UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf

Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica

Universitat Politècnica de València

ComeBack videojuego RPG en Unity 3D. Creación y Mantenimiento de los Sistemas de Inventario y de Diálogos

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Nathan Cole Tyler Llavador

Tutor: Ramón Pascual Mollá Vayá

2019/2020

Resumen

En este TFG plantea el desarrollo de un videojuego RPG (Rol Player Game) como proyecto de emprendimiento cuyo resultado final es un producto comercial mínimamente viable.

Un RPG requiere del uso constante de objetos que son encontrados en el transcurso del juego por el personaje o le son dados tras acabar una misión. Es por ello que este estilo de juego necesita de un módulo que realice las tareas de almacenamiento y gestione la interactuación del usuario con dichos objetos.

Este TFG creará el citado módulo adaptado a los videojuegos RPG como un producto final comercial que será expuesto en la tienda oficial de Unity. Paralelamente, con el fin de probar su viabilidad y generar un caso de uso, se insertará dicho plug-in en un juego RPG elaborado por un equipo de desarrollo con el que se ha desarrollado el videojuego ComeBack.

En el videojuego ComeBack, además de algunos desarrollos menores, me he encargado de diseñar, implementar y validar el sistema de diálogo que es otro de los elementos importantes en un RPG ya que permite interactuar con otros personajes del juego para que nos ayuden durante misiones, poder sumergir al usuario en la historia y en su cuerpo de tradiciones.

Palabras clave: Unity, RPG, dialogo, inventario, plug-in, videojuego

Summary

In this TFG we present the development of an RPG (Rol Player Game) as an entrepreneur project which the result of it is a minimum viable commercial product.

An RPG requires the constant use of objects which are encountered throughout the course of the game by the player or are given to him after finishing a mission. This is why this game style needs a module which performs the tasks of storing and manage the interaction of the user with said objects.

This TFG will create the beforementioned module adapted to RPG videogames as final commercial product which will be exposed in the official Unity store. Concurrently, with the purpose of proving its viability and generate a use case, the beforementioned plugin will be inserted in an RPG elaborated by a development team with whom we have developed the videogame ComeBack.

In addition to some minor developments, I have been in charge of designing, implementing and validation the dialogue system in the videogame ComeBack.

The dialogue system is another of the important elements in an RPG because it permits the interaction with other characters. The dialogue help us during missions, to immerse the user in the story and the lore of the videogame.

Keywords: Unity, RPG, dialogue, inventory, plug-in, video game

Tabla de contenidos

CAPÍTULO 1 INTRODUCCIÓN

Los Role-Playing Games (RPG) son, actualmente, unos de los videojuegos más populares del mercado gracias a contener mundos abiertos y una historia trabajada. Hoy en día, el género se ha extendido tanto que contiene una larga lista de subgéneros de los cuales se destacan los RPGs de acción, los RPGs multijugador online masivos (MMORPG), los RPGs de exploración de mazmorras (roguelike) o los RPGs tácticos.

Éste TFG se centra en el desarrollo de un videojuego RPG llamado ComeBack. ComeBack es un videojuego que contiene un mapa abierto en el que el jugador se despierta en un mundo distópico producido por la unión de dos realidades alternativas, en el cual el personaje principal deberá centrarse en como escapar.

Este proyecto ha sido realizado en el marco de un proyecto de emprendimiento, es decir, se abordarán tanto la creación de la idea de negocio, su evaluación como producto, el desarrollo del proyecto y experimentos con early adopters, y la publicación de una primera versión. Este proyecto de emprendimiento ha sido realizado en colaboración en Start.Inf, el espacio emprendedor de la ETSINF. En este proyecto han participado tres alumnos; Altea Taulet, David Guillen y Nathan Tyler. A pesar de haber realizado el proyecto de emprendimiento de forma colaborativa, cada alumno ha realizado TFGs centrándose en diferentes aspectos técnicos del desarrollo del producto.

En cuanto a este TFG se refiere, se centra en la creación de un *asset* de sistemas de inventario y de diálogos y su utilización y validación en ComeBack.

Se han utilizado para este proyecto Visual Studio como entorno de desarrollo, Unity[1] como motor de videojuegos, en concreto Unity 3D, además de otras herramientas para facilitar el trabajo en equipo. Se han utilizado Github y Bitbucket como repositorio de versiones para facilitar el desarrollo en equipo y TUNE-UP Process para administrar el proyecto.

La publicación de este proyecto se realizará a través de la plataforma de financiación colectiva de Patreon[2], pero una vez finalizado el producto completo se tratará la publicación Steam[3], que es la plataforma de venta de videojuegos líder actual en el mercado.

A su vez, para una mejor comprensión de términos técnicos o relacionados en exceso con el mundo de los videojuegos, se ha incluido en la parte final de la memoria un glosario explicativo de los términos mencionados anteriormente.

MOTIVACIÓN

General

En las ultimas décadas, se ha observado un rebrote de popularidad de los RPGs en el mundo de los videojuegos, que, tras sufrir una estancación en la década de los 90, se han convertido en el tercer genero de videojuegos más popular del momento detrás de los géneros de acción y *shooter*.

En cambio, la mayoría de los nuevos videojuegos de este genero han sido basados en características occidentales, poniendo a parte los modelos orientales, que a pesar de que siguen estando presentes en productos exitosos, no han sido aprovechados hasta su máximo potencial por empresas lideres en el desarrollo de los videojuegos.

A su vez, es rara la ocasión que se puede encontrar un híbrido con mezclas tanto orientales como occidentales.

Debido a esto se quiso iniciar este proyecto singular, en el que se pueda explotar el actual auge de los RPGs presentando una opción distinta a las del resto del mercado de videojuegos.

1 https://unity.com/es
2 https://www.patreon.com/
3 https://store.steampowered.com/?l=spanish

Personal

Uno de mis mejores recuerdos de la infancia son tardes con o sin amigos jugando a videojuegos, ya sea Pokémon o GTA San Andreas. Desde entonces mi sueño ha sido poder crear o ayudar en la creación de un videojuego, por lo que haber podido estar comprometido en este proyecto ha sido dar un pequeño paso a mi sueño desde joven.

OBJETIVOS

La meta del equipo era la formación en programación de videojuegos por medio del desarrollo del producto que se va a presentar en la memoria, con la visión de poder rentabilizar el proyecto mediante una plataforma de micro financiación.

De esta manera, nos serviría no solo como manera para recaudar beneficios, sino también para poder interactuar con usuarios directamente, sirviendo de puente entre el desarrollador y el usuario, pudiendo así compartir información y opiniones.

En consecuencia, se establecieron los objetivos comunes mostrados a continuación:

- Desarrollar una demo jugable.
  - Diseño de 3 niveles completamente acabados en su aspecto visual
  - Desarrollo del sistema de combate.
  - Programación del menú principal.
  - Modelado y animación el personaje principal.
  - Creación de un Sistema de Inventario Simple
  - Programación un Sistema de Diálogos que permita sumergir al jugador en la historia y el *lore* del videojuego

Constituyendo de esta manera el cimiento del proyecto con el que se pudiera continuar trabajando en el futuro, añadiendo niveles, trama y personajes.

- Publicación del videojuego en Patreon, con la finalidad de lograr conseguir un método de financiación lo más pronto posible.

De esta manera se pueden obtener beneficios a lo largo del proceso de desarrollo, una vez acabado el producto completo se procederá a intentar poner a la venta en Steam.

A su vez, para poder recaudar un mayor numero de fondos, se establecerá la creación de un Asset para la Unity Store conteniendo Sistemas de Inventario y de Diálogos, con los objetivos siguientes:

- Desarrollar un Asset reutilizable que contenga:
  - Modulo de Sistema del Inventario personalizable
  - Modulo de Sistema de diálogos personalizable y con interfaz reutilizable a lo largo del juego

Para la validación de este, se utilizará dicho Asset en ComeBack, personalizándolo a la temática del videojuego.

ESTRUCTURA DE LA MEMORIA

Esta memoria consta de siete capítulos diferentes, en el primero contiene la motivación y los objetivos tanto del equipo como personales en el momento de realizar el proceso de desarrollo.

Tras ello, se explica la creación de la idea de negocio desde su estado inicial hasta haber alcanzado el estado actual del producto.

En el tercer capítulo se explica el estudio de la historia de los videojuegos RPG. A su vez, se analizan videojuegos similares a ComeBack con la finalidad de poder realizar una comparación, ver en que destaca de nuestra idea y cuales son sus limitaciones. Todo el análisis se agrupa finalmente en una tabla comparativa con el fin de mostrar con mas detalle dicha comparación.

En el ultimo punto del tercer capítulo se explica la proyección económica del proyecto, calculando los posibles gastos e ingresos, terminando con la generación de un Lean Canvas y un análisis DAFO en los que se expone el análisis del proyecto.

En el cuarto capítulo consta la explicación del desarrollo de la idea de negocio, con el establecimiento de las características y los pasos a seguir para la producción de los MVPs. A su vez, comprende tanto el proceso de marketing como la cronología del proyecto.

En el quinto capítulo, se inicia la explicación técnica, explicando cuales han sido las herramientas utilizadas y elaborando porque se ha decidido hacer uso de estas.

En el sexto capítulo se lleva a cabo una explicación de la fase de desarrollo del Asset Simple Iventory and Dialogue System y su validación en el videojuego ComeBack. Comentando a su vez las experiencias adquiridas y las bases que han hecho posibles la realización de este producto.

El séptimo capitulo consta del capitulo donde se exponen las conclusiones de proyecto, explicando tanto los objetivos cumplidos como las asignaturas de la carrera que nos han ayudado a realizar este proyecto.

En el capítulo ocho, el ultimo de esta memoria, se explica cual será el trabajo futuro tras la conclusión de la memoria.

COLABORACIONES

Tanto el proyecto como la memoria han sido realizados en conjunto con otros dos alumnos de la Escuela Técnica Superior de Ingeniería Informática, los cuales ya han presentado su experiencia en este proceso como trabajo de fin de grado.

La encargada del diseño, modelado y animaciones de los personajes ha sido Altea Taulet Rius, que explica el todo el proceso en su TFG “ComeBack, un videojuego RPG desarrollado en Unity. Diseño y animación”

Por otro lado, el responsable del balance y control de eventos es David Guillén Mateu, cuyo trabajo toma el nombre de “ComeBack, un videojuego RPG desarrollado en Unity. Balance y Control de eventos”.

CONVENCIONES

A lo largo del documento se han utilizado fuente en cursiva con el fin de marcar extranjerismos utilizados en el mundo de los videojuegos o palabras extranjeras.

Para el nombre de otros productos se han utilizado dobles comillas siempre que estas no fueran repetidas con exceso. Tanto el nombre del producto como el de otras empresas de desarrollo de videojuegos se han dejado en formato estándar para no sobrecargar la memoria.

CAPÍTULO 2 GENERACIÓN DE LA IDEA DE NEGOCIO

Desde el principio del proyecto, el equipo ya tenía pensado que deseaba establecer relevancia a la historia del videojuego. Se eligió la idea de cimentar la trama en un mundo distópico del que se debe escapar para completar el videojuego.

Tras escoger la historia se realizó una sesión de *brainstorming* para poder reflexionar a cerca de las diferentes posibilidades a la hora de constituir una etapa histórica o temática definida. Tras haber definido una lista de trece alternativas diferentes, se desecharon todas una a una hasta acortarla a solo dos: una opción futurista y otra medieval.

Al llegar a este punto el equipo se topó ante un impase, ya que resultaba imposible elegir entre una de estas dos ideas, por lo tanto, se determinó por fusionar las dos ideas. Esto llevó a crear un mundo paralelo en el que el jugador debe escapar para poder terminar el videojuego.

A partir de este punto, se continuó desarrollando y perfilando la idea tras múltiples sesiones de *brainstorming*.

Por último, se determinó utilizar un mapa mental con el fin de especificar y aclarar informalmente la historia. Durante la elaboración de este se desarrolló una última idea que dio pie a lo que sería el verdadero trasfondo y la base del juego.

En este momento no se había determinado el porqué el personaje se encuentra atrapado en un mundo paralelo. Fue durante la deliberación del mapa mental donde se sugirió que el protagonista en realidad se encontrara inconsciente en el mundo real debido a la ingesta de drogas, y a partir de aquí se extendió a que todo lo que ocurriría en el juego sería una metáfora que hiciera referencia a la lucha interna que hace por sobrevivir en el mundo real.

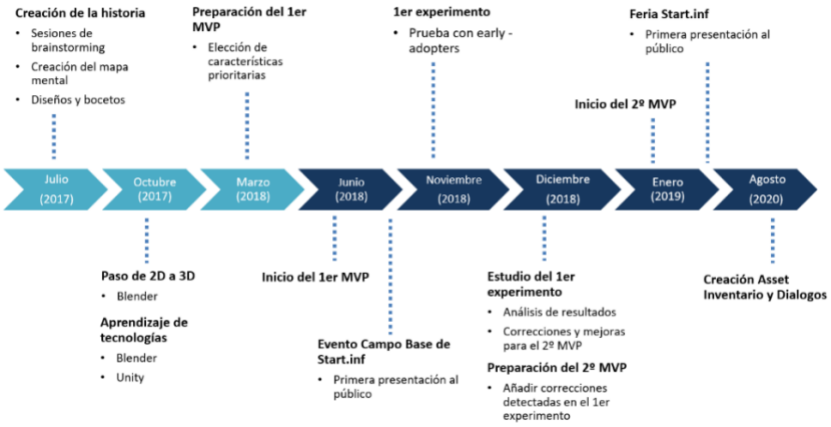


Figura 1: Esquema de las diferentes fases en las que se realizó el diseño e implementación del proyecto

CONCEPTO DEL JUEGO

ComeBack es el nombre que recibe el producto desarrollado durante este proyecto de emprendimiento, ya que el objetivo de este videojuego es el de regresar al presente.

El proyecto consta de un videojuego de género RPG, cuyo objetivo principal es el de transmitir una historia a medida que el jugador avanza en el juego.

La trama principal parece trata de las adversidades de una mujer atrapada sin memoria alguna en un mundo paralelo. Sin embargo, el jugador desvelará una vez avanzado el juego, que el verdadero objetivo es tratar de sobrevivir a un suceso ocurrido en el mundo real.

En consecuencia, se realizó un juego de palabras con el nombre del videojuego, dando una pista del verdadero objetivo del videojuego, ya que come back, en inglés, significa volver. Por otra parte, si se escribe junto, ComeBack, el concepto cambia sutilmente a volver con más fuerza. Con el fin de combinar ambos significados, se estableció ComeBack como nombre del juego al fusionar las dos concepciones en una sola.

El videojuego ha sido realizado para un único jugador. Se quiso evitar un enfoque multijugador por tres motivos:

- Si se pretende sumergir al jugador en la historia, resulta difícil utilizando el modo multijugador.
- Normalmente, cuando se da alta relevancia a la historia de un RPG se tiende a evitar el modo multijugador.
- Desconocimiento inicial de las tecnologías con las que se ha desarrollado el proyecto

Idea Inicial

La idea original fue desarrollada por dos de los integrantes del equipo un año antes del comienzo del proyecto.

La historia se definió y perfiló unos meses antes de empezar el desarrollo videojuego. Añadiendo en esta etapa el trasfondo temporal de la unión de pasado y futuro.

Una vez elegido RPG como modelo de videojuego, se estableció un sistema básico de clases basado en tres tipos de perfiles básicos: *sanador*, *dps* y *tanque*.

Evolución Hasta La Idea Final

Intentando equilibrar los combates y los poderes de cada espacio temporal se decidió modificar el concepto y se pasó de una fusión entre futuro y pasado a una realidad futurista mezclada con fantasía medieval.

Se decidió modificar la rama del pasado para que no esté en desventaja con los avances tecnológicos, añadiendo poderes mágicos a sus personajes.

Tras haber aclarado la temática, se decidió cambiar los poderes de los personajes con dos objetivos principales en mente: introducir una leve novedad en el sistema tradicional y hacer uso de la originalidad que aporta la trama de la historia.

Se pasó de tres a seis *clases*, haciendo que todas estén disponibles para el protagonista del juego. Con ello se intenta no limitar al jugador a una clase que no desea por elegir una estética temporal específica ni viceversa.

Las clases y los poderes se clasifican de la siguiente manera:

- Pasado:
  - Sacerdote (sanador): poderes curativos, aumentos de defensa y ataque y hace uso de hechizos sagrados para combatir.
  - Mago (dps): conjura magia de diferentes elementos y formas para derrotar al oponente.
  - Caballero (tanque): protege al equipo y aumenta la fortaleza de sus compañeros.
- Futuro:
  - Medico (sanador): hace uso de la medicina para remendar a los heridos, desarrolla viales que potencian las capacidades de combate y utiliza su instrumental para defenderse.
  - Ingeniero (dps): construye armas y artilugios que le ayuden a eliminar a todo aquel que se le enfrente.
  - Juez (tanque): imparte justicia defendiendo al inocente y castigando a todo aquel que ose plantarle cara.

Finalmente, se introdujo un *árbol de habilidades*, en el que se pueden elegir habilidades de las seis clases de forma indistinta. Esta decisión se toma con el mismo objetivo que la anterior: no limitar la jugabilidad en pos de la estética temporal que desee adoptar el jugador. Además, se le da al usuario una sensación más amplia de personalización y de elección del trasfondo.

De esta forma uno puede centrarse en una rama y alcanzar las habilidades más poderosas o combinar varias o todas las ramas y crear combinaciones que le den ventajas a la hora de combatir. En la Figura 2 se muestra a forma de ejemplo la parte del árbol correspondiente a las habilidades de la clase sacerdote.

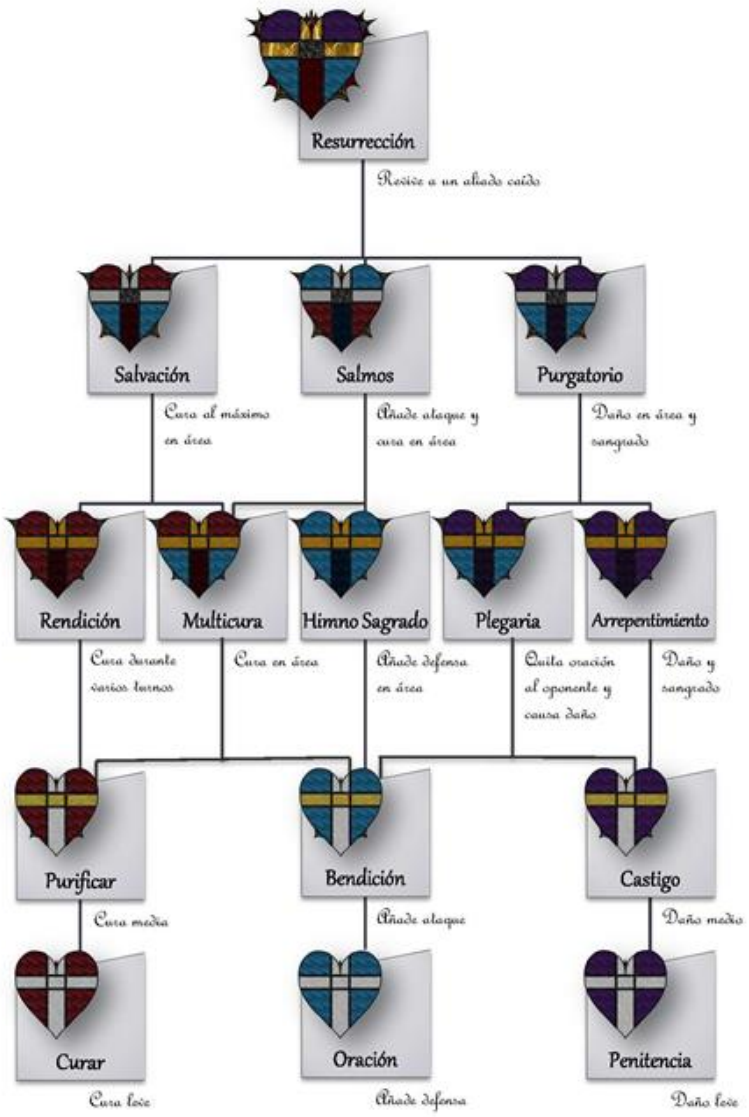


Figura 2: Rama de habilidades del sacerdote

CAPÍTULO 3 EVALUACIÓN DE LA IDEA DE NEGOCIO

ESTADO DEL ARTE

Entre los diferentes géneros de videojuegos, que hoy en día forman una lista considerablemente extensa[4] RPGs, o Role-Playing Games, que, aun estando presentes desde las primeras computadoras personales y consolas, han conseguido prevalecer hasta el presente manteniendo su presencia en el mercado actual.

Basados en un inicio en los juegos de rol de mesa, destacando como el más popular de ellos "Dungeons & Dragons", los RPGs han ido evolucionando hasta convertirse en un género que abarca un gran número de diversos tipos de videojuegos, los cuales ya no tienen por qué mantener la esencia básica de juego de rol que caracterizaban a sus ancestros.

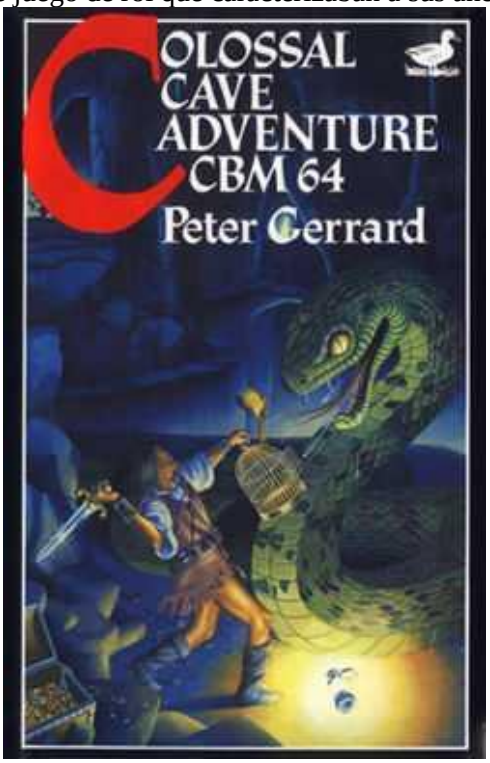


Figura 3: Colossal Cave Adventure, primer videojuego de aventuras

Con el traspaso de los juegos de rol al espacio virtual, se mantuvo el nombre de RPG debido a que la esencia era la misma, el jugador contaba de un personaje con estadísticas que aumentaban con el avance y desarrollo de este, contando además con el hecho de que los acontecimientos y el final de la historia cambiaban según las decisiones tomadas en el juego.

La concepción del género se ha visto tan ampliada durante el paso de los años que se ha alejado de una forma altamente significativa de la idea inicial. Hoy en día abarca a todo aquel juego en el que el usuario controle a uno o más personajes con los que cumplir misiones u objetivos, normalmente con el fin de presenciar el desarrollo y desenlace de una historia.

De esta forma, se distancia de la acepción original de la que tomó nombre en sus inicios, llegando incluso a tomar el nombre de CRPG en algunos ámbitos, *Computer Role-Playing Game*, con el fin de separar ambos conceptos debido a la digresión de sus significados con el paso del tiempo[5].

4 https://videojuegos.fandom.com/es/wiki/G%C3%A9neros_de_videojuegos
5 https://ia801001.us.archive.org/1/items/crpg-book-1.0/crpg-book-1.0.pdf

HISTORIA DE LOS RPGS

Como se ha mencionado previamente, los primeros videojuegos RPG nacen de la adaptación a ordenador y consola de los juegos de rol de mesa.

Partiendo de los juegos de simulación de beisbol, del rol de fantasía, generalmente basado la obra de J. R. R. Tolkien y en los ya mencionados juegos de mesa, parten los primeros antecesores de los RPGs.

Cabe mencionar la importancia del "Colossal Cave Adventure", pues aun tratándose de un juego de simulación, introduce mecánicas clave para el desarrollo de los primeros RPGs.

A finales de los años 70, con la revolución de los ordenadores personales, se empiezan a desarrollar los primeros CRPGs disponibles para un amplio público. Entre estos se encuentran "Wizard's Castle" y "Eamon", que además de ser pioneros en su campo, provocaron una nueva ola de juegos, inspirando a desarrolladores y jugadores a crear nuevo contenido.

Tras una expansión considerable del género en la década de los 80, el éxito previo obtenido empezó a decaer debido al cambio en la calidad de los productos desarrollados. Sin embargo, la caída de los principales desarrolladores de CRPGs no solo dejó un hueco en el mercado, sino que además sentó la base de una nueva remesa de juegos. Todo esto provocó una subida exponencial de los CRPGs, obteniendo productos que hoy en día son considerados clásicos del mundo de los videojuegos, como el "Elder Scrolls" de Bethesda o el "Diablo" de Bizzard, productos tan populares que sus secuelas siguen generando beneficios en la actualidad.

Dentro de los subgéneros de los RPGs, cabe destacar los JRPGs (Japanese Role-Playing Games), debido a la alta influencia de estos en el producto desarrollado.

Pese a ser muy parecidos, la corriente japonesa tiende a caracterizarse por una mecánica de combate basada en turnos, un estilo de diseño oriental, una historia fija con final único y una temática de fantasía.

Esta rama de los videojuegos surge a principios de los años 80, y aunque no se sabe con precisión cual fue el primero en ser desarrollado, uno de los posibles es el "Dragon and Princess" publicado en 1982. Sin embargo, se trata de un periodo extremadamente experimental en lo referente a los videojuegos por lo que no se conoce a ciencia cierta cuál fue el que inició este subgénero.

Aun con su convulso inicio, el género JRPG se consolidó rápidamente, gracias a juegos como "Dungeon", el cual marcó un antes y un después en la corriente asiática de los videojuegos de rol.

Aunque los JRPGs para ordenador sufrieron un gran estancamiento debido a la llegada a Japón de Nintendo y Sega, el género consiguió sobrevivir migrando a las consolas, abasteciendo el mercado de productos altamente lucrativos y populares incluso en la actualidad, entre los que destacan la saga "Pokémon" o la saga "Final Fantasy".

ESTUDIO DE MERCADO

Clientes Potenciales (Target)

Tras finalizar el desarrollo la idea del producto y realizar un análisis sobre el estado del mercado de los RPGs en la actualidad, se observó la gran amplitud de diferentes perfiles a los que se podía enfocar la venta del juego.

En primer lugar, el género CRPG, aun con sus decaídas, ha resistido al paso del tiempo, encontrándose en la actualidad en uno de sus mejores momentos y produciendo juegos altamente populares tales como las sagas Dark Souls[6] o Mass Effect[7].

Añadido a esto, se empieza a observar una nueva tendencia de remakes de juegos antiguos, encontrando versiones modernas en producción de clásicos como el "Neverwinter Nights" o el "Final Fantasy VII[8]". De esta forma la clientela de los RPGs se extiende más allá de lo habitual en cuanto a la edad, alcanzando a un tipo de usuario mayor de lo habitual.

Por todo esto, el público al que se puede destinar el producto es muy extenso, pudiéndose marcar un rango de clientes potenciales de 18 años en adelante.

Sin embargo, con el fin de marcar de forma más controlada los parámetros de los experimentos, se trató de buscar *early adopters* que tuvieran entre 18 y 39 años, teniendo en cuenta las franjas de edad en que se encuentran la mayoría de los jugadores de videojuegos.

Productos Similares

Con el fin de llevar a cabo una investigación y comparación de diferentes productos que compartan similitudes con ComeBack, se inició un proceso de búsqueda y análisis al principio del desarrollo del proyecto.

Tras valorar y comprobar una serie de juegos, se redujo la lista a cuatro opciones diferentes, elegidas tanto por su popularidad como por las características que presentan en común.

Pese a que el producto desarrollado no puede clasificarse en su totalidad como un JRPG, debido a que toma características tanto de la variable oriental como de la occidental, se ha decidido estudiar cuatro juegos de la variante japonesa, debido a la alta influencia del subgénero durante todo el proceso de desarrollo.

Aunque se pretendía que los videojuegos fueran lo más actuales posible, no se ha querido elegir la opción más reciente de cada saga, si no la más parecida o la que posea mayor relevancia en el mercado actual. Al tratarse de un género que tuvo su momento álgido hace dos décadas, se encuentran dentro de la comparación títulos que a simple vista pueden estar desfasados, pero que hoy en día aún cuentan con gran público.

Se describen y estudian a continuación los cuatro productos seleccionados y se comparan los puntos más relevantes de todos ellos posteriormente en una tabla.

6 https://store.steampowered.com/app/211420/DARK_SOULS_Prepare_To_Die_Edition/?l=spanish
7 https://www.origin.com/esp/es-es/store/mass-effect/mass-effect-trilogy
8 https://store.na.square-enix-games.com/en_US/product/562559/final-fantasy-vii-remake-ps4

Tales of Vesperia[9][10]

Tales of Vesperia es uno de los muchos juegos de la saga Tales of, desarrollados por Namco Tales Studio y publicados por Namco Bandai Games[11].

Salió al mercado en 2008 para Xbox 360. Sin embargo, tras la popularidad del juego con el paso de los años tanto en Japón como de forma internacional, se creó una nueva edición de este publicada a inicios de 2019 (*Tales of Vesperia: Defenitive Edition*), la cual cuenta con soporte para Xbox One, PlayStation 4, Nintendo Switch y Microsoft Windows.

El precio original de salida en Japón fue de 7800 yenes (actualmente la conversión aproximada sería de 63€). Hoy en día, si se hace uso de los distribuidores oficiales, sólo se puede adquirir la versión de definitiva, cuyo valor depende de la plataforma para la que se adquiera el juego y de si se obtiene el formato estándar o el premium, que incluye la banda sonora, un libro de arte y una caja metálica, además de una serie de objetos coleccionables. Los diferentes precios de la *Definitive Edition* se comparan en la Figura 4.

	Standard Edition	Premium Edition
Nintendo Switch	49'99€	79'99€
PlayStation 4	39'99€	49'99€
Xbox One	39'99€	49'99€
PC	39'99€	

Figura 4: Tabla comparativa de los precios de la Definitive Edition en cada plataforma

La *Defenitive Edition* cuenta con todas las novedades que se incluyeron en la versión de PlayStation 3, entre las que se incluyen nuevas zonas, enemigos, ataques, personajes y diálogos con estos mismos. También se añadió al juego internacional, que en su origen solo podía ser jugado en inglés, los doblajes de voz originales. Así mismo, se añadieron subtítulos en francés, alemán, italiano, ruso y español.

Tales of Vesperia cuenta una historia que transcurre en el planeta Terca Lumireis. La base de la trama se centra en que la vida cuotidiana de todo personaje se apoya sobremanera en el uso de unos antiguos artefactos llamados blastias, que son utilizados de mil formas distintas, desde propulsores de vehículos hasta generadores de barreras mágicas.

El potencial de todo blastia surge del aer, una poderosa fuente de energía que solo se puede controlar haciendo uso de esta tecnología. Es tal su alcance que puede ser utilizada como arma a través de un bodhi blastia, el portador que lleve este accesorio podrá hacer uso de la fuerza del aer utilizandolo como canalizador.

La historia gira alrededor de Yuri Lowell, un antiguo miembro de la guardia imperial, que, tras ser injustamente culpado del robo de un blastia, conoce a Estelle, a la que accede a ayudar, y juntos emprenden una aventura en la que derrocarán a una malvada organización y salvarán el mundo junto con sus compañeros.

9 https://tov10th.tales-ch.jp/ (japonés)
10 https://es.bandainamcoent.eu/tales-of/tales-of-vesperia-definitive-edition (español)
11 https://es.bandainamcoent.eu/

Figura 5: Capturas de Pantalla de Tales of Vesperia, versión de Playstation 3

El juego está basado en un sistema de estadísticas, como vida o experiencia, que alteran la capacidad de combate de los personajes, haciéndolos más fuertes, resistentes o débiles.

Como se puede observar en la figura 5, este videojuego consta de un sistema de inventario denominado regla de 99, el cual tras hacer click en un objeto nos muestra su descripción y una imagen de este.

En este videojuego existen cuatro tipos de habilidades diferentes, siendo estas:

- Daño
- Defensa
- Soporte
- Movimiento

Cada uno de los personajes se especializa en una o varias de estas, haciendo decisiva la combinación apropiada de perfiles a la hora de elegir al equipo. El grupo contará de cuatro integrantes, entre los que se encontrarán el protagonista, Yuri, y tres miembros adicionales a elección del jugador. Originalmente había seis compañeros entre los que tomar dicha decisión, sin embargo, se añadieron dos nuevos en la versión de PlayStation 3.

En la Definitive Edition el usuario puede elegir como luchar. Pues el juego cuenta con diferentes opciones de manejo de los acompañantes. Así pues, si se marca en automático solo tendrá que combatir con el protagonista. En modo semiautomático podrá seleccionar las habilidades del resto, pero no sus movimientos. Por último, si se hace uso del modo manual se podrá controlar a cualquier compañero de la misma forma que a Yuri. Este último se utiliza sobre todo para jugar con otros usuarios en la misma batalla.

Tales of Vesperia, junto con Tales of Symphonia, es una de las entregas más icónicas de la saga. En su versión original de Xbox 360 obtuvo una media de 79/100 en Metacritic y 83/100 en la versión de Nintendo Switch de la Definitive edition[12][13]. Nombrado juego del mes por IGN en 2008 y clasificándolo en el puesto 57 de los 100 mejores RPGs de la historia.[14][15]

12 https://www.metacritic.com/game/xbox-360/tales-of-vesperia
13 https://www.metacritic.com/game/switch/tales-of-vesperia-definitive-edition
14 https://www.ign.com/articles/2008/08/29/x360-game-of-the-month-august-2008
15 https://www.ign.com/lists/top-100-rpgs/57

Final Fantasy VII[16]

Aun contando con un gran número de entregas, el Final Fantasy VII sigue siendo en día de hoy un referente en el mundo de los RPGs. El juego desarrollado y publicado por la compañía Square[17] en 1997 es considerado el más emblemático de la serie Final Fantasy por gran mayoría de los usuarios, siendo el primero de su saga en poseer gráficos 3D y control intuitivo de la cámara.

El juego fue creado originalmente para PlayStation, no obstante, debido a su creciente y duradera popularidad se fueron publicando nuevas versiones y es jugable incluso en las consolas actuales como se indica en la Figura 6.

Año	Plataforma
1998	Windows
2009	PSN(PlayStation Network)
2012	PC (vía Steam)
2015	iOS PlayStation4
2016	Android
2018	Nintendo Switch Xbox One

Figura 6: Tabla de años de publicación del Final Fantasy VII según la plataforma

Es tal la notoriedad de este que en 2014 Square Enix anunció la creación de una adaptación para PlayStation 4 debido a las peticiones de los fans de la saga.

En cuanto al modelo de mercado, el juego se puso a la venta a 49’99$ originalmente. En la actualidad cuesta 12’99€ en su versión para PC y 15’99€ en la de PlayStation 4. La adaptación, que todavía no ha sido publicada, tiene un precio de preventa que asciende a 64’99€.

El juego da lugar en un mundo controlado por la corporación Shiura, que, con el fin de conseguir energía, lo que está provocando la destrucción del planeta drenando toda la vida que hay en este. El protagonista, Cloud Strife se une a la organización ecoterrorista AVALANCHA con el fin de detenerlos.

A medida que avance la historia encontrará toda serie de peligros y aliados, descubriendo una compleja trama hasta enfrentarse a su archienemigo Sefirot.

Figura 7: Capturas de pantalla del Final Fantasy VII

16 https://finalfantasyviipc.square-enix-games.com/en
17 https://www.square-enix.com/

El número de compañeros total asciende a ocho, y se puede utilizar a dos de ellos junto a Cloud en la batalla.

Como se puede observar en la Figura 7, este videojuego que se ha convertido en un clásico de los videojuegos RPGs utiliza un sistema de inventario Regla de 99 unicamente conteniendo elementos de texto.

En lo referente a la crítica, se ha convertido en un clásico del mundo de los videojuegos, obteniendo un 92/100 en Metacritic en su versión de PlayStation[18] y ganando los premios *Console Adventure Game of the Year y Console RPG of the Year* en los D.I.C.E awards de 1998.[19]

Pokémon Sol y Pokémon Luna[20]

Desarrollados por Game Freak[21] y distribuidos por Nintendo[22] en 2016, forman parte del enorme universo de Pokémon. Como gran cantidad de sus predecesores, fueron lanzados en pareja, con leves alteraciones entre ambos, por eso se ha decidido tomarlos como uno solo a la hora de realizar el estudio.

Ambos pueden ser jugados tanto en Nintendo 3DS como 2DS y en todas las versiones posteriores de estas y tanto los juegos originales como sus versiones superiores de 2017, Pokémon Ultrasol y Ultraluna, tienen un precio de venta de 39’99$ en sus versiones para Nintendo 3DS.

Aparte de producir los juegos de la saga en pares, Nintendo también tiende a desarrollar una nueva versión al poco tiempo, que incluya más contenido que la original. En este caso, se añadieron nuevas historias, pokémons, zonas, misiones y eventos.

La trama de los Pokémon sigue imperecedera desde el primero de ellos, con casi ninguno o con cambios muy sutiles. El protagonista de la historia desea ser entrenador pokémon y ganar la liga de combate. Para ello deberá elegir uno entre tres pokémons iniciales, usualmente de tipo agua, fuego y planta, y avanzar y explorar el mundo para formar un equipo de 6 con el que conseguir todas las medallas de los gimnasios, que finalmente conceden la entrada en la liga.

Usualmente la trama se mezcla con un amigo de la infancia o rival contra el que competir y con el Team Rocket, una malvada organización con la que el protagonista se topará en su aventura.

Figura 8: Capturas de pantalla del Pokemon Ultrasol

El sistema de inventario como muestra la Figura 8, consta de un sistema de inventario Regla de 99 que, tras seleccionar un objeto, nos muestra una miniatura de este y su descripción.

Tanto los juegos originales como sus versiones superiores poseen buenas críticas, obteniendo respectivamente 87/100 y 84/100 en Metacritic[23][24] y recibieron los siguientes premios en los

18 https://www.metacritic.com/game/playstation/final-fantasy-vii
19 https://www.interactive.org/games/video_game_details.asp?idAward=1998&idGame=722
20 https://www.pokemon-sunmoon.com/en-us/
21 http://www.gamefreak.co.jp/
22 https://www.nintendo.es/
23 https://www.metacritic.com/game/3ds/pokemon-sun
24 https://www.metacritic.com/game/3ds/pokemon-ultra-sun

Japan Game Awards de 2017: Award for Excellence, Best Sales Award y Global Award Japanese Product.[25]

Persona 5[26]

Siendo la entrega más reciente de la saga, el Persona 5 fue desarrollado y distribuido por Altus[27] y puesto a la venta en 2016 tanto para PlayStation 3 como PlayStation 4.

Al igual que sus antecesores, se trata de un juego que combina rol, exploración de mazmorras y simulación social. Además de un spin-off, el juego cuenta con la futura producción de una versión mejorada, Persona 5 Royal, programada para ser puesta a la venta en 2020 que incluirá un nuevo personaje, nuevas mazmorras, un nuevo semestre, nuevas interacciones y soporte para PlayStation 4 Pro.

El precio del juego en Estados Unidos del Persona 5 es de 19'99$ para su versión de PlayStation 4 y 24'87$ para la de PlayStation 3. El precio en el mercado europeo es diverso dependiendo de la región, pero tiende a rondar los 40€ en gran mayoría de los casos, siendo más cara también la versión para PlayStation 3.

Persona 5 transcurre aparentemente en un ambiente escolar, en el que el protagonista va a clases y trabaja a tiempo parcial para mejorar sus habilidades y atributos. Sin embargo, el juego cuenta con un plano paralelo, el Metaverso, donde el personaje debe luchar contra sombras haciendo uso de sus personas, unos monstruos utilizados para combatir.

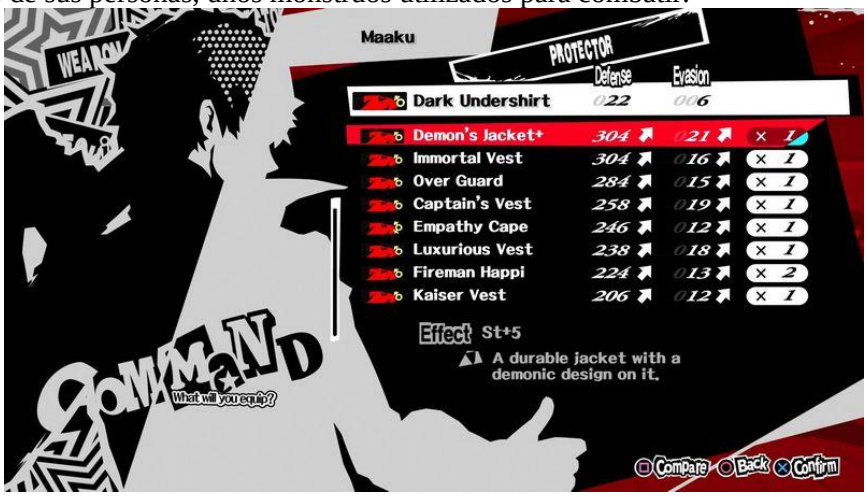


Figura 9: Capturas de pantalla del Persona 5

A medida que avanza la historia el protagonista puede mejorar su relación con sus siete compañeros de equipo e incluso llegar a entablar relaciones románticas con ellos. Prestar atención al grupo proporciona ventajas tanto a la hora de avanzar en la historia como de luchar en los combates.

El Persona 5 utiliza un sistema de inventario regla de 99 sin elementos visuales, únicamente con elementos de texto.

El Persona 5 no solo destaca por su trama si no por lo interesante que es su diseño en comparación al resto. Tal como se observa en la Figura 9, se trata de un estilo único y llamativo.

En lo referente a la crítica el juego contó con gran apoyo, obteniendo un 93/100 en Metacritic[28] y puntuaciones elevadas en otras plataformas. Siendo nombrado Best Role-Playing Game en The Game Awards 2017 [29] y Best RPG en los Best of 2017 de IGN[30].

25 http://awards.cesa.or.jp/2017/en/prize/year/11.html
26 https://atlus.com/persona5/home.html
27 https://atlus.com/

Comparación

Una vez realizado un estudio con el que obtener las características más interesantes de cada uno de los juegos y profundizar en los detalles que hacen que sean exitosos, se ha recopilado la información más útil de cada uno de ellos con el fin de poder aplicar dicho aprendizaje en este producto.

Se muestra en la Figura 10 la comparación entre los elementos más importantes de cada uno de los videojuegos, incluyendo ComeBack entre ellos.

	Tales of Vesperia	Final Fantasy VII	Pokémon Sol y Luna	Persona 5	ComeBack
Número de compañeros disponibles	6 - 8	8	302 - 403	7 - 8	3
Número de personajes en el combate	4	3	1 - 2	4	4
Modo campaña / historia	Si	Si	Si	Si	Si
Multijugador	Si	No	Si	No	No
Tipo de combate	EFR - LMBS	ATB	Combate por turnos	Combate por turnos	Combate por turnos con tiempo
Árbol de habilidades	No	No	Parcial	No	Si
Personaje personalizable	No	No	Total	En DLC	Parcial
Inventario	Regla de 99	Regla de 99	Regla de 99	Regla de 99	Regla de 99 con elementos visuales
Zonas	6	18	5	1	12
Formato	Físico (Original) Digital y físico (Definitive)	Digital y físico	Digital y físico	Físico	Digital
Logros	Si	Si	Si	Si	Si
Subgéneros	RPG	RPG	RPG Aventura	RPG Exploración de mazmorras	RPG
Ángulo / Cámara	Fija 3ªPersona	Móvil en el mapa, fija en combate	Fija 3ªPersona	Móvil 3ªPersona	Fija 3ªPersona

Figura 10: Tabla comparativa de características de los productos similares a ComeBack

28 https://www.metacritic.com/game/playstation-4/persona-5
29 https://thegameawards.wpengine.com/history/2017-2/
30 https://www.ign.com/wikis/best-of-2017-awards/Best_RPG

Plataformas de distribución

A la hora de distribuir el producto, se determinó desde el inicio hacerlo por medio de formato digital.

Esta forma distribución se ha popularizado en las últimas décadas, ya que reduce los costes de producción, y, proporciona una facilidad para la compra al usuario, el cual únicamente necesita una conexión a internet para adquirir el producto.

Debido a esto, se ha visto un aumento en ventas de los videojuegos en formato digital, incluso aun siendo comprados en tienda, muchos de ellos vienen con una clave de descarga en lugar de en formato físico.

Consecuentemente, se explican las plataformas que se van a utilizar para la distribución de ComeBack.

Patreon

Patreon es una plataforma de financiación colectiva creada en 2013 por Jack Conte y Sam Yam. La web se utiliza para financiar proyectos a través de un sistema de subscripción.

Los interesados en financiar un producto o proyecto se suscriben pagando de forma periódica a cambio de una serie de ventajas, desde acceso previo a las diferentes versiones hasta la obtención de merchandising relacionado con el proceso.

Los premios o ventajas que adquieren los inversores son completamente personalizables por los creadores del producto, dando así a los clientes infinitas opciones a la hora de configurar su patreon.

La plataforma superó los 50.000 creadores en 2017 y es una de las referentes en el campo del crowfunding, sirviendo de medio de financiación para un gran rango de diferentes artistas.

Se decidió hacer uso de una herramienta de financiación de este estilo para poder recibir ingresos durante el proceso de creación del producto y comenzar la captación de usuarios lo antes posible.

Steam

Steam consta de una plataforma de distribución de videojuegos para ordenador fundada por Valve en 2003. Aunque en sus orígenes se utilizó únicamente como medio de compra y actualización de los juegos de la empresa, evolucionó en los años posteriores a su lanzamiento a una especie de tienda digital de todo tipo de juegos.

Hoy en día, Steam cuenta con 90 millones de usuarios mensuales activos con casi 30.000 juegos disponibles y es accesible desde 237 países en 21 idiomas diferentes.

Además de una tienda digital, también ofrece servicios para compartir contenido creado por otros usuarios y un chat para que estos puedan hablar entre ellos.

Proyección económica

Tras la realización del estudio de mercado, es de gran importancia plantearse los posibles costes y beneficios provocados por la consecución del proyecto. De esta forma se podrá plasmar si la evolución de este puede llegar a ser beneficiosa.

Aunque se traten de parámetros hipotéticos, pues no se pueden saber de antemano los posibles gastos e ingresos, plantear una proyección económica puede ayudar a observar si el producto en cuestión es viable.

Se muestra en la Figura 11 la proyección realizada para el proyecto.

	Semestre 1 (01/18 - 06/18)	Semestre 2 (07/18 - 12/18)	Semestre 3 (01/19 - 06/19)	Semestre 4 (07/19 - 12/19)	Semestre 5 (01/20 - 06/20)	Semestre 6 (07/20 - 12/20)
Ingresos						
Patreon 1			20	50	100	600
Patreon 2			5	20	50	200
Patreon 3			0	5	20	100
Licencia del juego				200	600	2000
Gastos						
Personal	16.200	16.200	900	900	900	900
Cursos	30					
Equipo	64,99					
Beneficios (Ingresos-Gastos)	-16.294,99	-32.494,99	-33.036,49	-30.682,99	-22.479,19	12.146,81

Figura 11: Tabla de la proyección económica en 3 años de ComeBack

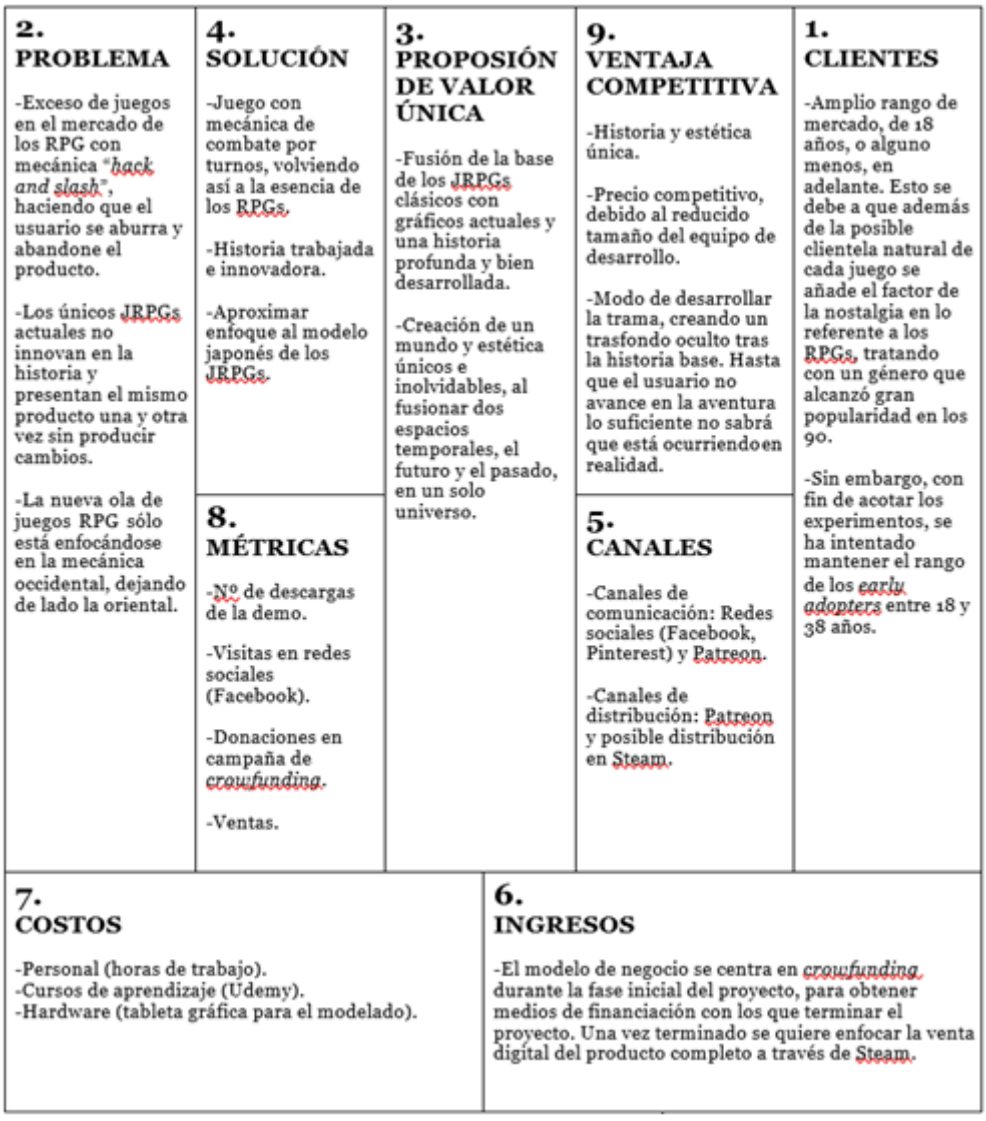
Debido a la magnitud del videojuego, se planea seguir trabajando en él aun después del sexto semestre, añadiendo contenido para los subscriptores de Patreon. En cambio, se planea reducir la carga de trabajo a partir del tercer semestre, las tareas a efectuar constarán de tareas repetitivas y que consumirán un tiempo mas reducido.

Los beneficios de la proyección son negativos durante los cinco primeros semestres, debido al tiempo necesario para llevar a cabo el primer producto mínimo. No obstante, a partir del sexto no solo se obtienen beneficios, sino que se espera un aumento exponencial de usuarios.

Lean Canvas

En cualquier tipo de proyecto, y en especial en los de emprendimiento, es altamente importante tener claros los motivos por los que se realiza el proyecto, los costes, el modo de distribución y todo punto vital que pueda afectar al producto a desarrollar.

Entre las diferentes herramientas con las que facilitar la especificación de todo esto, se encuentra el Lean Canvas, con el que se puede plasmar de forma clara y rápida de comprender todo lo esencial para tener en cuenta para llevar a cabo el proceso de desarrollo. Se muestra en la Figura 12 el Lean Canvas desarrollado durante este proyecto.



2. PROBLEMA

-Exceso de juegos en el mercado de los RPG con mecánica "*hack and slash*", haciendo que el usuario se aburra y abandone el producto.

-Los únicos JRPGs actuales no innovan en la historia y presentan el mismo producto una y otra vez sin producir cambios.

-La nueva ola de juegos RPG sólo está enfocándose en la mecánica occidental, dejando de lado la oriental.

4. SOLUCIÓN

-Juego con mecánica de combate por turnos, volviendo así a la esencia de los RPGs.

-Historia trabajada e innovadora.

-Aproximar enfoque al modelo japonés de los JRPGs.

8. MÉTRICAS

-Nº de descargas de la demo.

-Visitas en redes sociales (Facebook).

-Donaciones en campaña de *crowfunding*.

-Ventas.

3. PROPOSIÓN DE VALOR ÚNICA

-Fusión de la base de los JRPGs clásicos con gráficos actuales y una historia profunda y bien desarrollada.

-Creación de un mundo y estética únicos e inolvidables, al fusionar dos espacios temporales, el futuro y el pasado, en un solo universo.

9. VENTAJA COMPETITIVA

-Historia y estética única.

-Precio competitivo, debido al reducido tamaño del equipo de desarrollo.

-Modo de desarrollar la trama, creando un trasfondo oculto tras la historia base. Hasta que el usuario no avance en la aventura lo suficiente no sabrá que está ocurriendo en realidad.

5. CANALES

-Canales de comunicación: Redes sociales (Facebook, Pinterest) y Patreon.

-Canales de distribución: Patreon y posible distribución en Steam.

1. CLIENTES

-Amplio rango de mercado, de 18 años, o alguno menos, en adelante. Esto se debe a que además de la posible clientela natural de cada juego se añade el factor de la nostalgia en lo referente a los RPGs, tratando con un género que alcanzó gran popularidad en los 90.

-Sin embargo, con fin de acotar los experimentos, se ha intentado mantener el rango de los *early adopters* entre 18 y 38 años.

7. COSTOS

-Personal (horas de trabajo).
-Cursos de aprendizaje (Udemy).
-Hardware (tableta gráfica para el modelado).

6. INGRESOS

-El modelo de negocio se centra en *crowfunding* durante la fase inicial del proyecto, para obtener medios de financiación con los que terminar el proyecto. Una vez terminado se quiere enfocar la venta digital del producto completo a través de Steam.

Figura 12: Tabla: Lean Canvas de ComeBack

Análisis DAFO

Otra herramienta a la hora de analizar la calidad del proyecto es el análisis DAFO, el cual consiste en la clasificación de las diferentes debilidades, amenazas, fortalezas y oportunidades halladas comparando el producto a analizar con el resto del mercado.

Se presenta pues un análisis DAFO de ComeBack, mostrado en la Figura 13, con el fin de comparar sus puntos fuertes y defectos respecto a la competencia, utilizando para ello la información recopilada durante el estudio de mercado.

FORTALEZAS	**DEBILIDADES**
-Precio de mercado bajo. -Precio de desarrollo. -Diseño único y original de personajes. -Diseño único y original de escenarios. -Historia innovadora, con trasfondo e inusual. -Rango amplio de potenciales clientes. -Personificación del personaje. -Mundo amplio. -Fácil aprendizaje. -Fácil memorización de mecánicas. -Amplia rama de clases. -Amplia rama de talentos.	-Producto inicialmente pequeño. -Número reducido de compañeros. -Poca experiencia en financiación y marketing. -Falta de capital.
OPORTUNIDADES	**AMENAZAS**
-Hueco en el mercado de los RPG, en el estilo oriental. -Ausencia de productos similares. -Auge de los RPGs en la actualidad. -Estabilización del *crowfunding* como método de financiación de videojuegos.	-Gran cantidad de productos en el mercado. -La competencia son clásicos de la historia de los juegos.

Figura 13: Tabla: Análisis DAFO del producto

El grado de personificación del género con que se ha desarrollado el producto, escogiendo características tanto de la corriente oriental como occidental, junto con la originalidad de la historia y la estética dan como resultado un juego único y capaz de destacar sobre la competencia.

Conclusiones de la evaluación de la idea de negocio

Los RPGs han sufrido variaciones en su popularidad a lo largo de su historia, aun así, han logrado perseverar como uno de los géneros de videojuegos mas populares, extendiéndose a plataformas diferentes y combinándose con un gran numero de géneros.

En la última década se ha observado un regreso triunfante de los RPGs, especialmente aquellos de características occidentales. A pesar de no estar siendo explotado excesivamente, el subgénero oriental sigue funcionando como concepto actualmente con sagas altamente populares y lucrativas.

Se ha percibido un vacío en el mercado para productos que contengan características de ambas corrientes.

Tras un estudio de la competencia y de las características de ComeBack, se ha llegado a la conclusión de que el producto tiene unas mecánicas y una temática que lo diferencian del resto, ya que reúne propiedades de ambas corrientes. Pese no poseer experiencia en lo referente al marketing, se considera que la propuesta puede ser exitosa, y por ello se procedió con el desarrollo de la idea de negocio que se explicará en el próximo capítulo.

CAPÍTULO 4 DESARROLLO DE LA IDEA DE NEGOCIO

El proceso de desarrollo de software es la estructura que seguir para llevar a cabo la creación de producto específico. Aun existiendo un grandioso numero de modelos a seguir o en los que basarse, todavía no se ha logrado establecer un proceso universal que se pueda utilizar de forma estándar para enfocar como realizar la elaboración de un producto. Esto es debido al exceso de características que varían entre diferentes proyectos. Esto es debido a que generalmente, se tratan de ideas únicas.

El desarrollo de videojuegos no es una excepción a la regla. Se trata de un proceso difícil, a pesar de la simplicidad de la idea inicial, que tiende a complicarse a medida que evoluciona, en especial si no se trata de forma detallada.

No solo es fácil cometer errores, sino que también es mas difícil y costoso corregir o solucionar dichos errores cuanto mas avanzado esté el producto.

A continuación, se muestra una serie de cambios y problemas solucionados durante el desarrollo de ComeBack como ejemplo.

Durante la planificación, durante el análisis de los requisitos del proyecto, se olvidó planificar la fase de aprendizaje de las tecnologías que se requerían para el proyecto. Aun siendo una equivocación en la fase inicial del proyecto, el no haber tenido en cuenta desde el principio la falta de experiencia, produjo una reorganización de la planificación del resto de tareas a realizar. Debido a esto se tuvieron que retrasar todas las tareas dado que toda la fase inicial del proyecto se acabó dedicando al estudio de las herramientas de desarrollo del producto.

Otro percance que destacar fue la decisión de pasar de gráficos en dos dimensiones a tres dimensiones. Después de haber iniciado la etapa de diseño, se observó que los motivos por los que se había tomado la decisión de crear el juego en dos dimensiones, siendo uno de ellos acelerar el proceso de creación del producto, no compensaban la falta del efecto que se deseaba causar en el usuario.

Por ello, se decidió realizar el desarrollo en 3D tanto de los personajes como de los objetos. Esto no solo produjo que las fases siguientes fueran más complejas, sino que al tratarse una equivocación en la etapa de diseño, causó la necesidad de retornar a la planificación para cambiar de un enfoque de dos dimensiones a uno en tres dimensiones.

MAPA DE CARACTERÍSTICAS

Debido a que se trata de un proyecto del cual se busca conseguir rentabilidad mediante herramientas de *crowdfunding*, la mejor forma afrontarlo es mediante un desarrollo ágil del mismo, con el fin de obtener lo antes posible un producto mínimo que poder comercializar.

En consecuencia, se creó una lista de objetivos a desarrollar, como lo pueden ser crear a la protagonista, realizar el combate o implementar los menús. Todos estos objetivos formarán parte del mapa de características mostrado en la Figura 14, en el que incluirán todas las funcionalidades que se deseen agregar al proyecto.

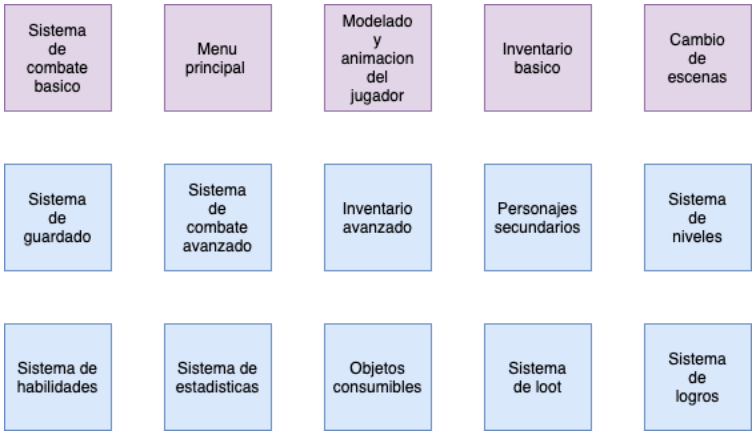


Figura 14: Mapa de características de ComeBack, en el que se remarca en morado la extensión del primer MVP

PRIORIZACIÓN

Tras haber obtenido el mapa de características, se pasó a organizar las tareas según su relevancia a la hora de poder crear un producto viable.

Como se ha indicado anteriormente, en el desarrollo ágil prevalece la obtención temprana de un producto que no comprenda la totalidad de las funciones pero que a su vez no contenga errores y funcione de forma correcta.

Así pues, se procede a determinar las funcionalidades esenciales para obtener dicho producto, que se denomina *MVP* o *Minimum Viable Product*.

Todo el desarrollo del proyecto será basado en MVPs, de forma que cada vez que se realice uno, se publicará una nueva versión del juego y se tomarán datos de opinión de usuarios con el fin de comprobar la satisfacción de estos.

A partir de la información recogida tras el despliegue de una versión, se analizarán los datos y se plantearan correcciones o cambios en los ámbitos que no resulten atractivos y se mantendrá o incluso fortalecerá todo aquello con lo que estén satisfecho.

Una vez se ha definido el mapa de características, se deciden cuáles de ellas serán incluidas en el Mínimo Producto Viable. Teniendo en cuenta los recursos y el tiempo disponible, fueron elegidas las siguientes cinco características.

1. Sistema de combate básico:

   Permite que el jugador ataque a los enemigos con ataques normales o mágicos y permite a los enemigos realizar lo mismo. Ambos dependen de una barra de tiempo que se va llenando y controla los turnos de la pelea.

2. Inventario básico:

   Permite al jugador abrir el inventario desde el menú del juego para observar objetos obtenidos y de los efectos de estos, también se muestra el número de objetos que posee de cada tipo.

3. Modelado y animación de los personajes:

   Proceso de creación de los personajes del juego a través del modelado 3D y de un proceso de animación.
   Tanto aliados como enemigos, tienen animaciones de movimiento, ataque y muerte en el combate, además las animaciones del personaje en la escena principal que responden al movimiento que decida el jugador.

4. Cambio de escenas:

   Permite cambiar entre la escena principal y la del combate al entrar en contacto con un enemigo y de vuelta cuando el combate termina en victoria, en caso de derrota se retorna a la pantalla del menú principal.

5. Menú principal básico:

   Crear una pantalla de menú principal con la posibilidad de al menos empezar una nueva partida.

La principal razón de la elección de las características anteriores se debe a que permiten una versión jugable del proyecto básica que permita empezar a recibir opiniones de los *early-adopters*.

DESARROLLO

Primer MVP

Tras haber definido las características del proyecto y aquellas que deben comprenderse en el primer Mínimo Producto Viable se pasa a definir el concretar.

Debido a la gran cantidad de características incluidas, se decidió ramificar cada una de ellas en tareas más pequeñas que fueran añadiéndose conforme se realizaran.

Se propuso un plazo de un semestre para el tiempo de desarrollo antes del primer experimento. No obstante, el progreso se vio retrasado en septiembre debido a un problema con el repositorio de versiones.

El repositorio de versiones utilizado en aquel momento, Bitbucket, tiene la opción de borrar un repositorio tanto en el propio Bitbucket como en local. La inexperiencia del equipo con Bitbucket causó que todo el desarrollo de la tarea del sistema de combate se perdiera.

En cambio, se logró recuperar una versión anterior de la tarea del sistema de combate que ayudó a acelerar el proceso de reescritura de la tarea.

Esta situación, junto a las equivocaciones que se han mencionado con anterioridad, retrasó el primer experimento hasta mitad del primer mes del siguiente semestre. En ese momento, el equipo contactó a varios estudiantes de la universidad de diferentes facultades para que pudieran probar la primera versión del juego y realizaran una encuesta con la que poder revelar sus opiniones sobre el producto.

La prueba consistía en comenzar el videojuego y completar una partida sin asistencia alguna. Generalmente los resultados fueron aceptables, a pesar de haberse encontrado diversos problemas, algunos que afectaban a los usuarios más experimentados y otros a los que apenas poseían experiencia en el campo en cuestión.

Requisitos

Así pues, se marcaron como requisitos mínimos para la obtención del primer MVP, el desarrollo de un nivel completo, el cual incluyera como mínimo:

- Personaje principal
- Un enemigo
- Sistema de combate, con cambio de escena a la pantalla de lucha y vuelta a la original al terminar la batalla.
- Menú principal para acceder y salir del juego
- Sistema de inventario que defina los tipos de objetos que pueda encentrar el jugador a lo largo del juego
- Menu in-game que muestre el estado del jugador y el inventario
- Sistema de diálogos que permita sumergir al jugador tanto en la historia como el lore del videojuego

Problemas enfrentados

Además del problema citado anteriormente con el repositorio de versiones, surgieron una serie de sucesos durante el desarrollo de la parte técnica del proyecto que aplazaron los plazos pronosticados en un principio.

A continuación, se especifican los problemas relacionados con el ámbito comprendido en el desarrollo de este documento, todos ellos serán explicados en sus respectivos apartados correspondientes en el capítulo 5.

- Cambio de tecnología de dos dimensiones a tres dimensiones, lo que provocó no solo una pérdida de tiempo, sino que además añadió una gran carga de trabajo.
- Cambio en el sistema de texturas. Al intentar importar las texturas se hallaron tantos problemas que se acabó optando por cambiar a otro sistema diferente.

Experimento

El primer experimento fue realizado compartiendo un enlace de descarga con la versión ejecutable del primer MVP, junto con un enlace a una encuesta de Google Forms. La encuesta fue realizada por 19 participantes, de los cuales entre un 70 y un 80 por ciento eran hombres de entre 18 y 30 años y se consideraban usuarios de videojuegos habituales y experimentados.

Las primeras preguntas, fueron acerca de los géneros y temáticas preferidas de los encuestados, las estadísticas de las respuestas a esta pregunta se pueden observar en la Figura 15. En esta imagen se puede observar que la mayoría tiene una preferencia por los juegos de Rol seguido de cerca por los géneros de Aventura y Estrategia.

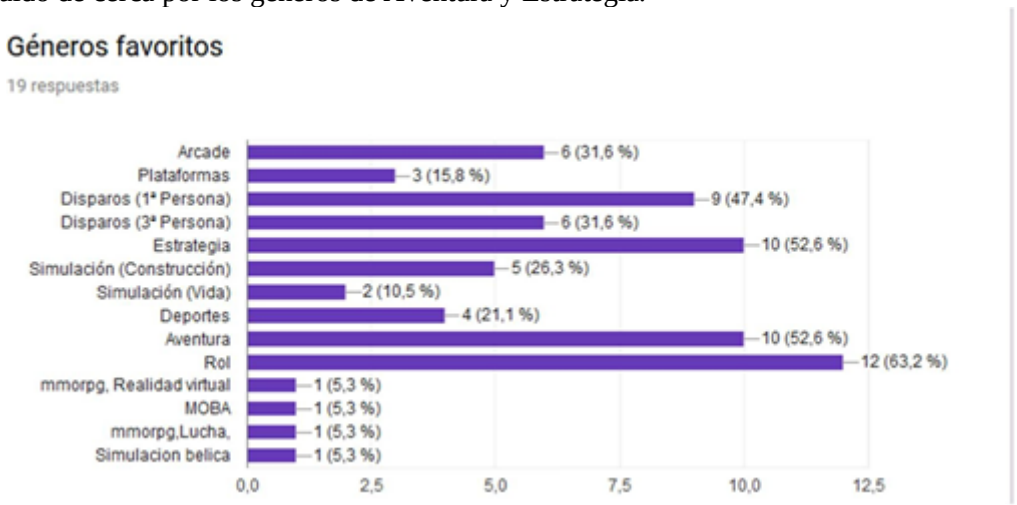


Figura 15:estadísticas referentes al primer experimento (Géneros favoritos)

A continuación, se puede comprobar que las temáticas preferidas por los encuestados son ciencia ficción y fantasía por un claro margen, se debe tener en cuenta que, en ambas preguntas, tanto géneros como temáticas favoritas son preguntas de respuesta múltiple puesto que no es habitual que un videojuego sea puramente de un único género o de una temática específica.

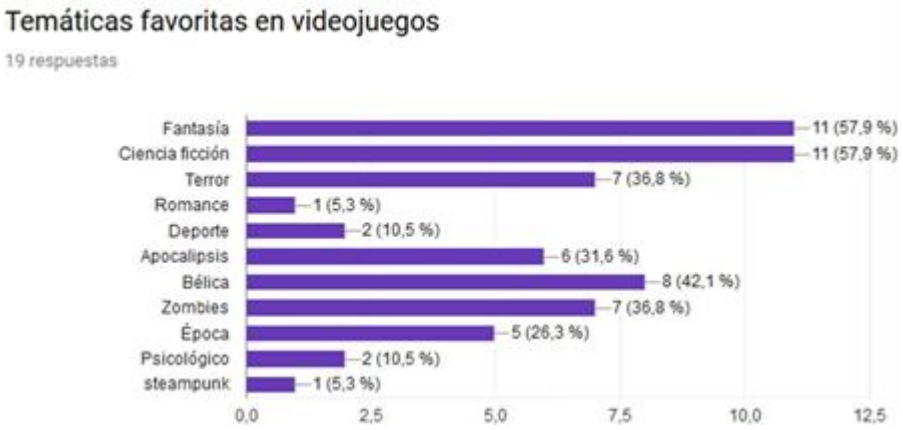


Figura 16: estadísticas referentes al primer experimento (Temáticas favoritas en videojuegos)

Los resultados fueron favorecedores, debido a que como se puede observar en la Figura 16, una gran cantidad de usuarios son partidarios de la temática de ComeBack.

Siguiendo con la encuesta, se preguntó a los usuarios acerca de si preferían juegos multijugadores o de un jugador, la respuesta de los usuarios en este caso fue una sorpresa para el equipo puesto que, en los últimos años, la industria del videojuego ha tendido al multijugador y existía la preocupación de que la falta de esta opción provocara el rechazo de muchos usuarios.

Sin embargo, y como se puede observar en la Figura 17, la cantidad de usuarios que prefieren los videojuegos multijugador son una minoría.

Figura 17: estadísticas referentes al primer experimento (Modelidad preferida)

También se preguntó a los encuestados acerca de los aspectos que consideraban más y menos importantes en un videojuego. A la pregunta de aspectos más importantes, las respuestas tuvieron una clara mayoría acerca del diseño del juego y su historia, dejando los diálogos relegados a la parte inferior de aspectos importantes.

De forma sorprendente, a la pregunta de aspectos menos importantes en un videojuego, mostrada en la Figura 18 el ganador claro fue la facilidad de aprendizaje, aunque se puede razonar que esto se debe a que, como se muestra en las preguntas anteriores, la mayoría de encuestados son jugadores con experiencia por lo que a lo mejor pueden adaptarse rápidamente a las mecánicas de un videojuego nuevo.

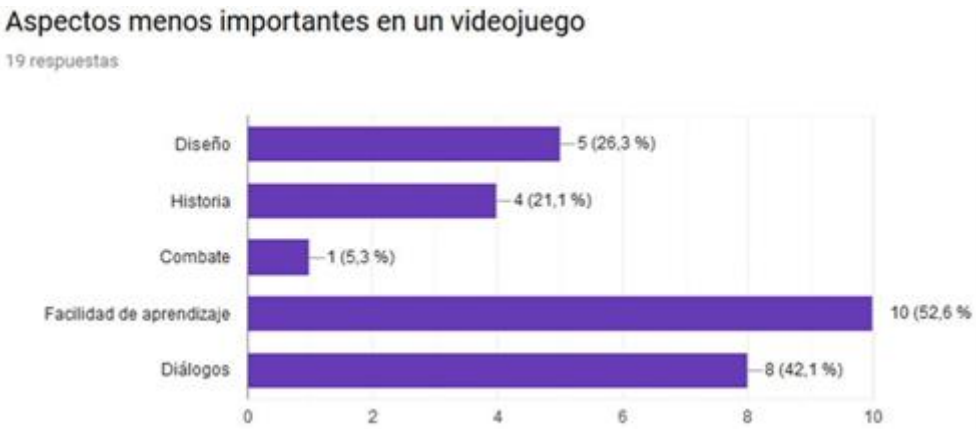


Figura 18: Estadísticas referentes al primer experimento (Aspectos menos importantes en un videojuego)

Tras ello, se iniciaron las preguntas específicas sobre el primer MVP de ComeBack.

La primera pregunta trata sobre el diseño del videojuego. Tras el análisis los resultados de las preguntas anteriores, donde el diseño del juego ha sido considerado importante por más de la mitad de los encuestados (63.2%), debe prestársele una gran cantidad de atención.

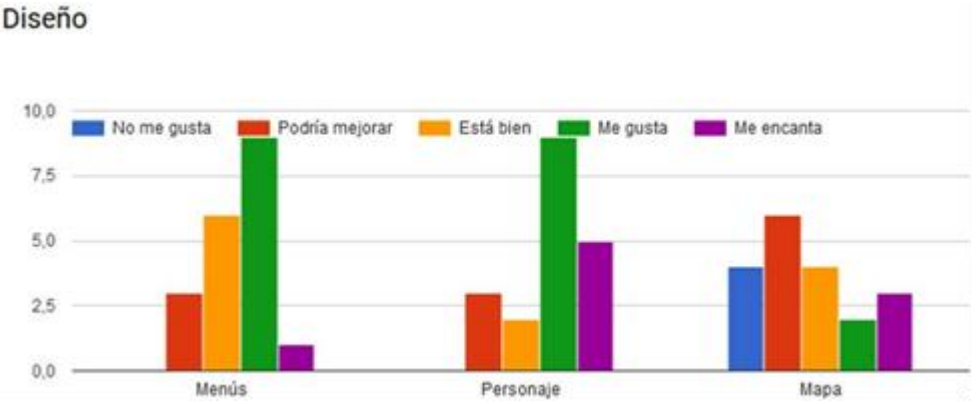


Figura 19: estadísticas referentes al primer experimento (Diseño de ComeBack)

En general, los resultados fueron bastante favorables. Tanto los menús como el personaje recibieron buenas críticas, en especial el último de estos. Por otro lado, el mapa necesita ser revisado para futuras versiones. Todo esto puede ser observado en los gráficos de la Figura 19.

Las siguientes preguntas acerca de la funcionalidad y otros aspectos diversos del juego fueron generalmente favorables, aunque, como toque de atención, hubo varios usuarios descontentos por el combate tal y como se muestra en la Figura 20. Viendo esta situación, resulta interesante la respuesta de los usuarios acerca de datos más específicos del combate.

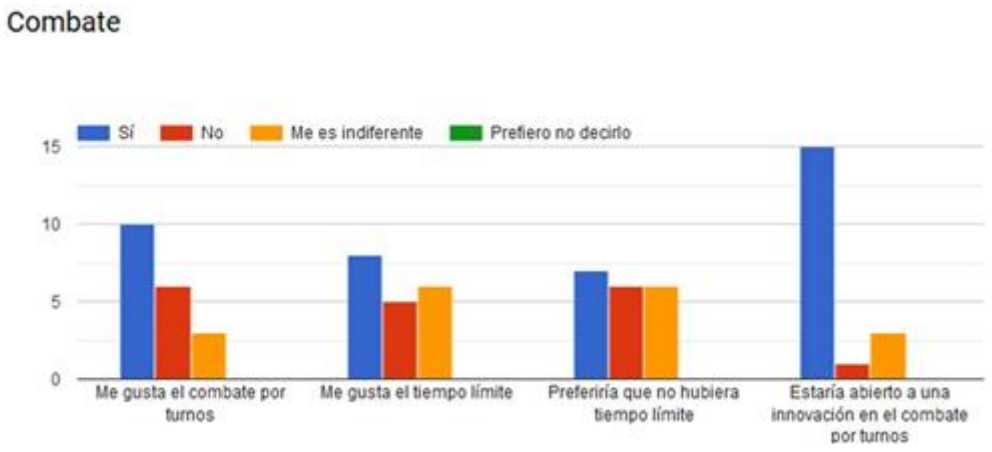


Figura 20: estadísticas referentes al primer experimento (Combate en ComeBack)

Como se puede observar en la imagen anterior, el problema del combate parece estar en el sistema de combate por turnos, o más específicamente en el sistema de la barra de tiempo para el turno. Es importante señalar que una gran mayoría de usuarios parecen interesados en un sistema de combate por turnos innovador.

Otro punto importante en la encuesta, especialmente en las primeras versiones, es el de la dificultad y aprendizaje preferido por los usuarios.

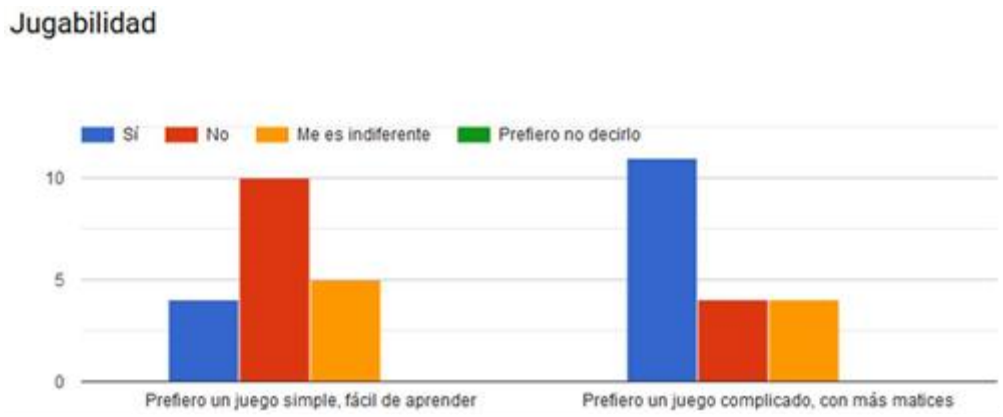


Figura 21: estadísticas referentes al primer experimento (Jugabilidad de ComeBack)

Como se puede observar en la Figura 21, la mayoría de los usuarios encuestados prefieren un videojuego que presente un desafío en lugar de uno casual.

Esta pregunta es bastante interesante puesto que la dificultad de un juego puede alejar a un tipo de usuarios y atraer a otro. De nuevo, es importante recalcar que la mayoría de los usuarios encuestados son usuarios experimentados.

También se realizaron preguntas abiertas sobre la opinión de los usuarios en diversos aspectos generales del juego. Estas preguntas permiten obtener datos más concretos sobre los puntos fuertes y débiles del videojuego.

En primer lugar, se hicieron preguntas sobre la mejor y la peor característica del juego en el momento de la entrevista.

En general, los usuarios comentaron que el diseño del juego y los personajes, además del sistema de combate, estaban muy logrados para una demo tan temprana. En cuanto a la peor característica actual, a los encuestados no les gustaron las animaciones actuales, así como la falta de control sobre la cámara y el diseño de los menús.

Luego se realizaron preguntas acerca del desarrollo futuro del juego. El sistema de habilidades descrito anteriormente y la temática que mezcla fantasía y ciencia ficción son lo que los encuestados han elegido sus partes favoritas. Por otro lado, los usuarios mostraron preocupación de que si el sistema de combate y habilidades no se implementaban con cuidado podrían resultar redundantes.

La última pregunta fue de respuesta abierta, dejando la posibilidad de que los usuarios dejaran comentarios adicionales. Algunos de los comentarios repetidos e interesantes fueron:

- Mejorar el control de la cámara
- Implementar alguna modificación innovadora al sistema de combate por turnos clásico para lograr un producto más interesante.

Segundo MVP

Aunque en la fecha de creación de esta memoria aún no se ha podido realizar el experimento relacionado con el segundo MVP, se ha querido introducir su definición en este documento debido al impacto que los resultados obtenidos en las pruebas del primero han tenido en este.

En primer lugar, tras observar las anotaciones hechas por los early adopters, se llegó a la conclusión de que los usuarios estaban descontentos con el funcionamiento de la cámara. Por ello, se incluyó en el segundo MVP el remodelado de esta.

También se añadieron las correcciones a bugs y fallos encontrados por los jugadores, como la caída en picado desde ciertas zonas del mapa o fallos de opciones en el menú principal.

Por último, se añadieron una serie de cambios que el equipo de desarrollo deseaba abordar con el fin de mejorar el producto, como cambios en la iluminación o en las fuentes de texto.

Requisitos

Teniendo en cuenta todo lo explicado, se marcaron además una serie de requisitos para la formación de este MVP que se enumeran a continuación:

- Diseño y enlazado de 3 niveles.
- Añadir uno de los compañeros del personaje principal.
- Niveles con decoración móvil.
- Modelado y recogida de armas.
- Combate contra 4 enemigos.
- Sistema de guardado de partida.
- Poblar niveles de enemigos.

Problemas enfrentados

Durante esta parte del proceso se encontraron una serie de problemas e impedimentos que entorpecieron el desarrollo más de lo esperado.

El más notable de ellos fue que dos de los miembros del equipo tuvieron problemas con sus respectivos ordenadores, viéndose obligados a adquirir nuevos componentes para poder seguir con el proyecto.

A esto se le suman los problemas encontrados durante la animación del compañero de la protagonista. De un guardado a otro del archivo se vio corrompido y no se halló forma posible de repararlo. Tras una semana de intentos fallidos, se optó por rehacer el esqueleto de cero y volver a hacer las animaciones.

Por culpa de este error se perdió el tiempo empleado en el primer esqueleto y en parte de las animaciones, el utilizado en tratar de solucionar el problema y finalmente el que se derivó a rehacer el esqueleto y las animaciones, empleando en total más de medio mes en un proceso que en el personaje anterior ocupó menos de la mitad.

CAMPAÑA DE MARKETING

A pesar de que el producto se encuentra en un periodo de crecimiento constante, se quiso desde un principio abordar el marketing, aunque fuera a pequeña escala.

Una de las principales motivaciones, fue el hecho de basar parte de la economía del proyecto en crowfunding. Es de vital importancia dedicar tiempo a la captación de un público al que vender tu idea para que se conviertan en potenciales inversores.

La primera toma de contacto en lo que a esto se refiere fue la quedada de colaboradores de la start.inf de la Universidad Politécnica de Valencia. Este espacio ofrece apoyo, asesoramiento y lugar de trabajo para proyectos de emprendimiento.

Así pues, se acudió a esta reunión con el fin de dar a conocer ComeBack a diferentes perfiles de emprendedores, publicitando de esta forma el producto y obteniendo a su vez retroalimentación sobre el estado del proyecto por parte de otros desarrolladores.

Se acudieron en total a dos eventos de la start.inf, el segundo consistió en una feria establecida en la Escuela Técnica Superior de Ingeniería Informática de la Universidad Politécnica de Valencia, donde se promocionó el juego a los estudiantes de la escuela, permitiéndoles probar una demo no finalizada del primer MVP y enseñándoles modelos del videojuego.

También sirvió como proceso de reafirmación de la historia del juego. Pues se comentó brevemente con gran cantidad de alumnos la temática del juego, los objetivos y sobre todo la historia en general.

No solo se obtuvieron respuestas muy positivas al respecto, sino que además muchos de los alumnos proporcionaron sus datos de contacto para que se le mantuvieran informados sobre el avance del juego.

Una vez finalizado el primer MVP, se procedió a la creación de un perfil de Facebook[31] de empresa para el proyecto.

A través de este se dan actualizaciones del trabajo más visual, relacionado con la parte estética del juego. Además de esto se creó una sencilla web y un perfil de Pinterest que se enlazaron a la cuenta de Facebook. Sin embargo, estos son más fijos, mientras que el primero es actualizado periódicamente, analizando sus estadísticas con el fin de ver que publicaciones son las más efectivas.

Hasta la fecha de entrega de esta memoria, se ha observado una tendencia de visitas más elevadas si el contenido de la publicación incluye un vídeo. Siendo la más popular entre ellas el vídeo del recorrido del segundo nivel del juego con un alcance total de 99 personas, lo cual puede observarse en la Figura 22.

[31] https://www.facebook.com/comebackdnagames/

Fecha de publicación	Publicación	Tipo	Segmentación	Alcance	Interacción
30/05/2019 0:26	La prime			24	3 5
28/05/2019 15:25	Here it is			29	15 5
26/05/2019 23:52	Crea ndo			31	11 4
16/05/2019 13:00	Bien dean			29	8 3
15/05/2019 12:57	Ups.. .			27	5 3
08/05/2019 2:51	#com eBac			44	6 7
04/05/2019 20:33	Para pode			31	9 4
03/05/2019 17:27	Vista gene			99	15 8
01/05/2019 23:22	Ojea ndo			50	9 4
30/04/2019 2:29	Tu única			42	8 8
27/04/2019 10:45	Bien venid			66	33 14
26/04/2019 22:53	Vels las			29	11 5
25/04/2019 20:24	W ~			42	19 8
24/04/2019 20:36	Miran a -			53	11 8
21/04/2019 21:29	Prime ros			32	8 5

Figura 22: Captura de estadísticas de alcance de las publicaciones en Facebook

En un futuro, cuando se termine el desarrollo del juego o se tenga un producto más completo, se planea ir a una convención de videojuegos con el fin de promocionar ComeBack. No obstante, todavía es demasiado pronto como para llevar a cabo ese paso en el estado actual del proyecto.

MÉTRICAS Y DATOS DE TRACCIÓN

En este apartado se tratan los parámetros elegidos para medir el desarrollo del proyecto desde el punto de vista orientado al negocio.

Se utilizarán datos como el número de descargas del producto y número de seguidores en Facebook, además de los precios elegidos para Patreon.

La página de Facebook del videojuego ha sido usada para mantener actualizados a los interesados acerca del progreso del juego. Al momento de la redacción de esta memoria, la página cuenta con cuarenta y siete seguidores, lo cual ha sido considerado por el equipo como un gran éxito y una muestra de que los usuarios parecen estar interesados.

La demo en el momento del primer experimento ha tenido quince descargas, gran parte de estos han realizado posteriormente la encuesta.

La monetización inicial se ha llevado a cabo mediante la plataforma de Patreon, en esta se tienen cinco posibilidades de pago cada una asociada a unos beneficios especiales mientras el juego está en desarrollo.

- 1,99 euros: Este nivel de Patreon da acceso a la última versión del proyecto, mientras que sin él el acceso es a la versión anterior a la más nueva.
- 3,99 euros: Este nivel otorga todos los beneficios que el anterior y además se le enviará al usuario una carta de agradecimiento de los desarrolladores y una imagen de salvapantallas de ComeBack.
- 5,99 euros: Este nivel da acceso a todos los beneficios de los niveles anteriores además de permitir que el usuario de nombre a un NPC (non-player character) del juego.
- 9,99 euros: Este nivel da acceso a todos los beneficios de los niveles anteriores además de un dibujo dedicado de la artista y acceso a un chat una vez al mes con los desarrolladores.
- 19,99 euros: En el máximo nivel posible de Patreon para este proyecto el usuario consigue todos los beneficios anteriores además de la posibilidad de personalizar un NPC del juego (limitado a los primeros en alcanzar este nivel, número aún por determinar) y una mención especial honorífica en los títulos de crédito.

Los dos últimos niveles se activarán más adelante, una vez la carga de trabajo se vea reducida.

CRONOLOGÍA DEL PROYECTO

A continuación, se hará un repaso acerca del desarrollo del proyecto desde sus inicios hasta el momento actual del proceso, explicando la situación tanto de los miembros del equipo como del proyecto en sí mismo.

Es posible dividir el progreso del proyecto en cinco secciones diferentes en función del trabajo que se estaba siendo realizado en cada momento.

El semestre base o de preparación tuvo lugar entre julio del 2017 y diciembre del 2017, en este semestre se tenían las nociones acerca del proyecto que se quería hacer, pero no había nada concreto. En estos seis meses equipo empezó a familiarizarse con las tecnologías que tenían que usarse, especialmente con Unity y Blender, mientras tanto, se iban puliendo datos e ideas para ir llegando a una definición más concreta del proyecto.

En el primer semestre, enero del 2018 a junio del 2018, se empezaron a realizar prototipos sobre ciertas partes del proyecto, ganando una visión más clara acerca del trabajo necesario para sacar el proyecto adelante, en este semestre se realizó una versión temprana del sistema de combate y los diseños y modelos iniciales.

El segundo semestre fue el comienzo del proyecto como tal, que duró entre julio de 2018 y diciembre de 2018. En este periodo de tiempo se comenzó con el proyecto conjunto, creando el repositorio de versiones en Bitbucket[32] para el equipo y estableciendo las tareas específicas a realizar.

Cuando la fecha se acercaba a diciembre, el proyecto se hallaba en el primer MPV, lo cual permitió lanzar el primer ejecutable del juego junto con la encuesta.

El tercer semestre, desde enero del 2019 hasta junio del 2019, fue una continuación del semestre anterior, continuando con el desarrollo del juego, implementando mejoras y añadiendo nuevas funcionalidades. A finales de marzo, el equipo se encontró con problemas con el repositorio de Bitbucket, los cuales serán discutidos con más detalle posteriormente en esta memoria, y que provocaron una migración del proyecto a GitHub[33].

Tras ello, al principio del cuarto semestre se vio como los otros dos integrantes abandonaron el proyecto, tras ello se decidió la creación del Asset Simple Inventory and Dialogue System para recaudar fondos hasta encontrar nuevos integrantes a este proyecto, pasando así a rediseñar el sistema de inventario por completo.

32 www.bitbucket.org
33 https://github.com/

CAPÍTULO 5 ASPECTOS TÉCNICOS

TECNOLOGÍAS UTILIZADAS

Posteriormente se describirán las herramientas que se han utilizado durante el desarrollo del proyecto. Explicando alternativas a las herramientas elegidas y las causas de porque se escogieron unas y no las otras.

Repositorio de versiones

En cuanto al repositorio de versiones, se han utilizado Github y Bitbucket, ambas son plataformas de desarrollo colaborativo en la nube que ayudan a desarrolladores a llevar un registro histórico, permitiendo a su vez almacenar el código y ofreciendo un sistema de control de cambios.

Tanto Github como Bitbucket utilizan el sistema de control de versiones Git, pero Bitbucket trabaja a su vez con Mercurial.

Debido a la utilización de Git por parte de ambas plataformas, ofrecía un amplio margen de posibilidades puesto que este permite la mantención de los proyectos actualizados en cualquier momento dado, la creación de ramas o la posibilidad de volver a versiones antiguas si se diese el caso de toparse con cualquier problema o defecto en el código.

A su vez, a parte de proporcionar las funcionalidades heredadas de Git, ambos ofrecen otras herramientas de las cuales se destacan: la herramienta que permite la creación de una Wiki del proyecto, una herramienta de revisión de código y un sistema de seguimiento de código.

Mediante la utilización de los sistemas de almacenaje de Git, permite una gran agilidad a la hora de compartir o llevar un seguimiento en las versiones entre todos los miembros del equipo, ya sean tareas pequeñas o versiones prototipo más grandes.

Simultáneamente, Git también posibilita un mejor retroceso de versiones si una nueva funcionalidad añadida produce un error, accediendo al repositorio para poder retornar a una versión antigua.

Se han incluido ambas plataformas ya que se han hecho uso de ambas durante el desarrollo de el proyecto, se comenzó con Bitbucket y tras una limitación de este se optó por la migración a Github.

Se decidió al principio la utilización de Bitbucket sobre Github ya que proporcionaba la posibilidad de crear proyectos privados de manera gratuita, permitiendo que los usuarios no autorizados no puedan acceder al código.

La necesidad de migrar a Github ocurrió ocho meses después del inicio del proyecto ya que el tamaño del proyecto era mucho más grande que lo permitido para una cuenta gratuita en Bitbucket y el sistema LFS (*Large File Storage*) se encontraba en estado de actualización para aumentar la cantidad de ficheros permitidos.

Github en ese momento permitía un mayor peso en los archivos almacenados y ya había cambiado su política de repositorios privados gratuitos. Fue por estas razones que se decidió la migración.

Entornos de desarrollo

Unity

La elección de un motor grafico es esencial a la hora de comenzar a desarrollar un videojuego. Entre todos lo motores gráficos que se encuentran actualmente en el mercado se eligió Unity.

Esta elección fue debido a que, entre todos sus competidores, Unity es el motor de videojuegos con una curva de aprendizaje mas reducida ya que el resto están más dirigidos al desarrollo profesional.

Otro motivo de peso en esta elección fue debido a que Unity es un software gratuito, a pesar de que posea una versión de pago que incluye mas funcionalidades, estas no eran de alta relevancia a corto plazo para este proyecto.

Finalmente, uno de los miembros del equipo ya tenía un mínimo de experiencia trabajando con este motor, facilitando también la curva de aprendizaje de este.

Desde su fecha de lanzamiento en 2005 Unity no ha dejado de evolucionar y hoy en dia es uno de los motores gráficos más utilizados, tanto por grandes empresas multinacionales como por pequeños estudios independientes.

Muchos videojuegos populares hoy en día han sido creados con Unity, entre ellos se pueden destacar juegos como “Hearthstone”[34] o “Kerbal Space Program”[35].

Figura 23: Interfaz grafica de Unity

En la figura 23 se puede observar la interfaz grafica de Unity, la cual destaca sobre sus competidores por ser la más sencilla e intuitiva. Contiene por defecto cuatro secciones, aunque es posible customizar la cantidad de ventanas y sus posiciones según las preferencias del usuario.

En la figura mencionada anteriormente también se puede observar las cuatro secciones básicas: las secciones laterales, la sección central y la sección inferior.

La parte central, por defecto cuenta con dos pestañas con el titulo “Escena” y “Juego” respectivamente.

En la primera se pueden manipular los objetos en una escena, ya sea situando los objetos en ella, moverlos, escalarlos o rotarlos.

34 https://playhearthstone.com/es-es/?gclsrc=aw.ds&
35 https://www.pokemongo.com/es-es/

La segunda pestaña consta de la visualización del videojuego, la cual se llama *game view*, esto permite al desarrollador ver el estado de la escena actual que esta modificando tras pulsar sobre el botón *play*, tras ello se podrá observar la escena como si hubiera lanzado el ejecutable.

Por otro lado, la ventana inferior, consta de dos pestañas que constan de la consola y la estructura de las carpetas del proyecto.

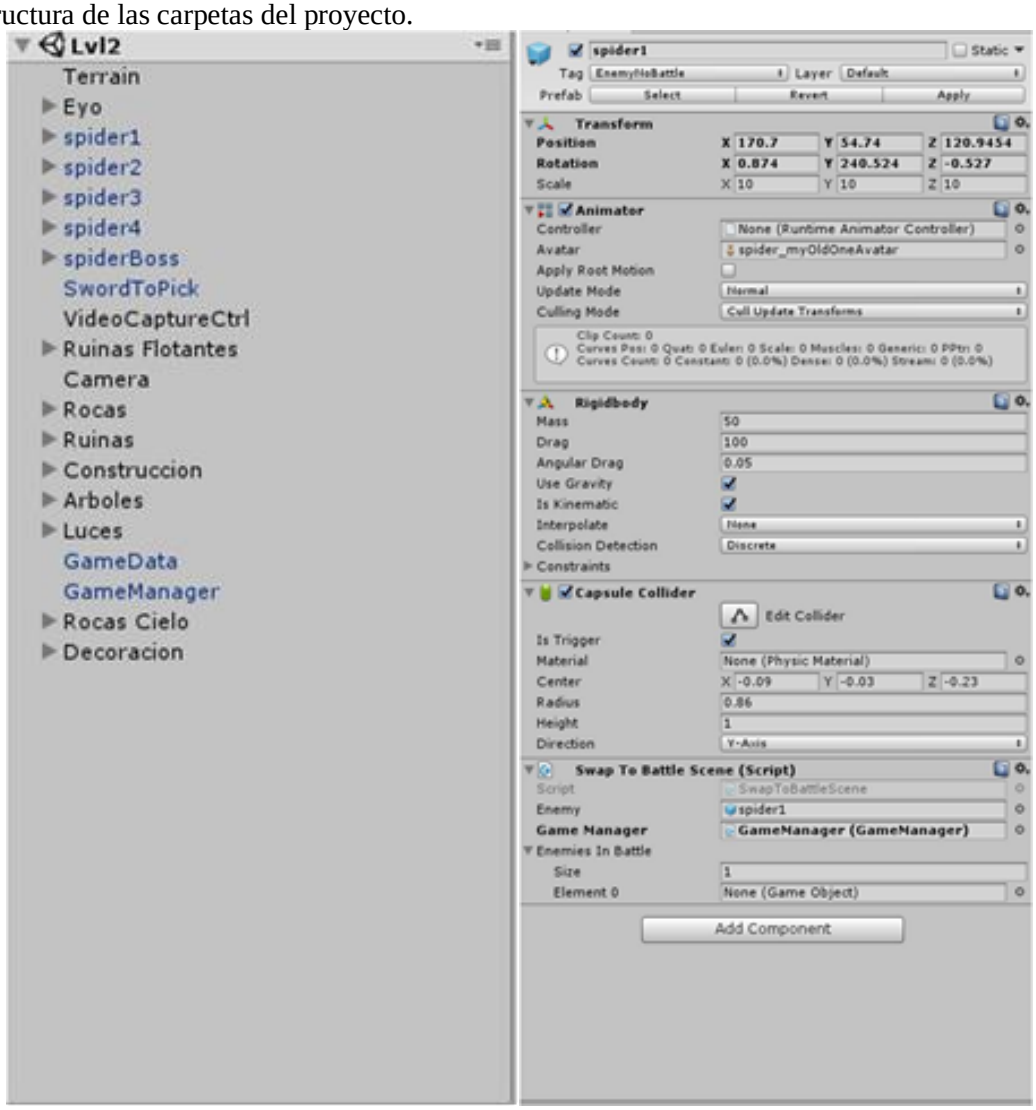


Figura 24: Sección izquierda y derecha de la interfaz de Unity

En la figura 24 se pueden observar las secciones laterales de la interfaz de usuario de Unity, la primera consta de los elementos de la escena actual como se ha mencionado anteriormente, mientras que la derecha consta del inspector, el cual muestra los detalles del objeto seleccionado en el momento, en este caso consta de spider1.

Microsoft Visual Studio[36]

Microsoft Visual Studio es un entorno de desarrollo compatible con un alto numero de lenguajes de programación tales como Python, C# o Visual Basic entre otros, junto con entornos de desarrollo web como Django.

Existen otros entornos de desarrollo que son compatibles con Unity, para la versión de Unity con la cual se comenzó el proyecto solo constaban de dos entornos de desarrollos integrados con Unity: Monodevelop y Visual Studio.

Se utilizó Visual Studio ya que se utilizó durante otras asignaturas a lo largo de la carrera para facilitar la curva de aprendizaje y también porque Monodevelop estaba en proceso de ser obsoleto para Unity.

Se ha utilizado Visual Studio para generar y ejecutar scripts en el lenguaje C# e importarlos al proyecto en Unity.

Herramienta de gestión de proyectos

Trello[37]

Trello es un software de administración de proyectos que emplea el sistema kanban para el registro de tareas con tarjetas visuales.

Estas tarjetas se pueden mover entre secciones diferentes correspondiendo al estado actual de dicha tarea. Para el proyecto se han creado cinco secciones que serán comentadas a continuación:

- Identificado: Esta sección comprende las tareas que se han identificado pero que no han sido iniciadas
- En Proceso: Se pueden observar las tareas que se encuentran en desarrollo en ese debido momento
- Pausado: En esta sección se pueden observar las tareas que, a pesar de haberse iniciado, por una razón u otra no se están trabajando en ese debido momento, este caso puede suceder, por ejemplo, si otra tarea bloquea a otra que aun no se ha podido terminar.
- Finalizado: En esta sección se encuentran todas las tareas que se han terminado de desarrollar.
- Cosas útiles: Seccion donde se pueden observar fichas con datos útiles para el equipo, ya sean enlaces o recordatorios. Estas fichas no constan de tareas, por lo tanto, son independientes a las otras secciones mencionadas anteriormente.

36 https://visualstudio.microsoft.com/es/
37 https://trello.com/

A continuacion se puede observar un ejemplo del sistema empleado y de la organización llevada como ejemplo, en este caso se muestra el estado del Trello del equipo durante la elaboración del segundo MVP.

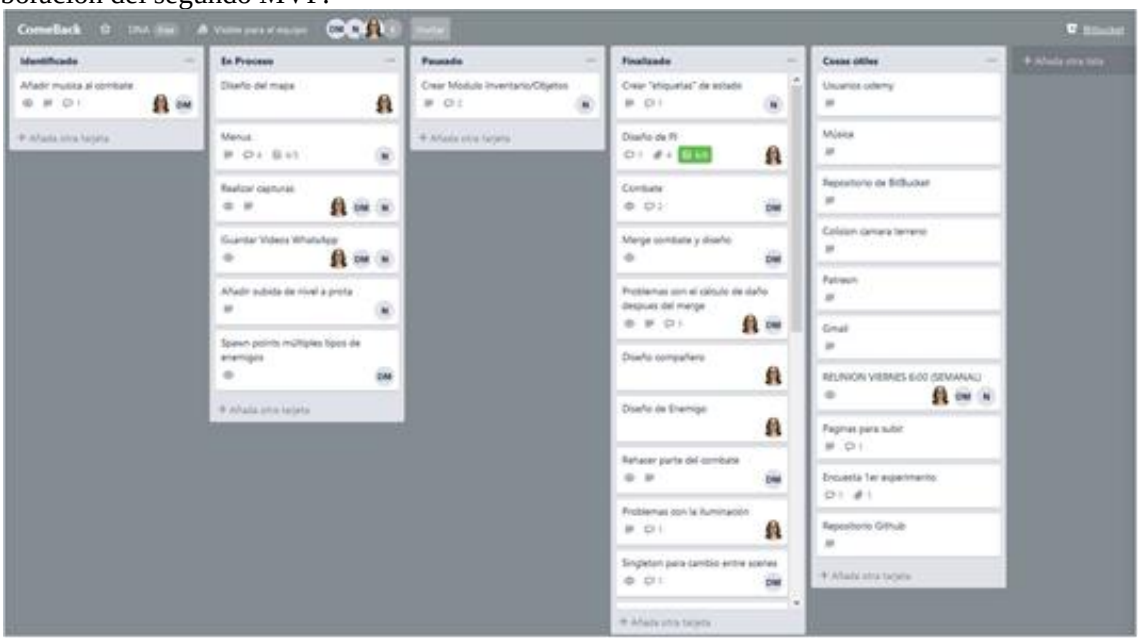


Figura 25: Captura de pantalla del estado del Trello del equipo de ComeBack durante la elaboración del segundo MVP

CAPITULO 6 SISTEMAS DE INVENTARIO Y DE DIÁLOGO

En ComeBack, como en cualquier otro videojuego RPG, requiere del uso constante de objetos que el personaje encuentra o se le dan tras acabar una misión, por lo que el estilo de juego necesita de un módulo que realice las tareas de almacenamiento y de interactuación con el usuario con dichos objetos.

Para ello se ha creado un sistema de inventario con interfaz gráfica para que el jugador pueda equipar, usar o simplemente almacenar dicho objeto dependiendo de las características de este.

A su vez, los RPG dotan de otros tipos de interfaces requeridas para el estilo de juego, tal como una interfaz del estado de los personajes, interfaces de dialogo u opciones del juego.

El sistema de diálogo es otro de los elementos importantes en un RPG ya que permite interactuar con personajes del juego para que ayuden durante misiones, poder sumergir al usuario en la historia y en su cuerpo de tradiciones, también conocido como *lore* en inglés. Este capítulo está dividido en tres puntos primero se explicará el desarrollo del Asset que contiene los sistemas de inventario y de diálogos, tras ello, se explicará como se ha realizado la validación mediante el uso de dicho Asset en ComeBack, separado en dos partes: el sistema de inventario y el sistema de diálogos.

ASSET SISTEMAS DE INVENTARIO Y DIALOGOS

Al crear los sistemas de inventario y de diálogos se observó la falta de un amplio abanico de dichos Assets personalizables en la Unity Store. Para ello se procedió a la creación de un Asset que contuviera ambos módulos, totalmente personalizables, en un mismo Asset no solo para la creación de dichos módulos para ComeBack, sino también para poder obtener algo de financiación a lo largo de la creación del producto.

Para ello se necesitaba pensar en que tipo de Inventario se quería implementar para ComeBack. En el caso de este videojuego se eligió el inventario más característico de los JRPGs llamado regla de 99. El porqué de esta elección se explicará más adelante tras enumerar los diferentes tipos de sistema de inventarios característicos en los videojuegos RPGs.

Para poder observar todo el trabajo puesto en este proyecto se enumerarán los diferentes tipos de inventario, una comparación del *Asset* creado contra la competencia potencial, la estructura del proyecto en Unity para poder entender los diferentes tipos de objetos predefinidos en Unity para la interfaz de usuario y se terminará por explicar la implementación y las funcionalidades del *Assset* mencionado anteriormente.

Tipos de Sistema de Inventarios

En el mundo de los videojuegos existen una enorme cantidad de sistemas de inventarios, es por ello por lo que para esta comparación se van a explicar únicamente los más relevantes.

Los diferentes tipos de sistemas de inventario que se van a explicar a continuación son:

1. Regla de 99
2. Ponderado
3. Cuadricula visual

Regla de 99

Uno de los primeros sistemas de inventario verdaderos implementados ampliamente por los desarrolladores de juegos de rol, los inventarios basados en la "Regla del 99" se asocian más comúnmente con JRPG clásicos como Final Fantasy VI, Chrono Trigger y las primeras entradas en la serie Pokemon. El paradigma de diseño detrás de este sistema es extremadamente simple: los jugadores pueden esconder una cantidad innumerable de artículos en su inventario, pero solo un número fijo de cada uno. Aunque ese número fijo es típicamente 99, puede ser fácilmente 50, 100 o 1,000. Además, los inventarios de la Regla de 99 son generalmente universales, lo que significa que se comparten entre todos los miembros del grupo.

A continuación, se puede observar un ejemplo de este inventario en el videojuego Final Fantasy VI.

Figura 26: Ejemplo de Inventario Regla de 99 en Final Fantasy VI

Pros:

- Riesgo de diseño bajo: este estilo de sistema de inventario elimina efectivamente la necesidad de microgestión. Mientras los jugadores se encuentren o puedan pagar un artículo, almacenarlo rara vez planteará un problema.
- Fácilmente escalable: equilibrar un sistema de inventario "Regla de 99" no es un proceso laborioso. Caso en cuestión: al cambiar el número 99 a 200 en el código, habrás duplicado efectivamente el tamaño del inventario de tu juego.
- Fácil de implementar: en la misma línea, los inventarios basados en cantidades estáticas son extremadamente fáciles de implementar y no deberían consumir un valioso tiempo de desarrollo.

Contras:

- Realismo: Los inventarios “Regla de 99” no son ni intentan ser realistas.
- Valor del objeto: la única limitación de diseño real de la "Regla del 99" es que los objetos que se encuentren a menudo no deberían ser demasiado poderosos en relación con la fuerza del enemigo, no cuando puedes almacenar tantos objetos. Una solución es garantizar que los artículos poderosos tengan un precio acorde en la tienda del videojuego o se puedan encontrar muy raramente.
- No visual: A menudo los inventarios “Regla de 99” parecen anticuados a causa de su falta de efectos visuales

Ponderado

Los sistemas de inventario ponderado han ganado un gran grado de popularidad en la última década, en gran parte debido al crecimiento del juego de rol occidental. Juegos como Fallout 3 y Oblivion han hecho un uso ejemplar del sistema, aprovechándolo para agregar una capa adicional de profundidad.

En este sistema, a cada elemento o pieza de equipo se le asigna un valor numérico que representa su peso. A los artículos que se perciben como más pesados se les asigna un valor más alto que los que se perciben como más ligeros. Por ejemplo, un lanzacohetes puede pesar 20, mientras que un botiquín de primeros auxilios puede llegar a pesar solo 1.

Los personajes solo pueden llevar una cierta cantidad de peso antes de experimentar los efectos nocivos de la carga o la fatiga. Esto, en última instancia, puede hacer que caminen más lentamente o pierdan gradualmente la salud. Alternativamente, el juego puede impedir que el personaje supere el límite de peso asignado por completo.

A continuación, se puede observar el uso de este inventario en el videojuego Fallout 3.

Figura 27: Ejemplo de uso del Inventario Ponderado en el videojuego Fallout 3

Se pueden observar dos puntos clave los cuales muestran que este inventario consta de un inventario ponderado.

1. Justo debajo de la palabra ítems se puede observar una variable *Wg* que corresponde a un acrónimo de la palabra *Weight* en inglés, correspondiendo al peso total de todos los elementos que contiene el personaje en su inventario.
2. A su vez al lado de la representación visual del objeto seleccionado se puede observar la variable mencionada anteriormente, correspondiendo al peso del objeto en cuestión, en este caso la chapa de la botella no contiene peso alguno.

Pros:

- Gestión de recursos: la necesidad de gestionar los recursos de manera efectiva se hace evidente, sin necesariamente convertirse en una gran carga.
- Realismo: Los sistemas ponderados agregan un grado de realismo que no está presente en el sistema "Regla del 99".
- Implementación: Los sistemas ponderados son casi tan fáciles de desarrollar como los sistemas "Regla de 99". Sin embargo, requieren un poco más de creatividad de diseño.

Contras:

- Equilibrio: se requiere una gran cantidad de pruebas de juego para garantizar que los jugadores puedan llevar la cantidad "correcta" de inventario. Permíteles cargar demasiado y el juego pierde profundidad; demasiado poco y sufrirán constantemente de fatiga.
- Tirar elementos: examinar los elementos e intentar encontrarlos para tirar puede ser engorroso y frustrante, más aún si el sistema no está bien equilibrado.
- Valor del objeto: un artículo con un peso de 2 (granada de mano) podría ser mucho más valioso que uno con un peso de 20 (poste de acero). Por lo tanto, los desarrolladores están restringidos en cuántos elementos de bajo nivel pueden presumir de estadísticas poderosas.
- No visual: los sistemas ponderados suelen estar basados en texto y hacen poco para mejorar el atractivo visual del juego.

Cuadricula Visual

Popularizado por los juegos de rol de acción como Diablo 2 y los juegos de terror de supervivencia de Resident Evil, el sistema de inventario de cuadrícula visual es solo eso: los jugadores tienen un número específico de ranuras en las que alojar objetos, representados visualmente por una cuadrícula rectangular. Las cuadrículas visuales también son populares en los MMO modernos como World of Warcraft y Star Wars: The Old Republic.

En lugar de que se le asigne un valor de peso, el equipo y otros artículos de inventario se dimensionan en consecuencia. Por ejemplo, una espada ancha podría tener tres ranuras de cuadrícula de alto y una de ancho, mientras que un hacha de batalla corta sería dos de alto y dos de ancho. Otras composiciones de elementos no rectangulares han llevado a los fanáticos de los juegos a comparar el sistema con un juego de Tetris.

A continuación, se puede observar en la figura 28 una representación este estilo de sistema de inventario.

Figura 28: Ejemplo de uso del sistema de inventario de cuadricula visual en el videojuego Resident Evil 4

Pros:

- Visual: debido a su naturaleza visual, la gestión de recursos es un proceso relativamente intuitivo. Los elementos visuales también son más atractivos para los sentidos que los inventarios ponderados o "Regla de 99".
- Drag and Drop: Debido a que el inventario se representa gráficamente, los jugadores pueden mover elementos de un lugar a otro con facilidad. Identificar artículos es menos complicado de lo que es con la "Regla de 99" o inventarios ponderados.
- Realismo: las cuadriculas visuales son pseudo-realistas. El tamaño es un factor determinante en la cantidad de peso que uno puede transportar.

Contras:

- Colocación: organizar los objetos para que se pueda liberar uno o dos espacios del inventario puede ser una tarea difícil
- Equilibrar el tamaño: las cuadrículas más pequeñas requieren que los jugadores descarten objetos o hagan más viajes a la tienda. Por otro lado, cuanto más grande es la cuadrícula, menos realistas y más difíciles de explorar se vuelven.

	Regla de 99	Ponderado	Cuadricula	Simple Inventory and Dialogue System
Visual	NO	NO	SI	SI
Realista	NO	NO (solo un grado más que Regla de 99)	SI	NO
Fácil de implementar	SI	SI	NO	SI
Fácilmente escalable	SI	SI	NO	SI
Fácil de equilibrar	SI	SI	NO	SI

Figura 29: : Tabla comparativa de los diferentes sistemas de inventario comparado contra Simple Inventory and Dialogue System

Tras haber analizado con detenimiento los estilos de inventario más populares, se decidió el uso del estilo de Regla de 99 gracias a que es fácil de implementar y altamente escalable. A su vez, añadir elementos visuales, que es uno de los puntos negativos de los inventarios Regla de 99 no sería algo complicado de realizar.

Comparación con otros Assets similares en la Unity Store

La Unity Store no contiene un amplio abanico de Assets que contengan sistemas de inventario o de diálogos personalizables y no contiene ningún Asset que contenga ambos sistemas en uno.

Dado este vacío en el mercado de la Unity Store se quería crear dicho Asset, que a pesar de no contener un sistema de inventario tan potente como el resto, a su vez contiene un sistema de diálogos.

A continuación, se va a realizar un estudio de mercado breve, dando una breve descripción de la competencia y explicando los puntos fuertes de este Asset y sus puntos más flojos.

Los Assets que se han utilizado para este estudio han sido:

1. DaD Inventory
2. Inventory Engine
3. Simple Dialogue

Se han utilizado estos Assets debido a que son los Assets configurables más extendidos con mayor número de descargas que no requieran utilizar otro Asset para su funcionamiento.

DaD Inventory

Este Asset contiene un sistema de inventario simple que utiliza la funcionalidad drag and drop para equipar objetos. A su vez, contiene una tienda para vender objetos. Pero debido a utilizar la funcionalidad mencionada anteriormente, si se quiere modificar el equipo completo del personaje, se requiere bastante tiempo para dicha modificación.

Figura 30: Captura de pantalla de la escena de demonstracion del Asset DaD Inventory

A su vez, no contiene una base para diferentes tipos de objetos, sino que contiene únicamente una clase de objetos simple y cada objeto diferente debe contener una clase para cada objeto que se debe crear para el videojuego. A su vez, todas las estadísticas que se pueden observar son simplemente elementos de la interfaz de usuario no vinculadas a un script de personaje.

Inventory Engine

Este inventario altamente personalizable contiene un ejemplo de un sistema de inventario minimalista creado para un videojuego RPG rogue-like. Como pasaba antes con el DaD Inventory, para crear un objeto también se necesita crear una clase para cada objeto que se quiere utilizar ya que no consta de una jerarquía de objetos por defecto.

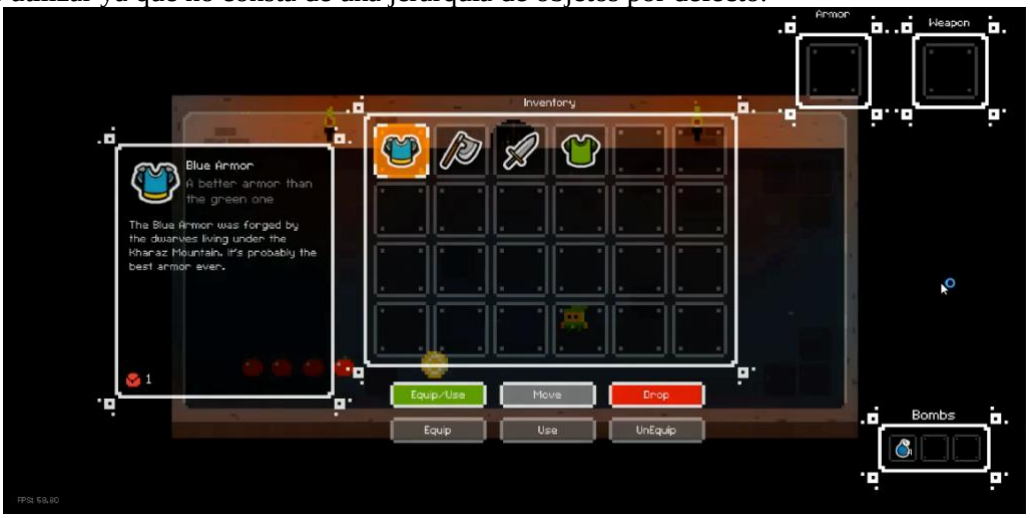


Figura 31: Captura de pantalla del Asset Inventory Engine

Una de las desventajas de este inventario es, para el uso o equipamiento de un objeto, el usuario debe pulsar de nuevo sobre otro botón diferente tras haber resaltado el objeto en cuestión que hacer uso.

Simple Dialogue

Este Asset no contiene un sistema de inventario, este contiene un sistema de diálogos simple entre dos personajes reutilizable y altamente personalizable. Este sistema de diálogos es muy similar al creado para este Asset ya que contiene imágenes para los diferentes personajes y el titulo se modifica por cada personaje que está hablando en cada momento.

Figura 32: Captura de pantalla del Asset Simple Dialogue

Simple Inventory and Dialogue System

A continuación, se va a realizar una breve descripción del Asset que se ha creado ya que será explicado con más adelante cuando se explique la fase de desarrollo de este.

Este Asset, llamado Simple Inventory and Dialogue System, contiene dos partes importantes:

- Sistema de inventario simple, altamente personalizable
- Sistema de diálogos simple personalizable

Este inventario, al contrario que el resto también contiene un script de jugador de ejemplo para que el usuario que quiera descargarlo tenga una idea de que elementos debe añadir a su propio script del jugador creado para su videojuego. A su vez, contiene una jerarquía de clases de objetos que sirve de base para poder crear todo tipo de objetos para cualquier RPG.

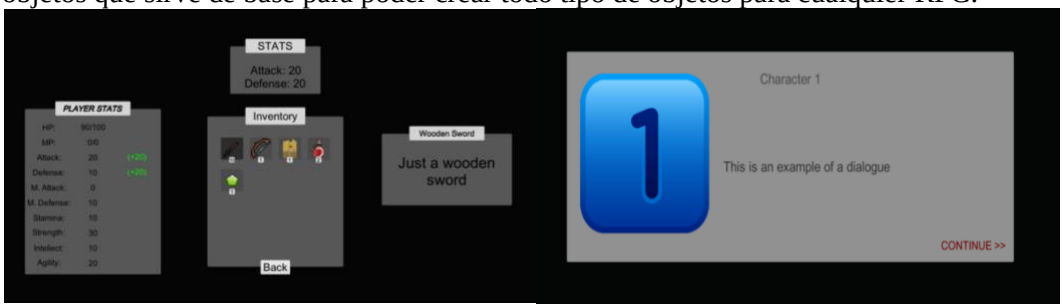


Figura 33: Captura de pantalla del Asset Simple Inventory and Dialogue System

Una de las ventajas que contiene el inventario sobre el resto es que tan solo requiere hacer click sobre el objeto que se quiere utilizar o equipar y no es necesario hacer *drag and drop* o pulsar sobre otro botón para utilizar o equipar dicho objeto.

	DaD Inventory	Inventory Engine	Simple Dialogue	Simple Inventory and Dialogue System
Inventario personalizable	Si	Si	No	Si
Jerarquía de objetos	No	No	No	Si
Script Jugador	No	No	No	Si
Tienda	Si	No	No	No
Drag and Drop	Si	No	No	No
1 click use	No	No	No	Si
Diálogos	No	No	Si	Si

Figura 34: Tabla comparativa de caracteristicas de productos similares a Simple Inventory and Dialogue System

Como se puede observar en la tabla comparativa, de los *Assets* comparados, el *Asset Simple Inventory and Dialogue System* es el más completo de la competencia. Esto es debido a que proporciona tanto un inventario personalizable, una jerarquía de objetos y un *Script* de jugador además de poder utilizar cualquier objeto directamente pulsando directamente sobre este. A su vez, contiene un sistema de diálogos. Se prevé que para la segunda versión del *Asset* también contenga un Sistema de compra y venta, solucionando así también la falta de una tienda en el *Asset*.

Tras haber realizado este análisis se llegó a la conclusión de que el Asset rellenaría un vacío en la Asset Store de Unity, a continuación, se explicará el diseño del Asset mencionado anteriormente.

Estructura del proyecto en Unity

El elemento más básico utilizado en Unity se llama *GameObject*. El *GameObject* representa el elemento básico de Unity y representa donde se encuentra cada elemento en la escena del videojuego, el *GameObject* puede representar: un menú, un personaje, el suelo, un objeto, etc.

A su vez, los *GameObject* están formados por componentes que componen las diferentes partes de este.

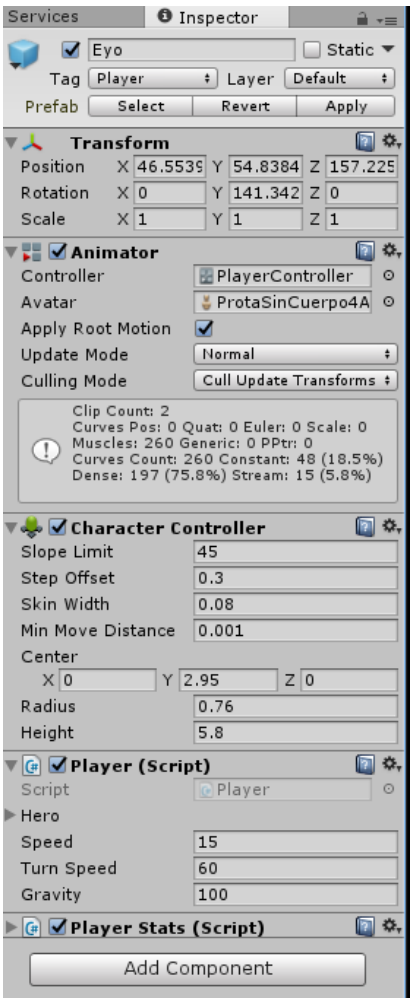


Figura 35: Captura de pantalla de los componentes del GameObject "Eyo"

En la Figura 35 se pueden observar los componentes del *GameObject* "*Eyo*", que es el nombre el personaje principal del videojuego, en la escena del mapa.

Se puede observar que el *GameObject* está formado por varios componentes, en este caso los componentes son:

- un *Transform*, el cual comprenden la mayoría de los *GameObject* independientemente de su función ya que indica la escala y la posición del *GameObject* en cuestión
- un *Animator*, el cual controla la secuencia de animaciones que deben de ocurrir durante el juego para este *GameObject* en específico
- un *Character Controller* que se encarga del movimiento del personaje (sin uso de físicas)
- un *Script* llamado *Player* y otro llamado *PlayerStats*

Además de esto, es importante recordar que debajo del nombre existe una característica llamada *Tag* la cual puede resultar muy interesante para encontrar elementos específicos del juego y por ello todos los *Tag* deben de ser únicos.

Otro concepto muy importante en Unity es la *Escena*, en la cual se ubican todos los *GameObject* que forman parte de esta.

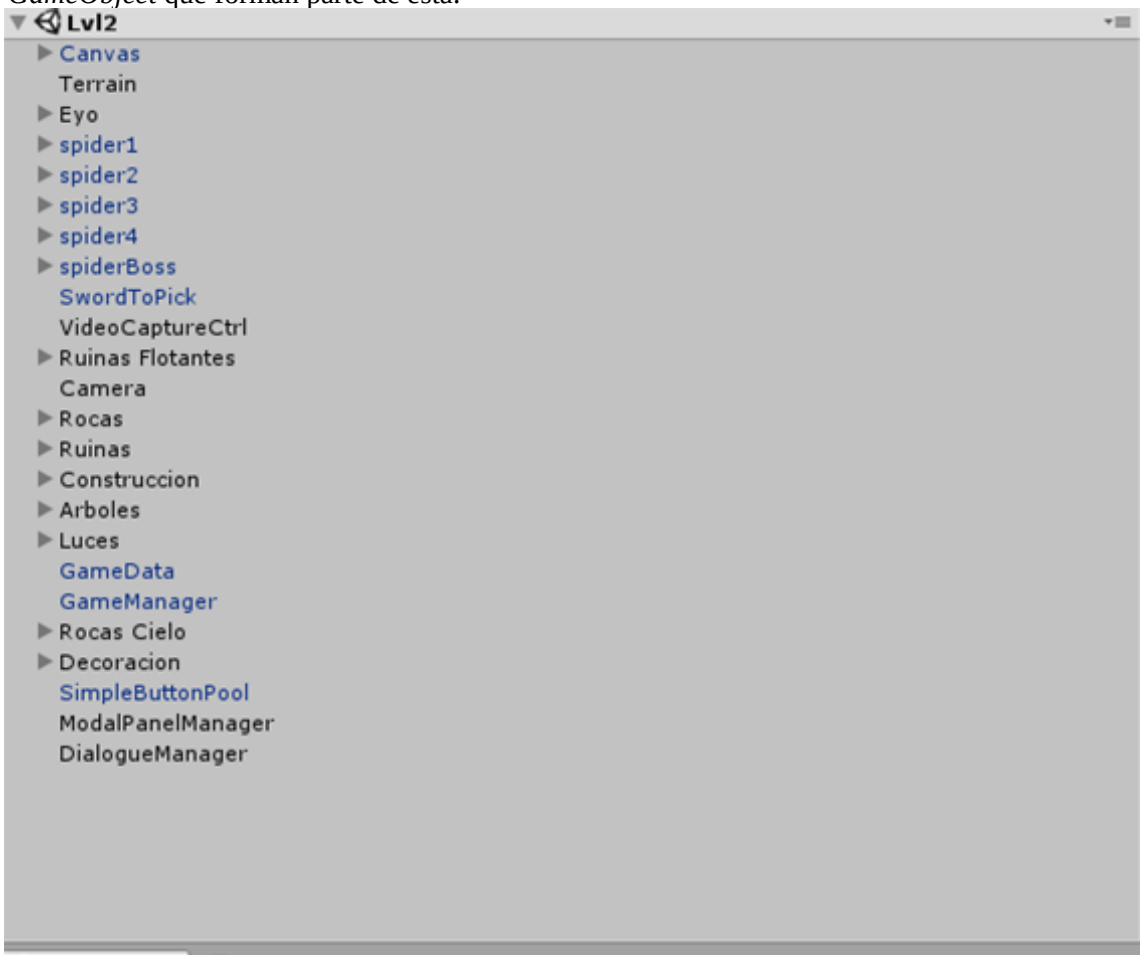


Figura 36: Captura de pantalla de los GameObject que componen la escena "Lvl2"

En la Figura 36 se muestra la jerarquía de la escena llamada “Lvl2”, esta es la escena por la cual el personaje se puede desplazar con libertad. Es importante notar diversos GameObject de interés tales como:

- “Eyo” que es el personaje jugador
- “spider” que son los enemigos que se encuentran en esta escena
- GameObject Canvas, el cual contiene todos los elementos de la interfaz de usuario
- GameData, el cual recoge todos los datos para poder guardar o cargar la partida
- SimpleButtonPool, el cual consta de una Pool de objetos para generar botones para la interfaz de usuario del inventario

Algunos de estos GameObject serán explicados en profundidad mas adelante.

Elementos de Interfaz de Usuario de Unity

Para poder entender cómo ha sido creado el Asset *Simple Inventory and Dialogue System*, es requerido en primera instancia comprender los elementos predeterminados de Unity utilizados para crear la interfaz de usuario.

El Canvas consta de la base de toda interfaz de usuario de Unity, todos los objetos tanto los paneles, botones, texto, etcétera, han de ser elementos hijos del Canvas. Este representa un espacio abstracto donde la UI es puesto y renderizado. A la hora de crear un elemento de la UI, si no hay un elemento Canvas en la escena, Unity creará un Canvas por defecto.

Uno de los elementos más utilizados dentro del Canvas es el Panel, el cual, por cada uno creado permite la creación de diferentes interfaces. Para explicar más a fondo las interfaces de usuario en Unity, se van a explicar utilizando como referencia el Menú Principal del juego también creado para el primer MVP. Este menú ha sido usado para aprender el funcionamiento de los diferentes tipos de objetos de UI que contiene Unity.

Para poder explicar dichos elementos de la UI de Unity, he utilizado voy a explicarles con detenimiento la implementación del menú principal de ComeBack, también desarrollado por mi para el primer MVP de ComeBack.

Menú Principal

El menú principal de ComeBack solo consta de dos menús diferentes, el *Main Menu* y el *Menú Opciones* ya que no se trata de una interfaz de usuario con características muy diferentes y no requiere de personalización avanzada se han creado dos *Canvas* como se puede observar en la figura 37.

Figura 37: Captura de pantalla de los GameObject que componen la escena "MainMenu"

Como se puede observar uno de los Canvas está inhabilitado, esto se trata a que, si estuviera habilitado, ambos menús aparecerían superpuestos el uno al otro y no crearía el efecto deseado.

Se pueden observar cuatro botones diferentes en la pantalla del menú principal, el *ButtonNewGame*, *ButtonLoadGame*, *ButtonOptions* y *ButtonQuit* a continuación se puede observar la representación de dichos botones en la escena del menú principal del juego:

Figura 38: Captura de pantalla del menú principal

Un botón en Unity realiza un evento al que es llamado solo si el usuario hace click en el mientras mantiene el ratón sobre el mismo. A continuación, se les explicará las funciones que realizan cada uno de los botones salvo el *ButtonLoadGame* que vendrá explicado más adelante en el punto de persistencia del juego.

ButtonNewGame

El *ButtonNewGame* realiza la tarea de empezar una nueva partida, esto requiere de una función que abra una nueva escena. Esto requiere la llamada a la función *LoadScene* de la clase *SceneManager* que se encarga de manejar las escenas, en concreto la función *LoadScene* carga la escena deseada, la llamada a esta función requiere el numero de la escena a la cual se quiere navegar, ya que la siguiente escena consta del dialogo inicial del juego simplemente se necesita hacer llamada a la siguiente escena, en este caso, se realiza una llamada a la función *GetActiveScene* que recoge en qué escena se encuentra actualmente, se recoge el atributo *buildIndex* de dicha escena y se añade un entero para realizar la llamada a la siguiente escena como se puede mostrar en la figura 39.

```
public void PlayButtonPressed()
{
    SceneManager.LoadScene(SceneManager.GetActiveScene().buildIndex + 1);
}
```

Figura 39: Captura de pantalla del método "PlayButtonPressed"

ButtonOptions

El *ButtonOptions* es probablemente el botón más simple de todo el menú principal del juego ya que se encarga simplemente de habilitar el menú de opciones e inhabilitar el menú principal. Para inhabilitar elementos de la interfaz de usuario de Unity no se necesita de ninguna función ya que existe una predeterminada por Unity por defecto para ello, por lo que para asignar dicho evento a la llamada *OnClick* del botón *ButtonOptions* simplemente se ha de asignar a través de la interfaz gráfica de *Unity* como se puede observar en la figura 40.

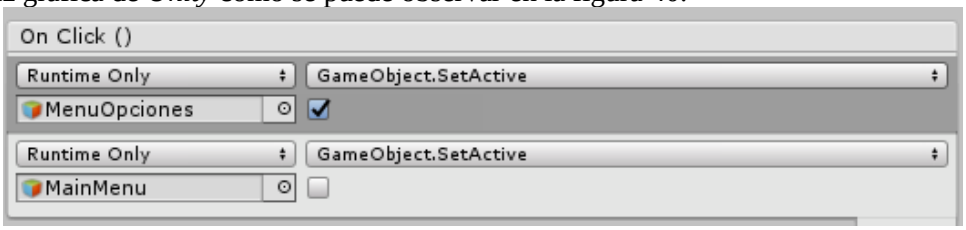


Figura 40: Captura de pantalla de la funcionalidad del botón "ButtonOptions"

ButtonQuit

El *ButtonQuit* es un botón muy sencillo que a primera vista parece innecesario ya que su funcionalidad es simplemente de cerrar la aplicación del videojuego. Para ello se debe hacer una llamada al método *Quit*, de la clase *Application*. La clase *Application* hace acceso a los datos en tiempo de ejecución de la aplicación. Por lo que es la única clase que se puede usar para hacer la llamada al cierre del viedeojuego.

```
public void QuitGamePressed()
{
    Application.Quit();
}
```

Figura 41: Captura de pantalla del método "QuitGamePressed"

Menú Opciones

Tras hacer click en el botón *ButtonOptions* se deshabilita el menú principal y se habilita el menú opciones.

Figura 42: Captura de pantalla del menú Opciones

En este menú se pueden ver tres elementos de interfaz de usuario customizados. Se trata de un Slider al interior de un TextBox, y dos dropdowns con TextBox al interior, esto es un método para alinear correctamente el texto a su elemento de la interfaz de usuario, así, el título del dropdown de resolución estará siempre alineado con el texto resolución. Un dropdown tiene cierta similitud con las acciones de un botón, solo que en vez de realizar una acción tras hacer click, muestra una lista de opciones de las que se pueden escoger una. Al hacer click en una de las otras opciones ésta lista se cierra, y el dropdown muestra la nueva opción seleccionada, si se hace click en el control la lista también se cierra, pero la opción no se ve cambiada.

Figura 43: Captura del Dropdown "Resolución"

Como los botones, las opciones de los *Dropdown* también pueden ser modificados por la interfaz gráfica de *Unity* o a través de código, y, como con los botones del menú principal, constan de ambas opciones en el menú opciones, por lo que se van a explicar las diferencias del *DropdownCalidad* y el *DropdownResolution*.

DropdownCalidad

El *DropdownCalidad* tiene la función de cambiar la calidad de los elementos renderizados. La lista de opciones del dropdown es populada a través de la interfaz gráfica de *Unity*, pero tras seleccionar uno de los elementos se hace una llamada a la clase *QualitySettings* y se pueden modificar a través de la función *SetQualityLevel*, la cual toma los valores de una lista customizada, creada a través de la interfaz gráfica de *Unity* en este caso este videojuego consta de tres opciones de calidad diferentes, como se muestra en la lista de opciones en la figura 44.

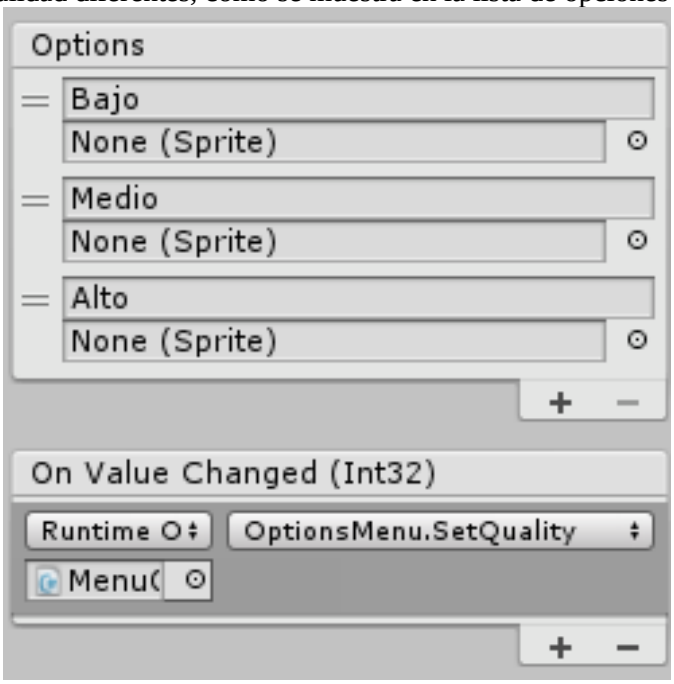


Figura 44: Captura de pantalla de las opciones del dropdown "DropdownCalidad"

DropdownResolution

El *DropdownResolution* es un poco más complejo, se debe a que debe recoger todas las resoluciones posibles del sistema del usuario para que este pueda decidir cuál es su resolución deseada.

Para ello se utiliza una clase llamada *Screen* que accede a la información de la pantalla. Para poder popular la lista de opciones de este dropdown, al habilitar el menú opciones, ya que no se quiere recoger todas las posibles opciones de resolución del usuario cada vez que se habilita el menú opciones para ahorrar en recursos, es necesario que la llamada a la clase Screen se haga sólo la primera vez que se habilita el menú opciones.

Para ello, se hace uso del método *Start*, común a todos los *Monobehaviour*, clase de la que heredan todos los scripts de la interfaz de usuario de *Unity*, por defecto, este método se ejecuta sólo la primera vez que se habilita el *GameObject* al cual está asignado este script.

Al interior del script, se hace la llamada al atributo *resolutions* de la clase *Screen* el cual devuelve un array de resoluciones.

Debido a que el dropdown no sabe identificar los valores de la resolución por sí mismo y, a su vez, solo se le puede pasar una lista como valor para popular sus opciones, se han tenido que realizar los siguientes pasos en el método *Start*:

1. Recoger los datos de la clase *Screen*
2. Crear una lista de Strings
3. Añadir una entrada a la lista por cada resolución, recogiendo la anchura y la altura
4. Identificar la resolución actual y mostrarlo como valor por defecto

Ya que el dropdown ahora sólo mantiene datos de tipo String se deben guardar los valores del array de resoluciones para poder modificar la resolución posteriormente, haciendo llamada al método *SetResolution* el cual toma como parámetros anchura, altura y un boolean que hace referencia a si se desea que la aplicación se vea a pantalla completa o no. Se puede observar la funcionalidad del método *Start* en la figura 45.

```
private void Start()
{
    resolutionsArray =  Screen.resolutions;
    ResolutionDropdown.ClearOptions();
    List<string> resolutionList = new List<string>();
     int currentResolution = 0;
    for(int i = 0; i<resolutionsArray.Length; i++)
    {
        string option = resolutionsArray[i].width + " x " + resolutionsArray[i].height;
        resolutionList.Add(option);
        if(resolutionsArray[i].width == Screen.currentResolution.width
            && resolutionsArray[i].height == Screen.currentResolution.height)
        {
            currentResolution = i;
        }
    }
     ResolutionDropdown.AddOptions(resolutionList);
     ResolutionDropdown.value = currentResolution;
     ResolutionDropdown.RefreshShownValue();
}
```

Figura 45: Captura del método Start de la clase "OptionsMenu"

TextoSonido

Como último elemento customizado a explicar, el elemento de interfaz *TextoSonido*, comprende a su vez dos elementos diferentes, una *TextBox* que contiene un *String* y un *Slider*, el cual puede servir para muchas funciones diferentes, desde una barra de vida a una barra de control del brillo.

Para este *Slider*, se ha creado con la finalidad de que el usuario pueda controlar el volumen general del juego.

Este *Slider* tiene como umbral de valores desde -80 a 10 debido a que éstos valores son los valores utilizados por todos los altavoces y a su vez por los *AudioMixers*, los cuales se les explicará a continuación, a su vez, los *Sliders* utilizan por defecto el método *OnValueChanged* el cual realiza una acción asignada cada vez que el valor del *Slider* es modificado.

Gracias a este método simplemente se necesita cambiar el valor del volumen del *AudioMixer* cada vez que se cambia el valor del *Silder*.

Un *AudioMixer* es un tipo de *asset* de *Unity* el cual se utiliza para controlar los sonidos en un videojuego, sus funciones comprenden desde controlar el volumen hasta a añadir efectos a diferentes sonidos del juego.

En este caso se va a explicar el *AudioMixer* llamado *MainMixer* el cual controla el volumen general del juego.

Figura 46: Captura de pantalla del AudioMixer "MainMixer"

Pero para poder controlar el *AudioMixer* se necesita realizar una acción por cada variable de este que se quiere alterar ya que no consta de un script asociado con el cual se puede alterar el comportamiento, como se puede observar en la figura 46, el *MainMixer* contiene un parámetro expuesto.

Para ello se tiene que hacer click derecho en una de las propiedades de este y hacer click en exponer parámetro, en este caso se ha expuesto el volumen del *MainMixer* y se ha llamado *VolumenPrincipal*.

Tras deslizar el *Slider* se ejecuta el método por defecto de este llamado *OnValueChanged*, que, a su vez llama a un método creado en el script *OptionsMenu* llamado *SetVolume*. Este recoge el valor del *Slider* el cual el usuario ha modificado y altera el volumen del juego haciendo llamada al método *SetFloat.*

Este método tiene como parámetros un *String* que hace referencia al parámetro expuesto y un *float* que hace referencia al nivel del volumen el cual el usuario quiere mantener durante el juego.

Objetos del Inventario

Antes de explicar el funcionamiento del inventario se deben explicar los tipos de objetos que puede poseer y su funcionamiento. En la figura a continuación se puede observar para qué sirve cada uno.

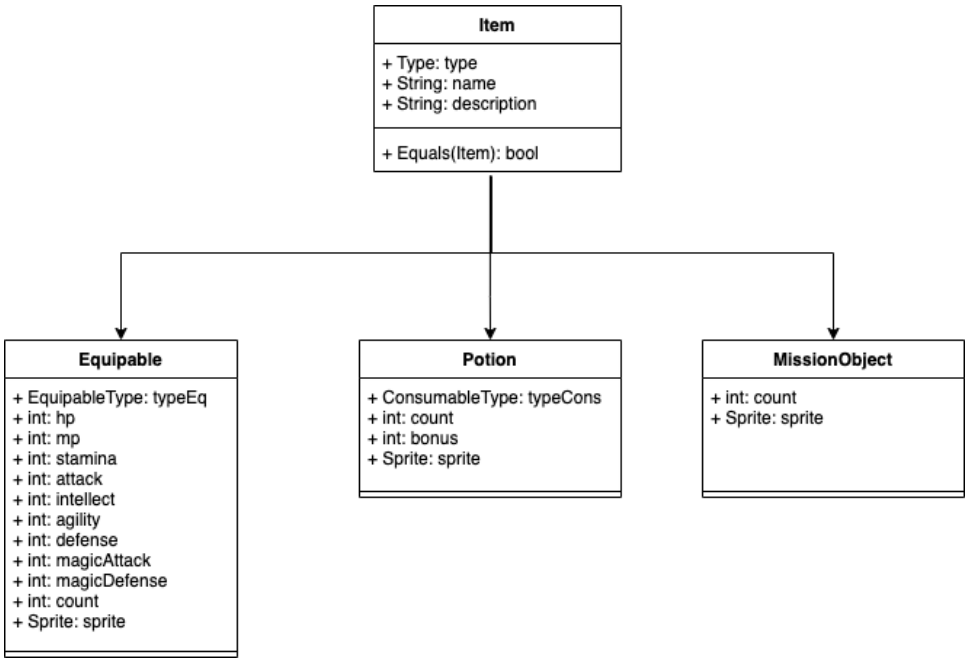


Figura 47: Estructura de los objetos del inventario

Item

Como se puede observar, constan de cuatro clases diferentes, la primera y posiblemente la más importante es la clase Item. Esta clase es la clase padre o superclase de las tres restantes y por tanto define el comportamiento común del resto.

Se puede observar un atributo *tipo* que contiene valores de tipo *Tipo*, esto se debe a que al interior de la clase Item existe un *Enum*, lo cual permite definir un diferenciador entre los tres tipos de clases restantes, por lo tanto, en el *Enum Tipo* hay tres variables diferentes: *Equipable*, *Consumible* y *ObjetoMision*.

A su vez se puede observar que la clase Item es la única que posee un método, este método llamado *Equals* hace la comparación con el *Item* desde el cual se hace la llamada con el *Item* pasado por parámetros, el cual devuelve una variable de tipo *bool*.

Equipable

La segunda clase que se va a explicar consta de la clase *Equipable*. Esta clase contiene, aparte de los atributos de la superclase *Item*, múltiples otros que extienden la funcionalidad de la clase Item. La mayoría de estos atributos constan de enteros que representan las estadísticas que le aportaran al personaje una vez dicho objeto sea equipado por el personaje.

Como también se puede observar, esta clase también contiene su propio *Enum*, llamado *TipoEquipable*. Éste contiene a todos los tipos diferentes de objetos que podrá equiparse el personaje principal.

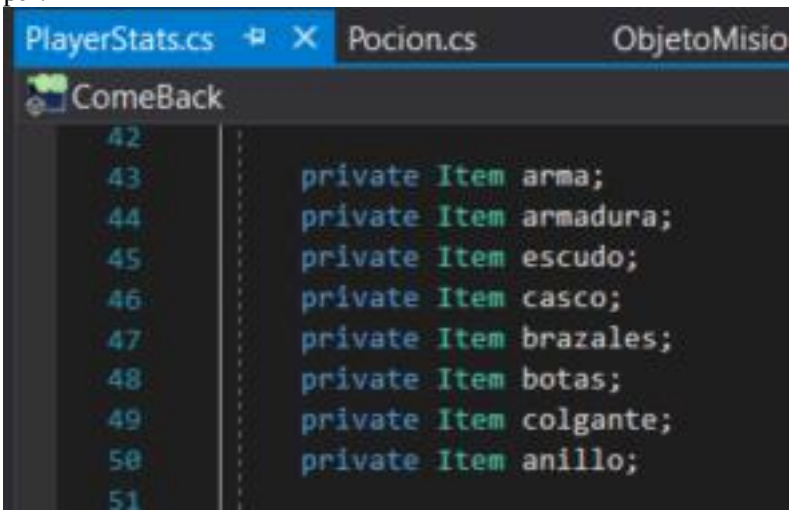


Figura 48: Captura de pantalla de los atributos de tipo "Item" de la clase "PlayerStats"

Como se puede observar en la figura 48, el personaje principal puede equiparse con ocho objetos de diferentes tipos, y, para poder diferenciarlos a la hora de equipar cada uno de los elementos se ha añadido una entrada para cada uno en el *Enum TipoEquipable* de la clase *Equipable*.

ObjetoMision y Pocion

Por último, se explicarán las clases ObjetoMision y Pocion.

La clase poción es utilizada para elementos consumibles solo una vez, en ComeBack existen diferentes tipos de dichos objetos y por ende se ha creado un *Enum* para poder diferenciarlos.

En el *Enum TipoConsumible* existen cuatro variables diferentes: *Vida*, *Mana*, *Fuerza* y *Defensa* que corresponden a los diferentes tipos de pociones que se pueden utilizar en ComeBack.

La clase *ObjetoMision* corresponde al tipo de objeto de un solo uso que no se puede consumir ya que no es ni una poción, ni un objeto equipable sino que es un objeto que se encuentra tras eliminar a un enemigo o en un cofre y su único uso es utilizarlo para pasar una misión del videojuego.

Interfaz de la primera versión del Inventario

Tras explicarles algunos de los elementos de la interfaz de usuario de Unity, se va a pasar a explicar los elementos que comprendían en la primera versión del sistema de inventario, pero antes se debe explicar otros dos elementos de UI de Unity: el *ScrollView* y el *Prefab*.

Figura 49: Captura de pantalla de la interfaz del inventario antes de terminar el 1er MVP

ScrollView

El *ScrollView* es un elemento de interfaz de usuario de *Unity* que permite hacer scroll si su contenido es demasiado grande para ver a simple vista.

Este elemento de interfaz de usuario contiene a su vez otros tres elementos, dos *Scrollbar*, una vertical, una horizontal y un *Viewport*. Este último contiene a su vez también otros dos elementos importantes. El primero consta de una propiedad llamada *Mask* que realiza la función de hacer solo visible parte de los elementos que sean igual de tamaño que su *GameObject* hijo llamado *Content*. El segundo consta del *GameObject Content* el cual permite crear *GameObject* hijos de este que serán finalmente los que se verán en la interfaz de usuario del inventario.

Prefab

Por otra parte, un *Prefab* es una copia de un *GameObject* guardado en una carpeta especifica dentro del proyecto llamada *Prefabs.* Esto permite la reutilización de este elemento múltiples veces, poder crearlo en tiempo de ejecución o simplemente usarlo como plantilla.

Para explicar más a fondo el funcionamiento de estos *GameObjects* se va a explicar los componentes del *Prefab* utilizado para los botones del inventario antes de terminar el primer MVP de ComeBack y su funcionalidad, el cual, tras su implementación, se llegó a la idea de de crear el Asset mencionado anteriormente.

ButtonInventory

Tras explicar lo que es un *Prefab* se puede pasar a explicar el comportamiento del prefab *ButtonInventor* y el cual será utilizado para popular la interfaz de usuario del inventario con los elementos recogidos a lo largo de la partida y cada botón hará referencia a un elemento diferente del inventario y al número de estos que contiene.

Este prefab constaba en primera instancia de un botón con dos referencias de texto diferentes, uno llamado *Name Label* y otro llamado *Quantity*, el primero se usará para mostrar el nombre del objeto y el segundo para mostrar la cantidad de este objeto que se puede usar o equipar, a su vez también consta de otros tres atributos que han de definirse una vez creado el botón ya que de ello depende su funcionalidad.

Estos atributos son: un *InventoryMenu* que es la clase que se hace cargo de la interfaz de usuario del inventario, un *Item* el cual se ha explicado anteriormente y una *ModalPanel* que se explicarán posteriormente.

A su vez, el *ButtonInventory* consta de un script que será común a todos los objetos del inventario llamado *SampleButton*, en este script cabe destacar el método *HandleClick* el cual en el método *Start*, explicado anteriormente su funcionamiento, se le asigna a la función *OnClick* del botón lo cual hace que este método se ejecute cada vez que se haga click sobre el botón.

El método *HandleClick* realiza la función de distinguir qué tipo de *Item* el botón tiene asociado y realiza funciones diferentes dependiendo de qué *Tipo* es el *Item* asociado a este botón.

Debido a la cantidad de elementos a explicar para esta parte se realizará una breve explicación sobre como procesaba el funcionamiento tras hacer click en el botón si el objeto asociado al botón es de tipo *Equipable*, el cual también es el más complejo, pero se dará también una breve explicación sobre los otros dos tipos.

Se empezará explicándo el funcionamiento tras hacer click sobre el botón si el *Item* asociado es de tipo *Pocion* o *ObjetoMision*.

Si se hace click sobre un botón el cual tiene asignado un objeto del tipo *ObjetoMision* aparecerá una *ModalPanel* el cual informará de que este objeto es un objeto utilizado únicamente para misiones y no se puede realizar ninguna acción sobre él.

Para la opción de un objeto de tipo Poción la acción es un poco más compleja, el método realiza la distinción de si la poción es de tipo maná o vida u otro tipo completamente.

Si la poción es de vida o maná primero comprueba si la vida o el maná del personaje está al máximo, si ese es el caso, aparecerá una *ModalPanel* indicando que no se puede utilizar este tipo de poción ya que la vida o el maná están ya al máximo.

En cualquier otro caso, antes de utilizar la poción aparecería una *ModalPanel* indicando si el usuario está seguro de utilizar la poción, si el bonus de la poción de maná o vida es más grande que el faltante del personaje se indicará cuánto se malgastará.

Por último, queda explicar el comportamiento tras hacer click en un botón al cual está asociado un objeto de tipo *Equipable*.

Tras hacer click en éste, aparecería una *ModalPanel* el cual le indicaba al usuario si está seguro de querer equipar este objeto.

En ésta *ModalPanel* le aparecerán dos opciones al usuario: SI y NO, tras hacer click en la opción NO la *ModalPanel* desaparecerá y se podría observar nuevamente la interfaz de usuario del inventario. Si el usuario hiciera click sobre la opción SI, se haría la llamada al método *Equipar* de la clase *PlayerStats*. Este método, si no dispone de objeto equipado previamente simplemente equipa el arma al personaje y sino, devuelve el objeto anteriormente equipado al inventario. Para diferenciar el tipo de objeto *Equipable* se hace uso de un switch el cual diferencia entre los diferentes tipos de objetos *Equipable* para poder hacer referencia al que se quiere utilizar en ese momento.

Otro elemento para tener en cuenta es el método *Setup* que se realizará a la hora de la creación de cada *ButtonInventory* diferente. Este método tiene tres parámetros:

- Item: correspondere al *Item* asociado a este botón
- Count: entero que referencia al número de objetos que contiene el inventario del *Item* asociado
- InventoryMenu: utilizado para poder colocar el botón exactamente como hijo del *Content* del *ScrollView* del menú del inventario

Al llamar al método Setup, para poder mostrar el nombre del objeto y el número que contiene el inventario de este se necesita modificar ambos valores de las propiedades *Text* del botón mencionados anteriormente, el *Name Label* y el *Quantity*, esta funcionalidad se puede observar en la figura 50.

```
public void Setup(Item currentItem,int count, InventoryMenu currentScrollList)
{
    itemData = currentItem;
    nameLabel.text = itemData.nombre;
    quantity.text = count.ToString();
    scrollList = currentScrollList;

}
```

Figura 50: Captura de pantalla del método "Setup" de la clase "ButtonInventory"

ModalPanel

Tras explicarles las diferentes funcionalidades del *ButtonInventory* habéis podido observar el extensivo uso de otro elemento de interfaz de usuario customizado diferente llamado *ModalPanel*.

Para el inventario, en un primer instante, dada su funcionalidad, se requería de un elemento de interfaz de usuario que se asemejara a la *MessageBox* por defecto en C# y, ya que *Unity* no consta de dicho elemento por defecto, se creó una versión propia de este elemento para ajustarlo a nuestras necesidades.

Se ha creado este elemento para que pueda ser usado a lo largo de todo el proyecto solamente cambiando algunas de sus propiedades.

El *ModalPanel* consta de un panel en blanco con un elemento de texto centrado y de uno a tres botones: *YesButton*, *NoButton* y *CancelButton*.

Para poder ser utilizado con las tres diferentes modalidades se ha creado un método *Choice* para cada una de las opciones, la más sencilla siendo la opción con un solo botón y la más compleja siendo con los tres botones.

Para poder ser usado en cualquier clase del proyecto, las funcionalidades de los botones se crearán en las clases desde las que se quiere crear dicho *ModelPanel*. Para ello, hay que utilizar como parámetro de los métodos *Choice* una clase propia de *Unity* llamada *UnityAction*, la cual es una clase usada para crear scripts dinámicos, esto permite asignar funciones que se pueden crear en diferentes clases y se llaman como eventos.

```
public void Choice(string question, UnityAction yesEvent, UnityAction noEvent)
{
    modalPanelObject.SetActive(true);

    yesButton.onClick.RemoveAllListeners();
    yesButton.onClick.AddListener(yesEvent);
    yesButton.onClick.AddListener(ClosePanel);


    noButton.onClick.RemoveAllListeners();
    noButton.onClick.AddListener(noEvent);
    noButton.onClick.AddListener(ClosePanel);

    this.question.text = question;


    yesButton.gameObject.SetActive(true);
    noButton.gameObject.SetActive(true);
    cancelButton.gameObject.SetActive(false);
}
```

Figura 51: Captura de pantalla del método "Choice" de la clase "ModalPanel"

Como se puede observar en la figura 51, el método *Choice* contiene dos botones *yesButton* y *noButton* a los cuales se le pasan como parámetros al método *onClick* por defecto, estos parámetros son *UnityAction* que en otros lenguajes se denominan eventos.

Tanto el *UnityAction yesEvent* como el *noEvent* se crearán en las clases donde se instanciará el *ModalPanel*, por otro lado, el *UnityAction ClosePanel* es común a todos los *ModalPanel* ya que está creado dentro de la propia clase de este.

Este *UnityAction* requiere el uso de la llamada al *GameObject* propio del *ModalPanel* para poder cambiar el atributo común *active* a falso, por ende, desactivando el *ModalPanel* o con otras palabras cerrando la ventana.

Para poder entenderlo se va a explicar el *ModalPanel* usado a la hora de equipar un objeto al personaje.

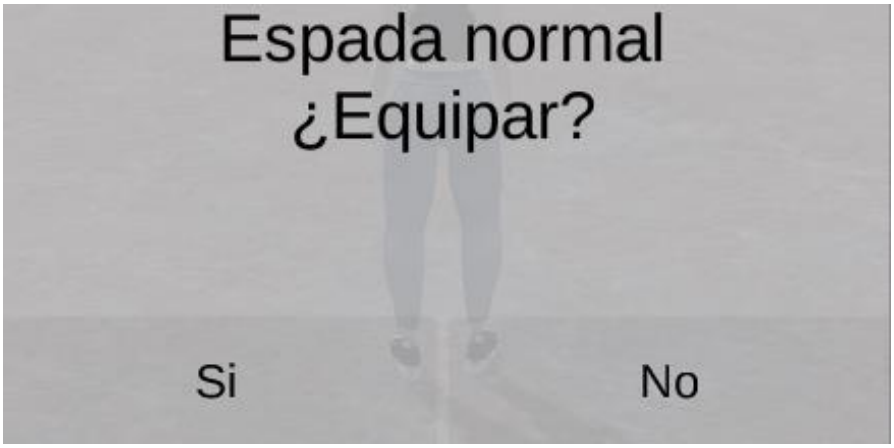


Figura 52: Captura de pantalla de una modal panel tras hacer click sobre un objeto equipable del inventario

Como se puede observar en la figura 52, se está intentando realizar la acción de equipar o no una espada normal, tanto el botón con texto *Si* como el que contiene el texto *No* tienen diferentes funcionalidades.

Si se pulsa sobre el botón Si se realizará la acción anteriormente dicha de equipar un arma. Si se pulsa sobre el botón No, llamará al método creado en la clase *ButtonInventory* que simplemente tiene un *return* vacío. Esto es debido a que no se quiere realizar otra acción más que la acción *ClosePanel* explicada anteriormente si se pulsa dicho botón.

Inventario

Tras haber explicado los diferentes elementos de la interfaz de usuario del inventario se pasará a explicar su funcionamiento completo.

Para poder explicar el funcionamiento del inventario hay que explicar tres clases diferentes:

- *PlayerStats* que será la clase en la cual se almacenará el inventario
- *InventoryMenu*, la clase que comunicará con la interfaz de usuario para mostrar los valores de este
- *SimpleObjectPool*, la cual consta de una clase que actúa como una *Pool* de objetos para crear botones

La clase *PlayerStats* es la clase que se dedica generalmente a modificar las estadísticas del personaje, ya sea porque este sube de nivel o porque se le equipa un arma con estadísticas o porque el personaje consume una poción que aumenta sus estadísticas, pero también es la clase en la que reside un *Dictionary* llamado *inventoryDic* el cual tiene una tupla de valores: un *Item* y un entero. Esto permite tener en una colección de tuplas con un objeto y la cantidad de este.

Esta tupla como se puede observar, caracteriza lo que posteriormente se utilizará como el *ButtonInventory* a la hora de crear y eliminar objetos he decidido utilizar una *Pool* de objetos para encapsular esta funcionalidad y que también pueda ser reutilizable por otros objetos que no sean del inventario.

Para ello se ha creado una clase llamada *SimpleObjectPool* en la se usan dos atributos, un *GameObject* el cual hace referencia al *ButtonInventory* y un *Stack* de *GameObjects* en C# un *Stack* consta de una pila de objetos, el cual representa el concepto LIFO (Last in first out), por lo que el ultimo objeto en entrar en la pila será el primero en salir.

La clase *SimpleObjectPool* consta de dos métodos llamados *GetObject* y *ReturnObject.* El método *GetObject* el cual realiza la siguiente acción: Declara un *GameObject*, si la pila de objetos contiene algún objeto se utiliza el método *Pop* de la clase *Stack* eliminar el objeto de la pila y devolver la cima de la pila.

Si la pila está vacía entonces se pasa a instanciar un *ButtonInventory* nuevo utilizando el método *Instantiate* de la clase propia de *GameObject* y se le asigna ésta *Pool* de objetos al objeto instanciado y se activa ejecutando el método *SetActive* pasándole el valor *True* como parámetro.

El método *ReturnObject* de la clase *SimpleObjectPool* realiza la acción de comparar la *Pool* de objetos desde la cual el objeto ha sido creado con al *Pool* actual, si son la misma entonces desactiva el *GameObject* y lo añade a la pila de objetos utilizando el método *Push* de la clase *Stack* el cual añade el objeto a la cima de la pila para posteriormente poder ser reutilizado.

Si el *GameObject* no proviene de esta pila entonces se mostrará un mensaje de error diciendo que el objeto no ha sido instanciado por esta *Pool* de objetos y se pasa a destruir el objeto con el método *Destroy*.

Todo esto permite que, cuando la cantidad de un objeto llegue a cero, o lo que es lo mismo, se utilicen todos los objetos que estén contenidos en el mismo botón creado con el prefab *ButtonInventory*, en vez de destruir el *GameObject*, pasa a estar inhabilitado. Tras ello, se guarda en la pila de la clase *SimpleObjectPool* para su reutilización en cuanto se recoja un nuevo objeto o uno de los objetos equipados se vuelva a añadir al inventario.

Recoger Objeto

Cuando el personaje principal recoge un objeto, ya sea de un cofre, al terminar una pelea o si se encuentra un objeto en el suelo, hay unas acciones predeterminadas que se llevan a cabo.

Lo único que cambia a la hora a la hora de recoger un objeto es desde dónde se hace la llamada de recoger el objeto. Si es desde el suelo se utiliza un trigger, propio de *Unity*, si se hace desde un cofre se hace a la hora de abrir el cofre y si es al terminar una batalla existe un método de *AñadirObjetos* al terminar la pelea.

Todo ello conlleva un método común que es explicado por la figura 53.

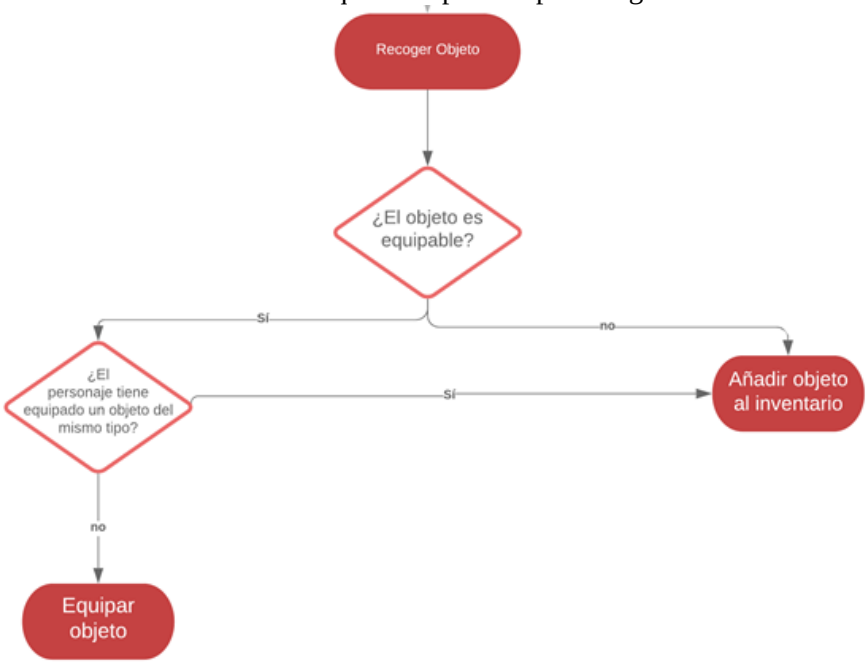


Figura 53: Proceso de recoger un objeto

Al recoger un objeto se pueden dar dos acciones finales, ya que si es una poción o un objeto de misión el objeto se añade directamente al inventario.

Si el objeto en cuestión es un objeto equipable primero hay que observar si existe un objeto equipable del mismo tipo ya equipado, si se da ese caso, el objeto recogido se añade al inventario, si no, se equipa automáticamente.

Esto hace que el usuario no tenga que acceder al menú del inventario para equipar dicho objeto cuando el personaje no tiene ningún objeto equipable de ese tipo.

Tras haber explicado la primera versión del sistema inventario, creado en ComeBack antes del primer MVP, se explicará la modificación de éste que resultó en el asset *Simple Inventory and Dialogue System*.

Tras haber finalizado esta primera versión del inventario, se notaba una falta importante de elementos visuales que atraigan al usuario. A su vez, se notó que el uso de una ventana modal para el inventario era algo tedioso y que el usuario no debería hacer click de nuevo si quiere usar o equipar un objeto.

Interfaz del Inventario en el asset Simple Inventory and Dialogue System

Muchos elementos de la interfaz de usuario han sido reutilizados para la creación de este sistema de inventario, uno de los elementos que se ha modificado drásticamente es el *ButtonInventory* el cual se ha modificado para que pueda contener una imagen y un elemento de texto que contenga la cantidad de dicho objeto contenido en el inventario, este cambio se puede observar en la figura 54.

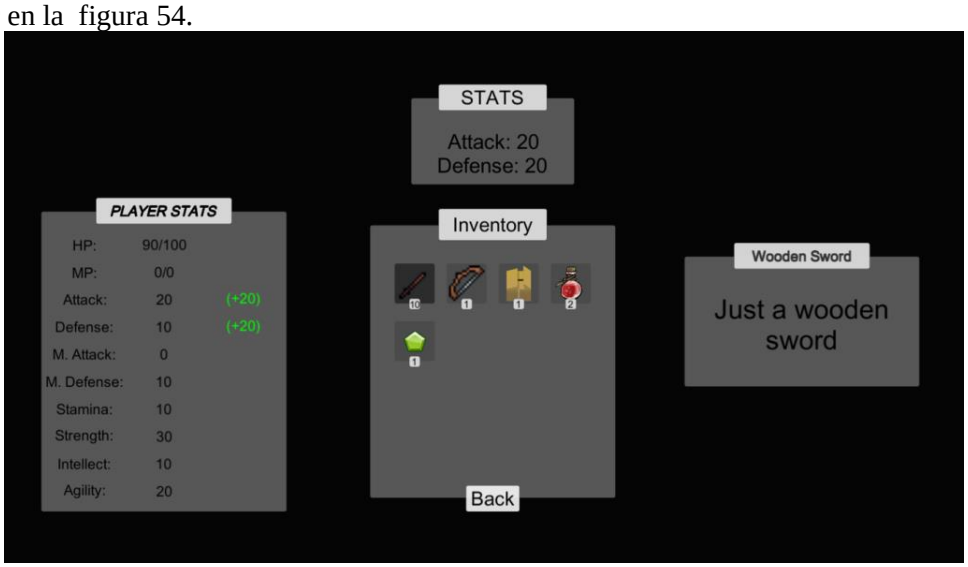


Figura 54: Captura de pantalla del Inventory Menu del Asset Simple Inventory and Dialogue System

Tomando como referencia la figura 54, se pueden observar diferentes objetos de interfaz de usuario que se va a explicar a continuación. Eestos son:

- Inventory Panel
- StatsPanel
- Info Panel
- Player Stats Panel

En este caso, al hacer click sobre el botón que contiene un objeto, se realizarán las acciones mencionadas antes directamente, por lo tanto, se requería de un elemento de UI que no resultase tedioso a la hora de utilizar el inventario, pero a su vez dé algo de información con respecto a los objetos del inventario, y dependiendo del objeto se mostrará un panel u otro.

Inventory Panel

El Inventory Panel, corresponde al Scroll View mencionado en el punto anterior creado para la primera versión del inventario, modificado para contener botones de un carácter más visual. En este caso, ya que los botones no contienen simples elementos de texto, sino que contienen una imagen, también se ha modificado el diseño de este, cambiando el componente Vertical Layout por uno llamado Grid Layout. Este, como representa su nombre, hace que todos los objetos hijos de este elemento se vean expuestos en forma de cuadricula.

Stats Panel

Este elemento de UI representa las estadísticas del objeto en cuestión, para crear este elemento se tomó como referencia la ventana modal explicada anteriormente, solo que este panel no se muestra tras hacer click, sino que se muestra al colocar el puntero del ratón sobre el *ButtonInventory*, este panel se muestra únicamente para los objetos de tipo equipable y de tipo poción.

Al colocar el puntero sobre un objeto de tipo equipable muestra las estadísticas de este que mejorarán las estadísticas del personaje.

En cambio, al colocar el puntero sobre un objeto de tipo poción, aparecerá un texto descriptivo de dicha poción, que, a su vez contiene funcionalidades diferentes si se trata de pociones de tipo vida o maná.

Si la vida o el maná del personaje están al máximo, aparecerá en dicho panel, una breve descripción de porqué no se puede utilizar dicha poción en ese momento, esta funcionalidad se puede observar en la figura 55 y también se podía observar anteriormente en la ventana modal.

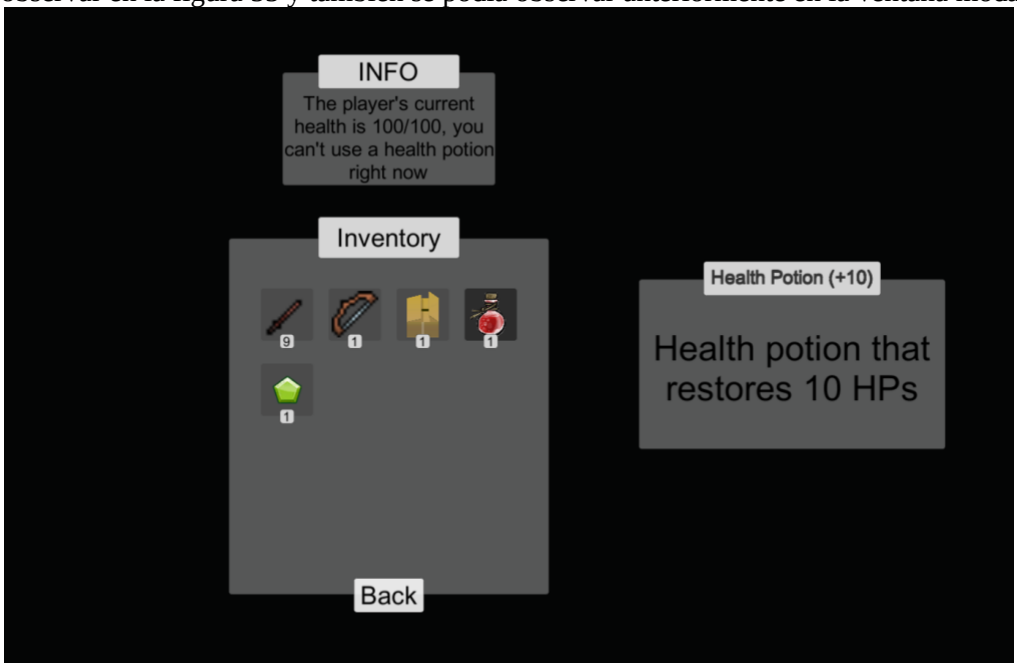


Figura 55: Captura de pantalla del Asset Simple Inventory and Dialogue System

Info Panel

El panel Info Panel, colocado al lado derecho del panel del inventario es un panel simple que contiene únicamente dos elementos de texto: un título y una descripción. El título corresponde al nombre del objeto sobre el cual se está colocando el puntero y la descripción hace referencia a la variable descripción de este.

Este panel se muestra al colocar el puntero sobre cualquier objeto del inventario, en la figura 55 se puede observar dicho funcionamiento al colocar el puntero sobre el objeto *Heath Potion (+10)*.

Player Stats Panel

Siguiendo la temática del resto de paneles explicados recientemente, este también se muestra únicamente al colocar el puntero sobre un botón del inventario, para ser más específico, únicamente sobre un botón que contenga un objeto de tipo equipable. Este panel permite al usuario poder observar las estadísticas del personaje y al mismo tiempo como se modificarían dichas estadísticas si el usuario decide equipar el objeto sobre el cual esta colocando el puntero, esta funcionalidad se puede observar en la figura 56 al colocar el puntero sobre el botón que contiene el objeto llamado *Wooden Sword*.

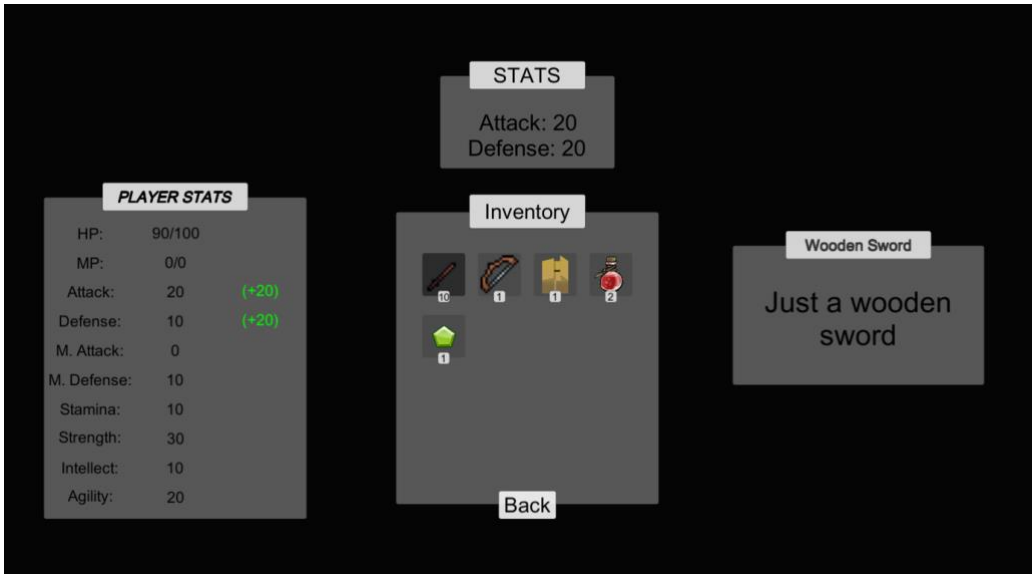


Figura 56: Captura de pantalla del Asset Simple Inventory and Dialogue System

Para realizar esto, se creó un panel que contiene una cuadricula con tres columnas que contienen 3 tipos de elementos de texto, la primera columna corresponde al nombre de las variables de las estadísticas del personaje, la segunda columna corresponde al valor actual de dichas estadísticas. La última columna corresponde a la modificación que proporcionaría dicho elemento equipable si el usuario decide equipar el objeto sobre el cual se está colocando el puntero.

Como se puede observar en la figura 56, si el objeto mejora una variable, esta modificación se mostrará de color verde.

Si se dan los siguientes puntos:

- Personaje equipado un objeto equipable
- Se coloca el puntero sobre otro objeto equipable del mismo TipoEquipable
- El objeto sobre el cual se coloca el puntero empeora una de las estadísticas

Entonces la estadística empeorada se mostraría de color rojo como se puede observar en la figura 57.

Figura 57: Captura de pantalla del Asset Simple Inventory and Dialogue System

En este caso, el personaje tiene ya equipado la *Wooden Sword* mencionada anteriormente, que contiene 20 puntos de ataque y 20 puntos de defensa, si se coloca el puntero sobre el objeto *Wooden Bow* que es otro objeto equipable de tipo arma, al contener únicamente 20 puntos de agilidad y 20 puntos de ataque, empeora la estadística de defensa del jugador si se decide

equipar, mostrándose en rojo la modificación de la estadística de defensa en el panel *Player Stats*.

El elemento encargado de administrar el inventario es el *GameObject* llamado *Inventory Manager*. Además de encapsular las funcionalidades de añadir elementos al inventario, contener el *inventoryDic* que antes estaba contenido en la clase *PlayerStats* de ComeBack, también contiene la funcionalidad de recoger los elementos que el usuario quiere almacenar en el inventario desde el comienzo del juego.

Estos elementos se añaden a través de tres listas diferentes con que contendrán los diferentes tipos de objetos que pueden ser almacenados en el inventario.

Estas listas son *EquipableList*, *PotionList y MissionObjectList*. Estas listas son serializables por lo que los objetos que pueden contener son editables a través del inspector de Unity como se puede observar en la figura 58.

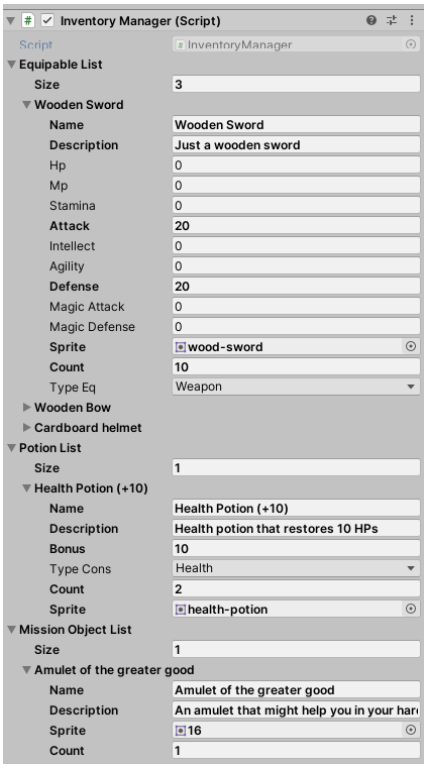


Figura 58: Captura de pantalla del editor de Unity con variables del Inventory Manager

En el método *Start*, ya explicado anteriormente, se ejecutan las funciones *GatherEquipableList*, *GatherConsumableList* y *GatherMissionObjectList* que son las encargadas de recoger todos los elementos almacenados en dichas listas y añadirlas al diccionario mencionado anteriormente.

Como se puede observar, los objetos también contienen un atributo *Sprite* que se utiliza para popular la imagen del botón *ButtonInventory*.

Sistema de diálogos

Los diálogos en un videojuego del estilo RPG es una de las facetas más importantes del este estilo. Los diálogos sirven para múltiples funciones, ya sea para dar una pista al jugador hacia dónde debe ir para cumplir la misión, para sumergir al jugador al jugador en la historia y en el lore o simplemente para añadir un comentario cómico en un momento tenso del desenlace del juego.

Por ello se ha decidido que es una de las funcionalidades con máxima prioridad para el primer MVP.

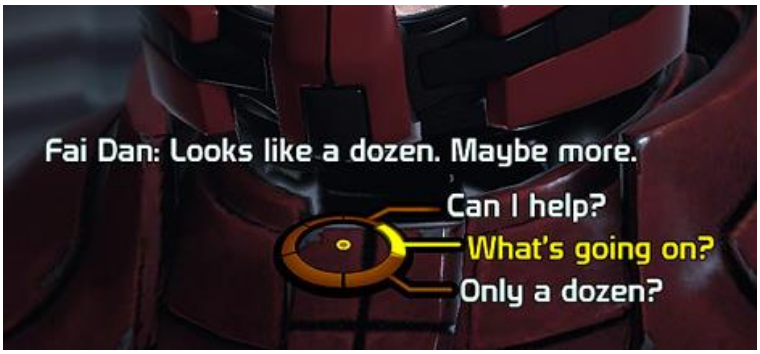


Figura 59: Ejemplo de diálogo del videojuego Mass Effect

Existen varias maneras de crear diálogos en los videojuegos hoy en día ya que pueden ser diálogos con respuesta que influye en el videojuego como en el videojuego *Mass Effect* como se puede observar en la figura 59, diálogos con voz los cuales son más realistas o diálogos de texto simples.

Debido a que tanto los diálogos con respuesta como los que son con voz requieren de un alto conocimiento de Unity y requeriría de una carga de trabajo incluso más alta, para el primer MVP se ha decidido realizar un Sistema de Diálogos de texto simple, el cual se puede reutilizar durante cualquier momento del videojuego y en cualquier escena tal y como se hizo previamente con la ventana modal.

Interfaz de usuario

La interfaz de usuario del sistema de diálogos de ComeBack consta de un prefab, que como ya se ha explicado anteriormente se utiliza para poder reutilizarlo tantas veces como sea necesario durante este videojuego.

El prefab *DialogueBox* consta de un panel, una imagen, dos elementos de texto y un botón como se puede observar en la figura 60.

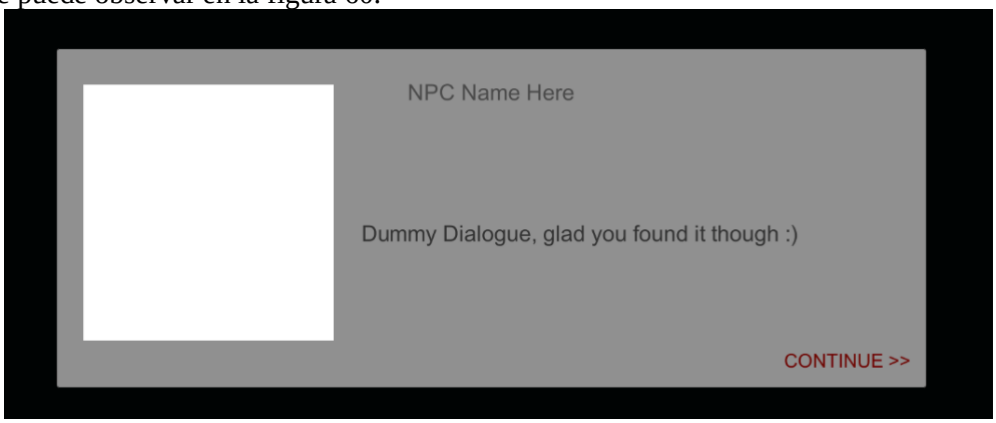


Figura 60: Captura del prefab de dialogo "DialogueBox"

Tanto el primer elemento de texto consta del título del dialogo como la imagen normalmente harán referencia al personaje el cual estará comunicándose en ese momento, el segundo será el

texto que hará referencia al mensaje el cual el personaje estará diciendo y, por último, el botón Continuar pasará al siguiente mensaje o, si es el último mensaje cerrará la ventana de diálogos.

Este prefab contiene una propiedad llamada *Dialogue* que es una clase la cual contiene un área de texto y un array de *strings*.

Para hacer que todo funcione es necesario crear una *GameObject* el cual controle los diálogos de la escena llamado *DialogueManager* el cual tiene asociado a su vez un script con el mismo nombre.

Este script tiene como atributos dos elementos de texto que corresponden con el prefab de la ventana de diálogo, un *Animator* el cual se dedica a realizar las animaciones de la ventana de dialogo y una *Queue* o cola de datos de tipo string la cual almacenará los diálogos de los personajes, a su vez contiene dos métodos muy importantes para el correcto funcionamiento de los diálogos, estos son *StartDialogue* y *DisplayNextSentence*.

```
public void StartDialogue (Dialogue dialogue)
{
    animator.SetBool("IsOpen", true);

    nameText.text = dialogue.name;

    sentences.Clear();

    foreach (string sentence in dialogue.sentences)
    {
        sentences.Enqueue(sentence);
    }

    DisplayNextSentence();
}
```

Figura 61: Captura pantalla del método "StartDialogue" de la clase "DialogueManager"

El método *StartDialogue* recoge los datos del título de la ventana de dialogo y las frases que deben mostrarse en ella, encolándolas en la cola llamada *sentences*, tras recorrer y encolar las frases hace una llamada al método *DisplayNextSentence*. La funcionalidad se puede observar en la figura 61.

El método *DisplayNextSentence* desencola las frases una a una y por cada una de ellas hace una llamada al método *TypeSentence* el cual escribe un carácter de la frase por cada *endframe* así dándole un efecto de que el texto se está escribiendo en el momento.

Todos los paneles y botones de este Asset contienen imágenes de fondo por defecto en Unity, haciéndolo altamente personalizable para el videojuego que el usuario quiera crear, esto se puede observar en el inspector de Unity, mostrado en la figura 62.

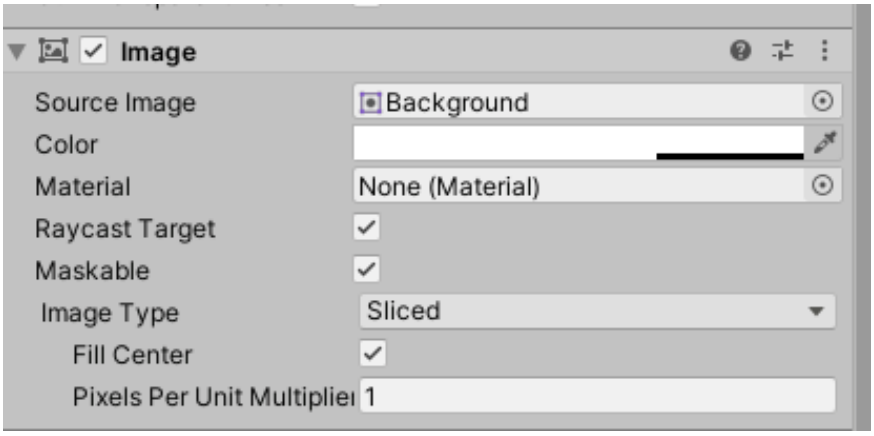


Figura 62: Captura de pantalla del componente Image en el inspector de Unity

Para información más detallada y como poder utilizar el asset en el videojuego, puede acceder al siguiente enlace para descargar el manual de usuario:

https://bit.ly/3jGyt3Z

Validación

Para la validación del asset *Simple Inventory and Dialogue System*, he realizado una personalización de este en ComeBack, a continuación, les explicaré la personalización de ambos sistemas en este videojuego.

Inventario

Figura 63: Captura de pantalla del Inventory Menu en el videojuego ComeBack

De un simple vistazo a la figura 63 se puede observar que tanto las imágenes de fondo de los paneles como la fuente del texto han sido modificadas para ajustar la temática de este videojuego.

A su vez se puede observar que el *InfoPanel* y el *StatsPanel* han sido unidos en uno y se ha añadido un panel nuevo llamado *EquipmentPanel*.

Este panel se activa tras colocar el puntero sobre un objeto del inventario de tipo equipable, lo que permite al jugador poder observar de manera visual que objetos el personaje tiene equipados.

Tras pulsar sobre el botón que contiene la espada normal, resaltado en la imagen anterior, este objeto equipable de tipo arma se le equipará automáticamente y si se coloca el puntero sobre otro objeto equipable, la imagen de la espada normal se podrá observar en el panel "Arma" del panel *EquipmentPanel* como se puede observar en la figura 64.

Figura 64: Captura de pantalla del Inventory Menu en el videojuego ComeBack

Diálogos

A pesar de que ComeBack contiene únicamente un personaje NPC con el que se puede interactuar, se han realizado dos modificaciones a los diálogos.

La primera ventana de diálogo que puede observar el jugador se encuentra en la escena de transición entre el menú principal tras pulsar sobre el botón Nueva Partida y el primer nivel de ComeBack.

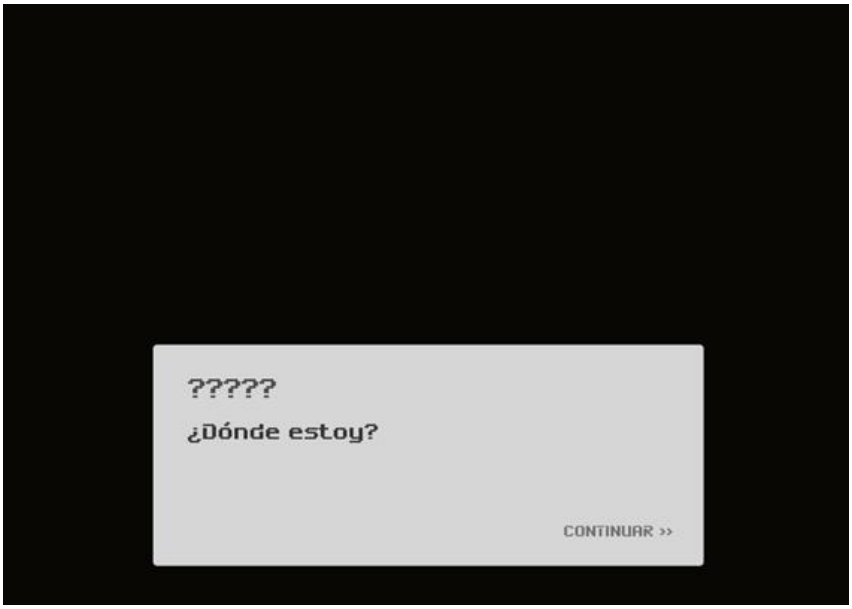


Figura 65: Captura de pantalla del dialogo en la escena PreLvl1

En esta escena se observa un fondo negro con una ventana de diálogos en la que el personaje principal no sabe lo que está sucediendo. El personaje acaba de aparecer en un mundo distópico y para el usuario, tras cargar la escena "PreLvl1" aparece una ventana de diálogo que corresponde a lo que el personaje principal está pensando en ese momento. Tras terminar el dialogo se carga la escena "Lvl2".

Con esto se da más información al usuario sobre lo que está sintiendo el personaje principal, sumergiendo así al usuario en la historia.

Tras ello, al aparecer en el primer nivel de ComeBack el jugador puede observar un NPC con aspecto esquelético, este NPC, a pesar de tener apariencia de un enemigo, este NPC es completamente inofensivo.

Al acercarse a este NPC, aparecerá un panel indicativo que sugerirá pulsar sobre la tecla "T" para iniciar el diálogo.

Figura 66: Captura de pantalla de la escena Lvl1 en el videojuego ComeBack

Este *GameObject* que representa un personaje NPC contiene de un *Collider* que, tras entrar tras él, activa el panel observado en la figura 64 y mencionado previamente.

Tras pulsar sobre la tecla "T" se iniciará el diálogo que muestra al jugador que este personaje es completamente inofensivo ya que informará que él no ha sido siempre un esqueleto, sino que algo lo ha debilitado, dejándolo en este estado y también que si se muere en este lugar, también se morirá en la vida real, dando al jugador más información acerca del mundo en el que se encuentra actualmente y que no se trata de la vida real.

Un ejemplo de este dialogo se puede observar en la figura 67.

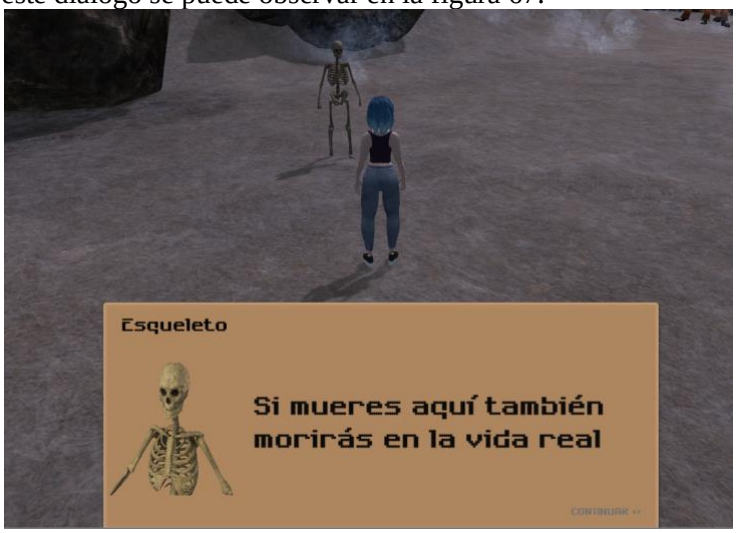


Figura 67: Captura de pantalla de la escena Lvl1 en ComeBack

Como se puede observar, el titulo muestra el nombre del personaje que está hablando en este momento y una imagen representativa de este. Al mismo tiempo, se puede observar como la imagen de la ventana de diálogos ha sido modificada para ajustar la temática de ComeBack como ha podido observarse también en los paneles del inventario.

CAPÍTULO 7 CONCLUSIONES

La originalidad de la historia y la fusión de conceptos orientales y occidentales en un mismo RPG comentados en el primer capítulo han sido alabadas por más del 90% de los jugadores de ComeBack tal y como se ha observado en el segundo capítulo. Es mas, el equipo ha recibido mensajes en sus redes sociales privadas de jugadores preguntando acerca de noticias del estado del proyecto.

CONSECUCIÓN DE LOS OBJETIVOS DEL TFG

A continuación, se van a exponer los objetivos logrados por parte del equipo, destacando el miembro del equipo que ha sido encargado de tal objetivo.

En primer lugar, cabe destacar la labor de Altea Taulet Rius, que ha sido la encargada del diseño de ComeBack, logrando:

- Diseñar 3 niveles completamente acabados en su aspecto visual
- Modelar y animar el personaje principal.

Por otra parte, David Guillen Mateu ha sido el encargado del diseno y desarrollo del sistema de combate, cumpliendo con uno de los objetivos marcados en el primer capitulo:

- Establecer sistema de combate.

Por ultimo, caben destacar los objetivos logrados por el autor de esta memoria, Nathan Tyler Llavador, el cual ha alcanzado los siguientes objetivos marcados en el primer capitulo:

- Programar menú principal.
- Creación de un Sistema de Inventario Simple
- Programar un Sistema de Diálogos que permita sumergir al jugador en la historia y el *lore* del videojuego
- Creacion del Asset Simple Inventory and Dialogue System

Al mismo tiempo, el autor de este documento ha logrado otros objetivos no establecidos en el primer capitulo:

- Menu in-game, el cual pausa el juego para poder realizar otras acciones:
  - Guardar/Cargar
  - Abir menú del inventario
  - Volver al menú principal
  - Abrir menú estado
- Menu estado, el cual nos informa de las estadísticas del personaje y los objetos que tiene equipados
- Sistema de guardado y cargado

Además de lo anterior, se ha realizado la publicación de dos demos jugables del producto a través de la plataforma Patreon.

DIFICULTADES Y RETOS ENFRENTADOS

Se han explicado a través de todo este documento, los desafíos y retos encontrados a la hora de realizar este proceso. Se mencionan ahora todos ellos de forma reducida.

- Aprendizaje de cero del motor de videojuegos Unity
- Aprendizaje desde cero sobre el desarrollo de videojuegos
- Aprendizaje sobre los repositorios de versiones utilizados durante este proyecto, tanto Github como Bitbucket

EXPERIENCIA ADQUIRIDA

Desde los inicios del desarrollo de la idea de negocio, se han estado aprendiendo tanto nuevas técnicas como herramientas desconocidas a los integrantes del equipo.

Ha sido la primera vez que se ha realizado un proyecto de este tamaño y que necesita una gran cantidad de tiempo, hasta ahora solo se tenía experiencia en proyectos de clase y pequeños proyectos propios.

Entrando en el ámbito más técnico, se ha tenido que aprender casi de cero el manejo de Unity y muchos recursos de programación que se han tenido que usar.

Se ha trabajado mucho con Unity aprendiendo mucho de su funcionamiento, desde la parte más gráfica de diseño de niveles hasta la parte más técnica de cómo funcionan los diversos componentes en conjunto con los scripts.

CONOCIMIENTO BASE EMPLEADO PARA EL DESARROLLO DEL PROYECTO

En cuanto al conocimiento utilizado el cual ha sido impartido durante el estudio del grado caben destacar las siguientes asignaturas:

- Proceso de Software
- Proyecto de Ingeniería del Software

Estas dos asignaturas han sido clave en este proyecto de emprendimiento en el momento de efectuar tanto el estado del arte y como el desarrollo de la idea de negocio. También cabe subrayar que las dos han estado enfocadas al trabajo en equipo, competencia que ha resultado primordial en el proceso de desarrollo.

Todas las asignaturas de programación de la carrera han cimentado la base para aprender a programar los scripts en C#, ya que los fundamentos de la mayoría de los lenguajes de programación estudiados durante la carrera son muy parecidos.

Otra asignatura importante que destacar es Análisis y Especificación de Requisitos, fundamental a la hora de aprender técnicas para la generación de la idea de negocio.

Cabe destacar a su vez:

- Gestión de Proyectos, dado el desarrollo realizado y la especificación de objetivos marcados
- Economía, que sentó las bases para el estudio de la proyección económica.

CAPÍTULO 8 TRABAJO FUTURO

A pesar del abandono de los otros dos integrantes del equipo tras haber presentado sus respectivos TFGs, se planea terminar el segundo MVP y tratar de presentar el producto a un numero más amplio de posibles clientes.

Para ello primero se tratará de encontrar nuevos integrantes mediante sesiones de *networking*, todo ello para poder terminar el segundo MVP e intentar presentar el producto a un público más amplio. Dicha presentación se realizaría a través de un evento publico relacionado con el mundo de los videojuegos o una quedada. Para un posible tercer MVP, aparte de añadir correcciones percibidas durante el experimento del segundo MVP, también se planificará incrementar el número de enemigos y niveles. También se querrá iniciar el desarrollo visual de los ataques de tipo mágico, a través de un sistema de partículas, mediante el sistema de edición de partículas de Unity y el sistema de efectos visuales de Unity llamado VFX.

Al mismo tiempo se perfilará el sistema de combate, añadiendo más profundidad al mismo, definiendo zonas que perjudiquen o ayuden a los diferentes tipos de clases y permitiendo que los personajes puedan desplazarse a las zonas que más les potencien sus habilidades.

Tras completar el producto y crear una base de usuarios más grande, se planificará la introducción del juego al mercado global mediante el uso de la plataforma Steam. Para poder llegar a este punto aun queda un largo camino, ya que el proceso de creación de un videojuego es extenso el cual se ve incrementado por el género de videojuego que se esta desarrollando y sus características.

En cuanto al trabajo que requiere acción inmediata, sin ninguna duda se trata de la terminación de los dos compañeros acompañarán a la protagonista, así como los personajes NPCs interactivos. Esto se debe a que una de las características del videojuego al que se le quiere dar más relevancia es al desarrollo de la historia principal. Esto se debe a que una de las causas que llevó a la creación de ComeBack fue la intención de querer contar una historia.

GLOSARIO

- Análisis DAFO: plantilla utilizada para analizar las posibles debilidades, amenazas, fortalezas y oportunidades de una idea de negocio.
- Árbol de habilidades: conjunto de habilidades que el jugador puede llegar a utilizar a medida que avanza en el juego. Toma este nombre ya que se disponen de forma que algunas solo sean accesibles a través de otras, formando así un árbol de decisión. De forma habitual, el árbol suele ser de elección completa, obligando al jugador a elegir una clase específica.
- *Brainstorming:* proceso llevado a cabo con el fin de crear nuevas ideas de forma grupal en el que se aportan gran cantidad de conceptos sin ser desarrollados en exceso.
- Campaña: modo establecido en los videojuegos caracterizado por ser jugado por un solo usuario, que pasa a través de una serie de sucesos y observa la narración de una historia.
- Clases: diferentes perfiles que pueden adoptar los personajes de un videojuego.
- *Crowfunding:* sistema público basado en la financiación colectiva de proyectos.
- Demo: versión reducida de un juego desarrollada con el fin de ser probada y poder obtener la opinión de los usuarios a la par que hallar errores en el producto.
- *Dps*: perfil del personaje dedicado a infligir la mayor cantidad de daño posible al enemigo. El término viene de la expresión inglesa *Damage Per Second.*
- *Early adopters*: primeros usuarios a los que se muestra el producto desarrollado con el fin de obtener su opinión al respecto.
- Lean Canvas: plantilla para gestionar y planificar una idea de negocio.
- Mapa mental: método visual para plasmar información de diferentes conceptos y la relación entre ellos.
- *Merchandising:* productos a la venta que representan o promocionan una marca.
- Modelado: proceso llevado a cabo para elaborar un objeto en tres dimensiones.
- Motor gráfico: software que facilita una serie de herramientas de las que se hará uso para el desarrollo de videojuegos.
- MVP: producto que posee las suficientes características como para poder ser analizado por los potenciales clientes.
- RPG: género de los videojuegos caracterizado por estar basado en un sistema de estadísticas que condiciona todo el funcionamiento del juego, desde el daño hecho hasta el recibido. Originalmente se basó en los juegos de mesa de rol como indican su nombre original en inglés Role-Playing Game.
- JRPG: Acrónimo del inglés Japanese Role Playing Game (Juego de Rol Japonés). Término para referirse a los RPG o juegos de rol desarrollados en Japón -y países limítrofes- dado que cumplen con una serie de premisas estéticas y formales muy similares, como aspecto desenfadado y muy colorista, personajes muy jóvenes y andróginos, estética tipo manga/anime, historias lineales o combate por turnos.
- CRPG: CRPG es un acrónimo para Computer Role Playing Game (Juego de Rol de computadora). Y estos juegos son una adaptación a la computadora de los primeros juegos de Rol. Ahora, han evolucionado un poco más para ser los RPG que se conocen hoy en día. Ya que no son únicamente para computadora, sino para móviles y consolas, ahora se les trata simplemente como RPG.
- Sanador: perfil del personaje dedicado principalmente a la cura y defensa de sí mismo y de sus aliados.
- Tanque: perfil del jugador dedicado a resistir la mayor cantidad de golpes, protegiendo así a sus aliados.

Bibliografia

1. **Wingfield, Nick.** *Game Maker Without a Rule Book*. *The New York Times.* [En línea] 8 de Septiembre de 2012. [Citado el: 3 de Mayo de 2019.] https://www.nytimes.com/2012/09/09/technology/valve-a-video-game-maker-with-few-rules.html?_r=2&pagewanted=all.

2. **Ries, Eric.** *The lean startup: how today's entrepreneurs use continuous innovation to create radically successful businesses.* Nueva York : Crown Business, 2011. ISBN 9780307887894.

3. **The CRPG Project.** *The CRPG Project: A Guide to Computer Role-Playing Games*. [En línea] [Citado el: 17 de Diciembre de 2018.] https://crpgbook.files.wordpress.com/2019/04/crpg_book_2.0-1.pdf.

4. **Barton, Matt.** *Dungeons & Desktops: The History of Computer Role-Playing Games.* Canada : A K Peters, Ltd, 2008. ISBN-13:978-1-56881-411-7.

5. *Distribution of video gamers worldwide in 2017, by age group and gender*. *Statista.* [En línea] 2018. [Citado el: 5 de Enero de 2019.] https://www.statista.com/statistics/722259/world-gamers-by-age-and-gender/.

6. *Age breakdown of video game players in the United States in 2018*. *Statista.* [En línea] 2019. [Citado el: 5 de Enero de 2019.] https://www.statista.com/statistics/189582/age-of-us-video-game-players-since- 2010/.

7. **Morris, Carolyn Pairitz.** *The Demographics of Video Gaming*. *Earnest.* [En línea] 19 de Abril de 2018. [Citado el: 5 de Enero de 2019.] https://www.earnest.com/blog/the-demographics-of-video-gaming/.

8. *Linear Motion Battle System*. *GiantBomb.* [En línea] [Citado el: 13 de Febrero de 2019.] https://www.giantbomb.com/linear-motion-battle-system/3015-219/.

9. **Valentine, Reb.** *Tales of Vesperia: Definitive Edition - Everything you need to know*. *Imore.* [En línea] 10 de Enero de 2019. [Citado el: 13 de Febrero de 2019.] https://www.imore.com/tales-vesperia-definitive-edition-everything-you-need-know.

10. *Active Time Battle*. *GiantBomb.* [En línea] [Citado el: 14 de Febrero de 2019.] https://www.giantbomb.com/active-time-battle/3015-95/.

11. **Robertson, Adi.** *Inside Patreon, The Economic Engine of Internet CultureE. The Verge* . [En línea] 3 de Agosto de 2017. [Citado el: 17 de Abril de 2019.] https://www.theverge.com/2017/8/3/16084248/patreon-profile-jack-conte-crowdfunding-art-politics-culture.

12. **Constine, John.** *Patreon doubles in a year to 1M paying patrons and 50K creators. TechCrunch.* [En línea] 2017. [Citado el: 17 de Abril de 2019.] https://techcrunch.com/2017/05/18/patreon-pushes-as-youtube-stutters/?guccounter=1&guce_referrer_us=aHR0cHM6Ly9lbi53aWtpcGVkaWEub3JnLw&guce_referrer_cs=8nOzqfSp-YFcSN6IZnV-Yw.

13. **Steam.** *About. Steam.* [En línea] [Citado el: 16 de Mayo de 2019.] https://store.steampowered.com/about/.

14. *30 Interesting Steam Stats and Facts (2019) . Video Game Stats.* [En línea] [Citado el: 16 de Mayo de 2019.] https://videogamesstats.com/steam-stats-facts/.
15. **Barton, Matt, and Shane Stacks.** *Dungeons and Desktops: The History of Computer Role-Playing Games 2e. AK Peters/CRC Press, 2019.*

16. **Pérez Latorre, Óliver.** *Géneros De Juegos Y Videojuegos : Una aproximación Desde Diversas Perspectivas teóricas». Comunicació: Revista De Recerca I d'anàlisi, junio de 2011, p. 127-46, https://www.raco.cat/index.php/Comunicacio/article/view/67950*