruebas de Unidad	3
9.3 Auras Jefes	
9.3.1 Diseño	1
9.3.2 Implementación	1
9.3.3 Pruebas de Unidad	0,5
9.4 Sistema de Generación de Items	
9.4.1 Diseño	1
9.4.2 Implementación	1
9.4.3 Pruebas de Unidad	0,5
9.5 Pruebas	
9.5.1 Pruebas de Integridad	12
9.5.2 Hito: Pruebas de Usuario	8
10 Sistema de Keybinds	
10.1 Diseño	1
10.2 Implementación	3
10.3 Pruebas de Unidad	1
11 Feedback Usuario	
11.1 Puntos de Característica sin Asignar	
11.1.1 Diseño	0,5
11.1.2 Implementación	0,5
11.1.3 Pruebas de Unidad	0,5

11.2 Números de Daño de Combate, Vida y Maná	
11.2.1 Diseño	0,5
11.2.2 Implementación	1,5
11.2.3 Pruebas de Unidad	0,5
11.3 Iconos del Minimapa	
11.3.1 Diseño	2
11.3.2 Implementación	3
11.3.3 Pruebas de Unidad	1
11.4 Transiciones entre Escenas	
11.4.1 Diseño	0,5
11.4.2 Implementación	0,5
11.4.3 Pruebas de Unidad	0,5
11.5 Subidas de Nivel	
11.5.1 Diseño	0,5
11.5.2 Implementación	0,5
11.5.3 Pruebas de Unidad	0,5
11.6 Pruebas	
11.6.1 Pruebas de Integridad	1
11.6.2 Hito: Pruebas de Usuario	3
12 Sistema misiones y Persistencia	
12.1 Controlador	
12.1.1 Diseño	1
12.1.2 Implementación	2
12.1.3 Pruebas de Unidad	1
12.2 Misiones	
12.2.1 Diseño	30
12.2.2 Implementación	90
12.2.3 Pruebas de Unidad	24
12.3 Persistencia	
12.3.1 Diseño	1
12.3.2 Implementación	2
12.3.3 Pruebas de Unidad	1
12.4 Pruebas	
12.4.1 Pruebas de Integridad	8
12.4.2 Hito: Pruebas de Usuario	25
13 Sonido	
13.1 Efectos de Sonido Jugador	
13.1.1 Diseño	1
13.1.2 Implementación	2
13.1.3 Pruebas de Unidad	0,5
13.2 Efectos de Sonido enemigos	
13.2.1 Diseño	6
13.2.2 Implementación	8
13.2.3 Pruebas de Unidad	4

13.3 Música Ambiental	
13.3.1 Diseño	12
13.3.2 Implementación	2
13.3.3 Pruebas de Unidad	3
13.4 Pruebas	
13.4.1 Pruebas de Integridad	5
13.4.2 Hito: Pruebas de Usuario	16
14 Cierre del proyecto	
14.1 Generación del Ejecutable	1
TIEMPO TOTAL:	489,5

Tabla 9 WBS

15 Demo	
15.1 Diseño	2
15.2 Implementación	10
15.3 Pruebas de Usuario	2

Tabla 10 Nuevo Grupo de Tareas

Estimación Resumida por Grupos de Tareas	
Grupo de Tareas	**Tiempo Estimado (Horas)**
1	2
2	8
3	19
4	6
5	3,5
6	10
7	33,5
8	46
9	94
10	5
11	17
12	185
13	59,5
14	1
TOTAL:	489,5

Tabla 11 Estimación Resumida por Grupo de Tareas

Como se puede observar, la duración del proyecto es excesiva a priori. Se ha decidido excluir del proceso de desarrollo los grupos 12 y 13 de tareas. Se introduce un nuevo grupo de tareas antes del Cierre, referentes a la creación de un nivel Demo (Grupo de tareas 15). Finalmente el proyecto queda estimado a **259 horas de trabajo**. Cabe remarcar que el proyecto ha sido planificado contando con sólo una persona en el equipo de trabajo. Si se hubiese dispuesto de un equipo completo, usando paralelización de tareas, el tiempo sería menor. **En este apartado no se refleja el tiempo dedicado a la creación de este documento**.

12.2. Estimación de la Duración

Se estima una duración de **3 meses**. El proyecto consta de 6 iteraciones, de una duración de dos semanas de duración cada una. Las tareas que comprende cada iteración se corresponden a los grupos de tareas definidos en el RBS (*Apartado 11.2*) y WBS (*Apartado 12.1*). La fecha de finalización será el 22 de abril de 2016. El tiempo restante hasta la entrega del proyecto será dedicado a la elaboración de este documento. Se estiman unas 90 horas de duración para este propósito.

TABLA DE ITERACIONES			
Iteraciones	**Tareas que comprende**	**Fecha Inicio**	**Duración Estimada (Horas)**
Iteración 1	1,2,3	01/02/2016	29
Iteración 2	4, 5, 6, 7	15/02/2016	53
Iteración 3	8	29/02/2016	46
Iteración 4	9.1	14/03/2016	53
Iteración 5	9.2, 9.3, 9.4, 9.5	28/03/2016	41
Iteración 6	10, 11, 15, 14	11/04/2016	37
		TOTAL:	259

Tabla 12 Iteraciones

12.3. Estimación de Recursos

Los recursos son los bienes o activos de una empresa que permiten la realización del proyecto. El recurso más importante son las personas, pero la empresa también debe disponer de infraestructuras tecnológicas y herramientas que hagan más fácil el trabajo, dando formas efectivas en tiempo y dinero de llevar a cabo las tareas necesarias.

12.3.1. Recursos Humanos

Si este proyecto fuese llevado a cabo por una empresa mediante un contrato con el cliente, el equipo de desarrollo mínimo necesario para afrontarlo exitosamente sería el siguiente.

- 1 Gestor de Proyectos
- 1 Diseñador Gráfico con experiencia en Photoshop
- 1 Modelador 3D con experiencia en Zbrush y Blender
- 1 Animador 3D con experiencia en Blender
- 1Programador con experiencia en Unity 3D
- 1 Diseñador de Videojuegos
- 1 Compositor, preferiblemente Ingeniero de Sonido

Como el proyecto actual sólo será llevado a cabo por una persona, se debe calcular el valor de su trabajo económicamente. Actualemente el sueldo de un programador de videojuegos ronda los 2.000 € brutos al mes, según el presidente de DEV (Asociación Española de Empresas Productoras y Desarrolladoras de Videojuegos), Ignacio Pérez (El País, 2014).

Esta mensualidad bruta debemos dividirla entre los 22 días laborables de media que tiene un mes y entre las 8 horas de la jornada laboral. El salario bruto a la hora es de 11,36 €.

Como nuestro proyecto está estimado a 259 horas, el coste de la mano de obra es **2942,24 €**.

12.3.2. Recursos Técnicos

Este apartado recoge las infraestructuras y software necesarios para llevar a cabo el proyecto. Los gastos se computan a lo largo de los 2 meses de duración en el caso de que el proyecto fuese real.

Infraestructuras

EQUIPO INFORMÁTICO	Precio de Adquisición	Cuota de uso (por hora)	Coste
Portátil MSI GE70	1.000,00 €	0,09 €	23,20 €
Tableta Digitalizadora Wacom Intuos Manga Pen & Touch S	87,00 €	0,01 €	2,02 €
		TOTAL:	**25,22 €**

Tabla 13 Precio Infraestructuras

Para cuantificar el coste del equipo informático necesario, se dividirá su precio de adquisición entre su vida útil (en horas) y se multiplicará por las horas totales estimadas de la vida del proyecto. Suponemos que estos activos tendrán un valor residual igual a cero y que su vida útil es de 5 años.

GASTOS SERVICIOS (2 MESES)		
Concepto	**Precio al mes**	**Coste**
Internet ADSL ONO 100Mb	60,00 €	120,00 €
Luz	70,00 €	140,00 €
Agua	12,00 €	24,00 €
Oficina en Jaén	150,00 €	300,00 €
	TOTAL:	**584,00 €**

Tabla 14 Gastos Fijos de Servicios

Estos son los gastos fijos por servicios de terceros de los que la empresa disfruta diario.

Software

En un caso real, se debe de disponer de mucho software dedicado a tareas concretas, como FL Studio, un software de edición de sonido y masterización. Aquí se va a contemplar el software que se usará realmente en nuestro caso.

GASTOS SOFTWARE (LICENCIAS)		
Concepto	**Razón**	**Precio (2 meses)**
Redmine	Gestión y Planificación del Proyecto	- €
Unity 3D 4.6.2	Framework para desarrollar el videojuego	- €
Adobe Photoshop CC	Edición/ Creación de Gráficos 2D y Texturas	24,18 €
Microsoft Office	Creación del Informe, creación de tablas...	14,00 €
Blender 2.76b	Modelado 3D y edición	- €
	TOTAL:	**38,18 €**

Tabla 15 Gastos Licencias Software

12.4. Estimación de Costes

En este apartado usamos al información anteriormente calculada para dar un presupuesto estimado. La primera tabla representa el resumen de gastos en el supuesto de tratarse de un proyecto real. El precio estimado es el total de gastos mas un 25% de beneficios. A estos beneficios brutos se les a aplicado el tipo reducido del impuesto de sociedades para emprendedores del 15%.

RESUMEN ESTIMACIÓN DE COSTE Y PRECIO FINAL	
Concepto	**Precio**
Mano de Obra	2.784,09 €
Equipo Informático	25,22 €
Servicios	584,00 €
Licencias Software	38,18 €
TOTAL GASTOS:	3.589,64 €
PRECIO ESTIMADO (+25% de Beneficios):	**4.487,05 €**
BENEFICIOS ANTES DE IMPUESTOS:	897,41 €
BENEFICIOS DESPUÉS DE IMPUESTOS:	**762,80 €**

Tabla 16 Resumen Costes y Precio Final

COSTES REALES	
Mano de Obra (Desarrollo)	2.059,57 €
Mano de Obra (Documentación)	715,68 €
Licencias Software	57,27 €
Licencia Camtasia Studio 8	259,00 €
TOTAL:	**3.091,52 €**

Tabla 17 Costes Reales del Proyecto

La tabla de costes reales refleja los gastos que deberá soportar por el alumno en la realización del proyecto. Nótese que la mano de obra es más barata, ya que no se deberán pagar deducciones. Las licencias son más caras, pues se tendrán que mantener 3 meses. También se incluye el precio de la licencia de Camtasia Studio 8 (259 €), el programa utilizado para grabar el video demostrativo.

Los gastos fijos existentes en la hipotética empresa aquí desaparecen ya que no están directamente relacionados con el proyecto. El equipo informático usado ya es propiedad del alumno antes de comenzar el proyecto, por lo que no se computará como gasto.

En este caso, se deben sumar las horas estimadas de documentación al coste total. Se estima que se dedicarán 90 horas, que sumadas a las de desarrollo el precio de la mano de obra asciende a **2.775,25** €.

12.5. Diagrama de Gantt

Como se ha mencionado en el apartado *12.2 Estimación de la Duración*, el proyecto ha sido planificado en 6 iteraciones con una duración de 2 semanas naturales cada una. A continuación reprentamos los diagramas de gantt de cada iteración. El progreso es lineal debido a que el equipo de desarrollo sólo consta de una persona. Las capturas de Redmine han sido tomadas al final de la fase de ejecución, de ahí que estén completas. Las tareas en el diagrama Gantt en Redmine se representan con una duración mínima de un día, por lo que este diagrama sólo refleja la precedencias entre tareas.

12.5.1. Gantt Iteración 1 (del 1-02-2016 al 14-02-2016) Versión 0.1

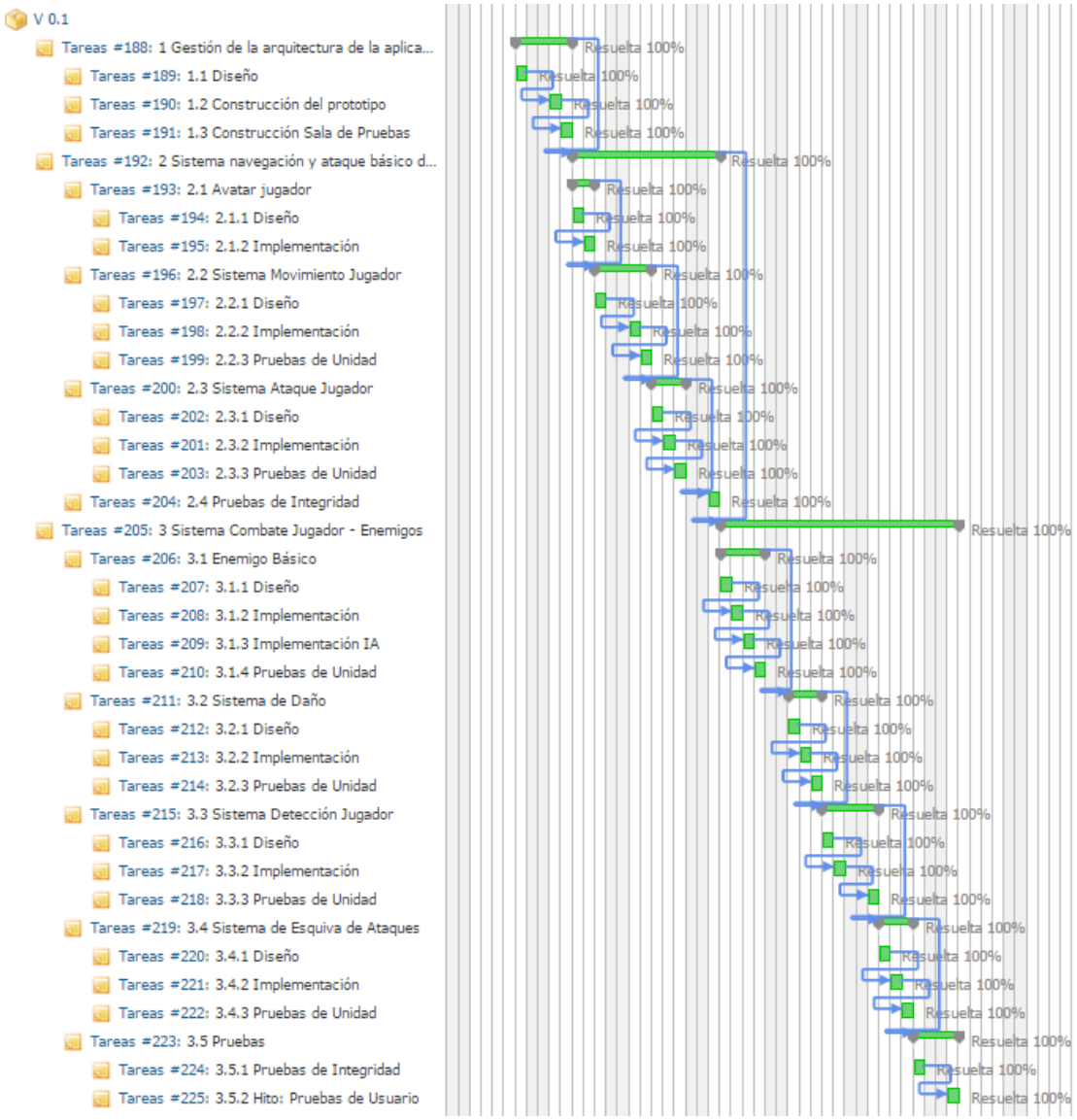


Ilustración 48 Gantt Iteración 1

12.5.2. Gantt Iteración 2 (del 15-02-2016 al 28-02-2016) Versión 0.2

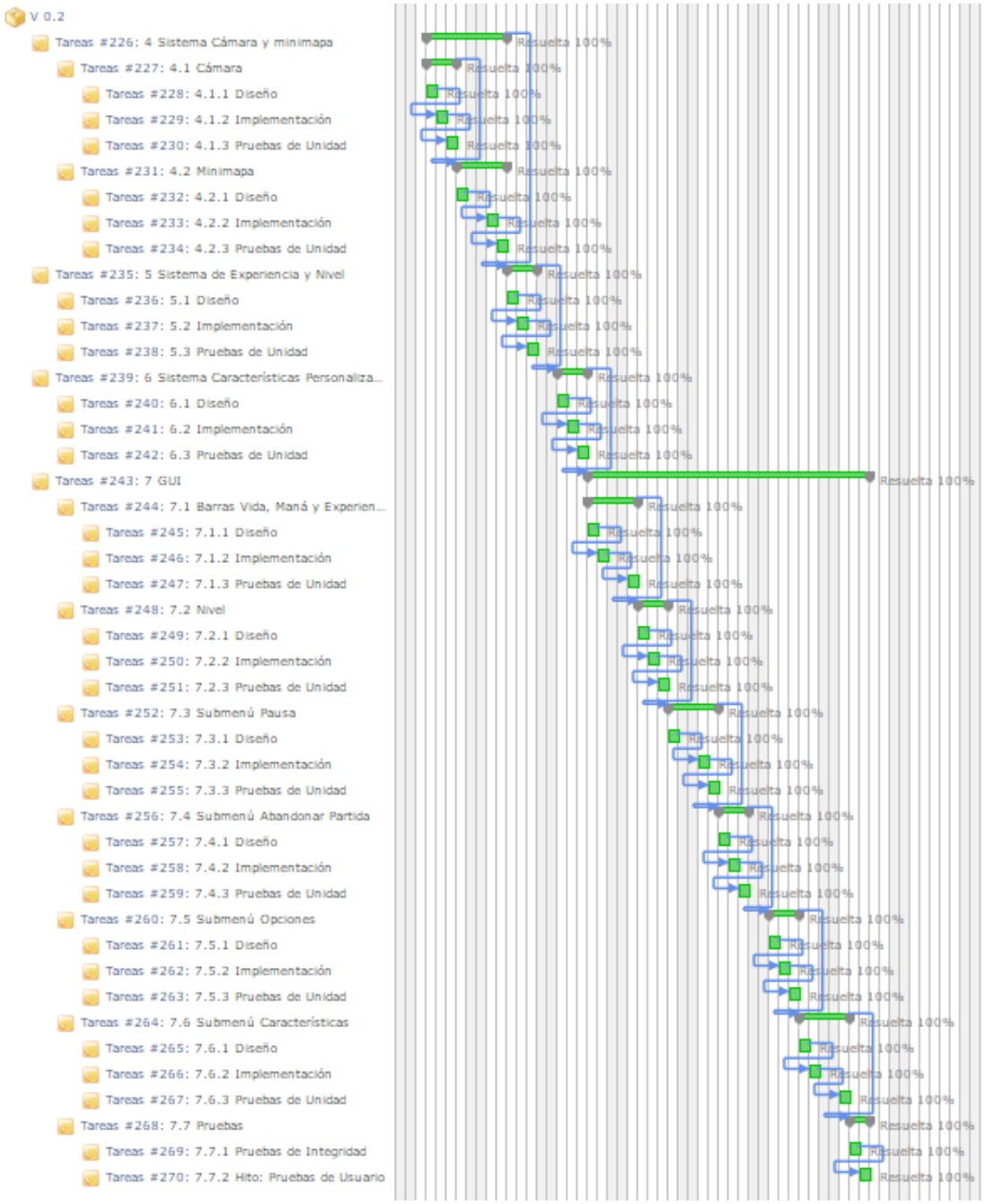


Ilustración 49 Gantt Iteración 2

12.5.3. Gantt Iteración 3 (del 29-02-2016 al 13-03-2016) Versión 0.3

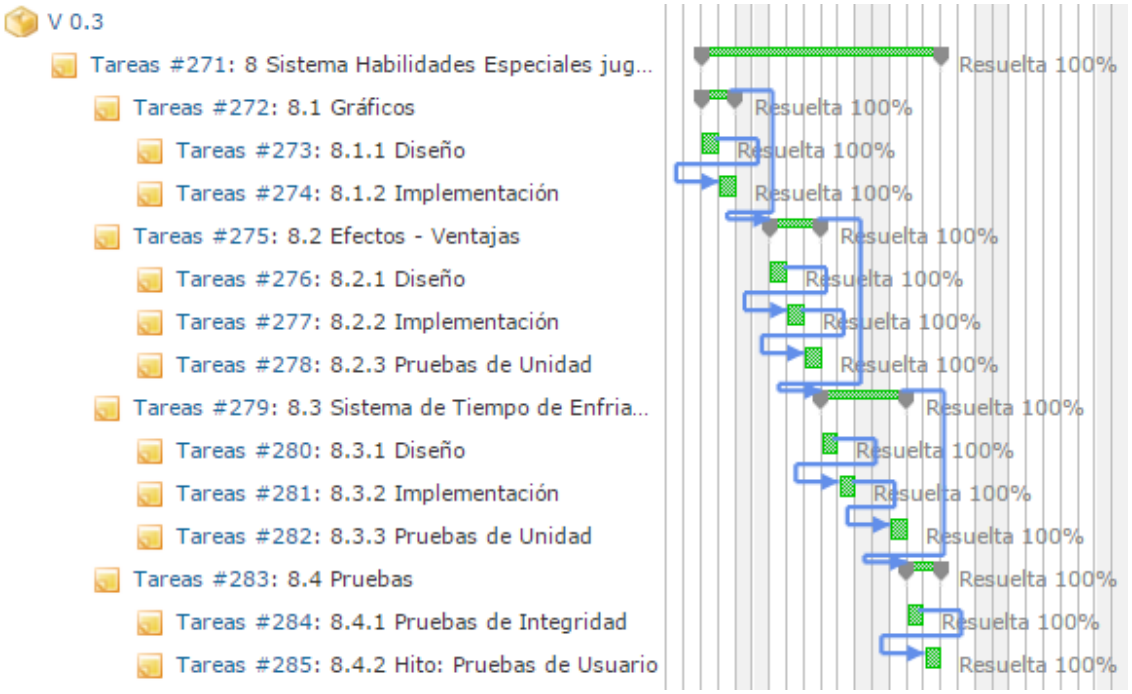


Ilustración 50 Gantt Iteración 3

12.5.4. Gantt Iteración 4 (del 14-03-2016 al 27-03-2016) Versión 0.4

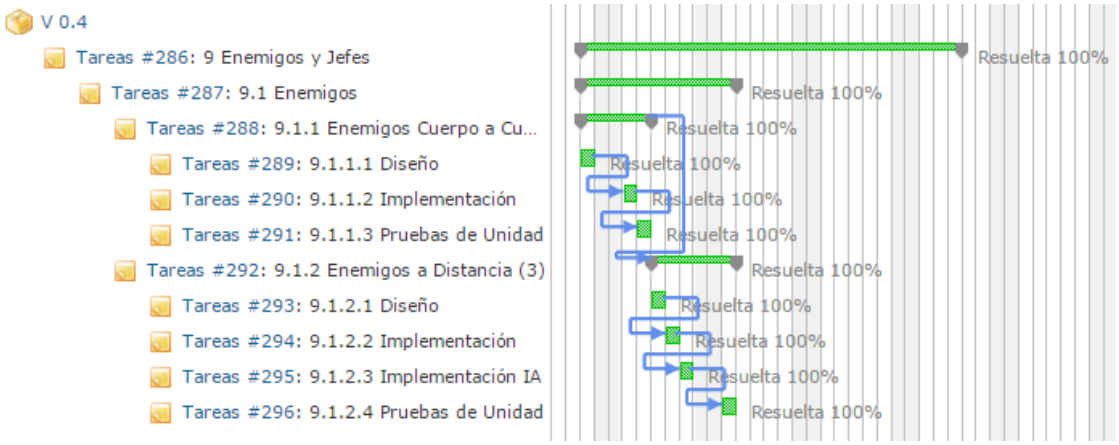


Ilustración 51 Gantt Iteración 4

12.5.5. Gantt Iteración 5 (del 28-03-2016 al 10-04-2016) Versión 0.5

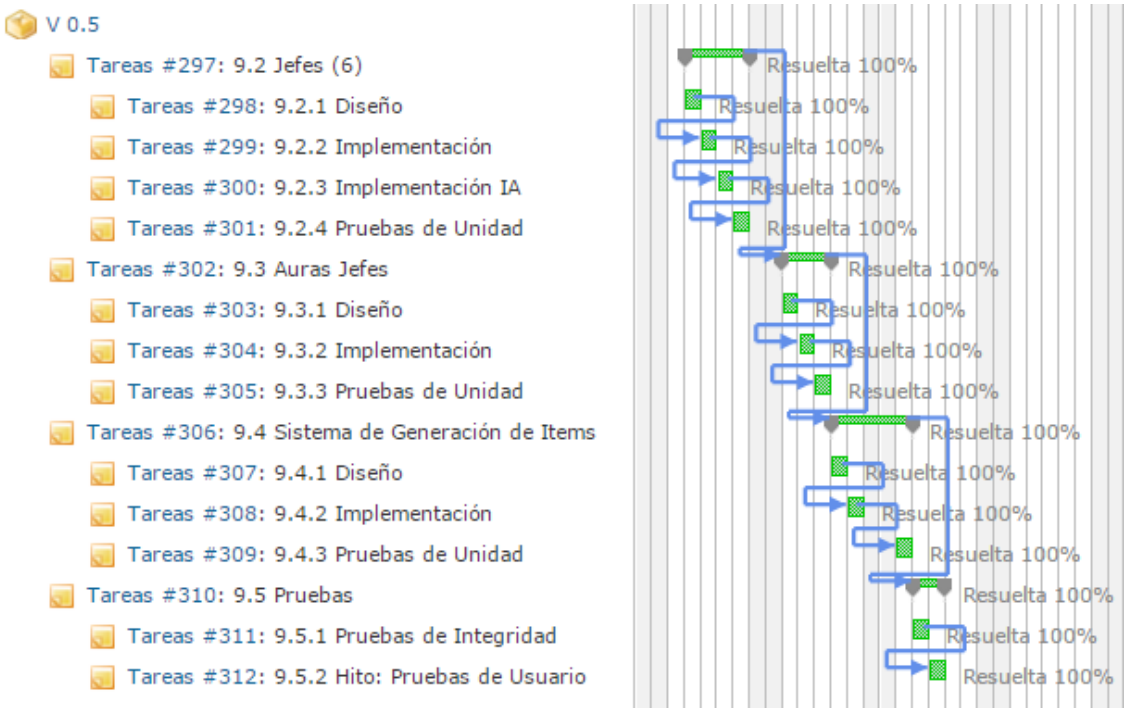


Ilustración 52 Gantt Iteración 5

12.5.6. Gantt Iteración 6 (del 11-04-2016 al 24-04-2016) Versión 1.0

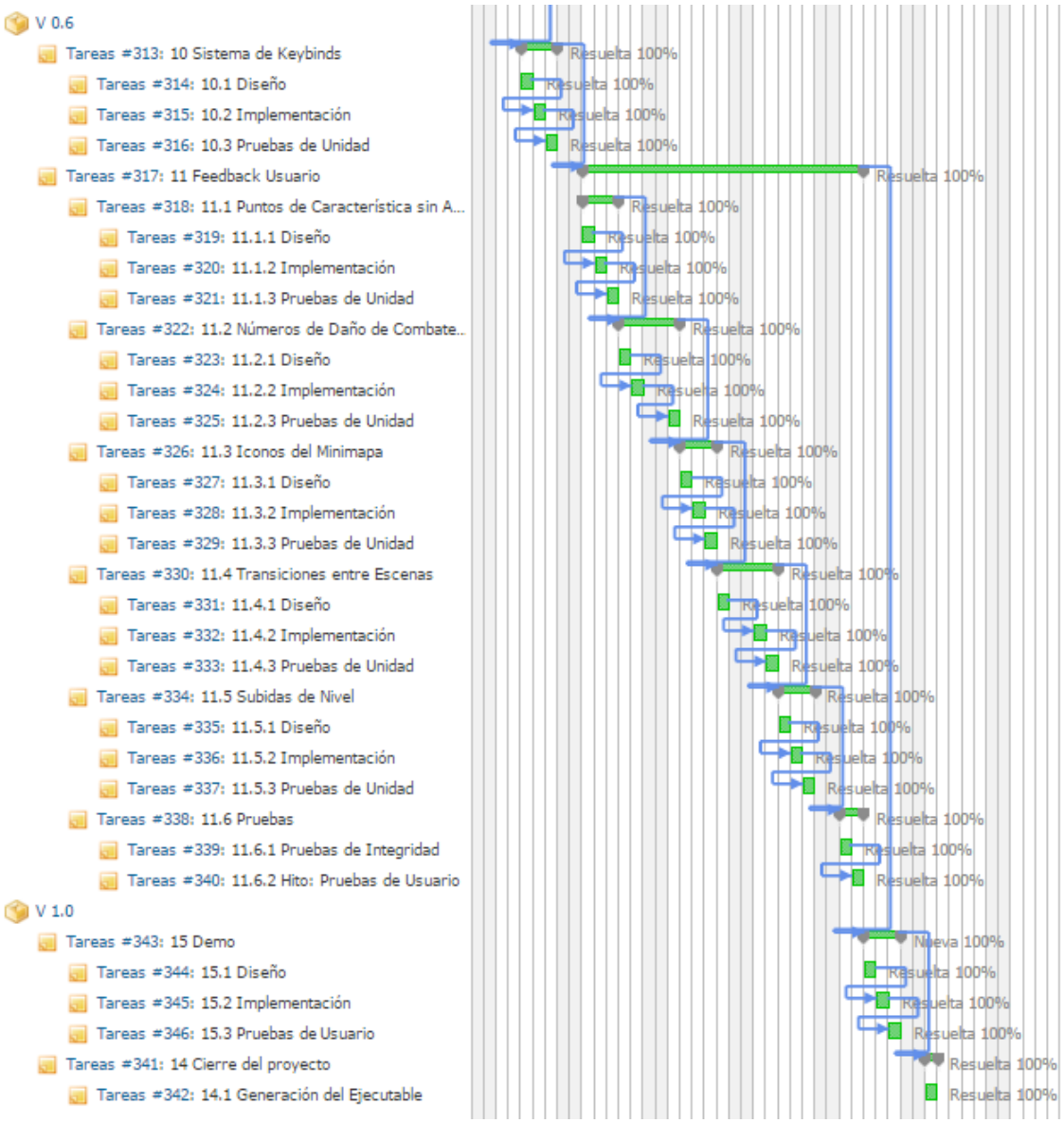


Ilustración 53 Gantt Iteración 6

Con esta última iteración se finalizará el desarrollo del prototipo, se creará el nivel demo, se generará el ejecutable que será entregado y se cerrará el proyecto.

13. EJECUCIÓN DEL PROYECTO

Una vez ha quedado lo suficientemente clara la fase de planificación, se empieza a trabajar en las iteraciones hasta completar el proyecto. En esta fase de ejecución se detallarán todas las tareas realizadas, los pasos necesarios para completarlas, los problemas encontrados y las pruebas realizadas. Se seguirá el típico esquema de actuación de tres pasos: diseño, implementación y prueba.

13.1. Iteración 1 (del 1-02-2016 al 14-02-2016)

Esta iteración inicial consta de 3 grupos de tareas. El primer grupo de tareas es común a la mayoría de los proyectos y engloba la preparación de los elementos base con los que se trabajará. El grupo de tareas 2 trata sobre la creación del personaje controlado por el jugador y su sistema de navegación. El grupo 3 engloba las tareas referentes a la creación de un enemigo para pruebas y al sistema de combate básico.

*13.1.1. **Grupo de Tareas 1**: Gestión de la Arquitectura de la Aplicación*

Grupo de tareas iniciales que consisten en creación de nuevos proyectos e infraestructuras sobre las que se basará toda la etapa de desarrollo.

13.1.1.1. Tarea 1.1: Diseño

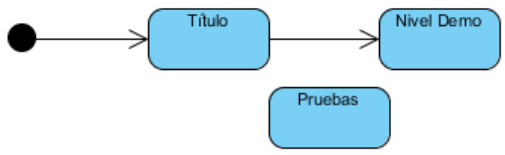


Ilustración 54 Escenas del Proyecto

El primer paso necesario será instalar Unity 3D. Los proyectos de este motor están divididos en escenas. Una pantalla de título, un nivel o los créditos pueden ser escenas diferentes. La mayor ventaja de trabajar con escenas es que la ejecución del juego requiere menos memoria al tener sólo una escena cargada a la vez. También resulta muy cómodo a la hora de trabajar, ya que diferentes miembros del equipo pueden crear o modificar diferentes escenas paralelamente. El proyecto constará de 3 escenas principales. La escena “Título” constará de la pantalla de bienvenida al iniciar el juego. La escena “Nivel Demo” será el mapa que el usuario podrá jugar. La

escena “Pruebas” será el laboratorio donde se implementarán y se probarán las funcionalidades que se vayan desarrollando.

13.1.1.2. Tarea 1.2: Construcción del prototipo

Una vez instalado Unity 3D, se crea un nuevo proyecto importando todos los paquetes disponibles por defecto y ajustando la configuración a 3D. Estos paquetes son utilidades que el motor trae por defecto. Se pueden descargar infinidad de paquetes creados por la comunidad desde la Unity Store para agregarlos a un proyecto. El hecho de ajustar la configuración a 3D es importante ya que prepara el editor para trabajar en 3D, ya que es diferente a la creación de un videojuego en 2D. Se crean las 3 escenas correspondientes y se dejan vacías por el momento.

Para poder trabajar cómodamente con los elementos que se vayan generando se ha usado el siguiente árbol de directorios.

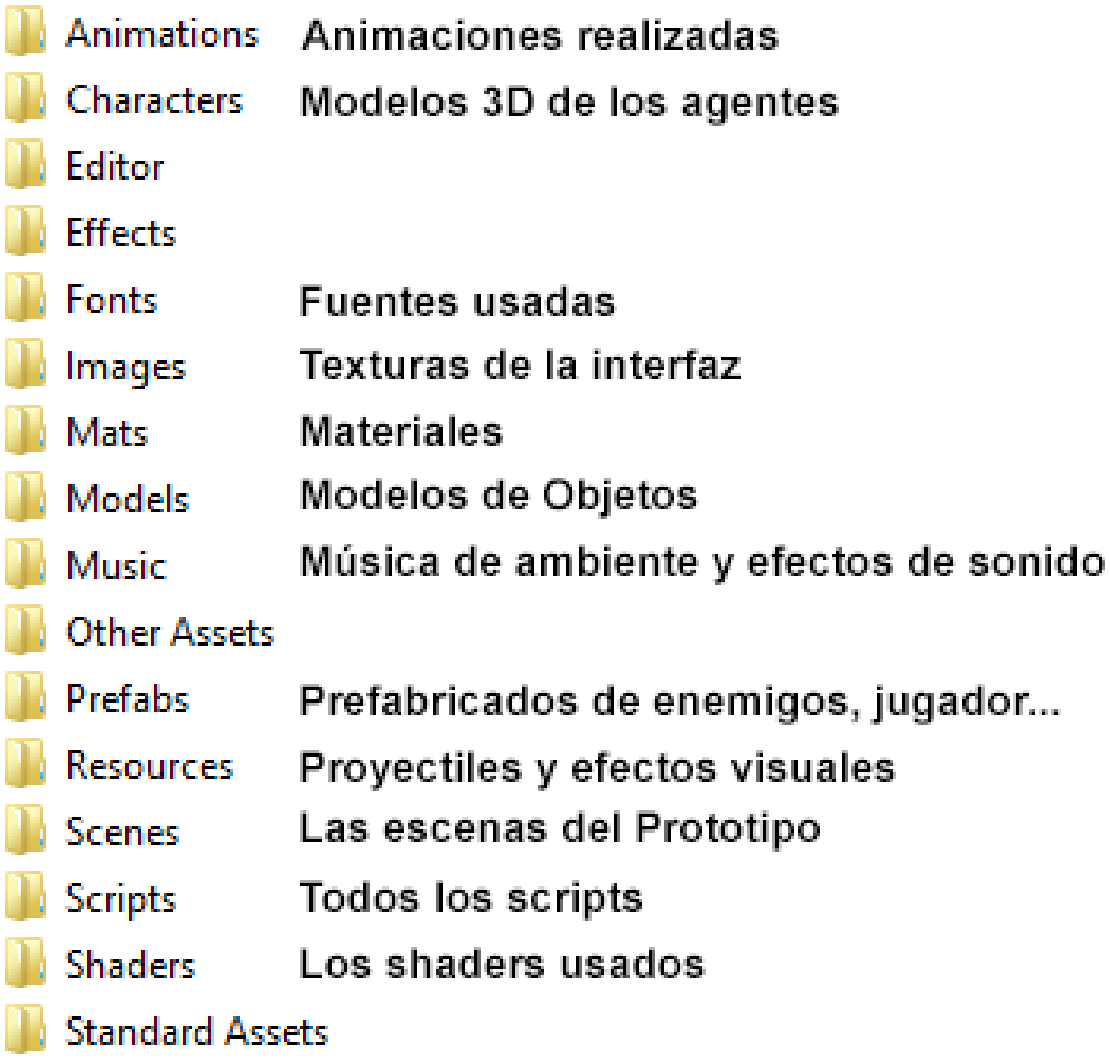


Ilustración 55 Organización de ficheros del Proyecto

13.1.1.3. Tarea 1.3: Construcción Sala de Pruebas

Abrimos la escena “Pruebas” y añadimos un objeto 3D llamado Terrain. Este objeto representa el terreno de juego. Unity 3D tiene unas cuantas herramientas para deformar esta malla, aplicar texturas y crear un terreno personalizado. Como la sala de pruebas es sólo para desarrollo se dejará el terreno plano y se aplicará una textura. No se modificarán los parámetros del terreno.

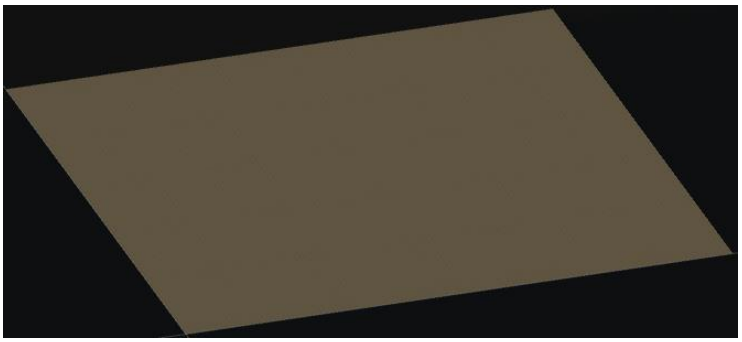


Ilustración 56 Sala de Pruebas

Ilustración 57 Textura Sala de Pruebas

13.1.2. **Grupo de Tareas 2**: Sistema Control del Jugador

En este grupo de tareas se define el personaje principal que controlará el usuario, el sistema de movimiento del mismo y el sistema de ataque.

13.1.2.1. Subgrupo de Tareas 2.1: Avatar jugador

13.1.2.1.1. Tarea 2.1.1: Diseño

Ilustración 58 Modelo Personaje Principal

El personaje principal será un humanoide que portará un arma para realizar sus ataques básicos. Se usará un modelo gratuito descargado de Unity Store (Animated Knight and Slime Monster). Este modelo ya viene con animaciones de ataque, correr, muerte y espera.

13.1.2.1.2. Tarea 2.1.2: Implementación

Se ha reemplazado el arma del modelo por otra más llamativa. Esta nueva espada ha sido descargada gratuítamente de TF3DM.com (Sword Medieval). Creamos en el proyecto una carpeta llamada “Prefabs”. En esta carpeta se incluirá el modelo modificado para instanciarlo cuando se desee. Este prefab es denominará “MyPlayer”.

Ilustración 60 Nuevo Modelo de Arma

Ilustración 59 Modelo Final del Personaje Principal

13.1.2.2. Subgrupo de Tareas 2.2: Sistema Movimiento Jugador

13.1.2.2.1. Tarea 2.2.1: Diseño

El movimiento del jugador se llevará a cabo a base de clicks sobre el mapa. Para este fin, usaremos las mallas de navegación. Esta herramienta automatiza las rutas de desplazamiento sobre un mapa en concreto, esquivando obstáculos.

Al capturar un click hecho sobre el mapa, el sistema moverá al personaje a la posición deseada.

13.1.2.2.2. Tarea 2.2.2: Implementación

Lo primero que se ha hecho ha sido añadir al jugador un componente llamado “Character Controller”, que define las colisiones del personaje con el mundo, y otro llamado “Nav Mesh Agent”, que controla la velocidad de desplazamiento, de giro y muchos más parámetros y es el encargado de generar y seguir las rutas de movimiento.

Otro paso necesario es hacer un “Bake” de la escena. Este paso consiste en precalcular la superficie transitable de la escena. Hay que tener en cuenta que si los obstáculos presentes en la escena están marcados como “Static” el cómputo de las rutas será mucho más óptimo. Si los elementos estáticos de la escena cambian o el terreno es modificado se deberá volver a realizar un bake.

Finalmente se ha creado un script en C# llamado “ClickToMove” que controla el evento de pulsación del botón izquierdo del ratón sobre el terreno. Cuando este evento ocurre, y la ruta es válida (es un punto alcanzable) se ejecuta la animación de andar hasta llegar al destino. El personaje siempre está ejecutando la animación de espera hasta que se deba ejecutar otra. En la cámara principal del proyecto se ha añadido un script llamado “Fixed Follow” que modifica la posición de la cámara con respecto al jugador.

13.1.2.2.3. Tarea 2.2.3: Pruebas de Unidad

Se han realizado pruebas sobre el sistema de movimiento. Se han realizado varios clicks seguidos en diferentes partes del terreno para comprobar satisfactoriamente que las rutas se van sustituyendo correctamente. Se ha ajustado la velocidad de desplazamiento y en la ejecución de animaciones para que el movimiento sea fluido y coherente.

13.1.2.3. Subgrupo de Tareas 2.3: Sistema Ataque Jugador

En este grupo de tareas se implementará el sistema de ataque del personaje principal.

13.1.2.3.1. Tarea 2.3.1: Diseño

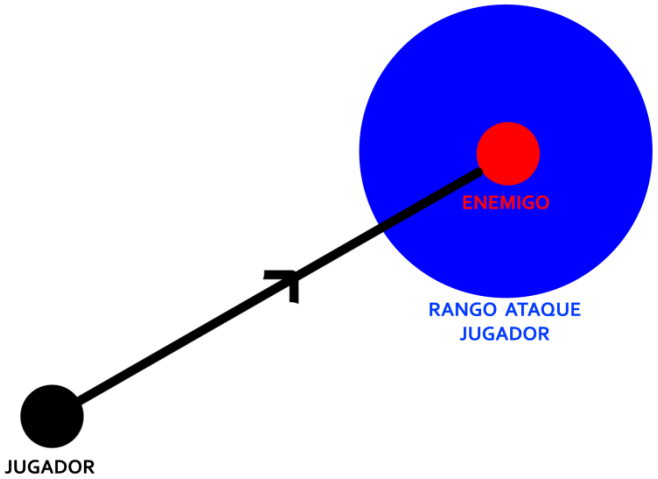


Ilustración 61 Rango Ataque Jugador

El jugador pulsará el botón derecho del ratón sobre un enemigo. El personaje principal se desplazará hasta estar a un rango aceptable del enemigo y lo atacará. El rango del personaje principal deberá ser coherente con el arma que porta. Si ya se está en ese rango, se ejecutará el ataque básico. Se usará un cilindro con un "Capsule collider" para realizar las pruebas.

13.1.2.3.2. Tarea 2.3.2: Implementación

Se ha creado un script llamado "Combat" añadido al personaje principal. Este script controla que cuando el usuario pulse el botón derecho del ratón sobre un enemigo si este está a rango (3 unidades) se ejecute la animación de ataque. Si no es está a rango, el personaje principal se desplazará al punto más cercano del mapa en el que lo esté.

Este desplazamiento se realiza a través del componente “NavMeshAgent”. Cuando se ejecuta la animación de ataque, en el momento justo en el que el arma es usada se invoca el método “GetHit” del script del enemigo llamado “Mob” que tendrá el daño como parámetro.

13.1.2.3.3. Tarea 2.3.3: Pruebas de Unidad

Se han realizado pruebas desde diferentes puntos del mapa de pruebas comprobando que el personaje principal se acerca al enemigo, realiza la animación de ataque y daña al enemigo. Este último hecho se ha comprobado por la consola, mostrando un mesaje cuando el enemigo recibe daño.

13.1.2.4. Tarea 2.4: Pruebas de Integridad

Se han realizado pruebas conjuntas del sistema de movimiento y el sistema de ataque. Se ha movido al personaje principal y mientras se efectuaba la ruta se ha atacado al enemigo y viceversa. Las pruebas son satisfactorias y el sistema en conjunto trabaja correctamente.

13.1.3. ***Grupo de Tareas 3***: Sistema Combate Jugador – Enemigos

En este grupo de tareas se diseñará e implementará el primer enemigo del juego. Se engloban tareas como la implementación de la inteligencia artificial (IA) y el rango de detección del jugador.

13.1.3.1. Subgrupo de Tareas 3.1: Enemigo Básico

Este subgrupo de tareas generará el modelo y la inteligencia artificial de un enemigo básico.

13.1.3.1.1. Tarea 3.1.1: Diseño

Ilustración 62 Modelo de Primer Enemigo

Ante la incapacidad por temas de tiempo de generar un modelo 3D propio, crear las texturas correspondientes, hacer un rigging (definir los huesos para ejecutar las animaciones) y crear las animaciones, se ha optado por usar un modelo gratuíto existente en la Unity Store (Skeletons Pack).

El enemigo se comportará de la siguiente forma. Una vez que el jugador entre en el rango de detección del enemigo, el enemigo lo perseguirá y atacará si está a rango, hasta que el jugador o él mismo muera. Es un comportamiento muy básico, pero es suficiente para avanzar en el proceso de desarrollo.

13.1.3.1.2. Tarea 3.1.2: Implementación

En la carpeta “Prefabs” se ha creado una subcarpeta llamada “Enemies”, la cual contendrá a los enemigos generados en el futuro. El enemigo ha sido creado con el nombre de “Skeleton Lancer”.

Lo primero que se ha hecho ha sido añadir al enemigo un componente llamado “Character Controller”, que define las colisiones del personaje con el mundo, y otro llamado “Nav Mesh Agent”, que controla la velocidad de desplazamiento, de giro y muchos más parámetros y es el encargado de generar y seguir las rutas de movimiento.

Finalmente se ha creado un script en C# llamado “Mob” que controla el movimiento del enemigo. Cuando el enemigo se mueve hacia el personaje se ejecuta la animación de andar hasta llegar al destino. El enemigo siempre está ejecutando la animación de espera hasta que se deba ejecutar otra.

13.1.3.1.3. Tarea 3.1.3: Implementación IA

Se ha modificado el script en C# llamado “Mob” para que, a partir de una referencia al personaje principal, el enemigo persiga al jugador sin descanso y lo ataque si está dentro de un rango concreto. Este rango será único de cada futuro enemigo. Se ha incluído un atributo de vida en el enemigo. Cuando la vida del enemigo llegue a cero, se ejecutará la animación de muerte y se destruirá el objeto que lo representa.

13.1.3.1.4. Tarea 3.1.3: Pruebas de Unidad

Se ha comprobado que el enemigo persigue al jugador y realiza ataques al estar cerca de él. Si el jugador se separa del enemigo, este vuelve a perseguirlo hasta que lo alcanza y vuelve a ejecutar ataques cuerpo a cuerpo.

13.1.3.2. Subgrupo de Tareas 3.2: Sistema de Daño

Se define y se implementa el sistema de daño de los enemigos al jugador en este grupo de tareas.

13.1.3.2.1. Tarea 3.2.1: Diseño

El diseño es bastante simple. Cuando un enemigo ejecute un ataque hacia el personaje, se llamará a un método del script “Combat” del jugador que restará de la vida actual el daño causado. Se creará una barra de vida básica para mostrar el daño al usuario.

13.1.3.2.2. Tarea 3.2.2: Implementación

Se define la variable de vida en el script “Combat” del personaje principal. Al iniciar un nivel, el personaje tendrá los puntos de vida completos. El daño causado por los enemigos irá decrementando este valor. El método que invocarán los enemigos será “GetHit”. Este método tiene como parámetro único el daño causado. Cada enemigo deberá tener un atributo con el daño que inflinje al atacar en su script “Mob”. Cuando la vida del jugador llegue a cero, el personaje principal ejecutará la animación de muerte y se reiniciará el nivel.

Se ha añadido a la escena un objeto llamado “Canvas” que representa el plano 2D de la pantalla. Este objeto servirá para crear la GUI en el futuro. Se ha añadido una barra de vida del enemigo que se mostrará cuando el jugador haga click derecho o izquierdo sobre un enemigo. Después de un tiempo sin atacar al enemigo se volverá a ocultar. Esta funcionalidad se implementa en el script “Combat” del personaje principal. La cantidad de vida se actualizará haciendo una interpolación lineal, para que el cambio del valor sea suavizado. Se usará una textura que hemos descargado de opengameart.org (loading-bar, StumpyStrust).

Ilustración 63 Barra de Vida del Enemigo

13.1.3.2.3. Tarea 3.2.3: Pruebas de Unidad

Se han instanciado 2 enemigos básicos “Skeleton Lancer” y se han hecho pruebas sobre el daño y la cadencia de ataque para sincronizar el momento en el que se hace el daño y la animación de ataque. Las pruebas han sido satisfactorias y el comportamiento del enemigo es correcto.

13.1.3.3. Subgrupo de Tareas 3.3: Sistema Detección Jugador

Un sistema de detección del jugador es necesario para evitar que, al comenzar el nivel, todos los enemigos comiencen a perseguir al jugador.

13.1.3.3.1. Tarea 3.3.1: Diseño

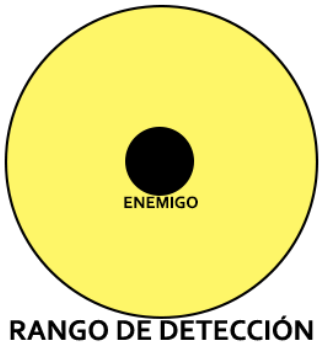


Ilustración 64 Rango de Detección del Enemigo

El rango de detección se ha diseñado como un rango de visión de 360 grados. En la siguiente imagen se puede apreciar una vista aérea de un enemigo y su rango de detección alrededor. Si el usuario mueve al personaje principal por el rango descrito, este enemigo comenzará a perseguirlo y atacarlo hasta que el personaje principal o el propio enemigo muera.

13.1.3.3.2. Tarea 3.3.2: Implementación

La implementación de esta característica se ha llevado a cabo comprobando cada cierto tiempo en la función “Update” del script “Mob” la distancia del jugador al enemigo. Se ha añadido un atributo que representa el rango de visión en el script “Mob”. Si la distancia entre el personaje y el enemigo es inferior o igual al rango de visión, el enemigo comenzará a perseguirlo.

13.1.3.3.3. Tarea 3.3.3: Pruebas de Unidad

El rango de visión se ha ido ajustando poco a poco haciendo pruebas. Se ha usado un rango de visión de entre 10 y 15 unidades para enemigos cuerpo a cuerpo. Las pruebas han consistido en instanciar copias del enemigo básico a diferentes distancias hasta dar con un rango aceptable.

13.1.3.4. Subgrupo de Tareas 3.4: Sistema de Esquiva de Ataques

Se ha querido hacer más interesante el sistema de combate básico añadiendo la posibilidad de esquivar los ataques físicos que los enemigos ejecutan contra el jugador.

13.1.3.4.1. Tarea 3.4.1: Diseño

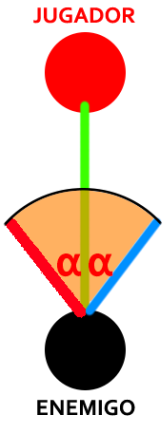


Ilustración 65 Zona de Daño del Enemigo

El jugador perderá vida si se encuentra en una zona de daño. Definiendo el vector que va desde el enemigo al jugador (línea verde), si disponemos del rango de ataque sabemos hasta donde llegará el ataque. Si añadimos un ángulo de ataque (α), podemos definir un cono que representará la zona de daño. La esquiva puede realizarse porque cuando un enemigo ataca no corrige su posición hacia el jugador como lo hace cuando lo está persiguiendo.

13.1.3.4.2. Tarea 3.4.2: Implementación

Simplemente se comprobará que el jugador esté en esta área cuando se vaya a computar el daño. Si no se está dentro de la zona el jugador no recibirá daño.

13.1.3.4.3. Tarea 3.4.3: Pruebas de Unidad

Se han realizado pruebas con varios ángulos de ataque en el enemigo básico. El ángulo usado finalmente ha sido de 30 grados, lo que hace un área de 60 grados en frente del enemigo. Se puede esquivar con relativa facilidad.

13.1.3.5. Subgrupo de Tareas 3.5: Pruebas

Este grupo de tareas tiene como objetivo probar todo lo desarrollado en el grupo 3. Se probarán conjuntamente el enemigo, su inteligencia artificial, su sistema de detección y el sistema de esquiva. Las pruebas de integridad han sido llevadas a cabo por el desarrollador y las pruebas de usuario por 2 “testers”.

13.1.3.5.1. Tarea 3.5.1: Pruebas de Integridad

Se han instanciado unos 5 enemigos en el mapa de pruebas y se ha combatido con ellos. Esta situación se ha recreado esta situación 7 veces y tanto el comportamento de los enemigos como los controles del jugador son correctos.

13.1.3.5.2. Tarea 3.5.2: Hito: Pruebas de Usuario

CCG (estudiante del grado de Ingeniería Informática) y JRC (estudiante del grado de Ingeniería Mecánica) han hecho el trabajo de testeo. Se ha generado un ejecutable para que puedan probar las funcionalidades desarrolladas y responder a un cuestionario. También se pone a su disposición un formulario para que tomen nota de posibles condiciones anómalas que puedan detectar.

A continuación se presenta el cuestionario que ha sido diseñado para obtener las impresiones de los testeadores y los resultados obtenidos. Se ha usado una escala del uno al cinco, donde cinco es “Sí, mucho” y uno es “No, nada”.

Cuestionario de Pruebas de Usuario - 1ª Iteración, Tarea 3.5.2			
Nº Pregunta	**Pregunta**	**CCG**	**JRC**
1	¿El movimiento del jugador es fluido?	5	5
2	¿El ataque del jugador es fluido?	4	5
3	¿Le parece correcto el rango de detección de los enemigos?	3	4
4	¿Le ha resultado fácil esquivar los ataques de los enemigos?	2	3
5	¿Le parece correcto el rango de ataque del jugador?	5	5

Tabla 18 Cuestionario de Pruebas de Usuario - 1ª Iteración, Tarea 3.5.2

Como se puede comprobar en la tabla, todas las respuestas han sido satisfactorias. La pregunta número 4 es la que ha tenido los resultados más bajos, pero esto es bueno ya que no implementamos el sistema de esquiva pensando en poder evitar todos los ataques fácilmente.

No se han detectado condiciones anómalas por parte de los testeadores.

13.2. Iteración 2 (del 15-02-2016 al 28-02-2016)

Esta segunda iteración comprende 4 grupos de tareas. En el primer grupo se desarrollará el sistema de la cámara principal y un minimapa que ayudará al jugador a orientarse por los niveles. El segundo grupo engloba el sistema de experiencia y nivel. En el tercer grupo se desarrolla el sistema de características personalizables y finalmente en el cuarto grupo se desarrollará la interfaz de usuario.

*13.2.1. **Grupo de Tareas 4**: Sistema Cámara y minimapa*

13.2.1.1. Subgrupo de Tareas 4.1: Cámara

13.2.1.1.1. Tarea 4.1.1: Diseño

La cámara principal estará dirigida al jugador desde una posición cenital. Ya definimos en un paso anterior una cámara para las pruebas que seguía al jugador.

13.2.1.1.2. Tarea 4.1.2: Implementación

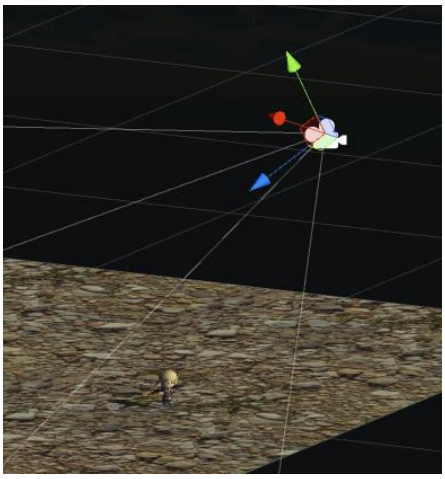


Ilustración 66 Cámara Principal

Se han incluído en el script “Fixed Follow” los atributos de distancia, altura y oscilación para mejorar el control de la cámara. No se han modificado otros atributos de la cámara como la proyección o los planos de corte.

13.2.1.1.3. Tarea 4.1.3: Pruebas de Unidad

Se ha probado el seguimiento del jugador por parte de la cámara por la sala de pruebas y se han ajustado los parámetros previamente citados hasta conseguir posicionar la cámara con una inclinación y distancia correctas.

13.2.1.2. Subgrupo de Tareas 4.2: Minimapa

13.2.1.2.1. Tarea 4.2.1: Diseño

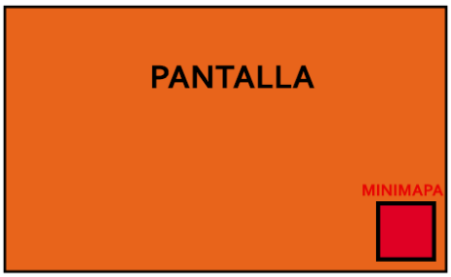


Ilustración 67 Diseño Minimapa

Se implementará un minimapa en la parte derecha inferior de la pantalla. Este minimapa mostrará una vista aérea de la zona cercana al personaje. Es importante que el minimapa no sea intrusivo.

13.2.1.2.2. Tarea 4.2.2: Implementación

Ilustración 68 Minimapa con Marco

Con una cámara cenital secundaria orientada al jugador, modificando las medidas y la posición del "Viewport" se ha implementado el minimapa. Se han creado 2 scripts, "Follow Player" para que la cámara se centre en el personaje siempre desde un punto aéreo y otro script ("Aspect Ratio") para que se adapte a cualquier pantalla y se dibuje un marco superficial al minimapa.

Este marco ha sido seleccionado de un pack de texturas gratuítas descargado de opengameart.org (ui-resources, Wyrmheart) que se usará para desarrollar la interfaz de usuario y los menús.

13.2.1.2.3. Tarea 4.2.3: Pruebas de Unidad

Una vez implementado el minimapa se ha comprobado que el comportamiento es el esperado. Cabe resaltar que este minimapa no es de mucha ayuda, ya que no se distinguen bien ni el jugador ni los enemigos. En fases posteriores se diseñarán iconos de minimapa para los agentes del juego.

13.2.2. **Grupo de Tareas 5**: Sistema de Experiencia y Nivel

En este grupo de tareas se diseñará e implementará el sistema de experiencia y nivel.

13.2.2.1. Tarea 5.1: Diseño

Un nuevo nivel otorgará un punto de característica. Estos puntos serán definidos e implementados más tarde. Al morir un enemigo, éste otorgará experiencia. Cada nivel necesitará más experiencia que el anterior. Para más detalles véase el apartado *7.9 Experiencia y Nivel*.

13.2.2.2. Tarea 5.2: Implementación

Se ha añadido la experiencia y el nivel como atributos del script “Combat” del personaje. También se ha añadido un método llamado “addExp”. Este método será invocado por un enemigo antes de morir y añadirá experiencia al personaje principal. Se ha modificado el script “Mob” del enemigo básico añadiendo un atributo que contendrá la experiencia que dicho enemigo otorgará al morir. En la función FixedUpdate del script “Combat” se comprobará si se ha llegado a la cantidad de experiencia necesaria y se subirá del nivel.

13.2.2.3. Tarea 5.3: Pruebas de Unidad

Se han ralizado pruebas con el enemigo básico. El enemigo otorgará 200 puntos de experiencia. El nivel dos es alcanzado si se acumulan 129 puntos. El nivel tres con 224. Al principio, subir de nivel es más fácil. Conforme se van ganando niveles, los puntos de experiencia requeridos son más altos. El sistema funciona correctamente.

13.2.3. **Grupo de Tareas 6**: Sistema Características Personalizables

Estas tareas engloban el diseño, creación y prueba del sistema de características personalizables. Este sistema otorga un fuerte componente de rol y estrategia, lo que permite al usuario personalizar ampliamente al personaje principal, eligiendo entre múltiples estilos de juego.

13.2.3.1. Tarea 6.1: Diseño

El usuario podrá personalizar al personaje principal a través de 5 atributos. Estos atributos son Fuerza, Inteligencia, Vitalidad, Velocidad de Ataque y Velocidad de Movimiento. Se puede ver toda la información detallada de este sistema en el apartado *7.8 Estadísticas y Sistema de puntos de Característica*.

13.2.3.2. Tarea 6.2: Implementación

Se han agregado atributos para los valores base, para los puntos de característica y para los efectos por punto en el script “Combat” del personaje principal. En la función “Awake” de dicho script se calculan las estadísticas iniciales. Esta función sólo se ejecuta una vez. Se ha creado un método público llamado “UpdateStats” que será invocado cuando se reinicien las puntos o se distribuyan desde el futuro menús de características. También se han creado dos co-rutinas en el mismo script que irán computando la regeneración de vida y maná que el jugador pueda tener. Las co-rutinas son funciones ejecutadas en hilos independientes al hilo del script, que pueden ser programadas para ejecutarse cada cierto tiempo. En este caso, como la regeneración de vida o maná se aplica cada segundo, el hilo de las co-rutinas se ejecutará, dormirá un segundo y volverá a ejecutarse infinitamente.

13.2.3.3. Tarea 6.3: Pruebas de Unidad

Para las pruebas, se ha incluído en la función “FixedUpdate” del script “Combat” una llamada a “UpdateStats” y se han modificado los puntos de característica directamente desde el inspector de componentes ya que no disponemos aún de una interfaz dentro del juego para este fin. Los valores de la tabla presente en la tarea de diseño han sido ajustados y obtenidos en esta fase, logrando evitar comportamientos anómalos como velocidades de movimiento y ataque excesivas o regeneraciones de vida y maná demasiado potentes.

13.2.4. ***Grupo de Tareas 7**: GUI*

13.2.4.1. *Subgrupo de Tareas 7.1: Barras Vida, Maná y Experiencia*

En este primer subgrupo será necesario crear el esqueleto de la interfaz de usuario, ya que las barras estarán incrustadas en ella.

13.2.4.1.1. *Tarea 7.1.1: Diseño*

Se ha creado en Photoshop un boceto de la interfaz de usuario principal. La interfaz estará posicionada en la parte inferior izquierda de la pantalla.

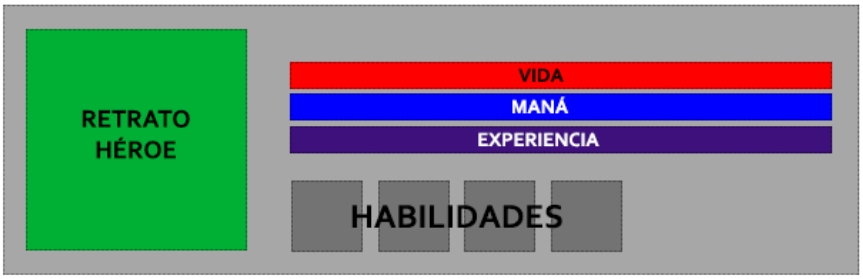


Ilustración 69 Boceto de la Interfaz Principal

Basándonos en este prototipo, se ha creado el fondo de la interfaz, las barras de vida, maná y experiencia y un fondo común para todas.

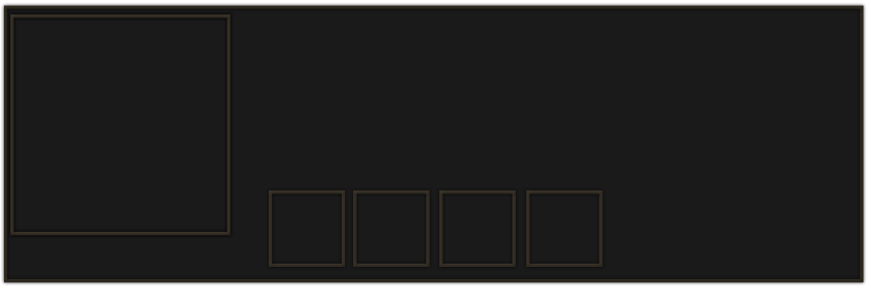


Ilustración 70 Fondo de la interfaz Principal

Ilustración 71 Fondo y Barras de Vida, Maná y Experiencia

Ilustración 72 Retrato del Héroe

Se ha usado una imagen que servirá como retrato del héroe (personaje principal) para el prototipo. Esta imagen pertenece a NCSOFT en su juego Guild Wars.

13.2.4.1.2. Tarea 7.1.2: Implementación

Se ha añadido al objeto “Canvas” de la escena todos los gráficos anteriores, dando como resultado la siguiente imagen. También se han añadido objetos de interfaz de usuario “Text” para representar los valores actuales y máximos de vida, maná y experiencia. Se han creado 3 scripts (uno por cada barra), que actualizarán los valores actuales y máximos de cada barra, usando interpolación lineal. Estos scripts son “HealthBarPlayer”, “ManaBarPlayer” y “ExpBarPlayer” y poseen una referencia al script “Combat” del personaje principal. Se usará el patrón arquitectónico Modelo-Vista-Controlador (*15.8*) para realizar las actualizaciones de la interfaz y el acceso de información del modelo de datos (Agentes y sistema).

Ilustración 73 Interfaz Principal de Usuario dentro del juego

13.2.4.1.3. Tarea 7.1.3: Pruebas de Unidad

Exceptuando la barra de maná, las restantes han sido probadas combatiendo con 2 enemigos básicos. Las barras de vida actualizan los cambios correctamente. La barra de maná ha sido probada cambiando el atributo de maná directamente desde el inspector de componentes y añadiendo regeneración de maná al personaje principal. Su funcionamiento también es correcto.

13.2.4.2. Subgrupo de Tareas 7.2: Nivel

13.2.4.2.1. Tarea 7.2.1: Diseño

Al boceto inicial se le ha añadido la zona en la que aparecerá el nivel del personaje. Se va a utilizar para el nivel y los menús una fuente llamada “DARK 11” descargada de fonts4free.net.

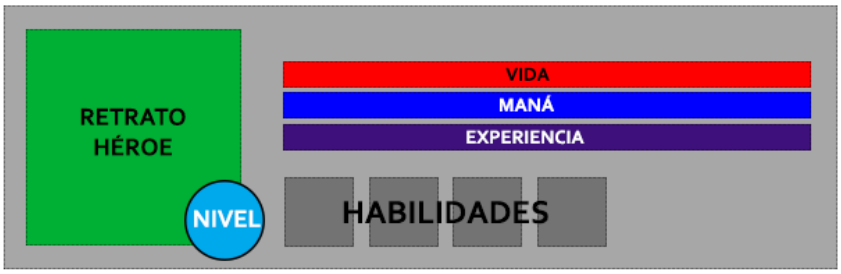


Ilustración 74 Boceto Interfaz de Usuario con Nivel

13.2.4.2.2. Tarea 7.2.2: Implementación

Se usará la siguiente imagen superpuesta a la interfaz de usuario actual. Se han añadido unos marcos meremente estéticos para adornar un poco la interfaz. Estos marcos han sido obtenidos del pack de texturas gratuíto descargado de opengameart.org (ui-resources, Wyrmheart) que se va a usar para los menús. El nivel será representado por un objeto de interfaz Text. Se ha creado un script en el objeto “Canvas” llamado “GUIController” que actualizará toda la información mostrada (el nivel actualmente) y gestionará en un futuro la navegación por los menús y los eventos propios de la interfaz relativos a sus botones. Finalmente se muestra la interfaz dentro del juego con estos cambios aplicados.

Ilustración 75 Zona de Nivel y Marcos de la Interfaz

Ilustración 76 Interfaz de Usuario con Nivel y Marcos

13.2.4.2.3. Tarea 7.2.3: Pruebas de Unidad

Por medio de combates con los enemigos básicos se ha ganado experiencia y se ha comprobado que el nivel que aparece en la interfaz de usuario es el correcto.

13.2.4.3. Subgrupo de Tareas 7.3: Submenú Pausa

13.2.4.3.1. Tarea 7.3.1: Diseño

El menú de pausa simplemente dispondrá de un botón para volver al juego. Al abrir este menú, directamente se congelará el tiempo en la escena y hasta no pulsar dicho botón o cerrar el propio menú no se restaurará el flujo normal del juego.

13.2.4.3.2. Tarea 7.3.2: Implementación

Ilustración 78 Fondo del Menú Pausa

Ilustración 77 Menú Pausa en el juego

La ilustración de la izquierda corresponde al fondo de menú generado con photoshop basándonos en el pack gratuíto de opengameart.org (ui-resources, Wyrmheart). Se han utilizado los gráficos del pack de texturas gratuíto anteriormente mencionado. La ilustración de la derecha es una captura del menú dentro del juego. El script “GUIController” es el encargado de controlar el acceso a este menú. Se ha definido en dicho script el uso de la tecla “P” para acceder o abandonar el menú en cuestión. El botón “Resume” simplemente cierra el menú. Al acceder a este menú la variable “timeScale” de la clase pública “Time” de Unity se iguala a cero, y al salir se le asigna su estado normal de uno. Esta clase Time es usada para obtener o modificar información relativa al tiempo del juego.

13.2.4.3.3. Tarea 7.3.3: Pruebas de Unidad

Se ha probado a pausar el tiempo en el juego mientras se realizan acciones de ataque y desplazamiento y al volver al juego estas acciones han seguido de forma normal ejecución.

13.2.4.4. Subgrupo de Tareas 7.4: Submenú Abandonar Partida

13.2.4.4.1. Tarea 7.4.1: Diseño

Al igual que el menú pausa, el diseño de este menú es bastante trivial. Para evitar que el jugador abandone por error el nivel, el menú tendrá un botón para abandonarlo definitivamente y otro para volver al juego. Se usará un fondo de menú y una fuente iguales a los del menú pausa.

13.2.4.4.2. Tarea 7.4.2: Implementación

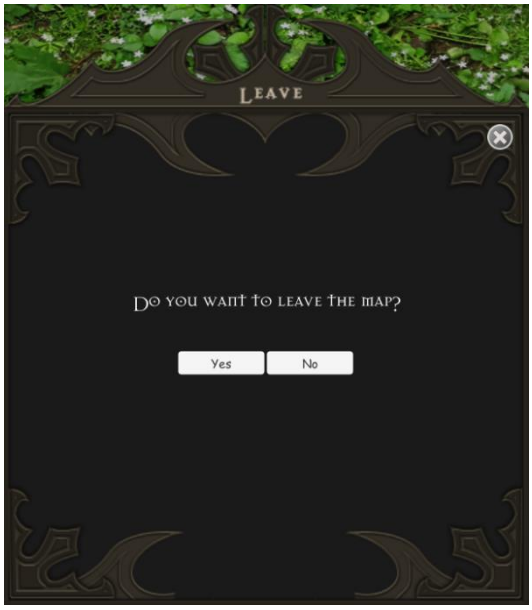


Ilustración 79 Menú Abandonar Partida

Como se ha trabajado en la sala de pruebas, el botón que confirma el abandono de nivel volverá a cargar el nivel actual (la sala de pruebas). Cuando se trabaje en el prototipo final se cargará la escena de título. Si no se confirma el abandono pulsando el botón “No” o pulsando el botón de cerrar nivel el menú se ocultará y se podrá volver al juego. Para evitar que el personaje se mueva al hacer click en estos menús se ha añadido un método al script “ClickToMove” del personaje principal llamado “canMove” que será invocado por el script “GUIController” para desactivar el movimiento si hay algún menú abierto. Se ha asignado el acceso rápido a la tecla “L” en el script “GUIController”.

13.2.4.4.3. Tarea 7.4.3: Pruebas de Unidad

Se ha comprobado que se desactiva el movimiento del personaje al acceder al menú, que vuelve a activarse al salir y que al confirmar el abandono de nivel se reinicia la escena.

13.2.4.5. Subgrupo de Tareas 7.5: Submenú Opciones

13.2.4.5.1. Tarea 7.5.1: Diseño

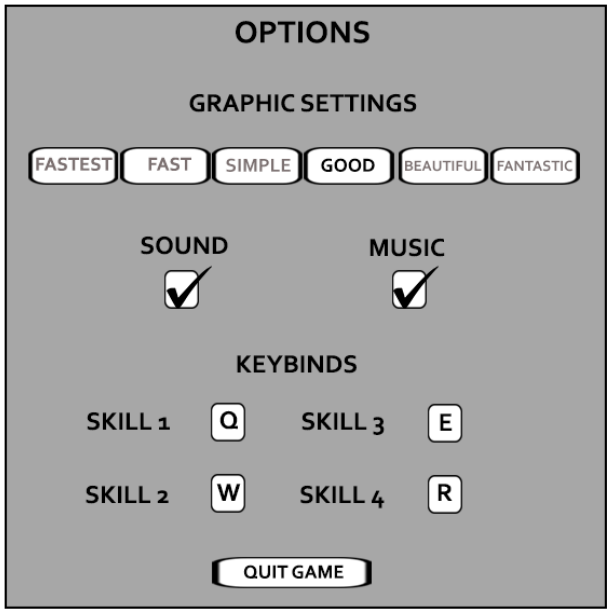


Ilustración 80 Boceto Menú Opciones

El menú de opciones constará de 3 elementos configurables y un botón para cerrar la aplicación:

- El nivel de gráficos.
- La reproducción de sonidos y música ambiental.
- La configuración de las teclas usada para ejecutar las habilidades especiales.

13.2.4.5.2. Tarea 7.5.2: Implementación

Como la música y los sonidos quedaron fuera del proyecto, se añadirán los controles de configuración, pero serán meramente estéticos. Añadirá la parte del sistema de keybinds (configuración de teclas por el usuario) para no tener que volver a manipular el menú cuando este sistema sea implementado.

Para la configuración del nivel de gráficos se usarán los niveles predeterminados de Unity. De peores gráficos a mejores y de mayor rendimiento a menor, estos son Fastest, Fast, Simple, Good, Beautiful y Fantastic. Para cambiar el nivel gráfico se usará un conjunto de botones que representan los anteriores niveles. Estos botones invocarán al método correspondiente del script “GUIController” para que éste realice los ajustes correspondientes.

El botón “Quit Game” cerrará la aplicación. Se ha asignado el acceso rápido para este menú a la tecla “O” en el script “GUIController”. Al igual que el anterior nivel, se deshabilita el movimiento del jugador al abrir el menú.

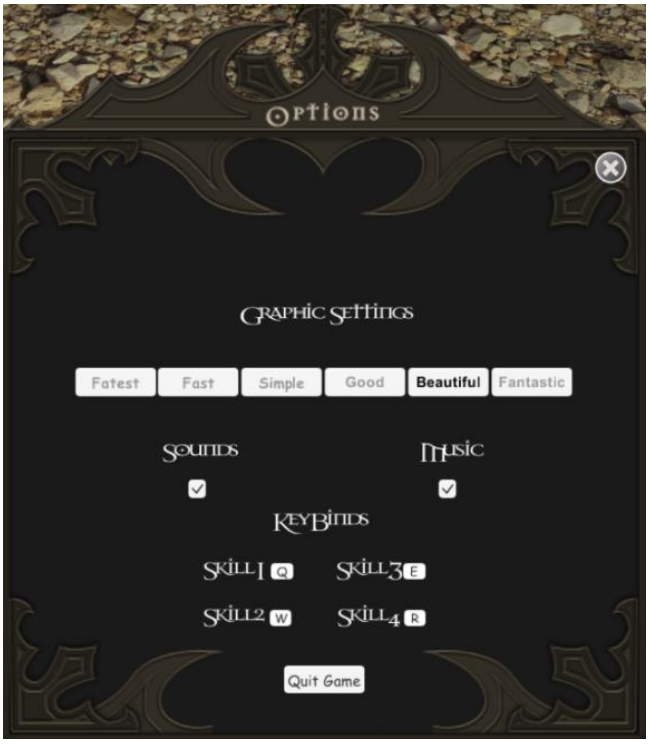


Ilustración 81 Menú Opciones

13.2.4.5.3. Tarea 7.5.3: Pruebas de Unidad

Se ha comprobado que el nivel de detalle cambia al seleccionar uno diferente al actual y que el botón que representa el nivel actual está resaltado. Como dentro del editor de Unity no podemos probar la función de salida de la aplicación, se ha generado un ejecutable del proyecto con la sala de pruebas como nivel y se ha podido comprobar que al pulsar el botón “Quit Game” la aplicación finaliza correctamente. También se ha comprobado que se desactiva el movimiento del personaje al acceder al menú y que vuelve a activarse al abandonarlo.

13.2.4.6. Subgrupo de Tareas 7.6: Submenú Características

13.2.4.6.1. Tarea 7.6.1: Diseño

Este menú mostrará los atributos del personaje así como los puntos de característica y los puntos no asignados. También contará con un botón para recuperar los puntos ya asignados y poder redistribuirlos libremente. Este grupo de tareas creará la interfaz para que el usuario pueda gestionar el sistema de características dentro del juego.

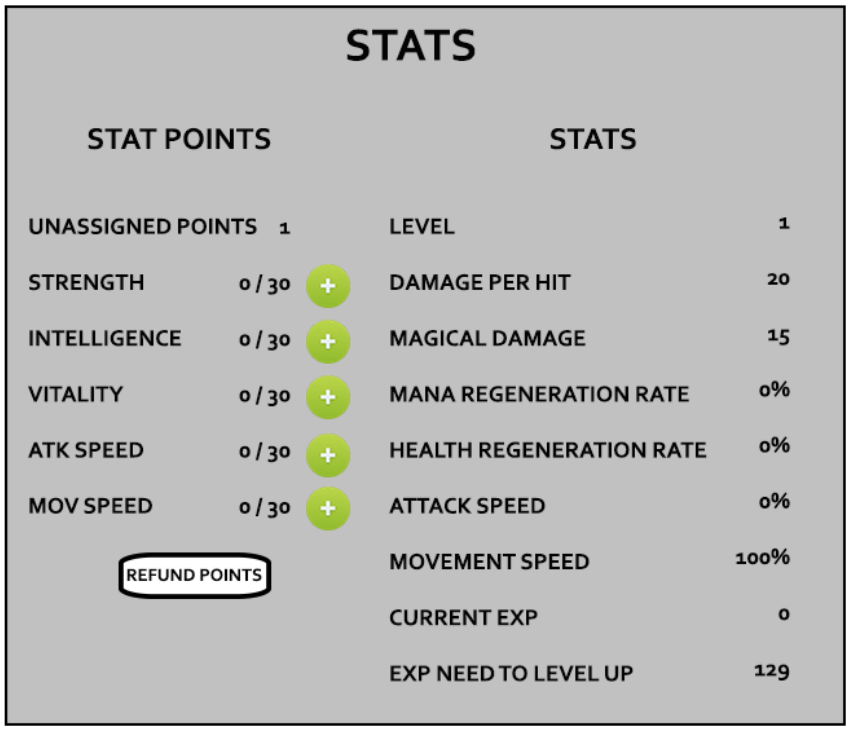


Ilustración 82 Boceto Menú Características

Adicionalmente se crearán 4 botones en la interfaz de usuario principal para acceder a los 4 menús implementados.

13.2.4.6.2. Tarea 7.6.2: Implementación

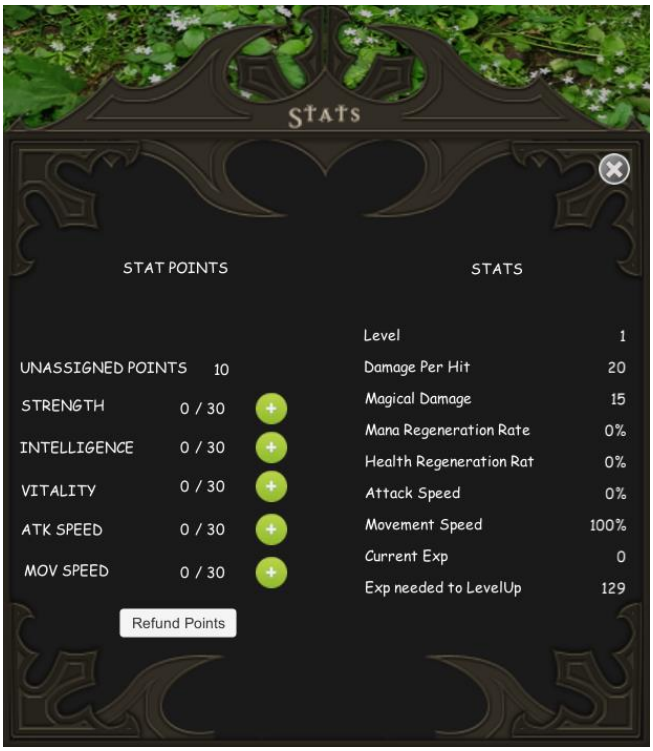


Ilustración 83 Menú Características

Para este menú se ha usado el mismo fondo que para los anteriores. El script "GUIController" del objeto "Canvas" será el encargado de gestionar los cambios en los puntos de característica y actualizar esta vista acorde al modelo (script "Combat" del personaje principal). Los botones para añadir los puntos se han creado en Photoshop.

Se seguirá el mismo procedimiento que con los otros menús y se deshabilitará el movimiento del persinaje principal mientras este menú permanezca activo. Se ha asignado el acceso rápido para este menú a la tecla “S” en el script “GUIController”.

Para facilitar el acceso a los menús sin necesidad de usar los atajos de teclado se han creado 4 botones en la interfaz principal obteniendo el siguiente resultado.

Ilustración 84 Interfaz de Usuario con Acceso a los Menús

13.2.4.6.3. Tarea 7.6.3: Pruebas de Unidad

Se ha comprobado que los menús funcionan correctamente. El menú de características se comporta como debería y las opciones gráficas cambian cuando el nivel de detalle es modificado.

13.2.4.7. Subgrupo de Tareas 7.7: Pruebas

Las últimas tareas de la iteración 2 consisten en pruebas de integridad y de usuario. Se contará con un nuevo testeador que reemplazará a JRC.

13.2.4.7.1. Tarea 7.7.1: Pruebas de Integridad

A lo largo de toda la iteración se ha comprobado el correcto funcionamiento de todos los componentes del juego. No obstante se han repetido numerosas pruebas para asegurar que todo funciona no debería. Se ha probado como los niveles de vida bajan cuando el personaje principal es atacado por los enemigos y que el movimiento de éste es deshabilitado cuando tenemos algún menú activo. Se ha comprobado que la cámara y el minimapa funcionan correctamente, que se gana experiencia al eliminar a un enemigo y se sube de nivel cuando se tiene la suficiente.

Se ha comprobando que si los puntos de característica se agregan a pocas características en lugar de repartirlos equitativamente es más beneficioso.

13.2.4.7.2. Tarea 7.7.2: Hito: Pruebas de Usuario

Los testeadores CCG y JJRG, ambos estudiantes del grado de Ingeniería Informática. Se ha limpiado la sala de pruebas y se han instanciado 15 enemigos básicos por la zona inicial del mapa. Esto será suficiente para que completen los 10 primeros niveles del personaje principal y puedan probar los menús. Se ha usado una escala del uno al cinco, donde cinco es “Sí, mucho” y uno es “No, nada”.

Cuestionario de Pruebas de Usuario - 2ª Iteración, Tarea 7.7.2			
Nº Pregunta	**Pregunta**	**CCG**	**JJRG**
1	¿Cree que la inclinación de la cámara es correcta?	4	5
2	¿Cree que la altura de la cámara con respecto al jugador es correcta?	4	4
3	¿Le resulta no invasivo el minimapa?	5	5
4	¿Le resulta no invasiva la interfaz de usuario?	5	5
5	¿Le ha resultado fácil subir los primeros niveles (1-5) del personaje?	4	5
6	¿Le ha resultado difícil subir los niveles sucesivos al 6 del personaje?	4	3
7	¿Le resulta atractiva la interfaz principal de usuario?	5	4
8	¿Le resultan atractivos los menús del juego (pausa, opciones, etc...)?	5	5
9	¿El sistema de características le permite adaptar al personaje principal a su propio estilo de juego?	3	5
10	¿Le resulta de utilidad la información mostrada en el menú de características?	4	4
11	¿Le resulta útil el botón que devuelve los puntos de característica y permite reasignarlos de nuevo?	5	5

Tabla 19 Cuestionario de Pruebas de Usuario - 2ª Iteración, Tarea 7.7.2

Después de unas 4 horas de prueba por parte de los testeadores se ha recogido el cuestionario y los formularios de condiciones anómalas detectadas.

Condiciones Anómalas detectadas - 2ª Iteración, Tarea 7.7.2	
Testeador:	CCG
El botoncito de cerrar ventana del menú pausa no funciona.	

Tabla 20 Condiciones Anómalas detectadas - 2ª Iteración - CCG

Este botón ha sido corregido inmediatamente. No llamaba al método correcto del script “GUIController”.

Condiciones Anómalas detectadas - 2ª Iteración, Tarea 7.7.2	
Testeador:	JJRG
El minimapa aporta poca información útil.	
A veces, al ir regenerando vida, nunca termina de llenarse totalmente la barra. Con el maná pasa lo mismo.	

Tabla 21 Condiciones Anómalas detectadas - 2ª Iteración - JJRG

Los iconos del minimapa serán incluídos en próximas iteraciones, por lo que ahora no es preocupante la primera condición anómala.

La segunda condición ha sido solucionada. Al hacer la interpolación lineal se redondeaba el resultado, y por eso nunca se completaban la barras. Sólo era un error gráfico, los valores internos de vida y maná eran correctos.

13.3. Iteración 3 (del 29-02-2016 al 13-03-2016)

Esta iteración será dedicada exclusivamente al diseño e implementación de las cuatro habilidades especiales de las que dispondrá el jugador. Está totalmente justificado, pues hay que:

- Diseñar las habilidades.
- Implementar los efectos especiales.
- Definir e implementar cómo estas habilidades interaccionan con el entorno (fundamentalmente con los enemigos).
- Desarrollar el sistema de tiempo de enfriamiento que ayudará enormemente al usuario.
- Probar la integridad con lo desarrollado anteriormente.
- Realizar exhaustivas pruebas de usuario con al menos 3 testeadores.

*13.3.1. **Grupo de Tareas 8**: Sistema Habilidades Especiales jugador*

13.3.1.1. Subgrupo de Tareas 8.1: Gráficos

En este subgrupo se definirán las habilidades y se crearan sus efectos especiales. Cabe destacar que no se tienen tareas de pruebas como tal, pues los efectos son meramente estéticos y se irá comprobando como quedan directamente mientras se implementan.

13.3.1.1.1. Tarea 8.1.1: Diseño

Esta es, sin duda, la tarea que ha sido menos definida durante el proceso de definición del ámbito. Aquí las posibilidades son ilimitadas y ha tomado bastante tiempo definir qué se va a construir. Las siguientes habilidades han sido escogidas según la experiencia del autor en videojuegos parecidos. Combinadas otorgan al usuario un kit de habilidades bastante completo con el cual enfrentarse a cualquier situación.

Se va a usar la siguiente tabla para presentar las habilidades y sus iconos. Todos los iconos son propiedad de Blizzard (presentes en World Of Warcraft) menos el de escudo de hielo, que es propiedad de Valve (Dota 2). Se ha dedicido usar estos iconos para el prototipo en lugar de crearlos.

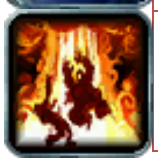
Gráficos de las Habilidades Especiales del Personaje Principal		
Icono	**Nombre**	**Descripción**
	**Bola de Fuego**	El personaje lanza una bola de llamas que atraviesa a los enemigos infligiendo daño mágico.
	**Teletransporte**	Teletransporta al personaje a la posición actual del ratón instantáneamente.
	**Escudo de Hielo**	Otorga un escudo temporal que mitiga el daño recibido por los enemigos.
	**Ragnarök**	Desencadena una explosión alrededor del personaje ocasionando grandes daños físicos y mágicos.

Tabla 22 Gráficos Habilidades Especiales del Personaje Principal

13.3.1.1.2. Tarea 8.1.2: Implementación

Para implementar la habilidad “Bola de Fuego” se usarán 2 sistemas de partículas con una textura de fuego que trae el motor. El primer sistema de partículas dará cuerpo a la bola y el segundo creará la estela de fuego que sigue al proyectil.

Ilustración 85 Habilidad “Bola de Fuego”

Para implementar el efecto visual de la habilidad “Teletransporte” se ha usado un sistema de partículas de un pack de componentes gratuíto del Unity Store llamado Elementals. El efecto se instanciará en el origen y el destino de la habilidad. Se ha creado un script llamado “Destroy On Time” que controlará el tiempo de destrucción de cualquier objeto al que se le añada, en este caso a los sistemas de partículas.

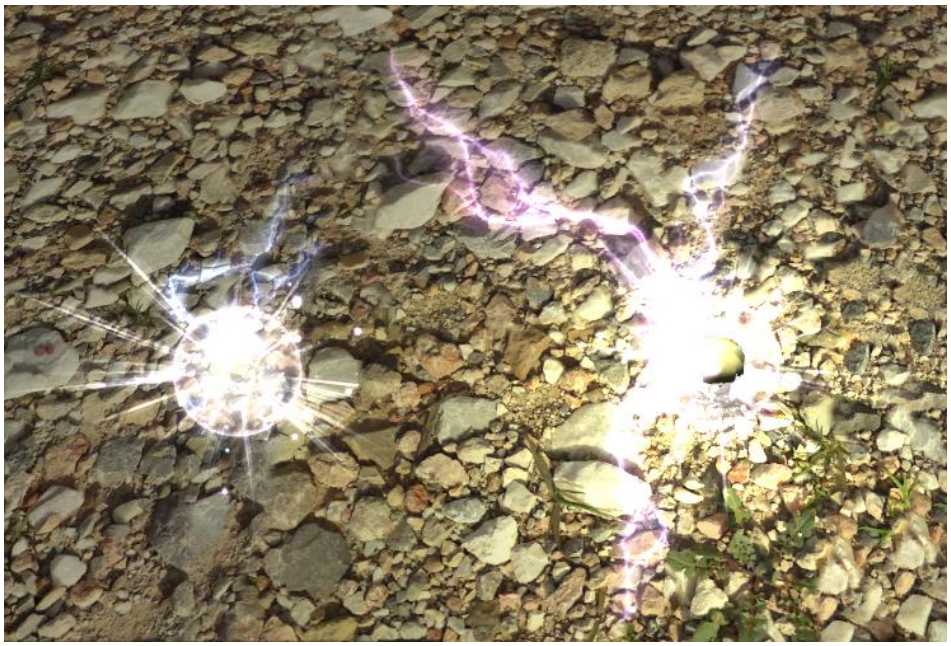


Ilustración 86 Habilidad "Teletransporte"

Ilustración 87 Habilidad "Escudo de Hielo"

Para crear el efecto visual de la habilidad “Escudo de Hielo” Se ha usado un sistema de partículas con efecto de niebla del pack gratuíto anteriormente mencionado (Elementals). Se ha modificado el color para que tuviese apariencia de hielo. Se ha descargado un shader gratuíto de un vídeo de youtube del canal de “Eleazar Celis” que hace un efecto “glow” (brillo) para crear los materiales del personaje cuando se usa esta habilidad.

Para crear el efecto visual de la habilidad “Ragnarök” se ha usado un complejo sistema de partículas del pack gratuíto descargado del Unity Store llamado Elementals.

Ilustración 88 Habilidad "Ragnarök"

13.3.1.2. Subgrupo de Tareas 8.2: Efectos – Ventajas

En este grupo de tareas se creará el sistema que permitirá lanzar las habilidades, computará los efectos y se incluirán los botones y atajos de teclado necesarios para usarlas con sus respectivos tiempos de recarga.

13.3.1.2.1. Tarea 8.2.1: Diseño

La tabla del apartado *7.7 Habilidades del personaje principal* (*Tabla 2 Características de las Habilidades Especiales del Personaje Principal*) muestra las características de las habilidades tales como el coste de maná, el tiempo de rehutilización, el beneficio que representan y el rango de acción en el que actuarán.

13.3.1.2.2. Tarea 8.2.2: Implementación

Para implementar la habilidad “Bola de Fuego” se ha creado un script llamado “Projectile Conf” que causará el daño a los enemigos al impactar por medio de un componente “Capsule Collider” esférico, controlará el movimiento del proyectil y su duración.

Para implementar el efecto, la habilidad “Teletransporte” se ha creado un método en el script que controla el movimiento del personaje “Click To Move” que cambiará la posición del personaje principal. Se ha creado un script llamado “Destroy On Time” que controlará el tiempo de destrucción de cualquier objeto al que se le añada, en este caso a los sistemas de partículas.

Para crear el efecto de la habilidad “Escudo de Hielo” se han incluído en el script “Combat” del personaje principal los atributos y materiales necesarios.

Para crear el efecto de la habilidad “Ragnarök” ha añadido al sistema de partículas un capsule collider para que impacte en los enemigos y se ha creado un script llamado “Ultimate” para computar el daño causado a los enemigos. Se le ha añadido el script “Destroy On Time” anteriormente creado.

Finalmente se han añadido los iconos de las habilidades como botones a la interfaz principal. Estos botones ejecutarán métodos del script “GUIController”. Este

script invocará los métodos necesarios en el script “Combat” del personaje principal que serán los encargados de crear los efectos visuales y beneficios si la habilidad no está en enfriamiento y se dispone del maná que cuesta. Se ha mapeado las teclas Q, W, E y R como los controles predeterminados para el uso de las habilidades.

Ilustración 89 Aspecto de la Interfaz de Usuario Final

13.3.1.2.3. Tarea 8.2.3: Pruebas de Unidad

Se ha comprobado que cada habilidad es lanzada por medio de su tecla de acceso directo y botón. Se ha comprobado que las habilidades realizan su funcón correctamente y que los costes de maná se descuentan bien.

13.3.1.3. Subgrupo de Tareas 8.3: Sistema de Tiempo de Enfriamiento

Estas tareas crearán el sistema de reutilización de las habilidades y añadirán un efecto de sombreado radial progresivo a los botones de las habilidades.

13.3.1.3.1. Tarea 8.3.1: Diseño

Cuando una habilidad sea usada su uso será deshabilitado durante unos segundos. Esto es conocido como tiempo de enfriamiento o tiempo de rehutilización. El botón de dicha habilidad mostrará los segundos restantes para volver a usar la habilidad y un leve efecto de sombreado que va desapareciendo conforme acaba el tiempo.

13.3.1.3.2. Tarea 8.3.2: Implementación

Se ha conseguido este efecto por medio de métodos que serán llamados con un retardo en el script “Combat” del personaje principal. Los efectos en la interfaz principal son gestionados por el script “GUIController” del objeto Canvas. Se usarán

corrutinas para temporizar el contador de segundos restantes. El efecto de sombreado se conseguirá con una imagen más oscura del icono en un objeto Image que desaparecerá de manera radial.

Ilustración 90 Tiempos de Enfriamiento en la Interfaz Principal

13.3.1.3.3. Tarea 8.3.3: Pruebas de Unidad

Se ha comprobado que cuando se lanza una habilidad se deshabilita temporalmente, se muestra en tiempo restante en el botón, se completa el efecto de sombreado y se vuelve a habilitar cuando este tiempo acaba.

13.3.1.4. Subgrupo de Tareas 8.4: Pruebas

En este grupo de tareas se llevarán a cabo las pruebas de integridad (por parte del equipo de desarrollo) y de usuario (por parte de los testeadores voluntarios).

13.3.1.4.1. Tarea 8.4.1: Pruebas de Integridad

Se ha limpiado la sala de pruebas y se han instanciado 30 enemigos. Se ha comprobado que el juego funciona correctamente con los ultimos elementos añadidos mientras se alcanzaba el nivel 18. No han surgido condiciones anómalas a causa de la inclusión de las nuevas funcionalidades.

13.3.1.4.2. Tarea 8.4.2: Hito: Pruebas de Usuario

Los testeadores serán CCG, JJRG (estudiantes del grado de Ingeniería Informática) y RVA (estudiante del grado de Ingeniería Eléctrica). Se ha limpiado la sala de pruebas y se han instanciado 25 enemigos básicos por la zona inicial del mapa.

No se han registrado condiciones anómalas por parte de los testeadores.

El siguiente cuestionario ha sido realizado por los testeadores. Se ha usado una escala del uno al cinco, donde cinco es “Sí, mucho” y uno es “No, nada”.

Cuestionario de Pruebas de Usuario - 3ª Iteración, Tarea 8.4.2				
Nº Pregunta	**Pregunta**	**CCG**	**RVA**	**JJRG**
1	¿El tamaño del proyectil de la habilidad "**Bola de Fuego**" es lo suficientemente grande?	5	5	5
2	¿El tiempo de enfriamiento de la habilidad "**Bola de Fuego**" (1,5 segundos) es suficientemente bajo?	5	5	4
3	¿El escalado del daño de la habilidad "**Bola de Fuego**" con respecto a la característica Inteligencia es notable?	4	5	5
4	¿El coste de maná de la habilidad "**Bola de Fuego**" (25) es coherente respecto al beneficio?	5	5	5
5	¿La distancia que a la que se desplaza el jugador por medio de la habilidad "**Teletransporte**" es suficiente?	4	4	5
6	¿El tiempo de enfriamiento de la habilidad "**Teletransporte**" (5 segundos) es suficientemente bajo?	5	3	5
7	¿El coste de maná de la habilidad "**Teletransporte**" (25) es coherente respecto al beneficio?	4	5	4
8	¿El efecto visual de la habilidad "**Escudo de Hielo**" es notable?	4	5	5
9	¿El tiempo de enfriamiento de la habilidad "**Escudo de Hielo**" (15 segundos) es suficientemente bajo?	3	4	4
10	¿El coste de maná de la habilidad "**Escudo de Hielo**" (200) es coherente respecto al beneficio?	5	4	5
11	¿El efecto visual de la habilidad "**Ragnarök**" es notable?	5	5	5
12	¿El tiempo de enfriamiento de la habilidad "**Ragnarök**" (60 segundos) es suficientemente bajo?	5	5	5
13	¿El coste de maná de la habilidad "**Ragnarök**" (85% del máximo) es coherente respecto al beneficio?	4	4	5
14	¿El escalado del daño de la habilidad "**Ragnarök**" con respecto a las características Inteligencia y Fuerza es notable?	5	5	5
15	¿El tamaño de la explosión de la habilidad "**Ragnarök**" es lo suficientemente grande?	5	5	5

Ilustración 91 Cuestionario de Pruebas de Usuario - 3ª Iteración, Tarea 8.4.2

Los resultados de las pruebas de usuario son muy satisfactorias. Las notas más bajas han sido de elementos que pretenden limitar la acción del jugador, como las preguntas número 5, 9 y 13. Tener una habilidad poderosa cada poco tiempo puede hacer el juego más fácil y aburrido para el usuario.

13.4. Iteración 4 (del 14-03-2016 al 27-03-2016)

En esta iteración se crearán los 15 enemigos nomrmales diferentes (12 cuerpo a cuerpo y 3 a distancia) que existirán en el videojuego.

*13.4.1. **Grupo de Tareas 9**: Enemigos y Jefes*

13.4.1.1. Subgrupo de Tareas 9.1: Enemigos

13.4.1.1.1. Subgrupo de Tareas 9.1.1: Enemigos Cuerpo a Cuerpo

13.4.1.1.1.1. Tarea 9.1.1.1: Diseño

Para crear los enemigos se usarán modelos descargados de la Unity Store y tf3dmodels.com. Modelar, texturizar, rigear y animar cada enemigo llevaría un tiempo muy superior a la duración de este proyecto.

Diseño de Enemigos Comunes Cuerpo a Cuerpo		
Imagen	Características	Fuente
	**Nombre:** Skeleton Lancer **Vida:** 200 **Ataque:** 10 **Modificaciones:** Ninguna.	Skeletons Pack, Unity Store

	**Nombre:** Skeleton Highlander **Vida:** 200 **Ataque:** 30 **Modificaciones:** Se le ha incluido un casco torcido, una segunda espada y se ha modificado la textura de los huesos.	Skeletons Pack, Unity Store
	**Nombre:** Skeleton Protector **Vida:** 400 **Ataque:** 20 **Modificaciones:** Se le ha modificado la textura de los huesos.	Skeletons Pack, Unity Store
	**Nombre:** Slime (Limo) **Vida:** 100 **Ataque:** 5 **Modificaciones:** Ninguna.	Animated Knight and Slime Monster Pack, Unity Store
	**Nombre:** Black Slime **Vida:** 300 **Ataque:** 25 **Modificaciones:** Se le ha modificado la textura y el tamaño del modelo.	Animated Knight and Slime Monster Pack, Unity Store

	**Nombre:** Mud Hunter **Vida:** 500 **Ataque:** 60 **Modificaciones:** Ninguna.	Monster2, Unity Store
	**Nombre:** Mud Opressor **Vida:** 600 **Ataque:** 100 **Modificaciones:** Se le ha modificado la textura y el material.	Monster2, Unity Store
	**Nombre:** Goblin Snatcher **Vida:** 500 **Ataque:** 70 **Modificaciones:** Ninguna.	Goblin, Unity Store
	**Nombre:** Wolf **Vida:** 125 **Ataque:** 25 **Modificaciones:** Ninguna.	Wolf 3d model, tf3dm.com

	**Nombre:** Troll **Vida:** 1200 **Ataque:** 160 **Modificaciones:** Se ha modificado el material de los accesorios para que parezcan más metálicos.	The Earthborn Troll, Unity Store
	**Nombre:** Cyclops **Vida:** 1500 **Ataque:** 250 **Modificaciones:** Ninguna.	Cyclop Soldier, Unity Store
	**Nombre:** Flesh Golem **Vida:** 1700 **Ataque:** 90 **Modificaciones:** Se ha modificado la textura para que parezca que está hecho de carne cruda y el material para que brille más.	Maze element Ice Golem Pack, Unity Store
	**Nombre:** Earth Golem **Vida:** 1500 **Ataque:** 180 **Modificaciones:** Se ha modificado la textura para que parezca que está hecho de barro.	Maze element Ice Golem Pack, Unity Store

Tabla 23 Diseño de Enemigos Comunes Cuerpo a Cuerpo

13.4.1.1.1.2. Tarea 9.1.1.2: Implementación

Para implementar los 12 enemigos cuerpo a cuerpo se ha usado el mismo procedimiento y los mismos componentes que para el enemigo de pruebas (*13.1.3.1 Subgrupo de Tareas 3.1: Enemigo Básico*). Se han ajustado parámetros para cada modelo, como por ejemplo el tamaño del Capsule Collider.

13.4.1.1.1.3. Tarea 9.1.1.3: Pruebas de Unidad

Se han realizado pruebas combatiendo contra los enemigos uno a uno varias veces (unas 8 de media). Los resultados obtenidos han sido usados para terminar de ajustar los parámetros propios de cada enemigo.

13.4.1.1.2. Subgrupo de Tareas 9.1.2: Enemigos a Distancia

13.4.1.1.2.1. Tarea 9.1.2.1: Diseño

Para crear los enemigos se usarán modelos descargados de la Unity Store. Modelar, texturizar, rigear y animar cada enemigo llevaría un tiempo muy superior a la duración de este proyecto.

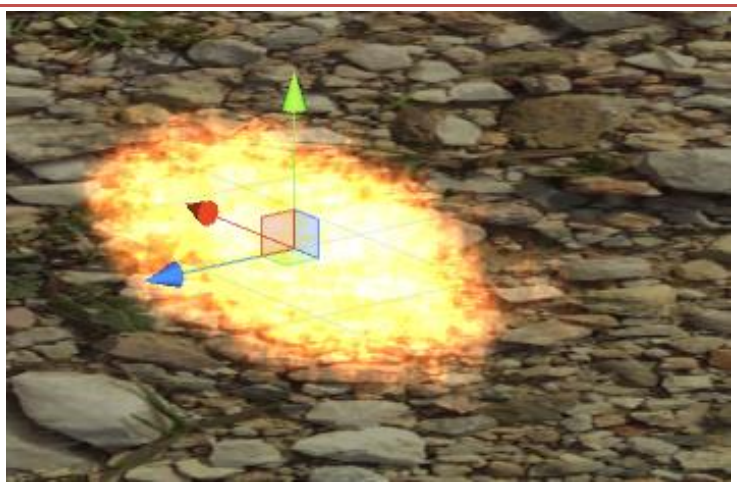
Los proyectiles que usarán estos enemigos para hacer daño al personaje principal serán creados con sistemas de partículas y texturas básicas de Unity.

Diseño de Enemigos Comunes a Distancia		
Imagen	**Características**	**Fuente**
	**Nombre:** Skeleton Mage **Vida:** 170 **Ataque:** 20 **Modificaciones:** Se le ha añadido el escudo de la mano izquierda.	Skeletons Pack, Unity Store

	**Nombre:** Red Slime (Limo) **Vida:** 100 **Ataque:** 10 **Modificaciones:** Se ha cambiado el color de la textura.	Animated Knight and Slime Monster Pack, Unity Store
	**Nombre:** CaveWorm **Vida:** 250 **Ataque:** 25 **Modificaciones:** Se ha generado un material especialmente para éste modelo.	FT_CaveWorm, Unity Store

Tabla 24 Diseño de Enemigos Comunes a Distancia

A continuación se presentan los proyectiles usados por cada enemigo.

Proyectiles de los Enemigos a Distancia	
Enemigo	**Proyectil**
Skeleton Mage	

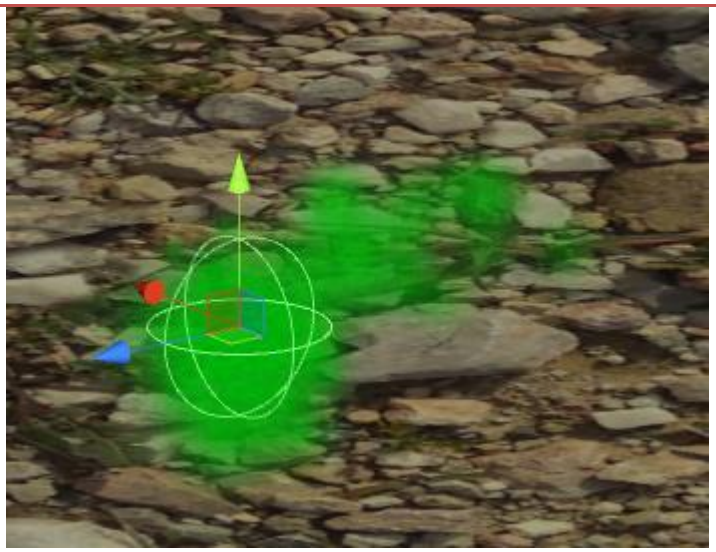
Red Slime	
CaveWorm	

Tabla 25 Proyectiles de los Enemigos a Distancia

Estos proyectiles se desplazan en la dirección del eje Z positivo (flecha azul en las imágenes).

13.4.1.1.2.2. Tarea 9.1.2.2: Implementación

Estos sistemas de partículas se han envuelto con un capsule collider que computará las colisiones. Se ha añadido a cada proyectil un script llamado “Proyectile Mob Conf” que calculará el daño, velocidad y tiempo de vida del objeto.

Se ha creado un nuevo script para los enemigos a distancia que hereda del script “Mob” de los enemigos cuerpo a cuerpo. La única diferencia entre los 2 script es que los enemigos a distancia deben disponer de un proyectil que instanciar.

13.4.1.1.2.3. Tarea 9.1.2.3: Implementación IA

La inteligencia artificial será la misma que para los enemigos cuerpo a cuerpo (*13.1.3.1.3 Tarea 3.1.3: Implementación IA*), salvo que el rango de ataque de estos enemigos será muy superior.

13.4.1.1.2.4. Tarea 9.1.2.4: Pruebas de Unidad

Las pruebas de unidad son las mismas usadas para los enemigos cuerpo a cuerpo (*13.4.1.1.1.3 Tarea 9.1.1.3: Pruebas de Unidad*).

13.5. Iteración 5 (del 28-03-2016 al 10-04-2016)

En esta iteración se crearán los enemigos finales del videojuego. Estos enemigos serán más fuertes que los comunes y presentarán mecánicas más complejas. Los modelos usados serán descargados gratuítamente de internet y retocados para obtener el resultado deseado. Esta iteración es la última que aporta funcionalidad al prototipo, exceptuando el sistema de Keybinds de la iteración 6.

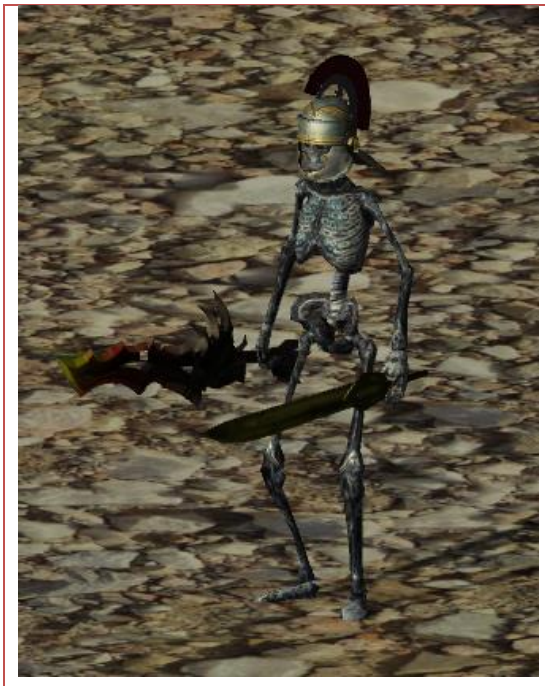
13.5.1. ***Grupo de Tareas 9**: Enemigos y Jefes*

13.5.1.1. Subgrupo de Tareas 9.2: Jefes

13.5.1.1.1. Tarea 9.2.1: Diseño

Aquí se diseñarán las características, se mencionarán los cambios realizados en los modelos originales y mostrarán los resultados de los 6 enemigos élite (jefes).

Diseño de Jefes		
Imagen	**Características**	**Fuentes**
	**Nombre:** Burp, Slime King **Vida:** 900 **Ataque:** 80 **Modificaciones:** Se le ha cambiado el color de la textura y el tamaño del modelo. Se le ha añadido una corona que se ha tenido que simplificar geométricamente usando Blender.	Animated Knight and Slime Monster Pack, Unity Store. Crown Jewels, 123dapp.com (Autodesk)
	**Nombre:** Gulp, Slime Queen **Vida:** 700 **Ataque:** 90 **Modificaciones:** Se le ha cambiado el color de la textura y el tamaño del modelo. Se le ha añadido una corona que se ha tenido que simplificar geométricamente usando Blender.	Animated Knight and Slime Monster Pack, Unity Store. Jewelled Crown, 123dapp.com (Autodesk)

	**Nombre**: Sir Thomas II, Relentless Commander **Vida**: 350 **Ataque**: 40 **Modificaciones**: Se ha cambiado la textura de los huesos. Se ha cambiado la textura de la espada básica para darle un toque dorado. Se ha añadido una segunda espada (mismo modelo que la del personaje principal). Se le ha añadido un casco romano.	Skeletons Pack, Unity Store. Roman Centurion Helmet, tf3dmodels.com
	**Nombre:** Abramelin, The Dark Mage **Vida:** 5000 **Ataque:** 300 **Modificaciones:** Partiendo del Skeleton Mage, se ha cambiado el color de las texturas y el material del bastón, capa, sombrero y escudo.	Skeletons Pack, Unity Store

	**Nombre**: Nightmare **Vida**: 2000 **Ataque**: 170 **Modificaciones**: Se ha generado de manera procedural un material exclusivo para este modelo.	FT_CaveWorm, Unity Store
	**Nombre:** SkullCrusher **Vida:** 6000 **Ataque:** 400 **Modificaciones:** Se ha cambiado el color de la textura de la piel. Se ha modificado el material y la textura del arma para que parezca de acero.	Cyclop Soldier, Unity Store

Tabla 26 Diseño de Jefes

13.5.1.1.2. Tarea 9.2.2: Implementación

Para implementar los jefes se ha usado el mismo procedimiento y los mismos componentes que para el enemigo de pruebas (*13.1.3.1 Subgrupo de Tareas 3.1: Enemigo Básico*). Se han ajustado parámetros para cada modelo, como por ejemplo el tamaño del Capsule Collider. Para algunos enemigos ha sido necesario crear script que hereden del script básico “Mob” (El script a Abramelin, por ejemplo) ya que el comportamiento y funcionalidades diferían demasiado.

Muchas partes de estos modelos han sido modificadas. Un ejemplo son las coronas en alta definición de los 2 reyes Slime, que tenían unos 30.000 triángulos y han sido simplificadas usando Blender y su herramienta “Decimate”, bajando esta cantidad a 4.000.

Muchas de los materiales que venían con los modelos por defecto han sido modificados, cambiando su textura o shader, como por ejemplo la textura de

SkullCruser (originalmente la textura del enemigo normal Cyclops, *Tarea 9.1.1.1: Diseño*).

13.5.1.1.3. Tarea 9.2.3: Implementación IA

A continuación se resumen los cambios y mecánicas particulares de cada jefe implementado.

13.5.1.1.3.1. Burp, Slime King

Este enemigo, aun siendo un jefe, tiene la inteligencia artificial del enemigo básico cuerpo a cuerpo. Sus puntos de vida y ataque son algo superiores.

13.5.1.1.3.2. Gulp, Slime Queen

Este enemigo, aun siendo un jefe, tiene la inteligencia artificial del enemigo básico a distancia (Red Slime). Sus puntos de vida y ataque son algo superiores. El proyectil que usa es más grande y su cadencia de disparo es menor. Este proyectil ha sido creado con un sistema de partículas y texturas por defecto de Unity,

Ilustración 92 Proyectil de Gulp, Slime Queen

13.5.1.1.3.3. Sir Thomas II, Relentless Commander

Posee la inteligencia artificial del enemigo básico cuerpo a cuerpo, pero debido a su gran tamaño su rango de ataque es mayor. Alrededor del jefe giran unas espadas espectrales a gran velocidad que infligen daño al jugador si se encuentra demasiado cerca. A esta espada (existente en el asset Skeletons Pack) se le ha aplicado un shader FX Flare para conseguir el siguiente resultado.

Ilustración 93 Espada Espectral de Sir Thomas II

Para crear el giro de las espadas se ha creado un script “Rotate Item” añadido a la espada que hace que ésta gire sobre un pivote central.

Después se ha creado un objeto vacío a los pies del modelo del jefe llamado “Area Rotation”. A este objeto se le ha añadido un script “Rotate Around Char” que instancia 4 espadas y las mueve alrededor del modelo del jefe con una velocidad angular de 1 y a una distancia de 0,411 unidades.

13.5.1.1.3.4. Abramelin, The Dark Mage

Este enemigo posee la inteligencia artificial del enemigo básico a distancia (Red Slime). Cada 20 segundos invoca a 1 Skeleton Protector que atacará al jugador. El proyectil (Bola de fuego) lanzado por este jefe es más ancho y dañino que el de sus versiones inferiores (Skeleton Mage).

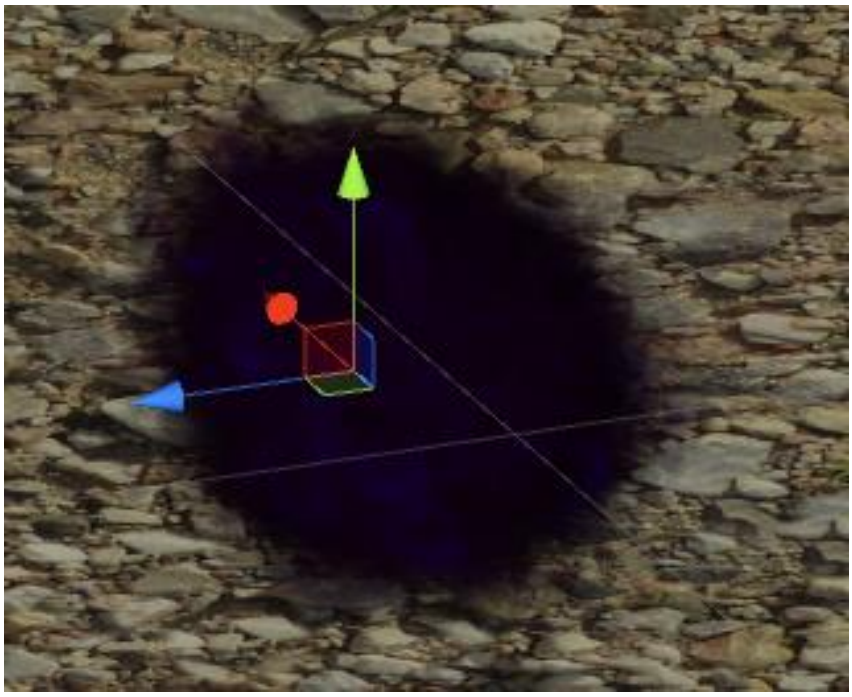


Ilustración 94 Proyectil de Abramelin

13.5.1.1.3.5. Nightmare

Este jefe es una versión potenciada del enemigo Caveworm. Su cadencia de disparo es menor, su vida, ataque y tamaño son mayores que las del enemigo Caveworm (*Subgrupo de Tareas 9.1.2: Enemigos a Distancia*).

13.5.1.1.3.6. SkullCrusher

Este enemigo élite (jefe) se comporta como el enemigo básico. Su ataque es muy fuerte, llegando a matar al jugador directamente de 1 golpe. Se ha añadido al jefe un script “Health Regeneration” que restaura un 2% de su vida máxima cada segundo.

13.5.1.1.4. Tarea 9.2.4: Pruebas de Unidad

Se ha combatido contra los jefes en numerosas ocasiones para comprobar que su comportamiento es el deseado. Estas pruebas han ayudado a ajustar parámetros propios de cada jefe como la regeneración de vida de SkullCrusher o la invocación periódica de esqueletos por parte de Abramelin.

COMBATES DE PRUEBA REALIZADOS	
Jefe	**Número de combates**
Burp, Slime King	8
Gulp, Slime Queen	7
Sir Thomas II, Relentless Commander	16
Abramelin, The Dark Mage	19
Nightmare	10
SkullCrusher	6

Tabla 27 Combates necesarios para completar exitosamente la fase de Pruebas

13.5.1.2. Subgrupo de Tareas 9.3: Auras Jefes

Las auras ayudarán al jugador a diferenciar a enemigos especiales del resto. Si bien el tamaño de los jefes es superior que el tamaño de los demás enemigos, no está de más resaltar esta diferencia de cara al usuario. En fases posteriores también se incluirá un icono representativo en el minimapa para alertar al jugador de la existencia de estos enemigos en el mapa.

13.5.1.2.1. Tarea 9.3.1: Diseño

Desde la base del modelo del jefe emanarán haces de luces de diferentes colores. Cada color representará unívocamente a un jefe en concreto. Estos colores irán variando con el tiempo.

13.5.1.2.2. Tarea 9.3.2: Implementación

Para las auras se usará un sistema de partículas circular (Ellipsoid Particle Emitter y un Particle Animator) que generará ráfagas de luz de colores periódicamente. Se puede encontrar un sistema muy similar en el asset gratuíto “Skeletons Pack” descargado de Unity Store.

Como el uso de estos elementos es novedoso para el autor, se ha decidido coger ese sistema ya existente y modificarlo para que cumpla las espectativas propias de funcionamiento.

Los resultados obtenidos se reflejan en el apartado *9.8 Auras de los Jefes*.

13.5.1.2.3. Tarea 9.3.3: Pruebas de Unidad

Para probar las auras se ha instanciado cada jefe en el mapa y se ha observado cómo el aura se iba generando y cambiando el color. Se han ajustado para que puedan verse fácilmente y sean reconocibles.

13.5.1.3. Subgrupo de Tareas 9.4: Sistema de Generación de Items

13.5.1.3.1. Tarea 9.4.1: Diseño

Se diseñará un componente Script reutilizable que, dado un conjunto de objetos y una probabilidad asociada a cada uno, instancie los correctos al morir un enemigo.

Los objetos que se incluirán en el juego serán pociones de vida y pociones de maná. En un futuro se podría incluir cualquier objeto, como bonificaciones temporales de resistencia, velocidad, etc.

Al morir un enemigo se instanciará aleatoriamente 1 poción de vida y otra de maná. Las pociones de vida restauran 50 puntos de salud y las de maná 50 puntos de magia. Al entrar en contacto con una poción del suelo se usará inmediatamente si puede restaurarse esa cantidad de salud o magia.

Los modelos para las pociones han sido descargados gratuítamente de turbosquid.com. El nombre del pack es “Health and Mana Potions” y el autor es Dagon19.

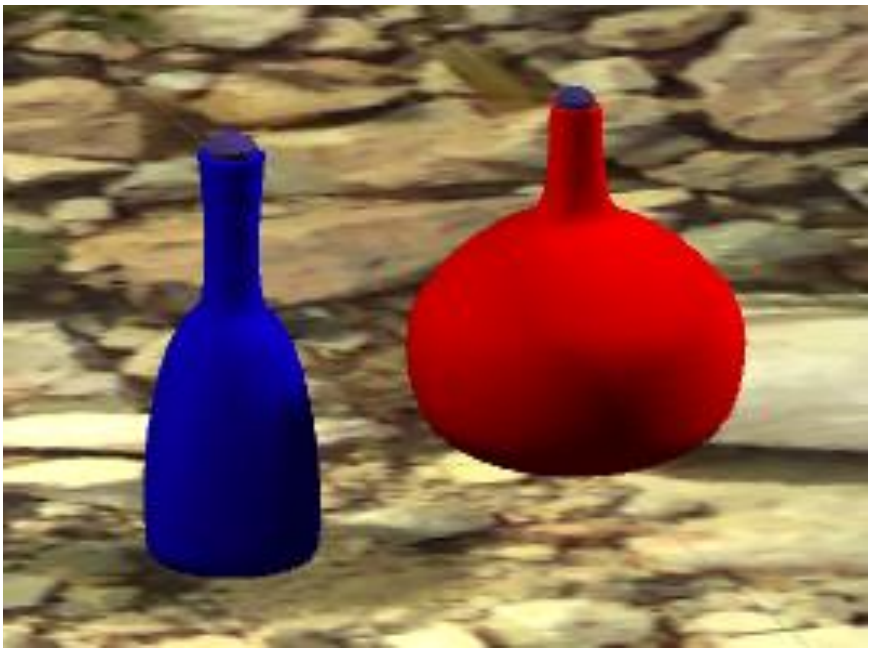


Ilustración 95 Modelos de Pociones de Maná (izquierda) y Vida (derecha)

13.5.1.3.2. Tarea 9.4.2: Implementación

Para implementar las pociones se ha incluído en el modelo un Capsule Collider para comprobar si el personaje principal está lo suficientemente cerca como para recogerlas. Un hecho importante es que, si el usuario no ha perdido vida no podrá coger la poción de vida. Con las pociones de maná pasa lo mismo. Implementando el sistema de esta forma evitamos que el usuario use pociones por error y que esas pociones permanezcan en el mapa por si se necesitasen más tarde.

Cada poción tiene un script “HealthPot” (o “ManaPot” en el caso de las pociones de maná) que añade a la vida o maná actual del personaje principal 50 puntos si se pueden recuperar. Si la poción es consumida el objeto se destruye.

El sistema de instanciación de objetos es un script llamado “Item Spawner” que posee un vector de objetos y otro vector de probabilidades. Al morir un enemigo aleatoriamente se intenta instanciar un objeto con una probabilidad dada. Para las pociones de vida se tendrá una probabilidad del 40% y para las de maná del 30%.

Este sistema ha sido incluído en todos los enemigos (normales y jefes) con la misma probabilidad.

13.5.1.3.3. Tarea 9.4.3: Pruebas de Unidad

Para probar las pociones se han instanciado “a mano”, se ha combatido con enemigos básicos y se ha comprobado que restauran la vida o maná sólo cuando es necesario.

Para probar el sistema de instanciación se ha combatido con 20 enemigos básicos (Skeleton Lancer). En los 20 combates se han obtenido 9 pociones de vida y 5 pociones de maná, por lo que el sistema funciona correctamente.

13.5.1.4. Subgrupo de Tareas 9.5: Pruebas

En este grupo de tareas se comprobará el sistema de objetos, las pociones de maná y vida, todos los jefes implementados y todos los enemigos básicos de la iteración anterior.

13.5.1.4.1. Tarea 9.5.1: Pruebas de Integridad

Se ha limpiado la sala de pruebas y se han instanciado al menos un enemigo de cada tipo (incluídos los jefes). Los enemigos han sido colocados en la escena en orden de implementación. Se ha dejado una distancia entre cada enemigo para poder realizar las pruebas uno a uno. Este mapa será usado en la siguiente tarea por los testeadores.

Se ha comprobado que los jefes suponen un reto muy superior en comparación con los enemigos básicos. Se ha comprobado que si muchos enemigos se juntan las probabilidades del jugador de sobrevivir bajan considerablemente.

Por lo general, los enemigos básicos son fáciles de derrotar y los jefes necesitan de estrategia y del correcto uso de las habilidades del personaje.

13.5.1.4.2. Tarea 9.5.2: Hito: Pruebas de Usuario

Estas pruebas de usuario evaluarán la dificultad de los enemigos, su aspecto gráfico y el sistema de generación de items. Para esta tarea se ha contado con 4 testeadores que han recibido el mapa anteriormente creado (la sala de pruebas con todos los enemigos).

El equipo de testeadores ha estado formado por:

- CCG (estudiante del grado de Ingeniería Informática)
- JJRG (estudiante del grado de Ingeniería Informática)
- RVA (estudiante del grado de Ingeniería Eléctrica)
- JRC (estudiante del grado de Ingeniería Mecánica)

Todos han tenido 1 semana para probar y responder el siguiente cuestionario. Como en las pruebas de usuario anteriores, se ha usado una escala del uno al cinco, donde cinco es “Sí, mucho” y uno es “No, nada”.

Cuestionario de Pruebas de Usuario - 5ª Iteración, Tarea 9.5.2					
Nº Pregunta	**Pregunta**	**CCG**	**RVA**	**JJRG**	**JRC**
1	¿La probabilidad de que los enemigos dejen una poción de vida (40%) le parece correcta?	4	5	3	5
2	¿La probabilidad de que los enemigos dejen una poción de maná (30%) le parece correcta?	5	4	5	4
3	¿Le parece correcta la cantidad de puntos de vida (50) que las pociones de vida restauran?	5	5	5	4
4	¿Le parece correcta la cantidad de puntos de magia (50) que las pociones de maná restauran?	5	5	4	5
5	¿Le parece atractivo visualmente el modelo del enemigo Skeleton Lancer?	4	4	5	4
6	¿Le ha resultado fácil de derrotar el enemigo Skeleton Lancer?	5	5	5	5
7	¿Le parece atractivo visualmente el modelo del enemigo Skeleton Highlander?	5	5	4	4
8	¿Le ha resultado fácil de derrotar el enemigo Skeleton Highlander?	3	4	5	4

9	¿Le parece atractivo visualmente el modelo del enemigo Skeleton Protector?	5	4	3	4
10	¿Le ha resultado fácil de derrotar el enemigo Skeleton Protector?	3	3	**2**	3
11	¿Le parece atractivo visualmente el modelo del enemigo Slime?	5	4	3	4
12	¿Le ha resultado fácil de derrotar el enemigo Slime?	5	5	5	5
13	¿Le parece atractivo visualmente el modelo del enemigo Black Slime?	5	4	4	4
14	¿Le ha resultado fácil de derrotar el enemigo Black Slime?	3	3	3	5
15	¿Le parece atractivo visualmente el modelo del enemigo Mud Hunter?	4	5	5	4
16	¿Le ha resultado fácil de derrotar el enemigo Mud Hunter?	4	3	5	4
17	¿Le parece atractivo visualmente el modelo del enemigo Mud Opressor?	3	5	4	3
18	¿Le ha resultado fácil de derrotar el enemigo Mud Opressor?	3	3	3	3
19	¿Le parece atractivo visualmente el modelo del enemigo Goblin Snatcher?	5	4	4	5
20	¿Le ha resultado fácil de derrotar el enemigo Goblin Snatcher?	4	5	4	4
21	¿Le parece atractivo visualmente el modelo del enemigo Wolf?	3	5	3	3
22	¿Le ha resultado fácil de derrotar el enemigo Wolf?	4	4	4	5
23	¿Le parece atractivo visualmente el modelo del enemigo Troll?	5	5	5	5
24	¿Le ha resultado fácil de derrotar el enemigo Troll?	**2**	**2**	3	**2**
25	¿Le ha resultado fácil de derrotar el enemigo Cyclops?	4	4	5	4

26	¿Le parece atractivo visualmente el modelo del enemigo Cyclops?	**2**	4	**2**	4
27	¿Le parece atractivo visualmente el modelo del enemigo Flesh Golem?	4	4	3	5
28	¿Le ha resultado fácil de derrotar el enemigo Flesh Golem?	4	3	3	4
29	¿Le parece atractivo visualmente el modelo del enemigo Earth Golem?	5	**2**	3	4
30	¿Le ha resultado fácil de derrotar el enemigo Earth Golem?	3	3	**2**	**2**
31	¿Le parece atractivo visualmente el modelo del enemigo Skeleton Mage?	5	5	5	5
32	¿Le ha resultado fácil de derrotar el enemigo Skeleton Mage?	4	3	4	4
33	¿Le parece atractivo visualmente el modelo del enemigo Red Slime?	**2**	4	3	**2**
34	¿Le ha resultado fácil de derrotar el enemigo Red Slime?	5	5	4	5
35	¿Le parece atractivo visualmente el modelo del enemigo CaveWorm?	5	4	4	5
36	¿Le ha resultado fácil de derrotar el enemigo CaveWorm?	4	5	5	5
37	¿Le parece atractivo visualmente el modelo del enemigo jefe Burp, Slime King?	4	5	4	5
38	¿Le ha resultado fácil de derrotar el enemigo jefe Burp, Slime King?	3	5	5	5
39	¿Le han resultado interesantes las mecánicas del jefe Burp, Slime King?	3	3	**2**	3
40	¿Le parece atractivo visualmente el modelo del enemigo jefe Gulp, Slime Queen?	3	4	4	3

41	¿Le ha resultado fácil de derrotar el enemigo jefe Gulp, Slime Queen?	5	5	5	4
42	¿Le han resultado interesantes las mecánicas del jefe Gulp, Slime Queen?	3	4	3	3
43	¿Le parece atractivo visualmente el modelo del enemigo jefe Sir Thomas II, Relentless Commander?	5	5	5	5
44	¿Le ha resultado fácil de derrotar el enemigo jefe Sir Thomas II, Relentless Commander?	3	3	3	**2**
45	¿Le han resultado interesantes las mecánicas del jefe Sir Thomas II, Relentless Commander?	5	4	5	5
46	¿Le parece atractivo visualmente el modelo del enemigo jefe Abramelin, The Dark Mage?	5	5	5	4
47	¿Le ha resultado fácil de derrotar el enemigo jefe Abramelin, The Dark Mage?	**2**	3	4	3
48	¿Le han resultado interesantes las mecánicas del jefe Abramelin, The Dark Mage?	4	5	5	5
49	¿Le parece atractivo visualmente el modelo del enemigo jefe Nightmare?	3	5	4	4
50	¿Le ha resultado fácil de derrotar el enemigo jefe Nightmare?	4	**2**	3	**2**
51	¿Le han resultado interesantes las mecánicas del jefe Nightmare?	**2**	3	3	3
52	¿Le parece atractivo visualmente el modelo del enemigo jefe SkullCrusher?	4	4	5	5
53	¿Le ha resultado fácil de derrotar el enemigo jefe SkullCrusher?	**2**	**1**	**2**	**2**
54	¿Le han resultado interesantes las mecánicas del jefe SkullCrusher?	5	3	4	4

Tabla 28 Cuestionario Pruebas de Usuario Iteración 5, Tarea 9.5.2

Como era de esperar las preguntas que han recibido las puntuacionaes más bajas son las referentes a la dificultad de los enemigos (pregunta 53, 50 y 47 por ejemplo). El objetivo no es hacer un juego fácil ya que es importante proponer retos al jugador desde los primeros minutos de partida. No se ha contemplado las configuraciones en los puntos de característica usados por estos testeadores, pues el cuestionario se alargaría demasiado.

Como puntos postivos se menciona la buena aceptación de los modelos 3D escogidos para la realización de este proyecto (preguntas 7, 13 o 15 entre otras) y de las mecánicas de algunos enemigos como Abramelin, The Dark Mage.

El sistema de generación de objetos y la cantidad de vida y maná restaurada tiene muy buena aceptación por parte de los testeadores (preguntas 1, 2, 3 y 4).

No se han encontrado condiciones anómalas en el sistema por parte de los testeadores, en gran parte debido a la ingente realización de pruebas de unidad e integridad.

13.6. Iteración 6 (del 11-04-2016 al 24-04-2016)

Esta es la iteración final del proyecto. En ella se desarrollan los últimos aspectos estéticos del prototipo, se crea la escena Demo y se cierra el proyecto generándose el entregable final.

*13.6.1. **Grupo de Tareas 10**: Sistema de Keybinds (Configuración de teclas del usuario)*

En este grupo de tareas se diseñará y creará el sistema de configuración de teclas del usuario. Este sistema es la única característica funcional aportada durante esta iteración.

13.6.1.1. Tarea 10.1: Diseño

El usuario podrá elegir qué teclas del teclado definir para el uso de las habilidades especiales. El usuario modificará su configuración de teclas en el menú opciones y esas teclas se mapearán automáticamente para su correcto

funcionamiento. Las teclas predefinidas son Q, W, E y R. Para evitar comportamientos anómalos no se podrá mapear teclas numéricas o de símbolos para cada habilidad. Si se intenta asignar una tecla ya asignada, la primera asignación volverá al estado predefinido. Las teclas O, S, L y P (las cuales corresponden a los atajos de teclado de los menús) no podrán ser usadas para el mapeo de habilidades.

13.6.1.2. Tarea 10.2: Implementación

Este sistema está basado en el patrón de diseño “Command” . Como el sistema no es muy complejo podemos obviar el uso de interfaces. En el script “Combat” del personaje principal se definen las habilidades especiales como funciones únicas y se ha implementado un mapeado de teclas dinámico para el uso de las habilidades especiales. Con esto se gana uniformidad a la hora de invocar a las funciones de las habilidades especiales. Se ha modificado la interfaz de usuario para que muestre las teclas configuradas correctamente en las habilidades especiales.

Ilustración 96 Sistema de Keybinds configurado para usar Z, X, C y V

Ilustración 97 Interfaz de Usuario mostrando unas nuevas asignaciones de teclas

13.6.1.3. Tarea 10.3: Pruebas de Unidad

Siguiendo con el proceso habitual de prueba se ha comprobado que el sistema de configuración de teclas de usuario funciona correctamente. Las teclas protegidas de asignación (O, S, L y P) no pueden asignarse. Sólo se pueden asignar teclas de letras (no numéricas ni de símbolos). La interfaz muestra correctamente los cambios en el sistema.

13.6.2. **Grupo de Tareas 11**: Feedback Usuario

Este grupo de tareas engloba actividades que pretenden mejorar la experiencia de usuario a través de retroalimentación y efectos visuales que lo ayuden a cerciorarse de eventos importantes dentro del juego.

13.6.2.1. Subgrupo de Tareas 11.1: Puntos de Característica sin Asignar

13.6.2.1.1. Tarea 11.1.1: Diseño

Cuando se tienen puntos de característica sin asignar es difícil darse cuenta de ello sin acceder al menú Stats. Por lo que se ha decidido resaltar el botón de dicho menú para alertar al usuario de este hecho.

13.6.2.1.2. Tarea 11.1.2: Implementación

La implementación de este comportamiento es bastante simple. El GUIController (controlador) tendrá acceso al personaje principal (modelo) y reflejará la existencia de puntos sin asignar en la interfaz de usuario (vista).

Ilustración 98 Acceso al menú Stats resaltado, indicando que se tienen puntos de característica sin asignar

13.6.2.1.3. Tarea 11.1.3: Pruebas de Unidad

Se ha comprobado que si se inicia una escena con puntos sin asignar aparece directamente resaltado este botón. Cuando se sube de nivel se resalta, al tener puntos disponibles. Si usamos en botón para recuperar los puntos y volver a distribuirlos el botón se resalta como era de esperar.

13.6.2.2. Subgrupo de Tareas 11.2: Números de Daño de Combate, Vida y Maná

13.6.2.2.1. Tarea 11.2.1: Diseño

Para mejorar la experiencia de juego se a adoptado un sistema de texto flotante muy parecido al de grandes juegos de género como Diablo 3 y World of Warcraft entre otros.

Ilustración 99 Texto flotante de daño en Diablo 3

También se implementará un efecto de brillo rojo al tener seleccionado a un enemigo para mejorar el conocimiento de este hecho por parte del usuario. Se ha decidido cambiar la barra de vida de los jefes para que sea algo más llamativa.

13.6.2.2.2. Tarea 11.2.2: Implementación

Para implementar este efecto, se han implementado 3 objetos (Prefabs) para el daño, vida y maná que contienen un scripty un GUIText. Se ha usado la fuente comic sans. Estos scripts controlan el tiempo de vida (0.6 segundos) y la dirección en la que el texto se mueve (siempre hacia arriba).

Al hacer daño, ganar vida o ganar maná se instanciará el objeto correspondiente en la pantalla, a través de una función de Unity que proyecta un punto del mundo en el viewport de la cámara.

Ilustración 101 Texto flotante de ganancia de vida

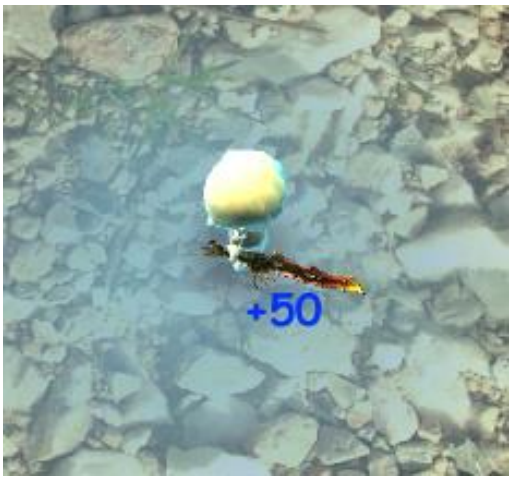


Ilustración 100 Texto flotante de ganancia de magia

Ilustración 102 Texto flotante de daño y efecto glow en Skeleton Lancer

Para dar un efecto de brillo rojizo a los enemigos se ha usado un shader descargado gratuítamente de un canal de Youtube (“Creando User FeedBack en Unity 4.5-6 Shaders Para Generar Efecto de Glow en los Materiales”, Avidosgamer). Este shader ha sido aplicado a todos los materiales de los enemigos para generar unos nuevos con este efecto. Cuando un enemigo es seleccionado cambi sus materiales por los nuevos.

Ilustración 103 Skeleton Lancer normal y con efecto Glow

Ilustración 104 Barra de vida de Jefes

Para la barra de vida de los jefes se ha usado una textura descargada gratuítamente de opengameart.org (loading-bar, StumpyStrust).

Se ha decidido incorporar un último efecto al sistema de combate. El jugador y los enemigos sangrarán cuando reciban daño.

Para cada enemigo se creará un color de sangre diferente. Este efecto ha sido descargado gratuítamente de Youtube (“Unity3D - Episode 19 - Hit Particles (Blood)!”, Adam Sims) y modificado para que se adapte a los modelos 3D propios del proyecto.

Ilustración 105 Capturas de efecto de sangrado del personaje principal

Ilustración 106 Efecto de sangrado verde del enemigo Slime

13.6.2.2.3. Tarea 11.2.3: Pruebas de Unidad

Se ha comprobado que los números flotantes se muestran correctamente y el tiempo suficiente para que el usuario se cerciore de ellos.

Se ha comporbado que los enemigos cambian su material al ser seleccionados. Después de un tiempo sin intercambio de daño o siendo deseleccionados intencionadamente vuelven a adoptar su apariencia original.

Se ha comprobado que la barra de vida de los jefes ha cambiado y que ésta se comporta de forma correcta.

13.6.2.3. Subgrupo de Tareas 11.3: Iconos del Minimapa

Como se mencionó en fases anteriores se añadirán iconos al minimapa representando jugador, enemigos y jefes.

13.6.2.3.1. Tarea 11.3.1: Diseño

Dado que el número de enemigos en pantalla puede ser bastante alto, se ha decidido usar icono pequeños pero muy representativos.

Para el jugador se usará una flecha verde (icons.mysitemyway.com, del generador de iconos, pack glossy silver icons arrows) que, además de mostrar la posición del jugador en el mapa, indicará la dirección qhacia la que nos dirijimos.

Para los enemigos simples usaremos un punto rojo, y para los jefes un gran punto negro con una cavalera (devianart.com, Skull icon by Sergioelmoreno).

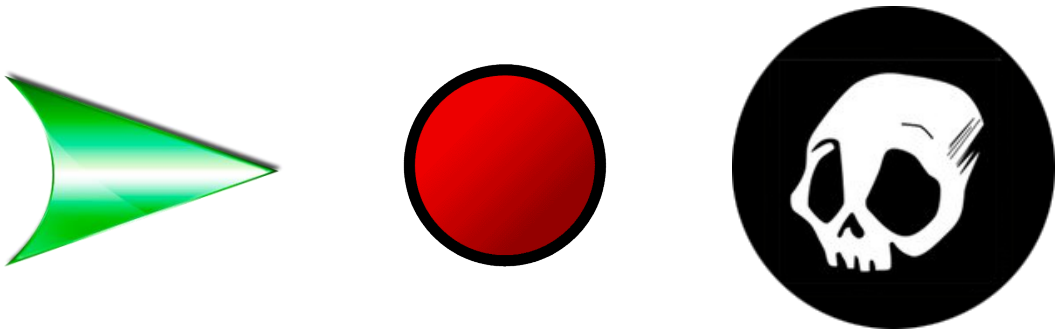


Ilustración 107 Diferentes iconos del Minimapa

13.6.2.3.2. Tarea 11.3.2: Implementación

Estos iconos han añadido a la parte superior de los agentes. Al icono del jugador se le ha añadido una luz (Point Light) para dé sensación de brillo.

Para implementar el efecto se usarán dos Layers. Estos layers permitirán mostrar por diferentes cámaras los elementos que pertenezcan a un layer en concreto o no.

Definimos un layer llamado NPC en el que sólo encuentran los modelos 3D de los enemigos. Definimos otro layer llamado GUI en el que sólo se encuentran los iconos.

En la opción “Culling Mask” de la cámara principal deseleccionamos el layer GUI. En la opción “Culling Mask” de la cámara del minimapa deseleccionamos el layer NPC. Así, en la cámara principal no se verán los iconos y en el minimapa no se verán los modelos 3d de los enemigos.

A continuación se muestra el efecto conseguido.

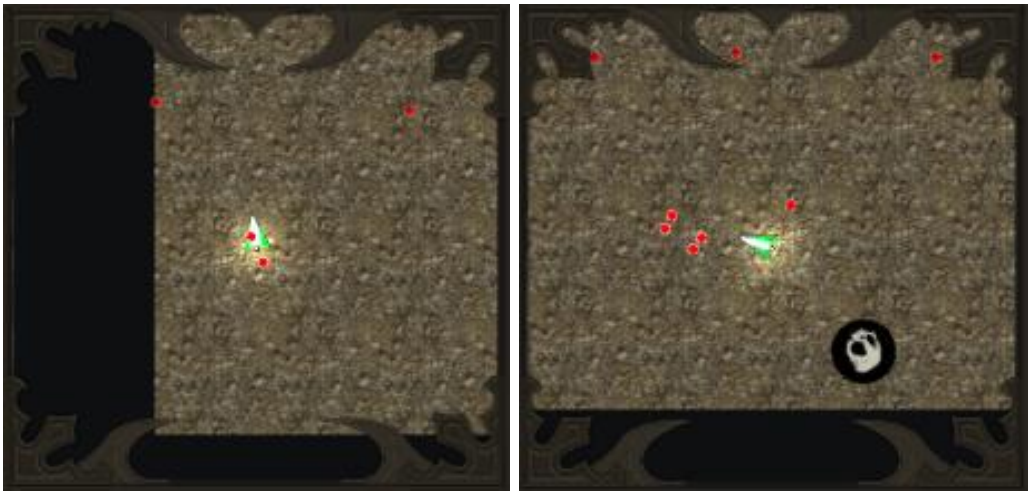


Ilustración 108 Ejemplos de Minimapa con iconos representando a los agentes

13.6.2.3.3. Tarea 11.3.3: Pruebas de Unidad

Se ha comprobado que todos los agentes poseen un icono y que éste representa su posición en el mapa en todo momento. Se ha ajustado el tamaño de los iconos y la luz del icono del jugador para que se renderizen correctamente en el minimapa.

13.6.2.4. Subgrupo de Tareas 11.4: Transiciones entre Escenas

La transición entre escenas es algo brusco, ya que cuando se realiza el cambio la pantalla queda completamente negra y de repente se carga la siguiente escena.

13.6.2.4.1. Tarea 11.4.1: Diseño

Se quiere conseguir un efecto de difuminación que suavice las transiciones entre escenas y la pantalla de carga.

13.6.2.4.2. Tarea 11.4.2: Implementación

Para implementar este efecto se ha usado un script, incluído en la cámara principal, descargado de un video de Youtube (How to Fade Between Scenes in Unity, Brackeys) que dada una textura y una velocidad ya realiza esta animación. Invocando al método “BeginFade” con el parámetro 1 (para hacer fade in) o -1 (para hacer fade out) comenzará el efecto.

La animación “Fade In” a partir de un fondo negro va mostrando poco a poco la imagen. La animación “Fade Out” va difuminando la imagen hasta llegar a un fondo negro .

Con Photoshop se ha creado una imagen con el texto “loading” que será usada para la transición entre escenas.

Ilustración 109 Texto de transición entre escenas

Simplemente invocando al método anteriormente citado antes de cargar y abandonar una escena habrá finalizado la implementación.

13.6.2.4.3. Tarea 11.4.3: Pruebas de Unidad

Se ha comprobado que se realiza el fade in y el fade out correctamente recargando la escena “Sala de Pruebas”. Al ejecutar el juego fuera del editor también se realiza el fade in y fade out al reiniciar el nivel.

13.6.2.5. Subgrupo de Tareas 11.5: Subidas de Nivel

13.6.2.5.1. Tarea 11.5.1: Diseño

Para las subidas de nivel se ha pensado en algún efecto que envuelva al personaje en energía durante unos segundos y que sea uno de los efectos más notorios dentro del juego.

13.6.2.5.2. Tarea 11.5.2: Implementación

Usando un sistema de partículas (Flame Enchant) del pack gratuíto Elementals y el shader del efecto glow para generar un material para el personaje principal de acuerdo al color de la llama se ha conseguido un efecto muy llamativo.

Ilustración 110 Efecto de subida de nivel

Ilustración 111 Efecto Subida de nivel detalle

Al subir de nivel se intercambian los materiales del personaje principal y se activa el sistema de partículas incluído en el personaje durante 5 segundos. Transcurrido ese tiempo, el sistema de partículas se desactiva y se vuelven a aplicar los materiales por defecto.

13.6.2.5.3. Tarea 11.5.3: Pruebas de Unidad

Se ha combatido con 15 enemigos en la sala de pruebas para alcanzar el nivel 12 y así comprobar que los materiales se intercambian adecuadamente y que el sistema de partículas se activa y desactiva correctamente.

13.6.2.6. Subgrupo de Tareas 11.6: Pruebas

Este subgrupo de tareas comprende las pruebas llevadas a cabo por el equipo de desarrollo (Pruebas de Integridad del sistema) y por el equipo de testeo (Pruebas de Usuario).

13.6.2.6.1. Tarea 11.6.1: Pruebas de Integridad

Se ha usado la sala de pruebas que se utilizó en las pruebas de usuario de la iteración 5 (*Tarea 9.5.2: Hito: Pruebas de Usuario*). Se ha comprobado que el sistema de configuración de teclas de usuario funciona correctamente. Se ha comprobado que la retoralimentación al usuario es correcta (puntos de característica sin asignar, iconos del minimapa y el efecto de subida de nivel). El efecto "Fading" de las transiciones entre escenas se realiza correctamente y cumple con el cometido de suavizar dichas transiciones.

13.6.2.6.2. Tarea 11.6.2: Hito: Pruebas de Usuario

Para que los testeadores puedan probar todas las funcionalidades y efectos implementados se ha creado un ejecutable para Windows (x64) de la sala de pruebas usada en las pruebas de integridad.

El equipo de testeadores estará formado en esta última iteración por:

- CCG (estudiante del grado de Ingeniería Informática)
- JJRG (estudiante del grado de Ingeniería Informática)

A continuación se muestra el cuestionario que los testeadores han cumplimentado después de probar el videojuego 4 horas durante 1 semana.

Cuestionario de Pruebas de Usuario - 6ª Iteración, Tarea 11.6.2			
Nº Pregunta	**Pregunta**	**CCG**	**JJRG**
1	¿Le parece útil el sistema de configuración de teclas del usuario (Sistema de Keybinds)?	4	4
2	¿Le parece cómodo de usar el sistema de configuración de teclas del usuario (Sistema de Keybinds)?	5	4
3	¿Le parece útil que el acceso al menú "Stats" sea resaltado si existen puntos de característica sin asignar?	5	5

4	¿Le parece que el resaltado anteriormente mencionado es lo suficientemente llamativo?	4	4
5	¿Le resultan de ayuda los números flotantes de daño, vida y maná?	5	5
6	¿Le parecen lo suficientemente grandes los números flotantes de daño, vida y maná?	4	5
7	¿El minimapa le resulta de más ayuda ahora que tiene iconos?	5	5
8	¿Distingue bien los diferentes iconos existentes en le minimapa?	5	5
9	¿Le parece suave el cambio entre escenas (en las pruebas recargar nivel)?	5	5
10	¿Le parece lo suficientemente notorio el efecto visual de subida de nivel?	5	5

Tabla 29 Cuestionario de Pruebas de Usuario - 6ª Iteración, Tarea 11.6.2

Como se puede apreciar en el cuestionario, todas las características desarrolladas en esta parte de la iteración han sido probadas satisfactoriamente y han cumplido con las espectativas de testeadores y equipo de desarrollo.

Como la mayoría de estas características son simplemente efectos visuales, es normal que no se encuentren condiciones anómalas en el funcionamiento del prototipo.

*13.6.3. **Grupo de Tareas 15**: Demo*

En este punto que ya se tienen todas las características desarrolladas se va a construir la escena Demo y la pantalla de Título.

13.6.3.1. Tarea 15.1: Diseño

No es necesario esbozar la escena de título ya que sólo conta de un botón para jugar y otro para cerrar la aplicación.

Para la escena Demo se ha usado Photoshop para bocetar la ruta que debería seguir el usuario y los enemigos de los que constará dicha escena.

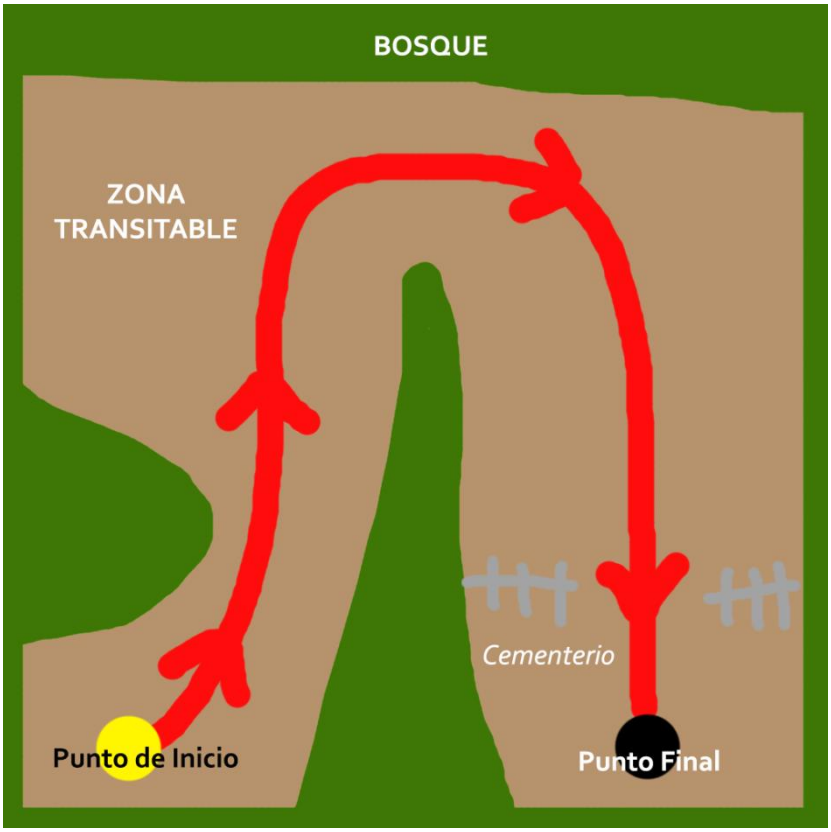


Ilustración 113 Esbozo del mapa Demo y la ruta de juego

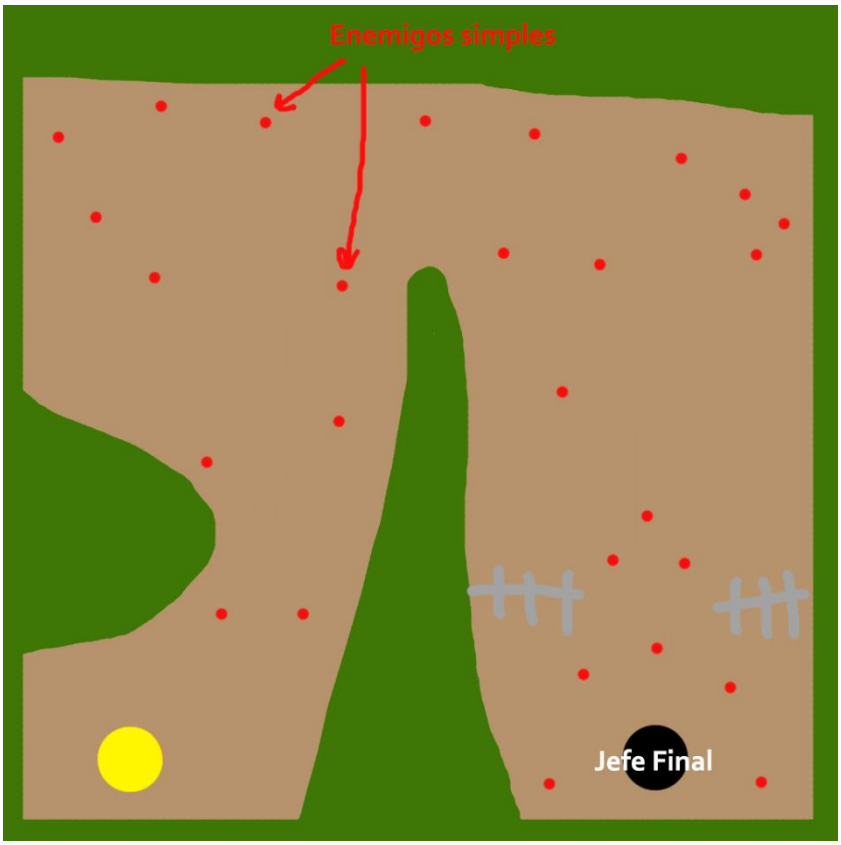


Ilustración 112 Esbozo de la posición de los diferentes enemigos de la escena Demo

Como podemos observar en esta última imagen, la escena contará con 27 enemigos comunes y 1 sólo jefe final.

El esbozo de la ruta de juego puede parecer simple, pero es lo suficientemente complejo para probar este prototipo.

13.6.3.2. Tarea 15.2: Implementación

Para la escena de título se ha usado un par de botones de la UI de Unity y se han creado las letras de título con Photoshop. Se a decidido usar el modelo de la espada que porta el personaje para que la escena de título no quede tan vacía. Se ha incluído en la punta de la espada un sistema de partículas que proporciona un efecto de chispas de soldadura para que la escena no sea tan estática.

No es parte de los requisitos, pero se ha encontrado una canción gratuíta en opengameart.org (Soloquy, matthew.pablo) que será añadida a la escena de título y reproducida en bucle. Se ha añadido un sonido gratuíto al botón “Play” (Heal8-bit.ogg, autor: Kizul Emeraldfire) de rpg.hamsterrepublic.com

Ilustración 114 Escena de Título

Para crear la escena Demo, se han usado texturas de un pack gratuíto (Yughues Free Ground Materials) y una colección de assets gratuítos (Make Your Fantasy Game - Lite), ambos de Unity Store.

También se usarán modelos de árboles descargados de la cuenta github de Trent Willis (pack Trees Ambient-Occlusion).

Se ha abierto la escena vacía creada en las primeras tareas del proyecto y se ha creado un terreno plano cuadrado lo suficiente grande para albergar enemigos y decoración.

Se ha añadido un música de ambiente (Radakan – old crypt.ogg, autor: Janne Hanhisuanto para Radakan) descargada de opengameart.org

Se han aplicado 3 tipos de texturas diferentes para marcar el camino al jugador y dar algo de variedad al terreno.

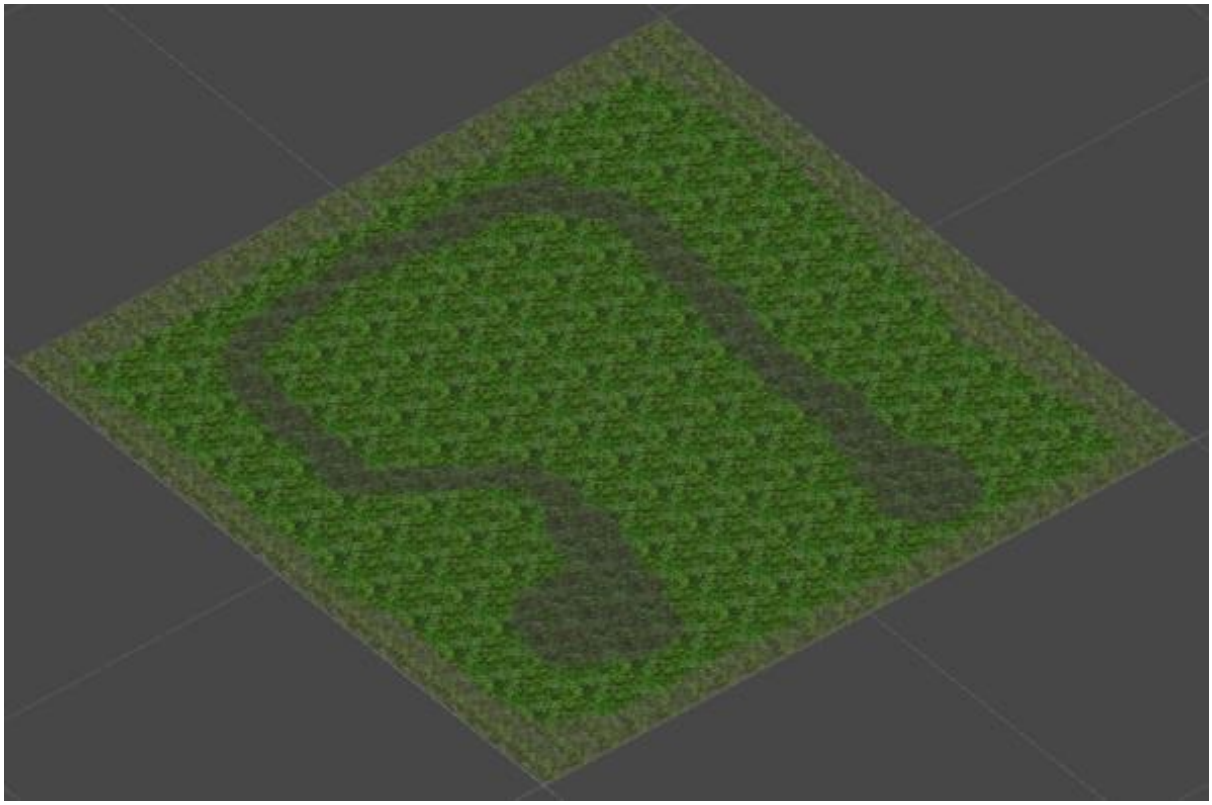


Ilustración 115 Terreno texturizado de la escena Demo

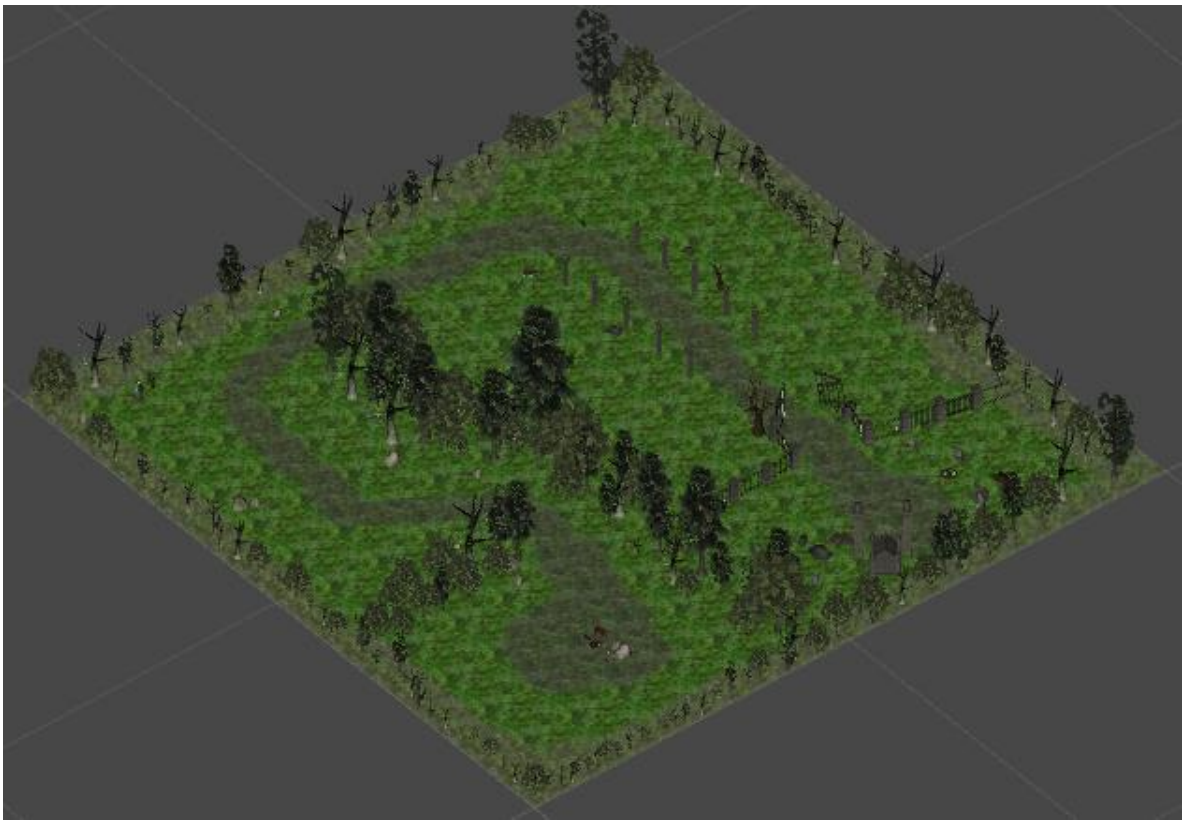


Ilustración 117 Escena Demo con los árboles y assets

Ilustración 116 Detalle cementerio escena Demo con árboles y assets

Después de terminar el mapa se realizará un bake de las escena para que los agentes puedan navegar por ella. Los árboles no serán transitables, en su lugar se usarán para limitar el paso del personaje principal y enemigos.

El jefe final es Sir Thomas II, y se ha añadido a la escena la mayoría de los enemigos básicos. Los demás jefes no aparecen en esta escena, pero añadirlos en un futuro es tán fácil como arrastrar su prefab dentro de la misma.

A continuación se muestran algunas capturas del prototipo. Para observar los detalles con más precisión se recomienda visualizar el video demostrativo adjunto.

Ilustración 118 Captura 1 de la escena Demo

Ilustración 120 Captura 2 de la Escena Demo

Ilustración 119 Captura 3 de la Escena Demo

Ilustración 122 Captura 4 de la Escena Demo

Ilustración 121 Captura 5 de la Escena Demo

Ilustración 123 Captura 6 de la Escena Demo

13.6.3.3. Tarea 15.3: Pruebas de Usuario

Para esta última fase de Pruebas de usuario se ha creado un cuestionario muy corto relacionado con la escena de título y la escena Demo.

Los testeadores con los que han contado para esta fase son:

- CCG (estudiante del grado de Ingeniería Informática)
- JJRG (estudiante del grado de Ingeniería Informática).

Se ha creado un ejecutable con la escena de título y escena Demo y éste ha sido entregado a los testeadores para que puedan probar el prototipo durante 1 semana.

Cuestionario de Pruebas de Usuario - 6ª Iteración, Tarea 15.3			
Nº Pregunta	**Pregunta**	**CCG**	**JJRG**
1	¿Le parece atractiva visualmente la pantalla de título?	5	5
2	¿Le parece atractivo visualmente el mapa Demo?	5	4
3	¿La dificultad del mapa Demo es demasiado alta?	2	3
4	¿El número de enemigos del mapa Demo es correcto?	3	4

Tabla 30 Cuestionario de Pruebas de Usuario - 6ª Iteración, Tarea 15.3

Al observar los resultados del formulario podemos afirmar que la escena de Título y el mapa Demo gustan a los usuarios y que la dificultad es media.

Se ha recibido por parte del testeador Jose J. un cuestionario de condiciones anómalas detectadas.

Condiciones Anómalas detectadas - 6ª Iteración, Tarea 15.3	
Testeador:	JJRG
Desde el comienzo se puede saltar al cementerio (zona final) usando la habilidad especial de Teletransporte 2 veces. Esta habilidad hace atravesar los árboles dejar al personaje como en un hueco.	

Tabla 31 Condiciones Anómalas detectadas - 6ª Iteración, Tarea 15.3

Se ha corregido esta situación. Al hacer el “Bake” de la escena quedó un espacio entre varios árboles que permitía este comportamiento. Se ha ampliado el radio de colisión de los árboles eliminando los huecos existentes entre ellos y evitando este tipo de trampas.

13.6.4. ***Grupo de Tareas 14**: Cierre del proyecto*

Con estas tareas se da el proyecto por incluído y se genera el ejecutable que se entregará junto a esta documentación.

13.6.4.1. Tarea 14.1: Generación del Ejecutable

Para generar el ejecutable final, seleccionaremos las escenas que se incluirán y su orden de aparición. Como es lógico, la primera escena que se cargará será la escena de título y posteriormente la escena Demo. No será necesario incluir la escena de la sala de pruebas en el ejecutable ya que no habrá forma de acceder a ella y no es conveniente que el usuario final lo haga puesto que pueden existir errores. El ejecutable será renombrado como Flame Knights Chronicles.exe y no se le definirá ningún icono.

13.7. Seguimiento y Control del Proyecto

Como anteriormente se mencionó en el apartado *12.2 Estimación de la Duración* el proyecto ha sido dividido en 6 iteraciones, con una duración de 2 semanas cada una. El conjunto de tareas que comprenden el proyecto ha sido dividido en 13 grupos de tareas. Esta organización permite controlar los restasos de tiempo con un nivel de detalle mucho más exacto (por iteración y por grupo de tareas). Se incluyen 2 subapartados que resumen las compatarivas y desviaciones entre tiempo estimado y real. También existe un apartado que resume las horas invertidas por el alumno en cada tipo de tarea en concreto.

13.7.1. Comparativa entre Tiempo Estimado y Real

TABLA ITERACIONES				
ITERACIONES	**TAREAS QUE COMPRENDE (Grupos)**	**Fecha Inicio**	**Duración Estimada**	**Duración Real**
Iteración 1	1,2,3	01/02/2016	29	27
Iteración 2	4, 5, 6, 7	15/02/2016	53	55,5
Iteración 3	8	29/02/2016	46	49
Iteración 4	9.1	14/03/2016	53	34
Iteración 5	9.2, 9.3, 9.4, 9.5	28/03/2016	41	36,5
Iteración 6	10, 11, 14,15	11/04/2016	37	41
		TOTAL:	259	243

Tabla 32 Tabla de Duración Estimada y Real de Iteraciones

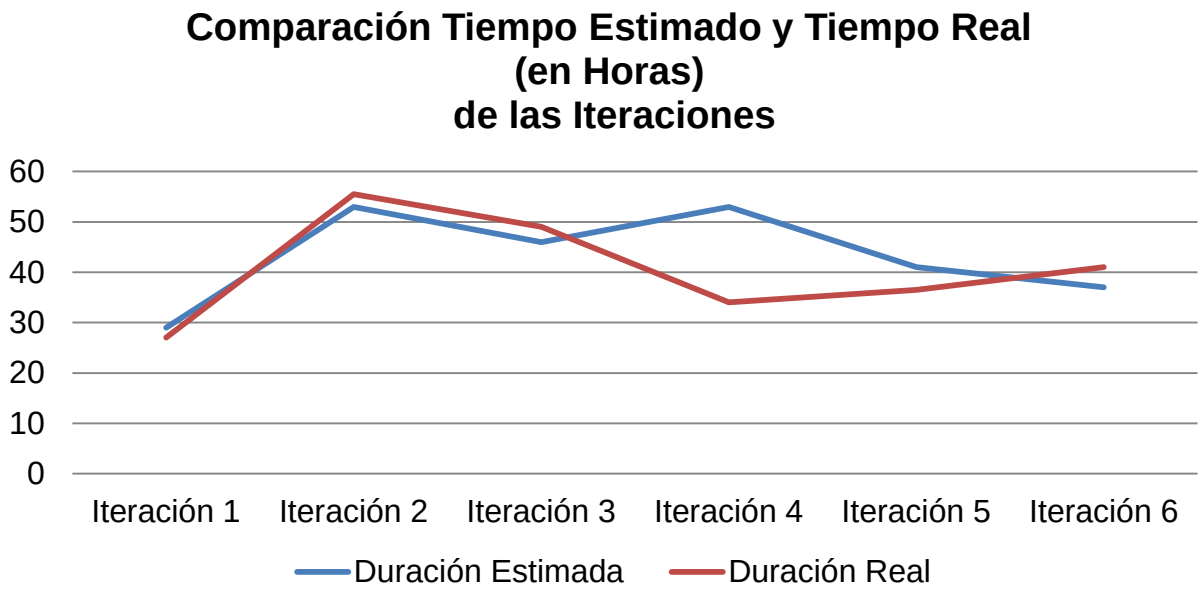


Ilustración 124 Comparación Tiempo Estimado y Real por Iteraciones

Como se refleja en el gráfico anterior, no existe una gran desviación entre el tiempo estimado y el real exceptuando la iteración 4. Esta iteración (*13.4 Iteración 4 (del 14-03-2016 al 27-03-2016)*) trata sobre la implementación de los enemigos comunes. El hecho de que el tiempo real sea menor al estimado es debido a varios motivos. Al no tener experiencia previa en modelado 3D, la estimación ha sido pesimista. También influye el hecho de haber usado modelos ya animados de terceros de gran calidad.

Estimación por Grupo de Tareas		
Grupo de Tareas	**Duración Estimada**	**Duración Real**
1	2	1,5
2	8	8
3	19	17,5
4	6	5,5
5	3,5	4
6	10	10,5
7	33,5	35,5
8	46	49
9	94	70,5
10	5	4
11	17	15,5
15	14	20,5
14	1	1
TOTAL:	259	243

Tabla 33 Tabla de Duración Estimada y Real de los Grupos de Tareas

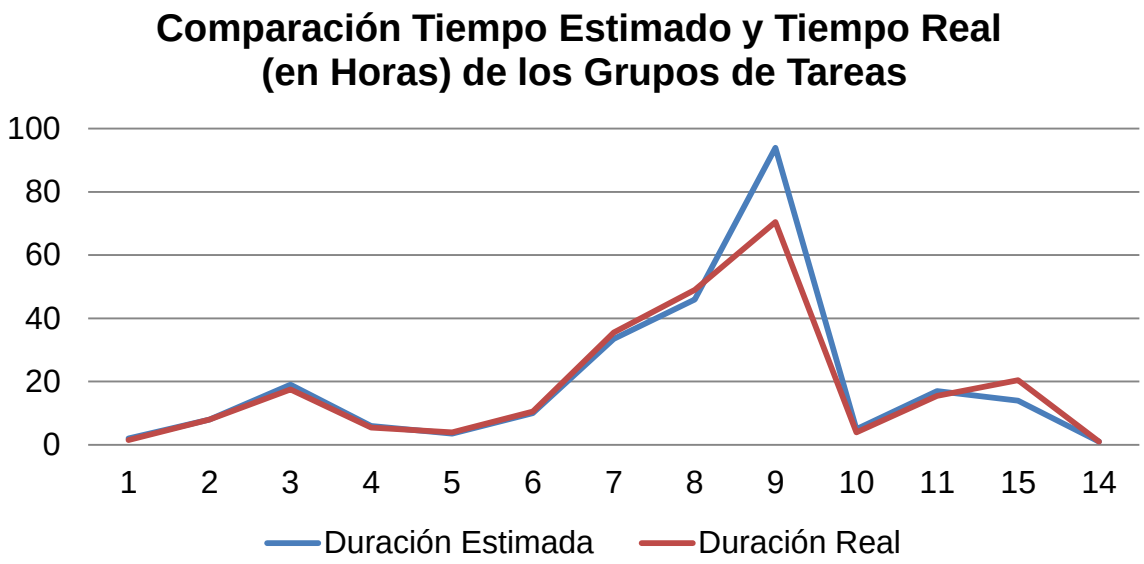


Ilustración 125 Comparación Tiempo Estimado y Real por Grupos de Tareas

En esta ilustración podemos apreciar con más detalle las desviaciones de tiempo de las tareas del proyecto. Como se ha mencionado anteriormente, la tarea 9 (*13.4.1*) que está distribuída en las iteraciones 4 y 5 tiene un tiempo real menor al estimado.

También se observa que al grupo de tareas 15 (*13.6.3*) referente a la creación de la escena Demo se le ha dedicado algo más de tiempo que el estimado. Esto es en gran parte debido a la complejidad de la escena.

13.7.2. Desviación de horas entre Tiempo Estimado y Real

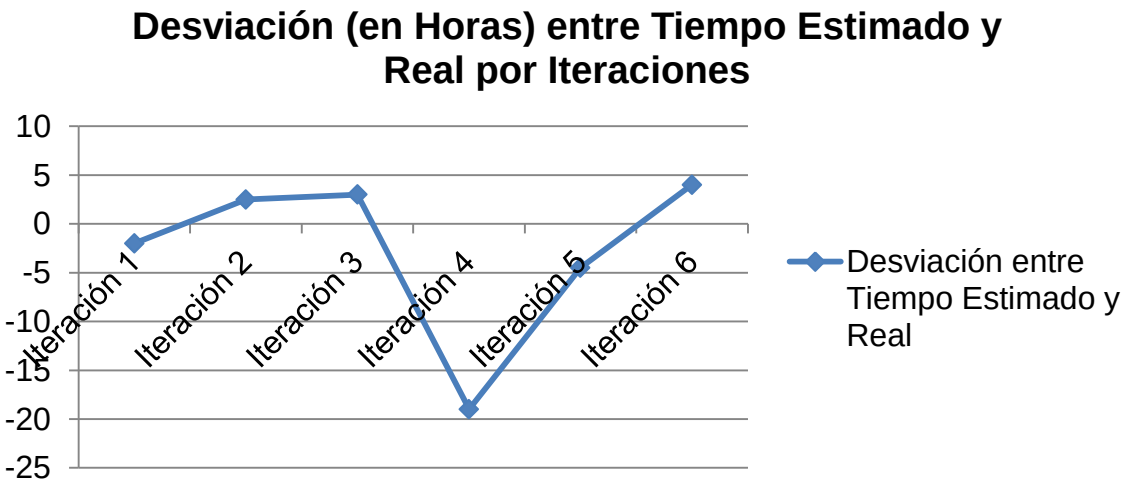


Ilustración 126 Desviación por Iteraciones (El tiempo negativo significa un adelanto conforme lo estimado)

Las iteraciones 2 ,3 y 6 han tenido un ligero retraso de entre 2 y 4 horas. La iteración 4 se ha adelantado unas 19 horas. Este adelanto ha sido justificado en el apartado anterior. No existen grandes desviaciones que sean injustificables.

Para un nivel de detalle superior, se muestra a continuación el mismo gráfico para los grupos de tareas.

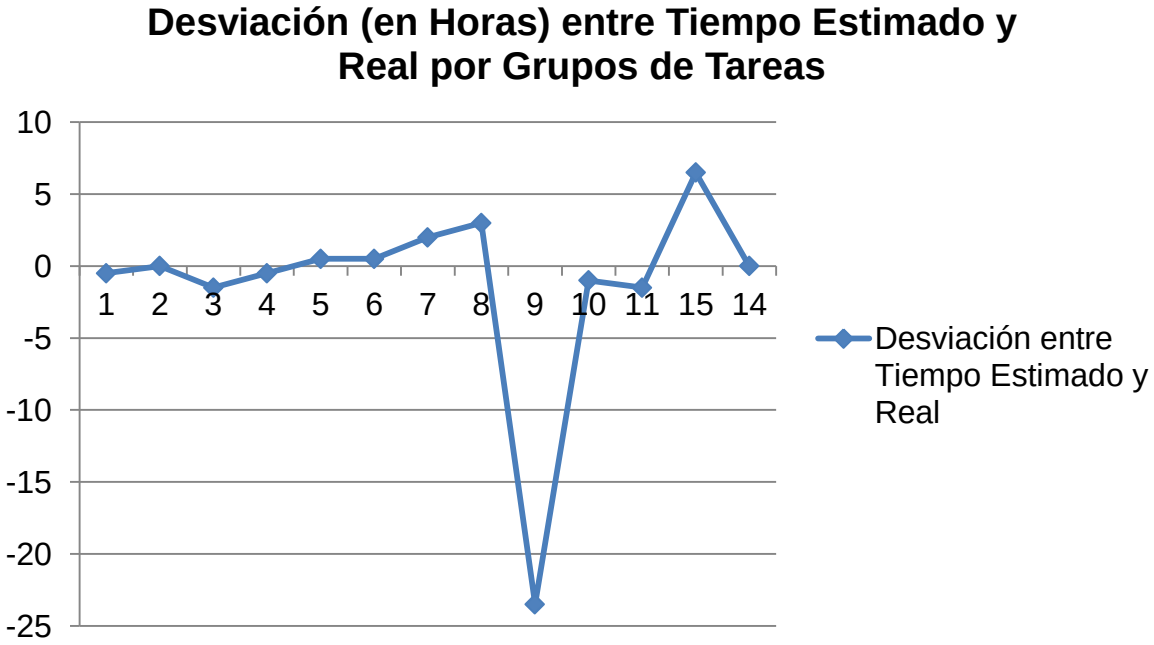


Ilustración 127 Desviación por Grupos de Tareas (El tiempo negativo significa un adelanto conforme lo estimado)

13.7.3. Horas invertidas por Tipo de Tarea

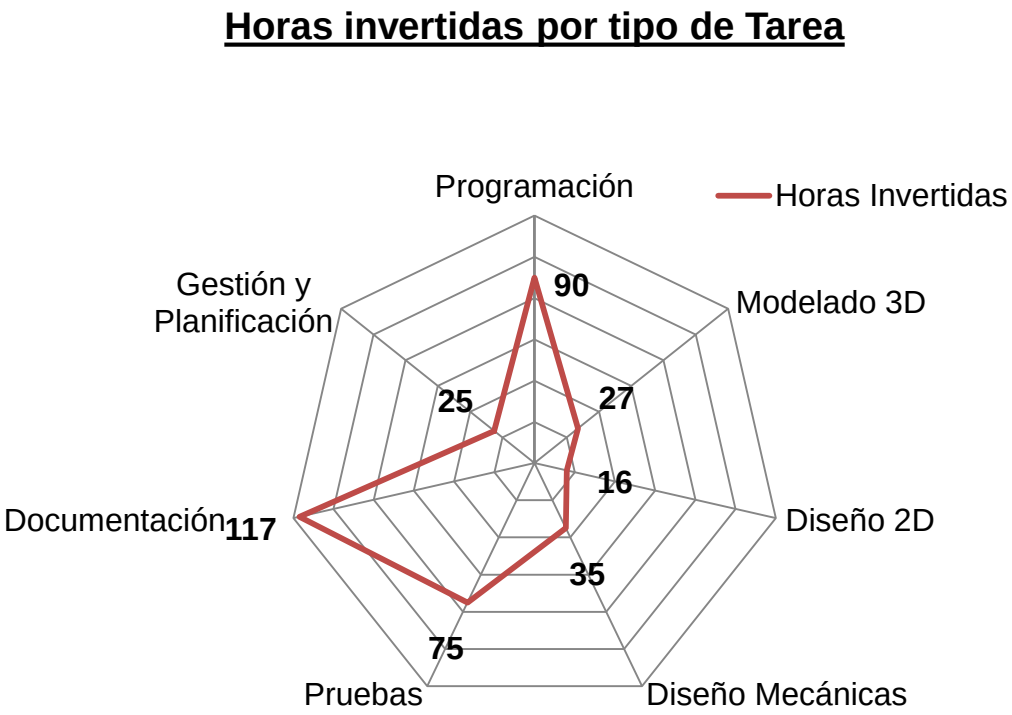


Ilustración 128 Horas invertidas por tipo de Tarea

Con un total de **387 horas**, el gráfico anterior muestra la distibución de tiempo invertido durante todo el proyecto. Las realización de las tareas de las iteraciones (Programación, Modelado 3D, Diseño 2D, Diseño Mecánicas y Pruebas) hacen un total de 243 horas, por lo que el 37,2% restante del tiempo total ha sido dedicado a tareas de Gestión, Planificación y Control del proyecto y a la elaboración de esta documentación.

14. FINALIZACIÓN DEL PROYECTO

Al final del proyecto se debe generar un informe dirigido al cliente y otro dirigido al equipo de desarrollo. El informe para el cliente es una memoria del proyecto (*Capítulo 13 EJECUCIÓN DEL PROYECTO*) y un resumen de las funcionalidades implementadas (*Capítulo 7 MECÁNICAS DE JUEGO Y CARACTERÍSTICAS*). El siguiente apartado está dirigido al equipo de desarrollo como forma de retroalimentación para una mejora en la ejecución de similares proyectos en el futuro.

14.1. Informe para el Equipo de Desarrollo

14.1.1. Técnicas utilizadas para conseguir los objetivos propuestos

- Se ha seguido exitosamente el típico esquema de actuación de tres pasos diseño, implementación y prueba.
- Todas las funcionalidades han sido probadas individualemente (en su fase de pruebas correspondiente), antes de realizar pruebas de integración en el sistema completo.
- Se han realizado pruebas de usuario con personas ajenas al equipo de desarrollo para recoger opiniones y críticas objetivas.
- Debido a las debilidades del equipo a la hora de modelar, rigear y animar los agentes se han usado unos gratuítos.
- Se han aplicado patrones tales como el patrón “Composite” (Componentes) en todo el proyecto (en los agentes sobre todo), el patrón “Command” (*13.6.1 Grupo de Tareas 10: Sistema de Keybinds (Configuración de teclas del usuario)*) y el patrón arquitectónico Modelo-Vista-Controlador (*13.2.4 Grupo de Tareas 7: GUI*).
- Cada día se han realizado copias de seguridad en un disco duro externo y en Google Drive.
- Se han usado numerosas fuentes de información para investigar cómo implementar una característica en concreto (*0*

- *FUENTES DE* INFORMACIÓN USADAS).

14.1.2. Fortalezas y Debilidades manifestadas durante el desarrollo

14.1.2.1. Fortalezas

- Capacidad de resolución de problemas ante nuevas situaciones.
- Rápida adaptación al uso de nuevas tecnologías y herramientas.
- Constancia en el proceso de desarrollo.
- Correcta aplicación del modelo de ciclo de vida de la gestión del proyecto.
- Decente estimación de la duración del proyecto.
- Colaboración con personas externas al equipo para la realización de pruebas de usuario.
- Estructuración del flujo de trabajo válida en la ejecución del proyecto.

14.1.2.2. Debilidades

- Poca experiencia en tareas de modelado 3D, técnicas de rigging y animación.

14.1.3. Recomendaciones al equipo

- Sería beneficioso formar al equipo de desarrollo en tareas de modelado 3D, técnicas de rigging y animación o delegar estas tareas en personal cualificado.
- Seguir usando las técnicas que han funcionado, reforzando los puntos fuertes del equipo de desarrollo.

15. DIAGRAMAS REPRESENTATIVOS

Aquí se presentan los diagramas generados para clarificar la estructura que tienen los elementos desarrollados en el sistema.

15.1. Diagrama de Componentes: Jugador

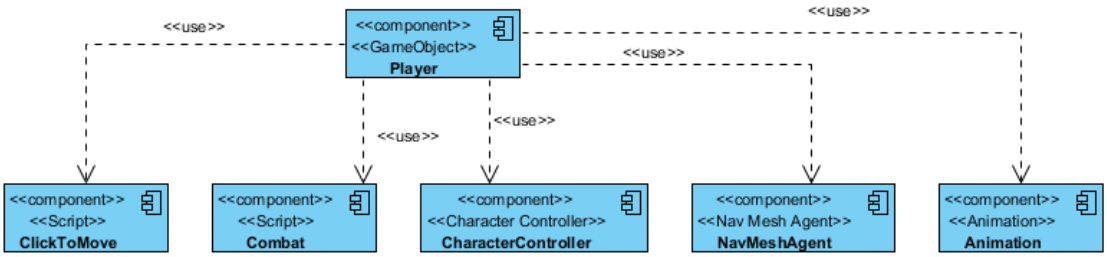


Ilustración 129 Diagrama de Componentes del Jugador

Player: El objeto que representa la entidad sobre la que tendrá el control el jugador. Contiene todos los componentes funcionales que lo forman, además del modelo 3D, animaciones e iluminación.

ClickToMove: Script en C# que implementa la funcionalidad de que el personaje se dirija a un punto del mapa cuando se hace click derecho sobre una localización a la que está permitida desplazarse.

Combat: Script en C# que contiene las estadísticas, las funciones de ataque y una parte del sistema de “keybinds” del jugador. También tiene funciones que se ejecutan periódicamente en segundo plano (concurrentemente en otro hilo) que se encargan de la regeneración de vida y maná a lo largo del tiempo.

CharacterController: Componente propio de Unity que no usa las físicas del motor. Controla la pendiente que el personaje podrá subir o no y su “hitbox” en el mundo 3D.

NavMeshAgent: Componente que, añadido a un personaje que pueda moverse, permite generar automáticamente rutas desde el punto en el que se encuentre hasta la nueva posición y que las recorra. El terreno debe tener una malla de navegación (Navigation Mesh) para definir las partes transitables y los límites.

En este componente se define la velocidad de movimiento, la velocidad de giro y la frenada del personaje entre otros parámetros.

Animation: Componente que contiene todas las animaciones del personaje y las hace accesibles desde los otros componentes. Se encarga de ejecutarlas, cambiar la velocidad de reproducción, pararlas y cambiar de una animación a otra.

15.2. Diagrama de Componentes: Enemigo cuerpo a cuerpo

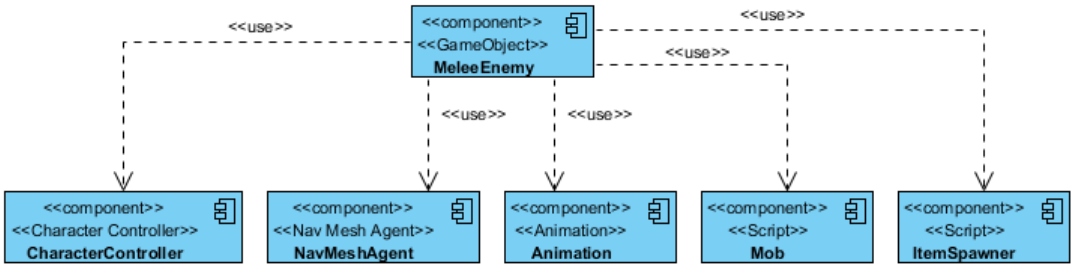


Ilustración 130 Diagrama de Componentes del enemigo Cuerpo a Cuerpo

MeleeEnemy: El objeto que representa a un enemigo básico que combate cuerpo a cuerpo.

CharacterController: Componente propio de Unity que no usa las físicas del motor. Controla la pendiente que el personaje podrá subir o no y su “hitbox” en el mundo 3D.

NavMeshAgent: Componente que, añadido a un personaje que pueda moverse, permite generar automáticamente rutas desde el punto en el que se encuentre hasta la nueva posición y que las recorra.

Animation: Componente que contiene todas las animaciones del personaje y las hace accesibles desde los otros componentes. Se encarga de ejecutarlas, cambiar la velocidad de reproducción, pararlas y cambiar de una animación a otra.

Mob: Script en C# que controla los parámetros del enemigo (experiencia que otorga al morir, vida restante, daño, rango de detección del jugador, rango de ataque y ángulo de ataque). Contiene un sistema de combate básico. Al detectar al jugador, el enemigo se desplazará hacia él hasta que esté lo suficientemente cerca para efectuar un ataque. Esto se consigue a través del rango de ataque.

Una vez se haya efectuado el ataque, el jugador recibirá daño si se encuentra en la zona de daño. El jugador se encuentra en la zona de daño si la diferencia absoluta entre ángulo que forman el vector entre las posiciones de ambos agentes y el vector dirección del enemigo es menor que el parámetro que define el ángulo de ataque del enemigo y el jugador se encuentra en su rango. La justificación del uso de este sistema es que el jugador puede esquivar los ataques enemigos moviéndose.

ItemSpawmer: Un Script en C# que ejecutan los enemigos al morir. Con una colección de objetos y una probabilidad para cada uno de ellos, los instanciará cerca del lugar de la destrucción del enemigo. Es configurable para cada enemigo, pudiendo ajustarse qué objetos serán instanciados y su probabilidad desde los parámetros públicos del script.

15.3. Diagrama de Componentes: Enemigo a distancia

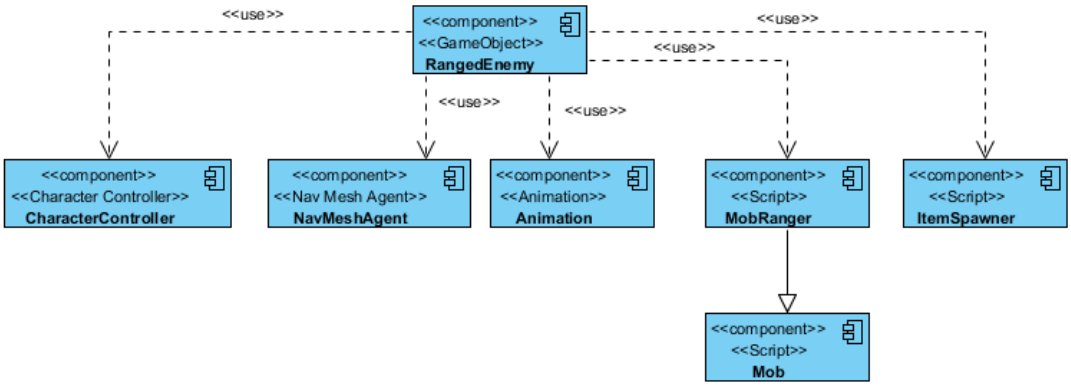


Ilustración 131 Diagrama de Componentes del Enemigo a Distancia

RangedEnemy: El objeto que representa a un enemigo que combate a distancia.

CharacterController: Componente propio de Unity que no usa las físicas del motor. Controla la pendiente que el personaje podrá subir o no y su “hitbox” en el mundo 3D.

NavMeshAgent: Componente que, añadido a un personaje que pueda moverse, permite generar automáticamente rutas desde el punto en el que se encuentre hasta la nueva posición y que las recorra.

Animation: Componente que contiene todas las animaciones del personaje y las hace accesibles desde los otros componentes. Se encarga de ejecutarlas, cambiar la velocidad de reproducción, pararlas y cambiar de una animación a otra.

MobRanger: Script (Clase) que hereda del Script (Clase) Mob visto anteriormente. La funcionalidad añadida es que, al atacar se instanciará un proyectil que se dirigirá a la posición actual del jugador ocasionándole daño si no se esquiva. El proyectil puede variar, desde una bola de fuego a una flecha. El rango de estos enemigos es mucho mayor, por lo que no tienen que acercarse demasiado para infligir daño.

ItemSpawmer: Un Script en C# que ejecutan los enemigos al morir. Con una colección de objetos y una probabilidad para cada uno de ellos, los instanciará cerca del lugar de la destrucción del enemigo. Es configurable para cada enemigo, pudiendo ajustarse qué objetos serán instanciados y su probabilidad desde los parámetros públicos del script.

15.4. Diagrama de Componentes: Jefe

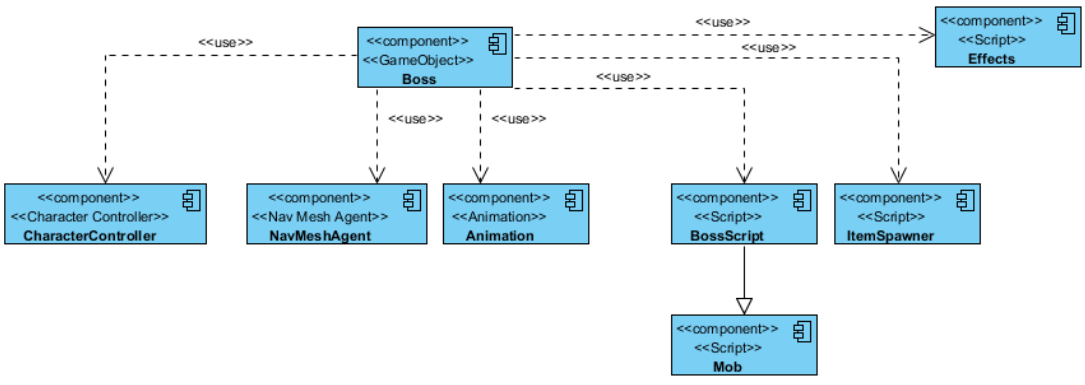


Ilustración 132 Diagrama de Componentes de los Jefes

Boss: Objeto que representa a un enemigo élite.

CharacterController: Componente propio de Unity que no usa las físicas del motor. Controla la pendiente que el personaje podrá subir o no y su “hitbox” en el mundo 3D.

NavMeshAgent: Componente que, añadido a un personaje que pueda moverse, permite generar automáticamente rutas desde el punto en el que se encuentre hasta la nueva posición y que las recorra.

Animation: Componente que contiene todas las animaciones del personaje y las hace accesibles desde los otros componentes. Se encarga de ejecutarlas, cambiar la velocidad de reproducción, pararlas y cambiar de una animación a otra.

BossScript: Este script es una especialización del script Mob. Un ejemplo es el script del Jefe Mago (Abramelin) que añade la funcionalidad de invocar esqueletos cada cierto tiempo a través de un portal oscuro. También puede ser simplemente el script Mob o MobRanger pero con estadísticas más potentes.

ItemSpawmer: Un Script en C# que ejecutan los enemigos al morir. Con una colección de objetos y una probabilidad para cada uno de ellos, los instanciará cerca del lugar de la destrucción del enemigo. Es configurable para cada enemigo, pudiendo ajustarse qué objetos serán instanciados y su probabilidad desde los parámetros públicos del script.

Effects: Este componente (o componentes) representan los pequeños scripts que se añaden a los Jefes para ciertos efectos. Auras que hacen daño, espadas giratorias o regeneración de la propia vida de los Jefes son ejemplos de pequeñas funcionalidades que se incluyen en este apartado.

15.5. Diagrama de Componentes: Canvas

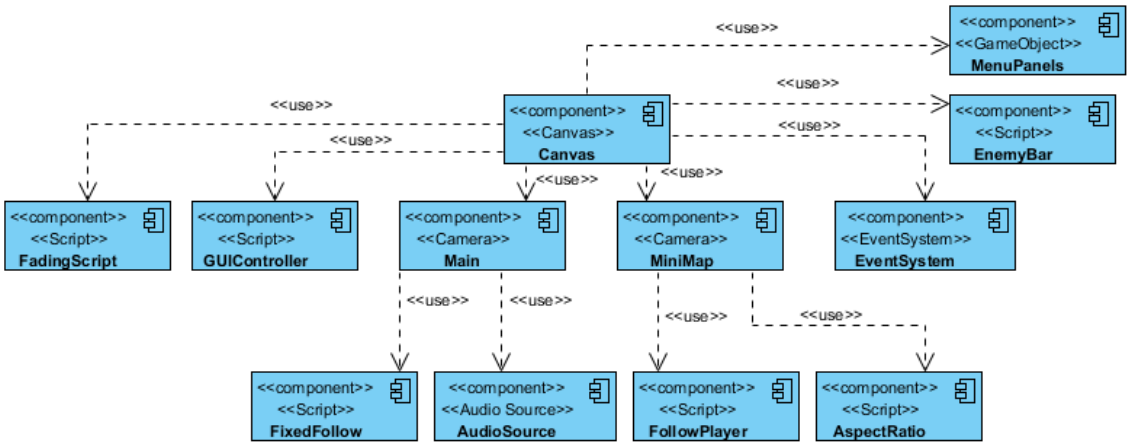


Ilustración 133 Diagrama de Componentes del Canvas

Canvas: Componente propio de Unity que contiene toda la interfaz de usuario (UI). Aparece como un plano 2D superpuesto en la cámara principal. Normalmente sólo hay uno por escena.

FadingScript: Clase en C# que suaviza el cambio de escena con un efecto de oscurecimiento gradual.

GUIController: El controlador de la interfaz. Permite interaccionar con los menús y actualiza la información que se muestra en pantalla.

Main: La cámara principal. De los elementos que contiene señalamos 2:

- FixedFollow: Script que hace que la cámara siga al jugador automáticamente a cierta distancia y con la inclinación adecuada.
- AudioSource: Es un reproductor de sonidos. Al llevarlo en la cámara es perfecto para reproducir sonidos como música de ambiente, en los que no es importante la posición del sonido o la distancia a él (sonido 3D).

MiniMap: Cámara cenital secundaria que, proyectada en una porción de la pantalla nos proporciona un minimapa. Tiene 2 Scripts como hijos:

- FollowPlayer: Más simple que Fixed Follow, ya que no necesita inclinación, pero su cometido es idéntico.

- AspectRatio: Este Script dibuja una textura en el minimapa dándole un aspecto mucho más atractivo.

EventSystem: Componente que gestiona la entrada (ratón y teclado). Es Necesario para usar el Canvas e interactuar con los botones de la UI.

EnemyBar: Script que "renderiza" la barra de vida de los enemigos. Si estos son Jefes usará una imagen más vistosa. Refleja los cambios en la barra de vida haciendo interpolaciones, lo que es un efecto bastante interesante.

MenuPanels: Componente que contiene los diferentes menús a mostrar en el Canvas. Se activan y desactivan por el GUIController, que es quien controla la navegación por ellos.

15.6. Diagrama de Componentes: Mapa/Misión

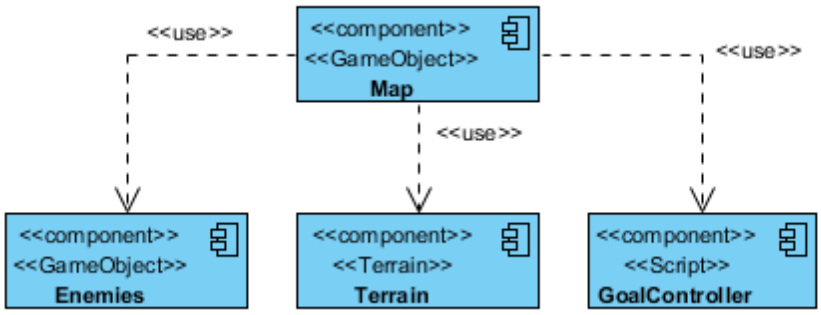


Ilustración 134 Diagrama de Componentes del Mapa/Misión

Map: Objeto que representa un nivel jugable.

Enemies: Objeto que engloba a todos los enemigos existentes en el mapa.

Terrain: El Terreno del nivel. También contiene la iluminación y todos los elementos estáticos del nivel.

GoalController: Script que controla las condiciones de éxito del nivel. Son variadas y propias de cada nivel.

15.7. Diagrama Máquina de Estados: Flujo del Juego

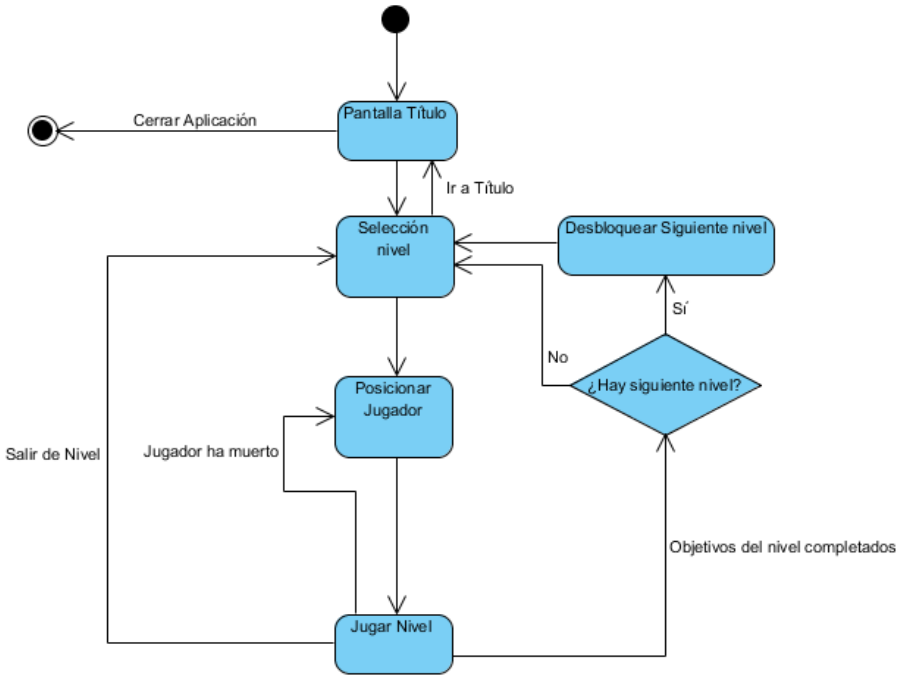


Ilustración 135 Diagrama Máquina de Estados del Flujo del Juego

Por claridad se han obviado las funciones de guardado/carga automáticas. La carga se realizaría al ir a la selección de nivel. El guardado se realizaría al abandonar o completar un mapa.

15.8. Diagrama de Clases: Interfaz y Patrón Arquitectónico MVC

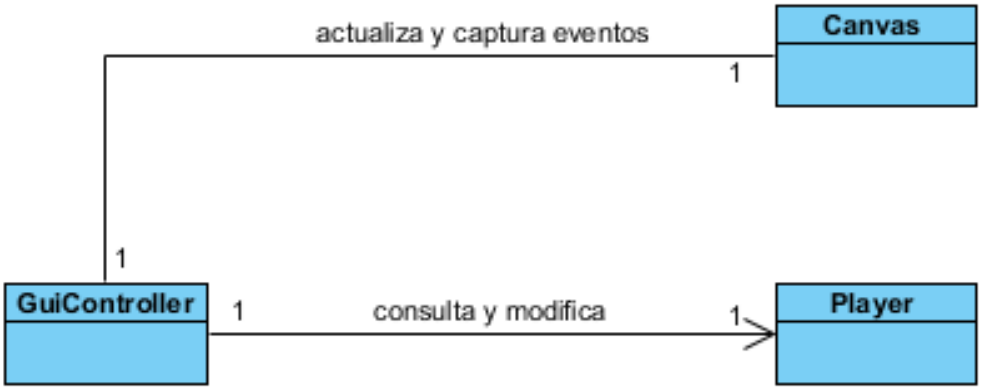


Ilustración 136 Patrón Modelo-Vista-Controlador de la Interfaz de Usuario

15.9. Diagrama Máquina de Estados: Enemigos

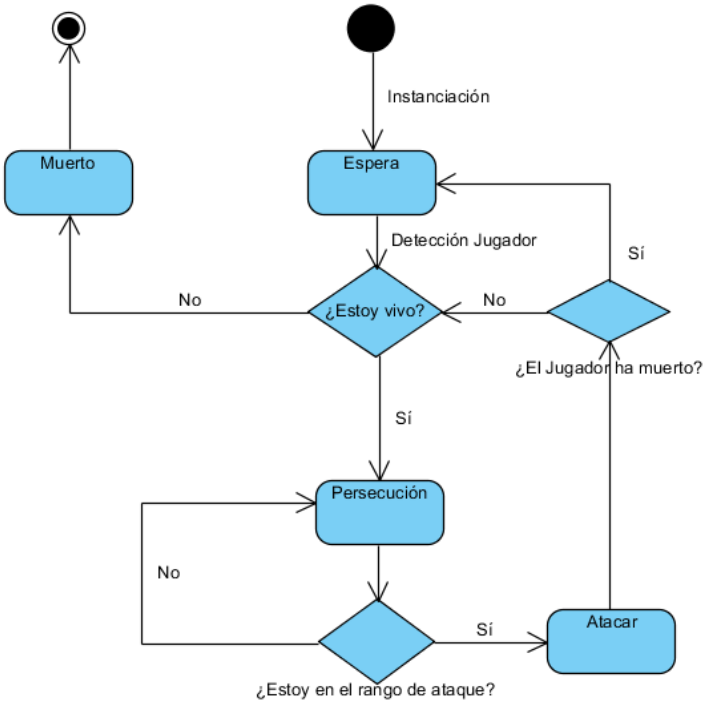


Ilustración 137 Diagrama Máquina de Estados de los Enemigos

15.10. Diagrama de Secuencia: Combate Jugador – Enemigo

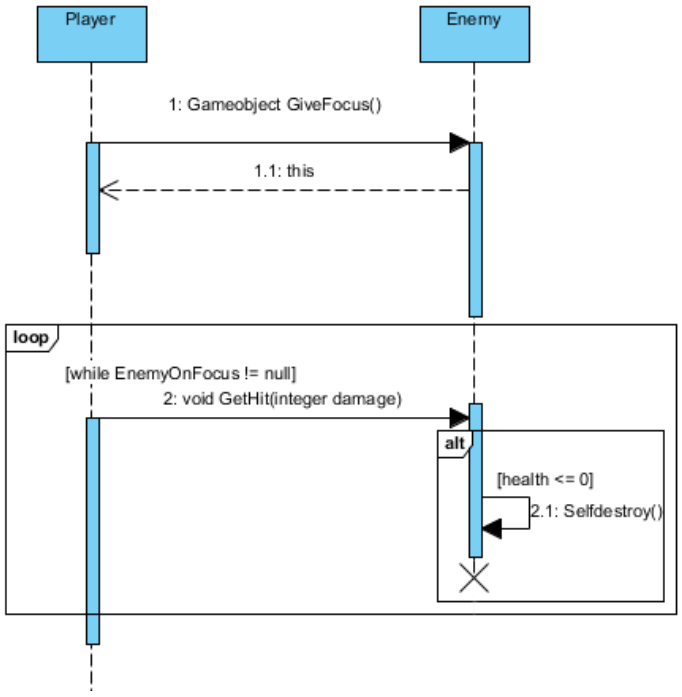


Ilustración 138 Diagrama de Secuencia: Combate Jugador – Enemigo

Este diagrama representa el Combate básico entre el jugador y un enemigo. Para simplificar el proceso sólo se contempla el daño causado por el jugador al enemigo.

El proceso es el siguiente. El jugador selecciona a un enemigo y le va provocando daños por medio de ataques básicos hasta que éste último muere.

16. PATRONES USADOS

Se ha usado el patrón “**Composite**” (componentes) debido a la forma que el motor de videojuegos usados tiene de gestionar los elementos de un proyecto.

El sistema de configuración de teclas del usuario (Keybinds) está basado en el patrón “**Command**”.

El patrón **Modelo-Vista-Controlador** ha sido usado para implementar el control de la interfaz de usuario y el acceso a los datos del modelo. Este patrón no es de diseño (es un patrón arquitectónico), pero es lo bastante relevante como para mencionarlo.

17. CONCLUSIONES: NOTAS DEL AUTOR

Este trabajo ha sido una gran oportunidad para poner en práctica todo el conocimiento adquirido en el grado de Ingeniería Informática referente a la gestión de proyectos y metodologías de desarrollo. Desde el primer momento, haber estado apoyado en técnicas ampliamente probadas y de validez contrastada, han propiciado el éxito en la gestión y desarrollo de este ambicioso proyecto. Se han alcanzado los objetivos dentro del plazo estimado y las pequeñas variaciones de tiempo han sido suplidas gestionando mejor el tiempo.

Según mi opinión, el desarrollo software de este tipo de productos puede ser de los procesos más complejos a los que puede enfrentarse un gestor de proyectos. Esto es debido a que el desarrollo de un videojuego concentra muchas disciplinas diferentes en un mismo proyecto.

Como un proyecto de estas dimensiones excedía los límites temporales del trabajo final de grado, se propusieron soluciones que han ayudado a acelerar algunas de las tareas críticas del proyecto. Usar modelos 3D gratuítos de terceros para la implementación de los agentes ha supuesto un ahorro de tiempo muy importante. Aun habiendo ahorrado tiempo, existen funcionalidades que han quedado fuera del proceso, como los efectos de sonido. Estas caracteristicas están recogidas en el capítulo *18 POR HACER (FUTURO)*.

Una parte importante de los esfuerzos han sido dirigidos a obtener un prototipo que resulte atractivo y amigable al usuario final. Se ha tenido la inestimable ayuda de un grupo de conocidos que han realizado las pruebas de usuario del producto en las distintas fases del proyecto. Su contribución a este trabajo ha sido muy importante. También he dedicado más tiempo a la interfaz y la jugabilidad (sistema de combate), pues creo que son elementos clave en el éxito de la mayoría de los videojuegos.

La participación en este proyecto ha supuesto un enriquecimiento a nivel de conocimientos muy importante. Sin duda ha sido una buena experiencia, que ha puesto a prueba mi capacidad de disciplina y autoplanificación, así como la habilidad

para tomar conciencia de las partes críticas propias del proceso de planificación y desarrollo de un producto software complejo.

En un futuro no descarto la idea de desarrollar algún videojuego para dispositivos móviles aprovechando la facilidad de distribución desde las plataformas Google App Store y Apple App Store. Tengo unas cuantas ideas que podrían ser interesantes y que tratarían de explotar al máximo tecnologías como GPS (sistema de posicionamiento global) y entornos multijugador desde estos dispositivos.

No teniendo experiencia previa en proyectos de esta naturaleza, finalizo este trabajo con una gran satisfacción y orgullo. He disfrutado mucho aprendiendo a utilizar numerosas herramientas nuevas y observando que paso a paso se consiguen los resultados deseados en el tiempo estipulado.

En conclusión, empezar a desarrollar productos de este tipo puede parecer duro debido a la complejidad, pero con el personal y medios adecuados puede ser uno de los procesos de desarrollo más divertidos e interesantes. Son proyectos que aúnan el arte, la tecnología y el entretenimiento en un único producto que puede ser disfrutado por todo el mundo.

18. POR HACER (FUTURO)

Aqui se contemplan las funcionalidades podrían ser implementadas para que el prototipo sea un videojuego apto para su comercialización.

- Sustitución de los modelos 3D usados por modelos propios.
- Creación del icono del juego
- Sistema de escalado de la dificultad en función del nivel del jugador.
- Funcionalidad de configuración de sonido en el menú opciones.
- Añadir a cada mapa un sistema de objetivos a cumplir y objetivos opcionales.
- Añadir a la GUI la lista de objetivos de cada mapa.
- Sistema de recompensas por superar una zona.
- Incluir sonidos para los ataques básicos, habilidades especiales, sonidos de ambiente, sonidos para los jefes.
- Incluir música de fondo original, y cambios en esta cuando aparezca un jefe.
- Incluir narrador para hacer más inmersivos los mapas y que el jugador siga la historia.
- Sistema de guardado y carga del progreso.
- Creación de una escena o un efecto visual para usarla cuando el jugador muere.
- Creación de selección de mapas.
- Creación de sistema de puntuación de mapas.
- Creación de contenido adicional (modo arena, modo oleada o similar).
- Sistema de waypoints para que cuando se muera no haya que empezar desde el punto inicial del mapa.
- Incluir puzzles en los mapas.
- Incluir una animación de presentación para mostrarla antes de la escena de título.
- Crear un sistema optimizado para una instanciación de enemigos más eficiente.
- Intentar optimizar los tiempos de carga en máquinas más modestas.

- Tooltips en zonas importantes de los mapas y descripciones de las habilidades.
- Incluir el producto en el registro de la Propiedad Intelectual.

19. BIBLIOGRAFÍA

La bibliografía en la que el autor se ha apoyado para la realización de este proyecto ha sido:

- Contenido teórico de la asignatura Desarrollo de Videojuegos, J. Roberto Jiménez, 2015.
- Contenido teórico de la asignatura Computer Graphics and Visualization, J. Roberto Jiménez, 2014.
- Contenido teórico de la asignatura Desarrollo Ágil, Pedro González, 2015.
- Contenido teórico de la asignatura Gestión de Proyectos Software, José Ignacio Gómez Espínola, 2015.

Estos contenidos han sido consultados para aplicar algunas de las metodologías que presentaban a este proyecto. No se ha citado contenido como tal, exceptuando el contenido teórico de la asignatura de Gestión de Proyectos Software en los capítulos *11 DEFINICIÓN DEL ÁMBITO DEL PROYECTO* y *12 PLANIFICACIÓN DEL PROYECTO*.

20. FUENTES DE INFORMACIÓN USADAS

Estas fuentes han supuesto una gran cantidad de información relativa al motor de videojuegos Unity 3D. Las fuentes de información consultadas durante la realización del proyecto han sido:

- La documentación oficial de Unity 3D (docs.unity3d.com)
- El foro oficial de Unity 3D (forum.unity3d.com)
- La comunidad de desarrolladores UnitySpain (unityspain.com)
- La comunidad de desarrolladores ArmedUnity (armedunity.com)
- Canales de Youtube de desarrolladores de videojuegos (Tyler Wissler, Naman Jain, SpeedTutor, Brackeys y PSD “Creating Games”).

21. ENLACE DE DESCARGA

https://drive.google.com/folderview?id=0By5565qxDtcORW5LNE9MY19HUzA&usp

Esta carpeta compartida contiene la documentación requerida, una carpeta con el ejecutable del juego para Windows x64 (Ejecutable), otra capeta para el proyecto de Unity (Proyecto Flame Knights Chronicles) y un video demostrativo (Flame Knights Chronicles Video Demo).

22. AGRADECIMIENTOS

Agradezco enormemente la ayuda que he recibido de las personas que han probado el videojuego en las fases de pruebas de usuario. Sin ellos, este trabajo no tendría la calidad que presenta.

Mis agradecimientos a mi familia y amigos más allegados. Ellos me han dado la fuerza para trabajar casi a diario en este proyecto sin desanimarme cuando las cosas no salían como esperaba y han tenido que aguantarme en esos días en los que el estrés puede con uno.

Agradezco la dedicación e implicación de mi tutor en este trabajo. Es un ejemplo a seguir como profesional y, lo que es más importante, como persona.

23. ÍNDICE DE ILUSTRACIONES

Ilustración 1 Motores Populares de Videojuegos 7
Ilustración 2 Logo Unity 3D 8
Ilustración 3 Logo Unreal Engine 10
Ilustración 4 Logo CryEngine 12
Ilustración 5 Logo Path Of Exile 17
Ilustración 6 Combate en Path of Exile 17
Ilustración 7 Cámara en Path of Exile 17
Ilustración 8 Logo Torchlight 2 18
Ilustración 9 Cámara y Ambientación Torchlight 2 18
Ilustración 10 Combate en Torchlight 2 18
Ilustración 11 Logo Diablo 3 19
Ilustración 12 Combate multijugador online en Diablo 3 19
Ilustración 13 Gráficos y Ambientación en Diablo 3 19
Ilustración 14 Icono PEGI Violencia 24
Ilustración 15 Icono PEGI Uso de lenguaje soez 24
Ilustración 16 Icono PEGI +12 24
Ilustración 17 Experiencia necesaria por Nivel 34
Ilustración 18 Una poción de maná (izquierda) y una poción de vida (derecha) 35
Ilustración 19 Pantalla de Título Principal del Prototipo 45
Ilustración 20 Interfaz Gráfica de Usuario (GUI) dentro del juego 46
Ilustración 21 Ejemplos de Minimapa con iconos representando a los diferentes agentes 47
Ilustración 22 De izquierda a derecha, icono del personaje principal, icono de un enemigo común e icono de un jefe 47
Ilustración 23 Menú Pausa 48
Ilustración 24 Menú Abandonar Partida 48
Ilustración 25 Menú Estadísticas 49
Ilustración 26 Menú Opciones 49
Ilustración 27 Habilidad Especial "Bola de Fuego" 49
Ilustración 28 Habilidad Especial "Teletransporte" 50
Ilustración 29 Habilidad Especial "Escudo de Hielo" 50

Ilustración 30 Habilidad Especial "Ragnarök" 51
Ilustración 31 Proyectil del enemigo Skeleton Mage 51
Ilustración 32 Proyectil del enemigo CaveWorm 52
Ilustración 33 Proyectil del enemigo Red Slime 52
Ilustración 34 Capturas de efecto de sangrado del personaje principal 53
Ilustración 35 Efecto de sangrado verde del enemigo Slime 53
Ilustración 36 Skeleton Lancer normal y con efecto Glow 54
Ilustración 37 Texto flotante de ganancia de magia 55
Ilustración 38 Texto flotante de ganancia de vida 55
Ilustración 39 Texto flotante de daño y efecto glow en Skeleton Lancer 55
Ilustración 40 Efectos de subida de nivel del personaje principal 56
Ilustración 41 Jefe Sir Thomas II son Aura 57
Ilustración 42 Jefe Burp con Aura 57
Ilustración 43 Jefe Burp con Aura 58
Ilustración 44 Jefe Nightmare con Aura 58
Ilustración 45 Jefe SkullCrusher con Aura 59
Ilustración 46 Jefe Abramelin con Aura 59
Ilustración 47 Diagrama RBS 68
Ilustración 48 Gantt Iteración 1 82
Ilustración 49 Gantt Iteración 2 83
Ilustración 50 Gantt Iteración 3 84
Ilustración 51 Gantt Iteración 4 84
Ilustración 52 Gantt Iteración 5 85
Ilustración 53 Gantt Iteración 6 86
Ilustración 54 Escenas del Proyecto 87
Ilustración 55 Organización de ficheros del Proyecto 88
Ilustración 56 Sala de Pruebas 89
Ilustración 57 Textura Sala de Pruebas 89
Ilustración 58 Modelo Personaje Principal 90
Ilustración 59 Modelo Final del Personaje Principal 90
Ilustración 60 Nuevo Modelo de Arma 90
Ilustración 61 Rango Ataque Jugador 92
Ilustración 62 Modelo de Primer Enemigo 93
Ilustración 63 Barra de Vida del Enemigo 95

Ilustración 64 Rango de Detección del Enemigo 96
Ilustración 65 Zona de Daño del Enemigo 97
Ilustración 66 Cámara Principal 99
Ilustración 67 Diseño Minimapa 100
Ilustración 68 Minimapa con Marco 100
Ilustración 69 Boceto de la Interfaz Principal 103
Ilustración 70 Fondo de la interfaz Principal 103
Ilustración 71 Fondo y Barras de Vida, Maná y Experiencia 103
Ilustración 72 Retrato del Héroe 104
Ilustración 73 Interfaz Principal de Usuario dentro del juego 104
Ilustración 74 Boceto Interfaz de Usuario con Nivel 105
Ilustración 75 Zona de Nivel y Marcos de la Interfaz 105
Ilustración 76 Interfaz de Usuario con Nivel y Marcos 105
Ilustración 77 Menú Pausa en el juego 106
Ilustración 78 Fondo del Menú Pausa 106
Ilustración 79 Menú Abandonar Partida 107
Ilustración 80 Boceto Menú Opciones 108
Ilustración 81 Menú Opciones 109
Ilustración 82 Boceto Menú Características 110
Ilustración 83 Menú Características 110
Ilustración 84 Interfaz de Usuario con Acceso a los Menús 111
Ilustración 85 Habilidad “Bola de Fuego” 115
Ilustración 86 Habilidad "Teletransporte" 116
Ilustración 87 Habilidad "Escudo de Hielo" 116
Ilustración 88 Habilidad "Ragnarök" 117
Ilustración 89 Aspecto de la Interfaz de Usuario Final 119
Ilustración 90 Tiempos de Enfriamiento en la Interfaz Principal 120
Ilustración 91 Cuestionario de Pruebas de Usuario - 3ª Iteración, Tarea 8.4.2 121
Ilustración 92 Proyectil de Gulp, Slime Queen 133
Ilustración 93 Espada Espectral de Sir Thomas II 134
Ilustración 94 Proyectil de Abramelin 135
Ilustración 95 Modelos de Pociones de Maná (izquierda) y Vida (derecha) 138
Ilustración 96 Sistema de Keybinds configurado para usar Z, X, C y V 145
Ilustración 97 Interfaz de Usuario mostrando unas nuevas asignaciones de teclas 145

Ilustración 98 Acceso al menú Stats resaltado, indicando que se tienen puntos de característica sin asignar 146
Ilustración 99 Texto flotante de daño en Diablo 3 147
Ilustración 100 Texto flotante de ganancia de magia 148
Ilustración 101 Texto flotante de ganancia de vida 148
Ilustración 102 Texto flotante de daño y efecto glow en Skeleton Lancer 148
Ilustración 103 Skeleton Lancer normal y con efecto Glow 149
Ilustración 104 Barra de vida de Jefes 149
Ilustración 105 Capturas de efecto de sangrado del personaje principal 150
Ilustración 106 Efecto de sangrado verde del enemigo Slime 150
Ilustración 107 Diferentes iconos del Minimapa 151
Ilustración 108 Ejemplos de Minimapa con iconos representando a los agentes 152
Ilustración 109 Texto de transición entre escenas 153
Ilustración 110 Efecto de subida de nivel 154
Ilustración 111 Efecto Subida de nivel detalle 155
Ilustración 112 Esbozo de la posición de los diferentes enemigos de la escena Demo 158
Ilustración 113 Esbozo del mapa Demo y la ruta de juego 158
Ilustración 114 Escena de Título 159
Ilustración 115 Terreno texturizado de la escena Demo 160
Ilustración 116 Detalle cementerio escena Demo con árboles y assets 161
Ilustración 117 Escena Demo con los árboles y assets 161
Ilustración 118 Captura 1 de la escena Demo 162
Ilustración 119 Captura 3 de la Escena Demo 163
Ilustración 120 Captura 2 de la Escena Demo 163
Ilustración 121 Captura 5 de la Escena Demo 164
Ilustración 122 Captura 4 de la Escena Demo 164
Ilustración 123 Captura 6 de la Escena Demo 165
Ilustración 124 Comparación Tiempo Estimado y Real por Iteraciones 168
Ilustración 125 Comparación Tiempo Estimado y Real por Grupos de Tareas 169
Ilustración 126 Desviación por Iteraciones (El tiempo negativo significa un adelanto conforme lo estimado) 170
Ilustración 127 Desviación por Grupos de Tareas (El tiempo negativo significa un adelanto conforme lo estimado) 170

Ilustración 128 Horas invertidas por tipo de Tarea 171
Ilustración 129 Diagrama de Componentes del Jugador 177
Ilustración 130 Diagrama de Componentes del enemigo Cuerpo a Cuerpo 178
Ilustración 131 Diagrama de Componentes del Enemigo a Distancia 179
Ilustración 132 Diagrama de Componentes de los Jefes 180
Ilustración 133 Diagrama de Componentes del Canvas 182
Ilustración 134 Diagrama de Componentes del Mapa/Misión 183
Ilustración 135 Diagrama Máquina de Estados del Flujo del Juego 184
Ilustración 136 Patrón Modelo-Vista-Controlador de la Interfaz de Usuario 184
Ilustración 137 Diagrama Máquina de Estados de los Enemigos 185
Ilustración 138 Diagrama de Secuencia: Combate Jugador – Enemigo 185

24. ÍNDICE DE TABLAS

Tabla 1 Comparativa Motores de Videojuegos 14
Tabla 2 Características de las Habilidades Especiales del Personaje Principal 32
Tabla 3 Efectos de los Puntos de Característica 33
Tabla 4 Enemigos Comunes Cuerpo a Cuerpo 40
Tabla 5 Enemigos Comunes a Distancia 41
Tabla 6 Enemigos Únicos (Jefes) 43
Tabla 7 Requisitos Principales 67
Tabla 8 Documento POS 70
Tabla 9 WBS 76
Tabla 10 Nuevo Grupo de Tareas 76
Tabla 11 Estimación Resumida por Grupo de Tareas 76
Tabla 12 Iteraciones 77
Tabla 13 Precio Infraestructuras 79
Tabla 14 Gastos Fijos de Servicios 79
Tabla 15 Gastos Licencias Software 80
Tabla 16 Resumen Costes y Precio Final 80
Tabla 17 Costes Reales del Proyecto 81
Tabla 18 Cuestionario de Pruebas de Usuario - 1ª Iteración, Tarea 3.5.2 98
Tabla 19 Cuestionario de Pruebas de Usuario - 2ª Iteración, Tarea 7.7.2 112
Tabla 20 Condiciones Anómalas detectadas - 2ª Iteración - CCG 112
Tabla 21 Condiciones Anómalas detectadas - 2ª Iteración - JJRG 113
Tabla 22 Gráficos Habilidades Especiales del Personaje Principal 115
Tabla 23 Diseño de Enemigos Comunes Cuerpo a Cuerpo 125
Tabla 24 Diseño de Enemigos Comunes a Distancia 127
Tabla 25 Proyectiles de los Enemigos a Distancia 128
Tabla 26 Diseño de Jefes 132
Tabla 27 Combates necesarios para completar exitosamente la fase de Pruebas 136
Tabla 28 Cuestionario Pruebas de Usuario Iteración 5, Tarea 9.5.2 143
Tabla 29 Cuestionario de Pruebas de Usuario - 6ª Iteración, Tarea 11.6.2 157
Tabla 30 Cuestionario de Pruebas de Usuario - 6ª Iteración, Tarea 15.3 166
Tabla 31 Condiciones Anómalas detectadas - 6ª Iteración, Tarea 15.3 166

Tabla 32 Tabla de Duración Estimada y Real de Iteraciones 167
Tabla 33 Tabla de Duración Estimada y Real de los Grupos de Tareas 168

25. ANEXO I: GLOSARIO DE TÉRMINOS

Los términos propios de este sector de desarrollo son los que aparecen a lo largo del proyecto. También se han incluído palabra que puedan no ser conocidas por el lector a nivel general. Por orden de aparición son los siguientes:

- **Plugin:** Un complemento es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy específica. Esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de la API.
- **Framework:** Estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software.
- **Indie Games:** Videojuegos creados sin el apoyo financiero de una distribuidora de videojuegos.
- **Sprite:** Se trata de un tipo de mapa de bits dibujados en la pantalla de ordenador por hardware gráfico especializado sin cálculos adicionales de la CPU. A menudo son pequeños y parcialmente transparentes.
- **Asset:** Un bien o activo de un proyecto.
- **IDE:** Un entorno de desarrollo integrado o entorno de desarrollo interactivo, en inglés Integrated Development Environment (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.
- **FPS:** Genero de videojuegos de disparos en primera persona.
- **Blueprint:** Sistema de programación de Unreal Engine.
- **Raycasting:** uso de la intersección rayo-superficie para solucionar una variedad de problemas en gráficos por ordenador y geometría computacional.
- **Royalties:** El pago que se efectúa al titular de derechos de autor, patentes, marcas o know-how a cambio del derecho a usarlos o explotarlos.
- **RPG:** Un género de videojuegos que usa elementos de los juegos de rol tradicionales.

- **Hack and Slash:** es un género de videojuegos basados en los combates. El estilo es exactamente el mismo que los beat 'em up con la única diferencia de que el personaje suele llevar algún tipo de arma cuerpo a cuerpo, normalmente de filo o contundente.
- **Beat 'em up:** Es un género de videojuegos en el que se destaca el combate cuerpo a cuerpo entre el protagonista y un gran número de antagonistas.
- **Action-RPG (A-RPG):** Un género de videojuegos que comparten muchas características con los RPG pero que, a diferencia de estos, ofrecen combates en tiempo real.
- **Perspectiva Top-Down:** Se conoce como vista ojo de ave, vista de pajaro, vista elevada o vista de helicóptero a un ángulo de cámara utilizado en los videojuegos que muestra al jugador y al área circundante desde arriba.
- **Micropagos Éticos:** (Micropagos justos) Son micropagos que ayudan a mantener un juego pero que el usuario no está obligado ha realizarlos para disfrutar del juego al completo.
- **Navigational Mesh:** (Malla de navegación). Es una estructura de datos abstracta que se utiliza en aplicaciones de inteligencia artificial para ayudar a los agentes en la búsqueda de caminos a través de espacios complicados .
- **Navigational Mesh Agent:** Agentes que se desplazan por una malla de navegación.
- **Versión Beta:** Un periodo donde el software está técnicamente acabado, lo cual significa que no se le añadirán de momento más funciones, y presumiblemente será lo suficientemente estable para trabajar con normalidad.
- **PEGI:** El sistema de clasificación por edades establecido por Información Paneuropea sobre Juegos se estableció con el objeto de ayudar a los progenitores europeos a tomar decisiones informadas a la hora de adquirir juegos de ordenador.

- **NPC:** (PNJ, personaje no jugador). Es generalmente parte del programa, y no controlado por un humano. El concepto opuesto es el personaje representado por una persona o PJ (personaje jugador).
- **GUI:** (Interfaz Gráfica de Usuario). Es un programa informático que actúa de interfaz de usuario, utilizando un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz.
- **Stats:** El conjunto de atributos de un agente (Sea jugador o NPC).
- **Glow:** Efecto de brillo en un material.
- **Sistema de Keybinds:** Sistema de configuración de teclas de usuario.
- **Demo:** Prototipo o versión incompleta o de evaluación de un determinado software con fines promocionales o para demostrar sus funcionalidades.
- **Escena:** Contienen los objetos del juego. Pueden ser usadas para crear un menú principal, niveles individuales, y cualquier otra cosa.
- **Bake:** Proceso por el cual se crea una malla de navegación transitable por los agentes.
- **Static:** Adjetivo que define un objeto como fijo en una escena.
- **Prefab:** Actúa como una plantilla a partir de la cual se pueden crear nuevas instancias de objetos en la escena . Las ediciones realizadas a un Prefab se reflejan inmediatamente en todos los objetos producidos con ella, pero también se pueden reemplazar componentes y configuraciones para cada caso individual.
- **Tester:** Usuarios que usan sus conocimientos informáticos y su tiempo para detectar errores en diferentes versiones del software y así poder informar de éstos para que los desarrolladores los corrijan.
- **Viewport:** Es un rectángulo 2D que define el tamaño de la superficie de representación en la que se proyecta una escena 3D.
- **Canvas:** En Unity, es el área donde todos los elementos UI deben estar. El Canvas es un Game Object con un componente Canvas en él, y todos los elementos UI deben ser hijos de dicho Canvas.

- **Instanciación:** Acción y efecto de crear una instancia. Crear en memoria un ejemplar de un conjunto de datos y código definido por una clase o estructura.
- **Rigging:** (“Rigear”). La acción o el efecto de hacer rigs. Los rigs son sistemas de cadenas de huesos y objetos de control, con o sin características interactivas, que sirven para definir deformaciones sobre un objeto geométrico.
- **Texturizar:** Aplicar texturas al material de un modelo geométrico.
- **Decimate:** Herramienta existente en Blender para simplificar modelos geométricos.
- **FeedBack:** Retroalimentación, generalmente de información.
- **Culling Mask:** Opción usada para “renderizar” selectivamente objetos de la escena.
- **Render: (Renderizar)** Proceso de generar una imagen o vídeo mediante el cálculo de iluminación partiendo de un modelo en 3D.
- **Fading, Fade:** (Desvanecimiento). Técnica visual que provoca un efecto de desvanecimiento de una imagen dejando un color de fondo o viceversa.
- **Ambient Occlusion:** Es una técnica de sombreado y representación que se utiliza para calcular el grado de exposición de cada punto en una escena a la iluminación ambiental .