VNiVERSiDAD D SALAMANCA

CAMPUS DE EXCELENCIA INTERNACIONAL

STRONGEST

Desarrollo de un videojuego con Unity estilo FPS

Memoria

Autor:

Rubén Sánchez Martín

Tutores:

André Filipe Sales Mendes

Juan Francisco De Paz Santana

Gabriel Villarrubia González

Julio de 2022

Certificado de los tutores

D. André Filipe Sales Mendes, D. Juan Francisco de Paz Santana y D. Gabriel Villarrubia González, profesores del Departamento de Informática y Automática de la Universidad de Salamanca.

HACEN CONSTAR:

Que el trabajo titulado “Desarrollo de un videojuego con Unity estilo FPS” ha sido realizado por D. Rubén Sánchez Martín, con el número de documento 70911474J y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado de la Titulación Grado de Ingeniería Informática de esta Universidad.

Y para que así conste a todos los efectos oportunos.

En Salamanca, a 6 de julio de 2022.

SALES MENDES ANDRE FILIPE - Y5601546W
Firmado digitalmente por SALES MENDES ANDRE FILIPE - Y5601546W
Fecha: 2022.07.07 12:34:15 +02'00'

D. André Filipe Sales Mendes

DE PAZ SANTANA, JUAN FRANCISCO (AUTENTICACIÓN)
Firmado digitalmente por DE PAZ SANTANA, JUAN FRANCISCO (AUTENTICACIÓN)
Nombre de reconocimiento (DN): c=ES, serialNumber=76125754D, sn=DE PAZ, givenName=JUAN FRANCISCO, cn=DE PAZ SANTANA, JUAN FRANCISCO (AUTENTICACIÓN)
Fecha: 2022.07.07 12:38:47 +02'00'

D. Juan Francisco de Paz Santana

VILLARRUBIA GONZALEZ GABRIEL - 71932845T
Firmado digitalmente por VILLARRUBIA GONZALEZ GABRIEL - 71932845T
Fecha: 2022.07.07 12:50:32 +02'00'

D. Gabriel Villarrubia González

Resumen

Hoy en día la industria de los videojuegos está en un gran crecimiento y no solo porque cada vez hay muchísimos más videojuegos sino también porque estos están teniendo mucha más repercusión en nuestras vidas. Para ser más explícito en el 2021 el tiempo medio dedicado semanalmente por la población a jugar a videojuegos ha sido de 8,1 h, casi el doble de lo que se consumía en 2013 y por no hablar de que actualmente existen 400 millones de usuario que juegan cada mes a nivel global. Además, palabras como los e-sports que eran casi desconocidas para todo el mundo años atrás, han cogido una gran importancia este último año, generando ganancias de casi $200.000 millones en 2020, superando incluso a la industria del cine.

Pues bien, aprovechando el impacto que pueden tener los e-sports, decidí crear mi proyecto enfocándome en este sector. Intenté buscar de cada juego actual referencia, los puntos fuertes y que más llamaban la atención de los usuarios, basándome en opiniones de usuarios. He realizado una investigación sobre estos puntos fuertes, para poder trabajarlos e implementarlos en mi juego, de manera que pueda obtener un juego final muy completo que pueda llevar a los usuarios a disfrutar sin parar de él, compitiendo.

El juego se basará en la competición de todos los usuarios que participen en él. Creo que es la característica que más tiempo hace disfrutar del juego al usuario, por lo tanto, sin duda la incluiré en mi proyecto. La competición la lograré creando un sistema de puntuación clasificatoria para cada usuario, correspondiendo cada intervalo de puntuación a una división distinta en el juego, de manera que entre amigos y entre todos los jugadores del juego de manera global puedan compararse y empujarse a ser mejores, disfrutando del juego. Para mantener este sistema de puntuación el juego también implementará un sistema de gestión de usuarios para que cada jugador pueda tener una puntuación propia y perdurable (mantenida en una base de datos).

También debe implementar sí o sí el multijugador, que es lo que hace que cada partida del juego sea diferente. Pero aparte de desarrollar un buen sistema de partidas con un buen sistema de gestión de usuarios para los jugadores, he visto de vital importancia incluir un sistema de economía y personalización ya que, de esta manera, permita a los jugadores tener su punto personal en el juego y hacerlo más suyo.

Finalmente, sobre el desarrollo del proyecto he decidido usar la planificación, prácticas y directrices del Proceso Unificado adaptándolo al proyecto de esta memoria.

Palabras claves: E-sports, videojuego, shooter, online, tablas de clasificación, acción, multijugador, Unreal Engine 5

Summary

It is undeniable the rise video games industry is undergoing nowadays. This is happening not just because of the huge and increasing quantity of video games there are, but also because they are having a much greater impact on our lives. As a matter of fact, in 2021 the average time spent weekly playing videogames was 8.1 hours, which almost doubles the stat in 2013. I would also like to point out that there are 400 million users playing on a global level each month, and something such as e-sports, which was highly unknown some years ago, in 2020 generated profits of almost $200 billion (which exceeded cinema industry profits).

Acknowledge the impact e-sport have, I decided to focus on this sector for my project. After an investigation I searched for the strengths of the reference videogames nowadays in order to implement them in my own videogame so the final version is very complete and users can enjoy it non-stop.

The game is about competition of all users who participate in it, which in my view is the characteristic that makes it the most enjoyable. This system of competition is based on a ranking scoring system for each user, in which each scoring interval corresponds to a different division of the game. This way, either friends or any other player can be compared and push themselves to improve. This scoring system is based on a user management system that allows every player to have their own profile.

Another fundamental feature is multiplayer mode, which makes each game unique.

In addition to the outstanding games system and user management system for players, I thought of and implemented an economy and personalization system so players can make of this video game their very own.

Finally, on the development of the project I have decided to use the planning, practices and guidelines of the Unified Process adapting it to the project of this report.

Keywords: E-sports, video game, shooter, online, leaderboard, action, multiplayer, Unreal Engine 5

Tabla de contenido

Certificado de los tutores 3
Resumen 5
Summary 7
Tabla de figuras 11
Índice de tablas 13
1.- Introducción 14
2.- Objetivos 17
2.1.- Gestión de usuarios 17
2.2.- Partidas estilo shooter 17
2.3.- Multijugador 17
2.4.- Online 17
2.5.- Clasificación global 18
2.6.- Gestión de amigos 18
2.7.- Personalizable 18
2.8.- Economía 18
2.9.- Variedad de armas y mapas 18
2.10.- Estadísticas 19
2.11.- Tiempos de respuesta bajos 19
2.12.- Usabilidad 19
2.13.- Portabilidad 19
2.14.- Rendimiento 19
2.15.- Fiabilidad 19
2.16.- Privacidad 20
3.- Estado del arte 21
3.1.- Clash Royale 21
3.2.- Call of duty 23
3.3.- Fornite 24
3.4.- Características comunes 25
4.- Técnicas y herramientas 26
4.1.- Unreal Engine 5 26
4.2.- Visual Paradigm for UML 28
4.3.- Plugin Online Subsystem Steam 29

4.4.- Advanced Session Plugin 30
4.4.- Plugin Playfab 31
4.5.- Microsoft office 32
4.6.- Microsoft Project 32
4.7.- Mixamo 33
4.8.- Visual Studio Code 34
4.9.- Unreal Engine Marketplace y ArtStation 34
4.10.- Lenguajes 35
4.10.1.- Blueprints 35
4.10.2.- JSON 36
4.10.3.- C++ 37
4.10.4.- UML 38
5.- Aspectos relevantes del desarrollo 39
5.1.- Marco de trabajo 39
5.2.- Estimación del esfuerzo 41
5.3.- Planificación temporal 43
5.4.- Especificación de Requisitos de Software 45
5.4.1.- Objetivos del sistema 45
5.4.2 Requisitos de información 47
5.4.3.- Requisitos funcionales 48
5.4.4.- Requisitos no funcionales 55
5.5.- Análisis del Sistema Software 56
5.5.1.- Modelo del dominio 56
5.5.2- Paquete de análisis y de servicio 57
5.5.3.- Clases de análisis y descripción de la arquitectura 58
5.5.4.- Realización de casos de uso en el modelo de análisis 60
5.6.- Diseño del sistema Software 61
5.6.1.- Patrones arquitectónicos y de diseño 61
5.6.2.- Subsistemas de diseño 63
5.6.3.- Clases de diseño 64
5.6.3.1.1 Capa de presentación 67
5.6.3.1.2 Capa de negocio - Singleton 67
5.6.3.1.3 Capa de datos 67
5.6.3.2 Subsistema Partidas 68
5.6.3.2.1 Capa de presentación 69

5.6.3.2.2 Capa de negocio 69
5.6.3.2.3 Capa de datos 69
5.6.4.- Realización de casos de uso del modelo del diseño 70
5.6.5.- Modelo de despliegue 71
5.7.- Implementación 73
5.7.1.- Subsistema Usuario 82
5.7.2.- Subsistema Partida 84
5.7.3.- Subsistema Economía 85
5.7.4.- Subsistema Clasificación 86
5.8.- Pruebas 87
6.- Conclusiones 88
7.- Líneas de trabajo futuras: 92
8.- Referencias 93

Tabla de figuras

Figura 1 Sistema clasificación Clash Royale 22
Figura 2 Ejemplo de interfaz fortnite 24
Figura 3 Captura ingame del juego, dónde se puede observar su apartado colorido y la lejanía del realismo 25
Figura 4 Algunas funciones adicionales del plugin Advanced Sessions 30
Figura 5 Ejemplo de blueprints de Unreal Engine 5 35
Figura 6 Disciplinas del Proceso Unificado 41
Figura 7 Captura EZEstimate del proyecto con los resultados obtenidos para él 42
Figura 8 Captura de Microsoft Project del proyecto 44
Figura 9 Diagrama de paquetes del sistema 49
Figura 10 Diagrama de actores del sistema 50
Figura 11 Diagrama de casos de uso Pt 1/2 53
Figura 12 Diagrama de casos de uso Pt 2/2 54
Figura 13 Diagrama de clases 56
Figura 14 Diagrama de paquetes de análisis 58
Figura 15 Descripción de la arquitectura 59
Figura 16 Ejemplo diagrama de secuencia 60
Figura 17 Patrón arquitectónico de capas 62
Figura 18 Patrón Singleton 62
Figura 19 Subsistemas de diseño 63
Figura 20 Clases de diseño 65
Figura 21 Clases de diseño subsistema Usuarios 66
Figura 22 Clases de diseño del subsistema partidas 68
Figura 23 Ejemplo de realización de caso de uso del modelo del diseño 70
Figura 24 Modelo de despliegue 71
Figura 25 Ejemplo de uso de blueprints de llamadas al servidor y multicast para la sincronización del agachado 73
Figura 26 Llamadas distintas si tiene autoridad o no, multicast o al servidor ... 74
Figura 27 Ejemplo blueprint controlador de evento, comenzar superposición . 74
Figura 28 Blueprint que vincula el evento OnFireProjectile a On Use Item 75

Figura 29 Ejemplo blueprint de evento personalizado con varios parámetros . 75
Figura 30 Funciones de la interfaz 76
Figura 31 Ejemplo 1 implementación función de la interfaz 77
Figura 32 Ejemplo 2 implementación función de la interfaz 77
Figura 33 Ejemplo de diferentes llamadas a las funciones de la interfaz 78
Figura 34 Captura del Game Mode de una partida donde se puede observar parte de la funcionalidad implementada en él, así como las clases vinculadas a él (a la derecha de la captura) 78
Figura 35 Captura del Game State..... 79
Figura 36 Captura del player state 79
Figura 37 Captura del BP_FirstPersonCharacter 80
Figura 38 Captura del BP_PlayerController 80
Figura 39 Ejemplo de construcción del JSON con su request y su correspondiente llamada a la API..... 81
Figura 40 Capturas de pantalla de la interfaz del juego 83
Figura 41 Capturas de pantalla de la partida 84
Figura 42 Capturas de pantalla de la interfaz personalización dónde se puede observar la tienda..... 85
Figura 43 Captura de la interfaz clasificación con los mejores jugadores, y de la interfaz perfil con las estadísticas del usuario 86

Índice de tablas

Tabla 1 Ejemplo Objetivo del sistema 46
Tabla 2 Ejemplo de requisito de información..... 47
Tabla 3 Ejemplo de actor del sistema 50
Tabla 4 Ejemplo definición caso de uso 51
Tabla 5 Ejemplo de requisito funcional..... 55

1.- Introducción

En esta memoria voy a documentar mi trabajo de fin de grado de Ingeniería Informática, el cual va a ser un juego llamado “STRONGEST”. El trabajo ha sido realizado por mí, Rubén Sánchez Martín, durante los meses de enero a julio de 2022 y he recibido el apoyo como tutores de D. André Filipe Sales Mendes, D. Juan Francisco de Paz Santana y D. Gabriel Villarrubia González.

Como he iniciado en el resumen, en la actualidad claramente estamos viviendo un auge en los videojuegos ya que el público de estos y el uso de ellos está en constante crecimiento. Los videojuegos están evolucionando por muchísimas ramas distintas, temáticas, objetivos, plataformas… Por lo tanto, a la hora de desarrollar mi juego, del que trata este proyecto, he intentado investigar las características más destacables y llamativas de cada uno de los distintos videojuegos más punteros en la actualidad.

A la hora de crear mi juego no he querido separar del sector de los e-sports, ya que creo que es el sector más llamativo y que más gente mueve dentro de la industria de videojuegos, por ello he decidido basar mi juego en la competición entre todos los jugadores globales.

Basándome en la competición propia de un e-sport y en las características más destacables de cada uno de los juegos punteros actuales, he decidido implementar un juego con una gestión de usuarios buena y completa, incluyendo estadísticas de cada uno, una gestión de amigos, un sistema de partidas clasificatorias que determinen unos puntos clasificatorios que acaben siendo la base de la competición en el juego mediante su comparación y un sistema de partidas personalizadas que permita a los jugadores disfrutar del juego de un modo más relajado.

Las partidas clasificatorias acabarán construyendo un sistema de clasificación en el que todos los jugadores puedan compararse con otros, especialmente los usuarios que son amigos entre sí.

Para dotar de recompensas a los jugadores altos de la clasificación se implementará un sistema de economía que recompense a estos jugadores a través de monedas virtuales en el juego. Estas permitirán a los jugadores comprar distintos objetos para personalizar su experiencia en el juego con un toque propio.

Además de todo lo explicado, como un buen juego shooter, debemos dar mucha importancia a las partidas multijugador. Para ello he de desarrollar un sistema de sincronización y de comunicación eficiente que dote al juego de una correcta replicación con el objetivo de que posea una buena experiencia de usuario. Claramente, además de la sincronización, he de crear un sistema de partidas completo con distintas armas, mapas y otras funcionalidades que permitan desarrollar enfrentamientos entre distinto número de jugadores en partidas.

Esta memoria tendrá la siguiente estructura:

- Objetivos: Conjunto de objetivos planteados para el proyecto y que debe cumplir al finalizar su desarrollo.
- Estado del arte: Descripción de diferentes productos e investigaciones relacionados con el ámbito de nuestro proyecto
- Técnicas y herramientas: Conjunto de técnicas y herramientas utilizadas en el desarrollo del proyecto.
- Aspectos relevantes del desarrollo: Descripción de los aspectos más relevantes del desarrollo de nuestro proyecto.
- Conclusiones: Conclusiones personales obtenidas al finalizar el desarrollo de este proyecto.
- Líneas de trabajo futuras: Descripción y avance de las posibles actualizaciones y mejoras que se podrían dar al proyecto en un futuro próximo.

Como documentación extra a esta memoria se entregarán los siguientes anexos:

- Anexo I Planificación temporal: Documentación de la estimación del esfuerzo y la planificación temporal.
- Anexo II Especificación de requisitos de software: Documentación de la especificación de requisitos del software del sistema.
- Anexo III Análisis del sistema: Documentación del análisis de requisitos del sistema.

- Anexo IV Diseño del sistema: Documentación del diseño del sistema.

- Anexo V Documentación técnica: Documentación técnica del código y de los diferentes archivos del proyecto para que puedan ser fácilmente comprendidos y leídos por otros desarrolladores o programadores.

- Anexo VI Manual del usuario: Guía para el uso e interacción con el juego por parte del usuario mostrando todas las funcionalidades de este y cómo utilizarlas.

2.- Objetivos

En este cuarto apartado se va a exponer los distintos objetivos que debe cumplir el sistema cuando se finalice el desarrollo del proyecto.

2.1.- Gestión de usuarios

El sistema deberá permitir a los distintos jugadores crear sus propios usuarios con sus datos y permitirles poder acceder a ellos en el juego en cualquier momento.

2.2.- Partidas estilo shooter

El sistema deberá ofrecer a los distintos jugadores participar en partidas controladas del género de disparos, en el que se enfrenten dos equipos y luchen por llevarse la victoria por medio de eliminaciones de jugadores del equipo contrario mediante el uso de armas.

2.3.- Multijugador

El sistema deberá permitir a los jugadores jugar con y contra otros jugadores que posean también el juego. De esta manera crearemos un juego en el que cada partida será diferente e imprevisible por la propia condición humana.

2.4.- Online

El sistema deberá ofrecer la opción a los jugadores de jugar con y contra otros jugadores del juego sin que necesariamente se encuentren conectados a la misma red o usen el mismo dispositivo.

2.5.- Clasificación global

Con el objetivo de crear la característica de competición en el juego, el sistema deberá implementar un sistema de puntuación clasificatoria para todos los jugadores que permita determinar su nivel de destreza en el juego. Además de que estos puntos clasificatorios de los jugadores deben poder ser accedidos y comparados por cualquier jugador en el juego para generar la competición y el empuje a la mejora de los distintos jugadores.

2.6.- Gestión de amigos

El sistema deberá implementar la funcionalidad necesaria para que cada usuario pueda tener su propia lista de amigos, añadiéndolos o quitándolos mediante algún sistema de identificación único de usuario. Este objetivo se tiene con el propósito de la competición para que los jugadores puedan compararse con jugadores conocidos por ellos mismo y provocar la competición entre ellos.

2.7.- Personalizable

El sistema deberá implementar de algún modo la personalización dentro del juego de manera que los usuarios puedan dar su toque personal a su personaje en el juego.

2.8.- Economía

Para favorecer la competición el sistema deberá implementar algún sistema de recompensas que le otorgue a los jugadores con más puntos ciertos premios en el juego, mediante una moneda virtual. Eso sí, para mantener un sistema de competición equilibrado, nunca estos premios deben generar algún tipo de ventaja en el desarrollo de las partidas en el juego.

2.9.- Variedad de armas y mapas

Para no caer en la monotonía y por lo tanto el aburrimiento del jugador, se deberá implementar más de un mapa y arma usable por el jugador en el juego.

2.10.- Estadísticas

Con el objetivo de que cada usuario pueda observar su progresión y destreza en el juego, el sistema deberá implementar la funcionalidad de mantenimiento y actualización de estadísticas de jugadores en partidas.

2.11.- Tiempos de respuesta bajos

Para la perfecta experiencia de usuario de los jugadores, el sistema deberá trabajar siempre con tiempos de respuesta bajos a las peticiones o controles del usuario, sobre todo en partida, pero también en toda la ejecución del juego.

2.12.- Usabilidad

El sistema deberá mostrar una interfaz lo suficientemente sencilla para que todos los jugadores puedan acceder a toda la funcionalidad del juego de forma fácil y eficiente.

2.13.- Portabilidad

El sistema deberá poder implementarse en los grandes sistemas operativos de ordenadores tanto Windows como Mac como Linux.

2.14.- Rendimiento

Con el objetivo de no ser un juego muy exigente con el ordenador para así ser más asequible para todo el público, el sistema deberá consumir la menor cantidad de recursos posibles optimizando sus operaciones.

2.15.- Fiabilidad

El sistema deberá tener el suficiente control de errores para no llegar a situaciones inconsistentes ni perjudicar la experiencia de usuario.

2.16.- Privacidad

El sistema bajo ningún concepto debe permitir a terceros acceder a los datos personales de cada usuario en el sistema.

3.- Estado del arte

En este apartado voy a llevar a cabo la descripción de diferentes productos e investigaciones relacionados con el ámbito de nuestro proyecto.

Como ya había adelantado, para mi juego quería buscar las características más destacables de los videojuegos más jugados en los últimos años, y estas son las que voy a explicar en este apartado.

3.1.- Clash Royale

Nadie puede negar que éste ha sido uno de los juegos más jugados en la telefonía móvil desde su lanzamiento en 2016, pues bien, este juego debe tener alguna característica que pueda exportar a mi juego para así dotarlo de un mayor valor.

Pues bien, yo creo, desde mi perspectiva de usuario recurrente del juego y conocedor de la opinión de gente de mi alrededor que también juega a él, que las características más destacables y que me gustaría implementar en mi juego serían las siguientes:

- Partidas rápidas: Las partidas de este juego son cortas de una media de cuatro minutos. Esta característica dota la partida de dinamismo, en apenas 10 minutos un jugador puede disputar tres partidas, contra tres personas distintas lo que hace que viva situaciones distintas y no se sienta atrapado en una sola, que si puede ir perdiendo se la haga larga, o que simplemente sienta aburrimiento por estar experimentando siempre lo mismo. Estas dos últimas características es probable que desemboque en una mala experiencia. Además, estas partidas hacen que los jugadores pueda disputarlas en cualquier momento, tengan el tiempo disponible que tengan, ya que con tan solo tres minutos de su tiempo pueden disputar una.

- Sistema de clasificación: Clash Royale ofrece un sistema de clasificación de manera que todos los jugadores puedan observar la clasificación global del juego, así como la de su país y amigos. Por lo tanto, se puede deducir fácilmente que uno de los motivos que empujan a los jugadores a seguir jugando es la de verse en los puestos altos de las clasificaciones. Además, hay que añadir que estas clasificaciones para que sean más dinámicas se reinician cada mes, proporcionando a todos los jugadores

las mismas posibilidades. En la Figura 1 se puede observar el sistema de clasificación del Clash Royale exponiendo los tres jugadores con mayor puntuación y el tiempo antes del reinicio de la misma.

Figura 1 Sistema clasificación Clash Royale

- Partidas clasificatorias 1vs1: He deducido que las partidas sean 1 vs 1 produce que el jugador piense que depende sólo de él la partida lo que hace que no desvía el motivo de sus derrotas a otros compañeros que podría tener en otros juegos y que acaben en una mala experiencia.

Estas tres características, con todo lo que conllevan, las tendré muy en cuenta a la hora de desarrollar el juego.

3.2.- Call of duty

No podemos desarrollar un shooter sin olvidar al que ha sido el rey histórico de esta categoría, el Call of Duty.

La característica destacable de este videojuego claramente ha sido su sistema de partidas con lo que implementa, así como puede ser los tiempos de asesinatos, características de las armas, mapas… Por lo tanto, para el videojuego del proyecto voy a tener en cuenta lo siguiente:

- Sistema de arma: Armas que producen asesinatos a corta distancia en escasos segundos aproximadamente entre un segundo y dos. Esto lo consigue mediante armas que dependiendo de su cadencia maten de 3 a 5 balas disparadas en los segundos expuestos. Estas estadísticas de las armas las exportaré a las propias en mi juego.

- Mapas: Los mapas de esta saga han sido siempre pequeños o medianos para generar más combates entre los jugadores y que haya más acción. También suelen tener en su mayoría tres caminos y una zona central donde se centre la mayoría de la acción. Estas referencias las tendré en cuenta en el juego.

- Modo de juego Duelo por Equipos: Ha sido el modo de juego base para esta saga por lo tanto también será la base para los enfrentamientos del juego.

3.3.- Fornite

Fornite ha sido el videojuego más jugado en todas las plataformas y edades y por ello tiene un montón de características destacables, pero yo en este momento me voy a centrar en una.

Aparte de la construcción, que es lo que le diferencia a Fornite del resto de juegos y que le hace llegar a jugadores de todas las edades y plataformas. Creo que lo que cumple el objetivo es su estética, tanto de personajes como de menús como el mapa. Toda la estética es muy colorida y se acerca bastante a la categoría cartoon (no aparece nada realista 100%, no hay sangre, ni humanoides que se acerquen a las categorías realistas). Este apartado estético, el cual lo voy a ejemplificar mediante capturas del juego en las siguientes imágenes (Figuras 2 y 3), lo voy a adaptar a mi juego con el objetivo explicado.

Figura 2 Ejemplo de interfaz fortnite

Figura 3 Captura ingame del juego, dónde se puede observar su apartado colorido y la lejanía del realismo

3.4.- Características comunes

La mayoría de los juegos también poseen las siguientes características que los hacen mucho más entretenidos y mejoran la experiencia de usuario:

- Gestión de amigos: Cada usuario pueda tener un conjunto de amigos en el juego con los que poder jugar y compararse.

- Apartado decorativo y personalizable: Para dotar al juego y al jugador de un punto personal propio suyo.

- Economía y tienda: Los juegos más destacables suelen tener su propia moneda del juego con la que comprar contenido en él. Pues bien incluiré esta característica también en el juego ya que creo que lo hace más llamativo y es una funcionalidad que a los jugadores le gustará mucho.
  A pesar de permitir a los jugadores comprar contenido dentro del juego, este contenido nunca generará una ventaja en él porque creo que esto es una de las bases de la competición igualitaria y sana, no caer en el pay to win.

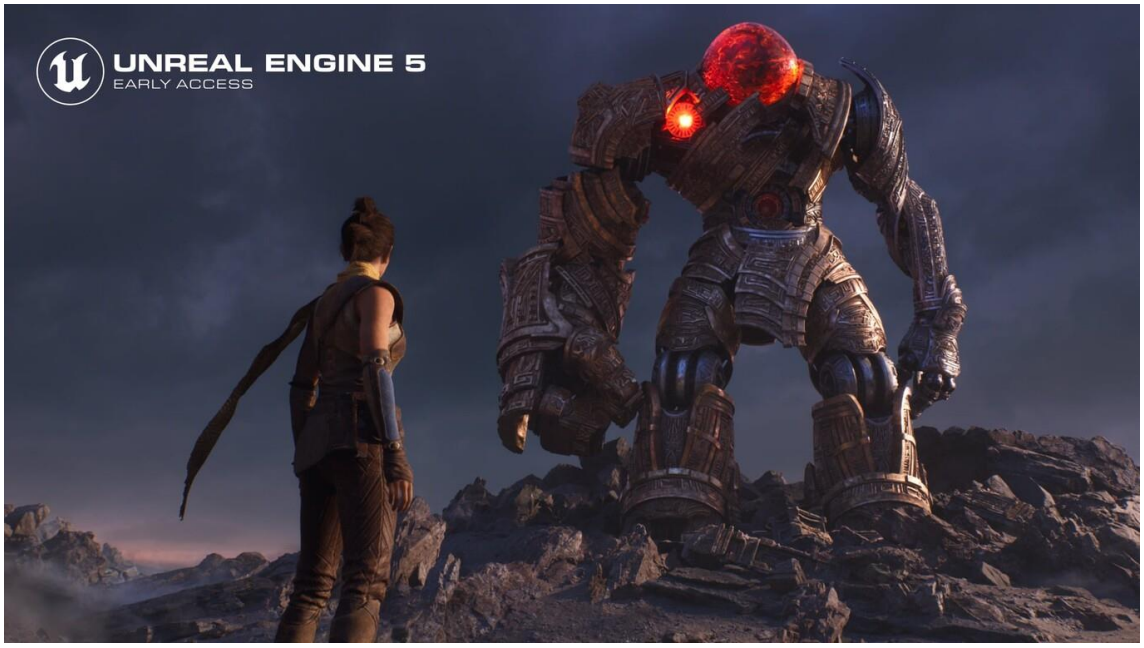
4.- Técnicas y herramientas

4.1.- Unreal Engine 5

Unreal Engine [1] [2] [3] es un entorno de desarrollo que proporciona al programador todas las herramientas necesarias para que pueda éste construir su propio juego o simulación ya que incluye incluso editor de vídeo, estudio de sonido, código y renderización de animaciones.

Unreal Engine también es conocido por ser uno de los motores de juego más populares y usados en la actualidad, pertenece a la compañía Epic Games. Su funcionalidad está basada en código C++, pero también nos permite usar blueprints haciendo el código mucho más sencillo y visual, y también, como opinión, permite el desarrollo más rápido y ágil, lo que se traduce en mayor funcionalidad y más comprensible en el código.

Dicho motor de juego tiene una gran cantidad de utilidades, aunque nosotros nos centraremos en su uso para desarrollar videojuegos, en la actualidad es utilizado por una gran cantidad de empresas y usuarios de diversos campos como de arquitectura, ingeniería, medicina, realidad virtual, desarrolladores de aplicaciones… Hasta tal punto ha llegado a ser potente y bueno que la NASA está utilizando el motor gráfico para crear entornos en los que entrenar a los futuros astronautas.



Este motor de juego compite de forma directa con Unity, pero para el desarrollo de este proyecto he decidido usar Unreal por las siguientes razones. La primera es que permite usar la tecnología de blueprints lo que me permite organizar el código en nodos haciendo el proceso de programar y crear el videojuego de una forma muy sencilla y visual. La segunda sería la nueva versión de Unreal Engine que promete cambiar la industria de los videojuegos Unreal Engine 5. Unreal Engine 5 ofrece con respecto a sus versiones anteriores un Editor rediseñado, un mejor rendimiento, herramientas de animación amigables con los artistas, un conjunto de herramientas de creación y edición de mallas ampliadas, trazado de ruta mejorado, entre otros. También ofrece nuevas herramientas de autor y flujos de trabajo fáciles de usar para que los desarrolladores aceleren su proceso creativo. Por todo lo anterior me generaba mucha curiosidad y ganas de utilizar y sacar el máximo a Unreal Engine 5.

Quiero añadir que salió la versión oficial de Unreal Engine 5 el 6 de abril de este mismo año, por lo tanto, al principio de este proyecto estuve trabajando con la beta y como contrapartida de usar esta versión existen menos assets y documentación, pero esto también dotará de más valor al proyecto ya que es uno entre pocos juegos actuales que usan Unreal Engine 5.

4.2.- Visual Paradigm for UML

Visual Paradigm [4] [5] es una herramienta de las conocidas como CASE (Ingeniería de Software Asistida por Computación) la cual proporciona un conjunto de asistencias para la creación y el consecuente desarrollo de diferentes sistemas y programas informáticos, abarca las etapas de la planificación, el análisis, el diseño e incluso la generación del código fuente del sistema o programa y su correspondiente documentación.

Esta herramienta tiene el objetivo de ayudar a los distintos equipos de desarrollo de software en la tarea de captura de los requisitos de software y su transformación en diseños completos y precisos, lo que permite por consecuencia a los desarrolladores implementar el software óptimo para los requisitos establecidos.

En mi proyecto la he utilizado para la realización de todos los diagramas de modelado UML como diagramas de casos de uso, de clase, de secuencia…

4.3.- Plugin Online Subsystem Steam

Online Subsystem Steam [6] es un módulo para Unreal Engine que permite para los videojuegos creados por este motor crear un enlace con Steam y así con las cuentas de usuario de los jugadores de Steam. Su principal utilidad (del apartado gratuito utilizado) es dotar al juego de la característica Online con su soporte de sesiones correspondiente.

Este módulo lo he utilizado en el desarrollo del proyecto para desarrollar en el apartado multijugador las diferentes partidas online, por el soporte de sesiones ofrecido por este. Más concretamente ha dotado al juego de la posibilidad de implementar la funcionalidad para todos los jugadores (instancias del juego) de hostear, buscar y unirse a partidas propias mientras el jugador esté conectado a Steam.

4.4.- Advanced Session Plugin

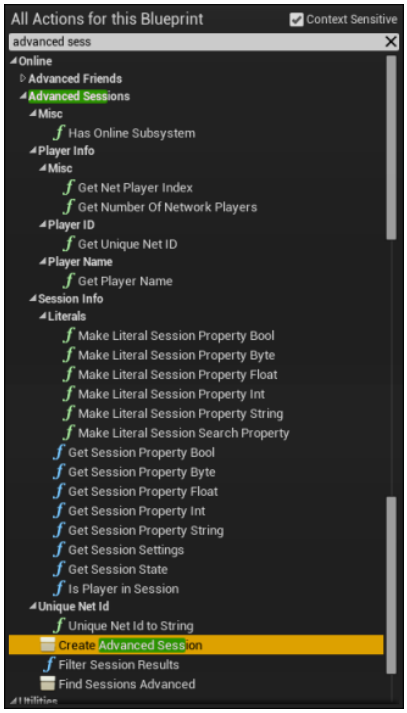


Figura 4 Algunas funciones adicionales del plugin Advanced Sessions

Advanced Sessions Plugin [7] es un módulo para Unreal Engine y Unity que permite dotar de más funciones específicas de Steam a los videojuegos creados por dichos motores (algunas de ellas las muestro en la Figura 4).

No es un plugin que podamos encontrar en la UE Marketplace, está desarrollado en C++, pero permite la creación de blueprints que implementen las funciones desarrolladas en dicho lenguaje. Como es desarrollado en C++ y no se encuentra en la Marketplace, para su uso he tenido que transformar el proyecto a uno mixto C++ y Blueprints, además de descargar y compilar todos los archivos fuente de Unreal Engine de github para compilarlo con el plugin.

Este plugin nos ha permitido en nuestro proyecto crear sesiones con nombres, características (mapa, número de jugadores actuales y máximos, ping…) y tags (ranked o no), lo que nos ha posibilitado distinguir desde la búsqueda unas sesiones de otras para encontrar las específicas interesadas en el momento de la ejecución.

La parte negativa del Steam Subsystem y este de plugin en concreto es que para poder explotarlo completamente se tiene que registrar el juego en la plataforma de Steam lo que conllevaría un precio muy alto para el proyecto.

4.4.- Plugin Playfab

Plugin disponible en la UE Marketplace para Unreal Engine que permite conectar con Azure Playfab el juego creado. Playfab [8] permite a los desarrolladores de videojuegos instaurar y administrar juegos en tiempo real con una sola plataforma. Playfab proporciona una gran cantidad de funcionalidad como servidores para juegos, bases de datos personalizables y otras herramientas en tiempo real para facilitar el desarrollo y mantenimiento de los videojuegos. Aunque toda esta funcionalidad no es al completo gratis y por lo tanto no se ha podido explotar completamente.

Mediante este plugin y haciendo uso de interfaz web hemos podido desarrollar un soporte para los jugadores, gestión de usuarios, gestión de base de datos, gestión de economía, gestión de catálogo, gestión de inventario, gestión de ítems, gestión de una clasificación y gestión de premios. No todos ellos han sido proporcionados directamente por Playfab, pero haciendo uso de sus herramientas y escribiendo un código extra he podido explotar toda la funcionalidad mencionada.

4.5.- Microsoft office

Para el desarrollo de toda la documentación, memoria y anexos, de este proyecto se ha utilizado en su gran mayoría Microsoft Office, sobre todo el programa de Word.

4.6.- Microsoft Project

Microsoft Project [9] [10] es un software de que permite a los desarrolladores la gestión de sus proyectos y planificación de proyectos creado por Microsoft. Project permite afrontar desde pequeños proyectos a grandes, permitiendo a los desarrolladores el planeamiento de proyectos, la asignación de recursos a tareas, la realización de un seguimiento completo del progreso del desarrollo del proyecto, la administración de presupuestos y el análisis de las distintas cargas de trabajo fácilmente.

Mediante la herramienta de Microsoft Project he podido incluir en la documentación el Anexo I, correspondiente a la estimación del esfuerzo y a la planificación temporal del proyecto, para así llevar a cabo la elaboración de un calendario de trabajo, además de la distinción de las tareas y subtareas con su correspondiente desarrollo de hitos, recursos empleados, duraciones, dependencias y resultados.

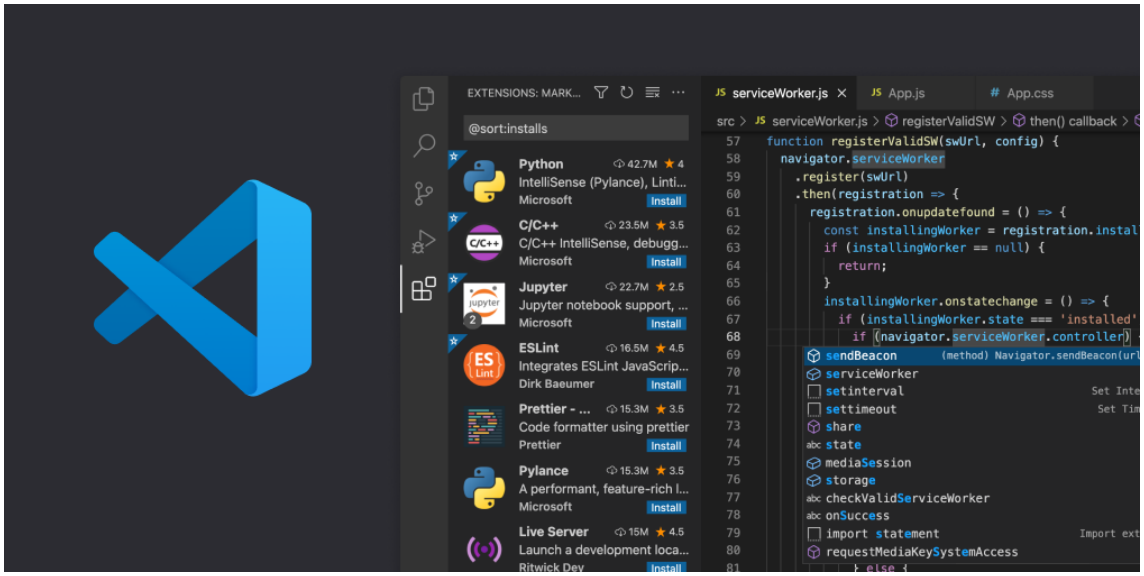
4.7.- Mixamo

Mixamo [11] es una web que nos ofrece una gran cantidad de animaciones para nuestros personajes en el juego, además de personajes listos para usar de alta calidad, en 3D y de diversos tipos, temáticas y propósitos.

Estas distintas animaciones y personajes nos permite exportarlas a diferentes plataformas como Unity y Unreal Engine (mi caso).

En este proyecto lo he utilizado para obtener las skins de los personajes, así como para obtener animaciones adicionales a las ya proporcionadas por defecto.

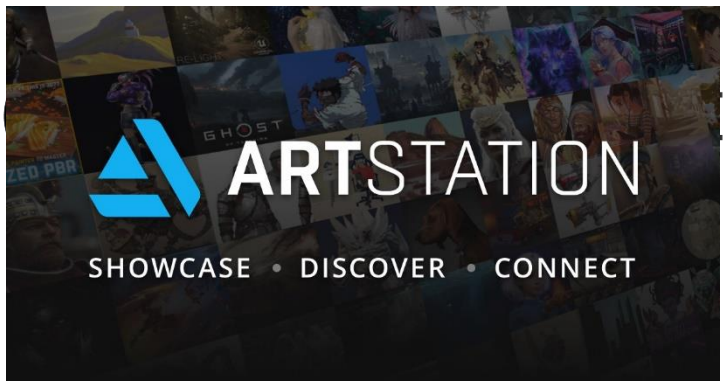
4.8.- Visual Studio Code



Visual Studio Code [12] es un IDE íntegro para desarrolladores sobre todo de .NET y C++. Está equipado con un gran número de herramientas y capacidades para enaltecer y mejorar todas las etapas del desarrollo de software.

En este proyecto ha sido utilizado para recompilar y manejar el código de Unreal Engine y del plugin de Advanced Sessions, así como para modificar distintos archivos de configuración para obtener el resultado deseados del soporte Online de Steam

4.9.- Unreal Engine Marketplace y ArtStation



ArtStation [13] y UE Marketplace [14] son dos páginas web dónde se pueden obtener de manera gratuita y de pago assets digitales para proyectos 3D.

En nuestro proyecto han servido para obtener los assets que no teníamos por defecto incluyendo mapas y armas.

4.10.- Lenguajes

En este apartado voy a explicar los distintos lenguajes utilizados en el proyecto en las distintas partes.

4.10.1.- Blueprints

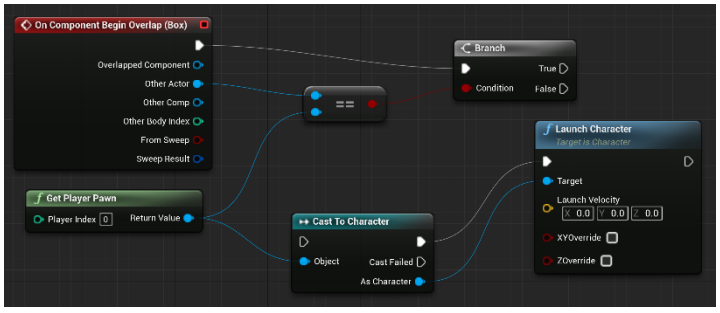


Figura 5 Ejemplo de blueprints de Unreal Engine 5

El sistema de blueprints [15] (ejemplo en la Figura 5) es una forma de programación diferente a la programación habitual en código, permite desarrollar un sistema o programa de una manera visual más visual. Uno de sus objetivos principales es que cualquier desarrollador o usuario pueda crear cualquier tipo de función desde complejas a sencillas y también otros elementos interactivos sin haber estudiado ni por lo tanto tener conocimientos amplios en programación. Este último objetivo lo consigue utilizando un sistema de nodos.

El sistema de nodos presenta una gran flexibilidad y potencia dotando a los desarrolladores del acceso un amplio abanico de posibilidades y capacidades

que ofrece Unreal Engine que, si tuvieran que desarrollarse mediante código requeriría el desarrollo por parte de usuarios expertos en programación, es decir permite crear sistemas y programas complejos a usuarios no tan experimentados mediante un sistema gráfico simple e intuitivo.

El uso del lenguaje blueprint ha sido el más usado en la implementación, ya que es el que hemos utilizado en la construcción del juego en Unreal Engine. Ha sido utilizado para el control de los diferentes eventos, llamadas, intercambios de mensajes, implementación de la funcionalidad específica, cálculos…

4.10.2.- JSON

JSON [16] que significa JavaScript Object Notation es un formato para la información que nos permite para guardarla e intercambiarla manteniendo la característica de que cualquier persona la puede leer. Los archivos con este formato albergan únicamente texto y se caracterizan por el uso de la extensión ".json".

Una definición más concreta de JSON sería que es un formato que almacena información estructurada y cuya principal utilidad consiste en permitir transferir datos entre servidores y clientes.

JSON es una alternativa más simple que un formateo de información más conocido como es XML, aunque los dos poseen funciones parecidas.

El uso del lenguaje JSON se utiliza para las llamadas a la API de Playfab desde el juego proponiéndoles los parámetros con las llamadas en dicho lenguaje.

4.10.3.- C++

C++ [17] es un lenguaje de programación que nace a partir de la extensión del lenguaje C con el objetivo de permitir la manipulación de objetos. Debido a su gran potencia se ha convertido en uno de los lenguajes de programación más utilizados en los últimos años.

Las ventajas más destacables de este lenguaje son:

- Alto rendimiento: El alto rendimiento es una de sus ventajas principales. Esto es debido a la posibilidad que ofrece de hacer llamadas directas al sistema operativo ya que se trata de lenguaje compilado para cada plataforma. También contiene una gran cantidad de parámetros para la optimización y su integración es de forma directa con el lenguaje ensamblador.

- Lenguaje actualizado: El lenguaje está siendo actualizado continuamente y esto les permite a los desarrolladores que lo usan crear, relacionar y operar con datos complejos y al uso de diferentes patrones de diseño para su proyecto o programa.

- Multiplataforma: Esto significa que se puede ejecutar en cualquier dispositivo o sistemas operativo. Lo que conlleva a que el mismo código puede ejecutarse en Windows, Android, iOS...

- Extendido: Este lenguaje está muy extendido como he venido indicando. Un gran porcentaje de proyectos, programas y sistemas están desarrollados al completo o de manera parcial en este lenguaje de programación (incluyendo navegadores web, sistemas operativos, bases de datos, incluyendo MySQL que está escrito en C++).

El lenguaje C++ se ha utilizado en el proyecto para modificar el uso de los diferentes archivos de configuración, ya que convertimos nuestro proyecto de Unreal en uno Mixto; Blueprints y C++. Además, lo utilizamos para la recompilación del proyecto desde el código fuente de Unreal Engine 5.

4.10.4.- UML

UML [18] significa Lenguaje Unificado de Modelado, es un lenguaje de modelado visual común el cual posee características tanto semánticas como sintácticas amplias para el desarrollo y la exposición de la arquitectura, el diseño y la implementación de todo tipo de sistemas software, tanto en su correspondiente estructura como en su comportamiento.

UML posee diligencias adicionales a las correspondientes al desarrollo de software como es en el flujo de procesos en la fabricación.

Quiero destacar también que no es un lenguaje de programación.

Ha sido utilizado en este proyecto a la vez que Visual Paradigm por lo tanto ha sido utilizado para la realización de todos los diagramas de modelado UML como diagramas de casos de uso, de clase, de secuencia…

5.- Aspectos relevantes del desarrollo

En este apartado voy a llevar a cabo la descripción de los aspectos más relevantes del desarrollo de nuestro proyecto.

5.1.- Marco de trabajo

Para el desarrollo del presente proyecto hemos seguido el marco de trabajo del Proceso unificado [19].

El proceso unificado es El Proceso Unificado es más que un simple proceso, es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos.

El proceso unificado posee las siguientes características:

- Está basado en componentes
- Utiliza UML
- Es un proceso conducido por casos de uso
- Está centrado en la arquitectura
- Es iterativo e incremental

El Proceso Unificado se repite a lo largo de una serie de ciclos de desarrollo que constituyen la vida de un sistema. Cada ciclo de desarrollo concluye con una versión entregable del producto y consta de cuatro fases:

- Inicio: Se define el alcance del proyecto y se desarrollan los casos de negocio
- Elaboración: Se planifica el proyecto, se especifican en detalle la mayoría de los casos de uso y se diseña la arquitectura del sistema.
- Construcción: Se construye el producto
- Transición: El producto se convierte en versión beta. Se corrigen problemas y se incorporan mejoras sugeridas en la revisión

Otro concepto clave del proceso unificado son los hitos, son puntos de control en los cuales los

participantes en el proyecto revisan el progreso del proyecto. Se pretende:

- Sincronizar las expectativas y la realidad
- Identificar los riesgos
- Se evalúa la situación global del proyecto

El siguiente concepto clave es el de iteración, que es otro apartado fundamental dentro del Proceso Unificado. Una iteración es una secuencia de actividades con un plan establecido y unos criterios de evaluación, cuyo resultado es una versión ejecutable no orientada a la entrega.

Volviendo a los hitos y con las iteraciones explicadas pueden ser de varios niveles:

- Hitos principales al final de cada fase
- Hitos secundarios final de cada iteración

Por último, procedo a explicar el concepto de disciplinas. Las disciplinas o flujos de trabajo organizan las actividades fundamentales de gestión y desarrollo del proyecto:

- Disciplinas de desarrollo: Requisitos, análisis, diseño, implementación, pruebas...
- Disciplinas de gestión o soporte: Gestión de proyecto, gestión de configuraciones, entorno, evaluación...

Podemos observar las disciplinas, fases e iteraciones en la siguiente imagen (Figura 6):

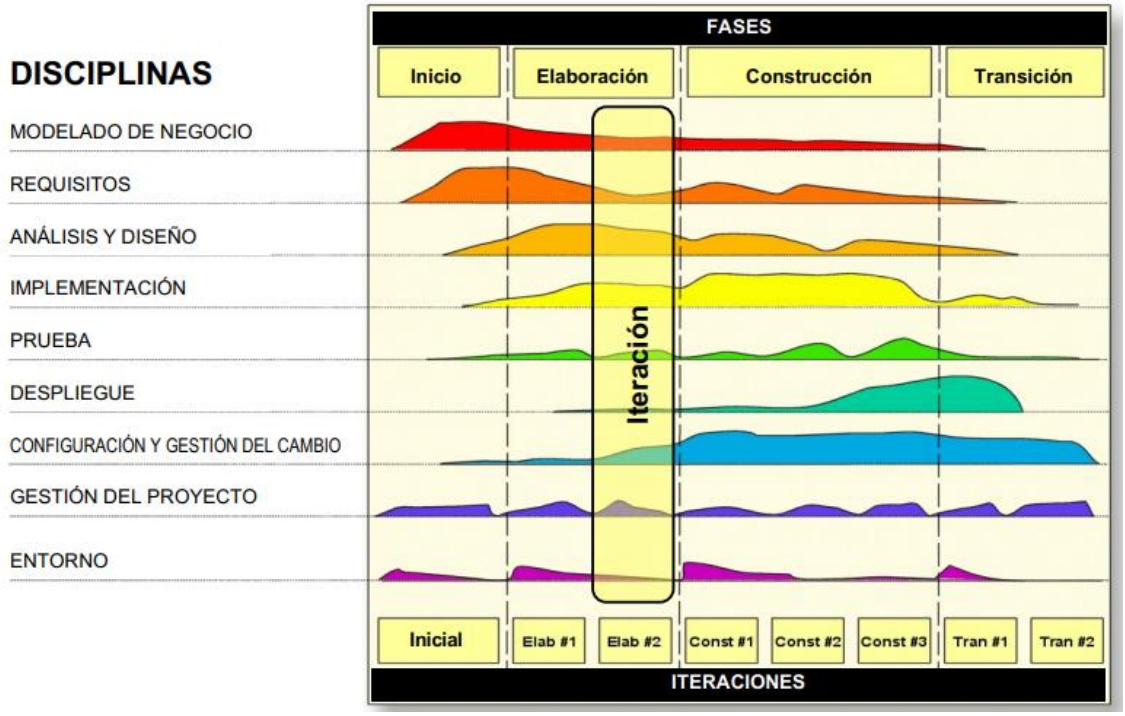


Figura 6 Disciplinas del Proceso Unificado

5.2.- Estimación del esfuerzo

La estimación del esfuerzo de este proyecto la realizaré después de la especificación de requisitos para tener todos los parámetros que participan en ella de un modo definido y claro haciendo que la estimación sea mucho más precisa.

Para comentar por adelantado la estimación del esfuerzo, antes del Anexo I donde se explicará con mayor profundidad se exponen las siguientes aclaraciones:

La estimación del esfuerzo que es la relación entre tiempo y personal necesario para desarrollar el proyecto, la mediré usando la unidad meses de persona.

Usaré la métrica puntos de caso de uso (UCP, Use Case Points) que nos permitirá evaluar la funcionalidad representada en forma de casos de uso y así estima el esfuerzo de desarrollo considerando actores, escenarios y factores técnicos y de entorno.

Para desarrollar y obtener los resultados de la estimación del esfuerzo he usado la herramienta EZEstimate, con la cual podemos introducir los parámetros siguientes:

- UUCP (Unadjusted Use Case Points): Corresponderá a la suma de las dos siguientes
  - UUCW (Unadjusted Use Case Use Weight)
  - UAW (Unadjusted Actor Weight)
- TCF (Technical Complexity Factor)
- ECF (Environment Complexity Factor)

Aplicando EZEstimate las distintas fórmulas correspondientes sobre ellos obtendremos los siguientes resultados (Figura 7):

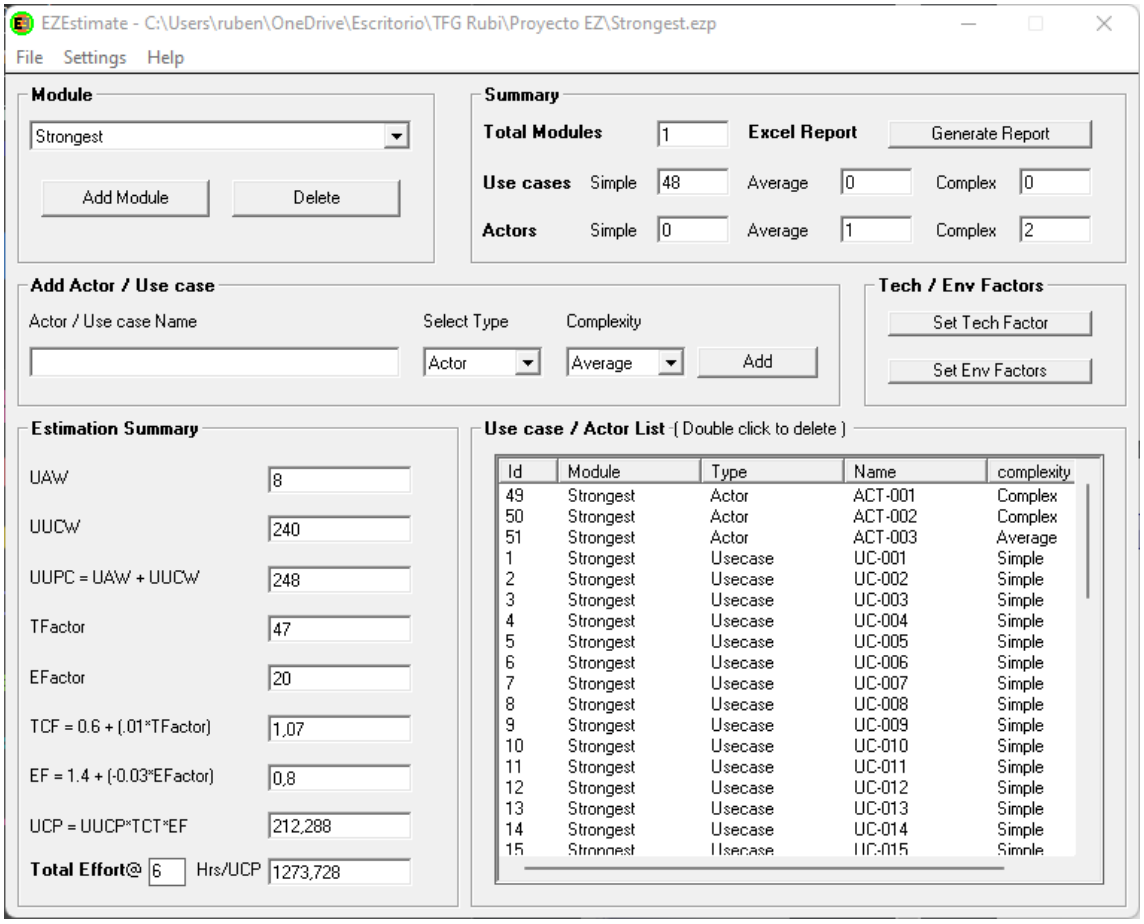


Figura 7 Captura EZEstimate del proyecto con los resultados obtenidos para él

5.3.- Planificación temporal

La planificación temporal [20] conlleva la elaboración de un calendario de trabajo, además de la distinción de las tareas y subtareas con su correspondiente desarrollo de hitos, recursos empleados, duraciones, dependencias y resultados.

La planificación temporal tiene como objetivos finales:

- Definir todas las tareas del proyecto, identificar las que son críticas y hacerles un seguimiento para detectar de inmediato posibles retrasos.
- Distribuir el esfuerzo estimado a lo largo de la duración prevista del proyecto.
- Proporcionar una guía, seguimiento y orientación de las tareas del proyecto.
- Proporciona una estimación de la duración del proyecto

Para conseguir la planificación temporal en nuestro proyecto se ha utilizado el programa Microsoft Project, herramienta muy útil y sencilla para este objetivo.

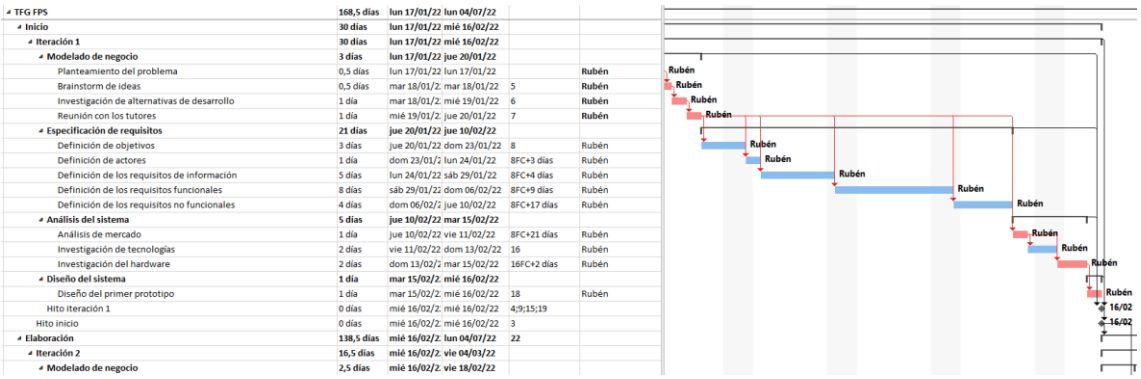
Las tareas que forman parte del desarrollo del proyecto y por lo tanto, de la planificación temporal se encuentran divididas en iteraciones y fases cuyos hitos finales de cada una serán las siguientes:

- Hitos de fases:
    - Hito Inicio: Definición de actores, objetivos y la gran mayoría de requisitos. Además, se investiga sobre alternativas, tecnología y hardware para el desarrollo del proyecto.
    - Hito Elaboración: Definición completa del análisis y especificación de los requisitos. Además, se ha empezado a implementar las Partidas y se ha comenzado a diseñar el sistema.
    - Hito Construcción: Diseño completo del sistema y la implementación completa de cada uno de los componentes.
    - Hito Transición: Despliegue del sistema resultado, corrección de los últimos fallos y completamiento de la documentación final del proyecto

- Hitos Secundarios:

- Hito iteración 1: Coincide con el hito de Inicio, ya explicado.
- Hito iteración 2: Terminación de la especificación de requisitos a falta del último refinamiento de casos de uso, se definen los paquetes de análisis y se empieza a investigar sobre el diseño. Finalización del Anexo I.
- Hito iteración 3: Coincide con el hito de Elaboración, ya explicado.
- Hito iteración 4: Completamiento de la realización casos de uso de diseño y comienzo de la implementación y pruebas del componente gestión de usuarios
- Hito iteración 5: Coincide con el hito de Construcción, ya explicado.
- Hito iteración 6: Coincide con el hito de Transición, ya explicado.

En la siguiente imagen (Figura 8) podemos observar una captura de la planificación temporal del proyecto en Microsoft Project donde se puede observar una porción de la distribución de las tareas y la duración estimada del proyecto:



TFG FPS	168,5 días	lun 17/01/22 lun 04/07/22		
Inicio	30 días	lun 17/01/22 mié 16/02/22		
Iteración 1	30 días	lun 17/01/22 mié 16/02/22		
Modelado de negocio	3 días	lun 17/01/22 jue 20/01/22		
Planteamiento del problema	0,5 días	lun 17/01/22 lun 17/01/22		Rubén
Brainstorm de ideas	0,5 días	mar 18/01/2: mar 18/01/22	5	Rubén
Investigación de alternativas de desarrollo	1 día	mar 18/01/2: mié 19/01/22	6	Rubén
Reunión con los tutores	1 día	mié 19/01/2: jue 20/01/22	7	Rubén
Especificación de requisitos	21 días	jue 20/01/22 jue 10/02/22		
Definición de objetivos	3 días	jue 20/01/22 dom 23/01/22	8	Rubén
Definición de actores	1 día	dom 23/01/2: lun 24/01/22	8FC+3 días	Rubén
Definición de los requisitos de información	5 días	lun 24/01/22 sáb 29/01/22	8FC+4 días	Rubén
Definición de los requisitos funcionales	8 días	sáb 29/01/22 dom 06/02/22	8FC+9 días	Rubén
Definición de los requisitos no funcionales	4 días	dom 06/02/2: jue 10/02/22	8FC+17 días	Rubén
Análisis del sistema	5 días	jue 10/02/22 mar 15/02/22		
Análisis de mercado	1 día	jue 10/02/22 vie 11/02/22	8FC+21 días	Rubén
Investigación de tecnologías	2 días	vie 11/02/22 dom 13/02/22	16	Rubén
Investigación del hardware	2 días	dom 13/02/2: mar 15/02/22	16FC+2 días	Rubén
Diseño del sistema	1 día	mar 15/02/2: mié 16/02/22		
Diseño del primer prototipo	1 día	mar 15/02/2: mié 16/02/22	18	Rubén
Hito iteración 1	0 días	mié 16/02/2: mié 16/02/22	4;9;15;19	
Hito inicio	0 días	mié 16/02/2: mié 16/02/22	3	
Elaboración	138,5 días	mié 16/02/2: lun 04/07/22	22	
Iteración 2	16,5 días	mié 16/02/2: vie 04/03/22		
Modelado de negocio	2,5 días	mié 16/02/2: vie 18/02/22		

Figura 8 Captura de Microsoft Project del proyecto

5.4.- Especificación de Requisitos de Software

En este apartado se presentará la especificación de requisitos mediante la metodología de Durán y Bernárdez [21].

El objetivo de esta metodología es la definición de las tareas a realizar, los productos a obtener y las técnicas a emplear durante la actividad de elicitación de requisitos de la fase de ingeniería de requisitos del desarrollo de software.

El resultado de dicha especificación usando la metodología desembocará en la siguiente documentación:

- Organizaciones en el proyecto
- Participantes den el proyecto
- Objetivos del sistema
- Catálogo de requisitos del sistema
    - Requisitos de información
    - Requisitos funcionales
        - Diagrama de paquetes
        - Definición de actores
        - Casos de uso del sistema
    - Requisitos no funcionales
- Matriz de rastreabilidad

Toda esta documentación se expondrá de un modo más extenso en el Anexo II.

5.4.1.- Objetivos del sistema

En este apartado se realiza la explicitación de cada uno de los objetivos que debe alcanzar el sistema a la hora de la entrega y la explotación. Los objetivos del sistema son los siguientes:

- Gestión de la partida
    - Gestión del estado de la partida
    - Gestión del estado del jugador en la partida
- Gestión de la movilidad del personaje
- Gestión de armas

- Gestión de usuarios
- Gestión de estadísticas de usuarios
- Gestión de clasificación global
- Gestión de economía, inventario y catálogo.

Para el desarrollo de cada uno de los objetivos citados se han utilizado las tablas proporcionadas también por la metodología Durán y Bernárdez. Un ejemplo de ellas aplicada a un objetivo de nuestro sistema sería la siguiente:

OBJ-01	**Gestión de la partida**
Versión	1.0 (06-07-2022)
Autores	Rubén Sánchez Martín
Fuentes	Equipo de desarrollo
Descripción	Se deberán gestionar la creación y unión a partidas (sesiones) de diferentes tipos y con diferentes configuraciones (modos de juego, jugadores, mapas…) además de todos los eventos que sucedan en la partida y que sean de importancia para todos los jugadores que participan en ella, ya sean visuales (como impactos de disparo, animaciones de recarga y disparo, partículas de disparos, movimientos de los jugadores y objetos) u ocultas (llamadas a eventos de otros jugadores por ejemplo en el impacto, o al estado del juego para aumentar puntuaciones, reducir el tiempo restante…)
Subobjetivos	• [OBJ-02] Gestión del estado de la partida • [OBJ-03] Gestión del estado del jugador en partida
Importancia	Alta
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	Ninguno

Tabla 1 Ejemplo Objetivo del sistema

5.4.2 Requisitos de información

En este apartado desarrollaré los requisitos de información, con los que especificamos los datos que debe almacenar nuestro sistema. Los requisitos de información serán los siguientes:

- Información de los jugadores
- Información de las partidas
- Información de las armas
- Información de los usuarios
- Catálogo
- Ítem
- Clasificación

Un ejemplo de ellos en nuestro proyecto utilizando las tablas proporcionadas por la metodología Durán y Bernárdez será el siguiente:

IRQ-001	**Información de los jugadores**
Versión	1.0 (06-07-2022)
Autores	Rubén Sánchez Martín
Fuentes	Equipo de desarrollo
Dependencias	
Descripción	Información completa y necesaria de cada jugador en la partida. Información sobre su vida, munición, puntuación…
Datos específicos	• Nombre • Nombre Equipo • Color Equipo • Puntuación personal • Vida • Cargador • Munición
Importancia	Alta
Urgencia	Media
Estado	Validado
Estabilidad	Alta
Comentarios	

Tabla 2 Ejemplo de requisito de información

5.4.3.- Requisitos funcionales

En cuanto a los requisitos funcionales, son con los que describiré cómo debe comportarse el sistema al interactuar con los actores. Los requisitos funcionales los expondremos cómo casos de uso en este apartado.

Diagrama de paquetes

Para ofrecer una visión global más sencilla del sistema, he diseñado un diagrama de paquetes en los que repartir los requisitos funcionales. Como base para la construcción, he diseñado un paquete de casos de uso para cada uno de los objetivos explicados anteriormente en el documento (objetivos del sistema):

- Paquete gestión de partidas
  - Paquete gestión de armas: Dentro de partidas, ya que siempre estas estarán vinculadas a partidas
  - Paquetes movilidad del personaje: Dentro del de gestión de partidas porque se refiere a la movilidad dentro de la partida

- Paquete gestión de usuarios
  - Gestión de estadísticas de usuario: Tanto este como el de amigos se refieren a un solo usuario por ello, los incluyo dentro del de gestión de usuarios

  - Gestión de amigos

- Gestión de clasificación global: Separado del de usuarios porque se puede operar con ella fuera del ámbito de los usuarios. Además de que cuando hay modificaciones en las puntuaciones de las clasificaciones de los usuarios, se actúa sobre ella no sobre los usuarios

- Gestión de economía, inventario y tienda: Separado del de gestión de usuarios por la misma razón que el de clasificación global, se puede operar sobre ella con independencia de los usuarios.

Con la explicación realizada el diagrama de paquetes (Figura 9) quedaría como sigue:

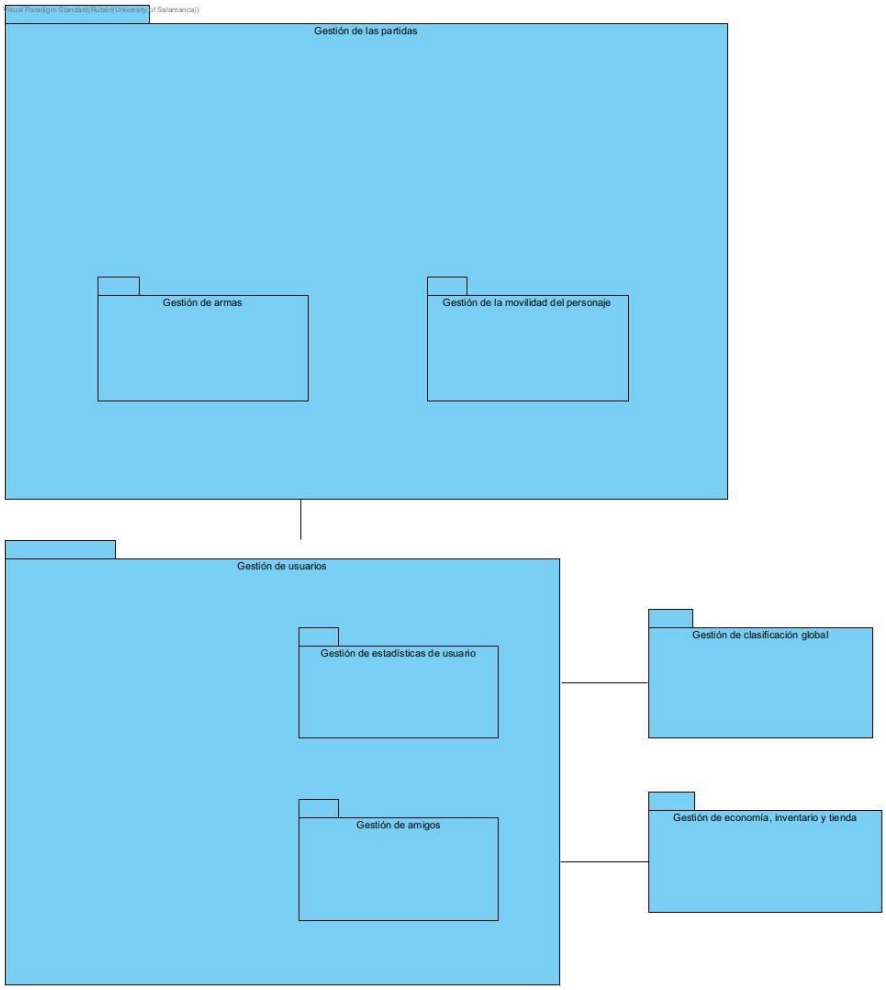


Figura 9 Diagrama de paquetes del sistema

Definición de actores

En este subapartado expongo los actores que participarán en el sistema, mediante las tablas de Durán y Bernárdez. Los actores corresponderían a los siguientes:

- Usuario identificado
- Usuario No identificado
- Sistema

Un ejemplo de tabla siguiendo la metodología sería la siguiente:

ACT-001	Usuario Identificado
Versión	1.0 (06-07-2022)
Autores	Rubén Sánchez Martín
Fuentes	Equipo de desarrollo
Descripción	Es el actor principal, representa al usuario iniciado en el juego y que por lo tanto puede acceder y utilizar toda la funcionalidad del juego
Comentarios	Ninguno

Tabla 3 Ejemplo de actor del sistema

El diagrama de los actores para el proyecto sería el siguiente (Figura 10):

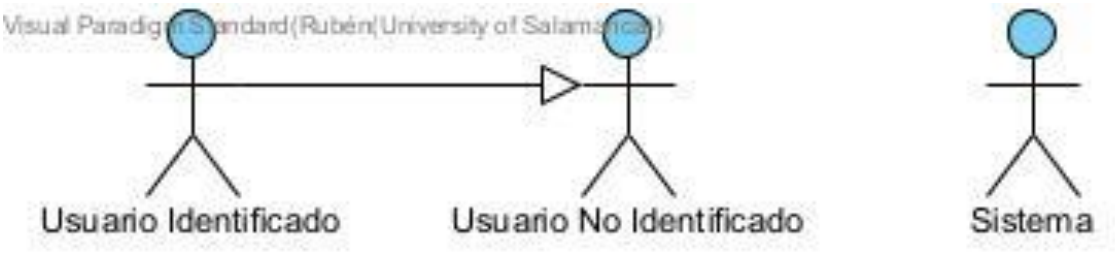


Figura 10 Diagrama de actores del sistema

Casos de uso del sistema

Mediante los casos de uso podemos llevar a cabo la especificación de los requisitos funcionales, modelando la funcionalidad del sistema desde los actores ya definidos. La lista completa de los casos de uso se podrá observar en el Anexo II junto sus especificaciones.

Un ejemplo de las tablas para la especificación de casos de uso siguiendo la metodología sería la siguiente:

UC-019	**Comprar**	
Versión	1.0 (06-07-2022)	
Autores	Rubén Sánchez Martín	
Fuentes	Equipo de desarrollo	
Dependencias	• [OBJ-09] Gestión de economía, inventario y tienda • [IRQ-005] Catálogo • [IRQ-006] Ítem	
Descripción	El usuario comprará la skin seleccionada, pagando con monedas, y será añadida a su inventario permanentemente.	
Precondición	La skin por comprar no la posee el usuario todavía en su inventario y sesión de usuario iniciada; Usuario identificado	
Secuencia normal	**Paso**	**Acción**
	1	El usuario selecciona la opción “COMPRAR” debajo de la skin correspondiente
	2	El sistema recuperará el precio del ítem y las monedas del usuario para comprobar si es posible la compra
	3	El sistema descontará el precio en las monedas del usuario y añadirá la skin al inventario del jugador
Postcondición	Inventario del usuario actualizado con el nuevo objeto y la nueva cuenta de monedas también actualizada	
Excepciones	**Paso**	**Acción**
	2	Si el usuario no posee suficientes monedas para comprar el objeto, se emitirá un mensaje informativo y se finalizará este caso de uso
Importancia	Media	
Urgencia	Media	
Estado	Validado	
Estabilidad	Alta	
Comentarios		

Tabla 4 Ejemplo definición caso de uso

Para ofrecer una mejor visión global sobre los casos de uso y su interacción con los actores he desarrollado el siguiente Diagrama de casos de uso (Figuras 11 y 12), donde aparecen todos ellos en sus diferentes paquetes y sus relaciones.

Como en una sola página no entraría todo el diagrama en este apartado voy a incluir dos imágenes del diagrama, pero para representar correctamente las conexiones voy a representar dos veces unos casos de uso en concreto de manera que, realmente no están duplicados dos veces, pero que los incluyo dos veces para facilitar la visibilidad global de las relaciones.

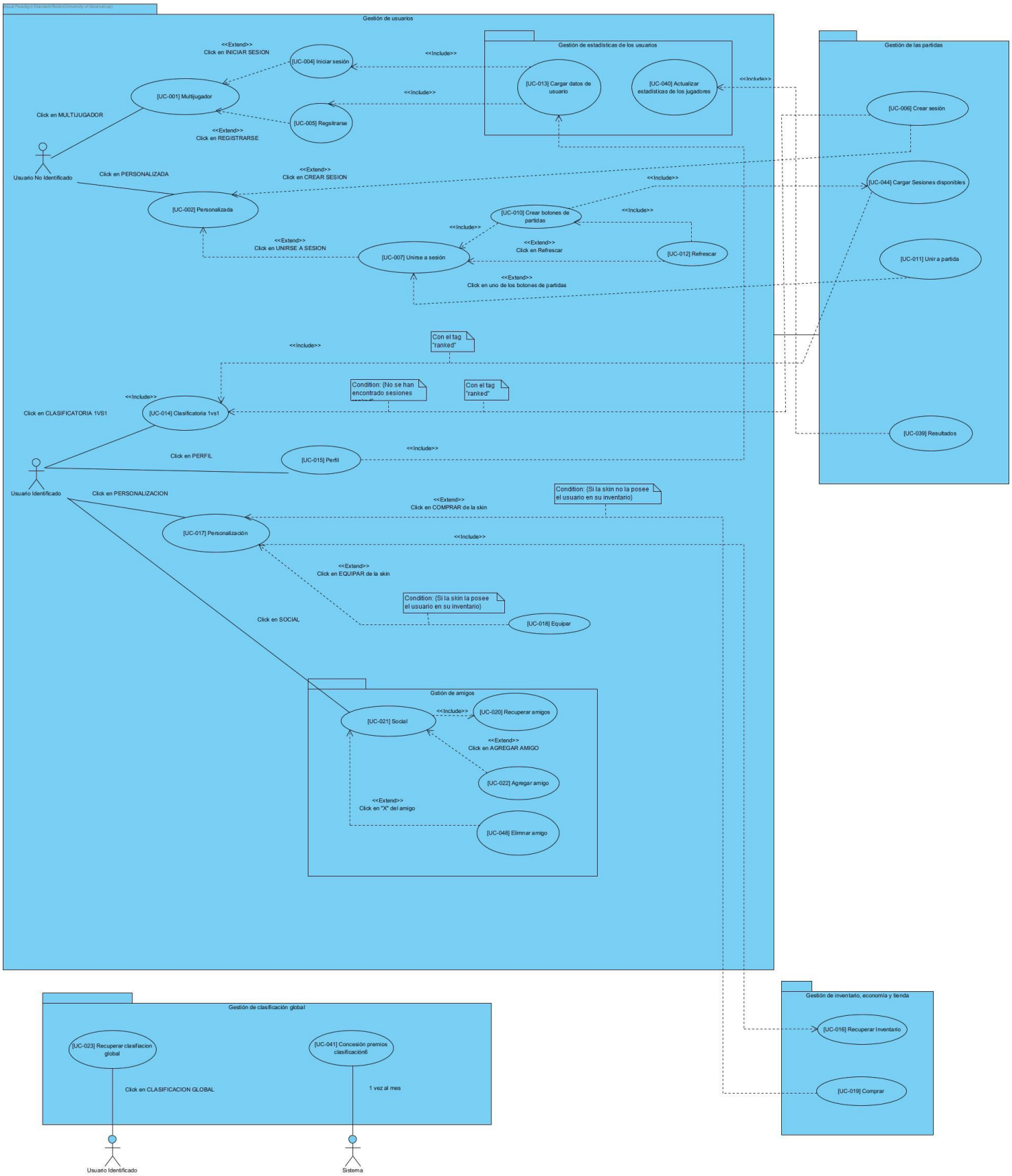


Figura 11 Diagrama de casos de uso Pt 1/2

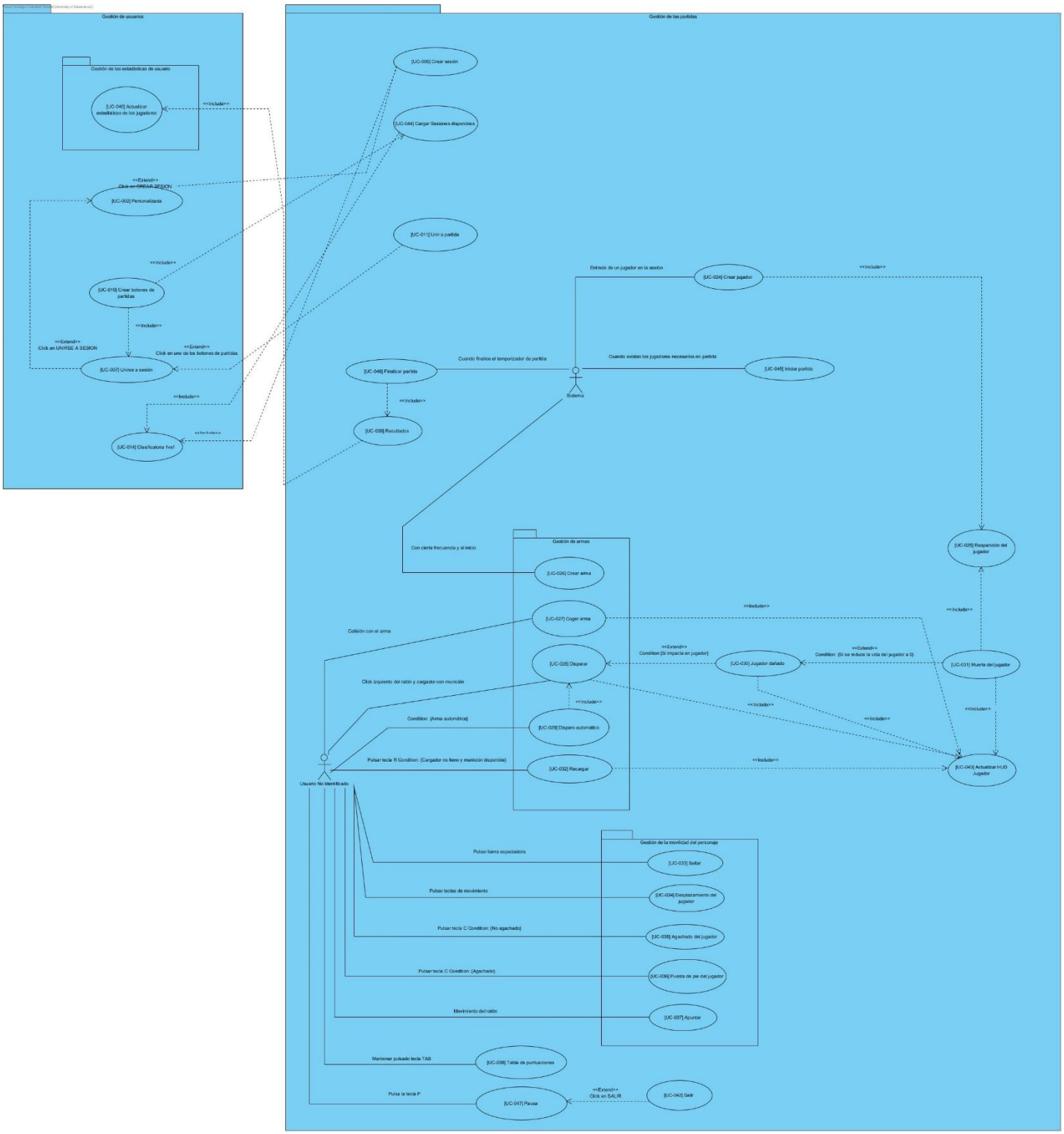


Figura 12 Diagrama de casos de uso Pt 2/2

5.4.4.- Requisitos no funcionales

En este apartado especificaré los requisitos no funcionales que considero que el sistema debe cumplir:

- Usabilidad
- Rendimiento
- Persistencia de datos
- Bajos tiempos de respuesta
- Fiabilidad
- Privacidad
- Portabilidad
- Interfaz de usuario sencilla y clara

Un ejemplo de tabla para los requisitos no funcionales siguiendo la metodología sería el siguiente:

NFR-004	**Bajos tiempos de respuesta**
Versión	1.0 (06-07-2022)
Autores	Rubén Sánchez Martín
Fuentes	Equipo de desarrollo
Dependencias	
Descripción	El sistema intentará ofrecer los mínimos tiempos de respuesta del juego al usuario para ello hará operaciones en segundo plano de manera transparente, para que cuando el usuario necesite información la obtenga al instante. Además, este requisito será muy importante en el desarrollo de las partidas para la excelente experiencia del usuario y el enfrentamiento igualitario entre jugadores.
Importancia	Media
Urgencia	Media
Estado	Validado
Estabilidad	Alta
Comentarios	

Tabla 5 Ejemplo de requisito funcional

5.5.- Análisis del Sistema Software

En este apartado se recogerá el análisis, refinamiento y estructuración referente a los requisitos del sistema. Posibilitando así la correcta y precisa comprensión del sistema para un mejor diseño de se estructura.

5.5.1.- Modelo del dominio

En este apartado mostraré las clases conceptuales significativas en el dominio del problema y las relaciones existentes entre ellas, mediante un diagrama de clases (Figura 13) cuya imagen se puede observar a continuación. Para el desarrollo de este diagrama de clases me he apoyado en los requisitos de información (IRQ) descritos en el Anexo II.

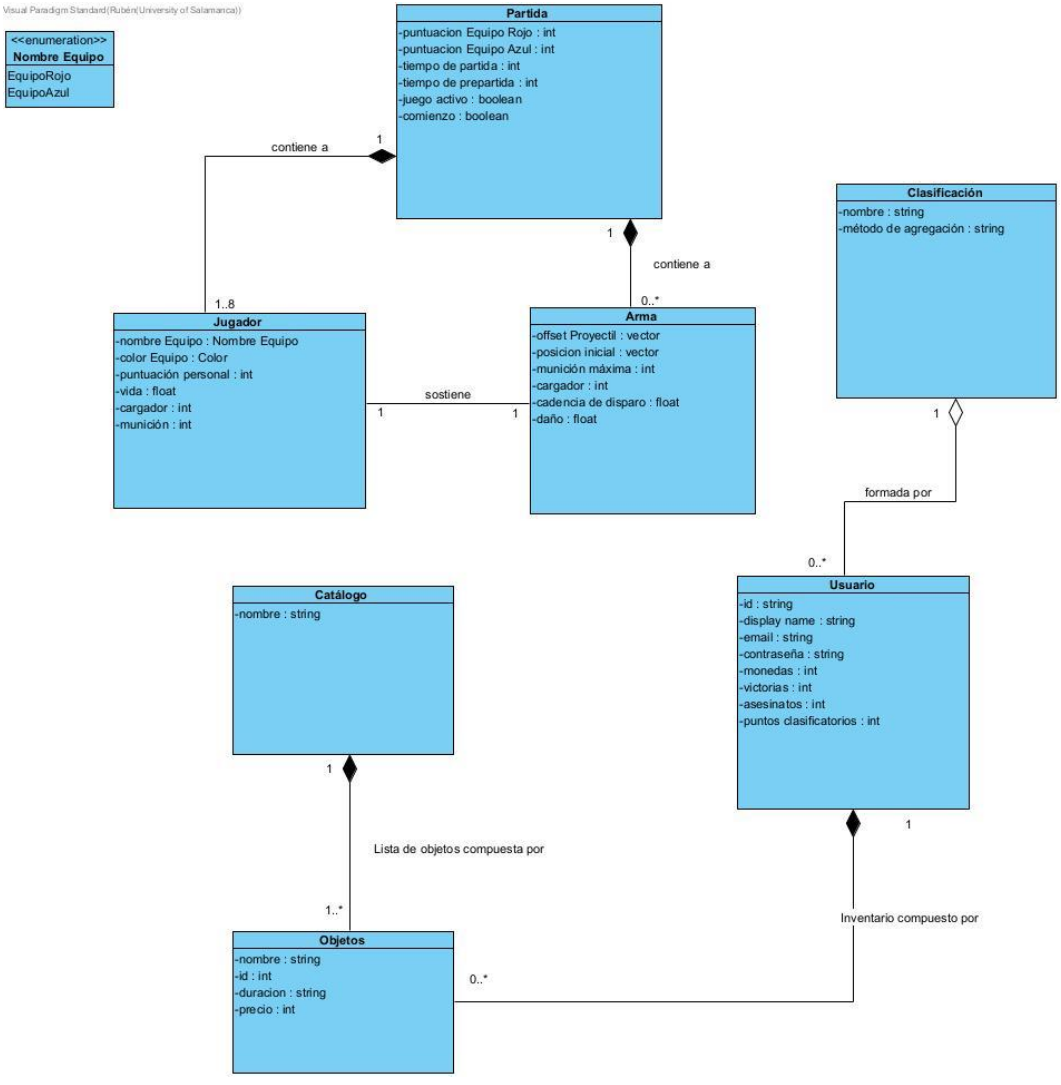


Figura 13 Diagrama de clases

5.5.2- Paquete de análisis y de servicio

En este apartado voy a exponer la descomposición del sistema en paquetes de análisis y de servicio, es decir, descomponer el sistema en partes más manejables. Para ello vamos a descomponer el sistema en los siguientes paquetes de análisis:

- Gestión de las partidas: Realiza la gestión, control, creación y unión a partidas. Manejando todos los eventos e información referentes a ella. La información también incluye información de jugadores global con respecto a posición, pose, estado…
- Gestión de armas: Realiza la gestión de las armas, con toda la funcionalidad asociadas a ellas y toda la información característica y diferente de cada una de ellas.
- Gestión de usuarios: Realiza la gestión de usuarios, su creación o registro e inicios de sesión. Además, almacena toda la información de cada uno de ellos incluido estadísticas y monedas del juego.
- Gestión de la clasificación global: Gestiona la tabla clasificatoria de los usuarios según sus puntos clasificatorios. Incluye también la funcionalidad de la entrega de premios.
- Gestión de economía, inventario y tienda: Gestiona el catálogo, con el mantenimiento y compraventa de los ítems, añadiéndolos a los inventarios de los jugadores. Además, realiza la gestión de la moneda del juego.

El diagrama de paquetes del análisis (Figura 14) correspondiente a los anteriores paquetes sería el siguiente:

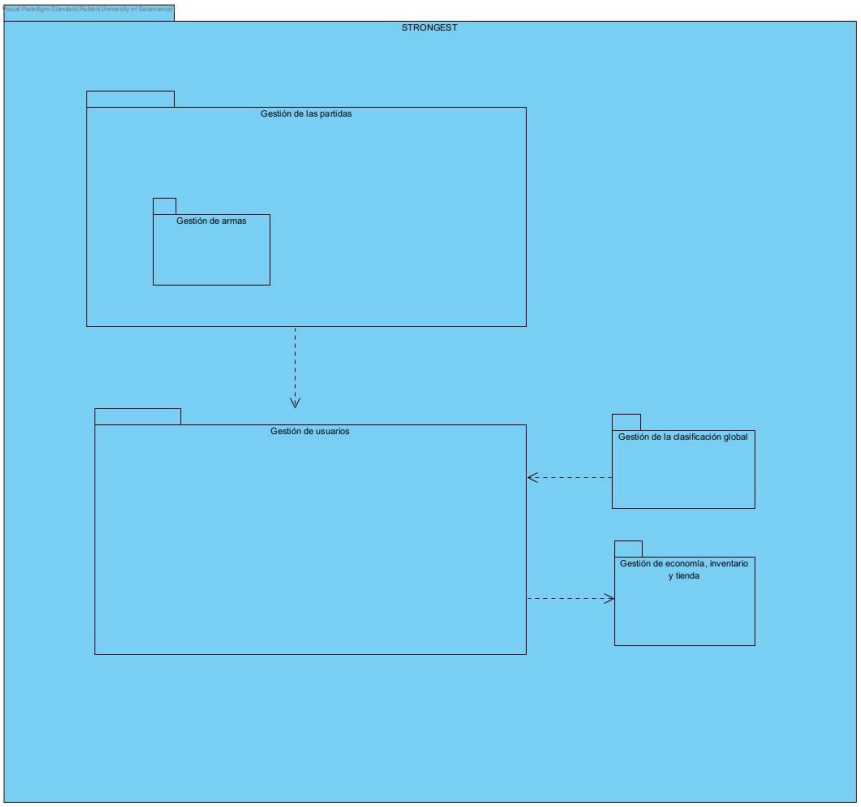


Figura 14 Diagrama de paquetes de análisis

5.5.3.- Clases de análisis y descripción de la arquitectura

En esta sección representaremos las clases de análisis que son abstracciones de las clases y subsistemas del diseño del sistema. Para ello utilizaremos los siguientes tres tipos de clases de análisis:

- Clase entidad: Modelo de información perdurable del sistema.
- Clase control: Agrupa los cálculos y los diferentes algoritmos. Actúa de intermediario entre las clases entidad e interfaz.
- Clase interfaz: Controla la interacción entre el sistema y los actores.

Las diferentes clases de análisis las podemos ver correctamente distribuidas en la siguiente descripción de la arquitectura (Figura 15):

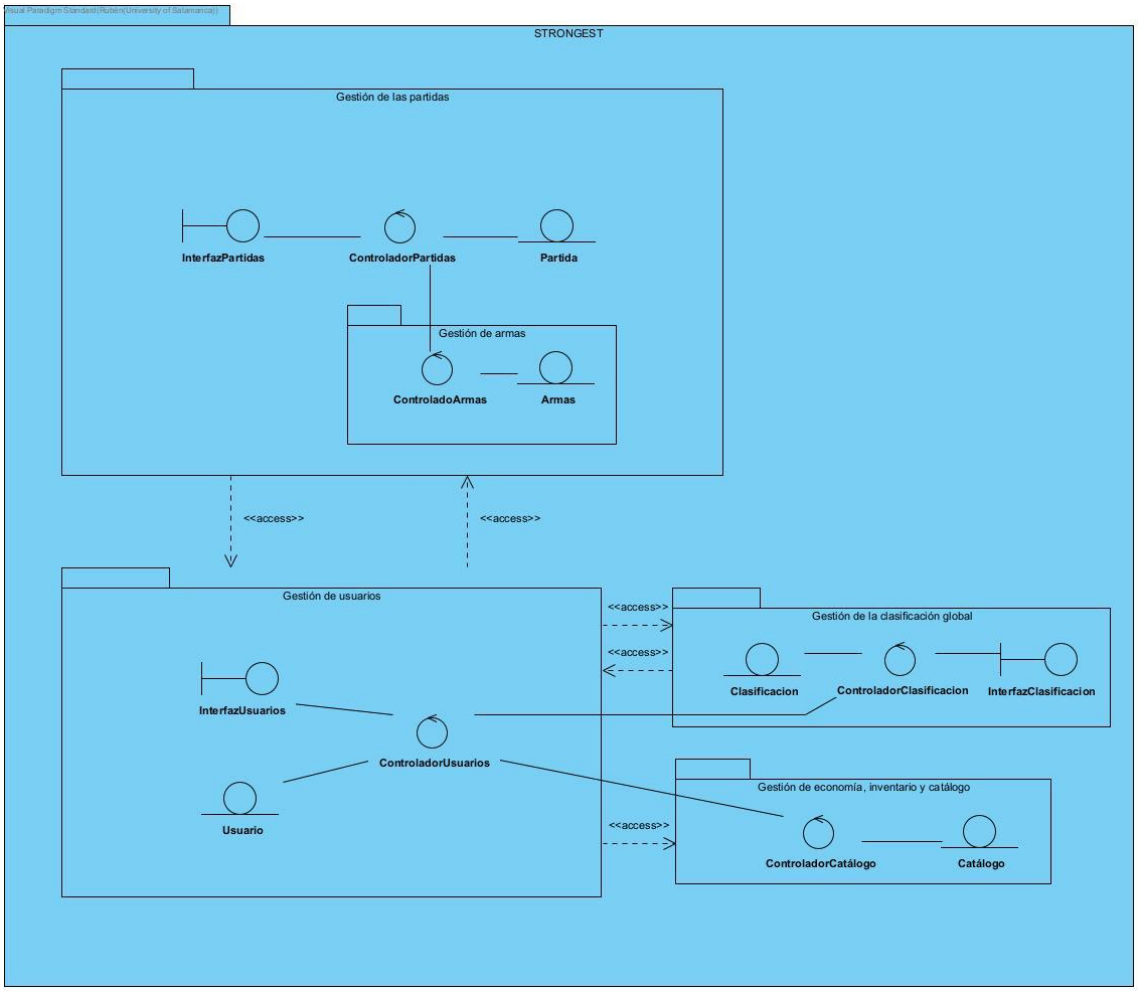


Figura 15 Descripción de la arquitectura

5.5.4.- Realización de casos de uso en el modelo de análisis

La realización de casos de uso en el modelo de análisis la llevaré a cabo mediante los diagramas de secuencia correspondientes a los casos de uso del sistema, mostrando así la interacción e intercambio de mensajes-respuestas entre cada una de las clases de análisis. Existirá un diagrama de secuencia por cada caso de uso, un ejemplo de diagrama de secuencia es el siguiente (Figura 16):

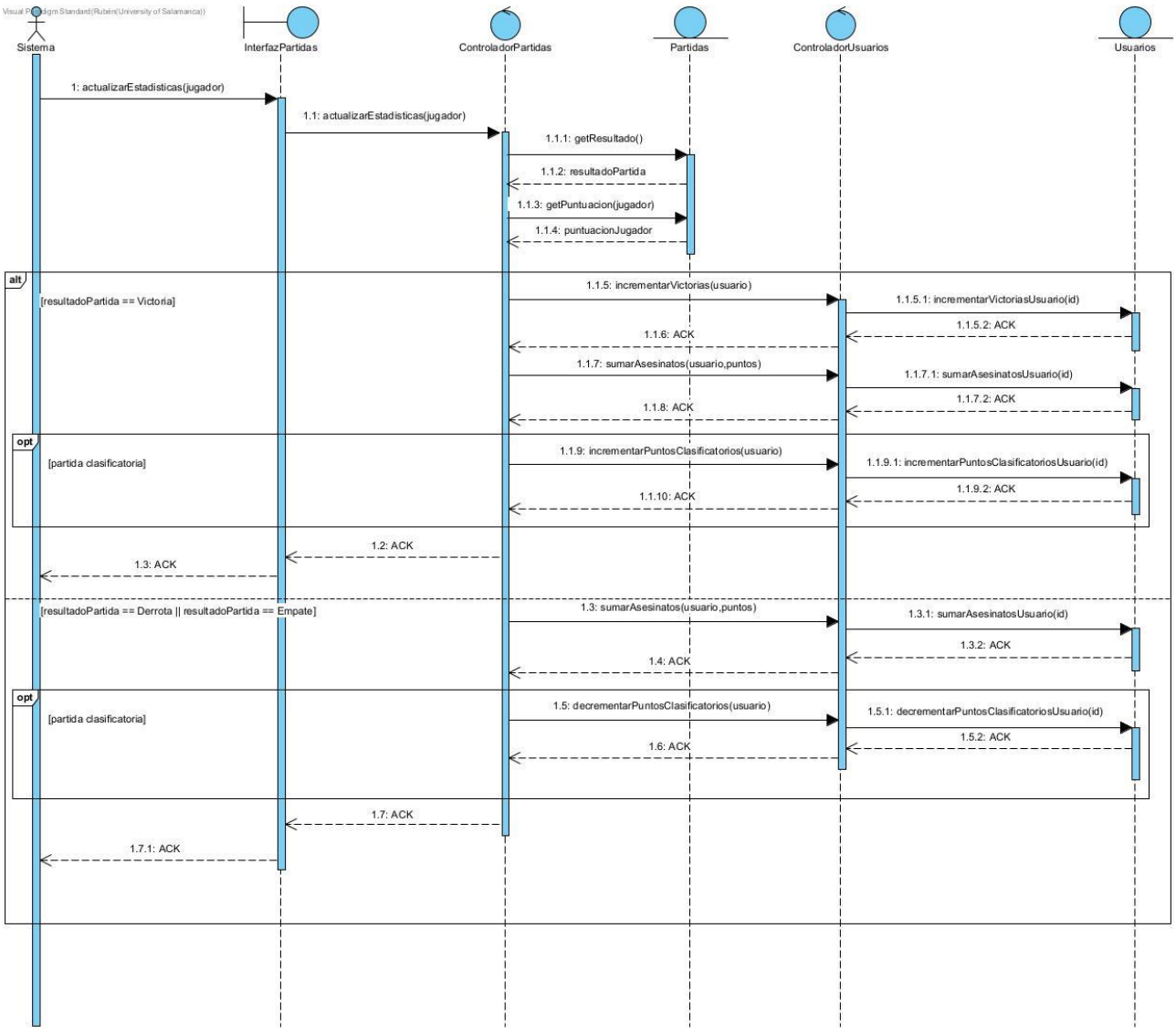


Figura 16 Ejemplo diagrama de secuencia

5.6.- Diseño del sistema Software

El diseño es el proceso de aplicar distintas técnicas y principios con el propósito de definir el sistema con los suficientes detalles para permitir su realización física, que aplicado al software sería el uso de principios científicos, información técnica e imaginación en la definición de un sistema software para que realice las funciones especificadas (especificación de requisitos) con la máxima economía y eficiencia. En cuanto a que ya nos encontramos en el dominio de la solución, es porque ya el objetivo en este anexo será la búsqueda de soluciones para los requisitos del usuario (traducción de requisitos en una representación software).

5.6.1.- Patrones arquitectónicos y de diseño

Patrón arquitectónico de capas / layers

En la gran mayoría del desarrollo del sistema voy a utilizar el patrón de capas cuya descripción voy a realizar en el siguiente subapartado.

El patrón de capas es un patrón de arquitectura de software que consiste en la distribución jerárquica de roles y responsabilidades en capas proporcionando una separación efectiva de las preocupaciones. La base de este patrón es que cada una de las capas sólo dependan de la capa inferior, es decir, que solo puedan invocar funciones de la capa inmediatamente inferior.

En el caso de mi proyecto voy a utilizar tres tipos de capas descritas a continuación:

- Capa de presentación: Presentará la información al usuario e implementará la funcionalidad relacionada con la interfaz del usuario.

- Capa de negocio o dominio: Implementará toda la funcionalidad y procesamientos de nuestro sistema.

- Capa de datos: Implementará el almacenamiento de los datos, así como el acceso e intercambio de información con la base de datos.

Las distintas capas y relaciones (Figura 17) se pueden observar en la imagen siguiente:

Figura 17 Patrón arquitectónico de capas

Patrón Singleton

Para el desarrollo del sistema voy a usar el patrón de diseño Singleton (Figura 18) que consistirá que en cada sistema haya una única instancia de una determinada clase (Singleton) y que podrá ser accedida desde cualquier punto del sistema (acceso global).

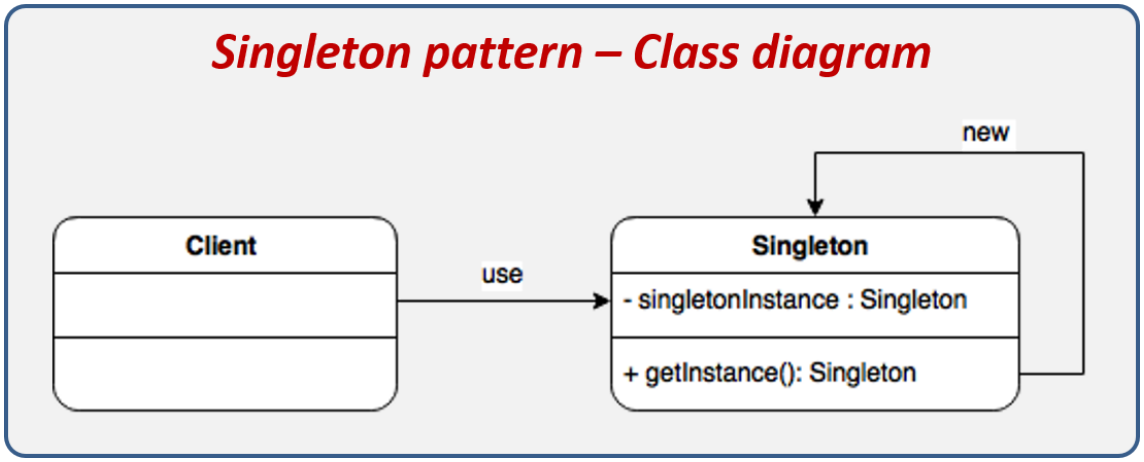


Figura 18 Patrón Singleton

5.6.2.- Subsistemas de diseño

En este apartado mostraré los diferentes subsistemas de diseño que, como ya habíamos explicado, corresponderán a los subsistemas en los que se descompondrá el sistema formando partes más manejables reduciendo la complejidad y facilitando el mantenimiento.

Los subsistemas de diseño serían los siguientes:

- Subsistema usuarios: Subsistema que contendrá todas las clases y funcionalidad relacionadas con los usuarios. He incluido también en este subsistema los paquetes Catálogo y Leaderboard del análisis, ya que estos están muy fuertemente relacionados con los usuarios.
- Subsistema partidas: Subsistema encargado de manejar toda la información y funcionalidad relacionada con las partidas, así como su fuerte tarea de sincronización. La capa de datos de este subsistema se implementará con Playfab aportando su base de datos particular.

Ambos subsistemas seguirán el patrón de capas de forma independiente, implementando los dos las tres capas.

El diagrama de los subsistemas de diseño (Figura 19) será el siguiente:

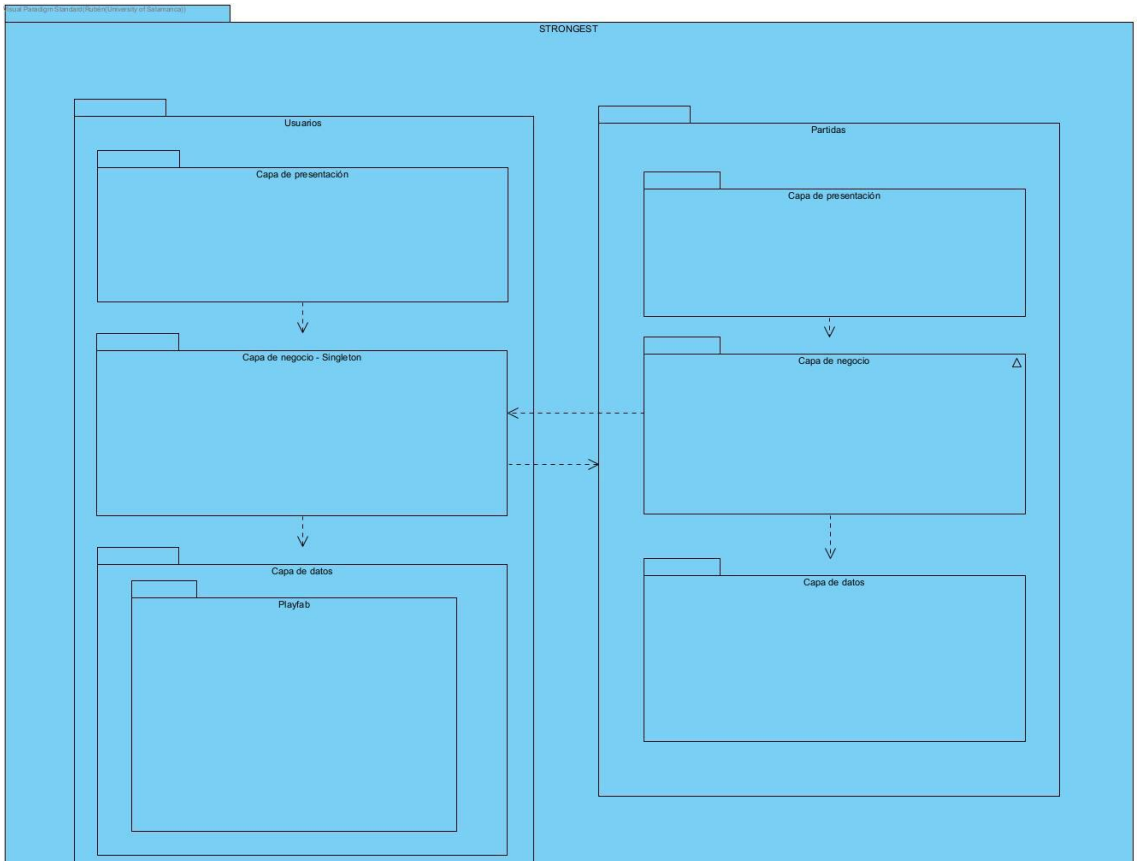


Figura 19 Subsistemas de diseño

5.6.3.- Clases de diseño

En este cuarto apartado llevaré a cabo la definición de las clases de diseño (Figura 20) que forman parte de los subsistemas de diseño. El diagrama general de las clases será el siguiente:

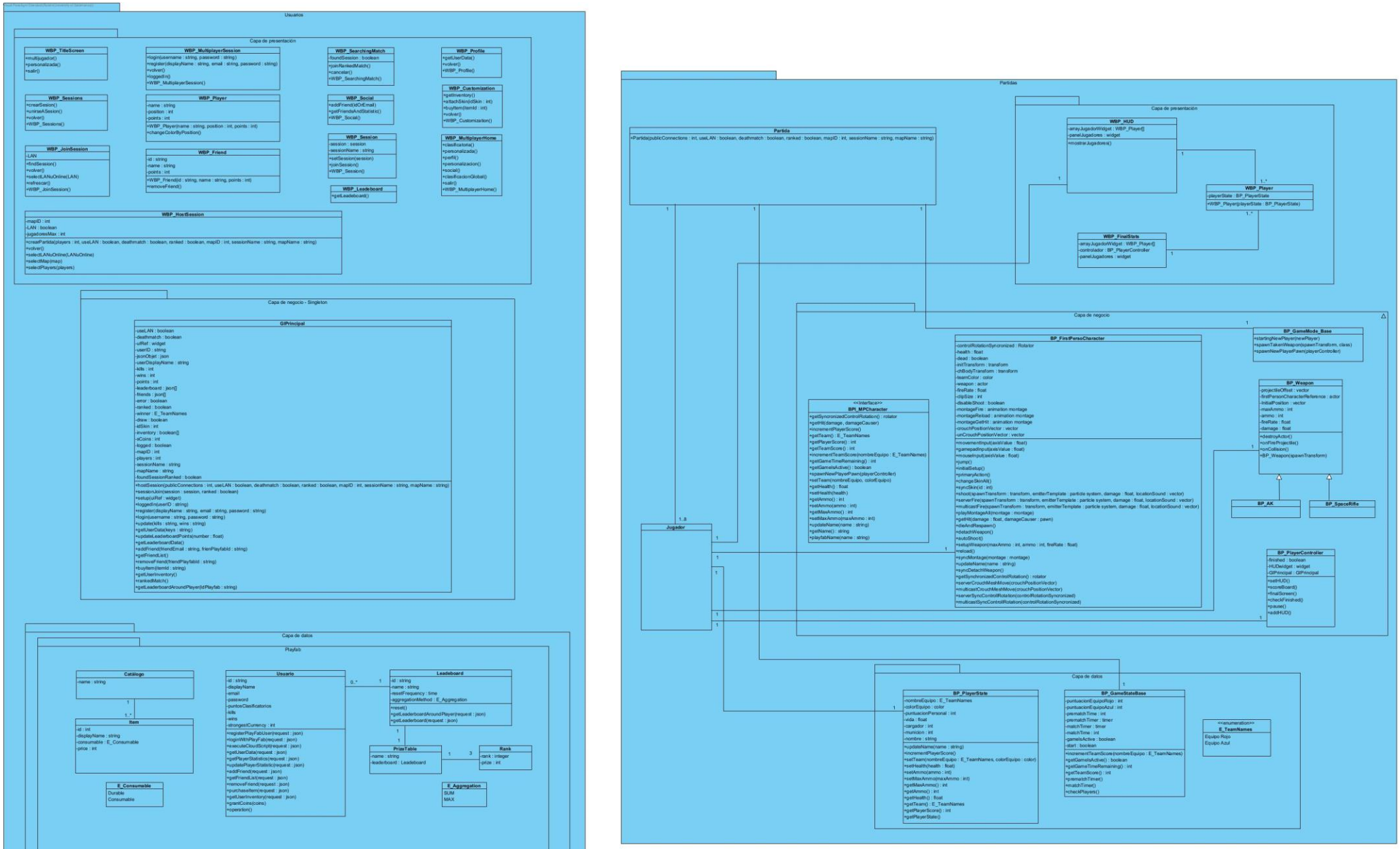


Figura 20 Clases de diseño

5.6.3.1 Subsistema usuarios

La visión general de las clases del subsistema usuarios (Figura 21) será la siguiente:

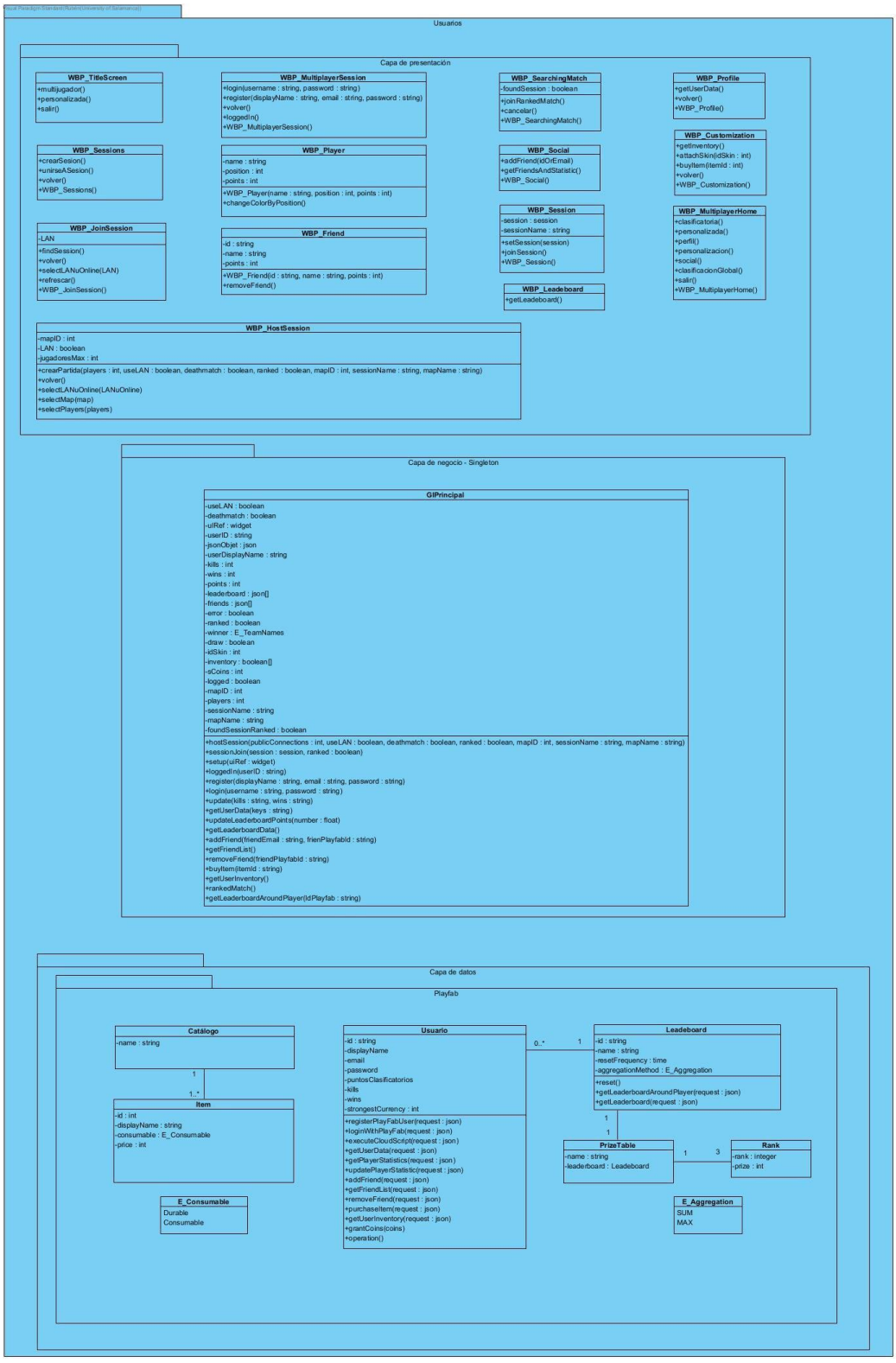


Figura 21 Clases de diseño subsistema Usuarios

5.6.3.1.1 Capa de presentación

Dentro de la capa de presentación tenemos las diferentes clases que representan a las pantallas o widgets (WBP) que se le mostrará al usuario, permitiendo a este interactuar con ellas para lograr utilizar toda la funcionalidad y observar toda la información de manera correcta.

5.6.3.1.2 Capa de negocio - Singleton

Dentro de la capa de negocio se ha implementado una única clase (siguiendo el patrón Singleton) de la cual sólo hay una instancia dentro de cada juego que implementa a su vez toda la funcionalidad y procesamiento de la información correspondiente a los usuarios, y, por lo tanto, también controlará todas las llamadas a la base de datos de Playfab mediante sus funciones.

5.6.3.1.3 Capa de datos

Dentro de la capa de datos se encontrarán todas las clases que representarán información de usuarios o información fuertemente relacionado con ellos (ya sea porque la usan o porque están directamente relacionados con ella). Entre estas clases se encuentran Usuario, por supuesto, Catálogo formado por ítems y Leaderboard formado por usuarios y una Prize Table con diferentes rangos y sus premios correspondientes.

5.6.3.2 Subsistema Partidas

La visión general de las clases del subsistema partidas (Figura 22) será la siguiente:

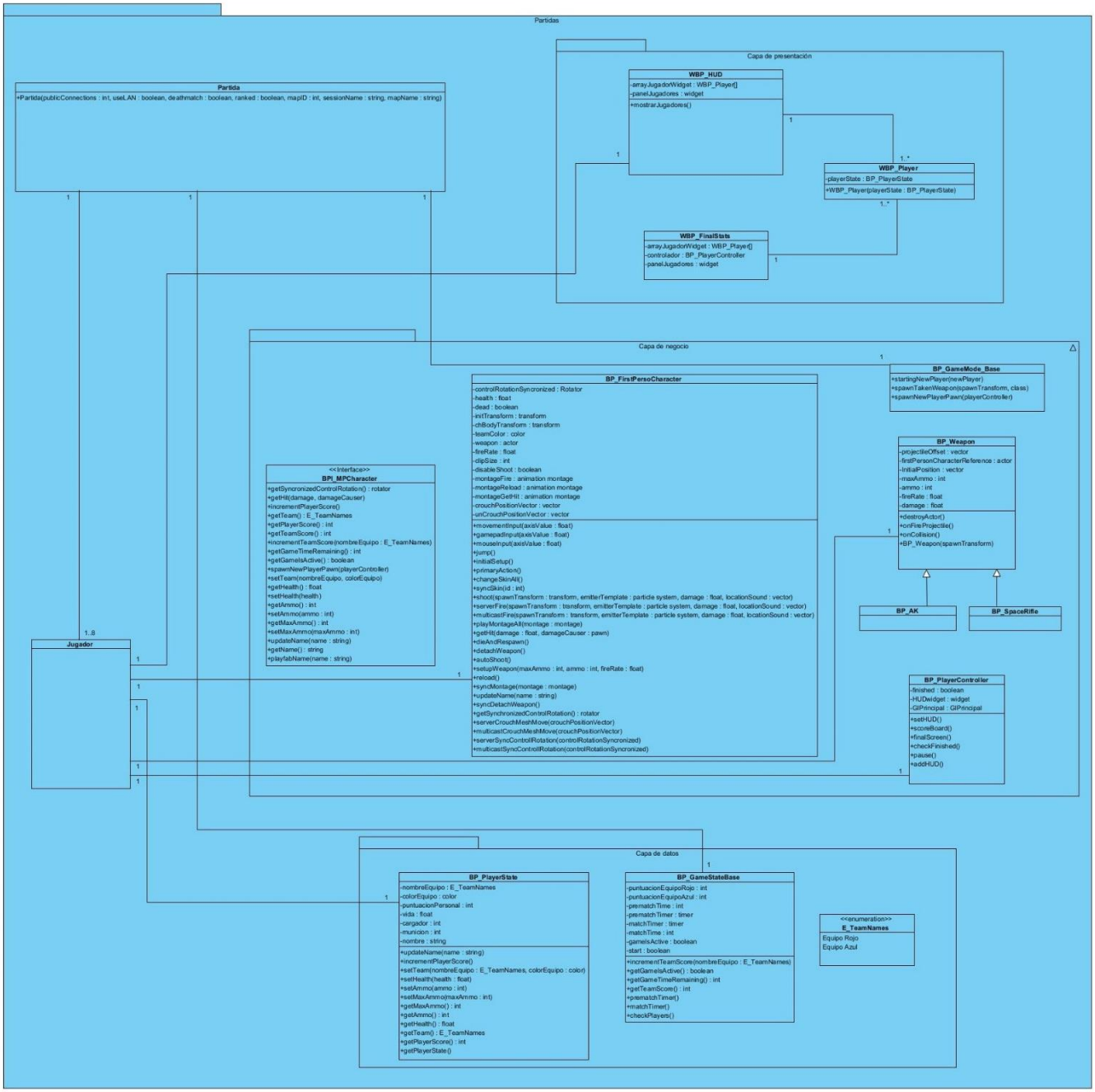


Figura 22 Clases de diseño del subsistema partidas

5.6.3.2.1 Capa de presentación

Dentro de la capa de presentación tenemos los diferentes widgets que el usuario podrá observar como jugador durante el desarrollo de la partida. Así son estos el HUD y la pantalla de resultados final, compuestos los dos por varias clases de información sobre los demás usuarios (jugadores) que están participando en la partida.

5.6.3.2.2 Capa de negocio

Dentro de la capa de negocio se han implementado varias clases que implementan a su vez toda la funcionalidad y procesamiento de la información correspondiente a la partida. Existen las siguientes clases:

- BP_FirstPersonCharacter: Englobará toda la funcionalidad del personaje del jugador en la partida, así como el procesamiento de su información (almacenada en el BP_PlayerState de la capa de datos).
- BP_GameModeBase: Implementará las características y referenciará a las clases vinculadas a las partidas. Además de implementar funcionalidad adicional general de la partida.
- BP_PlayerController: Implementará la funcionalidad de los widgets y, por lo tanto, la información que se le mostrará al usuario en partida. Además, implementará cierta funcionalidad adicional vinculada al control del jugador.
- BP_Weapon: Representa a todas las armas que se encuentran en la partida, así como la información relacionada con ellas.

5.6.3.2.3 Capa de datos

Dentro de la capa de datos se encontrarán todas las clases que almacenarán toda la información de importancia para la partida, tanto la de la partida general (BP_GameStateBase) como la de cada uno de los jugadores (BP_PlayerState).

5.6.4.- Realización de casos de uso del modelo del diseño

En este apartado llevaré a cabo la realización de casos de uso en el modelo del diseño que describirá cómo se realiza un caso de uso en el modelo del diseño mostrando los distintos objetos que intervienen (la realización completa de todos los casos de uso se encuentra en el Anexo IV).

Ejemplo de realización de caso de uso del modelo del diseño (Figura 23):

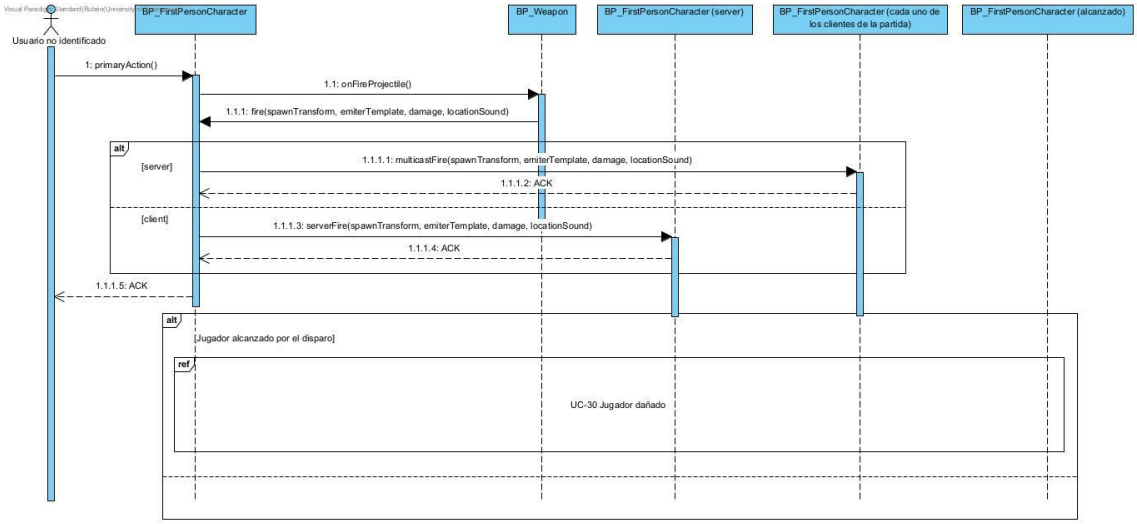


Figura 23 Ejemplo de realización de caso de uso del modelo del diseño

5.6.5.- Modelo de despliegue

En este apartado expongo el modelo de despliegue (Figura 24) que muestra la distribución de nodos y componentes que conforman en sistema, así como sus interacciones.

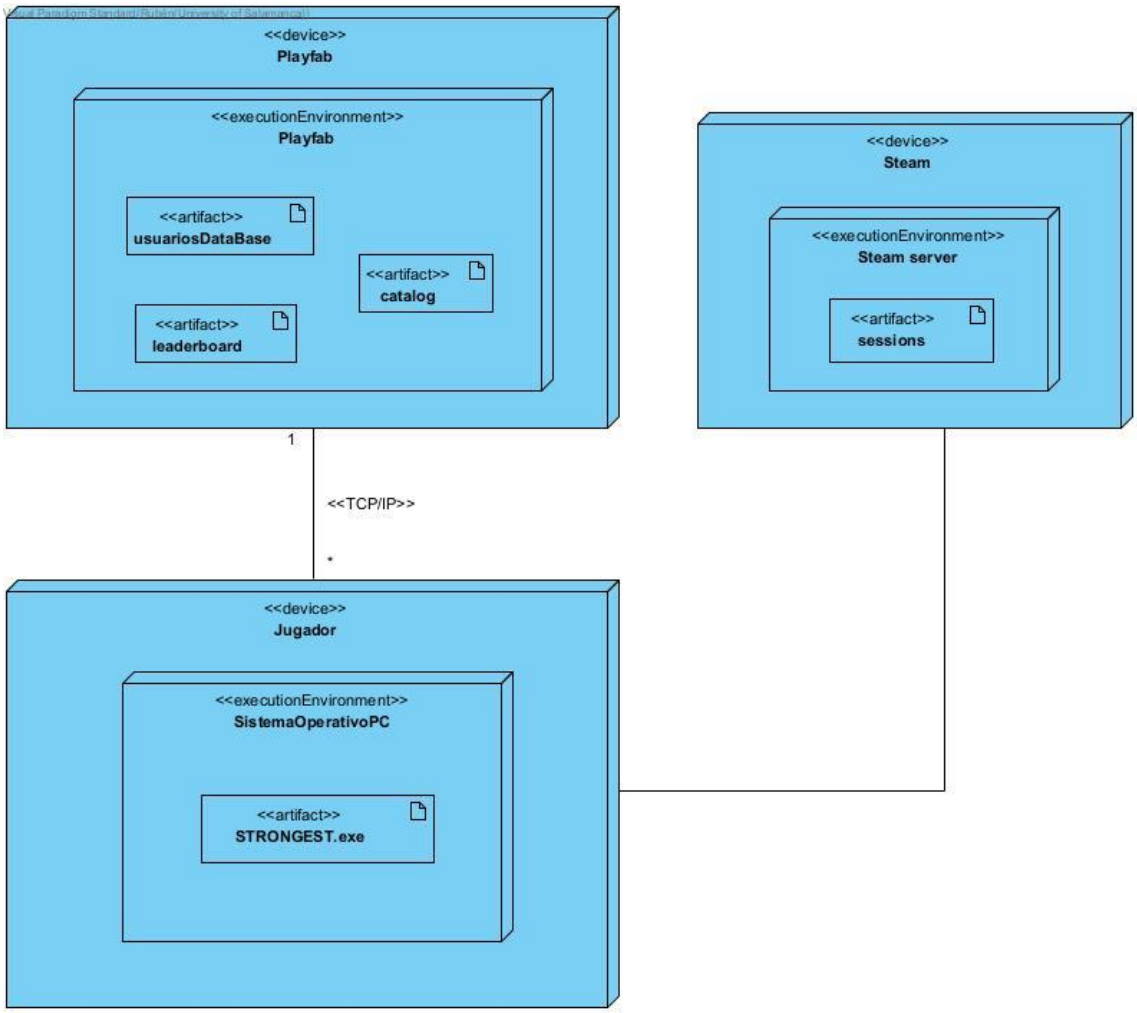


Figura 24 Modelo de despliegue

El sistema posee tres nodos distintos cada uno de ellos a la vez formado por artefactos:

- Nodo Jugador: Es un dispositivo cualquiera en el cual se encuentra una instancia ejecutable del juego.

- Nodo Playfab: Implementa el servidor con la base de datos de usuarios, la colección de la clasificación y del catálogo. A este se le conectarán los diferentes jugadores para recuperar y actualizar la información almacenada por él.

- Nodos Steam: Proporcionará el soporte para el multijugador del juego proporcionando sesiones explotables por los jugadores para desarrollar las partidas online.

5.7.- Implementación

La construcción del juego en sí mediante los blueprints, y la investigación que ha llevado consigo (sobre todo a la hora de cómo conseguir la replicación, sincronización y paso de información), ha sido sin duda la que ha conllevado más carga de trabajo en el proyecto.

El sistema de blueprints, el cual ya he explicado en el apartado 4.10.1 de esta memoria, ha sido ampliamente usado en este proyecto. Se han utilizado blueprints de números tipos que engloban mucha funcionalidad distinta, se han usado blueprints tanto proporcionados ya por Unreal Engine como construidos por otros desarrolladores (Advanced Sessions) con la costosa implementación que han tenido como consecuencia (reconstruir el código del proyecto desde los recursos de Unreal Engine). Algunos de los tipos de blueprints y clases blueprints usados en el proyecto son los siguientes:

- Blueprints de operaciones matemáticas: Para la funcionalidad de procesamiento de la información de la base de datos.

- Blueprints de llamadas al servidor y multicast (Figura 25): Para poder realizar la sincronización y comunicación entre las distintas instancias del juego.

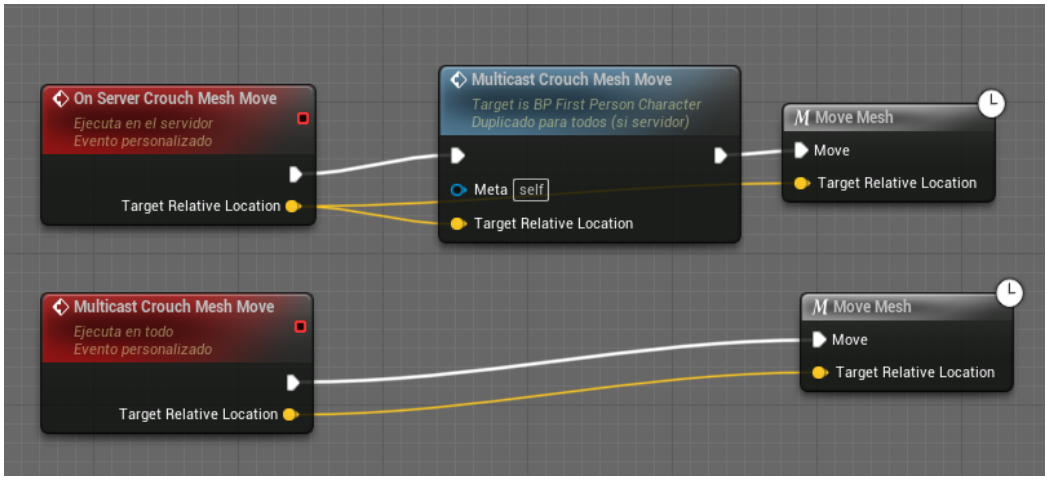


Figura 25 Ejemplo de uso de blueprints de llamadas al servidor y multicast para la sincronización del agachado

De esta manera cuando se produce cierto evento se comprueba que la instancia de juego en la que se produce es el servidor, es decir si tiene autoridad (Figura 26), si es este el caso se llamará directamente al blueprint multicast para que difunda el evento a todas las instancias del juego. Si es el caso en el que no tiene autoridad, se llamará al servidor para que este haga el multicast y se realice la sincronización.

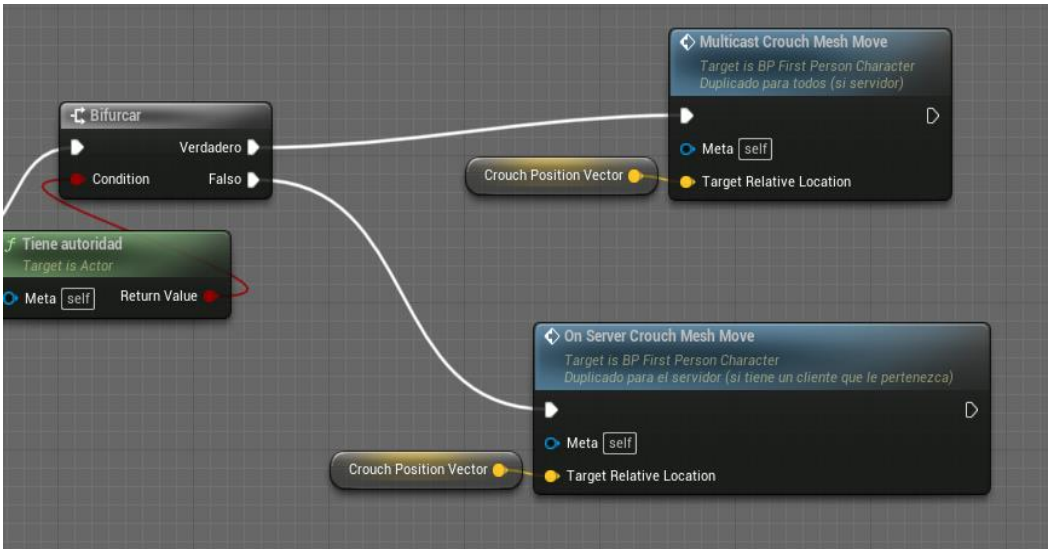


Figura 26 Llamadas distintas si tiene autoridad o no, multicast o al servidor

- Blueprints de eventos (Figura 27): Controlarán los distintos eventos que ocurren en las partidas y llevarán a cabo la funcionalidad si es necesaria.

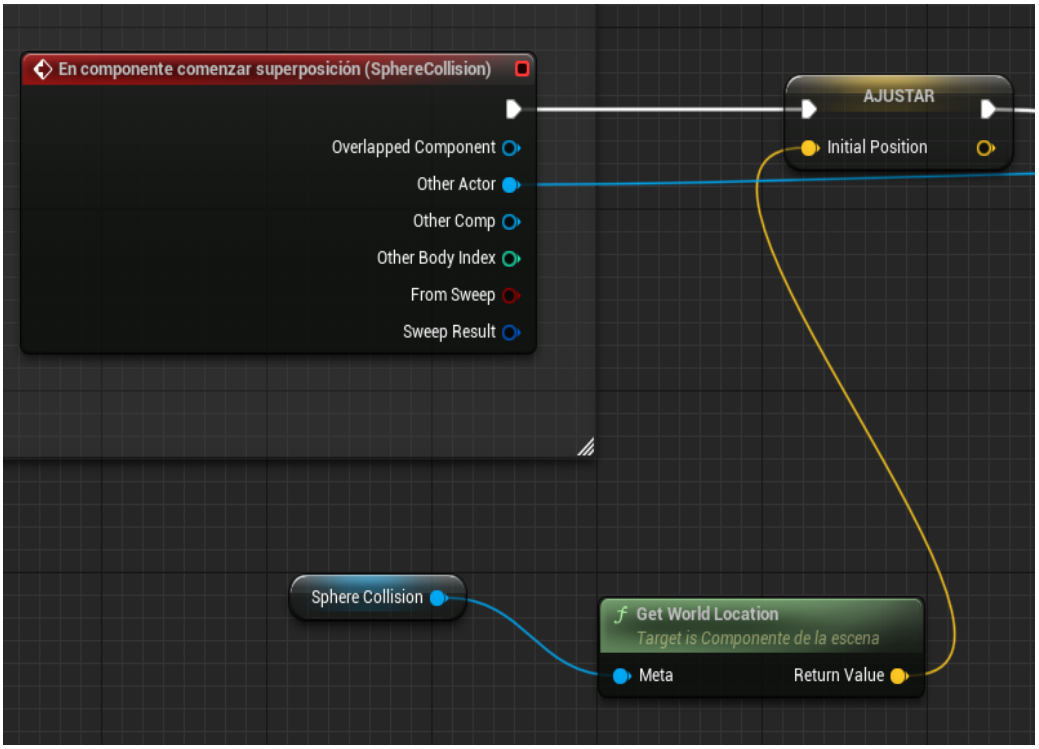


Figura 27 Ejemplo blueprint controlador de evento, comenzar superposición

- Blueprints que vinculan eventos a otros de forma dinámica (Figura 28) lo que permite en el proyecto el equipado de las distintas armas distintas:

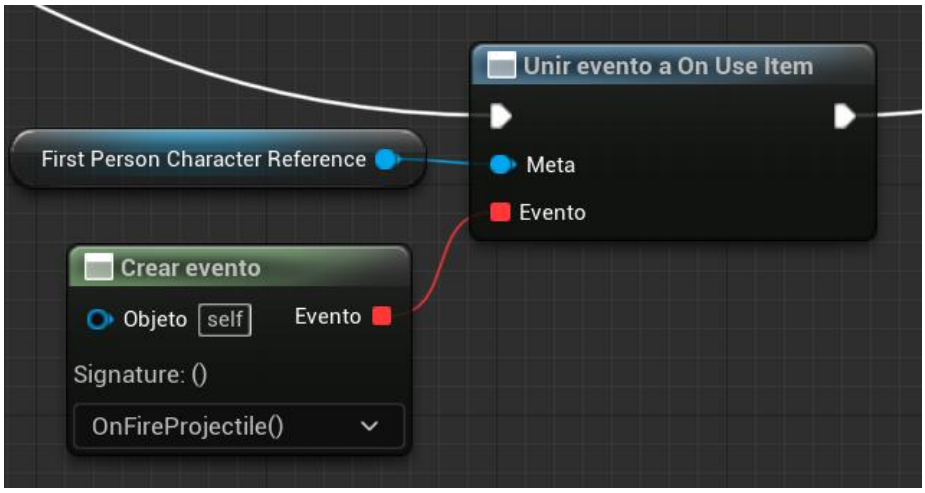


Figura 28 Blueprint que vincula el evento OnFireProjectile a On Use Item

- Blueprints de eventos personalizado (Figura 29): Implementarán casi toda la funcionalidad.

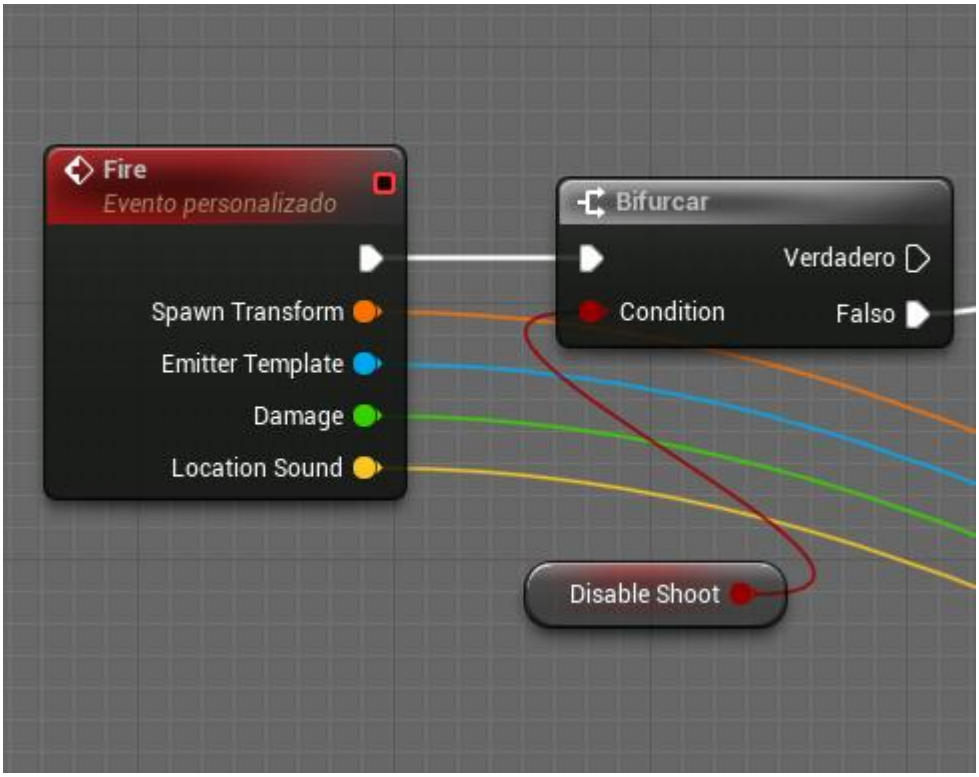


Figura 29 Ejemplo blueprint de evento personalizado con varios parámetros

- Blueprints que llaman a otros blueprints de otras clases de forma dinámica para obtener o modificar información remota (Figura 30). Esta funcionalidad se ha conseguido llevar a cabo mediante la creación de una interfaz blueprint que englobará diversas funciones.

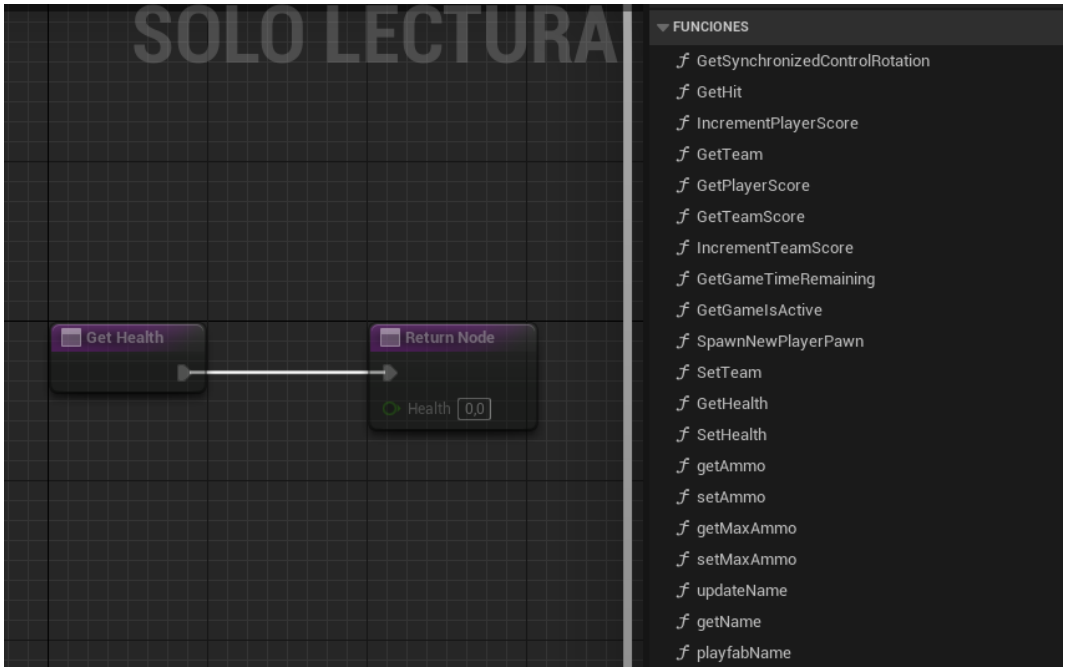


Figura 30 Funciones de la interfaz

Estas funciones de la interfaz mostradas en la imagen serán implementadas (Figuras 31 y 32) en una clase blueprint:

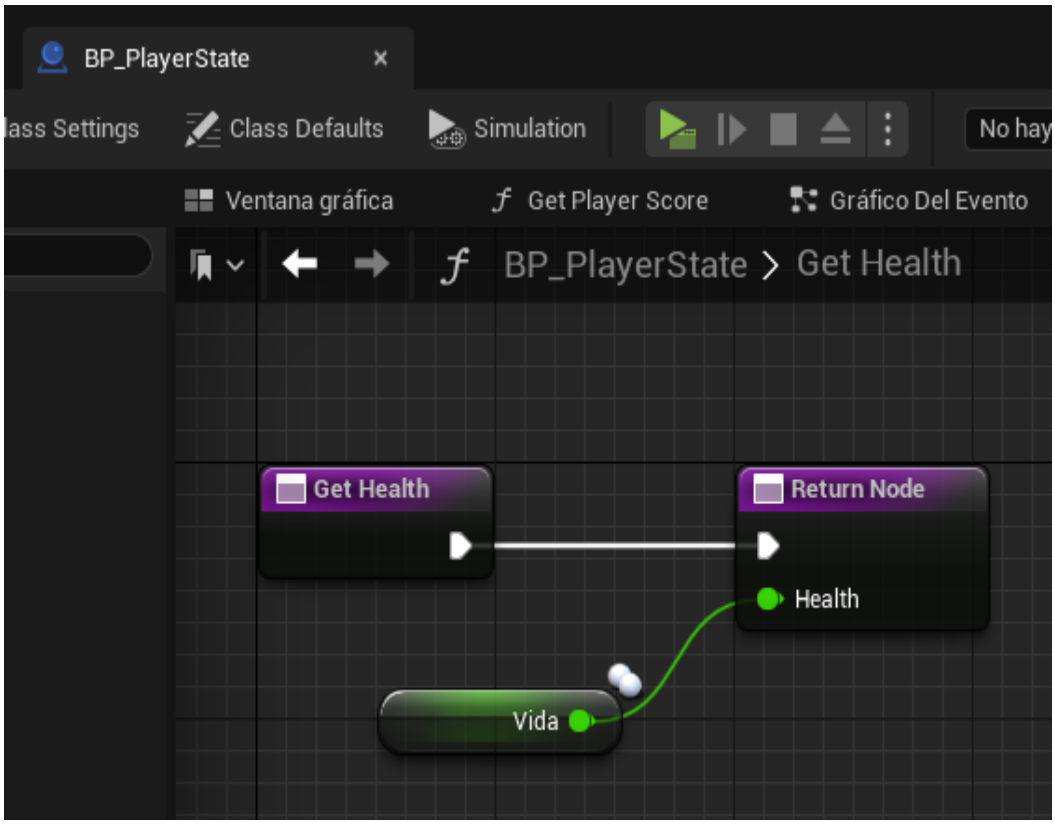


Figura 31 Ejemplo 1 implementación función de la interfaz

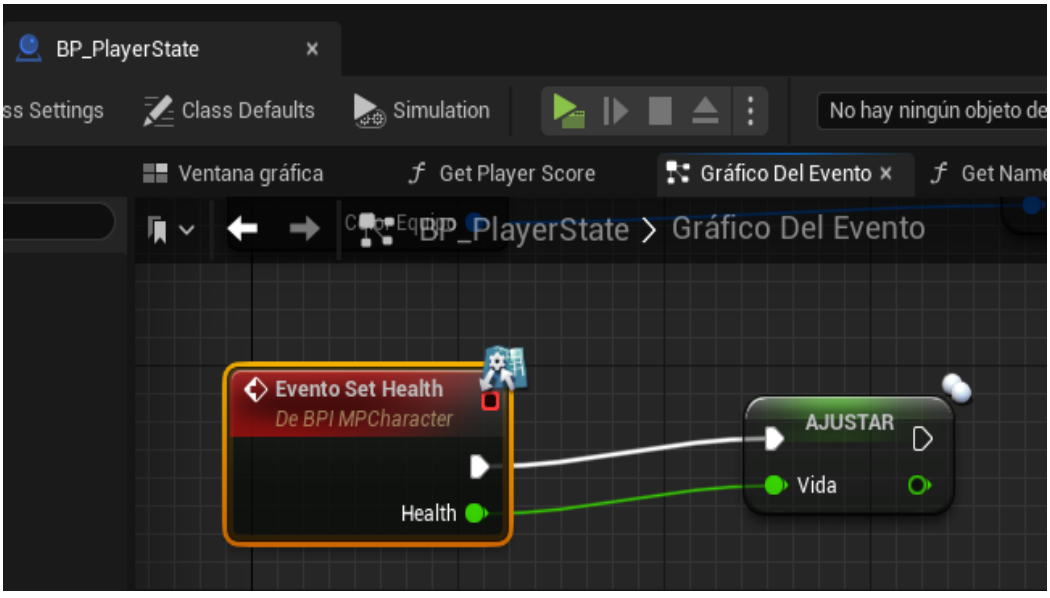


Figura 32 Ejemplo 2 implementación función de la interfaz

Para que después puedan ser llamadas desde cualquier parte del juego mediante una llamada a la función de la interfaz(Figura 33).

Figura 33 Ejemplo de diferentes llamadas a las funciones de la interfaz

- Blueprints vinculados al modo de juego: Cada nivel estará vinculado a un Game Mode (Figura 34) el cual determina las clases que se van a utilizar en el nivel además de implementar cierta función propia general del nivel:

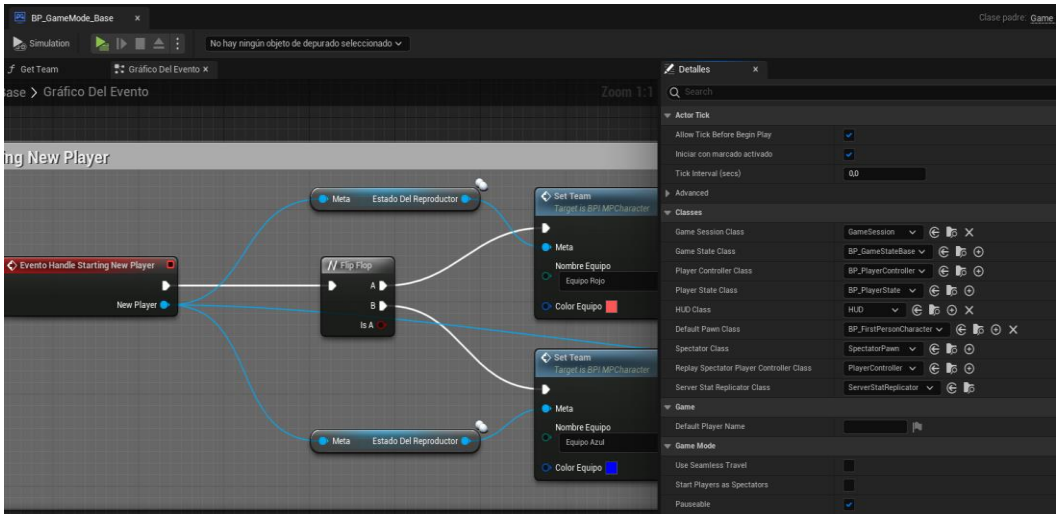


Figura 34 Captura del Game Mode de una partida donde se puede observar parte de la funcionalidad implementada en él, así como las clases vinculadas a él (a la derecha de la captura)

Las clases vinculadas a él personalizadas por mí serán las siguientes:

- BP_GameStateBase (Figura 35): Almacenará y controlará toda la información general de las partidas, así como controlará el tiempo de esta por medio de temporizadores.

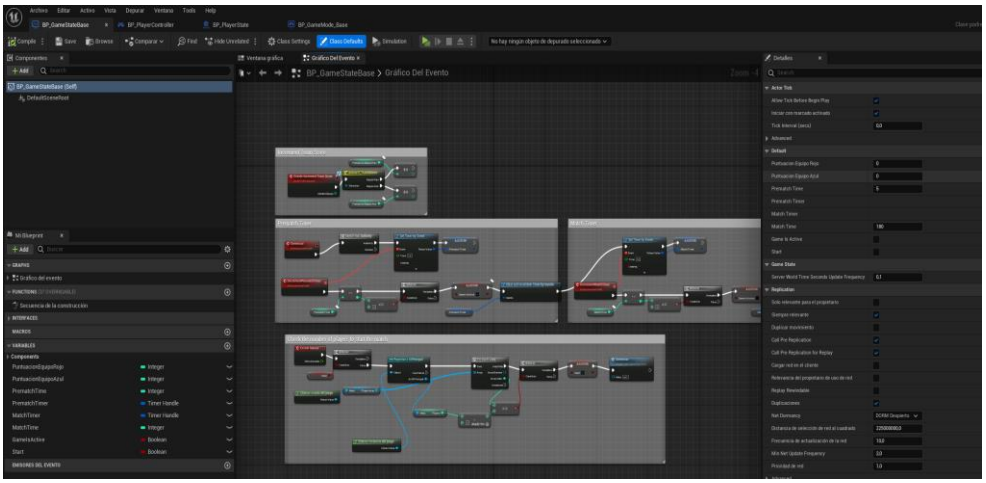


Figura 35 Captura del Game State

- BP_PlayerState (Figura 36): Almacenará y controlará toda la información propia de un jugador de la partida, a cada jugador le corresponderá una la cual se encuentra replicada y sincronizada (manualmente) en todas las instancias del juego vinculadas a la partida.

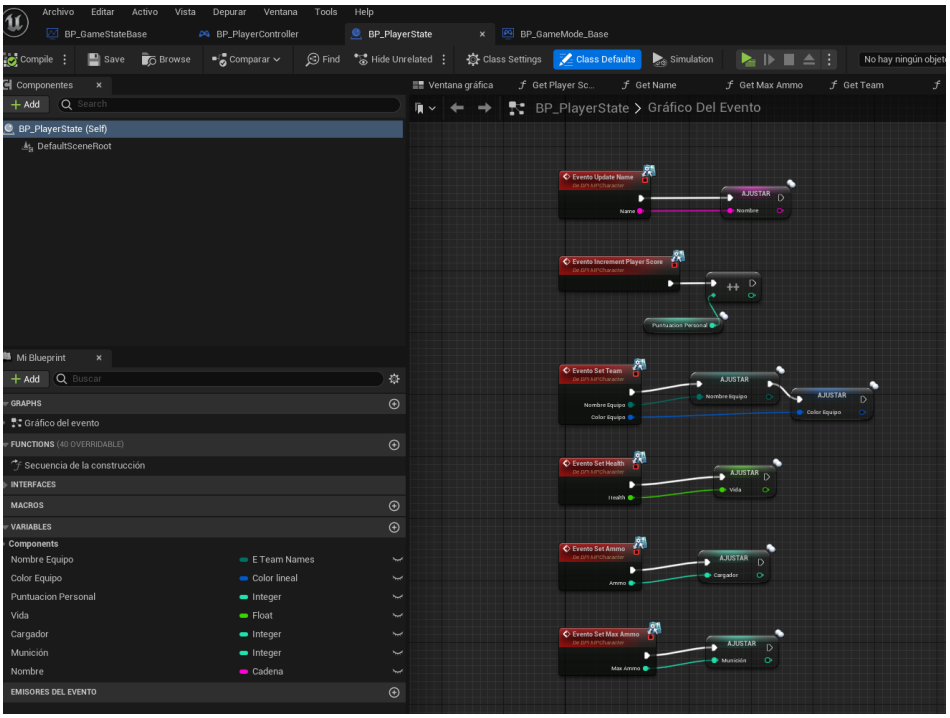


Figura 36 Captura del player state

- BP_FirstPersonCharacter (Figura 37): Es la clase blueprint que controlará que peón o personaje controlará el jugador. Además,

esta implementará toda la funcionalidad de este y llamadas las clases state para modificar la información de estas.

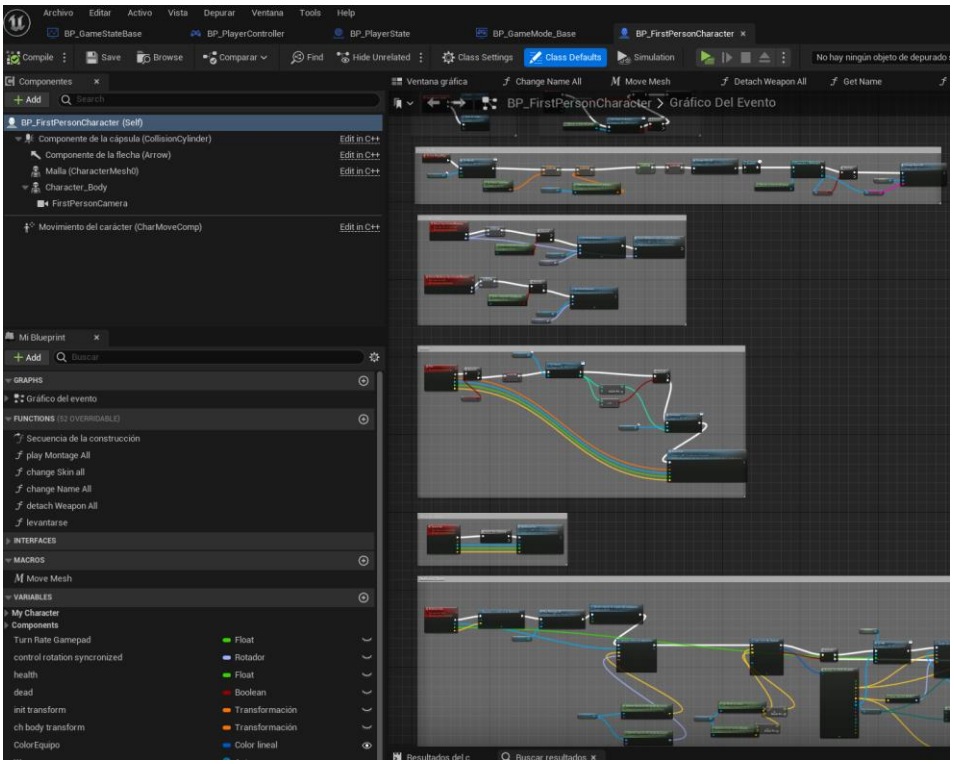


Figura 37 Captura del BP_FirstPersonCharacter

- BP_PlayerController (Figura 38): Controlará parte de la funcionalidad del jugador además de toda la información que se le muestra. Existe una por jugador.

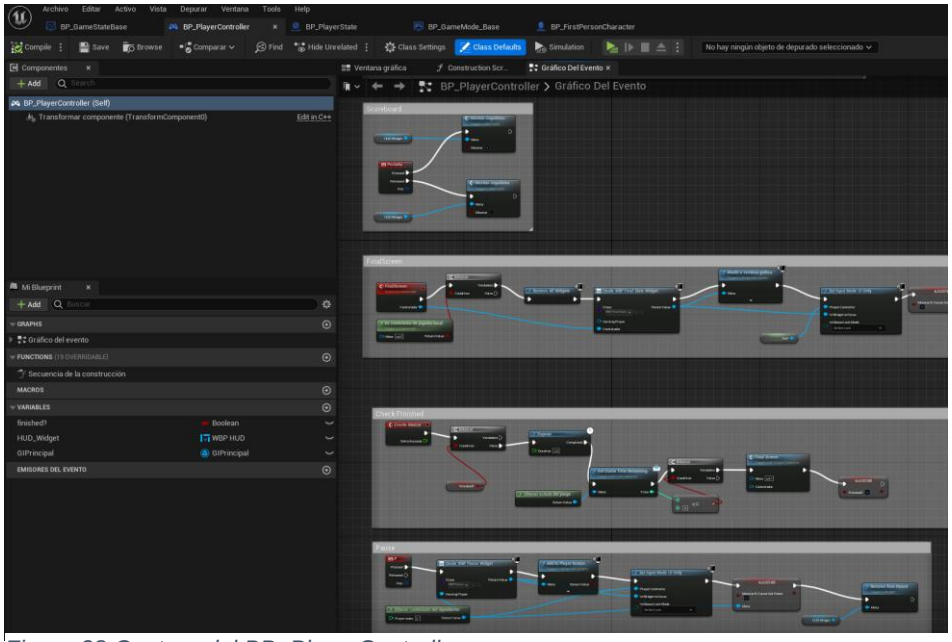


Figura 38 Captura del BP_PlayerController

- Blueprints de construcción de JSON y llamadas a la API de Playfab (Figura 39): Dado que los artefactos implementados mediante Playfab poseen cierta importancia en nuestro proyecto se han utilizado diferentes blueprints para la construcción de objetos JSON para su posterior envío en forma de request a la API de Playfab.

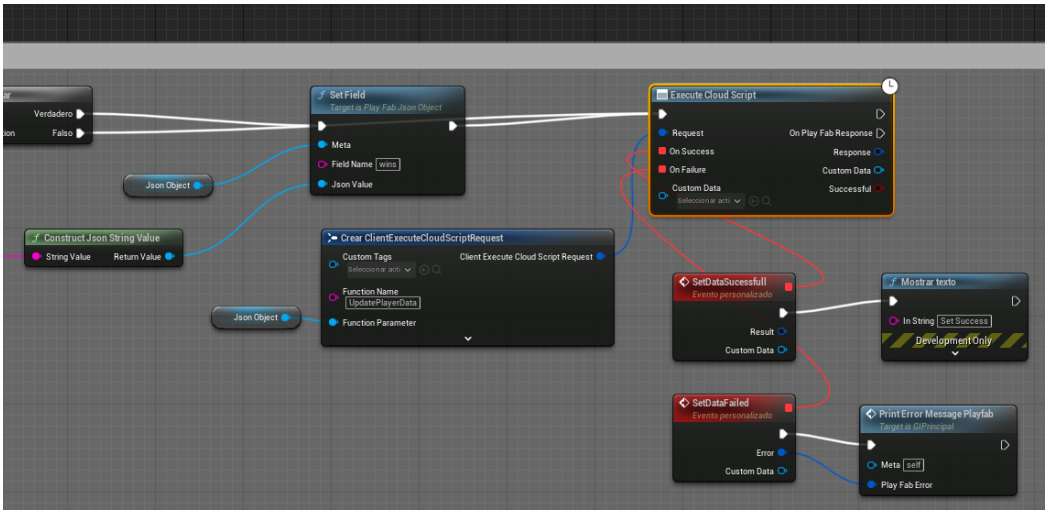


Figura 39 Ejemplo de construcción del JSON con su request y su correspondiente llamada a la API

- Otros blueprints como de animaciones (nuevas para añadir a nuestro personaje), montajes (para que el personaje pueda ejecutar las distintas animaciones), partículas (efectos visuales en el juego), materiales (controlar qué sistema niagara emitir según la superficie de impacto de los disparo o para cambiar el color del personaje para hacerlo corresponder con el equipo), sistemas niagara (efectos visuales en el juego), enumeraciones, blendspace (controlar las animaciones del jugador en cada momento), esqueletos (añadir huesos para colocar el arma), ik retargeter e ik rig (para el retargeteo de las nuevas skins de los personajes)…

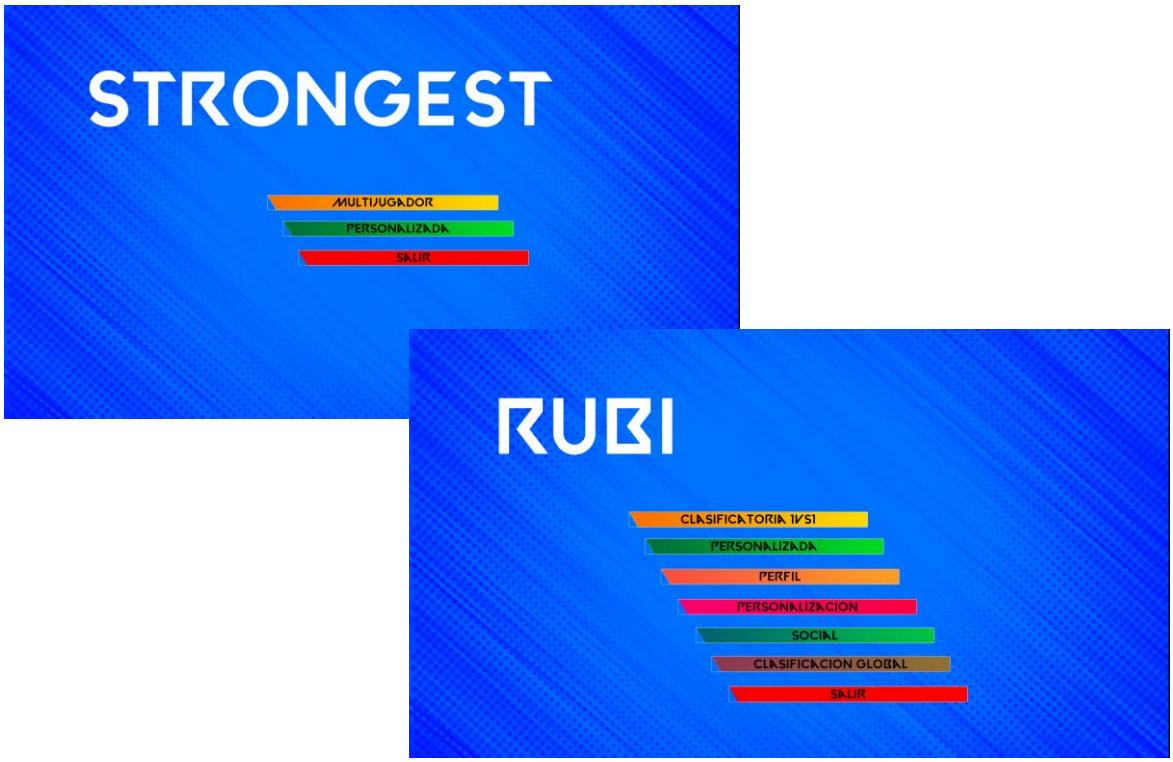
Para ofrecer una explicación más técnica de la implementación y organización de todo el juego explicando todos los archivos, atributos, operaciones, eventos… Se ha construido el anexo V de este proyecto.

La distribución de la implementación se ha realizado en cuatro subsistemas que van a ser los explicados en los siguientes subapartados.

5.7.1.- Subsistema Usuario

Ha sido implementado mediante el uso de Blueprints en el juego, pero también en este se ha llevado a cabo las llamadas a la API de Playfab dónde se almacenaban los distintos datos de los usuarios, necesarios para su gestión, la gestión de sesiones de usuario, estadísticas (número de asesinatos y victorias totales de cada jugador), monedas…

Este subsistema también incluye gran cantidad de interfaces para mostrar al usuario en la que se ha priorizado la sencillez, claridad y el minimalismo. Estas interfaces hacen llamadas a la instancia del juego la cual ya realiza las llamadas a la API de Playfab, para la gestión de usuarios y la recuperación de información de la base de datos y además proporcionan al usuario la capacidad de acceder a toda la funcionalidad del juego. Algunas capturas de pantalla serían las siguientes (Figura 40):



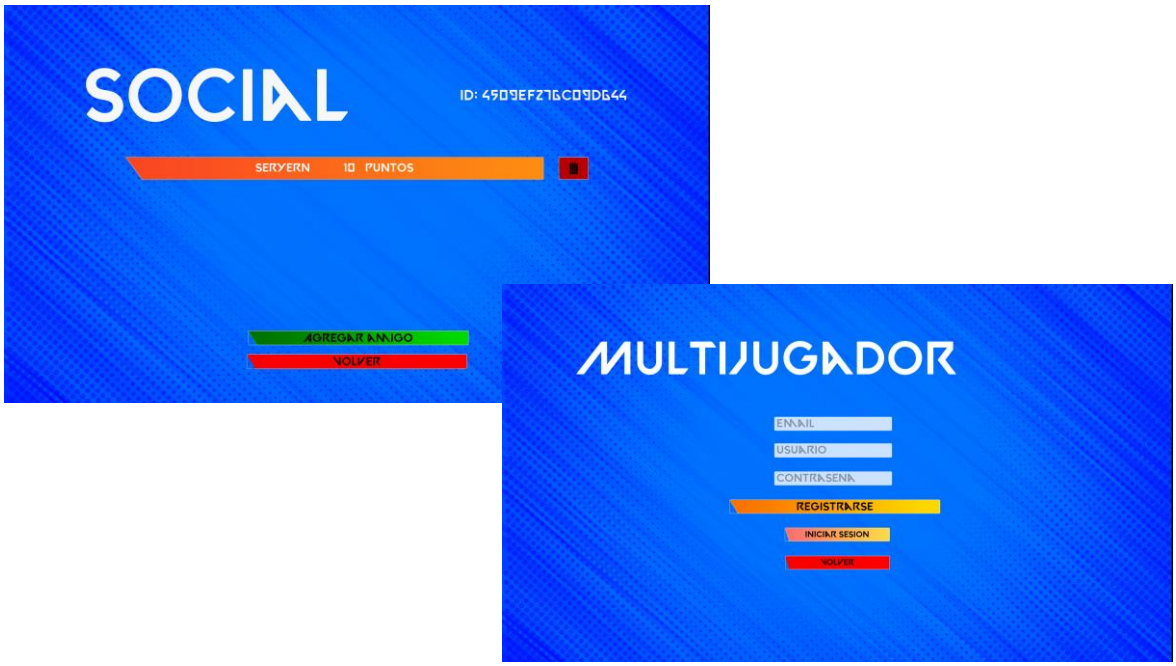


Figura 40 Capturas de pantalla de la interfaz del juego

5.7.2.- Subsistema Partida

Ha sido implementado en su totalidad mediante blueprints, pero ha realizado ciertas llamadas a la base de datos de Playfab para actualizar las estadísticas de los jugadores al finalizar las partidas. La funcionalidad implementada por esta ha sido la básica de shooter y alguna adicional, como disparos, recargas, reaparecer, control de puntuaciones personal y de equipo, armas…

Algunas capturas del desarrollo de las partidas, producto de la implementación de este subsistema serán las siguientes (Figura 41):

Figura 41 Capturas de pantalla de la partida

5.7.3.- Subsistema Economía

Mediante la herramienta de Playfab he desarrollado una moneda del juego; Strongest Coin “SC”, inventario (guardadas en base de datos los dos para cada usuario) y catálogo. El catálogo estará compuesto por diferentes ítems perfectamente identificados por un ID, también poseerán atributos como nombre, precio y duración. Los usuarios podrán comprar ítems de este catálogo mediante la moneda del juego y añadirán el ítem a su inventario, para luego después poderlo usar en las partidas.

A continuación, se mostrará la pantalla de tienda (Figura 42) para un usuario que no posee un ítem, y la segunda se mostrará a otro que sí lo posee:

Figura 42 Capturas de pantalla de la interfaz personalización dónde se puede observar la tienda

5.7.4.- Subsistema Clasificación

Finalmente, también mediante la herramienta Playfab implementaría el elemento clasificación, así como la tabla de premios. Se implementa una clasificación global en base a una puntuación clasificatoria de cada usuario (almacenada en base de datos), además se automatiza el reinicio de esta una vez al mes y la entrega de premios con la misma frecuencia. Los premios también los he implementado de manera personalizada: 10 SC al primero, 5 SC al segundo y 3 SC al tercero. En la figura 43 se puede observar cómo se ve la clasificación mediante la interfaz del juego y los puntos clasificatorios propios del jugador mediante la pantalla perfil, segunda captura de la figura.

Figura 43 Captura de la interfaz clasificación con los mejores jugadores, y de la interfaz perfil con las estadísticas del usuario

5.8.- Pruebas

Las pruebas es un apartado muy importante en todo proyecto debido a que nos permite detectar errores y situaciones inestables que podemos evitar a tiempo corrigiéndolas y además nos permite comprobar que se logran todos los requisitos del proyecto, así como los requisitos funcionales y no funcionales.

La realización de las pruebas a cada uno de los componentes implementados se ha ido desarrollando a lo largo de toda la implementación del proyecto, cada vez que se implementaba un componente se realizaban las pruebas unitarias sobre él y cuando se han implementado y probado más componentes se hacen las pruebas del sistema con todos implementados. Todas las pruebas que han dado errores o lugar a situaciones inestables se han corregido.

Incluso para aportar mayor fiabilidad a nuestro juego se ha implementado un control de errores en diversas partes del programa de manera que sea poco probable llegar a situaciones de inestabilidad.

Al final de todo el desarrollo se han llevado a cabo varias pruebas al sistema completo (de funcionalidad, rendimiento, sincronización, tiempos de respuesta…), comprobando que en cada una de ellas se llega a los resultados esperados y que no ocurra ningún error que empeore la experiencia de usuario. Estas pruebas han resultado todas correctas, pero debemos de tener en cuenta que es un juego que permite partidas Online a través de internet y por lo tanto también dependemos de este para su perfecto funcionamiento, lo que conlleva también a que si no existe una conexión estable observaremos retrasos en el juego que no se pueden controlar.

6.- Conclusiones

En este penúltimo apartado voy a desarrollar las conclusiones personales obtenidas al finalizar el desarrollo de este proyecto.

En cuanto al resultado del juego final estoy satisfecho con lo conseguido porque he logrado de cierta manera el juego casi ideal que deseaba.

He conseguido cumplir con el juego todos los objetivos planteados al principio del proyecto tanto funcionales como no funcionales, además de poder incluir en un solo juego las características que deseaba de los juegos más destacables de los últimos años, partidas rápidas, multijugador, sistema muy competitivo, shooter más o menos completo puro (sin funcionalidades raras adicionales como jet packs), estética futurista en un punto intermedio de cartoon y realista, menús, pantallas y controles muy sencillos… En general, estoy contento con el juego y lo consumiría.

Los objetivos conseguidos han sido los siguientes:

- Gestión de usuarios: He conseguido implementar en el sistema una base de datos capaz de realizar toda la funcionalidad de gestión de usuarios de manera muy satisfactoria, además de almacenar para cada uno de ellos información específica como lo son las victorias, puntos clasificatorios y asesinatos.

- Partidas estilo shooter: He conseguido implementar un sistema de partidas shooter que permite a los jugadores usar y coger distintas armas, disparar, recargar, correr, agacharse, saltar, apuntar, un sistema de retroceso, animaciones (de disparo, de recarga, al recibir un impacto…), sistemas de partículas de impacto distintos según la superficie en la que impacta, sistemas de partículas en disparos, niagara system implementados en el punto donde impacta el disparo… En resumen, un sistema de partidas shooter completo y estéticamente bueno.

- Multijugador: Mediante el uso de sincronización de información, llamadas remotas e intercambio de mensajes entre las instancias del juego de cada uno de los jugadores, he logrado crear un sistema de partidas multijugador en el que todos los jugadores pueden observar el desarrollo de la partida de manera equivalente sincronizada, para que así puedan participar en ella de un modo correcto. El multijugador como dije dota al juego de la característica de que cada partida es diferente, lo hace mucho más entretenido y hace cada partida distinta.

- Online: Una vez implementado el multijugador en local he hecho dar el salto al juego al online mediante la gestión de sesiones de partidas a través del subsistema Steam.

- Clasificación global: He conseguido establecer una clasificación global para todos los usuarios del juego, la cual puede ser accedida por todos y que dará la característica de competición propia de un juego de E-sport.

- Gestión de amigos: He incluido también a las características base de gestión de usuarios la gestión de amigos; adición y eliminación de ellos a través de su email o su ID en el juego. Además, para seguir favoreciendo la competición en el juego entre amigos entre ellos pueden ver y comparar sus puntos clasificatorios.

- Personalizable: He implementado en el juego un sistema de compra de diferentes ítems equipables (skins) con las monedas del juego para que cada jugador pueda ser diferente al resto dentro del juego, dando un toque personal a su personaje.

- Economía: He implementado un sistema de economía en el juego basado en la existencia de una moneda Strongest Coin (SC), que podrá ser conseguida por los jugadores quedando en los puestos más altos de la clasificación y que podrá ser posteriormente utilizada para comprar los diferentes ítems.

- Variedad de armas y mapas: Para el juego se han creado varias armas y mapas con el objetivo de que el jugador no se acomode o se aburra de un juego sin cambios.

- Estadísticas: La base de datos de usuarios también nos permitirá guardar información sobre estos como sus puntos clasificatorios, asesinatos y victorias, es decir, sus estadísticas.

- Tiempos de respuesta bajos: Para tener tiempos de respuesta bajos o al menos estos se consigan de manera transparente, el sistema realizará la petición de información a la base de datos cuando se prevé que el usuario o el juego van a usarla o procesarla pronto. Por ejemplo, nada más se inicia sesión se obtienen las estadísticas del usuario para que si entra a perfil las pueda observar al instante o para que al terminar una partida clasificatoria y se deba actualizar los asesinatos del usuario no se vuelva a realizar la petición y se actualice con una única de actualización a la base de datos.

- Usabilidad: La interfaz implementada es sumamente sencilla y clara para que todo tipo de jugador pueda acceder a toda la información y funcionalidad del juego de manera intuitiva fácilmente.

- Portabilidad: Gracias al uso de funciones genéricas (como primaryAction) podemos utilizar nuestro juego en todo tipo de sistemas operativos para PCs ya que no son funciones vinculadas a un solo sistema operativo.

- Rendimiento: Se ha realizado toda la implementación del juego buscando el uso mínimo de recursos, es decir, buscando el uso de funciones simples que no requieran cálculos complejos.

- Fiabilidad: Se ha implementado un control de errores para no llegar a situaciones incoherentes en el juego, por ejemplo, en las llamadas a la base de datos se hace un control para saber si el usuario ha iniciado sesión para evitar actualizaciones o consultas fraudulentas en la base de datos.

- Privacidad: La base de datos por sí misma es lo suficiente segura para no permitir a terceros el acceso a sus datos, es decir, al de los usuarios.

He de añadir también que he implementado el juego para que sea muy fácilmente escalable, añadir más variedad de mapas, armas, camuflajes para las armas, skins animaciones… Todo esto último mencionado no necesitaría mucho más esfuerzo debido a que la funcionalidad y las bases para ello ya están desarrolladas.

En cuanto a lo aprendido en el desarrollo del proyecto también me encuentro muy satisfecho:

- He ganado mucha agilidad con el manejo de los Blueprints/Nodos. En mi opinión pienso que puede ser el método de programación del futuro (respaldado por mi experiencia de prácticas de empresa dónde el 70% de la funcionalidad se hacía mediante nodos).

- He controlado mucho más todos los apartados de Unreal Engine:

    - He aprendido a crear animaciones propias, el significado de control rings, secuencias, retargeteo de personajes…

    - He aprendido a controlar y crear los archivos básicos de cada nivel, de gran manera nuevos para mí, como lo son el game state, player state, la alternación entre game modes, el uso del player controller para separar la funcionalidad de los controles y widgets…

    - He aprendido el uso de las interfaces y su importancia en los juegos multijugador.

    - He aprendido a usar las llamadas al servidor y el paso de información mediante multicast.

    - He aprendido sobre replicación de los elementos y cómo se puede llevar a cabo la sincronización

    - He aprendido nuevos tipos de elementos aprovechables como los Niagara System.

    - Me he introducido en el mundo de Unreal Engine 5, el que según muchos de los expertos va a ser el referente en el mundo de los videojuegos en los próximos años, sin tener mucha documentación al respecto sobre él, pero llevándome muchas ideas y una buena opinión de él.

- Sincronización, replicación y sistemas distribuidos al final. He aprendido a mantener réplicas con una misma información en varias máquinas distintas manteniéndolas coherentes mediante el uso de mensajes al servidor y multicast.

- He aprendido mucho sobre la herramienta Playfab. He adquirido experiencia sobre el uso de su potencia para poder desarrollar un buen juego completo.

- He seguido aprendiendo sobre las bases de datos, así como su acceso mediante APIs y el uso del formato JSON para su recuperación y actualización.

- Por último, he aprendido muchísimo sobre la estructuración, planificación y documentación de proyecto sobre todo gracias a los anexos y en el campo del Proceso Unificado y el uso de UML.

7.- Líneas de trabajo futuras:

En este último apartado voy a comentar las líneas futuras y actualizaciones que sobre las que podría evolucionar el juego:

- Modo de juego dominio: Es un modo de juego que siempre me ha llamado la atención y por ello desde las fases medias de la implementación he intentado implementarlo. Existe una carpeta dentro de los archivos del juego “STRONGEST/Partidas/Dominio” donde se puede encontrar una primera implementación del modo de juego. En esta carpeta se incluye los personajes, armas, modo de juego y todas las reglas necesarias para implementar el nuevo modo. Incluso los elementos de bandera con su apartado dinámico de cambios de estado, altura de la bandera, color… Debido al tiempo limitado del proyecto no he podido implementarlo al final completo el modo dominio multijugador; en varias instancias del juego, pero de manera local ya está implementado al completo.

- Mayor variedad de armas: El sistema de armas, así como el de su recogida y asociación al jugador ha sido implementado de tal manera que no haga falta añadir más código para la nueva arma a implementar que cambiarle los valores de daño, cadencia, cargador y munición en su blueprint, heredando las funciones del blueprint de cualquier arma. No hace falta añadir nada en el blueprint del jugador ni en ningún otro lugar, por lo tanto, este proceso de escalado es muy sencillo

- Más modos de juego: El sistema mantiene un sistema estable de partidas, por lo tanto, sobre él se puede construir un nuevo modo de juego fácilmente como atrapar la bandera, buscar y destruir, juego de armas…

- Inteligencia artificial: Se podrá añadir en el juego también fácilmente, con el sistema de partidas actual, bots controlados por AI. De manera que se podrá crear un apartado dentro de la partida personalizada que permita jugar contra bots a modo de entrenamiento.

- Con más presupuesto, se podrían incluir en el juego un conjunto de servidores remotos para mantener las diferentes sesiones de partidas y que así no dependan de un usuario. También de la misma manera se podría gestionar colas distintas para los distintos tipos de partidas.

8.- Referencias

[1] Soloaga, A. & Ver todas las publicaciones de Ana Soloaga. (2019, 19 julio). *Unreal Engine, qué es y para qué sirve*. El Blog de Akademus. https://www.akademus.es/blog/emprendedores/unreal-engine-que-es-y-para-que-sirve/

[2] Hernández, A. (2022, 12 abril). *Unreal Engine 5, la próxima revolución*. Esports México. https://theesportsmexico.com/2022/04/12/tech/unreal-engine-5-la-proxima-revolucion#:%7E:text=Lo%20que%20ofrece&text=Unreal%20Engine%205%20tambi%C3%A9n%20ofrece,de%20ruta%20mejorado%2C%20entre%20otros

[3] González, A. C. (2022, 7 marzo). *Unreal engine, todo lo que necesitas saber*. Profesional Review. https://www.profesionalreview.com/2022/04/17/unreal-engine-todo-lo-que-necesitas-saber/

[4] *Visual Paradigm*. (2018, 10 diciembre). Capterra. https://www.capterra.es/software/145716/visual-paradigm

[5] EcuRed. (s. f.). *Visual Paradigm - EcuRed*. https://www.ecured.cu/Visual_Paradigm

[6] *Online Subsystem Steam*. (s. f.). Unreal Engine Documentation. https://docs.unrealengine.com/4.27/en-US/ProgrammingAndScripting/Online/Steam/

[7] M. (2022, 15 marzo). *Advanced Sessions Plugin*. Unreal Engine Forums. https://forums.unrealengine.com/t/advanced-sessions-plugin/30020

[8] *PlayFab*. (s. f.). Microsoft Azure. https://azure.microsoft.com/es-es/services/playfab/

[9] Microsoft. (s. f.). *Software de administración de proyectos | Microsoft Project*. https://www.microsoft.com/es-es/microsoft-365/project/project-management-software

[10] colaboradores de Wikipedia. (2021, 13 diciembre). *Microsoft Project*. Wikipedia, la enciclopedia libre. https://es.wikipedia.org/wiki/Microsoft_Project

[11] *Mixamo*. (s. f.). Mixamo. https://www.mixamo.com/#/

[12] Microsoft. (2022, 23 junio). *Visual Studio: IDE y Editor de código para desarrolladores de software y Teams*. Visual Studio. https://visualstudio.microsoft.com/es/

[13] *ArtStation*. (s. f.). ArtStation. https://www.artstation.com/?sort_by=community

[14] *Unreal Engine Marketplace | Store of UE Assets for Games and 3D Rendering*. (s. f.). Unreal Engine. https://www.unrealengine.com/marketplace/en-US/store

[15] *Curso de Blueprints en Unreal Engine*. (2019, 5 junio). school-ing. https://school-ing.es/curso-blueprints-unreal-engine/#:%7E:text=El%20sistema%20de%20blueprints%20es,de%20un%20sistema%20de%20nodos

[16] A. D. (2022, 7 marzo). *¿Qué es JSON?* Tutoriales Hostinger. https://www.hostinger.es/tutoriales/que-es-json

[17] Robledano, A. (2021, 24 agosto). *Qué es C++: Características y aplicaciones*. OpenWebinars.net. https://openwebinars.net/blog/que-es-cpp/

[18] *Qué es el lenguaje unificado de modelado (UML)*. (s. f.). Lucidchart. https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml

[19] Dr. Francisco José García Peñalvo, Dra. Alicia García Holgado, & Departamento de Informática y Automática - Universidad de Salamanca. (2022). *Tema 5: Introducción al Proceso Unificado*. USAL. https://repositorio.grial.eu/bitstream/grial/2537/1/IS_I%20Tema%205%20-%20Proceso%20Unificado.pdf

[20] María N. Moreno García. (2021). *Gestión de Proyectos Tema 4: Planificación temporal de proyectos*.

[21] Universidad de Sevilla, Amador Durán Toro, & Beatriz Bernárdez Jiménez. (2000, octubre). *Metodología para la Elicitación de Requisitos de Sistemas Software* (Versión 2.1).