
Control Remoto de Videojuegos con Smartphones



MEMORIA DE TRABAJO DE FIN DE GRADO

Pablo Gómez Calvo
Sergio J. Higuera Velasco

Grado de Desarrollo de Videojuegos
Facultad de Informática
Universidad Complutense de Madrid

Mayo 2020

Control Remoto de Videojuegos con Smartphones

Memoria de Trabajo de Fin de Grado
Grado de Desarrollo de Videojuegos
Abril 2020

Director: Carlos León Aznar
Director: Pedro Pablo Gómez Martín

Versión 0.3

Grado de Desarrollo de Videojuegos
Facultad de Informática
Universidad Complutense de Madrid

Mayo 2020

Copyright © Pablo Gómez Calvo y Sergio J. Higuera Velasco

A todo aquel que confió en nosotros

Agradecimientos

Nadie es innecesario.

Yitán, Final Fantasy IX

El primer agradecimiento hay que dárselo a la Universidad Complutense por aceptar la creación de este grado, un grado que demuestra la importancia del mundo de los videojuegos en la sociedad actual. Con este grado se han conseguido romper muchas barreras, entre ellas está poder especializarse y adoptar los videojuegos como nuestra profesión.

Dar gracias a los profesores que nos han acompañado estos años y que han contribuido en el desarrollo del grado. Una mención aparte para las dos personas que han hecho posible la realización de este Trabajo de Fin de Grado, Carlos León Aznar y Pedro Pablo Gómez Martín.

Nuestro último agradecimiento va dirigido a nuestras familias por soportarnos en todos nuestros momentos durante el grado.

Resumen

¡No estáis preparados!

Illidan Tempestira, World of Warcraft

Índice

Agradecimientos	VII
Resumen	IX
1. Introducción	1
1.1. Introducción	1
2. Entrada de usuario en videojuegos	3
2.1. Evolución de los dispositivos de entrada	3
2.2. Herramientas para el desarrollo	7
2.2.1. Unity	8
2.2.2. Android Studio	8
2.2.3. ZXing	9
2.2.4. Android SDK	9
2.3. Interfaces de entrada	9
2.3.1. Wii U	10
2.3.2. Controlador de videojuegos inalámbricos	10
2.3.3. PlayLink	10
2.3.4. PS4 Remote Play	11
3. Objetivos y especificación	13
3.1. Objetivos	13
3.2. Plan de trabajo	14
3.3. Metodología	14
3.4. Herramientas utilizadas	15
4. Especificación	17
4.1. Introducción	17
4.2. Funcionalidad	17
4.3. Protocolo de comunicación	17
5. Implementación	19

5.1. Android	19
5.2. Unity	19
6. Pruebas con usuario y resultado	21
6.1. $\hat{A}_i?$	21
7. Conclusiones	23
Índice alfabético	25

Índice de figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Introducción

La industria del videojuego desde su inicio nos ha enseñado que la innovación a la hora de crear experiencias nuevas para los usuarios es algo que enriquece a muchos jugadores, por ende se está ayudando a que la experiencia de juego sea cada vez más cómoda y flexible para los jugadores.

Es por esta razón que empresas como Electronic Arts, Ubisoft, Kunos Simulazioni y Polyphony Digital, entre otras, dedican gran cantidad de sus recursos a hacer realidad muchas experiencias que los usuarios quieren tener como, por ejemplo, jugar a juegos deportivos realistas como en FIFA o conducir automoviles de competición por un circuito famoso como en Assetto Corsa.

Pero para invertir en crear este nuevo tipo de estilos de juego se necesita una gran financiación, cosa que estudios pequeños no tienen. Estos estudios independientes usan motores como Unity3D o Unreal Engine 4. Estos motores, cuyo pago es porcentual a las ganancias obtenidas por tu juego o en algunos casos mensual, ofrecen una gran cantidad de herramientas a disposición de sus usuarios para que los desarrolladores puedan ahorrar tiempo de implementación de una nueva característica y lo utilicen para que su juego siga creciendo. En el caso de Unity el pago es mensual dependiendo de la cantidad de dinero generado por el usuario o su empresa; los precios van desde gratuito si hace menos de 100 mil dólares anuales hasta 125 dólares mensuales si el usuario o su empresa genera más de 200 mil dólares anuales. Si hablamos del pago de Unreal Engine, si cualquiera de los productos realizados por el estudio se comercializa de forma oficial, Epic Games obtendría el 5 % de los beneficios de la obra cada trimestre cuando este producto supere sus primeros 3000 dólares. El desarrollo para dispositivos móviles en estudios independientes ha crecido de manera exponencial gracias a que motores como Unity lo hacen bastante accesible.

Unity es utilizado por el 34 % del top mil de juegos para dispositivos

móviles. Algunos de estos juegos son *Alto's Adventure*, *Monument Valley* o el famoso *Hearthstone* de Blizzard en su versión de Android e iOS.

Entonces, ¿Y si un juego se pudiese jugar en el teléfono móvil pero en verdad el juego se estuviese ejecutando en el ordenador?

Con esta pregunta se pretende poner encima de la mesa las tecnologías desarrolladas por diferentes empresas como **Nintendo** con Nintendo Switch que alterna modo portátil con modo sobremesa, lo que da una flexibilidad a la hora de jugar donde quieras que nunca se había experimentado, como **Sony** con sus aplicaciones **PS4 Remote Play** y **PS4 Second Screen** que te dan la posibilidad de controlar de manera remota e incluso jugar desde tu dispositivo iOS o Android.

Para la conexión de una consola o un ordenador a un dispositivo móvil o tablet es necesaria una conexión a internet estable. El tiempo de respuesta, lo que se conoce como *Input Lag*, puede llegar a convertirse en un quebradero de cabeza para un estudio pequeño ya que requiere de pruebas de rendimiento, pruebas con usuarios y pruebas con diferentes anchos de banda de red para buscar posibles cuellos de botella.

Este trabajo pretende aplicar los modelos propuestos anteriormente y crear una herramienta libre para el motor de videojuegos Unity con la finalidad de dar a estudios independientes y con escasa o nula financiación la oportunidad de habilitar nuevo gameplay para sus usuarios. La finalidad es dar todas estas herramientas de una forma rápida e intuitiva con documentación y una prueba de implementación de este Plugin con uno de los juegos que se ofrecen de manera gratuita en la tienda de Unity, la Asset Store. Además de esto, la herramienta consta de una licencia libre, lo que da la posibilidad de ampliación y modificación dependiendo de las necesidades de cada usuario/estudio.

Capítulo 2

Entrada de usuario en videojuegos

2.1. Evolución de los dispositivos de entrada

Los dispositivos de entrada son aquellos que permiten introducir datos o información en un ordenador para que este los procese u ordene. Otro término usado para estos dispositivos es periférico. A pesar de que este término implica a menudo el concepto de adicional y no esencial, en muchos sistemas informáticos son elementos fundamentales. Estos dispositivos de entrada pueden clasificarse según el tipo de entrada, ya sea sonora, visual, de movimiento mecánico, etc. o dependiendo de si su forma de entrada es discreta (pulsaciones de teclas) o continua (una posición).

Todos estos dispositivos de entrada son conocidos como **HID, Human Interface Device**. La principal motivación para HID era la de permitir innovaciones en los dispositivos de entrada a los ordenadores y así simplificar el proceso de instalación de este tipo de dispositivos. Antes de HID, los dispositivos de entrada se ajustaban a unos estrictos protocolos diseñados para ratones, teclados y joysticks. Con cualquier innovación en el hardware se requería de sobrecargar el protocolo existente o la creación de un driver. Un solo driver HID analiza los datos de entrada y permite una asociación dinámica de estos datos con la funcionalidad descrita por la aplicación. En el protocolo HID existen 2 entidades: el host y el dispositivo. El dispositivo es la entrada que intercatúa directamente con el humano, un ejemplo puede ser el teclado o ratón. El host es el que recibe los datos de entrada del dispositivo con las acciones ejecutadas por el humano, el host suele ser un ordenador. Los dispositivos definen sus paquetes de datos y luego presentan

un descriptor HID al host. El descriptor HID es codificado como un grupo de bytes que describen los paquetes de datos del dispositivo. Esto incluye: cuantos paquetes soporta el dispositivo, el tamaño de los paquetes, y el propósito de cada byte y bit en el paquete. Se espera que el host sea la entidad más compleja de las 2 ya que el host necesita obtener el descriptor HID del dispositivo y analizarlo antes de establecer la comunicación. El HID también define el modo arranque, este modo es limitado ya que paquetes de datos son utilizados durante ese modo. Los únicos dispositivos que soportan el modo arranque son teclados y ratones.

Un teclado es un HID que se representa como una disposición de botones o teclas. Cada una de estas teclas se puede utilizar para ingresar cualquier carácter lingüístico o hacer una llamada a cualquier función particular del ordenador. Los teclados tienen su inspiración en las máquinas de escribir que tienen su origen en el año 1877, cuando la marca Remington comenzó a comercializar de manera masiva las máquinas de escribir. Uno de los primeros avances de estas máquinas de escribir ocurrió en la década de 1930, cuando se combinaron la tecnología de la entrada e impresión de las máquinas de escribir con la tecnología de la comunicación del telégrafo. Además, los sistemas de tarjetas perforadas también se combinaron con las máquinas de escribir para crear perforadoras de teclado que fueron la base para las primeras calculadoras. La emergente máquina de escribir eléctrica mejoró aún más la unión tecnológica entre la máquina de escribir y el ordenador. Así, en 1955, el Whirlwind del MIT, se convierte en el primer ordenador del mundo que permite a sus usuarios introducir comandos a través de un teclado y confirma lo útil y conveniente que puede ser un dispositivo de entrada de teclado. En 1964 se impulsó el desarrollo de una nueva interfaz de usuario llamada terminal de visualización de video que permitió a los usuarios de las primeras computadoras ver qué caracteres estaban escribiendo. En los años 80, IBM lanzó su primer ordenador personal que incluía un teclado mecánico. Actualmente las principales mejoras que han sufrido los teclados de ordenador se basan en la eliminación de cables gracias al Bluetooth o WI-FI. Con la llegada de los dispositivos táctiles se añadió además el concepto de teclado virtual. Este teclado virtual elimina el uso de un teclado hardware para pasar a un teclado software que imita el teclado tradicional QWERTY pero en una pantalla táctil.

Además del teclado, el segundo dispositivo de entrada por excelencia es el ratón. La primera maqueta fue diseñada durante los años 60, disponía de 2 ruedas metálicas que al desplazarse por una superficie movían 2 ejes. Cada uno de estos ejes controlaba el movimiento tanto vertical como horizontal del cursor en la pantalla. Durante los años posteriores se añadieron botones, una rueda central o lateral para el desplazamiento, el sensor de movimiento óptico por diodo led o un sensor basado en un láser no visible. En los primeros años de la informática, el teclado era el dispositivo más popular para la

entrada de datos pero la aparición y éxito del ratón, junto con la evolución de los sistemas operativos, lograron facilitar y mejorar la comodidad a la hora de manejar ambos periféricos a la vez. Los ratones suelen estar preparados para manejarse con ambas manos pero algunos fabricantes también ofrecen modelos específicos para zurdos y diestros. Los sistemas operativos también han ayudado a que ambos tipos de personas tengan un uso satisfactorio de los ratones. Con el avance de los nuevos ordenadores, el ratón se ha convertido en un dispositivo esencial a la hora de jugar videojuegos, sirviendo no solo para seleccionar y accionar objetos en pantalla en juegos de estrategia, sino para controlar la cámara o cambiar la dirección del personaje en juegos de primera y tercera persona.

Los videojuegos han sido los principales responsables de la evolución de los dispositivos de entrada en las décadas posteriores a los años 70. En 1972 fue lanzada de manera oficial la **Magnavox Odyssey** y fue considerada la primera videoconsola. El dispositivo de entrada para poder jugar consistía de 2 diales que se utilizaban para el movimiento horizontal y vertical del personaje, un cuadrado blanco. Como juego para la consola Odyssey sacaron el Magnavox Odyssey Shooting Gallery en 1972. El mando que se usaba para poder jugar a este juego tenía la forma de un rifle y la peculiaridad que tenía este accesorio/mando era que los disparos se registraban siempre y cuando el rifle apuntase a una luz intensa por lo que era muy fácil de engañar si no fuese porque el juego no disponía de un registro de puntos. En 1976 la empresa Fairchild Semiconductor sacó al mercado la **Fairchild Channel F** cuya característica principal a nivel de entrada de usuario fue la incorporación de un joystick de 8 direcciones. Además de ofrecer un movimiento en 8 direcciones, la parte de arriba de este mando podía girarse para ser compatible con juegos como *Pong* y también podía ser pulsado y usarse normalmente como botón de disparo. En 1977 salió al mercado uno de los joysticks más famosos. Este joystick es el que se utilizaba en la consola **Atari 2600**. Este joystick se conocía como el **Atari CX40** y consistía de una palanca que permitía un movimiento en 8 direcciones y un botón. Junto con este modelo, Atari sacó al mercado un tipo de conexión que se convertiría en el estándar de la industria y que sería compatible con sistemas posteriores. Unos pocos años después, en 1982, Atari lanzó su nueva consola Atari 5200. Además de mejorar gráficamente, el controlador que llevaba incorporado proviene de un kit de controlador de un avión RC. El sistema combinaba un diseño mecánico demasiado complejo con un sistema de circuito flexible interno de muy bajo coste. Este controlador incluyó un botón de pausa, una característica única en ese momento.

En 1983 Nintendo sacó al mercado su **Nintendo Entertainment System (NES)** cuyo controlador sentó unas bases en el control del movimiento gracias a la cruceta que incorporaba. Esta cruceta permitía un movimiento en 4 direcciones y que pretendía reemplazar a las voluminosas palancas de

mando de los controladores. Además de la cruceta, el mando disponía de 2 botones redondos (A y B) y otros 2 botones rectangulares (START y SELECT). En lo sucesivo, se lanzaron varios dispositivos especiales diseñados precisamente para usarse con juegos específicos, aunque muy pocos de estos se volvieron populares. Uno de estos dispositivos era el **Power Glove**, el que sería considerado como uno de los primeros periféricos de interfaz en recrear los movimientos de la mano en una pantalla de televisión o de un ordenador en tiempo real. En 1990 Nintendo hizo evolucionar a la Nintendo NES y lanzó la **Super Nintendo Entertainment System (SNES)**, la cual dejó atrás un diseño cuadrado del controlador y se inclinó por un diseño más ergonómico, se mejoró la cruceta y se añadieron otros 2 botones (X e Y). El tiempo de respuesta del controlador era de 16 milisegundos. Dentro de la evolución de los controladores, en 1993 la compañía SEGA sorprendió con el lanzamiento de un nuevo accesorio para su consola **Sega Mega Drive**. Este accesorio consistía en un aro octogonal que se colocaba en el suelo y se conectaba directamente al puerto de controlador de la consola. Lo llamaron **Sega Activator** y fue el primer controlador que utilizaba el cuerpo completo. El jugador se tenía que situar en el centro del aro, el cual emitía rayos infrarrojos hacia arriba para detectar los movimientos del jugador. Los juegos orientados para el Sega Activator eran juegos que involucrasen el movimiento de brazos y piernas para que el jugador cruzase los rayos infrarrojos y así se detectase el movimiento. Al tratarse de 8 segmentos, cada uno de estos segmentos estaba mapeado como si fuera un botón en el mando tradicional, el cual se "pulsaría" cada vez que el jugador cruzase un segmento de los rayos infrarrojos.

Durante los años 90 **Sony** entró al terreno del desarrollo de consolas y por consecuencia, de modelos diferentes de controladores de videojuegos. Con su primera consola, la **Sony PlayStation**, incluyeron un nuevo diseño de mando que recogía muchos de los diseños vistos hasta el momento. A diferencia de Nintendo, este controlador cambió la nomenclatura de los botones A, B, Y y X por las figuras \triangle , \circ , \times y \square , mantenía la cruceta y los botones START y SELECT y además añadió 4 botones más en la parte lateral del mando para los dedos índice y corazón. 3 años más tarde Sony sacaría una re-edición del mando al que le incorporaron 2 sticks analógicos junto con un botón con un LED para cambiar entre los diferentes modos usados para el control del personaje. Este modelo fue el predecesor del famoso **DualShock** y únicamente la versión japonesa presentaba una función de retroalimentación de vibración. Por el lado de Nintendo, la consola sucesora de la Super Nintendo fue la **Nintendo 64** que fue acompañada por un nuevo diseño de mando que no pasó desapercibido. Disponía de una cruceta en la parte izquierda del mando, un stick de 360 grados y un botón START en el centro del mando y 6 botones en su parte derecha. Complementario a esto, en la parte trasera del mando había 2 botones más y también en la parte trasera

se daba la opción de introducir un dispositivo extraíble que proporcionaba retroalimentación de vibración. Este accesorio se activaba en ocasiones concretas como al disparar un arma y servía para sumergir al jugador en el videojuego.

Un Repaso de los diferentes dispositivos de entrada como son:

- Teclados
- Ratones

Historia a parte de los diferentes mandos que hubo con datos relevantes que encuentre como frecuencia y demás, eso es más trabajo de dedicar tiempo a leer y buscar toda esta información y plasmarla al final en una tabla comparativa:

- recreativas
- mandos clásicos con conexión por cable
- mandos inalámbricos como los de PS3 y Xbox360 (utilización del bluetooth, infrarrojos, velocidad, latencia)
- mandos con acelerómetros como wii y nintendo switch
- Volantes y joysticks para simulación de vuelo: Explicación del force feedback, cuando se empieza a usar, implementación de este tipo de dispositivos en juegos y simuladores para el ejército.

Sistemas de streaming:

- qué es un streaming de imágenes.
- auge de este sistema con las retransmisiones en directo en plataformas como Youtube y Twitch (cifras de visualizaciones, monetización y demás datos que encuentre), TV por cable, Netflix y demás servicios de streaming.
- tabla con diferentes anchos de banda necesarios para diferentes configuraciones en las retransmisiones (720p, 1080p 30fps, 1080p 60fps, etc)
- estándares para la compresión de video e imagen.

2.2. Herramientas para el desarrollo

En este capítulo se expondrán las diferentes tecnologías que se van a utilizar para la realización de este proyecto. Además de eso, se pondrán encima de la mesa las diferentes aplicaciones desarrolladas por empresas del sector de los videojuegos que han sido tomadas como referencia para la creación de este trabajo.

2.2.1. Unity

Unity es un motor de videojuegos multiplataforma creado por Unity Technologies. Se encuentra disponible para sistemas Windows, Mac Os X y Linux. Es una de las herramientas de desarrollo de videojuegos más populares actualmente en el mundo de los desarrolladores independientes. Con unity se han realizado algunos de los juegos más famosos del mercado como **Ghost of a Tale**, **Cuphead** o **Hollow Knight**.

Las principales características buscadas son:

- **Multiplataforma.** El hecho de que Unity sea sistemas multiplataforma, te garantiza poder hacer juegos/aplicaciones que puedan ejecutarse en cualquier dispositivo a un coste bastante bajo en cuanto a esfuerzo.
- **Herramientas y servicios.** Unity es una herramienta que no engloba únicamente motores para el renderizado de imágenes, simulaciones físicas 2D y 3D, animación de personajes y audio sino que al tratarse de un motor que usa C# como lenguaje de scripting, dispone de todas las herramientas de .NET para el desarrollo. Uno de los servicios que ofrece Unity es **Unity Analytics** que ayuda a los creadores a realizar analíticas para ver cómo juegan sus jugadores.
- **Documentación.** Los desarrolladores de Unity ponen a disposición de los usuarios una documentación amplia y detallada, tanto que incluso disponen de un historial de versiones de la documentación por si trabajas con un proyecto de Unity con una versión no actualizada. Además de esto, ofrecen documentación sobre la API y ofrecen proyectos realizados por los propios desarrolladores del motor y tutoriales de como sacar el mayor rendimiento posible a las herramientas que ofrece el motor en su plataforma **Unity Learn**..
- **Comunidad.** Debido a que Unity es un motor de videojuegos demandado y gratuito, cuenta con un gran número de desarrolladores que saben utilizarlo, lo que facilita la solución de problemas más específicos. Unity tiene una comunidad muy amplia y muy activa tanto en foros como **Stack Overflow** y en su propio portal de preguntas **Unity Answers**..

2.2.2. Android Studio

Android Studio es el entorno de desarrollo oficial de la plataforma Android en contrapartida también cabe la posibilidad de descargar las Android SDK sin necesidad del entorno completo de Android Studio. Fue desarrollada por Google y sustituyó a Eclipse en el desarrollo de aplicaciones para dispositivos Android. Las SDK de Android son imprescindibles para la creación

de una aplicación Android, las SDK ofrecen un Emulador para poder ver tu aplicación en ejecución. Android Studio tiene la función de encapsular estas SDK de Android y poner en marcha la aplicación. Este IDE puede utilizarse tanto en Windows, Mac OS X y Linux pero únicamente puede usarse para el desarrollo de aplicaciones para Android.

Las principales características de Android Studio son:

- **Específico.** Si el producto que quieres desarrollar va a ser exclusivo de un sistema Android, con Android Studio te vas a centrar en explotar las funcionalidades de ese sistema al máximo sin tener que lidiar con diferentes sistemas que no te interesan.
- **Editor de diseño.** Android Studio cuenta con un editor visual para poder acomodar el layout de tu aplicación Android de una manera mucho más sencilla.

2.2.3. ZXing

Este es un proyecto del tipo Open-Source. Esta librería de procesamiento de imágenes de códigos de barras y QR's está implementada en Java y tiene diferentes versiones en los distintos lenguajes. En el caso de este proyecto, se ha usado la versión de .NET para usarla desde Unity. Al ser una librería que tiene soporte en muchos otros lenguajes que no son Java, puede ser utilizada en proyectos multiplataforma si estas plataformas usan diferentes lenguajes como en este caso con Java y C#.

2.2.4. Android SDK

Cada vez que Android saca una nueva versión de su sistema operativo le acompaña su correspondiente SDK. Estas SDK de Android incluyen librerías que son necesarias para el desarrollo para dispositivos Android, además de documentación del API, un debugger y un emulador para probar los proyectos sin necesidad de tener un dispositivo físico. Este SDK suele usarse acompañado a un IDE que como se ha explicado anteriormente en el punto 2.1.2 ha sido **Android Studio**.

2.3. Interfaces de entrada

Hay empresas de videojuegos como Nintendo y Sony que han invertido mucho dinero en innovar y crear nuevas formas para que los usuarios disfruten de los diferentes juegos. Algunos de los ejemplos que ya existen en el mercado y han servido de inspiración para la realización de este proyecto son:

2.3.1. Wii U

Wii U es la consola doméstica que Nintendo creó en la octava generación de consolas. La innovación principal de esta consola era la de cambiar radicalmente el modo de jugar a una consola de sobremesa ya que desarrollaron el Wii U GamePad. Este mando tenía la función de una segunda pantalla y la de un mando tradicional a la vez. Esta pantalla portátil es táctil y recibe una señal de 480p.

Este nuevo mando permitía a los desarrolladores tener un HUD mucho más limpio dentro del juego ya que, en muchos casos, esta pantalla era utilizada para poner elementos como el minimapa y en algunos juegos como Mario Bros, había cambios de zonas que convertían al mando en pantalla principal.

2.3.2. Controlador de videojuegos inalámbricos

Un controlador de videojuegos es un dispositivo de entrada cuya función es interactuar con los elementos de un juego para realizar diferentes acciones. Los controladores de videojuegos están extendidos tanto en ordenadores como en consolas y en el último año están saliendo al mercado nuevos mandos para dispositivos móviles, los cuales se conectan por bluetooth. Estos últimos son mandos físicos que no están al alcance de todos y actualmente no son compatibles con todos los juegos, se utilizan para juegos muy concretos como PUBG Mobile. Los controladores de videojuegos han ido cambiando su forma y el número de botones de los que disponen. La necesidad del uso de mandos en videojuegos de móvil hizo que Android 9 incluyese la posibilidad de hacer que un usuario conectase su Dualshock 3 a su Android para poder jugar.

2.3.3. PlayLink

PlayStation es una de las compañías que más tráfico de jugadores mueve llegando a la cifra de 90 millones de usuarios activos mensuales en enero de 2019, sus consolas son de las más vendidas en todo el mundo y para que esto sea posible siempre tienen que intentar estar a la cabeza de nuevos periféricos, nuevos juegos y, por supuesto, darles a sus usuarios las mejores experiencias posibles. En 2017, Sony PlayStation sacó al mercado una nueva serie de juegos llamados PlayLink. Estos juegos tienen una particularidad con respecto a un juego convencional de consola, estos juegos están hechos para jugarlos con gente y usando un dispositivo móvil como mando/controlador. Conectando la consola y el móvil a la misma red WIFI y conectándolos a través de una aplicación, se pueden conectar de 2 a 8 jugadores, depende de los que admita cada juego, para poder jugar en familia o con amigos.

2.3.4. PS4 Remote Play

En el último trimestre del año 2019, Sony lanzó la aplicación **PS4 Remote Play** para que los usuarios pudieran controlar su PlayStation 4 desde un dispositivo móvil. Esta aplicación tenía como requisito una conexión a internet de entre 5 y 12 Mbps por conexión LAN para una mejor experiencia. Esta aplicación permite no solamente el manejo de una PS4 usando un dispositivo móvil sino que además permite conectar un Dualshock 4 para controlar el juego y usar el móvil únicamente como pantalla.

Capítulo 3

Objetivos y especificación

3.1. Objetivos

Tal y como se ha podido ver en los capítulos anteriores, las empresas de videojuegos han puesto todos sus esfuerzos para adaptarse e integrar los dispositivos móviles dentro del mundo de los videojuegos. Con las tecnologías mencionadas en el capítulo dos como referencia directa, los objetivos que se plantearon para el proyecto son:

- Desarrollar una librería para Unity3D. Esta consistirá en un servidor que ofrecerá su IP y el puerto elegido para la comunicación mediante un QR que aparecerá en pantalla.
- Desarrollar una aplicación Android para usar el propio móvil como mando virtual. Esta aplicación utilizará la cámara para leer el código QR e iniciar una conexión con el servidor.
- Integrar las herramientas mencionadas en los puntos anteriores en un proyecto ya cerrado para demostrar su funcionalidad.

Con la integración del proyecto en un juego ya cerrado se pretende hacer una serie de pruebas con usuarios para ver el rendimiento de la herramienta en ordenadores y dispositivos móviles con diferentes características. Para eso se implementarán herramientas para la recogida de datos de los usuarios y las acciones realizadas durante las sesiones. Estos datos serán analizados en el capítulo 5 con mayor profundidad.

A continuación plantearemos la metodología y el plan de trabajo seguido.

3.2. Plan de trabajo

La primera fase estará dedicada a la investigación y el estudio de las herramientas ya existentes que exploran los aspectos en común con este TFG. Se usará Github¹ como sistema de control de versiones, donde estarán 2 repositorios: uno para la memoria y otro donde se encontrará el código de la demo. El uso de Github es debido a su amplio reconocimiento a nivel mundial, en enero de 2013 ya contaba con 3 millones de usuarios junto con 4.9 millones de repositorios. Unos datos más actuales indican que Github en Junio de 2018 alojaba casi 80 millones de proyectos. Dado a la gran cantidad de usuarios de dicha plataforma existe una gran ayuda para todo problema que surja.

Las reuniones con los directores serán cada 2-3 semanas, dependiendo de la disponibilidad y horarios. El objetivo de estas reuniones será llevar un control de lo que se haya avanzado y plantear las próximas 2-3 semanas de trabajo. Se han marcado unos hitos específicos:

1. **Hito 1:** Conexión entre Android y Unity3D establecida. Con esto se pretende que un personaje en Unity3D sea capaz de moverse gracias al Input que recibe desde Android.
2. **Hito 2:** Se deberá de haber conseguido tener una cámara en Unity3D enviando una imagen via streaming al dispositivo Android mientras se juega con el Input recibido del dispositivo móvil.
3. **Hito 3:** Aquí se debe tener una demo que sea capaz de demostrar las capacidades de la herramienta.
4. **Hito 4:** Pruebas con usuarios, recogida de información y análisis.

3.3. Metodología

La metodología utilizada para la realización de este proyecto está basada en el desarrollo ágil. Se han marcado pequeños objetivos de 2 semanas de duración llamados sprints hasta llegar a las metas grandes o hitos mencionados en el apartado anterior.

La metodología que se iba a usar estaba definida desde el primer día ya que es la que los miembros del proyecto siguen en todos los trabajos que realizan. Se decidió usar **Scrum**. Scrum es una metodología ágil que adopta una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del proyecto en cuestión. La particularidad que se tiene en este proyecto es el constante cambio de las tareas nuevas que se pueden llegar a incluir tras cada reunión.

¹<https://github.com/>

3.4. Herramientas utilizadas

Es resultado final del proyecto involucra tanto un ordenador como un dispositivo móvil conectados a la misma red. Para el desarrollo usaremos las siguientes herramientas software de comunicación, seguimiento y planificación:

1. **L^AT_EX**²

Para la realización de este documento se ha usado L^AT_EX en su distribución de MiKTeX³ para Windows. Las características más apreciables de MiKTeX son su habilidad de actualizarse por sí mismo descargando nuevas versiones de componentes y paquetes instalados previamente, y su fácil proceso de instalación.

2. **GitHub**

GitHub es una plataforma de desarrollo cooperativo en la que se pueden alojar proyectos utilizando el sistema de control de versiones Git⁴. Se utiliza para la creación y almacenamiento de código fuente de manera pública. La herramienta Git se ha usado para mantener un control de versiones sobre 2 repositorios, uno para la demo y otro para la memoria.

3. **Discord**⁵

Discord es una plataforma de comunicación por voz, texto y vídeo. Discord ofrece la posibilidad de crear canales dedicados, lo que se ha usado para guardar enlaces de interés para el desarrollo prematuro de este proyecto. Esta herramienta es multiplataforma y no requiere de instalación ya que puede usarse desde navegadores como Chrome o Firefox. Además de esto consta con un sistema de transferencia de archivos y de retransmisión de tu pantalla a las personas que se encuentren en la llamada. Todas estas características han sido utilizadas para las sesiones de trabajo grupal.

²<https://www.latex-project.org/>

³<https://miktex.org/>

⁴<https://git-scm.com/>

⁵<https://discord.com/>

4. Unity⁶

Unity ha sido el entorno de desarrollo elegido, la versión de Unity utilizada ha sido la 2018.4.18f1. La elección de Unity fue por la gran comunidad de usuarios y la extensa y detallada documentación con la que cuenta este motor. Además, al usar C# como lenguaje de programación, se ha contado con toda la API y documentación de Microsoft sobre .NET.⁷

5. Visual Studio 2019⁸

La decisión de usar Visual Studio como IDE fue por la integración que tiene con Unity. Gracias a esta integración se ha podido crear y mantener archivos de proyectos de Visual Studio automáticamente. Algo a tener en cuenta es que no se usa el compilador de Visual Studio, sino que se usa directamente el compilador de C# para la compilación de los scripts creados en Unity.

6. Android Studio⁹

Se ha decidido usar este IDE para el desarrollo de la aplicación para Android. Android Studio dispone de una construcción del proyecto basada en Gradle¹⁰, lo que hace que su construcción se haga de manera dinámica y un editor de diseño enriquecido que permite a los usuarios arrastrar y soltar componentes de la interfaz de usuario. Además de incluir las SDK de Android y todo lo que estas incorporan para que el desarrollo en Android sea posible.

⁶<https://unity.com/es>

⁷<https://docs.microsoft.com/es-es/dotnet/>

⁸<https://visualstudio.microsoft.com/es/>

⁹<https://developer.android.com/studio>

¹⁰<https://gradle.org/>

Capítulo 4

Especificación

4.1. Introducción

Con este proyecto lo que se busca es convertir a un dispositivo móvil en un dispositivo de entrada para jugar a videojuegos. Para conseguir esto se necesita que al menos 2 dispositivos se comuniquen entre si, en este caso un ordenador y un teléfono Android. En el ordenador lo que se quiere es ejecutar un videojuego y que este se controle através de una aplicación móvil que se conecte y transfiera datos de pulsaciones y acciones del jugador para que pueda jugar al juego desde su teléfono móvil. Para que esta comunicación sea posible, es necesario un protocolo de comunicación. Con este protocolo de comunicación ambas aplicaciones tendrán un control sobre los datos que envían y reciben para poder actuar en consecuencia.

4.2. Funcionalidad

- Envío de pulsaciones y gestos admisibles en pantallas táctiles.
- Envío de imágenes de manera constante. Streaming de video.
- Vibración y el uso de vibración como feedback.
- ¿Nombrar aquí posible trabajo futuro y ya desarrollarlo en el capítulo que corresponde?

4.3. Protocolo de comunicación

- Diagrama del protocolo con explicación exhaustiva sobre el uso del little o big endian (coger por ejemplo el protocolo HTTP para ver como se realiza una especificación detallada)

- Explicación de cada uno de los mensajes que se envían y reciben. Había pensado poner los clásicos diagramas de cajas para ver la estructura de los bytes que se envían y el orden sobretodo para una posterior lectura si alguien utiliza solamente una de las partes del proyecto.

Capítulo 5

Implementación

5.1. Android

- Explicación muy despacito de qué es un ciclo de vida. Hablar sobre Actividades y los estados que tienen estas Actividades.
- Explicación de niveles de API y permisos. La aplicación está preparada para leer un código QR si o si en el formato IP:Puerto. Todo esto hace que mencione el QR al hablar de Android, sería recomendable hacer una subseccion para hablar del uso del QR, "historia" de este tipo de códigos, etc?
- ¿Veis mejor hacer aquí una subseccion para explicar arquitectura y otra para explicación del uso?
- Subseccion dedicada únicamente a la concurrencia y comunicación entre hilos.
- subseccion para la arquitectura?

5.2. Unity

- Uso de Unity (entidades, ciclo de vida de entidades, lenguaje de scripting usando .NET)
- Usos habituales del QR aquí? Mejor hacerlo en el apartado de Android?
- Uso de las hebras para el tratamiento de los datos que se leen desde la hebra de red que tenemos pendiente de la llegada de datos al socket.
- Uso de una hebra para mandar datos por la red.
- Subseccion para explicar la utilización de esto?

- subseccion para la arquitectura?

Capítulo 6

Pruebas con usuario y resultado

6.1. ¿?

Capítulo 7

Conclusiones

*—¿Qué te parece desto, Sancho? — Dijo Don Quijote —
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.*

*Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes*

*—Buena está — dijo Sancho —; fírmela vuestra merced.
—No es menester firmarla — dijo Don Quijote—,
sino solamente poner mi rúbrica.*

*Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes*

