Droid Fighters (TFM)

Autor: Pablo Molina Parellada
Tutor: Helio Tejedor Navarro
Profesor: Joan Arnedo Moreno

Máster universitario de Diseño y Programación de Videojuegos
Programación Avanzada

18/06/2023

Créditos/Copyright

Esta obra está sujeta a una licencia de Reconocimiento- NoComercial-SinObraDerivada 3.0 España de CreativeCommons.

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Droid Fighters*
Nombre del autor:	*Pablo Molina Parellada*
Nombre del docente:	*Helio Tejedor Navarro*
Nombre del PRA:	*Joan Arnedo Moreno*
Fecha de entrega:	*06/2023*
Titulación o programa:	*Máster universitario de Diseño y Programación de Videojuegos*
Área del Trabajo Final:	Programación Avanzada
Idioma del trabajo:	*Castellano*
Palabras clave:	*Juego de lucha, sistema de control*
Video demostrativo:	*https://youtu.be/W60_XQGRxwk*
Repositorio:	*https://github.com/Engineeringworkshop/TFM*
Ejecutable:	*Ejecutable del juego*

Resumen del Trabajo: *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo*

La finalidad de este trabajo es el desarrollar un videojuego para ordenador y consola. Para ello se expondrán todos los pasos necesarios para crearlo y poder publicarlo en un portal de distribución.

Los juegos de lucha suelen tener el mismo esquema de control. Obviando la espectacularidad de los ataques y centrándonos en los controles encontramos los mismos esquemas de control. Son pocas las excepciones que incluyan algún tipo de mini juego para poder usar los ataques más avanzados y poderosos. Droid Fighters entra en este punto, ofreciendo un sistema de control en tiempo real que permite personalizar de forma precisa los ataques. El jugador será recompensado en función de lo preciso que sea en sus movimientos haciendo que el jugador se sienta poderoso.

El juego estará realizado en Unity, se reutilizarán elementos disponibles en los diferentes servicios web para realizar el prototipo que permitirá a este trabajo convertirse en un videojuego terminado y publicado.

El proceso de prueba por diferentes perfiles de jugador es parte esencial de este proyecto. El ajuste de los controles con el feedback de los probadores hará que el juego esté equilibrado en dificultad para el público objetivo.

Abstract:

The purpose of this work is to develop a video game for computer and console. For this, all the necessary steps will be exposed to create it and be able to publish it in a distribution portal.

Fighting games often have the same control scheme. Obviating the spectacularity of the attacks and focusing on the controls, we find the same control scheme. There are few exceptions that include some kind of mini game to be able to use the most advanced and powerful attacks. Droid Fighters enters this point, offering a real-time control system that allows you to customize precisely your attacks. The player will be rewarded based on how precise they are in their movements making the player feel powerful.

The game will be made in Unity, elements available in the different web services will be used to make the prototype. That will allow this work to become a finished and published video game.

The testing process for different player profiles is an essential part of this project. Adjusting the controls with feedback from testers will make the game balanced in difficulty for the target audience.

Agradecimientos

Quiero mostrar mi gratitud a todas aquellas personas que estuvieron presentes en la realización de esta meta, de este sueño que es tan importante para mí, agradecer todas sus ayudas, sus palabras motivadoras, sus conocimientos, sus consejos y su dedicación.
A mis compañeros, con quienes a través del tiempo fuimos fortaleciendo una amistad, muchas gracias por toda su colaboración, por convivir todo este tiempo conmigo, por compartir experiencias, alegrías, frustraciones, llantos, tristezas, celebraciones y múltiples factores que ayudaron a que hoy seamos como una familia, por aportarme confianza y por crecer juntos en este proyecto, muchas gracias.
Agradecimientos a la UOC por la formación obtenida sin la cual este proyecto no sería posible.
Por último, quiero mostrar mi agradecimiento a la base de todo, a mi familia, en especial a mis padres, quienes con sus consejos fueron el motor de arranque y mi constante motivación, muchas gracias por su paciencia y comprensión.

Índice

Figuras y tablas

Índice de figuras

Índice de tablas

1. Introducción

1.1. Introducción/Prefacio

El tema a abordar es crear un sistema de control dinámico que permita mejorar la experiencia de usuario en el género de los juegos de lucha. Aunque los juegos de lucha son un gran clásico en el mundo de los video juegos sus sistemas de control tienden a seguir un estándar. Esto dificulta el poder irrumpir en un mercado tan bien provisto de títulos, especialmente para un desarrollo con un presupuesto tan reducido como el de un proyecto de final de master. Para poder sobresalir en este mercado hay que aportar algo llamativo e innovador, en este proyecto es el sistema de control. Gracias a este sistema podrá proporcionar una gratificante opción a los jugadores al poder optar por esta propuesta en lugar del de una opción convencional. Esta propuesta atraerá a jugadores meramente veteranos por su novedad y esto abrirá la puerta a una mejor visibilidad en el mundo de los videojuegos atrayendo más jugadores.

1.2. Descripción/Definición

El proyecto parte del objetivo de crear e implementar un nuevo sistema de control. Para ello se ha creado un videojuego de lucha para poder desarrollar y probar la propuesta. Ello permitirá llenar un vacío en los juegos de lucha aportando frescura al género.

Una mecánica nueva, en este caso un sistema de control, siempre es interesante porque permite a un videojuego irrumpir en el mercado ofreciendo una novedad nunca vista antes, generando su propio mercado atrayendo a jugadores del género que buscan algo novedoso, fresco y divertido y jugadores de otros mercados que pueden verse atraídos hacia los juegos de lucha.

Actualmente, para llamar la atención de los jugadores, los estudios tienen que crear gráficos cada vez más espectaculares, modelos cada vez más realistas y grandes campañas de publicidad. Cosa que está fuera del alcance de la mayoría de estudios.

Se quiere crear un videojuego de lucha con todas las mecánicas habituales del género y añadirle el sistema de control dinámico para aportar una novedad a un mercado muy explotado. Para ello este proyecto pretende conseguir un prototipo de videjuego lo suficientemente avanzado como para ser publicado como en estado de acceso anticipado (early acces).

1.3. Objetivos generales

1- Definir las mecánicas del juego. Definir las mecánicas que tendrá la versión final del juego. Definirlas antes de implementarlas ayudará a agilizar el proceso y clarificar el objetivo.
2- Implementar las mecánicas en un prototipo jugable. Para poder hacer probar y ajustar las mecánicas del juego, y más concretamente el sistema de control, es necesario tener un entorno de pruebas. Para ello se creará directamente un prototipo jugable donde se añadirá el sistema de control y se probará en condiciones reales.
3- Ajustar las mecánicas. Una vez implementadas las mecánicas hay que ajustarlas. Mediante un proceso iterativo se probará todo el juego en conjunto y se irán corrigiendo los fallos.
4- Mejora de la demo. Para terminar el prototipo se le añadirán decoraciones, música, y efectos para ayudar a definir el tipo de juego y por donde seguirá su desarrollo.
5- Pruebas. Antes de poder ser publicado tiene que ser probado por jugadores de diferentes espectros para poder analizar la experiencia de usuario y decidir si lanzar el juego o si hay que hacer algún cambio. De ser necesario modificar algo se volvería a la fase de implementación, después a la de ajuste y se tendría que pasar por una nueva ronda de pruebas.
6- Publicar. Una vez superadas las fases iterativas de implementación, ajuste y pruebas el producto estará listo para ser publicado en las plataformas de distribución.

1.3.1. Objetivos principales

Objetivos de la aplicación:

- El juego debe ser completamente jugable. Para poder ser publicado, el juego tiene que estar en un estado jugable, con todas las mecánicas core implementadas y ajustadas. También tiene que contar con un mínimo de trabajo artístico que permita transmitir la atmosfera que se pretende conseguir con el juego en concreto, no solo el sistema de mecánicas.
- El nuevo sistema de control debe estar implementado y funcionar acorde a las especificaciones. Sin este sistema, el juego solo sería un juego indie de ajo presupuesto más, no una alternativa emergente y prometedora.

Objetivos para el usuario:

- Proporcionar una nueva forma de control para el jugador y el género. Ofrecer una novedad en un género clásico siempre es positivo para la industria y los jugadores. El objetivo no es solo crear el sistema de control si no desarrollarlo para que sea atractivo, divertido e intuitivo.
- El nuevo sistema de control debe producir una experiencia satisfactoria.
- Que sienta que el clásico genero de los juegos de lucha sea renovado.

Objetivos personales:

- Crear un juego en el estado de acceso anticipado que implemente el sistema de control propuesto. Poder llegar al punto de publicación es un objetivo ambicioso pero necesario para que el juego tenga un futuro. Por otro lado, el objetivo de final de master no puede ser otro que tener un prototipo publicable con el que irrumpir en el mercado de videojuegos directamente o buscar hacerlo a través de un Publisher.

1.3.2. Objetivos secundarios

- Implementar el sistema de control y el feedback en pantalla de una forma orgánica y natural para mantener la inmersión en el juego.
- Conseguir un juego divertido y adictivo que esté acorde a su precio.
- Desarrollar el aspecto estético del juego lo máximo posible antes del lanzamiento. Aunque una vez lanzado sea aún un proyecto en desarrollo es necesario que el jugador se haga una idea de la ambientación y la atmosfera del juego.

1.4. Metodología y proceso de trabajo

La metodología seguida para realizar este trabajo fue la siguiente. Primero se empezó estableciendo el concepto de juego, un juego de lucha de robots. A continuación, se eligieron los modos de juego y sus mecánicas. Después, las mecánicas de lucha para los combates. Establecidas las bases del juego se procedió a crear un prototipo jugable al que se añadieron los modelos básicos de los primeros luchadores. Posteriormente se diseñó la decoración del escenario, aunque este paso pudiera realizarse en un estado más avanzado del proyecto se incluyó en este punto con propósitos ambientales para ayudar a establecer el concepto y poder armonizar el diseño general del juego.

Una posible alternativa a este desarrollo es coger un juego existente y añadirle esta mecánica. Con esto se podría invertir mucho más tiempo en el desarrollo de la mecánica de combate. Lo malo de esta opción es que no acabaríamos con un producto publicable. Aunque puede ser un objetivo válido, el cual te puede abrir las puertas en la industria para trabajar en un estudio importante, no sería tan beneficiosa en caso de querer fundar un estudio, en cuyo caso, un prototipo jugable es mucha mejor opción para atraer a un Publisher o ser financiado directamente por la comunidad.

Para conseguir el objetivo se ha creado la mayor parte del código del juego, y los personajes se han diseñado, creado y animado. Pero para agilizar el proceso, se han utilizado recursos de diferentes autores como decoraciones, música y sonidos.

1.5. Planificación

1.5.1. Fechas clave:

Entrega de la PEC 1 el 19/3/2023:
En la primera entrega se definirá la temática del juego y se creará la ficha del juego y un boceto de la planificación.

Entrega de la PEC 2 el 16/4/2023:
Para la entrega de la PEC 2 se corregirá la planificación, se expondrá el estado del arte, se realizará un estudio de mercado, se hará una propuesta de proyecto y se definirá la estrategia de comercialización.

Entrega de la PEC 3 el 21/5/2023:
En esta entrega se procederá al diseño del juego, su implementación y se grabará un video de demostración del mismo.

Entrega de la PEC 4 el 18/6/2023:
En la cuarta entrega se implementarán las correcciones de la PEC3 y se expondrán las conclusiones y las líneas de futuro, así como se grabará un tráiler y se hará la presentación.

Exposición del 26-6-23 al 2-7-23:
Se realizará la defensa del trabajo

1.5.2. Hitos

1- Definición de la temática del juego
2- Ficha del juego
3- Planificación
4- Entrega de la PEC 1
5- Estado del arte
6- Estudio de mercado
7- Propuesta de proyecto
8- Estrategia de comercialización
9- Entrega de la PEC 2
10- Diseño de juego
11- Implementación
12- Video demostrativo
13- Entrega de la PEC 3

14- Conclusiones
15- Líneas de futuro
16- Tráiler
17- Entrega de la PEC4
18- Defensa

1.5.3. Diagrama de Gantt

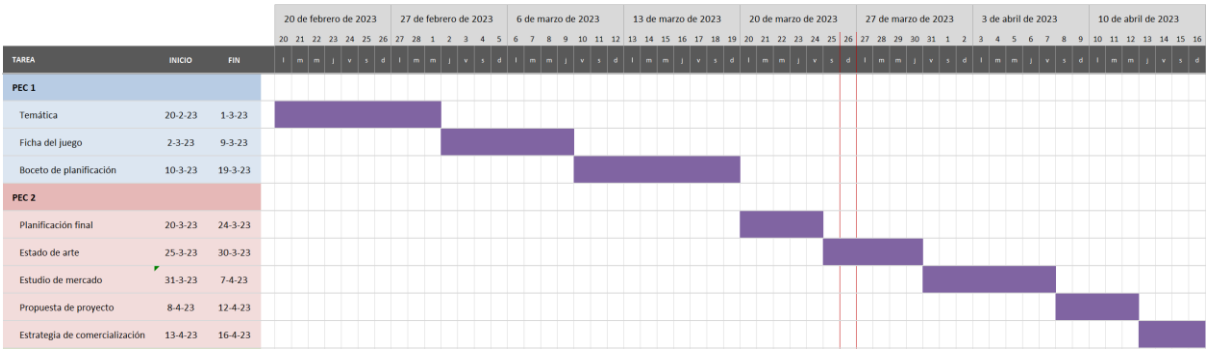

Figura 1: Diagrama de Gantt parte 1.

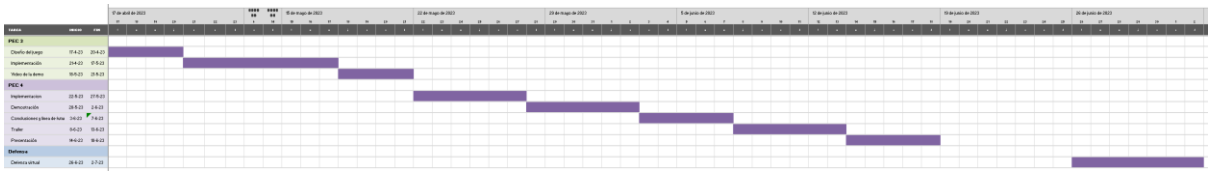

Figura 2: Diagrama de Gantt parte 2.

1.6. Presupuesto

1.6.1. Presupuesto del proyecto

A continuación, se desglosan los costes asociados al desarrollo del proyecto hasta el estado publicable de acceso anticipado. En este presupuesto, al tratarse de una obra de un solo autor, se considera que un solo trabajador realiza el proyecto durante el tiempo establecido. El autor necesitará un equipo para concebir y crear el videojuego. Por último, se consideran los constes de una pequeña campaña de marketing.

Mano de obra

El proyecto se inicia en 20/2/2023 y finaliza el 18/6/2023. Lo que significan 180 días para realizar el proyecto. Considerando solo los días laborales tenemos 85 días hábiles. Estimando un coste de 35€/hora resulta en un gasto de 23.800€.

Concepto	Coste unitario	Cantidad	Total
Diseñador y programador	35 €/h	85	23.800€

Tabla 1: Coste de mano de obra del proyecto.

Equipamiento técnico

El equipamiento técnico utilizado es un ordenador personal de sobremesa con sus periféricos. El coste total del equipo es de 2.350€

Concepto	Coste unitario	Cantidad	Total
Torre	1.500€	1	1.500€
Monitor	350€	2	700€
Auriculares	35€	1	35€
Teclado	30€	1	30€
Ratón	25€	1	25€
Mando	60€	1	60€
		Total	2.350€

Tabla 2: Coste de equipo del proyecto.

Marketing

Para promocionar el juego se realizará una pequeña campaña de marketing. En esta campaña se combinarán gameplays promocionales junto con claves del juego para sorteos. Se contactarán a diversos creadores de contenido para que jueguen en directo y/o suban videos jugando al video juego. Para mejorar la distribución y reducir el coste de los videos promocionales se regalarán claves del producto para que los creadores puedan sortearlos en sus canales.

Concepto	Coste unitario	Cantidad	Total
Gameplays promocionales	800€	10	8.000€
Claves de juego	0€	50	0€
		Total	8.000€

Tabla 3: Coste de marketing del proyecto.

Coste del proyecto

A modo de resumen se muestra la tabla con el coste de cada apartado y el total.

Concepto	Total
Mano de obra	23.800€
Equipo	2.350€
Marketing	8.000€
	34.150€

Tabla 4: Coste total del proyecto.

1.6.2. Presupuesto del desarrollo completo

Una vez lanzado en acceso anticipado empieza una nueva etapa que escapa del ámbito de este proyecto. En esta segunda fase se procederá a expandir el videojuego creando un pequeño estudio con distintos perfiles de empleados. El equipo constará de un diseñador y productor, un artista y un programador para mantener y evolucionar el código del juego.

Mano de obra

La mano de obra la formarán 3 personas. El diseñador y productor, encargado de diseñar las mecánicas y gestionar la cohesión de todo el proyecto. El artista, responsable de crear el arte del juego, diseñar los personajes y crearlos. Y por último el programador, se encargará de programar las mecánicas que se añadirán al juego, mantenimiento del código y creación de herramientas para facilitar el trabajo al artista y al diseñador.

Concepto	Coste unitario	Cantidad	Total
Diseñador y productor	40€/h	1788h/año	71.520€
Artista	35€/h	1788h/año	62.580€
Programador	35€/h	1788h/año	62.580€
		Total	196.680€

Tabla 5: Coste de mano de obra del proyecto futuro.

Equipamiento técnico

Cada uno de los trabajadores necesitará su equipo.

Concepto	Coste unitario	Cantidad	Total
Equipo	2.350€	3	7.050€

Tabla 6: Coste de equipos del proyecto futuro.

Marketing

La campaña de marketing se ampliará, pero se mantendrá la filosofía de distribución contratando generadores de contenido.

Concepto	Coste unitario	Cantidad	Total
Gameplays promocionales	800€	20	16.000€
Claves de juego	0€	150	0€
		Total	16.000€

Tabla 7: Coste de marketing del proyecto futuro.

Coste del proyecto

A modo de resumen se muestra la tabla con el coste de cada apartado y el total.

Concepto	Total
Mano de obra	196.680€
Equipo	7.050€
Marketing	16.000€
	219.730€

Tabla 8: Coste total del proyecto futuro.

1.7. Estructura del resto del documento

A continuación, se explicará brevemente el contenido de los siguientes capítulos.

Capítulo 2 Análisis de mercado: En este capítulo se definen el público objetivo, se analizarán los competidores actuales y juegos anteriores.
Capítulo 3 Propuesta: Aquí se muestran los objetivos y las especificaciones del producto
Capítulo 4 Diseño: En esta parte se desgrana la estructura del juego con la estructura de datos y lenguajes de programación utilizados.
Capítulo 5 Implementación: Aquí se recogen los requisitos y las instrucciones de instalación
Capítulo 6 Demostración: Contiene las instrucciones de uso, prototipos y las pruebas realizadas.
Capítulo 7 Conclusiones y líneas de futuro: Capítulo final con las conclusiones del proyecto y las futuras líneas de desarrollo.

2. Estado del arte

La masificación de las computadoras, después de que se crearan durante la segunda guerra mundial, permitió que los ordenadores personales estuvieran disponibles para el gran público. Con ello, nacieron los primeros videojuegos, no obstante, no fue hasta la década de los 70 cuando empezaron a comercializarse, inicialmente, con las máquinas recreativas. Uno de los primeros géneros que parecieron con estas máquinas fue el género de los juegos de lucha.
En este punto, los juegos de lucha ya desarrollan sus características más reconocibles. Dos personajes en un ring, cada uno a un lado de la pantalla y controlables por los jugadores, en el caso de dos jugadores, o solo uno en el caso de un jugador. Los jugadores deben usar los movimientos, ataques disponibles y combos para derrotar al otro jugador.

En este capítulo se analizan primero los principales motores gráficos para crear videojuegos hoy en día, se expondrá el género de juegos de lucha, se profundizará en los sistemas de control de este género, se definirá el público objetivo y por último se repasarán los juegos más icónicos del genero de lucha.

2.1. Motores de videojuegos.

Los motores gráficos modernos permiten acercar al público general la posibilidad de desarrollar su propio videojuego. La complejidad del desarrollo de videojuegos ha ido descendiendo según han evolucionado los lenguajes de programación. Hoy en día existen aplicaciones de alto nivel con las que cualquiera puede crear su propio videojuego. Algunas de las herramientas más importantes hoy en día son: UbiArt, Cryengine, Source engine, GameMaker o Stencly. Pero hay dos motores que son mucho más utilizados que los anteriores: UnrealEngine y Unity.

2.1.1. Unreal Engine

Figura 3: Logotipo de Unreal Engine 4.

Unreal Engine (Ver: Figura 3) Desarrollado por Epic Games en la década de los 90 y lanzado oficialmente en 1998 es uno de los gigantes del mercado. Ha evolucionado mucho desde entonces,

inicialmente concebido para un tipo específico de juegos, los shooters, ha ido expandiendo hasta poder ser utilizado para cualquier tipo de videojuego o cualquier necesidad que pueda tener un usuario.

La versión más actual, en el momento de redactar esta documentación, es Unreal Engine 4 (ver: Figura 4). Usa C++ como lenguaje del propio motor y para crear los video juegos. Esto permite que no solo puedas crear tu videojuego con tu código, si no modificar el código interno de Unreal para adaptarlo a tus necesidades.

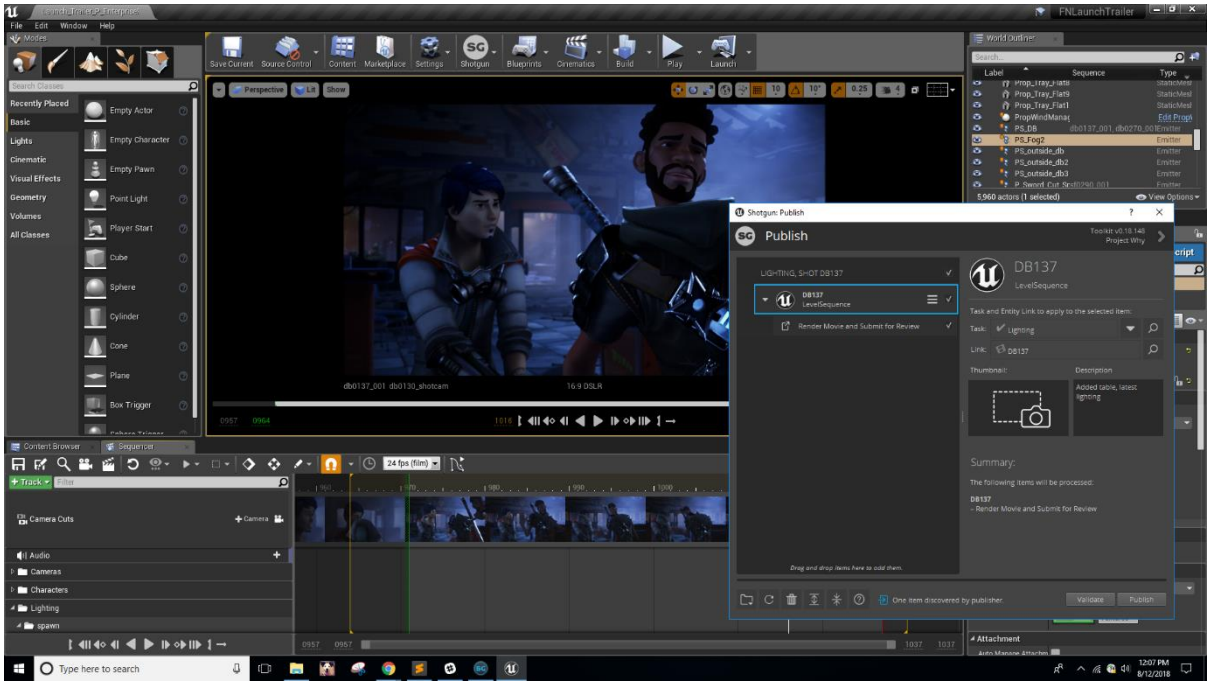

Figura 4: Entorno de Unreal Engine 4.

Inicialmente fue de pago, pero hace tiempo que decidieron ofrecerlo de forma gratuita, teniendo solo que pagar una comisión en caso de obtener beneficios con los juegos que se realicen con él.

Algunos ejemplo de videojuegos que utilizan Unreal Engine son Fornite de Epic Games, Kindom hearts III, Aftermath (ver: Figura 5) o la saga de ARK.

Figura 5: Imagen del gameplay correspondiente a Aftermath.

2.1.2. Unity

Figura 6: Logotipo de Unity.

Unity (ver: Figura 6) Ofrecido desde 2005 por Unity Technologies ha ido evolucionando igualmente, inicialmente tenía dos versiones, una gratuita y otra de pago con más funcionalidades. Desde la versión 14 la versión gratuita ha dejado tener esas limitaciones exceptuando aquellas que requieren servicios online: trabajo conjunto o almacenamiento en la nube entre otras. En la Figura 7 se puede ver el entorno de trabajo e Unity.

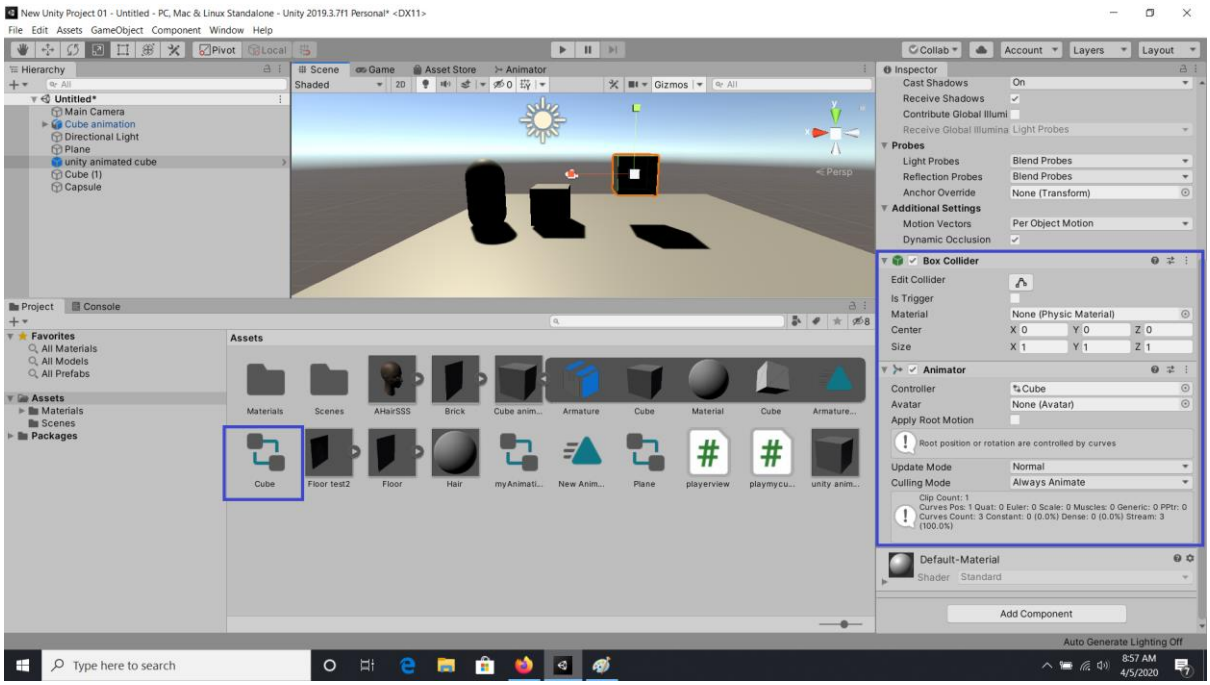

Figura 7: Entorno de Unity.

Destaca por su facilidad y velocidad para aprender y crear escenas. Usa C# o JavaScript como lenguajes de programación. Tiene soporte de para VR.

Algunos títulos hechos con Unity: Cuphead, Aubnautica, Rust (ver: Figura 8).

Figura 8: Imagen del gameplay de Rust.

2.2. Genero de lucha

3.2 Género “Victory Royale” es un videojuego del género conocido generalmente como “fighting” o, en castellano, de peleas, un género englobado dentro de los videojuegos de deportes y de estilo “arcade”. Este género comparte una serie de características en todos sus juegos. Algunas de estas características, son su movimiento 2D o la posición de las barras de vida de los personajes siempre dispuestas en la zona superior.

2.3. Sistemas de control para videojuegos de lucha

2.3.1. Mecánicas base o de “core”

Esta sección está basada en el análisis del artículo (Iyer, 2021). En este artículo analiza las mecánicas que deberá implementar un juego de lucha.

Condición de victoria

Las sesiones de juegos de lucha consisten en partidas y las partidas consisten en rondas. Dado que los juegos de lucha son juegos de competición, un combate debe tener un ganador y un perdedor. Debido a que los enfrentamientos deben concluir, la cantidad de rondas es finita: a partida única, el mejor de 3, 5, o 7.

El propósito de una ronda es derrotar al oponente. Para hacer esto, los jugadores deben agotar un recurso finito que representa la capacidad de juego del personaje del oponente. El nombre de este recurso puede variar de un juego a otro, pero a menudo se llama salud y se indica con HP para puntos de salud. También se le puede llamar energía. Un elemento de visualización llamado barras de vida muestra este recurso en pantalla, es decir, una interface no diegética. Dado que el resultado de la ronda gira en torno a los valores de estos recursos, las barras de vida se muestran de forma destacada en la parte superior de la pantalla (ver: Figura 9).

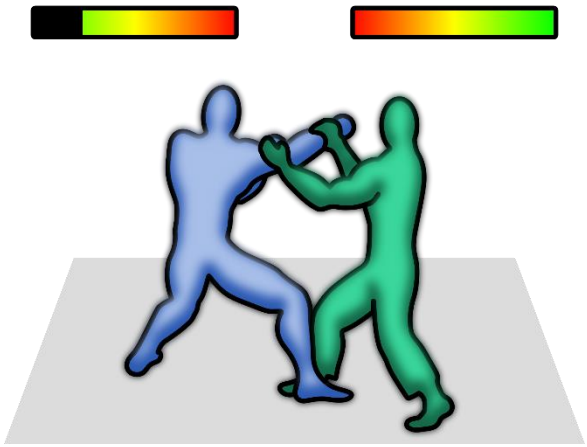

Figura 9: Esquema básico de un juego de lucha.

Algunos juegos complementan (o incluso renuncian) a las barras de vida mostrando estas estadísticas en el propio personaje, interface diegética. Éstas incluyen:

- Daños corporales como hematomas y cortes en humanos
- Eliminación de armaduras o apéndices
- Efectos visuales especiales como chispas o sangrado

Cuando el personaje de un jugador se queda sin HP, este ya no puede luchar y ha perdido la Ronda. Pero, si todos los oponentes del jugador pierden todos los HP, el jugador gana la ronda.

En ocasiones las partidas pueden tener límites de tiempo, los Jugadores deben ganar la ronda antes de que se agote el tiempo. Si el temporizador expira antes de que cualquier jugador gane una ronda, el jugador con más HP gana la ronda.

A veces es posible que todos los jugadores se queden sin HP al mismo tiempo o que tengan la misma cantidad de HP cuando se acaba el tiempo. Los empates se pueden resolver de varias maneras:

- Otorga la ronda a todos los jugadores
- Otorga la ronda al azar a un jugador
- Anular y reiniciar la ronda
- Participa en una ronda especial de desempate

Los desempates pueden tener reglas diferentes, pero una común es la muerte súbita. En muerte súbita, el jugador que causa el primer agotamiento de recursos (de cualquier cantidad) gana la Ronda.

Logrando la victoria

Si un jugador quiere ganar, todos sus oponentes deben perder el recurso que les permite luchar. Esto se puede hacer conceptualmente de tres maneras.

Agresor

La forma más común de reducir HP es realizar un ataque al oponente. El jugador usa los controles para que el personaje ataque. Si el ataque del personaje no logra impactar al personaje del oponente, es un fallo y no se pierde HP. Si el ataque tiene éxito, el personaje opuesto pierde una cantidad predeterminada de HP. Si se realizan suficientes ataques exitosos, el jugador puede agotar por completo el HP de su oponente y ganar la ronda. Pero la mayoría de los ataques conllevan cierto costo y riesgo, por lo que los jugadores deben evaluar y ejecutar según sea necesario.

Contraataque

Los ataques inoportunos pueden colocar a los personajes en situaciones en las que son más susceptibles de sufrir los ataques de un oponente. Lo mejor para el jugador es permitir que sus oponentes alcancen ese estado. Esto se logra al permitir que los oponentes ataquen mientras el jugador esté en un estado defensivo. Esto puede llamarse defensa o bloqueo. Mientras bloquea un ataque, un jugador puede evitar la pérdida de HP; si pierden HP, a menudo es una fracción de lo que hubieran perdido si no estuvieran defendiendo. La compensación es que puede crear una ventana de oportunidad para poderosos contraataques, el jugador puede aprovechar a atacar mientras el oponente se recupera de su acción anterior. Algunos juegos también cuentan con técnicas de evasión y parada para lograr lo mismo.

Desplazamiento

El posicionamiento de los personajes es crucial. A menudo, determina si los ataques impactarán en el objetivo. Pero en los juegos de lucha donde los escenarios (arenas donde ocurren las peleas) tienen elementos que pueden afectar el HP, como, por ejemplo, algunos escenarios contienen peligros como púas o cercas eléctricas. Al obligar a los oponentes a entrar en esos peligros, puede hacer que pierdan HP. En algunos juegos, el personaje también se puede eliminar del escenario si está demasiado cerca de sus límites. En otros juegos, cualquier desplazamiento del personaje fuera del escenario reduce inmediatamente su HP a cero, lo que da como resultado una condición llamada salida del ring, llamada en ingles “Ring Out”.

Tiempo y espacio

No debería sorprender que el tiempo y el posicionamiento sean elementos críticos de los juegos de lucha. Por ejemplo, la mayoría de los ataques tienen una fase de inicio distinta llamada “Windup”.

Este es el período en el que el personaje se prepara para ejecutar un ataque. Los ataques que tienen un “Windup” largo generalmente causan una gran pérdida de HP, al contrario de los que tienen un tiempo de preparación más corto. En contra partida, los ataques con largo tiempo de preparación son propensos a ser interrumpidos por ataques con “Windups” más cortos. Los ataques más rápidos pueden causar menos pérdida de HP al oponente, pero con una sincronización perfecta, pueden interrumpir un ataque entrante más fuerte. Ser bueno en los juegos de lucha significa estar en el lugar correcto, en el momento correcto, haciendo el movimiento correcto.

Conclusión

El concepto central de los juegos de lucha es simple: agotar la capacidad de lucha de tu oponente. Pero la profundidad y la estrategia involucradas en hacer esto son inmensas. Es fácil de aprender, pero difícil de dominar.

2.3.2. Diseño de juego

Aquí se analizan otras mecánicas que puede o debe incluir un juego de lucha.

Características

Además de moverse por un espacio restringido, los juegos de lucha limitan las acciones del jugador a diferentes maniobras ofensivas y defensivas. Los jugadores deben aprender qué ataques y defensas son efectivos entre sí, ya sea mediante prueba y error o comunicándose con otros jugadores fuera del juego. (Rollings & Adams, 2006) El bloqueo es una técnica básica que permite a un jugador defenderse de ataques básicos. (Leone, 2009) Algunos juegos presentan técnicas de bloqueo más avanzadas: por ejemplo, Street Fighter III de Capcom presenta un movimiento denominado "parar", que hace que el atacante detenido quede momentáneamente incapacitado. Un estado similar se denomina "recién defendido" en Garou: Mark of the Wolves de SNK. (Gerstmann, 1999) (Chau, 2001)

Ataques especiales y combos.

Una característica integral de los juegos de lucha es el uso de "ataques especiales", también llamados "movimientos secretos", (Parish, 2009) que emplean combinaciones de entradas direccionales y pulsaciones de botones para realizar un movimiento particular más allá de los puñetazos y patadas básicos. (Towell, 2009) Algunos movimientos especiales, que reproducen una animación que representa un aspecto de la personalidad del personaje, se conocen como burlas. Presentados originalmente por la empresa japonesa SNK en su juego Art of Fighting, (Park, 2007) se utilizan para agregar humor, pero también tienen un efecto en el juego en ciertos juegos, como mejorar la fuerza de otros ataques. (Rose, 2008) A veces, un personaje puede incluso destacarse

especialmente por sus burlas (por ejemplo, Dan Hibiki de Street Fighter Alpha). (Daily, Top 20 Street Fighter Characters of All Time, 2009) (Daily, Top 25 Most Bizarre Fighting Characters, 2009) Super Smash Bros. Brawl introdujo un nuevo ataque especial que es exclusivo de la serie Super Smash Bros, conocido como Final Smash.

Los combos, en los que se encadenan varios ataques, son otra característica común en los juegos de lucha y han sido fundamentales para el género desde el lanzamiento de Street Fighter II. La mayoría de los juegos de lucha muestran un "medidor combinado" que muestra el progreso del jugador a través de un combo. La efectividad de tales movimientos a menudo se relaciona con la dificultad de ejecución y el grado de riesgo. Estos movimientos, a menudo, son difíciles de realizar y requieren que el jugador tenga una buena memoria y una sincronización excelente. (Rollings & Adams, 2006).

Contraataque

Predecir los movimientos de los oponentes y contraatacar, conocido como "contraataque", es un elemento común del juego. (Treit, 2009) Los juegos de lucha también enfatizan la diferencia entre la altura de los golpes, que van desde ataques bajos hasta ataques de salto. (Parish, 2009) (Ekberg, 2007) Por lo tanto, la estrategia se vuelve importante a medida que los jugadores intentan predecir los movimientos de los demás, de forma similar a piedra, papel o tijera. (Rollings & Adams, 2006).

Agarre y derribo

Además de golpes como puñetazos y patadas, los jugadores pueden utilizar lanzamientos o agarres para eludir bloqueos. La mayoría de los juegos de lucha le dan al jugador la capacidad de ejecutar un movimiento de agarre presionando dos o más botones juntos, o simplemente presionando puñetazo o patada mientras está directamente adyacente al oponente. Otros juegos de lucha, como “Dead or Alive”, tienen un botón exclusivo para lanzamientos y derribos.

Proyectiles

Utilizados principalmente en juegos de lucha en 2D, los proyectiles son objetos que un luchador puede lanzar a otro luchador para atacar desde la distancia. Si bien pueden usarse simplemente para infligir daño, los proyectiles también se usan a menudo para maniobrar a los oponentes en posiciones desventajosas. El proyectil más notable es el “Hadouken” de Ryu y Ken de Street Fighter.

Elementos de juego emergentes

Tortugas y zonificación

En el mundo de los juegos de lucha, especialmente los de la variedad 2D, la zonificación se refiere al juego defensivo que se enfoca en usar ataques relativamente libres de riesgos para mantener alejado al jugador contrario. El resultado deseado de la zonificación es obligar a un oponente a tomar riesgos significativos para acercarse al personaje del jugador de zonificación, o detener el temporizador del juego, lo que hace que el jugador con más salud (generalmente el que realiza la zonificación) gane. La efectividad de la última estrategia varía de un juego a otro, según la efectividad de las herramientas de zonificación, así como la duración del temporizador en el juego y las recompensas que los personajes pueden recibir por conseguir contrarrestar la zonificación.

Precipitación

“Rushdown”, lo opuesto a las tortugas, se refiere a una serie de estrategias agresivas, filosofías y estilos de juego específicos en todos los juegos de lucha. El objetivo general de un estilo de juego apresurado es abrumar al oponente y forzar errores costosos, ya sea mediante el uso de configuraciones rápidas y confusas o aprovechando a un oponente impaciente que se ve obligado a jugar a la defensiva durante períodos prolongados de tiempo. Los jugadores de “Rushdown” a menudo prefieren atacar a los oponentes en la esquina de un escenario o cuando se levantan de una caída; ambas situaciones limitan severamente las opciones del oponente y, a menudo, permiten que el jugador atacante fuerce escenarios de tácticas de alto riesgo.

Espaciado y footsies

El espaciado es el acto de posicionar a un personaje en un rango donde sus ataques y herramientas de movimiento conllevan el riesgo más bajo y la recompensa más alta. El concepto es algo similar al del juego de pies en las artes marciales. La posición deseada para jugar varía según las herramientas disponibles para el personaje de cada jugador.

2.3.3. Controles simplificados

Análisis de videojuegos con controles simplificados. Traducción e interpretación del análisis de (Dhami, 2021). En el videojuego “DNF Duel” Los combos se sienten muy satisfactorios y de forma libre, especialmente con la mecánica de conversión que recuerda a “Tatsunoko vs. Capcom”, uno de los títulos anteriores de Eighting. Por otro lado, el estilo de movimiento de “Granblue Fantasy Versus”, combinado con la adición de un conjunto de botones con macros, hace que el enfoque y las mezclas se sientan lineales. Si bien todos los personajes en el juego se sienten absurdamente poderosos, los medios para sacar provecho de ese poder no son claros. Tampoco se puede decir realmente cuál es la diferencia entre ejecutar movimientos especiales con un solo botón comparado

con la entrada de movimiento adecuada, especialmente porque parece que recuperas toda la energía para atacar rápidamente.

GBVS también recibió críticas similares en su lanzamiento por incorporar mecánicas como especiales de un solo botón y un botón de bloqueo, y la reciente actualización del progreso del “Proyecto L” provocó discusiones controvertidas sobre la simplificación en los juegos de lucha. Superficialmente, estas conversaciones tienden a girar en torno a si el género está dirigido a jugadores más nuevos o inexpertos que, de lo contrario, se quejarían de entradas complicadas, y por qué esto es algo malo. Incluso fuera de juegos como “Project L”, “GBVS” y ahora “DNF Duel”, los argumentos sobre la simplificación plagan la discusión sobre los nuevos juegos cada vez que se lanzan, debido a las quejas de que el género pierde la profundidad que tenían los títulos anteriores. Es revelador que el argumento se haya trasladado a otros géneros.

No se consideran todos estos argumentos aquí. Se sabe por qué existen las entradas de movimiento, incluidos los movimientos de carga: la ejecución es una habilidad en sí misma, y controlar movimientos más fuertes detrás de una pequeña barrera de ejecución evita que se abuse de ellos mientras recompensa a los jugadores por practicar. Con eso en mente, veremos específicamente los caminos que toman algunos juegos cuando intentan "simplificar" su motor o introducir controles o mecánicas más fáciles en lugar de equilibrar la ejecución. También vamos a ver estos cambios y mecánicas dentro del ecosistema en el que existen, para comprender cómo se implementan, cómo se equilibran y qué tan efectivos son en su trabajo. En general, cualquier cambio que haga que los juegos de lucha como género sean más fáciles de ingresar es un bien neto: cualquier cosa que fomente las ventas y haga crecer la base de jugadores, incluso si no todos se quedan, mantiene vivo el género. Sin embargo, mientras que algunos juegos pueden incorporar con éxito estas opciones de diseño, otros requieren más concesiones para que funcionen.

Personajes amigables para principiantes

No se trata solo de personajes como shotoclones (Clon de un personaje original que tiene los mismos ataques, pero es más fácil de manejar) o aquellos con planes de juego más simples. También hay personajes que están diseñados deliberadamente para ser fáciles de controlar, presentando herramientas o comandos que requieren una ejecución significativamente menor que el resto de los jugadores disponibles. La iteración inicial de Jack-O en la serie “Guilty Gear Xrd” es un excelente ejemplo de esto. Jack-O' usa solo dos entradas de movimiento, una 236 y una SPD, las cuales están en sus alzas. Todo lo demás es una entrada 22X (para comandar sus casas y sirvientes) o un comando normal de un botón y una dirección asignado a Dust. Cuando se combina con lo fácil que es obtener combos simples usando sus Gatlings, Jack-O' es muy fácil de entender para los jugadores principiantes y solo tiene un comando excepcionalmente difícil en relación con el

resto. En la Figura 10 se puede ver el esquema de control de Jack-O'. Otros personajes similares en otros juegos de la saga son: Mai Natsume (cuyos comandos más difíciles son todas variaciones de las mismas entradas 214 y 236) en BlazBlue Central Fiction, y Giovanna (que solo tiene cuatro movimientos especiales en total) en Guilty Gear Strive. Si bien todos estos personajes llenan arquetipos específicos, seleccionarlos al aprender un nuevo juego puede ayudarlo a aclimatarse al sistema más rápido. Hay muy poco que aprender de inmediato, y todo lo que hay que aprender es fácil de aprender.

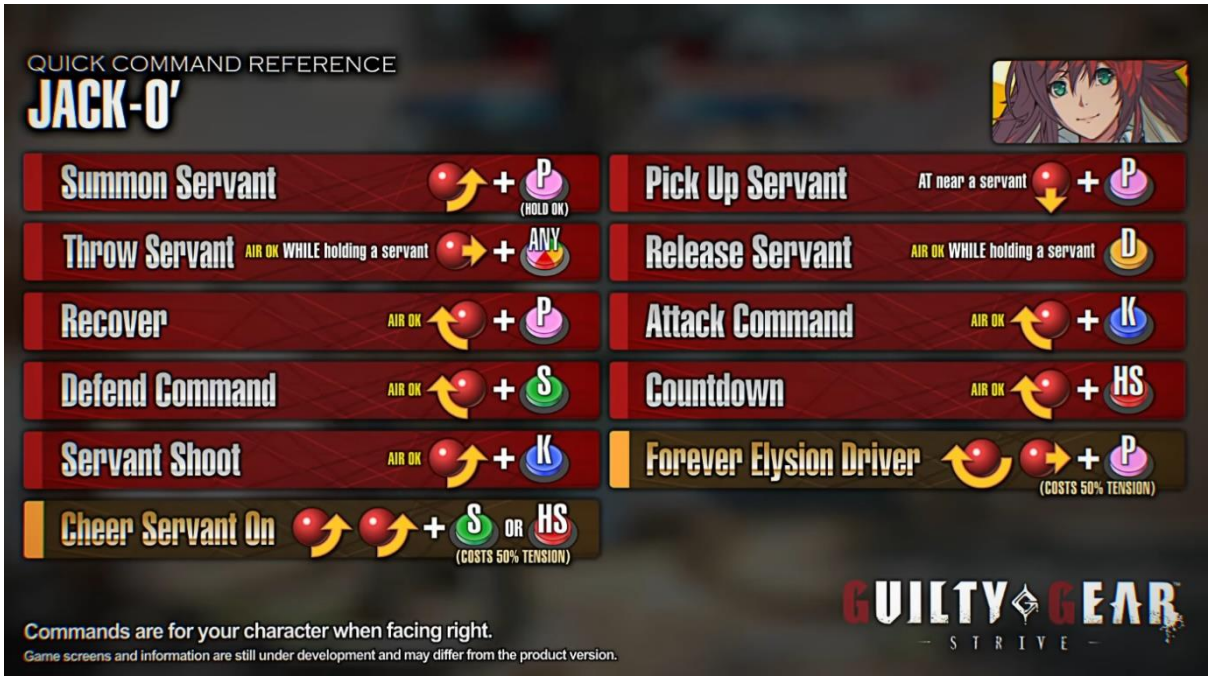

Figura 10: Lista de comandos de Jack-O' del juego Guilty Gear.

Entradas de movimiento especiales de una sola dirección

Es fácil olvidar que juegos como Super Smash Bros. técnicamente han estado usando un botón sin entradas de movimiento para sus movimientos especiales durante más de dos décadas. El concepto no es nada nuevo, y es fácil ver que la profundidad mecánica en los luchadores de plataforma no proviene del requisito de ejecución de la ofensiva o la duración de sus combos, sino de su movimiento y cómo se lleva a cabo la ofensiva. Este también es el caso de los luchadores tradicionales que implementan movimientos especiales de una sola dirección. Para equilibrar la facilidad con la que se pueden ejecutar estos poderosos movimientos, los propios movimientos deben diseñarse para evitar que dicho spam sea demasiado fuerte. Si bien hay excepciones obvias, como el infame Reflector 'brillante' en Melee, la mayoría de los movimientos especiales no son invencibles, se castigan con bloqueo, o solo son útiles en situaciones específicas, como volver al escenario.

Pokkén Tournament también tenía requisitos similares de baja ejecución para sus entradas de movimientos especiales e hizo concesiones de equilibrio similares. En Pokkén, ningún movimiento especial tiene invencibilidad en absoluto, incluso las inversiones solo tienen uno de los dos tipos de armadura. Los movimientos especiales generalmente también se encuentran dónde termina su cadena de bloqueo o combo, lo que los hace punibles en bloqueo con un espacio o tiempo inadecuado. Algunos personajes incluso tienen costos específicos adicionales para ejecutar sus movimientos especiales: Blaziken y Shadow Mewtwo, por ejemplo, gastan HP en ellos, mientras que Mewtwo gasta Synergy en los ataques. Del mismo modo, se implementó una compensación en el sistema de comando fácil de “GBVS”, donde los movimientos especiales realizados con el comando simple saldrían más rápido, pero entrarían en un "enfriamiento" donde no podrían ejecutarse nuevamente. Esto mantuvo los comandos de movimiento relevantes, haciéndolos más útiles en neutral y combos, al mismo tiempo que les daba a los jugadores experimentados razones para usar los comandos simples, como inversiones instantáneas. “Melty Blood Type Lumina” y “DNFD” implementarían sistemas inspirados directamente en los comandos de “GBVS”.

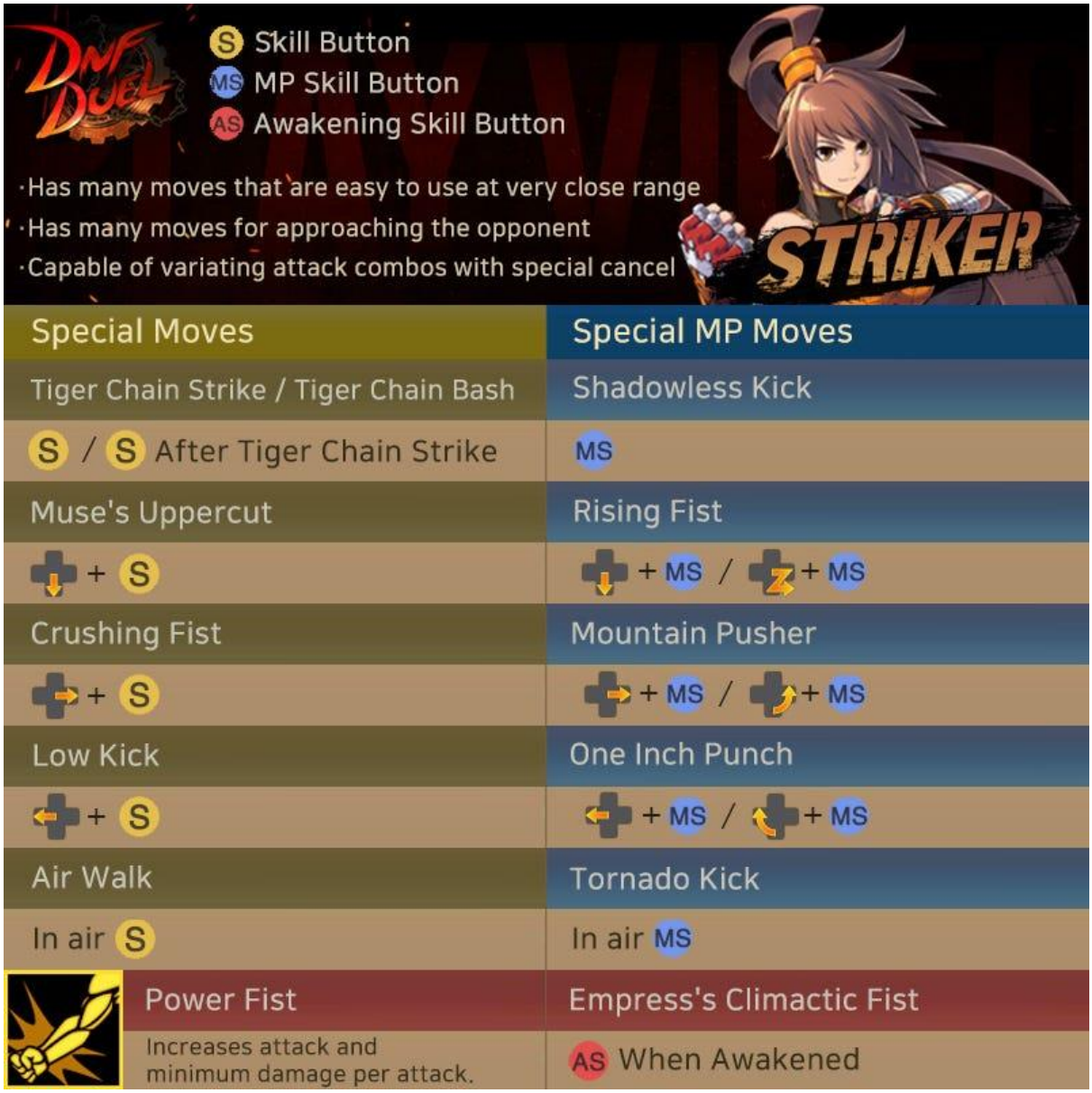

Figura 11: Esquema de control de movimientos especiales de DNFD.

“DNFD” tiene dos botones de movimiento especiales (Ver: Figura 11). El botón “S” presenta movimientos especiales de una dirección, y el botón “MS” tiene entradas de movimiento opcionales que le permiten reducir el costo de “PM” de esos movimientos.

Vale la pena señalar como un aparte que los sistemas con movimientos especiales de una dirección, especialmente aquellos con botones específicos que se usan solo para especiales, tienden a generar personajes con sets de control más pequeños en general. Así es como el diseño tiende a sacudirse, ya que en realidad solo puedes obtener entre cuatro y ocho direcciones vinculadas a un solo botón (potencialmente tan solo 3, ya que en la mayoría de los juegos hay que tocar 7–9 para saltar y los diseñadores no quieren que la entrada de salto se superponga con la entrada de ataque), mientras

que las entradas de movimiento se pueden asignar y reutilizar en varios botones normales para lograr diferentes resultados. Por supuesto, esto también se puede eludir con un diseño inteligente.

Macros y accesos directos del sistema

Se pueden hacer más fáciles de ejecutar otras funciones y comandos en los juegos de lucha asignando esas funciones a un botón o entrada más simple. Una de las macros más comunes en los juegos de lucha es la macro dash. Juegos como “Marvel vs. Capcom”, “Skullgirls”, “Under Night In-Birth” y “GGST” tienen macros de tablero donde se pueden presionar dos botones de ataque en lugar de tocar muchos más en la palanca, o un solo botón específico puede tomar el lugar del normal del dash. Si bien las macros de dash no se usan a menudo para el movimiento general (aunque pueden serlo, ya que son especialmente útiles para la accesibilidad), pueden hacer que los combos que requieren cancelar los movimientos de esquiva sean mucho más fáciles de realizar. Dado que puede pulsar o combinar la macro de botones para un dash, esto también significa que puede lo puede obtener en el fotograma más temprano posible, lo que a menudo lo hace incluso más óptimo que la ejecución normal. Eso ni siquiera es entrar en macros que te permiten avanzar mientras mantienes presionadas las entradas de carga en la palanca, como en “GGST”.

Una adición más reciente, como se vio en “GBVS” y “DNFD”, ha sido la inclusión de botones de bloqueo en juegos en los que ya puedes mantener o retroceder para bloquear. Si bien la aplicación principal para esto ha sido hacer que el bloqueo, especialmente el bloqueo de pie, sea más intuitivo para los jugadores menos familiarizados con el género, también tiene otras funciones. En “GBVS” y “DNFD”, por ejemplo, el botón de bloqueo tiene la característica más práctica de permitirle acceder a estados invulnerables temporales de rodar o esquivar que también pueden moverlo hacia adelante o hacia atrás. Por otro lado, poder mantener presionado un botón de bloqueo también le permite aprovechar una función que normalmente solo es posible mediante la manipulación de entradas SOCD: protección automática cruzada. Dado que mantener presionado un botón para bloquear significa que siempre estás bloqueando con el botón presionado, eso significa que tu personaje continuará bloqueando en la dirección correcta incluso cuando tu oponente salte sobre él o lo atraviese, por lo que las únicas situaciones de las que debes preocuparte son high-low y strike-throw. (Esto incluso bloquea la ofensiva de cambio lateral mientras bloquea un ataque persistente en el lado opuesto). Por supuesto, esto también depende de que mantenga presionado correctamente el botón de bloqueo en primer lugar, y aún debe bloquear alto y bajo apropiadamente.

Conceptos de equilibrio potencial

Aunque estas herramientas pueden ser útiles incluso para jugadores más experimentados, también crean un nuevo conjunto de desafíos para que los diseñadores de batalla y los jugadores los superen. Por ejemplo, como se mencionó anteriormente, el perjuicio de usar movimientos especiales unidireccionales en “DNFD” no parece superar los beneficios. Se siente como si fueras a perder muchas combinaciones por accidente, ya que es difícil recordar los controles. Siendo capaz de cambiar de dirección fácilmente o defender con un simple input o una macro significa que es fácil encontrar una respuesta a la situación. Si el oponente intentara una acción, simplemente la podrías evitar con una pulsación.

2.4. Público objetivo y perfiles de usuario

El público objetivo de un videojuego de lucha puede variar dependiendo del estilo del juego y del alcance de la marca. Sin embargo, generalmente se puede decir que su audiencia está compuesta por:

- Hombres jóvenes y adolescentes: Entre 15 y 34 años. Este género y grupo de edad ha sido históricamente el principal público de los videojuegos de lucha. A menudo les atrae la emoción de la lucha, así como el desafío de enfrentarse a oponentes difíciles.

- Fanáticos de juegos de lucha: esta audiencia suele ser fiel a la marca y busca nuevas experiencias de juego que satisfagan sus necesidades y expectativas.

- Personas que disfrutan de la competición: a menudo, los videojuegos de lucha se centran en la competición en línea o en torneos. Por lo tanto, muchos fanáticos de la competición pueden ser atraídos por este tipo de juegos.

- Jugadores casuales: aunque no son el público principal, los jugadores casuales también pueden disfrutar de estos juegos por su accesibilidad y diversión rápida.

- Adultos jóvenes: con el aumento de la popularidad de los deportes electrónicos, los videojuegos de lucha pueden atraer a adultos jóvenes que buscan una forma de participar en competiciones en línea.

- Personas interesadas en la cultura pop: algunos videojuegos de lucha incluyen personajes populares de películas, cómics y otros medios de entretenimiento. Por lo tanto, aquellos interesados en la cultura pop pueden sentir curiosidad por estos juegos.

En resumen, el público objetivo de un videojuego de lucha varía, pero suelen ser hombres jóvenes y adolescentes, fanáticos de juegos de lucha, personas competitivas, jugadores casuales, adultos jóvenes y fanáticos de la cultura pop.

2.5. Competencia

2.5.1. Street Fighter

“Street Fighter” es un videojuego perteneciente a la empresa Capcom lanzado en 1987 para máquinas recreativas (ver: Figura 12). Actualmente la franquicia sigue creciendo con títulos para múltiples plataformas, PC, Xbox, PlayStation y Nintendo manteniendo temática arcade.
El control de “Street Fighter” ha implementado un control de movimiento sencillo en todas sus versiones. Sin embargo, los ataques que pueden realizar los personajes son una larga lista de combinaciones de botones llamadas combos. Estos combos hacen que cada personaje sea único, tanto por el combo de su ataque más poderoso como por la animación del mismo. En la Figura 13 se puede ver el gameplay de “Street Fighter VI” en desarrollo en el momento de crear este documento.

Figura 12: Gameplay de Street Fighter I de 1987.

Figura 13: Gameplay de Street 6 (en desarrollo)

2.5.2. Mortal Kombat

“Mortal Kombat” es otra saga mítica en el género de lucha, iniciada por Midway Games en 1992 (ver: Figura 14). Empezó, al igual que “Street Fighter”, como un título para máquinas recreativas. No fue hasta la 4 entrega de la saga cuando los juegos fueron portados o desarrollados para consolas a través de la distibuidora Acclaim Entertainment. Desde 2009 la franquicia forma parte de Warner Bros. La última entrega de la saga “Mortal Kombat 12” (ver: Figura 15) será lanzada en 2023.

A día de hoy la saga sigue siendo fiel a sus orígenes conservando personajes y movimientos especiales. Otra característica más icónica de “Mortal Kombat” son los movimientos finales “Fatalities”. Este movimiento está disponible cuando un rival está casi derrotado y permite al jugador realizar un ataque especial para acabar con él de una forma extrema, cruel e irrespetuosa, con una animación única para cada personaje.

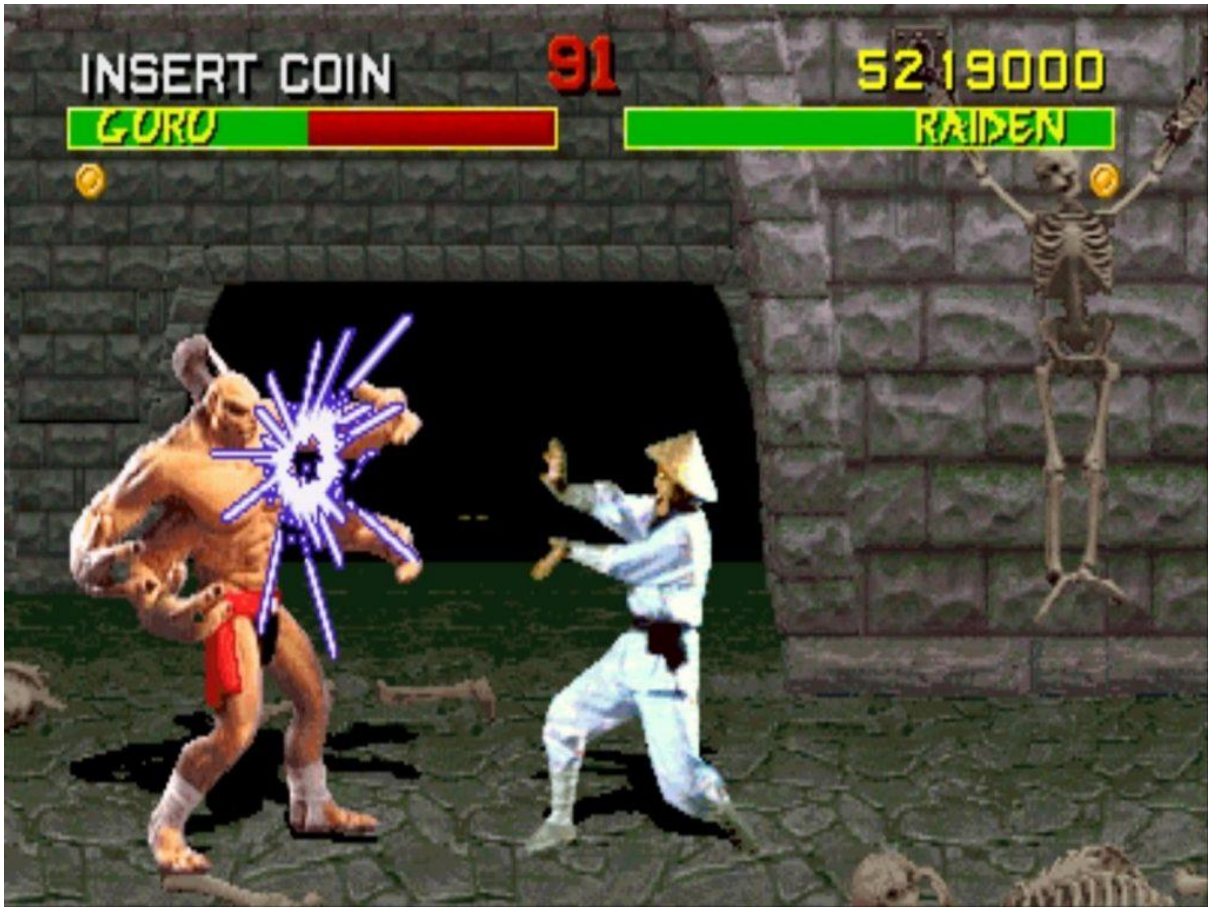

Figura 14: Gameplay de Mortal Kombat 1.

Figura 15: Gameplay de Mortal Kombat 13.

2.5.3. Dragon Ball

Otra saga icónica dentro del género de juegos de lucha es la saga de “Dragon ball”. Inspirada en el manga, esta saga de lucha tiene sus propias características que le hacen único.
Algunos de los títulos, como los “Budokai” permiten al jugador moverse en tres dimensiones, en lugar de las dos dimensiones de “Street Fighter”, “Mortal Kombat” y “Droid Fighters”. Pero salvo eso, comparte las características de los juegos anteriores, movimiento sencillo y complejos combos de ataque. En el caso de “Dragon Ball” los personajes tienen una infinidad de ataques, la mayoría únicos para cada personaje. La posibilidad de combinar esos ataques, el movimiento 3D y los cambios de forma de los personajes le hacen un juego de lucha bastante completo y complejo. En la Figura 16 se puede ver una captura del juego “Dragon Ball Budokai Tenkaichi 2” para la plataforma Play Station 2.

Figura 16: Gameplay de Dragon Ball Budokai Tenkaichi 2.

3. Propuesta

A partir del análisis de mercado hecho en el capítulo anterior, este tercer capítulo pretende explicar de manera resumida la propuesta del TF, haciendo énfasis en sus particularidades que lo diferencian de la competencia.

3.1. Definición de objetivos/especificaciones del producto

El objetivo final del proyecto es tener un videojuego de lucha completamente jugable y listo para ser publicado como acceso anticipado. El juego implementará un nuevo sistema de control para dotar al género de los juegos de lucha de frescura y dinamismo. El objetivo de este sistema de control es atraer nuevos jugadores al género mientras ofrece una jugailidad novedosa a los veteranos de los mismos.

El juego resultado de este proyecto será multiplataforma en consola y PC. El juego implementará multi-idioma, por las características de juego esto es muy sencillo puesto que no hay diálogos con audio que encarecerían el producto. En este caso, solo hay que traducir una corta lista de cadenas de caracteres.

3.2. Modelo de negocio

A corto plazo el modelo de negocio para este proyecto consiste en ofrecer la adquisición del videojuego mediante un pago único que proporcionará al jugador el acceso completo al juego. El pago único será de 9,5€. Este modelo es el inicial cuando salga al mercado al terminar el proyecto.

A largo plazo se ampliará el modelo de negocio, se mantendrá el pago único para acceder al videojuego, pero se añadirán dos opciones más. La primera de ellas es la posibilidad de adquirir la banda sonora del juego en las propias plataformas. Esto solo será posible en la segunda parte del proyecto, donde el equipo crecerá y se podrán destinar recursos a crear una banda sonora original. La segunda opción añadida consistirá en la posibilidad de comprar modelos y texturas (“skins” en inglés). Esto se implementará con micro pagos dentro del juego.

Según el artículo (Aevi, 2022), publicado por la Asociación Española de Videojuegos. En el 2021 se vendieron 6,9 millones de copias de juegos a 18 millones de usuarios en España. El articulo también expone el crecimiento del mercado de un 2,75% respecto al año anterior. En cuanto al tiempo que dedican los jugadores se concluye que dedican 8,1h semanales, por debajo, pero cerca de los países colindantes a España.

En el apartado 1.6.1 se han calculado los cotes del proyecto. El resultado ha sido un coste de 34.150€. Considerando un margen de ganancia de 20% tenemos que ingresar un total de 40.890€. Lo que significa que a un precio de 9,5€ deberíamos vender 4.314 copias del videojuego.

Para el proyecto completo se ha concluido en el apartado 1.6.2 se ha concluido un gasto de 219.730€. Considerando el mismo margen de ganancia del 20% y precio de 9,5€ tendríamos el objetivo de vender 27.756 copias al año para poder mantener el equipo.

La distribución de los jugadores por edades y género en España 2021, de acuerdo con los datos estadísticos del portal Statista por (Orús, 2022) (ver: Figura 17) podemos ver que para el rango de edades de 15 a 34 años, el volumen de mercado representa un 26% del mercado total.

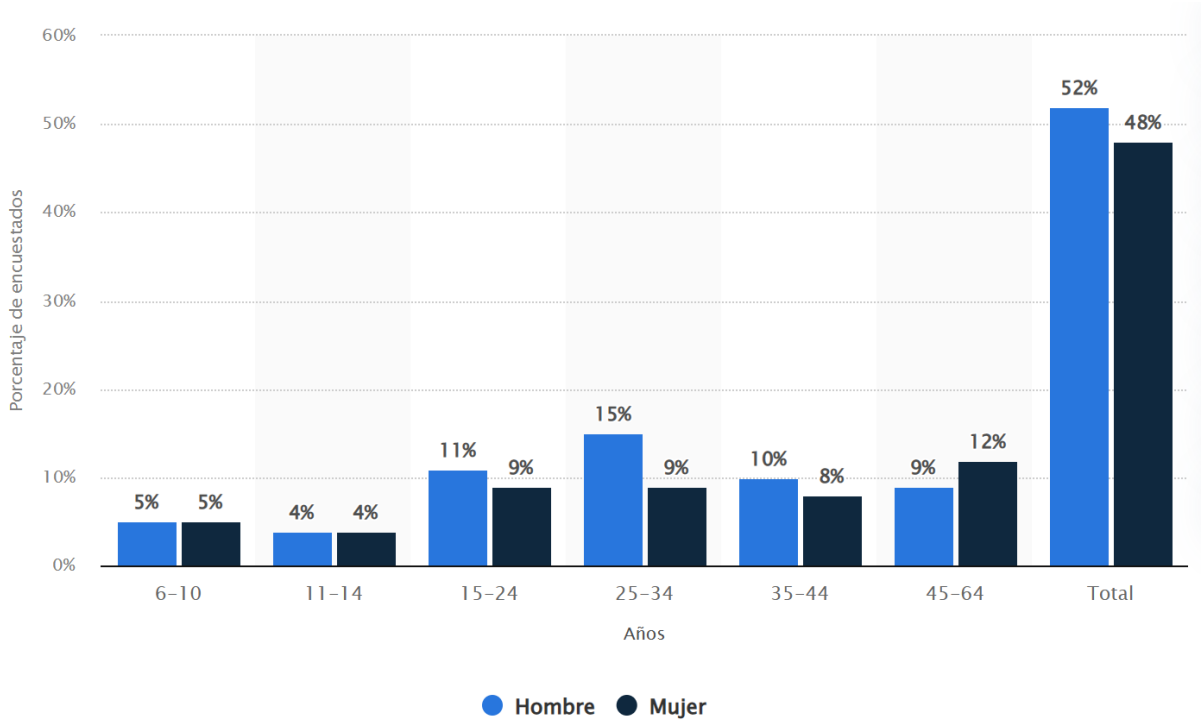

Figura 17: Distribución del mercado de videojuegos en España en 2021 por edad y género.

En cuanto a la distribución del mercado según el género de los videojuegos, (Berzal, 2013) recopila los datos en la Figura 18.

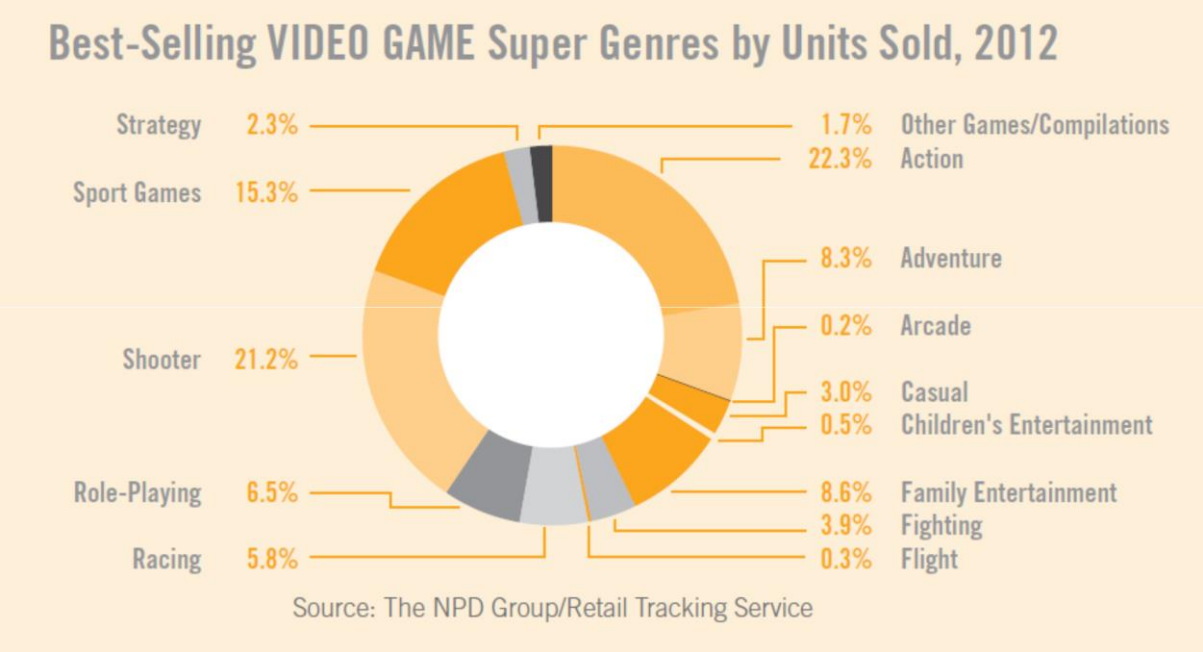

Figura 18: Distribución del mercado de videojuegos mundial 2021 por género.

Considerando el número de jugadores de España de 2021 (Aevi, 2022) tenemos 18 millones de usuarios. De los cuales, según (Orús, 2022) hay un 26% de jugadores que entrarían en nuestro público objetivo, varones de 15 a 34 años, de los cuales, de acuerdo con (Berzal, 2013) un 3,9% compraría juegos de lucha. Con esos datos tendríamos un potencial de 182.520 jugadores en España que podrían estar interesados en la propuesta.

3.3. Estrategia de marketing

Para la estrategia de marketing se va a utilizar a creadores de contenido que promocionen el videojuego grabando gameplays en sus plataformas, Twitch y YouTube principalmente. Los últimos años hemos visto como práctica habitual en la industria de los videojuegos como los estudios y desarrolladores de videojuegos pagan a canales de las diversas plataformas para que creen contenido relacionado con el juego. Este contenido puede ser un análisis, una entrevista y/o un gameplay. Estos gameplay consisten en partidas reales del videojuego retrasmitidas en directo o grabadas y editadas para retransmitirlas en diferido. Estos videos incluyen, en la mayoría de los casos, comentarios del creador de contenido sobre los controles, detalle gráfico e historia entre otros. Estos comentarios ayudan a los espectadores a hacerse una idea de que se enfrentarán si deciden probar el videojuego. Para aumentar más la visibilidad se entregarán claves del videojuego para que los canales puedan hacer sorteos entre sus espectadores. Esto hace crecer a los propios canales, no solo nuestro

videojuego. Con estas claves se pretende incrementar la visibilidad de la campaña publicitaria a la vez que se consiguen mejores precios por los servicios de los canales de contenido.

4. Diseño

A partir de este capítulo (y en los sucesivos, ya que el contenido se puede dividir en más de un capítulo) se tiene que explicar todos los detalles del producto/servicio realizado.

Nota: No todas las sub-secciones propuestas a continuación son aplicables a todos los tipos de TF, por lo cual hay que escoger las más apropiadas según cada caso. También se pueden modificar sus títulos o resumir según se considere conveniente.

4.1. Arquitectura general del sistema

La arquitectura general del sistema, como para cualquier proyecto creado en Unity, consiste en una serie de escenas que contienen los elementos de cada parte del juego. Estas escenas contienen los elementos, “assets”, que dan vida a la escena. Estos elementos pueden ser cámaras, iluminaciones, efectos, imágenes, objetos con físicas, botones para menús, HUD, Contenedores de scripts y cualquier elemento personalizado que cumpla los requisitos de diseño del juego.

En Droid Fighters, estas escenas se agrupan en: Game menu, lobby y mapas. El diagrama de navegación se puede ver en la siguiente imagen (ver Figura 19).

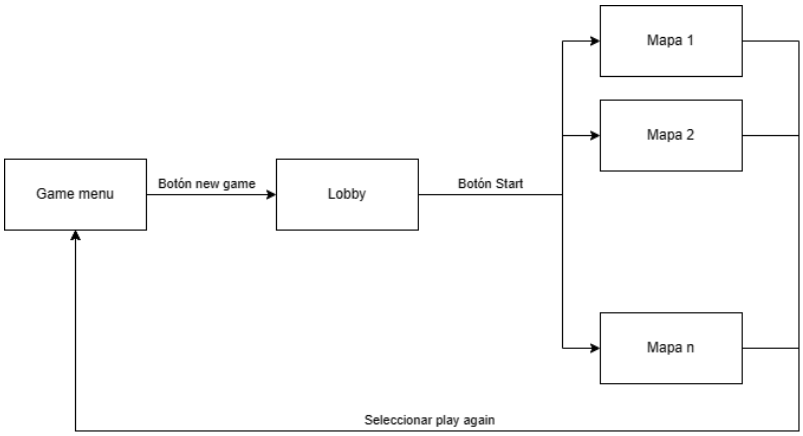

Figura 19: Diagrama de navegación entre escenas

4.2. Arquitectura de la información y diagramas de navegación

4.2.1. Escenas

Game menu

El game menu, ver Figura 20. Es la primera escena que el jugador verá al ejecutar el juego. Puede comenzar una partida nueva, cambiar los ajustes de audio y salir de la aplicación.
Seleccionando partida nueva, “new game”, se accederá al lobby, el cual se describirá en el siguiente apartado (0).

Figura 20: Game menu de Droid Fighters.

En los ajustes de audio se pueden cambiar los niveles de audio, ver Figura 21. Los sonidos del juego están agrupados por categorías y cada categoría está asignada a un mezclador de audio. Esto permite al jugador modificar los niveles del juego por grupo dependiendo de sus preferencias. Por otro lado, esto también permite a los desarrolladores añadir efectos al sonido a un grupo determinado. Como ejemplo en este proyecto, un filtro de paso bajo se activa sobre la música cuando se abre el menú durante una partida.

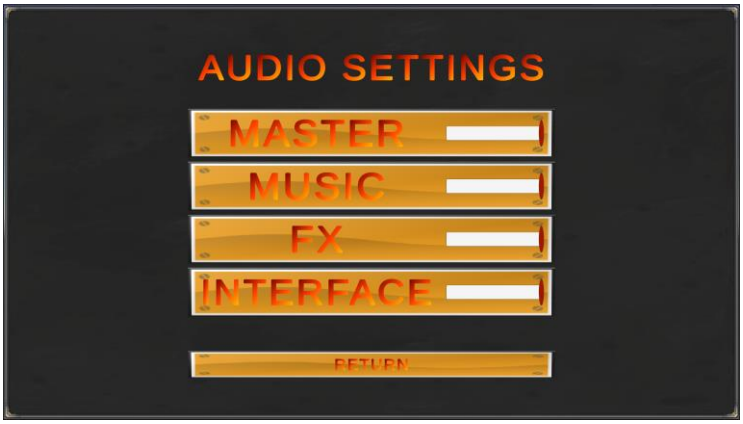

Figura 21: Menú de ajustes de audio.

La última opción cierra el juego, por tanto, se vuelve al escritorio.

Lobby

Al acceder al lobby, ver Figura 22, se puede configurar la partida. Ambos jugadores pueden seleccionar su personaje, seleccionar el mapa y empezar partida.

Figura 22: Escena del lobby.

Mapas

En la categoría de mapa se engloban las diferentes escenas que corresponden con las escenas de combate, arenas, del juego. Todos los mapas tienen los mismos elementos en común, todos los elementos que hacen que el juego de lucha pueda ocurrir, solo se diferencian estéticamente con la posibilidad de añadir jugabilidad extra sin modificar las mecánicas base. Gracias a esto es posible introducir elementos diferenciados únicos para cada escena o algunas de las escenas, estos hipotéticos elementos pueden ser trampas aleatorias que aparecen por la escena. Estos elementos están diferenciados de los objetos estéticos por ser capaces de interactuar con los jugadores. En la Figura 23 se puede ver el mapa 1 del juego visto desde el editor de Unity.

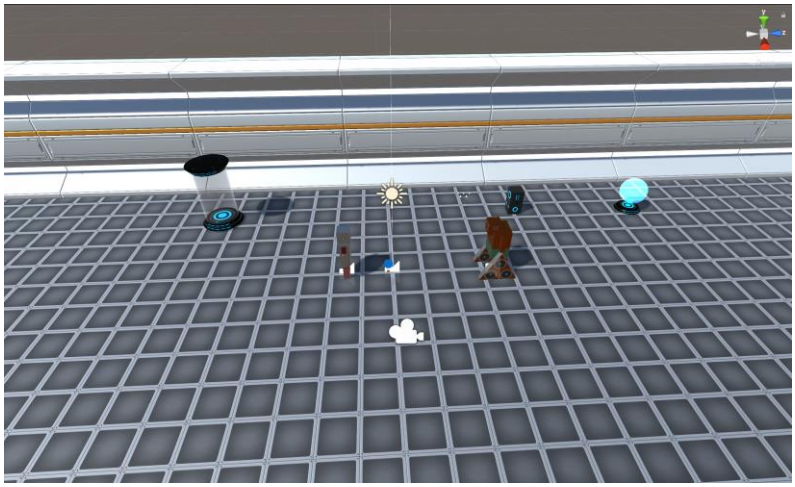

Figura 23: Mapa del juego visto desde el editor de Unity.

4.2.2. Control de estado de juego

El control del estado de juego se realiza mediante un objeto en las escenas de mapa. Este objeto llamado GameplayManager contiene los scripts de control. En lugar de implementar todas las funcionalidades en un solo script se han divido en diferentes archivos. Cada uno de esos archivos se encarga de gestionar una única mecánica permitiendo una mejor legibilidad de código, mejorando su mantenimiento y facilitando añadir funcionalidades manteniendo la modularidad.

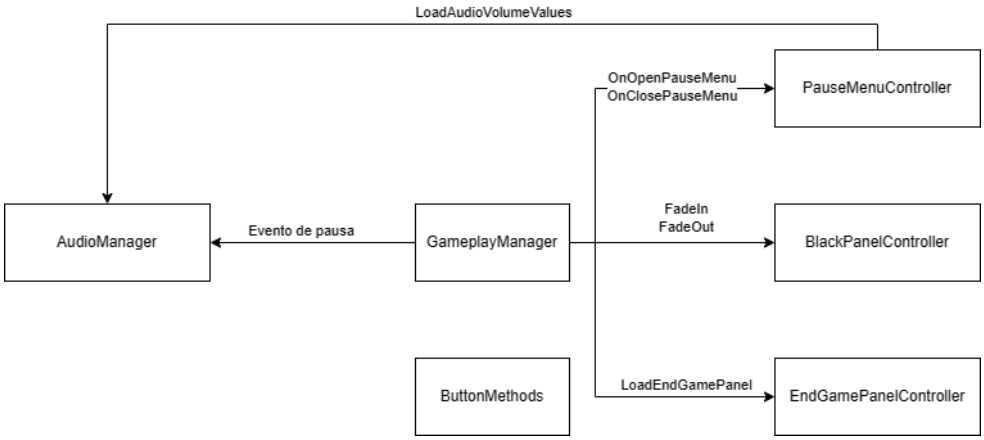

Figura 24: Diagrama de control de estado de juego

GameplayManager

El script GameplayManager se encarga de gestionar la lógica de la partida. En la Figura 25 se puede ver el bucle de control del GameplayManager. Está controlado por corutinas (coroutines en inglés). Estas funciones de Unity permiten pausar la lógica hasta que un evento ocurra, esto permite reducir la carga de procesado durante la ejecución evitando comprobaciones de variables en cada fotograma.

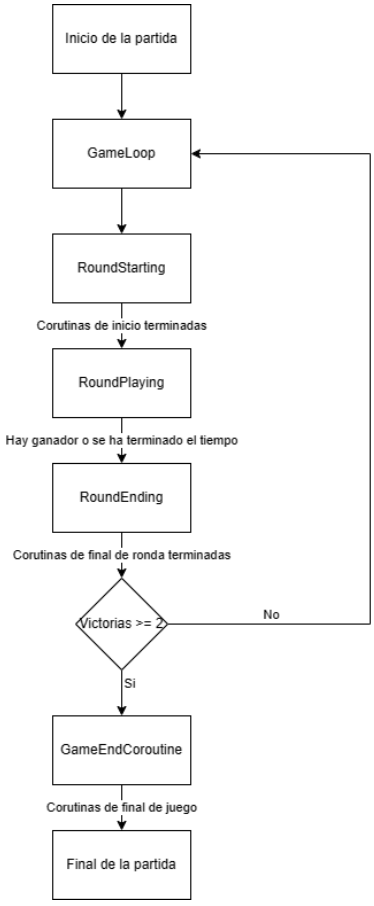

Figura 25: Lógica del GameplayManager

La corutina GameLoop se encarga de mantener el bucle lógico durante la partida. Este se inicia cada vez que una ronda termina.

La corutina RoundStarting es la que gestiona la lógica en el inicio de la partida. Al cargar la partida los caracteres se mueven a la posición de inicio, se les resetean los atributos de salud y se cargan los valores iniciales de los stats. Esta rutina permite tener siempre las condiciones iniciales

independientemente desde donde se haya accedido a la escena. Esto consigue separar la funcionalidad de las escenas de Game Menu y Lobby de los mapas, permitiendo modificar cualquier escena sin interferir a las demás. Esto permitiría incluso hacer saltos entre mapas sin pasar por el lobby permitiendo crear un modo torneo de forma rápida y ordenada.

Para gestionar la lógica entre rondas hay dos corutines, RoundPlaying y RoundEnding.

RoundPlaying gestiona la lógica durante la partida. Inicia el contador de tiempo, el cual va restando una unidad al tiempo cada segundo y actualiza el panel para que los jugadores puedan consultarlo en tiempo real. Una vez un jugador ha ganado o se ha acabado el tiempo activa la corutina RoundEnding.

RoundEnding comprueba si ha sido una victoria por tiempo o por K.O. Si ha sido por tiempo, comprueba que jugador ha terminado en pie para declararlo vencedor. Si ambos tienen la misma vida, la ronda se informará a los jugadores del empate y se volverá a jugar la ronda. En caso de finalización por K.O. se declarará vencedor al jugador superviviente.

La última corutina, GameEndCoroutine, activa el final de la partida, se activa el FadeIn del black panel y muestra el panel con el vencedor y las opciones para jugar de nuevo la misma partida, volver al menú de juego o salir de la aplicación.

También gestiona el HUD de los jugadores actualizando sus barras de vida y el contador de victoria.

EndGamePanelController

Se encarga de habilitar el panel de final de juego mostrando al ganador y los botones para volver al Game Menu o salir del juego.

BlackPanelController

Este script se encarga de habilitar o deshabilitar el panel negro (BlackPanel). El objeto BlackPanel consiste en una imagen negra que abarca toda la pantalla y se encuentra por encima de los demás objetos del Canvas. Esto quiere decir que si el panel está activo solo se verá la pantalla en negro. Cuando la partida comienza este panel se vuelve transparente. Para suavizar la transición de negro a transparente y transparente a negro se utilizan los métodos de FadeIn y FadeOut. Estos métodos hacen que la transición sea progresiva aumentando o disminuyendo la transparencia del panel a lo largo de los frames durante los segundos establecidos.

PauseMenuController

Controla los paneles del menú de pausa, permite la navegación entre menús. Activa desactiva los paneles de los menús según le indique el usuario mediante los botones.

ButtonMethods

Agrupa los métodos de los botones comunes de los menús. Este script es útil para evitar tener que duplicar trabajo permitiendo que todos los botones para salir del juego usen el mismo método.

AudioManager

El AudioManager controla los ajustes de audio, se encarga de administrar las barras de desplazamiento, slides, del menú de ajustes de audio. También guarda los valores usando la clase PlayerPrefs de Unity para guardar de forma persistente los ajustes. Guardar los parámetros de ajuste permite cargar automáticamente los ajustes cada vez que se cargue una escena, desde el inicio del juego hasta al entrar en una partida.

4.2.3. Estadísticas de personaje

Las estadísticas de personaje son atributos del personaje. Estos atributos influyen en el comportamiento del personaje, en su velocidad y fuerza. En lugar de usar una variable se ha decidido crear un objeto para facilitar la gestión de los atributos. Con esta estructura se puede agregar modificadores a los atributos de los personajes. El jugador puede generar estos modificadores al ejecutar los combos que los otorgan.

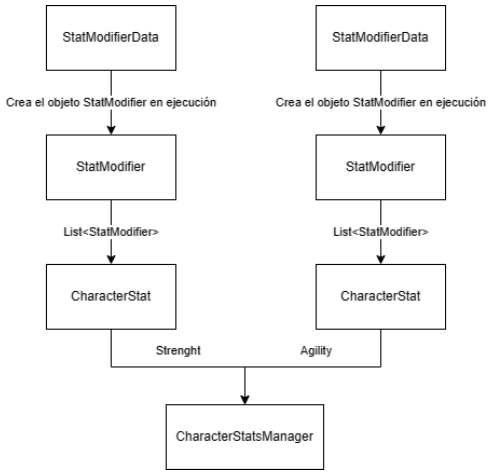

Figura 26: Diagrama de las estadísticas de personaje.

El script StatModifierData contiene la información para crear los StatModifiers en tiempo de ejecución. Este objeto es un ScriptableObject para poder añadirlo fácilmente, con el editor de Unity, a otros objetos, o en nuestro caso, generar los modificadores cuando se ejecuta esa habilidad.

StatModifier guarda la información del modificador a la estadística. Guarda el valor, el tipo (numérico o porcentual), el orden en el que se aplicará y la fuente que lo otorgó.

El script CharacterStat proporciona el valor de la estadística y la lista de modificadores. Independientemente de los modificadores, el valor de la estadística no se actualiza en tiempo real. Solo se actualiza cuando se consulta el valor, esto evita operaciones innecesarias.

CharacterStatsManager guarda los stats para que cualquier script pueda acceder a esos valores cuando los necesite. También gestiona los efectos temporales, que en este juego son la totalidad, activa una corutina que eliminará el modificador cuando haya pasado el tiempo. Además, guarda una lista con los modificadores de las estadísticas para eliminarlos todos al final de cada ronda y partida.

4.2.4. Control de combos

El sistema de combos sigue el diagrama de la Figura 27. Cuando el jugador usa el botón de combos el panel se activa y la lógica de los combos comienza a ejecutarse con el DynamicComboManager, el cual se encarga de cargar los nodos de los distintos tipos de acciones disponibles. Cada vez que se alcanza un nodo, se ejecutarán sus efectos y/o ataques si los tuviera y cargará la siguiente lista de nodos hasta llegar al último nodo. En el cual se ejecutará la acción final, un ataque y/o un efecto.

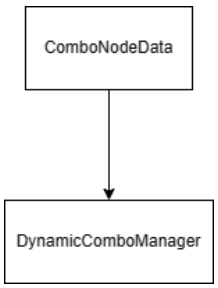

Figura 27: Diagrama del control de combos.

El árbol de nodos se puede consultar en la Figura 28. El árbol muestra todas las posibles combinaciones de nodos. El jugador podrá elegir que ataque final quiere usar moviendo el Joystick hasta el siguiente nodo deseado.

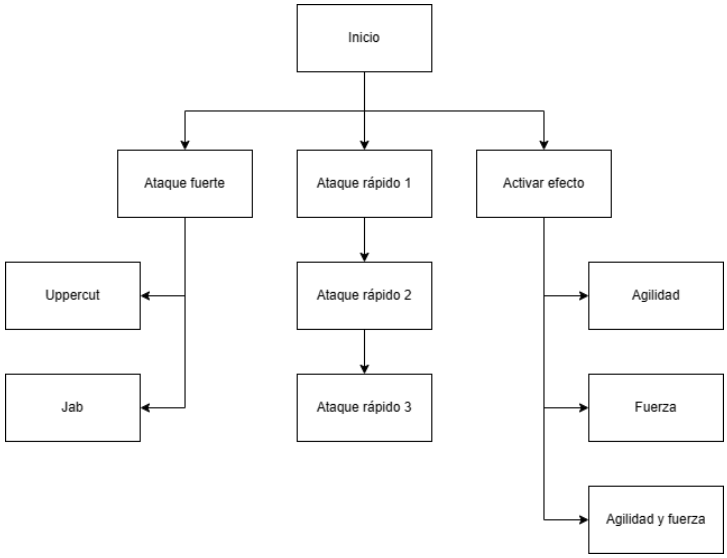

Figura 28: Diagrama de nodos del sistema de combos.

4.2.5. Control de cámara

El sistema de control de cámara es muy sencillo. Un solo script CameraControl se encarga de mantener enfocados a los jugadores. Para ello el script accede a la lista de jugadores (PlayerList) para analizar sus posiciones y calcular la distancia que la cámara tiene que tener para mantener a los personajes en escena.

La segunda interacción se produce cuando un jugador sufre daño. Este evento activa el método StartCameraShake del script CameraControl desde el controlador del personaje Robot_Controller cuando este recibe daño.

Figura 29: Diagrama del control de cámara.

4.3. Lenguajes de programación y APIs utilizados

4.3.1. Software

Sistema operativo

Se ha escogido el sistema operativo Windows 10. Se ha escogido este sistema por ser el disponible del equipo de desarrollo. La compatibilidad de herramientas, las alternativas disponibles y los usuarios en esta plataforma es mayor que en las demás. Otras opciones son iOS o Linux como alternativas más viables. Según el estudio estadístico de (Fernández, 2022), un 75% de usuarios utiliza sistemas con Windows frente a 14,5% de usuarios de MAC y 2,8% de usuarios con Linux. Esto ayudará a encontrar talento cuando se decida ampliar el equipo o encontrar colaboradores.

Desarrollo

En este apartado se describen y analizan los programas utilizados para la creación del videojuego. Con Unity como motor de físicas y Visual Studio como editor de código.

Para el desarrollo de este proyecto se ha escogido Unity en su versión 2020.3.18f1. Se ha escogido Unity por la familiaridad del equipo con esta herramienta. Y la versión escogida es la 2020.3.18f1 por el mismo motivo. Al tener que desarrollar un videojuego completo en solo 4 meses se ha elegido utilizar la misma versión que para proyectos anteriores, esto ha evitado tener que configurar la aplicación y tener que actualizar el código de los assets reutilizados para que fueran compatibles con versiones más actuales de Unity. Otras opciones, como hemos visto en el capítulo 2.1, podrían haber sido Unreal Engine (Ver capítulo 2.1.1), UbiArt, Cryengine, Source engine, GameMaker o Stencly.

Al elegir Unity como software de partida se ha tenido que escoger entre C# y Java como lenguajes de programación. Se ha escogido C# por la mayor experiencia del equipo con este lenguaje, por tener más alto nivel de código que Java y por la mayor documentación y comunidad que tiene Unity con C# de la que tiene con Java.
Para el entorno de programación se ha escogido Visual Studio 2022. Al elegir C# como lenguaje de programación el uso de Visual Studio es una elección inmediata. Esta aplicación incorpora C# como lenguaje base con que se simplifica la instalación. El mismo Unity te recomienda usar Visual Studio como editor de código externo y viene con plug ins preinstalados para conectar ambas aplicaciones.

En cuanto a la versión utilizada, se ha decidido usar la 2022 debido a que incorpora mejorar sustanciales al ya eficiente predictor de código. Esto hace que se pueda programar mucho más rápido sugiriendo las opciones recomendadas según el contexto. También selecciona las recomendaciones comprobando los tipos de variables, haciendo que la predicción sea mucho más precisa ofreciendo muchas menos opciones, pero con mayor lógica.

Las alternativas a Visual Studio son: Visual Code, Xcode, NetBeans y Firebase entre otros. Programas que no cuentan con tanta documentación específica para ser usados con Unity ni con tanta documentación online.

Diseño

Las aplicaciones utilizadas para diseñar los elementos gráficos del videojuego son, por un lado, Blender para la creación y animación de los personajes del juego, los robots que lucharán. Audacity para la edición de los clips de audio, Photoshop para la creación de texturas y sprites, Adobe Premier Pro para la edición de video.

Para el diseño y animación de los personajes se ha escogido Blender por mayor familiaridad del equipo con esta herramienta y por precio, es gratuito a diferencia de las otras opciones profesionales disponibles como Maya.

Audacity se ha utilizado para la edición de audio, se ha elegido por la cantidad de herramientas que tiene este programa, su ligereza, optimización y manejo intuitivo. Otros softwares alternativos para la edición de audio son Ocenaudio o Ardor.

Para la edición de imágenes se ha elegido Photoshop por ser una herramienta bien conocida por el equipo de desarrollo, las herramientas ilimitadas y la documentación disponible hacen que sea la opción predilecta. Otras opciones son GIMP y Glimpse.

Se ha escogido Adobe Premier Pro para la edición de video por los mismos motivos que Photoshop, al fin y al cabo ambas aplicaciones forman parte de la misma suit de software. Otros softwares son Sony Vegas y Cyberlink PowerDirector entre muchas opciones.

Otros

En este apartado se exponen otros softwares utilizados, estos software no tienen un efecto directo en el proyecto pero son necesarios para mantener un workflow natural y simplificar tareas durante el desarrollo del proyecto.

Para el control de versiones se ha escogido GitKraken por su facilidad de configuración y uso. Inicialmente el equipo trabajaba con SourceTree pero después de muchos fallos de sincronización y de credenciales se cambió a GitKraken. Otras alternativas son el ya mencionado SourceTree, SmartGit o TortoiseGit.

Como repositorio del proyecto se ha usado GitHub, se ha elegido GitHub sobre GitLab por el carácter gratuito del primero, aunque en funcionalidades ambos están a la par. Otra opción es SourceForge.

Los gameplays se han grabado utilizando OBS por ser gratuito y la gran comunidad de internet, siendo uno de los softwares más utilizados para la captura de video y retransmisiones en directo. Otras opciones posibles son Movavi Screen Recorder o Icecream Screen Recorder.

La plataforma para alojar los videos grabados ha sido YouTube. Aunque por las características de este proyecto hubiera sido más adecuado usar Twich ya que esta plataforma está más enfocada en el mundo de los videojuegos. En este caso, al tratarse de plataformas de distribución nada impide utilizar varias a la vez para aumentar la visibilidad del proyecto y futuras campañas publicitarias. Otras plataformas son Twich, InstaGib.TV o Vimeo.com.

4.3.2. Assets de terceros

Modelos de nivel

3D Scifi Kit Starter Kit | 3D Environments | Unity Asset Store
SciFi Props | 3D Interior | Unity Asset Store

Sonido

Download Free Game Sound Effects | Mixkit
Boss Fight Bounce | OpenGameArt.org
"A Fight in the Fields" (RPG Orchestral Essentials) - Combat Music | OpenGameArt.org
Sci-Fi RTS War Unit Sounds | OpenGameArt.org

Interface

1-Bit Input Prompts Pixel 16× | OpenGameArt.org
Game icons (expansion) | OpenGameArt.org
Analog Stick for Mobile Games | OpenGameArt.org
700+ RPG Icons | OpenGameArt.org
48 Swordsman Skills Icons Pixel Art | OpenGameArt.org
Shield | OpenGameArt.org
Explosion effects and more | OpenGameArt.org
Public Domain Alphas | OpenGameArt.org

4.3.3. Hardware

Como hardware se ha utilizado un PC IBM compatible, esta plataforma tiene soporte nativo para Windows, concretamente Windows 10 para nuestro caso. Esto causa la elección de Windows como sistema operativo para desarrollar el proyecto entre otras razones que hemos visto en el apartado 4.3.1.

Los componentes del PC son un procesador Intel i7 3930k en una placa Asrock X79 EXTREME6, 16Gb (4x4Gb) de RAM DDR3 en configuración de 4 canales, una gráfica NVIDIA 1080 Ti, 2 discos duros SSD Samsung EVO de 256Gb cada uno.

Se han utilizado dos pantallas. La principal es una LG 27GP950-B de 27" y resolución 4k, una segunda pantalla DELL S Series S2417DG de 23.8" de tamaño y resolución 2K.

Como dispositivos de entrada se han usado un teclado Razer Huntsman, un ratón logitec G502 Hero y un mando Xbox One. Aunque el juego es compatible también con cualquier mando genérico.

Para grabar el audio de los videos se ha utilizado un TAKSTAR SGC-598.

A mitad del desarrollo del proyecto, el rendimiento del equipo ha sido insuficiente y ha sido necesario actualizar la CPU con estas características: procesador Intel i9 12900k en una placa GIGABYTE GiBy Z790 UD AX Z790, 32Gb (2x16Gb) de RAM DDR5 en configuración de 2 canales. Se ha mantenido la gráfica NVIDIA 1080 Ti y se ha añadido un disco duro Samsung 980 PRO M.2 NVMe SSD (MZ-V8P1T0BW) de 1 TB. Los dos discos duros de 256Gb se han mantenido.

5. Implementación

5.1. Requisitos de instalación

5.1.1. Requisitos de hardware

Para instalar el juego se requiere un PC compatible con Windows 7 o superior con arquitectura 64 bits. El procesador mínimo debe ser un Intel I3 de cuarta generación o superior o un AMD A8-7680 o superior. La tarjeta gráfica mínima equivalente a una GeForce GT 730 o una rx 550 o modelos superiores. La RAM mínima requerida es de 8Gb de RAM. En cuanto al espacio de almacenamiento requiere 300Mb para los archivos del juego.

Como periféricos se requiere se necesita un monitor, ratón y un mando, todos ellos genéricos.

5.2. Instrucciones de instalación y ejecución

Para instalar el juego hay que descargar el archivo comprimido de la sección de este documento de la ficha del trabajo final en el apartado ejecutable. Una vez descargado el archivo hay que descomprimirlo en la ubicación que desee el usuario. Como paso opcional se puede entrar en la carpeta descomprimida y crear un acceso directo al ejecutable (archivo .exe).

La ejecución del juego es directamente a través del ejecutable o del acceso directo como cualquier otra aplicación de ordenador. Una vez ejecutado, para empezar una nueva partida seleccionar el botón “New Game” ver Figura 30: Menú del juego.Figura 30.

Figura 30: Menú del juego.

Para empezar la partida hay que marcar a ambos jugadores como listos para empezr. Esto se hace seleccionando los botones “Not Ready” de cada jugador hasta que cambien el texto del botón a

“Ready”. Una vez ambos jugadores están listos de puede pulsar el botón “Start” para comenzar la partida. Ver Figura 31.

Figura 31: Menú de configuración de partida.

5.3. Controles

Los controles se han programado usando el asset de Unity “New input system”. Al usar este asset se consigue simplificar la gestión de controles pudiendo configurarlos por dispositivos y por mapas de acciones (Action maps). Estos mapas de acciones permiten activar o desactivar mapas según el contexto del juego. En este caso, se han creado tres mapas principales. Uno, de nombre “Actions” con los controles del juego, ataque, defensa, movimiento, pausar juego y menú de combos. El segundo mapa, de nombre “Combo”, contiene el movimiento del cursor y terminar el combo. Y el tercero para poder salir del menú de pausa.

Al configurar los mapas con el “New input system” de Unity podemos cambiar de un mapa otro directamente usando el código. En Droid Fighters esto se usa para desactivar los controles del juego al iniciar un combo, al igual que se desactivan los controles del juego cuando se pausar el juego.

5.3.1. Control en menús

El control en los menús se realiza mediante el ratón. Para desplazar el cursor se utiliza el movimiento del ratón como una aplicación normal. Para seleccionar un botón hay que usar el click izquierdo. Ver Figura 32.

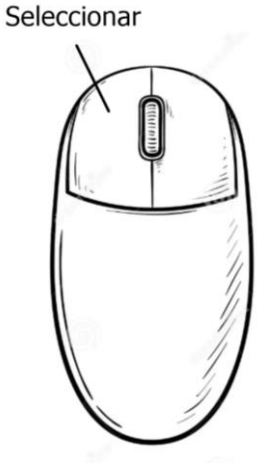

Figura 32: Esquema de controles de ratón.

5.3.2. Control en juego

Los mapas de control en la pantalla de juego se dividen en 3 mapas.

Mapa de acciones básico

El mapa de acciones está activo al iniciar la partida. Nos permite desplazar el personaje y realizar acciones básicas de defensa y ataque, ver Figura 33. También proporciona botones para abrir el panel de combo y activar la pausa, la cual abre el menú de pausa.

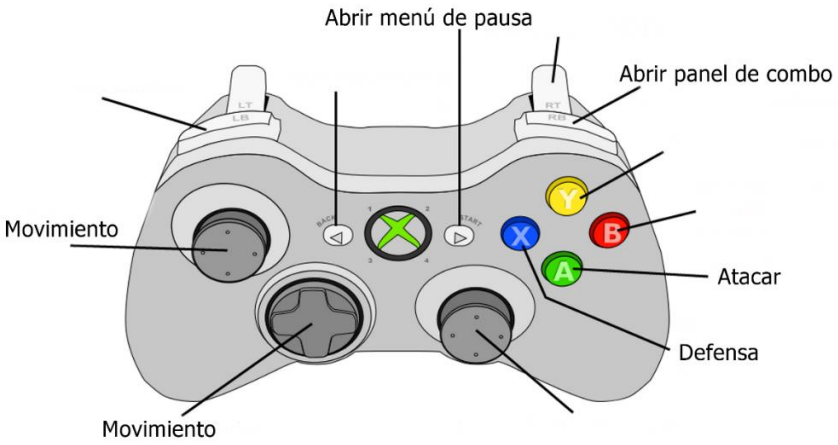

Figura 33: Esquema del mapa de acción.

Mapa de acciones de combo

Si el jugador activa el panel de combo, el mapa de acciones básico se desactiva y se activa el mapa de acciones de combo, el cual permite mover el cursor dentro del panel de combo. El jugador puede también cerrar el panel de combo a costa de perder el progreso del combo, ver Figura 34. Los efectos que haya conseguido durante el combo permanecerán activos durante el tiempo correspondiente. Al salir de combo, ya sea por interrupción o por que se haya completado, cerrará el panel de combo, desactivará el mapa de acciones de combo y activaría el mapa de acciones básico.

La opción de interrumpir el combo proporciona una opción para defenderse si el enemigo decide acercarse para atacar en lugar de aprovechar y realizar un combo el mismo.

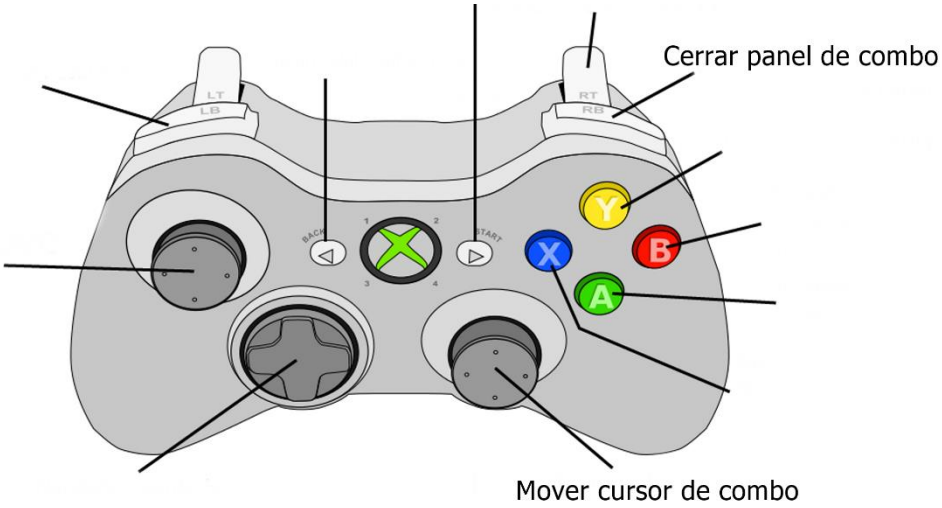

Figura 34: Esquema del mapa de combo.

Mapa de menú de pausa

El mapa del menú de pausa desactiva el menú de acciones básico y permite salir del menú de pausa. Ver Figura 35.

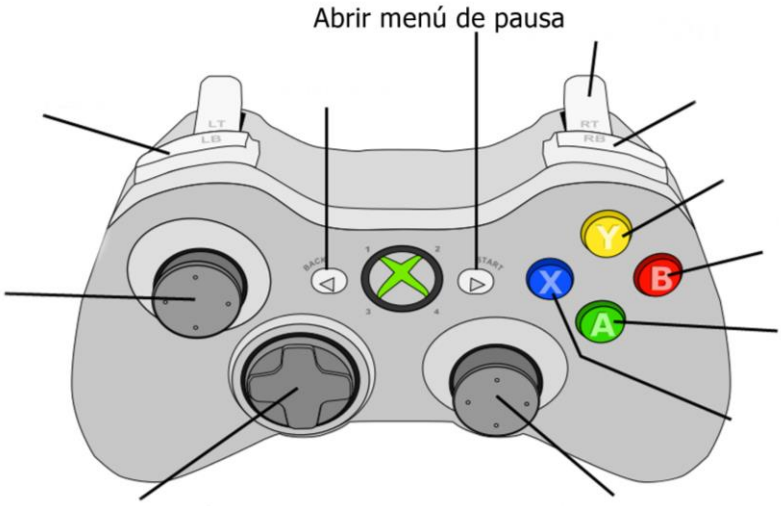

Figura 35: Esquema del mapa de pausa.

6. Demostración

6.1. Guía de usuario

Para realizar ataques hay que tener en cuenta que cada ataque tiene una distancia efectiva. Si estamos demasiado lejos de nuestro objetivo, nuestro ataque no tendrá efecto. Si estamos demasiado cerca no seremos penalizados en nuestro ataque, pero seremos vulnerables a ataques de corto alcance de nuestro adversario.

Si el jugador quiere defenderse de un ataque puede ponerse en modo defensa, mientras mantenga la tecla de defensa pulsada el carácter del jugador estará en defensa. Para dejar de defender y recuperar el resto de controles se tiene que soltar el botón de defensa.

Para realizar acciones especiales hay que usar el sistema de combos, una vez activo el jugador tiene 3 opciones o caminos.

Ataque fuerte

El ataque fuerte da acceso a dos ataques fuertes, cada uno con sus propias características y efectos.

Figura 36: Icono de los ataques fuertes.

Ataques rápidos

El ataque rápido abre una cadena de nodos, todos con un único icono y siempre el mismo, el de la Figura 37. Una vez tdrminado el combo, se realizará la serie de ataques rápidos.

Figura 37: Icono del ataque rápido.

Effectos

El nodo reprensetado en la Figura 38 es el nodo de los efectos. En este nodo podemos seleccionar efectos para potenciar la velocidad de movimiento y la fuerza de los ataques durante un corto periodo de tiempo.

Figura 38: Icono del nodo de efectos.

7. Conclusiones y líneas de futuro

7.1. Conclusiones

7.1.1. Lecciones aprendidas

La primera de los conocimientos adquiridos es haber aprendido como estructurar el proyecto de creación de un videojuego. Desde idear un concepto novedoso hasta implementarlo en una versión jugable.
Otro conocimiento es el de calcular presupuestos para un pequeño estudio de videojuegos. Teniendo en cuenta el salario de los empleados, el coste de los equipos, campañas publicitarias y estimar el público objetivo para analizar el potencial del mercado y los beneficios del producto.
La última de las lecciones aprendidas principales es la mejora en las habilidades de programación. Mejorando la agilidad de programación y optimizando el diseño de código para la compartimentación de clases y módulos.

7.1.2. Reflexión sobre los objetivos iniciales

Hemos conseguido crear un prototipo funcional del juego en el que probar el concepto del nuevo sistema de control para un juego de lucha. Utilizando este prototipo como punto de partida se puede terminar de añadir las funcionalidades que quedan pendientes ver el apartado 9 Anexos.

No se ha conseguido tener un prototipo vendible, en estado de venta anticipada. Como se ha mencionado en el apartado anterior, hay algunas funcionalidades, más enfocadas en las mecánicas complementarias de un juego de lucha que en el sistema de combos propuesto, que no se han podido implementar al centrar los esfuerzos en la legibilidad y mantenimiento del código.

Durante el desarrollo de este proyecto se ha priorizado el facilitar el mantenimiento del código. Todos los scripts se han diseñado e implementado teniendo en cuenta las posibilidades de actualización y expansión. Se han separado cada una de las mecánicas del juego en scripts individuales. Esto permite modificar fácilmente las mecánicas existentes y añadir nuevas al facilitar el análisis del código para encontrar los puntos a actualizar y reducir los mismos. Lo mismo se aplica a la actualización del código, cualquier colaborador o programador que continúe el proyecto podrá orientarse fácilmente.

7.1.3. Reflexión sobre la planificación y la metodología

La planificación elegida se ha seguido según lo establecido. Se han cumplido las fechas de entrega con resultados dentro de las expectativas.

La metodología utilizada es la misma que la planteada inicialmente. Se ha conseguido un proyecto funcional centrándose en implementar el sistema de control de combos. Se ha hecho hincapié en la flexibilidad y el mantenimiento del código. La estructura del código y el resultado del juego demuestran que la metodología escogida ha sido la correcta. Si se hubiera escogido un proyecto existente para añadir el sistema de combos hubiera sido complicado añadir el sistema o incluso imposible. Partiendo de un proyecto ajeno se hubiera perdido mucho tiempo en implementar cada una de las hipotéticas modificaciones para dejarlo a nuestro gusto.

No se han tenido que introducir cambios, en el sentido de que se han mantenido los objetivos planteados al inicio, aunque no se hayan llegado a cumplir. Se han mantenido para mejorar el entendimiento de lo que se ha querido conseguir con este proyecto y como guía para aquellos que lo quieran continuar.

7.2. Líneas de futuro

En este apartado se exponen las líneas de futuro para el proyecto. En ella se describen funcionalidades adicionales y contenido para añadir posteriormente a la defensa de este proyecto.

7.2.1. Añadir IA de combate

Actualmente el contrincante no es más que un espantapájaros que no realiza ninguna acción. El primer punto implementar en el futuro es implementar una IA y entrenarla para que pueda seguir un mínimo de estrategia de ataque y sea capaz de defenderse.
Este punto es el más importante, pues sin un rival, el juego de lucha solo sería un juego de sparring sin posibilidad, ni aliciente de progreso y/o victoria.

7.2.2. Animaciones nuevas

El siguiente punto a desarrollar sería crear animaciones nuevas de los personajes. En lugar de tener una única animación para atacar, se pueden crear animaciones diferentes para cada ataque. Esto dará mayor inmersión y profundidad a la jugabilidad.

Otras animaciones que se pueden añadir son cambiar la postura del jugador cuando tiene alguno de los efectos activos. Para el efecto de velocidad, cambiar la animación de estado por defecto por una con una postura que indique que esté preparado para correr. Al igual que la animación de moverse, que tendría que modificarse para mostrar la nueva velocidad.

Los mismo podría hacerse para el efecto de fuerza añadiendo una postura que indique que está preparado para atacar complementando el aura del efecto.

7.2.3. Añadir capacidad de salto

Una mecánica que se puede introducir para mejorar la jugabilidad y aumentar las acciones disponibles es la capacidad de salto. Esta mecánica abre la posibilidad de saltar por encima del oponente, hacer ataques desde el aire y tener que defenderse de ataques aéreos.

Al haberse programado el control de estado de personaje con una máquina de estados. Implementar esta mecánica es una tarea sencilla. Simplemente creando un estado nuevo y programando la funcionalidad de salto en él.

7.2.4. Nuevos mapas y personajes

Actualmente el juego cuenta con dos personajes y un mapa. Lo ideal para una primera versión de acceso anticipado (Early Access) sería 8 personajes diferenciados y al menos 4 mapas.
Para aumentar el número de personajes diferentes se pueden sacar simplemente una decoración alternativa para cada personaje, duplicando el abanico de personajes disponibles. La segunda versión del personaje puede ser más difícil de conseguir. Esto aumentará la competitividad del jugador y la satisfacción cuando consiga desbloquearlo.

7.2.5. Modo torneo

El modo torneo consistiría en una serie de combates a eliminación directa en el que el jugador tendrá que superar todos los combates para llegar a la final. Si gana la ronda final, y por lo tanto el torneo, el jugador desbloqueará contenido del juego, un personaje o una arena nueva.
El generador de torneos comprobará que luchadores y mapas le quedan al jugador por desbloquear y elegirá uno de ellos para el combate final. Garantizando que si el jugador gana desbloqueará un elemento nuevo si aún le queda alguno.

8. Bibliografía

DEV COMUNITY website: https://dev.to/goldenxp/core-mechanics-of-fighting-games-24ej, consultado 27/03/2023

LITE THE IRON MAN website: https://litetheironman.medium.com/simplified-controls-in-fighting-games-7953d6e45224, consultado 29/03/2023

Aevi. (2022, 4 20). *El videojuego facturó 1.795 millones de euros en 2021, con una base superior a los 18 millones de usuarios en España*. Retrieved from Aevi: http://www.aevi.org.es/videojuego-facturo-1-795-millones-euros-2021-una-base-superior-los-18-millones-usuarios-espana/

Berzal, F. (2013). *Desarrollo de videojuegos.* Retrieved from DECSAI Departamiento de ciencias de la computación e I.A.: https://elvex.ugr.es/decsai/games/slides/lab/P0-Propuesta.pdf

Chau, A. (2001, 12 1). *Fatal Fury: Mark Of The Wolves*. Retrieved from Dreamcast: https://web.archive.org/web/20081007042313/http://uk.dreamcast.ign.com/articles/166/166258p1.html

Daily, G. (2009, 1 11). *Top 20 Street Fighter Characters of All Time*. Retrieved from Game Daily: https://web.archive.org/web/20090224060235/http://www.gamedaily.com/articles/galleries/top-20-street-fighter-characters-of-all-time/?page=2

Daily, G. (2009, 1 11). *Top 25 Most Bizarre Fighting Characters*. Retrieved from Game Daily: https://web.archive.org/web/20090206220705/http://www.gamedaily.com/articles/galleries/top-25-most-bizarre-fighting-characters/?page=24

Dhami, N. (2021, 12 27). *medium.* Retrieved from https://litetheironman.medium.com/simplified-controls-in-fighting-games-7953d6e45224

Ekberg, B. (2007, 9 22). *TGS '07: K-1 World Grand Prix Hands-On*. Retrieved from Gamespot UK: https://archive.ph/20120715174733/http://uk.gamespot.com/ds/action/k1worldgp/news.html?sid=6179720&mode=previews&tag=result;title;0#selection-2089.0-2089.38

Fernández, R. (2022, Febrero 20). *Cuota de mercado de los sistemas operativos para ordenador a nivel mundial en el tercer trimestre de 2022.* Retrieved from https://es.statista.com/: https://es.statista.com/estadisticas/576870/cuota-de-mercado-mundial-de-los-sistemas-operativos/

Gerstmann, J. (1999, 12 29). *Street Fighter III: Double Impact Review*. Retrieved from Gamespot: https://archive.ph/20120715010218/http://uk.gamespot.com/dreamcast/action/streetfighter3doubleimpact/review.html?om_act=convert&om_clk=gssummary&tag=summary;read-review

Iyer, S. (2021, 03 21). *Core Mechanics of Fighting Games.* Retrieved from Dev Comunity: https://dev.to/goldenxp/core-mechanics-of-fighting-games-24ej

Leone, M. (2009, 1 11). *Virtua Fighter*. Retrieved from 1UP: https://archive.ph/20120719110526/http://www.1up.com/features/essential-50-virtua-fighter

Orús, A. (2022, Julio 28). *Distribución de los jugadores de videojuegos en España en 2021, por edad y género*. Retrieved from Statista: https://es.statista.com/estadisticas/481369/jugadores-de-videojuegos-en-espana-por-edad-y-genero/

Parish, J. (2009, 1 15). *Street Fighter II*. Retrieved from 1UP: https://archive.ph/20120720141819/http://www.1up.com/features/essential-50-street-fighter-ii#selection-1239.0-2044.0

Park, A. (2007, 6 5). *Art of Fighting Antology Review*. Retrieved from Games spot: https://archive.ph/20120717135451/http://uk.gamespot.com/ps2/action/artoffightinganthology/review.html?om_act=convert&om_clk=gssummary&tag=summary;read-review

Rollings, A., & Adams, E. (2006). *Companion Website for Game Development*. Retrieved from https://wps.prenhall.com/bp_gamedev_1/54/14053/3597646.cw/index.html

Rose, M. (2008, 12 5). *Designing Kung Fu Chaos, Part Three*. Retrieved from xbox.com: https://web.archive.org/web/20081205175426/http://www.xbox.com/en-US/games/k/kungfuchaos/themakers5.htm

Towell, J. (2009, 1 29). *"The Best Special Attacks Ever"*. Retrieved from Games Radar: https://www.gamesradar.com/the-coolest-stuff-in-zelda-breath-of-the-wilds-first-dlc-a-tingle-costume-and-a-map-tracker-that-changes-everything/

Treit, R. (2009, 5 15). *Novice guides: Fighting*. Retrieved from Xbox.com: https://web.archive.org/web/20090515013224/http://www.xbox.com/en-US/games/tips/noviceguides/fighting.htm

9. Anexos

Anexo A: Glosario

- **Asset:** Elemento de Unity3D que tiene valor por sí mismo y es independiente del motor de juego actual, por lo que se puede usar en distintos proyectos: elemento audiovisual, base de datos, ...
- **Frame:** Fotograma o imagen que se repite un número de veces por segundo en un videojuego, entre 30 y 60 frames por segundo. Implica la velocidad de refresco y sensación de fluidez en el juego.
- **GameObject:** Elemento estructural de Unity3D que puede contener distintos subelementos o componentes y que describen su comportamiento: scripts, sprites, sonido, ...
- **IDE:** Integrated Development Environment, entorno de desarrollo integrado.
- **NPC:** Non player Character, personaje no jugador.
- **Prefab:** Conjunto de GameObjects almacenado como un único objeto para poder ser utilizado o instanciado y que se usa como plantilla para generar otros objetos.
- **Rigging:** Añadir un sistema de control en un modelo 2D o 3D, normalmente conocido como huesos y su influencia en modelo, para su futura animación.
- **Script:** fichero de código, en el caso de Unity3D en lenguaje C#.
- **Sprite:** imagen en formato mapa de bits.
- **Early Access Game:** Juego de acceso anticipado. Los juegos publicados en este estado no están completados

Anexo B: Entregables del proyecto

Tráiler: https://youtu.be/yLnGavu4hgY

Este enlace lleva al video promocional del juego. Está alojado en la plataforma YouTube.

Video demostrativo: https://youtu.be/nApEX9w581E

Este enlace conduce al video demostrativo de cómo funciona el juego y las mecánicas que incluye. Está alojado en la plataforma YouTube.

Repositorio: Engineeringworkshop/TFM (github.com)

El link da acceso al repositorio del proyecto. En él se encuentra el código fuente y todos los assets necesarios para ser cargado con Unity (Versión de Unity: 2020.3.18f1). Está alojado en la plataforma GitHub.

Ejecutable: TFM/Droid Fighters.zip at main · Engineeringworkshop/TFM (github.com)

Enlace al archivo comprimido del juego. Para poder disfrutarlo hay que seguir las instrucciones del apartado 5.2. Está alojado en la plataforma Google Drive.

Nota biográfica del autor.

Pablo Molina Parellada, nacido en Valladolid en 1987. Ingeniero industrial con 6 años de experiencia como ingeniero de ensayos en automoción. Actualmente trabajando como diseñador y programador de entornos de ensayo para bancos. Como hobbies tiene la programación de microcontroladores y videojuegos, diseño de pequeños proyectos de electrónica y/o mecánica.