Herramienta de Planificación Estratégica para Videojuegos MOBA Orientada a Entrenadores de e-Sports

Trabajo de Fin de Grado
Curso 2018–2019

Autores
Lucía Carrión García
Jeremy J. Ruzicka Peinado

Director
Prof. Dr. Federico Peinado Gil
Codirector: Maximiliano Miranda Esteban

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

Herramienta de Planificación Estratégica para Videojuegos MOBA Orientada a Entrenadores de e-Sports

Trabajo de Fin de Grado
Grado en Desarrollo de Videojuegos
Depto. de Ingeniería del Software e Inteligencia Artificial

Autores
Lucía Carrión García
Jeremy J. Ruzicka Peinado

Director
Prof. Dr. Federico Peinado Gil
Codirector: Maximiliano Miranda Esteban

Convocatoria: *Septiembre 2019*
Calificación: –

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

20 de septiembre de 2019

Resumen

En este Trabajo de Fin Grado se investiga el vacío que hay en el área de herramientas dirigidas a entrenadores de e-Sports, en concreto de juegos tipo MOBA (del inglés, Multiplayer Online Battle Arena).

Los e-Sports son competiciones de videojuegos que se han vuelto extremadamente populares, generando altos niveles de empleo y de capital. Uno de los géneros más populares en estas competiciones son los MOBA. Estos son juegos multijugador de estrategia en los que el jugador controla a un sólo personaje, miembro de un equipo que se enfrenta en combate contra otro equipo. Los e-Sports han creado un rol esencial: los entrenadores. Tanto en equipos profesionales como en academias, estos tratan de transmitir sus conocimientos y estrategias a sus jugadores. Consecuentemente, se ha generado un nicho en éste área: el de las herramientas orientadas a los entrenadores.

En los siguientes capítulos se investiga este nicho mediante una entrevista a un entrenador de e-Sports y un análisis de las herramientas ya existentes. Gracias a esto, se descubre la necesidad de la creación de herramientas mejores.

A continuación, se propone, diseña, desarrolla y prueba una herramienta que tiene por objetivo cubrir ese vacío: MOBA Tactician. Esta herramienta sirve para planear, en solitario o en equipo, estrategias de juego para el título *League of Legends*, uno de los más relevantes en el mercado actual, aunque es extensible a otros juegos de este género.

Palabras clave

Academias de e-Sports, entrenamiento en videojuegos, League of Legends, Unity, multiusuario, desarrollo de aplicación

Title

Strategic Planining Toolkit for MOBA Video Games Oriented to e-Sports Trainers

Abstract

In this project, a research is carried out regarding the gap existing in the field of tools oriented to e-Sports coaches, specifically the ones for MOBA (Multiplayer Online Battle Arena) games.

E-Sports are videogame competitions that have become extremely popular which generate high amounts of employment and capital. One of the most popular genres in these competitions is MOBA games. These are multiplayer strategy video games in which the players handle one character who is a member of a team that fights against another team. E-Sports have created an essential role which are coaches. Both in professional teams and academies, these coaches try to transmit their knowledge and tactics to their players. Consequently, this has generated a niche in this field: tools oriented to coaches.

In the next chapters, this niche is investigated through an interview with an e-Sports coach and an analysis of existing tools. Thanks to this, the need for creating better tools is discovered.

Next, we propose, design, develop and test, a tool whose objective is to fill that gap: MOBA Tactician. This tool can be used to plan, alone or in a team, game strategies for the game *League of Legends*, one of the most relevant in the actual market, although it can be expanded to other games of this genre.

Keywords

e-Sports academies, videogame training, League of Legends, Unity, multi-user, app development.

Índice

Capítulo 1

Introducción

El mundo de los e-Sports, también llamados deportes electrónicos, es relativamente nuevo[1]. Cuando hablamos de e-Sports, nos referimos a campeonatos, online o presenciales, de videojuegos en los que compiten jugadores, a menudo profesionales o semi-profesionales. Una de las primeras competiciones de las que se tienen constancia ocurre en 1972, en la universidad de Stanford, donde estudiantes compiten jugando al videojuego *Spacewar!* (Steve Russel, 1962). En él, dos jugadores pilotan sendas naves espaciales y deben destruir al jugador contrario. El primer evento público es ocho años más tarde, en 1980, y recibe bastante atención de los medios, sobre todo la competición de *Space Invaders* (Taito Corporation, 1978), un juego de arcade en el que un único jugador pilota una nave espacial y debe destruir a las naves marcianas enemigas. Los jugadores se retan por conseguir la puntuación más alta. Más adelante, en los años 90 y con la introducción de Internet, los usuarios por fin pueden conectarse y competir desde distintas partes del mundo. Gracias a esto, los videojuegos con opciones multijugador se extienden a gran velocidad. Diferentes compañías, como Nintendo, comienzan a organizar sus propios campeonatos y, en 1997, se organiza el torneo Red Annihilation para el juego *Quake* (id Software, 1996), un videojuego de disparos en primera persona cuya utilización de gráficos en 3D es revolucionaria. Se compite en modo multijugador, conectándose por red de área local o de Internet. Este torneo atrae alrededor de 2000 jugadores y es considerado el primer campeonato moderno de e-Sports. A partir de este momento, se continúan organizando torneos y los e-Sports van ganando cada vez mayor popularidad. En 2018, se organizan alrededor de 3985 competiciones en todo el mundo y se reparten unos 161,8 millones de dólares en

[1] Medium.com (2018). The history and evolution of e-Sports [online] Disponible en: `ttps://medium.com/@BountieGaming/the-history-and-evolution-of-esports-8ab6c1cf3257`

premios[2].

Parece claro que, a pesar de su reciente historia, los e-Sports mueven grandes cantidades de dinero y puestos de trabajo. Es por ello que, con este Trabajo de Fin de Grado, nos proponemos adentrarnos en este mundo, en concreto en el área tecnológica relacionada con las aplicaciones software de interés para este sector. Gracias al proceso de investigación inicial, detectamos un gran vacío en cuanto a herramientas orientadas a entrenadores de e-Sports y, por eso, vemos la necesidad de proporcionarles útiles específicos para su trabajo. Este proyecto tiene por principal motivación reducir en parte ese vacío.

Desde hace un par de años, las academias de e-Sports comienzan a proliferar. En ellas, profesores y entrenadores tratan de transmitir sus conocimientos y estrategias a sus jugadores pero, como hemos dicho antes, esto se hace más difícil si no se cuenta con los medios adecuados. Por ello, planteamos el desarrollo de una aplicación con la que profesores/entrenadores profesionales puedan plantear y explicar estrategias de juegos de tipo MOBA (del inglés, Multiplayer Online Battle Arena) a sus alumnos/jugadores. Los juegos MOBA son juegos multijugador de estrategia en los que el jugador controla a un sólo personaje, miembro de un equipo que se enfrenta en combate contra otro. Normalmente, el objetivo en este tipo de juegos es derrotar al equipo contrario destruyendo su base. Escogemos este género de videojuegos ya que es extremadamente popular en los e-Sports, su estructura y jugabilidad principal es muy parecida en todos ellos, por lo que se pueden abarcar bastantes títulos; y además se nos presenta la oportunidad de seguir las directrices de un cliente real que trabaja con juegos MOBA. Utilizamos como referencia el juego *League of Legends* (Riot Games, 2009), ya que nuestro cliente está muy interesado en disponer de una herramienta que cubra específicamente este título, uno de los más populares en este tipo de competiciones. Sin embargo, nuestra herramienta se concibe como ampliable a cualquier otro juego del mismo género, con el objetivo de cubrir las necesidades de entrenadores especializados en otros títulos.

1.1. Estructura de la memoria

La memoria de este trabajo se compone de los siguientes capítulos:

1. El presente capítulo de introducción donde presentamos nuestra propuesta, su origen y los motivos por los que nos hemos decidido embarcarnos en el desarrollo de este trabajo.

2. Análisis de la situación actual en la que se encuentra el entorno de nuestra herramienta, el estado de la cuestión, así como su competen-

[2]Esportsearnings.com (2018). Overall e-Sports stats for 2018 [online] Disponible en: `https://www.esportsearnings.com/history`

cia en el mercado, y una referencia a la entrevista realizada a José de Matías, director del centro de entrenamiento The Global ESports Academy.

3. Definición de los objetivos principales del trabajo y especificación de los requisitos de la aplicación a desarrollar.

4. Descripción de las diferentes técnicas y recursos que hemos utilizado en el proceso de diseño y desarrollo de todo nuestro proyecto.

5. MOBA Tactician, los detalles del diseño de la aplicación, así como cuestiones relacionadas con la implementación de esta herramienta.

6. Explicación de la evaluación externa realizada por José de Matías sobre MOBA Tactician, la aplicación desarrollada.

7. Conciso resumen en torno al cumplimiento de los objetivos y requisitos propuestos en el tercer capítulo, junto a una reflexión sobre las líneas de trabajo futuro que permitirían continuar ampliando este sistema.

Capítulo 2

Estado de la cuestión

En este capítulo se revisa el estado de la cuestión en materia de videojuegos tipo MOBA y herramientas de planificación estratégica para ellos, estén o no orientadas específicamente a entrenadores profesionales de e-Sports.

2.1. Videojuegos MOBA

Los videojuegos de tipo MOBA (del inglés, Multiplayer Online Battle Arena), también llamados de Estrategia de Acción en Tiempo Real o ARTS (del inglés, Action Real Time Strategy), son un subgénero de los videojuegos de estrategia en el cual dos equipos de varios jugadores se enfrentan por combate directo entre sí.

El formato estándar de los equipos es de cinco contra cinco jugadores, aunque hay otros juegos más recientes que tienen un formato de tres contra tres. Los personajes que los jugadores controlan son, generalmente, únicos en el equipo y tienen unas habilidades y características determinadas. Además, hay diferentes objetos y NPCs (del inglés, Non-Playable Characters, o personajes no jugables) con los que los jugadores pueden interactuar, destruyéndolos para ganar experiencia y subir de nivel, mejorando así sus habilidades y estadísticas. También pueden conseguir recompensas o comprar objetos que potencien sus habilidades.

La estructura de los mapas (véase Figura 2.1) suele ser la misma: un cuadrado con una base por equipo, una en la esquina inferior izquierda y otra en la superior derecha, y tres carriles que las unen. Estas calles se sitúan una por arriba (Top) recorriendo los bordes superiores, otra por en medio en diagonal (Middle) y una última por abajo (Bottom), que recorre los bordes inferiores. En estos carriles se desarrolla la mayor parte de la acción de las partidas. Es aquí donde cada equipo cuenta con unas torres defensivas que hay que destruir antes de llegar y poder atacar la base enemiga.

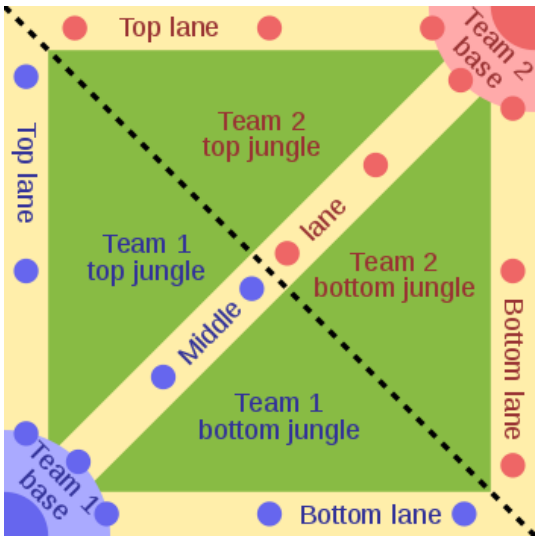

Figura 2.1: Diagrama de la estructura habitual del escenario de un videojuego MOBA, con las zonas de cada equipo, las bases, los carriles que las conectan y las torres, representadas por puntos.

Este formato de juego se populariza gracias al escenario del conocido título de estrategia en tiempo real *StarCraft* (Blizzard Entertainment, 1998), *Aeon of Strife*, que es creado a principios de los 2000 con la herramienta StarCraft Campaign Editor, también llamada StarEdit[1]. Esta permite a los jugadores crear sus propios mapas. *Aeon of Strife* contiene los elementos básicos de los juegos MOBA de hoy en día: dos equipos, tres calles, NPCs, y ciertos elementos del género RPG (del inglés, Role-Playing Games), esto es, personajes de un universo ficticio cuyas habilidades van progresando.

Tras la gran acogida de este género, se publican otros juegos que siguen la misma estructura y la van evolucionando[2], como *Warcraft III: Reign of Chaos* (Blizzard Entertainment, 2002), un RTS multijugador con elementos y ambientación RPG que consiste en desarrollar una ciudad para conseguir recursos con los que crear tropas y derrotar a los oponentes. Destaca su mapa personalizado, *Defense of the Ancient: Allstarts* (IceFrog, 2003). Este enfrenta a un equipo de cuatro jugadores, cada uno controlando a un personaje único y ayudado por NPCs, contra otro equipo controlado por la máquina y, estructuralmente, es como el diagrama anterior (véase Figura 2.1). Creado para una expansión del juego,es el precedente del título *DOTA 2* (Valve Corporation, 2013), un MOBA que sigue la estructura mencionada, tanto visual como de jugabilidad. Otros títulos dignos de mención son *League of Legends*,

[1] ign.com (2012). StarCraft: Campaign Editor. [online] Disponible en: https://www.ign.com/wikis/starcraft-campaign-editor/StarCraft:_Campaign_Editor

[2] liquipedia.net. (2017). DOTA History. [online] Disponible en: https://liquipedia.net/dota2/Dota_History

del que hablamos en el siguiente apartado, o *SMITE* (Titan Forge Games, 2014), un MOBA con la ambientación de diferentes mitologías. Todos estos juegos son extremadamente populares en las competiciones de e-Sports.

2.1.1. League of Legends

League of Legends (Riot Games, 2009) es el título sobre el que hemos decidido trabajar en nuestro proyecto, al tratarse de uno de los juegos MOBA más populares en las competiciones de e-Sports. Cuenta con las principales características de los videojuegos RPG pero, considerando la frecuencia de sus actualizaciones y expansiones, que fuerzan a incorporar nuevos personajes (cuenta con 144 campeones a día de hoy[3]), nos encontramos con una amplia colección de temas que van desde criaturas fantásticas a piratas o robots, entre otros.

En el modo competitivo juegan dos equipos de cinco jugadores cada uno. Cada jugador debe escoger un personaje que será único en esa partida. En este modo siempre se juega en el mapa *Summoner's Rift* (véase la Figura 2.2). Sin embargo, en otros modos de juego, que no se usan para las competiciones de e-Sports, hay más variedad de mapas. *Summoner's Rift* está basado en *Defense of the Ancients: Allstars*, un mapa personalizado de *Warcraft III: Reign of Chaos* que fue muy popular. Este, a su vez, es una evolución del mapa antes mencionado, *Aeon of Strife*.

Summoner's Rift sigue el formato estándar mencionado anteriormente, donde en las tres calles ocurre la mayor parte de la acción. En cada calle hay tres torres por equipo que tratan de bloquear el avance del equipo contrario, disparándoles a ellos y a sus NPCs aliados. El equipo contrario debe destruir todas las torres de un carril para poder llegar hasta la base contraria.

Estas bases son los objetivos a conquistar. En ellas se encuentran sendos nexos, los cuales hay que destruir para ganar la partida. Los nexos están rodeados por tres inhibidores, uno por calle, y para destruir el nexo hay que destruir previamente los inhibidores de aquellas calles que se hayan despejado de torres. Las torres son destruidas permanentemente, pero los inhibidores se regeneran pasado un tiempo.

Las zonas rodeadas por las calles se llaman "junglas" y son simétricas. Si nos referimos al esquema anterior (véase Figura 2.1), las zonas *Top Jungle* son iguales que las *Botton Jungle* del equipo contrario, y viceversa. En ellas podemos encontrar NPCs variados que reaparecen unos minutos después de ser destruidos. Los más destacados son: el Espino Rojo (*Red Brambleback*) y el Centinela Azul (*Blue Sentinel*).

Al destruirlos, aparte de dar experiencia y oro, proporcionan potenciadores de daño y maná (un recurso necesario para poder usar la mayoría de las habilidades de los campeones), respectivamente. En la parte inferior del

[3] 22 de agosto de 2019

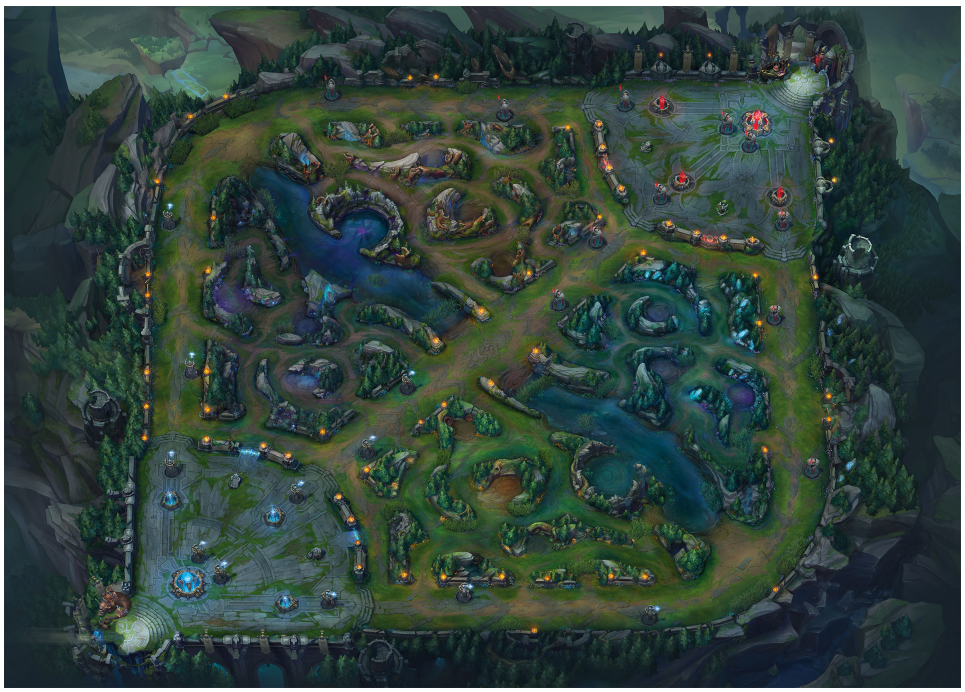

Figura 2.2: Mapa de *League of Legends*, *Summoner's Rift*.

mapa se encuentra el Dragón Elemental (*Elemental Drake*), el cual puede ser de cuatro tipos que se cambian aleatoriamente cada vez que se regenera: agua (otorga una bonificación de regeneración de vida y maná), viento (bonificación de velocidad de movimiento), tierra (bonificación de daño a estructuras) y fuego (bonificación de daño). Estos aparecen hasta el minuto treinta y cinco. A partir de ese momento, aparece el Dragón Anciano (*Elder Dragon*), el cual potencia todos las bonificaciones obtenidas de los otros dragones durante la partida.

Hasta el minuto veinte, en la parte superior del mapa se puede encontrar al Heraldo de la Grieta (*Rift Herald*). Este es el único NPC que no se regenera una vez destruido, y su bonificación puede ser usada una única vez para ayudar a destruir estructuras enemigas. A partir del minuto veinte, aparece el Barón Nashor (*Baron Nashor*). Este potencia las estadísticas de vida, maná y daño del jugador y de los súbditos (*minions*) aliados cercanos.

Los súbditos son los NPCs aliados de cada equipo. Estos se generan en el nexo en oleadas, una por cada calle, y atacan a cualquier estructura, súbdito o campeón del equipo contrario que esté más cerca. Cada oleada tiene tres súbditos Cuerpo a Cuerpo (*Melée*) y tres Magos (*Casters*). Una de cada tres oleadas tiene un súbdito de Asedio (*Siege*). Por último, cuando se destruye un inhibidor, cada oleada de esa calle incorpora un Supersúbdito (*Super-Minion*). Esto será así mientras el inhibidor permanezca destruido, es decir,

sin haberse regenerado.

A nivel competitivo, los cinco jugadores toman posiciones diferentes, estas son: *top-laner*, *mid-laner*, *carry* (antes llamado *AD Carry*), *support* y *jungler*. El *top-laner* suele ir a la calle superior a enfrentarse contra el jugador del equipo contrario que tiene la misma posición. Suelen ser personajes de tipo tanque (capaz de resistir una gran cantidad de daño) o duelistas (con gran capacidad de supervivencia, capaz de hacer y recibir daño considerable, y muy buenos en combates uno contra uno). El *mid-laner* se enfrenta al del equipo contrario en la calle central. Estos suelen tener la capacidad de hacer mucho daño instantáneo con el objetivo de poder acabar con un campeón enemigo rápidamente.

En el carril inferior se enfrentan las dos *bot-lanes* (dos jugadores que se encargan de la calle de abajo) de cada equipo, cada una de ellas formada por un *carry* y un *support*. El *carry* es la principal fuente de daño del equipo y, a diferencia del *mid-laner*, tiene que hacer daño constante durante todo el combate. Suelen ser personajes con gran cantidad de daño pero muy fáciles de matar. El *support* tiene la función de ayudar y proteger al resto de miembros de su equipo, en especial al *carry*, aunque en ocasiones tenga que sacrificarse para ello. Suelen ser personajes que se caracterizan, no por su resistencia ni por su daño, sino por poseer habilidades como curar, dar movilidad a los aliados o reducir la de los enemigos. Por último, los *junglers* son jugadores que no van a ninguna calle específica, sino que se van moviendo por todo el mapa, apareciendo en los diferentes carriles para ayudar a sus compañeros o entorpecer al rival. Los *junglers* obtienen oro y experiencia al matar a las criaturas neutrales de los campamentos de la jungla.

En este juego, al igual que en la mayoría de los MOBA, la visión es muy importe. Es esencial saber cuál es la posición de todos los enemigos con el fin de evitar emboscadas, de que no destruyan estructuras o puedan cumplir objetivos importantes como matar a los dragones o al Barón Nashor. Cada jugador tiene visión sobre su propia posición, la de sus aliados, sus súbditos y sus torres. Esta visión no es suficiente para evitar todo lo comentado anteriormente, pero para ello existen los guardianes (*wards*). Estos son objetos, invisibles para el equipo contrario, que otorgan visión adicional en el lugar donde hayan sido colocados durante un periodo de tiempo. Se suelen posicionar en lugares estratégicos con el fin de tener el máximo control sobre la posición de los enemigos. Esto lleva a la existencia de otros objetos para contrarrestar a los guardianes del equipo contrario y poder luchar por la visión del mapa, como la Lente de Barrido (*Sweeping Lens*), que hace visibles a los guardianes del equipo contrario en una zona concreta. Todos estos elementos de visión hacen de este juego una batalla estratégica en la que hay que tener en cuenta hasta el más mínimo detalle.

Como podemos comprobar, *League of Legends* es un juego de estrategia donde intervienen muchos factores: tiempo, estadísticas y características de

los NPCs, etc. Todo esto lo hace extremadamente complicado de conocer a fondo y convierte las estrategias en algo esencial para poder sobresalir en las competiciones de e-Sports. Esta es una razón más para apoyar la necesidad de disponer de una herramienta que ayude, incluso a los profesionales, a crear dichas estrategias.

2.2. Entrevista con un entrenador de e-Sports

Como experto para asesorar en este proyecto, hemos entrevistado a José Eloy de Matías Alcántara[4]. Actualmente es el Director de The Global ESports Academy[5], una academia de e-Sports que abre sus puertas en Madrid en 2018. José comienza su trayectoria en el ámbito de los e-Sports en 2008, cuando se traslada a Estocolmo a hacer su Trabajo de Fin de Licenciatura como entrenador deportivo del equipo profesional de *Counter Strike* (Valve Corporation, 1999), Virtus.pro. Desde entonces se ha dedicado al ámbito de los e-Sports en calidad de profesor de la Universidad Internacional de Valencia, director del club wSystems y entrenador de varios equipos profesionales, entre otras cosas. También ha fundando varios clubes de e-Sports, estudiado los e-Sports y publicado diversos artículos sobre ellos. Él mismo es un jugador de alto ranking en *League of Legends*, juego en el cual tiene una década de experiencia. Ahora, en la academia, pone en práctica todo el conocimiento adquirido para crear un área de aprendizaje donde jóvenes pueden convertirse en mejores jugadores, educando y promoviendo también valores como el trabajo en equipo o el respeto al rival.

Nos reunimos con él al principio del proyecto con el objetivo de ver cómo se entrena a los alumnos en las academias de e-Sports, ya que nuestra idea en su origen consiste en crear una herramienta dirigida a los alumnos para ayudarles a desarrollar las habilidades óculo-motrices necesarias para sobresalir en el ámbito profesional. Sin embargo, tras esta entrevista, descubrimos un nicho que nos interesa aún más: el de las necesidades de profesores y entrenadores profesionales de e-Sports.

Como nos explica José durante la entrevista (véase Apéndice D), la mayor parte del entrenamiento (el 95 por ciento, según nos dice) se lleva a cabo dentro del juego directamente. Esto puede ser con partidas, ya que cuantas más horas de juego, más experiencia y entendimiento del mismo se genera, o con las herramientas de entrenamiento que la mayoría de los juegos llevan incorporadas. Por ejemplo, en el juego *League of Legends* se tiene la opción de crear una partida personalizada en la que se puede adaptar el escenario para practicar ciertos movimientos claves del juego. Sin embargo, esta preparación del mapa conlleva mucho tiempo, aunque una vez hecho este proceso, nos dice José, es bastante eficiente.

[4] linkedin.com/in/josé-de-matías-86aa2a174
[5] https://theglobalesportsacademy.com/

Si nos centramos en ese 5 por ciento restante del entrenamiento, tenemos herramientas de terceros. Estas herramientas sirven sobre todo para proporcionar retroalimentación (o feedback) a los jugadores y entrenadores. Como nos dice José, el feedback es clave para mejorar en cualquier ámbito de la vida, pero lo es aún más en los deportes, tanto físicos como e-Sports. Aquí entra la figura del entrenador: él es quien analiza la información de las partidas y prepara el feedback. Si bien esta es una tarea que cualquier jugador o atleta puede hacer por su cuenta, requiere de un tiempo esencial que podría ser mejor aprovechado. Utilizando herramientas que muestran las estadísticas de las partidas de cada jugador y analizando sus partidas e incluso las de los rivales, los entrenadores son capaces de proporcionar feedback extremadamente valioso a sus jugadores.

Ante la existencia de estas herramientas, oficiales y de terceros, que ayudan a entrenar, nos preguntamos qué más necesitan los entrenadores para que ellos también consigan sobresalir en su trabajo. Es evidente que el tiempo es oro y, si los jugadores no deberían perder el tiempo en construir ellos mismos ese feedback, los entrenadores también trabajarán mejor cuanto menos tiempo tengan que dedicar a crearlo y más tiempo tengan para trabajar sobre él.

A continuación preguntamos a José qué herramientas utilizan ellos en la academia y si son suficientes. La respuesta a esta pregunta desencadena todo este proyecto. En el caso de *League of Legends*, José especifica la necesidad de dos herramientas diferentes: por un lado, una pizarra interactiva con la que planear estrategias y, por otro, una herramienta de seguimiento de las estadísticas de los jugadores del equipo entero.

Nos comenta que ellos ya utilizan una pizarra (un mapa sobre el que se pueden añadir imágenes de los personajes), pero que es muy básica, está desactualizada y es incompleta. Según nos dice, le faltan varias cosas:

- Dejar un rastro al mover a los personajes.
- Poder añadir a los NPCs "súbditos".
- Que los NPCs de la jungla den información sobre su estado.
- Que se puedan eliminar los NPCs de la jungla.
- La existencia de una barra del tiempo que vaya del minuto de juego 1 al 60 y que, conforme el profesor la vaya moviendo, los elementos que dependen del tiempo, cambien automáticamente (i.e., se actualicen los NPCs o la información de los mismos).
- Poder destruir las torres.
- Que esté actualizada en cuanto a datos del juego.

José insiste en que una herramienta que incluya estas funcionalidades es muy necesaria, y por eso le interesa mucho que su desarrollo se ponga en marcha.

Tras finalizar esta entrevista tenemos un objetivo muy claro sobre qué hacer en este Trabajo de Fin de Grado, y de aquí surge la idea de MOBA Tactician, una herramienta para crear estrategias de juegos MOBA. Es decir, aunque utilicemos como ejemplo al juego *League of Legends*, nuestro objetivo es que la herramienta pueda expandirse de forma sencilla (mediante paquetes) a otros juegos de este género. Así pues, comenzamos a explorar ideas sobre qué tendría que tener una herramienta como tal. Nuestro primer paso es analizar las aplicaciones que ya existen, lo cual exploramos en el siguiente apartado.

2.3. Herramientas para entrenadores de e-Sports

Dado que este área es relativamente nueva, el software que hemos podido encontrar es escaso y está poco desarrollado. Si bien existen algunas herramientas que pueden ayudar a los jugadores a mejorar ciertas habilidades (por ejemplo, las óculo-motrices) o a hacer un seguimiento individual de sus estadísticas en el juego, no existen muchas que faciliten el trabajo de los entrenadores.

Tras realizar la entrevista de la que hablamos en el apartado anterior, comenzamos una recopilación de las herramientas que ya existen y que vamos a considerar nuestra competencia en el mercado. Estas son parecidas a la aplicación que queremos desarrollar: se tratan de pizarras interactivas con contenido específico del juego *League of Legends* en la que los profesores y entrenadores profesionales pueden crear y explicar estrategias a sus alumnos.

2.3.1. RiftKit

RiftKit[6] son una serie de herramientas creadas por el programador independiente Simone Masiero[7] y orientadas a la estrategia del juego *League of Legends*. La primera se trata de un mapa interactivo en el cual se pueden planear estrategias en tiempo real con compañeros, y la segunda es una aplicación de Android con la que se puede seguir la partida de otros jugadores.

Nos centramos en concreto en la primera herramienta, llamada RiftKit LOL Planner (véase Figura 2.3). Esta consiste en una imagen del mapa *Summoner's Rift* y una barra lateral con las herramientas que ofrece la aplicación. Hay cinco personajes en cada equipo: T, J, M, A, S. Estas son las letras que corresponden a los nombres de las diferentes posiciones que pueden ocupar los jugadores en el juego: top-lanner, jungler, mid-lanner, carry (antes lla-

[6] https://riftkit.net/
[7] https://github.com/duiker101

mado AD Carry) y support. Un equipo está caracterizado por el color azul y otro, por el naranja.

Las herramientas del menú lateral permiten (en orden de arriba a abajo):

- Desplazarse por el mapa.
- Mover a los campeones, guardianes y lineas.
- Pintar lineas, que representan las rutas de los jugadores.
- Borrar guardianes y lineas.
- Poner guardianes verdes.
- Poner guardianes rosas.
- Restaurar el mapa borrando todo lo que se había creado y movido.
- Abrir un bloc de notas en el lado derecho.
- Crear una sesión online, abriendo también un menú en el lado derecho que permite realizar ciertas acciones del modo online.

Para activar una opción, se debe seleccionar previamente con el botón principal del ratón. Una vez seleccionada, cualquier acción se usa de la misma forma: pinchando con el botón principal del ratón o manteniendo dicho botón pulsado y arrastrando.

El modo online está implementado con la librería TogetherJS[8]. Esta es una librería de código abierto creada por Mozilla en JavaScript, la cual permite añadir, a una página web, opciones y herramientas de colaboración en línea entre usuarios. Al seleccionar la opción de compartir, crea una sesión online a la que se pueden unir otros usuarios que ven el mapa en tiempo real, pudiendo interactuar con todas las diferentes opciones. Las herramientas de este menú permiten (en orden de arriba a abajo):

- Personalizar el usuario que se está utilizando (su nombre, avatar y color).
- Cerrar la sesión de TogetherJS, volviendo al modo de usuario único.
- Copiar el link de la sesión con el que se da acceso a nuevos usuarios.
- Activar el micrófono para comunicarse con el resto de usuarios.
- Abrir un chat escrito para comunicarse con el resto de usuarios.

[8]Togetherjs.com. (n.d.). TogetherJS. [online] Disponible en: https://togetherjs.com/

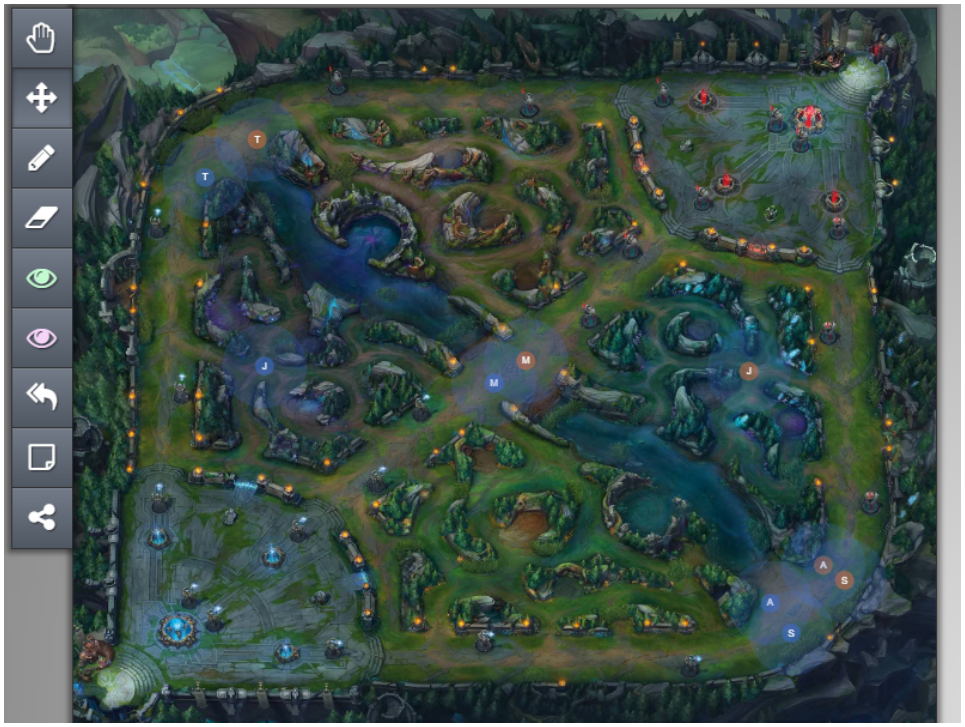

Figura 2.3: Interfaz de la herramienta Riftkit LOL Planner.

Ventajas o puntos positivos que vemos en esta herramienta:

- Posee las funcionalidades básicas de una herramienta de este tipo (hacer zoom, moverse por el mapa, mover elementos, borrar elementos).
- Se pueden pintar rutas.
- Se puede restaurar el mapa al estado inicial.
- Funciona en linea.
- Se pueden abrir canales de comunicación con el equipo.
- Se puede hacer a cada usuario único.
- Se puede ver el cursor del resto de usuarios.

Desventajas o puntos negativos que vemos en esta herramienta:

- No permite escoger diferentes campeones.
- No permite interactuar con el mapa en sí.
- No se puede cambiar el color de las rutas, por lo que acaban siendo confusas.

- Solo permite tener dos tipos de elementos en el mapa: campeones y guardianes.
- Los guardianes verdes han sido eliminados del juego, por lo que la opción de añadirlos está obsoleta.
- El tener que seleccionar una nueva opción cada vez que se quiere usar, hace que sea poco intuitivo y ralentiza el manejo de la aplicación.
- La herramienta del bloc de notas no funciona (no deja escribir).
- Si un usuario se conecta más tarde, no tiene los elementos del mapa que se han modificado o añadido, actualizados.
- No hay forma de evitar que un participante mueva elementos de la pizarra.
- Cuando un usuario se desconecta, sigue apareciendo conectado.
- Tan solo funciona para el juego *League of Legends*.

Riftkit es una herramienta muy parecida a la que, en el momento de la entrevista, se está utilizando en la academia (una herramienta propietaria). Es muy básica, va a quedar obsoleta pronto y además tiene algunas opciones que, en las pruebas realizadas hemos visto, no funcionan correctamente. Tampoco pensamos que sea una herramienta que se pueda utilizar en el contexto de una clase, ya que cualquier alumno puede modificar el mapa en todo momento. Además, hay funcionalidades que no hacen falta, como los canales de comunicación, ya que las clases son presenciales. Sin embargo, tiene una buena base sobre la que comenzar a diseñar nuestra propia herramienta.

2.3.2. Wottactics

Wottactics[9] es una página web desarrollada por Kalith[10] que pretende servir para crear estrategias para el juego *World of Tanks* (Wargaming, 2011), un simulador multijugador masivo en línea de guerras de tanques. Sin embargo, también se puede usar para otros juegos, entre ellos el *Clash of Clans* (Supercell, 2012), un videojuego multijugador masivo en línea de estrategia y construcción de aldeas; *Counter Strike: Global Offensive* (Valve Corporation, 2012), un videojuego de disparos en primera persona de modo multijugador; o el *Rainbow Six - Siege* (Ubisoft, 2015), un videojuego de disparos en primera persona táctico multijugador; por nombrar algunos.

Nos centramos en la versión de *League of Legends*, LOL Tactics[11]. Nada más comenzar, se puede crear una sala nueva (véase Figura 2.4) en la cual,

[9] https://wottactic.eu
[10] https://github.com/karellodewijk
[11] https://wottactic.eu/lol

automáticamente, se carga una foto del mapa *Summoner's Rift*, aunque se puede cambiar al mapa que se prefiera. El mapa incluye a las criaturas de la jungla, aunque no se pueden modificar (quitar o mover).

En el menú de la izquierda se tienen varias opciones:

- Volver a la página principal.
- Bloquear la sesión para que ningún otro usuario pueda modificar nada.
- Copiar el enlace de la sesión.
- Cambiar el idioma de la aplicación.
- Iniciar sesión, aunque no es necesario hacerlo.

También se tiene la lista de usuarios conectados, con la cual los usuarios dueños de la sesión pueden dar o quitar permisos. Los niveles de permisos de usuarios son tres: observador, editor y dueño. Por último se tiene una ventana con tres pestañas:

1. Las capturas de pantalla que se han hecho durante la sesión y que se pueden poner en vez del mapa actual o ver como diapositivas.
2. Un chat para hablar con el resto de usuarios.
3. Una opción de exportación. Exportar los mapas como imágenes se puede hacer de dos formas, una de ellas simplemente incluye un borde donde se podría, por ejemplo, escribir notas. Al exportarlo también se puede guardar una copia de la sesión, esto incluye las capturas de pantalla previamente guardadas y un JSON con los datos del mapa que se está modificando en ese momento. Por último, existe una opción que permite mandar el mapa modificado a una sesión si se tiene su enlace.

En el menú de la derecha tenemos las opciones para modificar el mapa:

- Añadir imágenes de los personajes, objetos, runas, hechizos o NPCs.
- Pintar rutas, lineas, curvas y formas geométricas.
- Añadir texto, con o sin fondo.
- Deshacer o rehacer la última acción.
- Borrar (de nuevo se abre un menú que nos permite borrar todas la líneas o todas las curvas o todo los textos, etcétera).
- Una goma de borrar para borrar aquello que se pinche individualmente.
- Añadir notas.

- Una herramienta de selección de todo lo que entre en un cuadrado que podemos dibujar.
- Una regla.
- Herramientas de cortar, copiar, y pegar.
- Devolver el mapa al estado original.

Al pinchar cada opción, se cambian las opciones de configuración que se ven abajo de la barra de herramientas. Todo es extremadamente personalizable.

Por último, arriba a la derecha podemos ver tres botones (de izquierda a derecha):

- El resto de usuarios solo pueden ver el mapa del dueño de la sesión en vez de poder visualizar otras diapositivas.
- El resto de usuarios solo pueden ver el mapa con el mismo zoom que el dueño de la sesión.
- El resto de usuarios no pueden arrastrar los objetos del mapa.

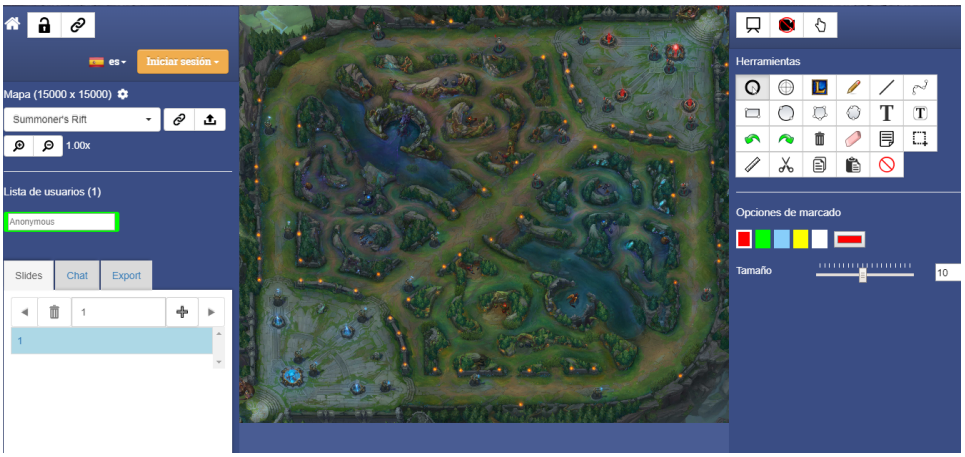

Figura 2.4: Interfaz de la herramienta Wottactics para el juego *League of Legends*.

Ventajas o puntos positivos que vemos en esta herramienta:

- Posee las funcionalidades básicas de una herramienta de este tipo (hacer zoom, moverse por el mapa, mover elementos, borrar elementos).
- Se pueden pintar rutas.
- El mapa incluye a las criaturas de la jungla.

- Permite escoger diferentes campeones.
- Al hacer zoom, las imágenes se escalan.
- Permite escoger diferentes mapas.
- Funciona en línea.
- Se puede seleccionar que el resto de usuarios vean el mapa con el mismo zoom que el dueño de la sala o que vean la misma imagen que él.
- Se puede seleccionar qué tipo de permiso tiene cada usuario.
- Permite guardar capturas de pantalla del estado de la pizarra.

Desventajas o puntos negativos que vemos en esta herramienta:

- Más allá de poner imágenes sobre el mapa, no es interactuable.
- La calidad de los otros dos mapas (*The Twisted Treeline* y *Howling Abyss*) es muy mala.
- Los campeones no están actualizados (algunas imágenes están obsoletas, otras faltan).
- Algunas criaturas de la jungla están desactualizadas (en el juego ya no tienen esa imagen).
- Se pueden realizar acciones fuera del mapa (hacer zoom, dibujar, mover personajes).
- Tarda en actualizar el estado del mapa al resto de usuarios.
- El tener que seleccionar una nueva opción cada vez que se quiere usar, hace que sea poco intuitivo y ralentiza el manejo de la aplicación.

Aunque la primera impresión es que esta es una herramienta bastante completa, tras unos minutos analizándola nos damos cuenta de que la cantidad de elementos y características no es sinónimo de calidad de la herramienta. Al usuario se le bombardea con opciones que son redundantes o simplemente innecesarias. Por ejemplo, la opción de seleccionar cualquier mapa de *League of Legends*, nos resulta interesante. Sin embargo, en el ámbito competitivo esta opción no es útil, ya que solo se usa el mapa *Summoner's Rift*, como ya hemos mencionado. También consideramos que tener siete opciones diferentes de dibujado es superfluo. Además, creemos que tener que dibujar las rutas de los personajes a mano es una pérdida de tiempo.

También nos sirve para darnos cuenta de que, para que una aplicación como esta sea útil durante mucho tiempo, necesita constante mantenimiento.

Es necesario arreglar los errores que hemos encontrado y mantener actualizados los mapas y los objetos y personajes del juego. No se conseguirá la misma calidad en las sesiones de entrenamiento utilizando una herramienta que no se ha actualizado desde hace un año, a una herramienta que tenga revisiones frecuentes.

2.3.3. Comparación de la competencia

Primero, si nos referimos a las interfaces de las dos aplicaciones analizadas previamente, nos resulta más atractiva Riftkit, por ser más limpia y minimalista. Nos parece que cargar la interfaz con barras de herramientas y botones, distrae del verdadero objetivo de la aplicación. Por esta razón, nos proponemos hacer un diseño similar, usando solo las funcionalidades y menús verdaderamente necesarios.

Segundo, respecto a los controles, ambas aplicaciones siguen el mismo proceso: antes de usar una opción, hay que seleccionarla. Consideramos que esto hace al manejo de la aplicación más lento de lo que podría ser. Por esto, tenemos en mente prescindir de la barra de herramientas, reiterando el punto anterior de crear una interfaz minimalista, y hacer que casi todo el manejo se base en acciones directas con el ratón, es decir, que las funcionalidades ya estén incorporadas en los diferentes botones del ratón en vez de tener que seleccionarlas de un menú fijo (i.e., para arrastrar se usa el botón principal y para añadir imágenes se usa el botón secundario, que activa un menú).

A continuación, analizamos con detalle las funcionalidades y ventajas que hemos extraído de las aplicaciones (véase Figura 2.5). Si observamos la tabla, notamos que faltan algunas ventajas que hemos mencionado en los dos apartados anteriores. Esto se debe a que, aunque son características que sí hacen atractiva a la aplicación, en nuestro caso no nos interesan. Recordamos que la herramienta que tenemos por objetivo desarrollar va a ser usada por entrenadores para crear estrategias y compartirlas con los jugadores. Este proceso de creación puede ser previo a las sesiones de entrenamiento, pero la otra posibilidad es la utilización de la herramienta en las propias sesiones, que serían presenciales. Por esta razón, nuestra aplicación no requiere tener canales de comunicación, ya sean escritos o con la posibilidad de usar micrófono. Además, tampoco se necesita la funcionalidad de ver dónde están los cursores de todos los usuarios, ya que no queremos recargar la pantalla con información y elementos innecesarios. Por último, no nos parece necesario dar la opción de poner otro mapa que no sea *Summoner's Rift* en el caso de *League of Legends*, como se puede hacer en la aplicación de Wottactics, ya que en las competiciones de e-Sports no se utilizan.

Sin importar cuándo se utilice la aplicación (previo a las sesiones o en las mismas), guardar las estrategias nos parece algo fundamental, ya que los entrenadores y jugadores las deben tener a disposición para futuras referencias.

Reiniciar el mapa es importante para una utilización de la herramienta lo más eficaz posible. Si nuestro objetivo es ahorrar tiempo a los entrenadores y facilitarles su trabajo, de nada sirve que tengan que borrar los objetos creados uno a uno.

Características	RiftKit	Wottactic
Funcionalidades básicas de una herramienta de este tipo	✓	✓
Funciona en línea	✓	✓
Guarda capturas del mapa	✗	✓
Resetea el mapa	✓	✓
Sistema para administrar el control sobre los usuarios	✗	✓
Permite escoger diferentes campeones	✗	✓
Permite añadir diferentes NPCs	✗	✓
Da información de las estadísticas de los NPCs	✗	✗
Mapa interactivo	✗	✗
Al hacer zoom, las pegatinas se escalan	✗	✓
Permite cambiar el color de las rutas	✗	✓
Los elementos del mapa dejan rastro	✗	✗
Contine las últimas actualizaciones del juego (i.e., nuevos campeones)	✗	✗
Funciona para varios juegos	✗	✓
Controles cómodos e intuitivos	✗	✗
Limitación de posición de la cámara y otros elementos funciona	✓	✗
El mapa y los NPCs se actualizan según el minuto de la partida	✗	✗

Figura 2.5: Tabla comparativa de las herramientas analizadas.

Teniendo en cuenta que, como nos comenta José de Matías en la entrevista, los alumnos de las academias son sobretodo adolescentes, para mantener el orden de las clases necesitamos una forma de dar y quitar el permiso de edición del mapa a los alumnos. Este permiso debe ser individual, por si se da el caso de que el profesor quiera hacer una actividad más interactiva con el mapa y quiera preguntar a un alumno en concreto cada vez.

En el caso de las imágenes utilizadas en los juegos, ya sean los campeones o los NPCs, debemos dar una selección que no se quede corta, como en el caso de Riftkit, ni que abrume al usuario, como es el caso de Wottactics. Debemos incluir solo los necesarios, siempre actualizados. Además, José nos pide que la aplicación tenga la opción de quitar NPCs de la jungla y torres (estos personajes y objetos no se mueven de su sitio, simplemente están, no están o están en ruinas). A esta funcionalidad es a la que nos referimos como "mapa interactivo", ya que estamos modificando el mapa en sí, no solo poniendo imágenes sobre él. Además, tenemos el requisito de José de que estos NPCs den información sobre su estado. En el caso de *League of Legends*, esta información equivale a las estadísticas de vida, por ejemplo.

Sobre el zoom y su correcto funcionamiento, nos parece importante que, al acercar la visión al mapa, las imágenes se escalen, es decir, se hagan más pequeñas para que no ocupen todo el carril y se pueda tener una mayor precisión de análisis de esa zona. Además, hemos encontrado muy molesto que en la aplicación de Wottactics se puedan colocar imágenes fuera del mapa, o que se pueda hacer zoom en esa zona.

Tal como explicamos en el apartado en el que analizamos la entrevista con José de Matías, es importante la creación de rutas y, si bien en estas aplicaciones se pueden pintar a mano, creemos que es más conveniente que se generen automáticamente cuando se mueva a un personaje. Además, para el entendimiento de las estrategias es esencial que se pueda diferenciar cada ruta, por eso creemos necesaria la opción de cambiar cada ruta a un color único.

Por último, nos ha parecido muy interesante que la aplicación de Wottactics pueda extenderse a otros juegos. Consideramos que una aplicación como esta debe poder usarse con otros juegos del mismo género, simplemente cambiando sus imágenes y menús personalizados. Así, este se convierte en uno de los objetivos principales de este trabajo.

Nuestra intención es incorporar todas estas funcionalidades que consideramos ventajosas a nuestra aplicación, haciendo así que supere la calidad de nuestra competencia y que su propuesta de valor tenga un lugar en el mercado.

Capítulo 3

Objetivos y especificación

En este capítulo se presentan los objetivos del trabajo y se especifican los requisitos del sistema que vamos a desarrollar: la herramienta de planificación estratégica que consideramos más adecuada para los entrenadores de e-Sports que quieren trabajar con juegos de tipo MOBA.

3.1. Objetivos

Durante el proceso de investigación nos damos cuenta de ciertos puntos que en absoluto pueden ser ignorados si queremos que, por un lado, nuestra herramienta sea más atractiva que las que ya existen en el mercado y, por otro, sea útil en el contexto en el que pretendemos que se use.

En primer lugar, *League of Legends* no es el único MOBA popular en las competiciones de e-Sports, por lo que idealmente la herramienta debe ser extensible para soportar otros juegos.

En segundo lugar, si nuestra herramienta pretende ser usada con más de un usuario, es necesario que todos puedan conectarse a la misma sesión. Como todos estos usuarios están en la misma habitación durante las sesiones de entrenamiento, decidimos que podemos hacer esa conexión por red de área local, utilizando la conexión directa entre dos máquinas para que una sirva de servidor y cliente (esta es del profesor/entrenador) y otra sólo de cliente (del alumno/jugador), sin utilizar los servicios web de un servidor externo.

Por último, en el Capítulo 2 creamos una tabla con las especificaciones que consideramos más ventajosas de las aplicaciones que nos hacen competencia. Consideramos que, para poder crear una aplicación que supere la calidad de estas, tenemos que integrar la mayoría de esas funcionalidades en nuestra herramienta. Debemos recordar, además, los requisitos obtenidos durante la entrevista que se analiza en ese mismo capítulo.

Así, tenemos los siguientes objetivos principales:

- Identificar las necesidades de los entrenadores de e-Sports.
- Crear una herramienta, o un prototipo de ella, que cubra esas necesidades.
- Que dicha herramienta sea colaborativa, genérica y extensible a cualquier otro juego MOBA.
- Comprobar que el prototipo creado es interesante para la comunidad y determinar cómo hay que mejorarlo para que llegue a usarse a nivel profesional.

3.2. Especificación de requisitos

A continuación desglosamos todos los requisitos que nuestra aplicación debe tener. Todos estos los recopilamos durante el proceso de investigación y de análisis de las herramientas existentes en el mercado.

3.2.1. Interfaz

Como hemos comentado anteriormente, nuestra intención es diseñar una interfaz lo más limpia y minimalista posible, con el objetivo de que los alumnos tengan las menos distracciones posibles. Sin embargo, hay elementos indispensables que deben ser visibles en la interfaz, como, por ejemplo, la barra del tiempo que nos pide José en la entrevista.

- El fondo tiene el mapa de juego que se usa en las competiciones, en el caso de *League of Legends*, *Summoner's Rift*.
- Hay una barra en la parte superior de la pantalla que representa el tiempo de la partida en minutos.
- Hay una barra a la derecha que muestra los usuarios que están conectados.
- Hay un botón que permite al usuario desconectarse de la sesión.
- Hay un botón que permite al usuario hacer capturas de pantalla.
- Hay un botón que permite al profesor resetear el mapa al estado inicial.

3.2.2. Gestión de la pizarra

Llamamos "stickers" a las imágenes de los personajes, objetos y NPCs que podemos colocar en los mapas de las aplicaciones analizadas anteriormente. Nuestra aplicación también debe contener estas para representar a los personajes de cada jugador y a los NPCs que pueden moverse por el mapa. Sin embargo, dado que también queremos que se pueda interactuar con el mapa en sí, debemos tener otro tipo de stickers que no se muevan, sino que puedan cambiar su imagen para poner otra en el mismo lugar. Por ejemplo, en el caso de *League of Legends*, las torres pueden estar intactas o destruidas. Podemos solucionar esto poniendo una sticker estática que represente la torre intacta, pero cuando se haga clic sobre ella, se cambie su imagen a la torre destruida. Así pues, tenemos dos tipos de stickers, con diferentes características y funcionalidades.

- Existen dos tipos de stickers: estáticas (no se pueden mover) y dinámicas (se pueden mover).
- Se pueden crear stickers dinámicas.
- Se pueden borrar stickers dinámicas.
- No se pueden crear o mover stickers fuera del mapa.
- Las stickers dinámicas se pueden diferenciar por equipo con un marco de diferente color: azul o rojo.
- Al mover las stickers dinámicas, estas dejan un rastro sobre el mapa.
- Se puede cambiar el color de dicho rastro.
- Se puede borrar el rastro.
- Se puede esconder el rastro.
- Las stickers estáticas se leen por fichero y se crean al principio de la sesión para representar objetos y NPCs del mapa que no se mueven.
- Al clicar sobre las stickers estáticas se cambia su imagen a su siguiente estado (i.e., las torres pasan de estar intactas a estar en ruinas).
- Al poner el ratón sobre las stickers estáticas, se muestra información sobre ellas, si tienen información que mostrar.
- Al mover la barra del tiempo, las stickers estáticas actualizan su estado o información, si así lo necesitan.
- En el caso de *League of Legends*, las imágenes tienen que incluir todos los personajes y NPCs existentes hasta la última actualización del juego.

3.2.3. Menús

Hay dos tipos de menús: el principal y los menús contextuales.

El menú principal es lo primero que se encuentra el usuario al entrar en la aplicación y le permite unirse a una sesión o crear una nueva. También puede escoger de qué juego será la sesión de entrenamiento entre los diferentes paquetes que hay en la aplicación.

- El menú principal tiene dos botones (unirse a una sesión o crear una nueva), un campo donde se puede escribir a qué sesión se une el usuario (a qué puerto), y un menú desplegable con los diferentes paquetes integrados en la aplicación (por defecto, *League of Legends*).

Los menús contextuales son los que permiten añadir y modificar stickers dentro de la sesión. Estos deben ser personalizados para adaptarse a cada juego, ya que no en todos los juegos se añaden los mismos NPCs o campeones, o puede que no interese dar ciertas opciones.

- Las opciones de los menús contextuales se leen dinámicamente por un fichero.
- Las opciones tienen tipos comunes (i.e., crear una sticker, eliminar una sticker).
- Los menús pueden tener submenús con más opciones.
- Puede haber menús de búsqueda que enseñan las diferentes opciones en forma de imágenes pequeñas, de 10 en cada fila.
- Los menús pueden dar la opción de elegir el equipo del que es la sticker creada.
- Los menús pueden tener la opción de generar varias stickers a la vez.

3.2.4. Controles

Uno de nuestros objetivos principales a la hora de diseñar esta aplicación es que el usuario no necesite de barras de herramientas siempre presentes, sino que las funcionalidades estén integradas en los botones del ratón y el teclado. Con esto en mente, el diseño es como sigue:

- **Click con el botón secundario sobre el mapa:** Aparece un menú que permite añadir objetos y personajes.
- **Scroll sobre el mapa:** Se hace zoom. No se puede hacer zoom fuera del mapa.

- **Scroll sobre un menú:** Si el menú lo permite, se hace scroll por sus opciones.
- **Click con el botón principal sobre una sticker estática:** La hace desaparecer o cambia su imagen principal a la secundaria y viceversa.
- **Click con el botón secundario sobre una sticker dinámica:** Se abre el menú especifico de esa sticker.
- **Manteniendo el click con el botón principal sobre una sticker dinámica:** El usuario puede arrastrar esa sticker.
- **Flechas del teclado:** Sirven para mover la cámara por el mapa.

3.2.5. Conectividad

Debemos tener en cuenta que nuestra aplicación está pensada para ser utilizada junto con el equipo, por lo que favorecemos conectar a los usuarios por red. Por esta razón también desechamos la idea de tener una opción de chat online, ya sea escrito o por micrófono.

- Los alumnos se pueden conectar y desconectar cuando deseen.
- Al conectarse un nuevo usuario, este tiene el mapa actualizado al estado actual de la sesión.
- El profesor puede conceder y quitar permiso a los alumnos para interactuar con el mapa usando la barra donde aparece la lista de usuarios.

3.2.6. Carga dinámica de recursos

Notar que nuestro deseo es poder hacer una aplicación en la que se puedan cargar diferentes paquetes, siendo cada paquete de un juego diferente. Así pues, el código de la aplicación no puede tener nada específico de un solo juego, sino que esas características específicas de cada paquete se deben cargar de alguna forma.

- Se carga un fichero con información específica del juego (i.e., eventos que ocurren según el tiempo).
- Se carga un fichero con la información de las estadísticas y posiciones de las stickers estáticas.
- Se carga un fichero con los menús específicos para el juego.

Capítulo 4

Metodología y herramientas

En este capítulo se expone la metodología que va a utilizarse en este proyecto, tanto para el desarrollo software como para el diseño orientado al usuario del producto. Además se explican las decisiones tomadas en cuanto al uso de herramientas para facilitar nuestro trabajo.

4.1. Metodología de gestión del proyecto

Para este proyecto, decidimos hacer uso de una metodología ágil y scrum (Jeff Sutherland & Ken Schwaber, 1995), todo esto aprendido en la asignatura Metodologías Ágiles de Producción (MAP) y puesto en práctica en las diferentes asignaturas de Proyecto (P1, P2, P3).

Una metodología ágil es un proceso basado en el desarrollo iterativo e incremental. En ella, la planificación del proyecto, aunque necesaria, es más flexible. También anima a que se esté en contacto con el cliente durante todo el proceso. El cliente es el que define milestones con fechas y características. Además, la documentación del diseño del producto se puede alterar si los requisitos cambian o si nos encontramos con problemas durante el proceso.

La metodología que nosotros adoptamos se caracteriza por la realización de sprints de dos semanas y milestones cada mes y medio donde entregamos una release de lo que tengamos en ese momento. Además, tras cada sprint realizamos una reunión con nuestro director de nuestro Trabajo de Fin de Grado, donde analizamos lo que hemos hecho durante ese sprint y fijamos metas para el siguiente. Nos decantamos por esta metodología debido a la flexibilidad que nos aporta en la toma de decisiones debido a la posibilidad de hacer cambios en la planificación, estructuración e implementación del proyecto, que sabemos que son dados a cambiar, además de que nos fuerza a estar en todo momento produciendo.

El marco de trabajo (o framework) que hemos adoptado está basado en el scrum: partiendo de la lista de requisitos, cada uno de ellos lo dividimos en tareas más pequeñas que dejamos en un backlog. En cada sprint, cogemos tareas de ese backlog que debemos realizar durante ese periodo de dos semanas. Cuando están terminadas, las probamos y, si las damos por aprobadas, pasamos a la siguiente tarea. El objetivo es terminar en cada sprint todas las tareas que hemos puesto en ese sprint backlog.

4.2. Metodología de diseño del proyecto

Decidimos desde un principio realizar un diseño centrado en el usuario, como el explicado en el libro "About face" (Cooper et al., 2007), para ello utilizamos un metodología de diseño guiado por objetivos. Esto significa que diseñamos nuestro producto a partir de los objetivos que el usuario persigue a la hora de utilizar nuestra aplicación. El proceso de esta metodología de diseño se divide en cinco fases:

- La primera es una fase de investigación en la que realizamos un estudio para obtener datos sobre los usuarios potenciales del producto y cuáles son sus necesidades. Además, llevamos a cabo otro en relación con la herramienta en sí: si existen herramientas similares, el análisis de la competencia, recopilación de literatura y otras referencias que nos puedan ayudar en el futuro.

  Esta fase del proyecto se puede ver reflejada en el Capítulo 2 de esta memoria, en el cual analizamos nuestra entrevista con un entrenador de e-Sports, además de analizar con detalle las dos herramientas, Riftkit y Wottatic, que consideramos nuestra competencia en el mercado.

- La segunda fase es de modelado, en la que se describe, con los datos recopilados en la fase anterior, a un usuario arquetipo de nuestra herramienta. Este modelo lo utilizamos en fases posteriores para proporcionarnos feedback y dar coherencia al diseño de la aplicación.

  Nuestro usuario arquetipo, el cual hemos tenido en mente durante el resto del proceso para tomar cualquier decisión, es un entrenador o un profesor de e-Sports. En concreto, un entrenador de *League of Legends*. Creemos que el contexto en el cual se va a utilizar nuestra herramienta es, idealmente, durante una clase o una sesión de entrenamiento presencial, donde los alumnos o jugadores están a disposición del entrenador. Existe el caso de que alguno de ellos se uniera más tarde a la clase, y esto lo hemos tenido también en cuenta durante la siguiente fase. Como observamos durante nuestra entrevista con el director de

la academia de e-Sports, The Global ESports Academy, estas sesiones pretenden ser interactivas, dato también importante a la hora de diseñar la aplicación.

- La tercera fase es la definición de requisitos en la que cogemos todos lo datos anteriores (las necesidades del usuario arquetipo, del cliente, el análisis de la competencia, el contexto del uso de la aplicación) e identificamos los requisitos de nuestra herramienta en función de todo lo anterior.

  El resultado de esta fase se puede ver reflejado en el Capítulo 3. Ahí también apuntamos unos objetivos principales que queremos cumplir con este trabajo. Después pasamos a analizar, a partir de todo lo aprendido hasta ahora en el proceso de diseño, los requisitos que nuestra aplicación necesita para sobresalir en el mercado al que nos dirigimos.

- En la cuarta fase definimos el framework de diseño, es decir, el concepto general del sistema. Definimos, por una parte, cuál es el comportamiento de la aplicación (framework de interacción) y, por otro, su aspecto (framework visual).

  Durante esta fase hemos conseguido, a su vez, identificar otros requisitos relacionados con cómo el usuario va a interactuar con la aplicación: qué acciones puede realizar y cómo puede realizarlas. Durante esta fase nos proponemos que esta interacción se lleve a cabo únicamente con el ratón y el teclado, prescindiendo de las barras de herramientas con dos objetivos en mente: simplificar el manejo de la aplicación y, en un futuro, adaptarla para que pueda ser utilizada en una pizarra digital con un lápiz táctil.

- La última fase del diseño es el refinamiento, esta es una fase iterativa en la que se analiza todo lo decidido hasta este momento: evaluamos los frameworks, comprobamos que se cumplen las necesidades de los usuarios y del cliente, y cambiamos las partes del diseño que crean problemas.

  Aunque sabemos que, una vez entrados en la fase de desarrollo, el diseño puede cambiar —de hecho tiende a hacerlo al aparecer problemas con los que no se contaban antes—, hemos realizado esta última fase para, de alguna forma, cerrar el ciclo de diseño con unos requisitos y especificaciones robustos.

Tras esta última fase, pasamos al desarrollo de la aplicación, el cual detallamos en el Capítulo 5.

4.3. Herramientas

Para el diseño y producción de nuestra aplicación hemos utilizado y considerado diferentes herramientas y librerías que explicaremos a continuación.

4.3.1. Unreal Engine

Unreal Engine (Epic Games, 1998) es un motor de videojuegos multiplataformas creado por Epic Games. Su primera versión se utiliza en 1998 en el juego *Unreal*, un shooter en primera persona. Sin embargo, la versión considerada para este trabajo es la más reciente, Unreal Engine 4 (UE4), que se lanza para el público de forma gratuita en 2015.

En el comienzo de este trabajo, al barajar los motores con lo que podemos llevar a cabo la implementación de la aplicación, Unreal es nuestro preferido. Si bien nuestro concepto inicial del trabajo es bastante diferente a la idea por la que finalmente nos decantamos, en ese momento Unreal nos proporciona un motor potente con el que trabajar en 3D con unos gráficos excepcionales. Sin embargo, una vez el concepto del proyecto cambia, Unreal se nos queda grande por dos motivos: primero, es demasiado potente para alguno de los equipos que vamos a utilizar para nuestro proyecto, lo que dificultaría el trabajo, y, después, no estamos familiarizados con esta herramienta ya que no la hemos utilizado durante nuestra carrera universitaria.

Aunque nuestra primera noción es aprovechar este trabajo para también aprender a usar esta herramienta, desechamos la idea ya que creemos que el proceso de diseño y desarrollo va a ser lo suficientemente largo y complicado como para, además, realizarlo con una herramienta cuya curva de aprendizaje es bastante pronunciada. Por esta razón esencialmente, decidimos decantarnos por otro motor con el que estamos habituados: Unity.

4.3.2. Unity

Unity (Unity Technologies, 2005) es un motor de videojuegos multiplataforma creado por Unity Technologies en 2005, pero la versión que se utiliza en este trabajo es de 2018.

Hemos decidido utilizar Unity ya que es una herramienta que hemos usado previamente y estamos familiarizados con ella. Además, por sus características, se adapta al tipo de entorno de desarrollo que estamos buscando para este proyecto.

Lo más notable es que es la primera vez que cualquiera de los participantes trabaja con el servicio multijugador de Unity, por lo que nuestro trabajo también se ha visto influenciado por este factor. Consideramos que la curva de aprendizaje ha sido bastante notable, sin embargo, una vez superados unos inconvenientes iniciales, conseguimos aprender dónde están nuestros problemas y cómo buscar la información más precisa para solucionarlos. Aunque

el dominio de este servicio solo se ha alcanzado al final del desarrollo, como explicamos en el Capítulo 5.

4.3.3. JSON

JSON (JSON.org, 2001) (del inglés, JavaScript Object Notation) es un formato de texto sencillo para el intercambio de datos, utiliza texto que es legible por el ser humano y parseable y fácil de generar para la máquina. Con él se pueden crear y configurar objetos, que consisten en pares de atributos-valor, y listas ordenadas de valores, o arrays, (o cualquier otro valor serializable). Se populariza en el año 2001, sin embargo, se ha comprobado que ya existe en 1996.

Hemos decidido realizar la carga de recursos por JSON ya que estamos familiarizados con el uso de este junto con Unity, habiéndolo utilizado previamente durante la asignatura de Videojuegos Para Dispositivos Móviles.

4.3.4. Context Menu

Librería[1] implementada en el lenguaje C# para Unity y creada por el usuario de GitHub mogoson.

Teniendo en cuenta que es nuestra intención tener un sistema de menús complejos y que Unity no tiene una herramienta por defecto que ayude a crearlos con facilidad, recurrimos a esta librería. Nos parece que crea una base suficientemente fuerte para nuestro sistema, por lo que la incluimos en nuestro proyecto, aunque la hemos modificado para adaptarla completamente a nuestras necesidades.

La librería viene con un ejemplo en el que nos hemos basado para comenzar el sistema. Hay dos scripts que deben añadirse a diferentes objetos: *ContextMenuAgentExample* se añade a los objetos que van a activar los menús al clicar sobre ellos, y *ContextMenuUI* se añade al objeto que contiene los botones del menú. El nombre de este último objeto es el nombre del menú que *ContextMenuAgentExample* debe tener como referencia. Esta referencia es pública y se puede poner desde el editor, aunque en ocasiones nosotros lo hacemos dinámicamente según las especificaciones de un JSON.

Los scripts que hemos tenido que modificar para adaptar la librería a nuestras necesidad, por ejemplo, para tener submenús, son *TypesOfMenu*, *ContextMenuAgentExample* y *ContextMenuTrigger*. Hablamos más de estos scripts en el Capítulo 5.

[1]Librería Context Menu. (2018). Librería Context Menu. [online] Disponible en: https://github.com/mogoson/ContextMenu

4.3.5. HSV Color Picker

Librería[2] implementada en el lenguaje C# para Unity y creada por el usuario de GitHub judah4.

Ante la necesidad de tener una interfaz con la que el usuario pueda cambiar el color de las rutas dibujadas en la pantalla de nuestra aplicación, y teniendo en cuenta que Unity no tiene un componente por defecto que cumpla nuestras necesidades, hemos encontrado esta librería que nos permite cambiar el color de objetos.

Primero se debe acoplar el Color Picker, es decir, la interfaz que muestra los diferentes colores a escoger, a nuestro canvas. En nuestro caso, hemos creado un prefab que además tiene un botón que permite eliminar el Color Picker de nuestro canvas, para no visualizarlo constantemente como es en el caso del ejemplo encontrado en GitHub.

A continuación, en el prefab de los stickers que podemos generar en pantalla (los personajes y el resto de objetos), añadimos el componente Color Picker Tester, el cual está desactivado hasta que se selecciona la opción de cambiar de color.

Como lo que nos interesa es cambiar el color del componente Trail Renderer de los stickers antes mencionados, hemos modificado el script *Color Picker Tester*, que es el que añade un listener[3] a la sticker para que cambie el color de ese componente al color elegido con la librería.

[2]Librería HSV Color Picker Unity. (2018). Librería HSV. [online] Disponible en: https://github.com/judah4/HSV-Color-Picker-Unity

[3]El patrón Observer es un concepto común de la arquitectura de aplicaciones. En él, un objeto se "subscribe" a un evento y es notificado cuando ese evento ocurre, reaccionando de alguna manera. En nuestro caso, la sticker se subscribe al evento del cambio de color para actualizar su estado.

Capítulo 5

MOBA Tactician: Herramienta de planificación estratégica

En este capítulo describimos en detalle el sistema desarrollado: una herramienta de planificación estratégica para videojuegos MOBA pensada para ser usada por entrenadores de e-Sports, que hemos llamado MOBA Tactician.

5.1. Diseño

Una vez recopilados los requisitos y el framework de diseño, hacemos unos borradores de cómo debe ser la interfaz de la aplicación. Este diseño lo hacemos según los puntos que hemos descrito en el Capítulo 3 y para el juego *League of Legends*.

Recordamos que, según estas especificaciones, la interfaz debe tener:

- Una barra en la parte superior de la pantalla que simbolice el tiempo.
- Una barra a la derecha que muestre los usuarios que hay conectados.
- Un botón que permita al usuario desconectarse de la sesión.
- Un botón que permita al usuario hacer capturas de pantalla.
- Un botón que permita al profesor resetear el mapa al estado inicial.

Como podemos ver en el borrador del diseño (véase Figura 5.1), la interfaz creada es la del profesor. Los alumnos tienen la misma pero sin el panel de control de permisos ni el botón de resetear, con el objetivo de que solo el profesor pueda realizar esas acciones. Este panel de control de permisos está situado a la derecha, donde se tiene la lista de usuarios conectados, y tiene

una flecha que planeamos que sirva para esconder la barra. Los usuarios que tienen permiso de edición del mapa están caracterizados por un marco verde y los que no, por uno rojo. Por último, la barra del tiempo tiene un campo donde se puede escribir directamente el minuto que se quiere.

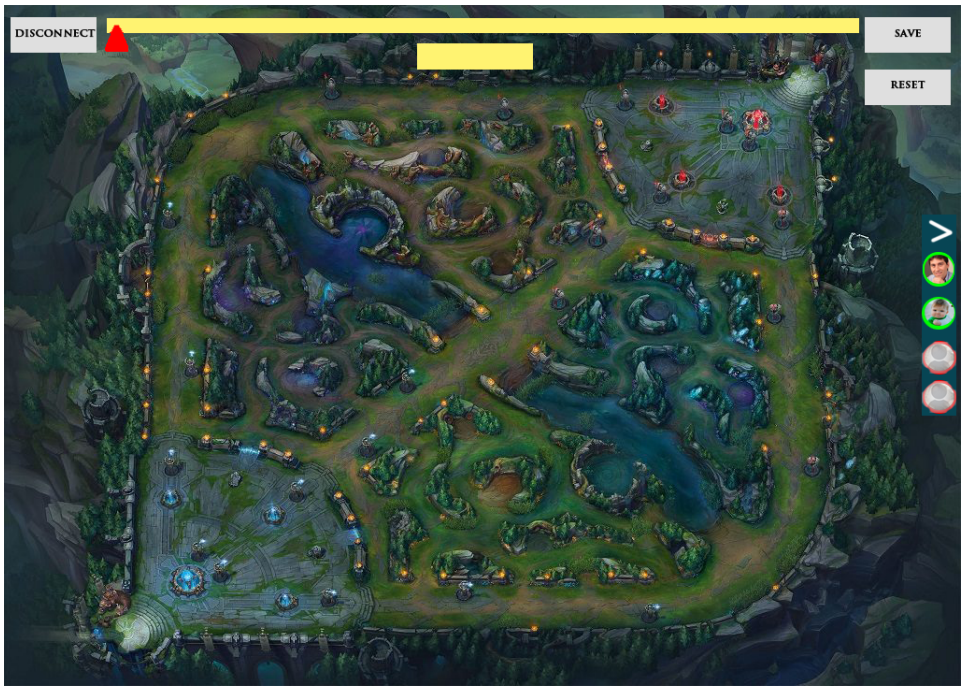

Figura 5.1: Diseño inicial de la herramienta MOBA Tactician.

A continuación, planeamos un diseño de algunas de las acciones que se pueden realizar con los menús en la aplicación para demostrar una secuencia en la que se pone un campeón y se modifica el color de su rastro.

Figura 5.2: Abrir menú.

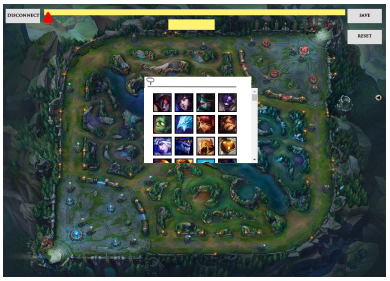

Figura 5.3: Abrir submenú

Primero se pincha el mapa con el botón secundario del ratón y, donde se haya clicado, se abre un menú contextual (véase Figura 5.2). Este menú podemos ver que es personalizado para el juego *League of Legends*, ya que podemos añadir NPCs como guardianes, súbditos o heraldos. En algunas de

las opciones hay que escoger un color, azul o rojo. Esto sirve para elegir el equipo del que será el personaje escogido. Tras seleccionar el equipo azul en la opción de *Añadir Campeón*, se cierra ese menú y se abre otro submenú (véase Figura 5.3). Con este se puede buscar a los campeones, de los que se ven sus imágenes para que sea más sencillo identificarlos.

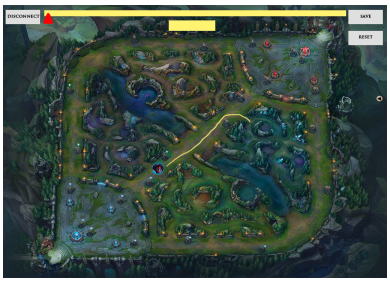

Figura 5.4: Campeón y rastro.

Figura 5.5: Menú de edición.

Una vez seleccionado el campeón (véase Figura 5.4), este se crea con un marco del color del equipo seleccionado, en este caso, azul. A partir de ahora, el usuario puede mover la sticker haciendo *click and drag*. Esta también puede dejar un rastro que indica por dónde se ha movido, para simular el recorrido del jugador que maneja a ese personaje por el mapa. El usuario puede hacer que estos rastros sean visible o invisible, a través del menú contextual que aparece al clicar sobre las pegatinas correspondientes a dichos rastros.

En el caso de que se quiera cambiar el color del rastro para poder diferenciar las rutas de diferentes personajes mejor, se puede clicar con el botón secundario del ratón sobre la sticker del campeón del cual se quiere modificar el color de su rastro. Entonces se abre un menú de edición (véase Figura 5.5) en el que una de sus opciones es *Cambiar color ruta*.

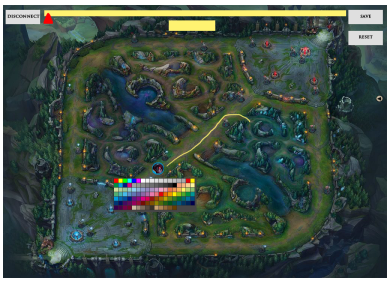

Figura 5.6: Menú de color.

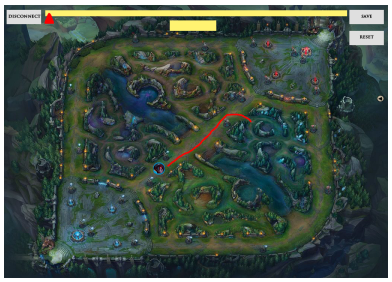

Figura 5.7: Rastro nuevo.

Si se selecciona esa opción, se abre una interfaz con el que el usuario puede escoger un color nuevo para la ruta de ese personaje (véase Figura 5.6). Una vez seleccionado un color nuevo, la ruta se actualiza a ese color (véase Figura 5.7).

5.2. Implementación

En este apartado desarrollamos cronológicamente el proceso seguido para implementar MOBA Tactician.

5.2.1. Prototipación

Tras las primeras fases de investigación y diseño explicadas en capítulos anteriores y en las que hemos reunido los requisitos de nuestra herramienta y hemos hecho un borrador de la interfaz, tenemos suficiente información como para poder desarrollar la herramienta en sí.

Lo primero en lo que nos centramos es en Unity Networking ya que el hecho de que sea una aplicación que funciona en red es uno de los requisitos más importantes, pero también es el que más complica la implementación. Aunque nosotros nunca hemos programado aplicaciones en red, somos conscientes de la gran diferencia que hay entre el código de las aplicaciones monousuario y las multiusuario, y aún así decidimos que lo mejor es familiarizarse con estas tecnologías lo antes posible. Esto lo hemos hecho implementando algunas pruebas en las que actualizamos algunos elementos intentando que el estado de la aplicación sea constante en servidor y clientes.

5.2.2. Menús contextuales I

Una vez terminadas las pruebas iniciales nos centramos en los menús contextuales, ya que consideramos que es la mejor forma de poder realizar la mayoría de acciones sin tener que estar recurriendo constantemente a una barra de herramientas. Para ello incluimos la librería Context Menu, mencionada en el Capítulo 4.

Creamos un menú base llamado AddMenu, que tiene la opción de instanciar una sticker. Recordamos que, en nuestro trabajo, llamamos stickers a las imágenes que añadimos encima del mapa. En el caso de *League of Legends*, estas imágenes son los personajes que los jugadores pueden escoger como avatar, además de NPCs y otros objetos. Como base hacemos que estos stickers se puedan mover pinchando sobre ellos y arrastrándolos por la pantalla. A estas stickers en concreto las denominaremos stickers dinámicas.

Comenzamos a integrar los menús con la parte multiusuario. El primer paso es hacer que tanto clientes como servidor puedan crear stickers y que estas se actualicen en el servidor y en el cliente al ser movidas en cualquiera de los dos lados.

El siguiente menú que incluimos aparece al pinchar sobre una sticker dinámica: el menú EditMenu. Implementamos la opción en el menú EditMenu de borrar la sticker dinámica seleccionada, en el servidor y en el cliente.

Considerando esto como la base del proyecto, nos centramos en expandir el sistema de menús y hacemos que se lean desde un JSON. Decidimos que

AddMenu y EditMenu son dos menús que deberían estar siempre. AddMenu es el menú que aparece cuando se pincha en el mapa y sirve para añadir todos los objetos. EditMenu es el menú que aparece cuando se pincha en los objetos principales y muestra las acciones que se les pueden aplicar.

Durante la creación del sistema de menús también vamos desarrollando diferentes funcionalidades de las opciones de las que disponemos. Cuando se pincha en una de las opciones, lo que se registra en código es el nombre de la opción que se ha pinchando, por lo que necesitamos una forma de saber, dinámicamente, qué tipo de acción es (i.e., si se selecciona una opción con el nombre "Delete Champ", en código se tiene que saber que esa opción destruye el objeto seleccionado). Por esta razón tenemos un diccionario en el que metemos como *key* el nombre de la opción y como valor metemos las características de dicha opción (en este momento sólo se compone por el tipo de acción, pero lo ampliamos más adelante). Se puede acceder a este diccionario desde cualquier *script*, por lo que el *script* de "Context Menu Agent Example" se comprueba, con el nombre de la opción que se ha seleccionado, el tipo de menú que es. De esta forma, nos aseguramos que esta comprobación se haga gracias al diccionario, que se rellena dinámicamente.

Continuado con el desarrollo de las opciones, hacemos que el tipo de opción "Create" genere stickers dinámicas con el nombre de la opción. Esto significa que el nombre del *asset* y el nombre de la opción que se pone en el JSON y que tiene que generar una sticker dinámica con esa imagen, tienen que coincidir.

5.2.3. Sistema multiusuario I

En este momento nos centramos en desarrollar el sistema multiusuario. Aprendemos cómo funcionan los comandos RPC. Montamos dicho sistema de manera que, cuando se selecciona una opción en cualquiera de los usuarios, se manda dicha instrucción al *server* con un método Command. Este, con todos los datos necesarios, llama a un método de tipo RPC, el cual se ejecuta en todos lo clientes. De esta manera nos aseguramos de que todo lo que pase en un cliente, con excepción de visualizar menús, pase en el resto a la vez.

Esto funciona, siempre y cuando no se una un nuevo cliente una vez realizadas la acciones, puesto que no se le actualizará el estado de la aplicación al que tienen el resto de los usuarios. De este hecho nos damos cuenta más adelante, y explicamos cómo lo hemos solucionado.

5.2.4. Menús contextuales II

Nuestro objetivo es tener un sistema de menús contextuales complejos, y uno de nuestros requisitos es tener submenús (al darle a cierta opción del menú, otro menú se abre). Sin embargo, la librería Context Menu no es tan completa como para tener esta funcionalidad, por lo que nosotros la tenemos

que implementar.

Continuando con nuestra lista de requisitos, las stickers dinámicas deben dejar un rastro. Esto es posible gracias al componente TrailRender de Unity. Simplemente lo añadimos al *prefab* de las stickers dinámicas y configuramos la opción que tiene sobre el grosor de la linea.

Creamos una opción en el menú "EditMenu" que permite acceder a los componentes de la sticker dinámica seleccionada y habilitar y deshabilitar este componente, así conseguimos mostrar o esconder el rastro. También creamos una opción que permite borrar el rastro. Y conseguimos que funcionen desde el cliente y desde el servidor.

Uno de nuestros objetivos cuando realizamos el diseño de la aplicación es tener la opción de cambiar el color de los rastros para que cada sticker dinámica pueda diferenciarse. Sin embargo, Unity no tiene un componente o una herramienta que te dé una interfaz para hacerlo. Por esta razón, incluimos la librería HSV en nuestro proyecto. Esta librería nos da la interfaz que necesitamos para cambiar el color de estos componentes, aunque tenemos que modificarla. En el ejemplo de la librería, El HUD siempre está visible, y esto a nosotros no nos interesa. Por eso modificamos su *prefab* para que tenga un botón que al cerrarlo, borre el *game object* completamente. Esto lo hacemos para que, al crearlo de nuevo, coja la referencia del componente del objeto al que tiene que cambiar de color, que es el objeto que se ha seleccionado para abrir esa opción del menú.

Para finalizar con este requisito, hacemos que el color no cambie solo localmente, sino que se actualice en el resto de usuarios y servidor.

5.2.5. Cámara I

En esta fase del proceso consideramos oportuno empezar a implementar el funcionamiento de la cámara. Nada más empezar nos encontramos con un problema: existen diferentes alternativas para la implementación pero todas tienen algún inconveniente y consideramos que se nos pueden complicar de alguna forma. La primera implementación que realizamos es usando el componente ScrollRect y RectMask de Unity. Una vez implementado, confirmamos los posibles inconvenientes de los que nos percatamos antes. Esta alternativa nos obliga a modificar toda nuestra implementación para que todos los elementos pasen de encontrarse en la escena, a estar en el *canvas*. Una vez hecho esto tendríamos que cambiar el funcionamiento interno de la librería *Context Menu* para que en vez de funcionar con elementos de la escena, funcione con elementos del *canvas*. La complejidad de esta tarea es uno de los motivos que nos lleva a rechazar esta opción. También hay otros inconvenientes como que el hecho de que los elementos estén adjuntos al *canvas* y no al mapa. Esto haría que cuando hiciésemos acciones sobre el mapa, como hacer zoom o movernos por él, las stickers no estarían fijas en su lugar sino

que se moverían con la cámara debido a que están unidas a su *canvas*. Esto nos lleva a nuestra segunda opción de implementación, la cual consiste en hacer todo a través de modificar directamente la posición de la cámara. Un inconveniente de esta alternativa es que no sabemos cómo mantener siempre el mapa dentro del *frustum* de la cámara de una forma centrada para que quede estético y para evitar que el usuario aleje la cámara del mapa hacia una zona donde solo haya fondo vacío. Otro problema es que al hacer *zoom* directamente con la cámara sobre el mapa, este se ampliaría pero también se ampliarían las stickers que hay sobre él, algo que no queremos. Al final nuestra decisión fue pausar la implementación de la cámara hasta estar más seguros de cómo realizarla y retomarla en el futuro.

5.2.6. Menús contextuales III

Siguiendo con el sistema de menús, necesitamos un menú que permita ver todos los campeones de una vez. Para que sean más fáciles de buscar, en vez de hacer *scroll* por una lista de nombres, haremos *scroll* por una lista de imágenes. Por lo que creamos un nuevo tipo de submenú: search menu. Este crea un menú con pequeñas imágenes de todos los objetos o personajes que se especifiquen en su JSON, y los adapta para que se vean diez por fila. También se puede hacer *scroll* con la rueda del ratón gracias al componente ScrollRect.

5.2.7. Stickers estáticas I

Necesitamos unas stickers que no se puedan mover al hacer *click and drag* sobre ellas, pero que al clicarlas se cambie su aspecto. Las denominamos stickers estáticas (*static stickers*) y las usamos para elementos inmóviles del mapa, pero que pueden tener diferentes apariencias, como por ejemplos las torres que pueden estar destruidas o sin destruir, o los campamentos de la jungla que pueden estar vacíos o con criaturas dentro. Para esto creamos un nuevo *prefab* llamado "StaticSticker" que tiene un componente llamado "SwapSprite" el cual contiene los dos *sprites* que pueden intercambiarse. También hemos hecho una nueva capa en la escena llamada "Static" para que con el uso del *raycast* en cada clic y en caso de que sea sobre una sticker estática, se llame al método de cambiar su aspecto. A continuación hacemos que estas stickers estáticas se creen automáticamente en el inicio de la escena leyendo toda su información de un archivo JSON.

Ahora lo que queremos es que algunas de las stickers estáticas, las que van a mostrar a criaturas de la jungla, al pasar el puntero del ratón sobre ellas muestren una etiqueta (*poptag*) con información o estadísticas de esas criaturas. Para hacer esto, en primer lugar hemos separado la creación de las stickers estáticas en dos, ya que las criaturas de la jungla tienen *poptag* pero las estructuras como torres o inhibidores no. Esto lo hemos hecho creando

dos JSON (StaticStickers y StaticStickersWithData). Las stickers estáticas creadas a partir de StaticStickersWithData (i.e., criaturas de la jungla) tienen un componente extra llamado "ShowPoptag", que es el encargado de detectar cuándo y dónde hay que mostrar el *poptag* con el texto correspondiente. También cambiamos la estructura del JSON, ya que esa información que se muestra en la etiqueta también tiene que ser leída de él.

También necesitamos actualizar esas estadísticas de las stickers estáticas que acabamos de mencionar y esto se debe a que las criaturas neutrales, a lo largo de la partida, se van haciendo más poderosas y con ello sus estadísticas cambian. Para esto implementamos una barra del tiempo que, dependiendo del minuto que marque, hace que las pegatinas estáticas muestren las estadísticas que tendrían las criaturas en ese minuto concreto en una partida de verdad. Para hacer esto volvemos a modificar los JSON de las pegatinas estáticas para que almacenen los rangos de tiempo y sus estadísticas en esos rangos.

5.2.8. Menús de contexto IV

Queremos hacer una opción que permita generar más de una sticker dinámicas a la vez, por lo que creamos la opción de "multispawn". Se especifica en el JSON que esta es su opción y se añade una lista con qué personajes se deben generar y cuántos de cada. Estas características se guardan en el diccionario de menús, en la parte de valores, que ahora hemos expandido a un *struct* con diferentes variables que no tienen porqué tener un valor, por lo que no hace falta hacer referencias a ellas en los JSONs si no se necesitan. Al seleccionar esta opción, el *script* Context Menu Agent Example comprueba en el diccionario cuántas stickers dinámicas tiene que generar y de qué objetos, y llama a la instrucción de generar stickers dinámicas las veces que sea necesario.

La siguiente opción que creamos es la diferenciación por equipos. Como en los juegos de tipo MOBA hay generalmente dos equipos, queremos la opción de seleccionar previamente el equipo del que cierta sticker será miembro. Para esto creamos un nuevo tipo de botón del menú, con el que primero seleccionas un color, rojo o azul, y solo entonces se abre el submenú correspondiente. En nuestro caso, usamos la diferenciación antes de seleccionar campeones, con lo que la pegatina del campeón tiene un marco del color del equipo, y también antes de seleccionar súbditos, con lo que se crean súbditos rojos o azules. Esto último se consigue porque el nombre de *asset* del súbdito acaba con el color que representa (i.e., CasterBlue) y en esta opción se añade el color al final del nombre.

5.2.9. Guardar el estado de la pantalla

Centrándonos en la serialización para guardar sesiones, investigamos varias formas de realizarla, sin embargo, nos encontramos con un problema: cómo serializar componentes como el TrailRender. Por lo que nos decantamos por una forma más sencilla de guardar el estado de la pantalla y que hemos visto que es suficientemente eficaz en el resto de herramientas que hemos investigado en la primera fase de este proyecto: capturas de pantalla. Consideramos que es suficiente para las clases que los alumnos acaben con un par de capturas de pantalla en una carpeta, que se puedan llevar y estudiar a fondo.

5.2.10. Sistema de control de permisos I

Teniendo en cuenta que el contexto en el que pensamos que nuestra aplicación se va a utilizar es una clase con alumnos mayoritariamente adolescentes, creemos que, para mantener el orden de las clases, es una buena medida hacer que el servidor, que será controlado por el profesor, tenga que darle control a los alumnos para que puedan interactuar con la herramienta. Por eso empezamos a crear un sistema que compruebe si el usuario que está utilizando la aplicación es el servidor o el cliente. Sabiendo esto, le damos al servidor una opción en forma de botón en la pantalla, que permite dar control a los clientes para que ellos también puedan mover objetos, generar nuevos, etc. Sin embargo, en la primera fase de creación de este sistema, el control se le da a la vez a todos los clientes. Como nuestro diseño consiste en que se escoja a qué cliente se le da o se le quita el control, ampliamos este sistema más adelante.

5.2.11. Cámara II

En este momento decidimos retomar la implementación de la cámara, siendo el *script* "CameraMovement", implementado por nosotros, el que controle todas sus acciones y movimientos. Primero hemos hecho que, con las flechas del teclado, la cámara se pueda mover en todas las direcciones de los ejes x e y. En segundo lugar implementamos que con el uso de la rueda del ratón, se pueda hacer *zoom in* y *zoom out*. Por último hemos hecho que, dada una imagen que se tiene como referencia (en este caso el mapa de juego) y de la cual se obtiene sus dimensiones, la cámara genera unos límites que evitan que se pueda salir de unas zonas concretas de la escena y siempre tenga el mapa en pantalla. En concreto, esta implementación asegura que, cuando hagamos *zoom out*, el mapa siempre va a estar centrado en la pantalla y que, cuando hagamos *zoom in* y nos intentemos salir del mapa, no podamos hacerlo.

5.2.12. Menús contextuales VI

Para terminar el sistema de menús queremos hacer que cada sticker dinámica pueda ser *trigger* de un menú diferente, ya que no todas las stickers dinámicas tendrán las mismas opciones de edición. Lo hacemos con una variable en el JSON. Si esta variable no se rellena, se considera que esa sticker dinámicas necesita el menú por defecto: EditMenu.

5.2.13. Menú principal de la herramienta

A continuación, creamos Network Manager Custom. Cuando se crea una aplicación multiusuario en Unity, se debe utilizar un Network Manager con el que los usuarios puedan, por ejemplo, crear sus propias salas o unirse a una ya creada. Unity proporciona una HUD por defecto, muy básico y que no deja configurar lo que enseña y tiene opciones que nosotros no queremos, por lo que debemos crear una que se adapte a nuestras necesidades. Como se puede ver en Network Manager Custom, es un *Manager* que se mantiene durante toda la vida de la aplicación, aunque se cambie de escena, y se encarga de crear la interfaz para que el usuario empiece la sesión o se desconecte de ella. En la herramienta tenemos pues dos escenas: el menú principal y la pizarra. En el menú principal, Network Manager Custom crea dos botones: uno crea una sala nueva y el otro une al usuario a una sala creada. La sala que se crea o a la que se une el usuario, se crea en el puerto de red que se especifica en el campo de entrada que hay debajo de estos dos botones, y que tiene por defecto el puerto 7777. En la escena de la pizarra, se crea un único botón: desconectar. Este lleva al usuario de vuelta al menú principal.

5.2.14. Sistema multiusuario II

Llegados a este punto nos damos cuenta de un error que como hemos mencionado anteriormente, llevamos arrastrando desde principios del desarrollo: cuando un cliente nuevo se une a la sesión, no se actualiza su pantalla al mismo estado que tienen el resto de los usuarios. Esto sucede porque los métodos RPC se han llamado antes de que el nuevo usuario se una, por lo que todo lo que aparece en su pantalla está desactualizado. Achacamos este error a nuestra falta de conocimiento sobre cómo funciona el sistema multijugador de Unity. Sin embargo, rápidamente lo arreglamos. Hacemos que las stickers guarden su información en un *script* llamado infoSticker, que mantiene todos los datos importantes de dicha sticker (imagen, posición, color del rastro) como *syncvars*. Las *syncvars* son variables de Unity que se mantienen actualizadas al valor que hay en el servidor. Además, hacemos uso de *hooks*. Estos son *listeners* que permiten actualizar las variables *syncvars* cada vez que se llame a la función que se especifique en el *hook*. De esta manera conseguimos que cuando un usuario se une, las stickers que aparecen

en su pantalla actualizan todas sus variables *syncvars* al valor que tienen en el *server*, que es el que tienen el resto de usuarios.

5.2.15. Sistema de control de permisos II

En este momento decidimos retomar y rehacer el sistema de control de usuarios mencionado anteriormente con el fin de que el profesor, desde el servidor, sea capaz de dar y quitar permisos individualmente a cada uno de los alumnos. Esto está hecho con el fin de que los alumnos no puedan modificar la explicación del profesor, pero si este último en algún momento quiere permitir a un determinado alumno que la modifique, lo puede hacer de una forma controlada. Esto lo implementamos haciendo que cada cliente que se une a una sesión cree un botón con su referencia en el servidor y, en el momento en el que ese cliente se desconecte, este botón desaparezca. El profesor (usuario del servidor) puede dar y quitar permisos a los diferentes usuarios a través de estos botones. Si el botón está verde quiere decir que el usuario tiene permiso y clicando sobre él se lo quita y se vuelve rojo, y viceversa. En cada uno de los botones se muestra el ID único de cada uno de los usuarios para que se puedan diferenciar. Con botones apareciendo y desapareciendo constantemente necesitamos un gestor que se encargue de organizarlos y debido a esto, creamos un panel a la derecha de la pantalla que contiene todos los botones y los ordena. Esto lo hace gracias a su componente "ControlPanelBehaviour", que gestiona el número de botones que hay y cada vez que se añade o quita uno, reajusta el tamaño del panel y la posición de los botones con el fin de que quede todo lo más centrado y estético posible. También en caso de que todos los clientes se desconecten, al no haber botones que mostrar, ocultará el panel.

5.2.16. Stickers estáticas II

Ahora decidimos que es el momento adecuado de montar el mapa y de colocar todas las stickers dinámicas (i.e., torres, inhibidores y criaturas neutrales). Decidimos extraer todos los recursos directamente del juego y editarlos con *PhotoShop*. Con las criaturas neutrales, como sus campamentos (los lugares donde se encuentran) ya están vacíos en la imagen del mapa, decidimos que las stickers estáticas contengan una imagen de la criatura correspondiente de cada campamento y una imagen transparente. Con esto tenemos las dos alternativas que queremos, campamentos con criaturas y campamento vacío. Respecto a las torres y los inhibidores, en un principio las stickers estáticas iban a tener una imagen de la estructura destruida y otra sin destruir, pero nos damos cuenta de que visualmente no queda estético. Lo que decidimos al final es imitar lo mismo que hacemos con las criaturas neutrales, editamos el mapa para que todas las construcciones estén destruidas y encima de ellas, las pegatinas estáticas pueden cambiar entre una imagen con la

construcción sin destruir y una imagen transparente. En total, editamos y añadimos veintidós torres, seis inhibidores y doce criaturas neutrales.

En este momento la herramienta está prácticamente acabada e incluye todas las funcionalidades establecidas durante la fase de planificación.

5.2.17. Cámara III

Entrando en la última fase de implementación, resolvemos el problema comentado anteriormente con la forma decidida para implementar la cámara. Concretamente con el *zoom*, que al ampliar la imagen del mapa también se amplían las stickers dinámicas que hay en él. Esto no es nuestro objetivo ya que al hacer *zoom in*, las stickers dinámicas ocuparían una gran parte de la pantalla y la aplicación perdería funcionalidad. Además no sería útil para el entrenador o profesor que la imagen de las stickers dinámicas fuesen tan grandes en una zona del mapa tan pequeña, ya que se perdería precisión. Esto lo hemos resuelto creando un componente "StickerScales" y añadiéndoselo a las sickers dinámicas, que son los elementos que queremos que modifiquen su tamaño en proporción al *zoom* de la cámara. Este "StickerScales" es un *listener* del componente de la cámara "CameraMovement" y este último cuando haya acciones de *zoom in* o *zoom out*, llama a todas las stickers dinámicas para que se escalen a través del método "Resize" del "StickerScaler". Por último, en relación al reescalado de las stickers, mencionar que, es dentro del componente "SickerScales" y gracias al *zoom* máximo y mínimo (obtenido de la cámara en el "Start") y al valor del "zoom" en el momento que se realiza, que se pueden hacer los cálculos necesarios. Estos nos permitiran hacer que el escalado de las stickers dinámicas sea proporcional a la amplificación de la cámara.

5.2.18. Stickers estáticas III

Algunas criaturas neutrales no se mantienen constantes durante toda la partida. Como mencionamos en apartado "League of Legends" de capitulo 2, el Heraldo De La Grieta está presente hasta el minuto veinte y luego en su lugar aparece el Barón Nashor. Por otro lado, hasta el minuto treinta y cinco, los dragones son elementales (pueden ser de fuego, tierra, agua o aire) y a partir de ese minuto pasa a estar presente el Dragón Anciano. Con esta gran variedad de aspectos que tienen estas criaturas y que además, dependan del minuto de la partida, nos vemos obligados a rehacer la estructura de las stickers estáticas para poder recoger todas las criaturas del mapa al completo.

Como primer paso modificamos nuestro código (en especial "SwapSticker") y los archivos JSON, para que en lugar de poder intercambiar dos *sprites*, podamos tener una lista ilimitada de *sprites* y podamos iterar por ellos. A esta lista ilimitada de *sprites* la denominamos "sprite Pack" y con esto resolvemos el problema de todos los tipos de dragones elementales.

Nuestro segundo paso es hacer que cambie el sprite pack de las stickers estáticas dependiendo del minuto de la partida marcado por la barra de tiempo. Para hacer esto volvemos a modificar el código con el fin de que cada sticker estática pueda tener dos sprite packs que pueda intercambiar dependiendo del minuto de la partida. También cambiamos los JSON para que almacenen información de los dos sprite packs y de en qué rangos de tiempo usan cada uno. Por último, se hace un nuevo componente "TimeBarListener", que solo poseen las stickers estáticas del Barón Nashor y del dragón, ya que son las únicas que van a cambiar de sprite pack. Este componente es un *listener* de "TimeEvents", que es un componente de la barra del tiempo que, cada vez que el valor de la barra se modifique, llamará a los "TimeBarListeners" pasándoles el nuevo valor, para que cambien de sprite pack en caso de que sea necesario.

5.2.19. Resetear el mapa

Para finalizar la última tarea que realizamos, a parte de arreglar pequeños errores y pulir algunos detalles, es darle la opción al profesor o entrenador de resetear el mapa a su estado inicial. Esto lo hemos hecho por comodidad ya que si se quiere empezar a planear una estrategia desde cero, los usuarios perderían mucho tiempo teniendo que eliminar todos los elementos uno a uno. Implementamos esto haciendo que para cada sticker dinámica que creamos, guardamos una referencia suya en una lista del GameManager y pulsando en el botón *Reset* que aparece en pantalla, se accedan a todas las referencias de la lista y se borren todos sus respectivos stickers. Todo este almacenado de referencias se hace de forma local únicamente en el servidor, ya que él va a ser el único que tenga el botón *Reset*, porque queremos limitar tal decisión únicamente al profesor debido a su importancia.

Con esto cerramos la fase de desarrollo.

5.3. Extensiones a la herramienta

5.3.1. Creación

En primer lugar se tiene que crear una carpeta cuyo nombre será el mismo que el de la extensión y esta carpeta se incluirá en la ruta */Assets/Resources*. Esta carpeta incluirá una imagen del mapa del juego de dicha extensión llamada "map" y cuatro carpetas llamadas: "Items", "Menus", "StaticObjets" y "TimeBar".

5.3.2. Items

En esta carpeta se incluyen todos los *sprites* de las stickers que se pueden crear en el mapa. En el caso del paquete del *League of Legends* se incluyen los

campeones, los guardianes, los súbditos, el Heraldo de la Grieta y el Cangrejo del Río.

5.3.3. Menús

Los menús de contexto y sus respectivos submenús están generados a partir de los archivos JSON que se encuentren en esta carpeta. Esta incluye un JSON por cada menú que queramos incluir en la extensión y será llamado por el nombre que le queramos dar a su respectivo menú. Los dos únicos menús comunes a todas las extensiones son AddMenu y EditMenu. Debido a esto, en la creación de una extensión es necesario crear un JSON para cada uno de ellos. El menú AddMenu es la base para todo el sistema de menús. Se activa al clicar el mapa con el botón secundario del ratón y en él deberían aparecer los objetos que se pueden añadir al mapa. El EditMenu es el menú por defecto para editar los elementos añadidos al mapa.

La estructura que debe seguir un JSON es la siguiente:

```
{
    "options": [
        {
            "name": "name",
            "type": "actionOnPress",
            "menu": "menuThatWillBeTriggered"
        },
        {
            "name": "name",
            "type": "CreateMulti",
            "items": [
                {
                    "name": "name",
                    "number": fullNumber
                },
                {
                    "name": "name",
                    "number": fullNumber
                }
            ]
        }
    ]
}
```

Figura 5.8: Estructura de los archivos JSON utilizados para crear los menús de contexto.

Se tiene un único objeto el cual contiene un atributo "options", este es una lista de objetos y cada uno de estos contiene la información de cada una de las opciones del menú generado por ese JSON.

Cada uno de estos objetos puede tener hasta cuatro atributos: *name*, *type*, *menu* e *items*, siendo los 3 primeros de tipo *string*. "name" es el nombre que tendrá la opción y es el texto que se mostrará en el botón del menú. "type" indica el tipo de acción que realiza el botón al ser pulsado y estas pueden ser:

- **Create:** crea una nueva sticker.
- **CreateMulti:** crea varias stickers al mismo tiempo, que pueden ser de diferente tipo. Esto es especificado en el atributo "items" del JSON.
- **CreateTeam:** utilizado para crear stickers de una criatura que tiene diferente aspecto dependiendo del equipo (i.e. los súbditos).
- **Delete:** elimina el objeto seleccionado.
- **Submenu:** crea un submenú normal.
- **SearchMenu:** crea un submenú de búsqueda. En el paquete de *League of Legends* lo usamos para el submenú de "change champ".
- **SearchMenuTeam:** crea un submenú de búsqueda pero tiene en cuenta el equipo seleccionado. En el paquete de *League of Legends* lo usamos para el submenú de "create champ".
- **ViewRoute:** sirve para mostrar o esconder la ruta del componente "TrailRenderer" del objeto seleccionado.
- **DeleteRoute:** sirve para eliminar la ruta del componente "TrailRenderer" del objeto seleccionado.
- **ChangeColor:** sirve para cambiar el color de la ruta del componente "TrailRenderer" del objeto seleccionado.

El atributo "menu" es una opción que indica el menú que se abrirá al pulsar con el botón secundario sobre el objeto que se cree utilizando esa opción. Este atributo puede no rellenarse ya que no todas las opciones van a crear objetos. Además, en caso de que si al crearlos se quiere que abra el menú EditMenu, no hace falta especificarlo ya que es el menú que se pone por defecto.

Por último, el atributo "items" es el atributo que contiene la información adicional utilizada por el "CreateMulti", mencionado antes para crear varias stickers a la vez. Esta información es una lista de objetos donde cada uno de estos corresponde a un tipo de sticker que se crea. Estos objetos tienen

un atributo "name" y un atributo "number". El primero es un *string* para indicar el tipo de sticker que se va a crear y el segundo un *int* que señala el número de stickers que se van a crear de ese tipo.

5.3.4. StaticObjects

Esta carpeta es utilizada para almacenar toda los archivos necesarios para la creaciones de las stickers estáticas. Tiene que incluir una carpeta "Sprites" la cual contenga todos las imágenes que se vayan a usar para ello y dos archivos JSON, que contienen toda la información de las stickers estáticas, la cual será explicada más adelante. Como se menciona en el capítulo de implementación, los dos archivos JSON se llaman *StaticObjects* y *StaticObjectsWithData*. El primero se utiliza para crear todas las estructuras del mapa (en el *League of Legends* tenemos torres e inhibidores). Estas son stickers estáticas que no van a tener información extra, por lo que tampoco van a tener *poptag*. En cambio *StaticObjectsWithData* es el utilizado para las stickers estáticas de las criaturas neutrales del juego, las cuales tienen *poptag*, debido a que estas criaturas sí tienen estadísticas que mostrar.

La estructura que se tiene que seguir para escribir estos archivos JSON es:

```
{
    "data": [
        {
            "x": positionX,
            "y": positionY,
            "sprites1": [ Sprite1_Pack1, Sprite2_Pack1 ],
            "sprites2": [ Sprite1_Pack2, Sprite2_Pack2 ],
            "data": [
                {
                  "spritePackIndex": 0/1,
                  "key": [ BeginingRange, EndRange ],
                  "value": PoptagText
                }
            ]
        }
    ]
}
```

Figura 5.9: Estructura de los archivos JSON utilizados para crear y situar las stickers estáticas.

Tenemos un único objeto el cual tiene un atributo "data", el cual es una lista de objetos donde cada uno de estos va a contener la información de una sticker estática.

Los primeros dos atributos de estos objetos son "x" y "y", que son *floats* y determinan la posición de cada sticker estática en el mapa.

A continuación se tienen "sprites1" y "sprites2" que corresponden a los dos sprite packs que pueden tener las stickers estáticas. Cada uno de estos atributos es una lista de *strings* que son los nombres de las imágenes de ese sprite pack y en caso de querer una imagen transparente ponemos *null*. Las imágenes que se quieran incluir en estos sprite packs tienen que estar en la carpeta "Sprites" mencionada anteriormente. El segundo sprite pack existe debido a las stickers estáticas que puedan cambiar su aspecto dependiendo del minuto de la partida, como en el *League of Legends* el Barón Nashor y el dragón. En caso de que su aspecto sea constante durante toda la partida solo es necesario el primer sprite pack y el segundo se pondría a *null*.

Por último el atributo "data", el cual incluye la información extra que es: los rangos de tiempo (tienen que ser proporcionales a la barra de tiempo, que es de cero a cuarenta), el sprite pack que se usa en cada uno y la información que se muestra en cada uno. Como ya se comentó antes las stickers creadas a partir del archivo *StaticObjects* no tiene información extra por lo que "Data" será *null*. El archivo *StaticObjectsWithData* seguirá la estructura mostrada en la imagen superior donde "Data" será una lista de objetos en los que cada objeto será un rango de tiempo. En estos objetos el atributo "spritePackIntex" en un *int* que puede ser cero o uno, esto indicará si en ese rango de tiempo se usa el primer o segundo sprite pack respectivamente. El atributo "key" es una lista de *ints* de dos atributos, el primer atributo marca el comienzo del rango de tiempo y el segundo marca el final, estando los dos incluidos en el rango. El atributo "value" es un *string* que es el mostrado en el *poptag* que aparece encima de la sticker estática cuando pasa el ratón por encima suya.

5.3.5. TimeBar

En esta carpeta se almacena un único JSON el cual, se llama *TimeBar*. Este contiene el minuto máximo que va a tener la barra de tiempo, la cual irá desde el minuto cero hasta el minuto definido en el JSON. La estructura de este archivo es un único objeto con un solo atributo llamado "time", el cual es de tipo *int*, que será el límite superior de la barra de tiempo.

5.4. Instrucciones de uso de la herramienta

En esa sección comentaremos brevemente cuales son las instrucciones de uso de la herramienta MOBA Tactician.

5.4.1. Menú principal

Lo primero que se ve al abrir la aplicación es un menú inicial con tres elementos: dos botones y un campo de texto. El botón "Start host" es el

utilizado por el profesor o entrenador para crear una sesión de la cual va a ser el dueño. El botón “Join session"permitirá a los alumnos unirse a la sesión ya creada por el profesor. Por último, el campo de texto sirve para indicar en qué puerto de red quieres crear la sesión o a cuál te quieres unir. Por defecto el número de puerto es el 7777.

5.4.2. Menús de contexto

Una vez dentro de una sesión, en la escena de planificación de estrategias destacamos que los controles están diseñados para poder hacer todas las acciones de la forma más cómoda y rápida posible.

Con la idea de no tener que estar recurriendo constantemente a una barra de herramientas, hemos implementado un sistema de menús de contexto que se activa clicando con el botón secundario del ratón. Si se hace clic en cualquier parte del mapa se crea un menú de contexto que permite crear en el sitio pulsado: guardianes, campeones, Helado del Grieta, dragón, súbditos y el cangrejo de río. La opción de los campeones tiene dos pequeños botones, uno rojo y otro azul, que permiten indicar a qué equipo se quiere que pertenezca el campeón que se va a crear. Cuando se hace clic sobre uno de ellos se abre un submenú el cual muestra todos los campeones del juego a través de sus *sprites*, donde se puede hacer *scroll* con la rueda del ratón y seleccionar el campeón deseado clicando sobre su imagen. Con la opción del dragón se abre un submenú que permite elegir entre dragón de fuego, tierra, agua o aire. Por último, con la opción de “minions” se abre otro submenú que da las variantes de crear un súbdito mago, melée, de asedio, un super-subidito, o crear directamente una oleada completa. Al igual que con los campeones, con cada una de estas opciones se tienen los dos botones con los que se puede indicar a qué equipo va a pertenecer el súbdito creado.

5.4.3. Objetos dinámicos

Una vez creada una sticker haciendo clic con el botón secundario sobre ella, se abre el menú de opciones específicas de esa pegatina. Estas opciones son: mostrar o ocultar su rastro, cambiar el color del rastro, eliminar el rastro o eliminar la sticker. En caso de que la sticker sea la de un campeón, también aparecerá una opción que permite cambiar su *sprite*.

5.4.4. Mapa interactivo

Pasando a explicar el funcionamiento del mapa interactivo, las torres y los inhibidores se pueden destruir o devolver a su estado normal clicando sobre ellas. Lo mismo se puede hacer con las criaturas neutrales de la jungla, las cuales se pueden cambiar de forma o hacer desaparecer haciendo clic sobre ellas. Con estas últimas, al pasar el ratón sobre ellas aparece una eti-

queta encima que muestra información de esa criatura concreta en el minuto marcado en la barra de tiempo.

5.4.5. Barra de tiempo

La barra de tiempo, como acabamos de mencionar sirve para actualizar la información que aparece en las etiquetas al minuto marcado en ella, pero también sirve para actualizar las criaturas del mapa a lo largo de la partida, ya que dependiendo del minuto que sea, puede haber unas criaturas u otras. Esta actualización del minuto de la partida se puede hacer moviendo el *slider* o escribiendo directamente el minuto en el campo de texto.

5.4.6. Cámara

Con respecto a la cámara, se puede mover hacia arriba, abajo, izquierda, derecha y diagonales con las flechas del teclado. También se puede hacer *zoom in* y *zoom out* con la rueda del ratón.

5.4.7. Botones

Con relación a los botones que encontramos en pantalla, en la esquina superior izquierda se encuentra el botón "Disconnect" que permite al usuario desconectarse y salir de la sesión. En la esquina superior derecha está situado el botón "Save" que da la opción de hacer una captura de la imagen mostrada en pantalla. Esta se guardará de forma local en el ordenador del usuario.

5.4.8. Controles especiales del entrenador/profesor

Para terminar, vamos a explicar los controles que únicamente tiene el usuario del servidor. Este, a la derecha de su ventana, tiene un panel con una lista de botones, uno por cada usuario activo en su sesión. Con estos botones puede dar y quitar permisos de modificación de la sesión a los usuarios. Cada uno de los botones tiene un identificador único para que se puedan diferenciar. Estos pueden estar verde o rojo. Si está verde, es que ese usuario tiene permiso y si está rojo, el usuario lo tiene denegado. Por último, el servidor también tiene un botón "Reset" que permite borrar todos los elementos creados del mapa para que no se tengan que eliminar uno a uno.

Capítulo 6

Evaluación de la herramienta

En este capítulo se describen los resultados obtenidos tras la evaluación realizada con un entrenador de e-Sports, de la aplicación MOBA Tactician.

MOBA Tactician, como se expuso en el capítulo anterior, es una aplicación desarrollada por Lucía Carrión García y Jeremy J. Ruzicka Peinado, dos estudiantes del Grado de Desarrollo de Videojuegos de la Universidad Complutense de Madrid, como parte de su Trabajo de Fin de Grado. La idea detrás de este proyecto es crear una herramienta que pueda ocupar el vacío que existe en el mercado de las herramientas profesionales orientadas a los profesores y entrenadores de e-Sports y que pueda ayudar a estos a la hora de planificar sus ideas y estrategias de juego, así como a transmitir esos conocimientos a sus alumnos y jugadores. José de Matías (ex-entrenador de varios equipos profesionales de e-Sports y actual profesor de *League of Legensds* y director de "The Global ESports Academy" en Madrid) es quien recibe la solicitud de realizar una evaluación externa del sistema desarrollado y de la cual se obtienen ciertas conclusiones sobre el mismo.

6.1. Descripción básica

MOBA Tactician es un pizarra interactiva sobre el cual un usuario puede reproducir cualquier situación posible dentro de una partida de un videojuego MOBA. Esto permite plantear tácticas y que estas sean compartidas entre todos los miembros del equipo y así cuando, estos se encuentren ante una casuística similar en una partida de verdad, sepan afrontar la situación. Es una aplicación en red en la cual cada usuario puede observar en su pantalla en tiempo real, las acciones realizadas por profesor e incluso las realizadas por otros alumnos. Esto es debido a que el profesor posee la opción de permitir individualmente a alumnos modificar el mapa, gracias a un sistema de control de permisos que solamente está a disposición del profesor. También

es una herramienta que actualmente, sólo cuenta con contenido para trabajar sobre *League of Legends* pero que se puede ampliar con contenido de otros juegos gracias a su sistema de extensiones. Así que se puede crear una sesión personalizada y basada en otro MOBA a partir de la inclusión de una extensión de MOBA Tactician específica para dicho juego.

Destacar los controles de esta aplicación que a través de un sistema de menús de contexto, hace que esta sea cómoda e intuitiva de utilizar y en la que las acciones se realizan de una forma rápida gracias a evitar sistemas alternativos como la "barra de herramientas".

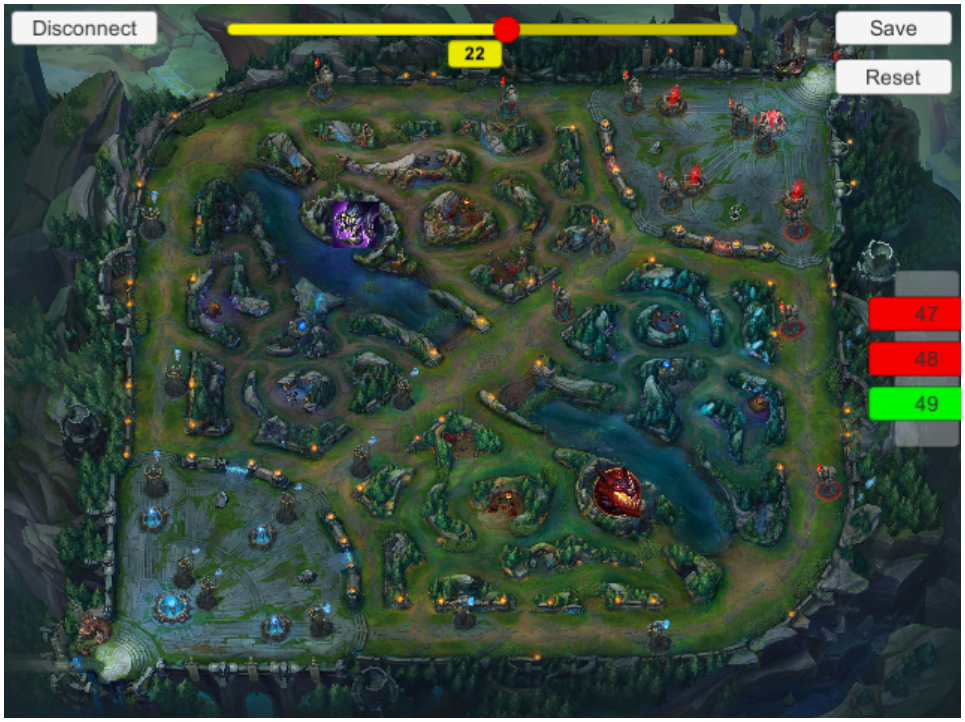

Figura 6.1: Interfaz de la herramienta MOBA Tactician para el juego *League of Legends*.

Las funcionalidades más destacadas que hemos encontrado en MOBA Tactician son:

- El mapa es interactivo, lo cual quiere decir que clicando sobre sus elementos se puede hacer que estos cambien de forma o desaparezcan, al pasar el ratón por encima suya muestren información adicional.
- Es *online*, aunque el profesor mantiene la seguridad de que ningún otro usuario modifique la sesión sin su consentimiento gracias al sistema de control de permisos.

- Puede servir para una gran cantidad de videojuegos debido a que la aplicación crea la sesión de un juego concreto a partir de una cierta extensión con contenido específico para dicho juego.
- Hay una barra de tiempo que permite actualizar el mapa y las estadísticas de sus criaturas en función del minuto indicado en dicha barra.
- Gracias al sistema de menús de contexto, la creación y manejo de elementos en el mapa es sencilla y rápida. Incluso da la posibilidad de crear un grupo de elementos a la vez.
- Los elementos movibles del mapa pueden dejar un rastro si el usuario lo desea. También resalta la funcionalidad de cambiar el color de ese rastro.

6.2. Crítica de un entrenador de e-Sports

Es gracias a la retroalimentación proporcionada por José de Matías a través de la segunda entrevista que los autores tuvieron con él en la que se realiza la evaluación de la herramienta MOBA Tactician. En dicha entrevista se le mostró la aplicación, todas sus funcionalidades y a continuación el profesor pudo probarla. Seguidamente se le hicieron una serie de preguntas para que expresase su opinión al respecto.

Resumiendo, los temas que comentó relacionados con las funciones básicas de la aplicación fueron los siguientes:

- Los controles de la aplicación.
- El sistema de control de usuarios.
- El mapa interactivo
- La barra de tiempo.
- La funcionalidad de captura de pantalla.
- Las stickers.

También comentó aspectos más generales de la aplicación como:

- Funcionalidad.
- Estética.
- Comodidad.
- Utilidad.

- Futuro.

A continuación desarrollamos estos temas a partir de las respuestas de la entrevista, junto con su crítica al proyecto.

6.2.1. Funciones básicas

Como acabamos de comentar, a José de Matías se le preguntó primordialmente sobre las principales funcionalidades de la aplicación. De las cuales se obtuvieron las siguientes críticas:

6.2.1.1. Controles

Según nuestro evaluador, el movimiento de la cámara con las teclas es correcto e intuitivo pero estaría bien tener una alternativa para poder moverla con el ratón. *Clic and drag* sería una buena opción, lo que se tendría que realizar con el botón principal o con el botón central (rueda del ratón), porque el secundario se usa para los menús de contexto. Otra alternativa sería el poder mover la cámara hacia la dirección marcada por el margen de la ventana que está tocando el ratón. Es decir, si el ratón toca el margen superior, la cámara se mueve progresivamente hacia arriba, y lo mismo se haría en las cuatro direcciones. Esta última es una opción interesante ya que así es como se mueve la cámara en muchos juegos, entre ellos el propio *League of Legends*.

6.2.1.2. Sistema de control de permisos

José opinaba que funciona correctamente y es muy útil para hacer que el entorno de la clase o de planificación de un equipo sea más interactivo y participativo. También destacó la importancia de que funciones como el "Reset", estén únicamente limitadas al profesor/entrenador debido a su relevancia.

6.2.1.3. Captura de pantalla

El evaluador consideraba que lo más oportuno es que el profesor pudiese decidir si habilitarlo o deshabilitarlo, a diferencia de como está ahora, donde está siempre habilitado. También razonaba que en ocasiones un profesor puede que no le importe que el contenido mostrado en la aplicación pueda ser guardado por los alumnos, pero puede haber otras ocasiones en la que no quiere que sea así. Esto es un problema, pero tiene difícil solución ya que siempre van a existir diferentes alternativas para guardar lo mostrado en pantalla a pesar de deshabilitar la opción desde MOBA Tactician. Un usuario siempre podría hacer una captura de pantalla desde Windows o usar un programa de terceros para grabar la pantalla.

6.2.1.4. Mapa interactivo

Según el entrenador, el funcionamiento es correcto y muy útil. La capacidad de poder destruir y construir torres y quitar y poner criaturas neutrales es muy buena, pero se destaca la carencia de cuidados gráficos. Estos, si estuviesen más trabajados, harían que la aplicación fuese más atractiva, más agradable a la vista, e "invitaria" al usuario a utilizarla. Entre algunos de estos detalles, José de Matías destacó la carencia de *feedback* de los elementos del mapa, como por ejemplo podría ser que cuando se pase el ratón por encima de un elemento con el que se puede interactuar, este enviase algún tipo de señal. También resaltó el hecho de que el mapa no se debería trata de una imagen plana. El poder darle relieve (3D) lo haría mucho más realista y estético al ojo del usuario.

6.2.1.5. Barra de tiempo

El entrenador opinaba que la funcionalidad actual de la barra de tiempo es correcta, pero la información otorgada por ella (del minuto de juego) es insuficiente. Las estadísticas mostradas en el *poptag* de las criaturas neutrales tienen que, al menos, contener la vida y el daño. En casos como el Barón Nashor o el dragón, donde se calculan a partir de una fórmula, tendría que aparecer tanto la fórmula como el dato una vez calculado. José de Matías comentó que la funcionalidad que actualiza a las criaturas en función del minuto marcado en la barra de tiempo funciona correctamente. También dijo que sería útil una representación gráfica del oro que se ha obtenido de manera pasiva hasta el minuto marcado en la barra.

6.2.1.6. Sistema de extensiones

Según el evaluador era muy interesante y útil para poder ser utilizado con otros juego. Destacaba las posibilidad y la facilidad que supondría crear una extensión para un título tan popular como Fortnite.

6.2.1.7. Stickers

Por el lado positivo el entrenador opinaba que lo más destacable es la funcionalidad del rastro que puede dejar una sticker al moverse y el poder cambiar su color. También resaltaba el poder cambiar de campeón en una stickers sin tener que borrarla y crear otra. Otro aspecto que señaló como interesante es la funcionalidad de poder crear varias stickers a la vez.

Por el lado negativo mencionó que esta es la parte en la que es necesaria más mejora, ya que estéticamente la apariencia de las stickers no es agradable para el usuario. Las stickers en general son demasiado grandes y ocupan demasiado espacio en el mapa. Algunas stickers estáticas podrían estar mejor ajustadas a su posición en el mapa. Incluso en algunos casos, como el

de los súbditos, sería mucho más estético y realista que en lugar de usar su icono, se usase una imagen suya *ingame*. También relacionado con los súbditos, comentó que sería interesante que al crear una oleada, se creasen los súbditos correspondientes separados y ya orientados a la dirección a la que se van a dirigir, en lugar de todos los iconos juntos. Por último, mencionó la importancia de que los guardianes marquen el radio de visión que otorgan, lo cual es más importante que la propia posición del guardián.

6.2.2. Aspectos generales

En esta sección se analizan los factores más relevantes de MOBA Tactician como herramienta dentro del mercado concreto de las herramientas para entrenadores de e-Sports. Esto se realiza a partir de la retroalimentación de la entrevista antes mencionada y de las ideas y comentarios sobre las funcionalidades explicadas en el apartado anterior.

Los aspectos analizados son:

6.2.2.1. Funcionalidad

Según José de Matías la aplicación en este aspecto estaba muy lograda. Todas las acciones que se pueden realizar en la aplicación funcionan correctamente y de una forma eficiente. Algunas de las más destacadas son: el sistema de control de permisos, los menús de contexto, el rastro de las stickers y el mapa interactivo.

6.2.2.2. Estética

Nuestro evaluador consideraba que este era el principal punto a mejorar de MOBA Tactician. Consideraba que visualmente no es atractiva para el usuario y eso podría invitar a estos a usar otras herramientas de la competencia que funcionalmente sean más incompletas pero más "agradables a la vista". Los principales puntos a mejorar que comentó fueron: la interfaz, el tamaño y aspecto de las pegatinas y el mapa, que habría que hacerlo en 3D para que sea más realista.

6.2.2.3. Comodidad

Uno de las principales objetivos al diseñar los controles de MOBA Tactician era dar al usuario una herramienta donde pudiera realizar todas las acciones de forma intuitiva y en el menor tiempo posible. Gracias al sistema de menús de contexto lo hemos conseguido. El único inconveniente encontrado por el evaluador fue el movimiento de la cámara con las teclas. Según él, sería bueno implementar una alternativa para también poder mover la cámara con el ratón.

6.2.2.4. Utilidad

José de Matías comentó que MOBA Tactician es una herramienta útil que se podría utilizar en entornos como una clase o el entrenamiento de un equipo. Por lo general muchos profesores y entrenadores la usarían, aunque no todos porque cada uno tiene su manera de trabajar y podría preferir métodos alternativos.

6.2.2.5. Futuro

El profesor opinaba que a pesar de ser una herramienta muy completa funcionalmente, estéticamente tiene varios puntos a mejorar. Muchos entrenadores o profesores preferirán herramientas más incompletas funcionalmente pero más agradables a la vista. Esto es debido a que pasan mucho tiempo con este tipo de aplicaciones y prefieren algo que les permita trabajar de una forma más agradable, a pesar de sacrificar funcionalidad.

También nos comentó que para que MOBA Tactician tuviese un hueco en el mercado real de herramientas para entrenadores o profesores de e-Sports, tendría que mejorar su aspecto visual y alcanzar un mayor nivel de pulido para que sea un producto más atractivo y cómoda para los usuarios.

Capítulo 7

Conclusiones

En este capítulo exponemos las conclusiones de nuestro trabajo y esbozamos unas líneas de trabajo futuro para siguientes proyectos que quieran continuar con el desarrollo y la mejora de esta herramienta o proponer desarrollos similares.

Recordamos los objetivos propuestos para este trabajo en el capítulo 3 de esta memoria:

- Identificar las necesidades de los entrenadores de e-Sports.
- Crear una herramienta, o un prototipo de ella, que cubra esas necesidades.
- Dicha herramienta debe ser colaborativa, genérica y extensible a cualquier otro juego MOBA.
- Comprobar que el prototipo creado es interesante para la comunidad y determinar cómo hay que mejorarlo para que llegue a usarse a nivel profesional.

A continuación analizaremos si efectivamente hemos cumplido dichos objetivos durante este trabajo y en caso de ser así, desarrollaremos el proceso seguido para su realización:

- **Identificar las necesidades de los entrenadores de e-Sports:**

  Este objetivo se define una vez que descubrimos que la necesidad dentro del mercado de herramientas de e-Sports no está en el lado de los jugadores, lo cual era lo que se pensaba inicialmente. Esto se debe a que ya existen numerosas herramientas orientadas al desarrollo de las habilidades de los jugadores. Incluso en numerosos casos es el propio juego el que proporciona herramientas o alternativas para que los jugadores puedan mejorar sus técnicas, sus tácticas y desarrollar

sus cualidades. Fue gracias a la primera entrevista cuando se descubrió que la necesidad de ayuda estaba en el lado de los entrenadores. Una vez que ya nos habíamos focalizado en los entrenadores, José de Matías también ayudó a identificar que tipo de herramienta se necesitaban. En este proceso aparecieron varias ideas las cuales podrían ser de utilidad para entrenadores y profesores. Entre estas destacaron dos de ellas, una herramienta capaz de "trackear" las estadísticas de los jugadores para seguir su progreso y actividad y un mapa interactivo para planear estrategias, siendo esta última por la que se acabaría optando.

Debido a lo anteriormente mencionado, se considera que en este objetivo se ha realizado un gran trabajo a la hora de reconocer necesidades de los entrenadores y profesores de e-Sports. Se ha sido capaz de identificar un hueco en el mercado de herramientas que puedan ayudar a estos entrenadores y profesores a ejercer su labor de una forma más eficiente. Como ya se ha mencionado esto se consiguió no solo a partir de la retroalimentación obtenida de la primera entrevista con José de Matías sino también con el estudio realizado en el Capítulo 2. Ahí se analiza la situación actual del mercado y las herramientas presentes en él tales como RiftKit y Wottactics.

- **Crear una herramienta que cubra esas necesidades:**

Con relación a este objetivo, se ha conseguido crear una herramienta que cumpla todas las necesidades requeridas por usuarios de aplicaciones pertenecientes a este nicho de mercado. Gracias al estudio mencionado anteriormente, en el Capítulo 2, se identifican todas las necesidades que tendría un usuario de nuestra aplicación. A partir del reconocimiento de estas necesidades, en la fase de diseño (en el apartado 5.1 - "Diseño de la aplicación") se esbozan unas funcionalidades para poder satisfacer estas carencias con ayuda de la retroalimentación obtenida en la primera entrevista. En la tabla comparativa (véase Figura 2.5), se puede apreciar todas estas funcionalidades y si las aplicaciones actuales del mercado (RiftKit y Wottactics) las poseen.

Tras finalizar nuestra aplicación, se compararon sus funcionalidades con las dos herramientas analizadas en el Capítulo 2. Como se puede comprobar en la nueva tabla (véase Figura7.1), MOBA Tactician no sólo cubre todas las funcionalidades que cada herramienta posee sino también las que no poseen.

Características	RiftKit	Wottactic	MOBA Tactician
Funcionalidades básicas de una herramienta de este tipo	✓	✓	✓
Funciona en línea	✓	✓	✓
Guarda capturas del mapa	✗	✓	✓
Resetea el mapa	✓	✓	✓
Sistema para administrar el control sobre los usuarios	✗	✓	✓
Permite escoger diferentes campeones	✗	✓	✓
Permite añadir diferentes NPCs	✗	✓	✓
Da información de las estadísticas de los NPCs	✗	✗	✓
Mapa interactivo	✗	✗	✓
Al hacer zoom, las pegatinas se escalan	✗	✓	✓
Permite cambiar el color de las rutas	✗	✓	✓
Los elementos del mapa dejan rastro	✗	✗	✓
Contine las últimas actualizaciones del juego (i.e., nuevos campeones)	✗	✗	✓
Funciona para varios juegos	✗	✓	✓
Controles cómodos e intuitivos	✗	✗	✓
Limitación de posición de la cámara y otros elementos funciona	✓	✗	✓
El mapa y los NPCs se actualizan según el minuto de la partida	✗	✗	✓

Figura 7.1: Tabla comparativa de las herramientas analizadas con nuestra herramienta.

- **Herramienta colaborativa y extensible a otros juegos:**

  Este objetivo también se ha completado, aunque no haya sido posible crear suficiente contenido como para ilustrarlo. En la fase de implementación explicada en el Capítulo 5, se desarrolla el proceso seguido para hacer de MOBA Tactician una aplicación multiusuario *online* y

que pueda ser extensible para varios juegos. Para hacerla una herramienta *online* se usa el sistema de aplicaciones en red de Unity llamada UnityNetworking. Para hacer que MOBA Tactician pudiese funcionar para varios juegos, se hizo un sistema de extensiones (en el apartado 5.3 - "Extensiones de la herramienta" se explica como crear una extensión) donde la aplicación puede crear una sesión de trabajo para juegos concretos, a partir de los recursos encontrados en estas extensiones.

- **Comprobar que el prototipo creado es interesante para la comunidad y determinar cómo hay que mejorarla para que llegue a usarse a nivel profesional:**

  Este último objetivo también se ha conseguido realizar satisfactoriamente. Es a través del *feedback* obtenido de la segunda entrevista con José de Matías que llegamos a las ideas explicadas en el capítulo 6. Estas manifiestan como MOBA Tactician es una herramienta que funcionalmente cumple todos los requisitos que necesitaría un entrenador o profesor para poder ser utilizada en un aula o para entrenar a un equipo. A pesar de su correcto funcionamiento y buen diseño, visualmente no es llamativa y atractiva para los usuarios. Algunos detalles visuales como la interfaz o el realismo del mapa, hacen que a pesar de cumplir todas las necesidades de los usuarios, estos puedan ser reacios a usarla.

  Con esta idea determinamos que aunque se ha desarrollado un prototipo valioso como acercamiento al mundo de las herramientas para e-Sports, para conseguir que esta herramienta pueda ser adoptada por equipos y empresas a nivel profesional, MOBA Tactician tendría que mejorarse y cuidar mucho más sus elementos visuales y su estética.

7.1. Trabajo futuro

Como se acaba de mencionar en el apartado anterior MOBA Tactician es una herramienta que funciona y cumple las necesidades del mercado al que está orientado por lo que puede ser usado perfectamente en la actualidad. El inconveniente para tener futuro en esta industria es la falta de cuidados estéticos, los cuales la harían más agradable a los usuarios quienes se sentirían "invitados" a usarla. En conclusión, se considera que la mayoría del trabajo que se debería hacer a la herramienta en un futuro, se tendría que centrar en su mejora estética.

Los aspectos que tendrían que ser mejorados para que MOBA Tactician pudiese abrirse un hueco en el mercado serían:

- Darle relieve 3D al mapa para darle más profundidad y realismo.

- Mejorar la estética de los elementos del HUD (i.e. botones, barra de tiempo, etc.).
- Hacer que cada alumno se registre en el menú inicial con un nombre de usuario único para que el profesor pueda identificarles de una manera más cómoda desde el panel de control de permisos.
- En algunos casos (i.e., súbditos) sería más estético y cercano a la experiencia real el usar sus *sprites in-game*, en vez de sus iconos.
- En los elementos de visión, convendría añadir el rango de visión que otorgan.
- Reajustar el tamaño de las stickers.
- Añadir una funcionalidad que permita mover la cámara con el ratón.

Bibliografía

Blizzard Entertainment. Starcraft. PC, entre otros, Disponible en https://us.shop.battle.net/en-us/family/starcraft-remastered.

Blizzard Entertainment. Warcraft iii: Reign of chaos. PC, Disponible en https://eu.shop.battle.net/en-us/product/warcraft-iii-reforged.

Cooper, A., Reimann, R. y Cronin, D. *About face 3.0*. Wiley, 2007.

Epic Games. Unreal engine. Disponible en https://www.unrealengine.com/.

IceFrog. Defense of the ancient: Allstars. PC, Disponible en https://www.dotawc3.com/.

id Software. Quake. PC, entre otros, Disponible en https://quake.bethesda.net/.

Jeff Sutherland & Ken Schwaber. scrum. Disponible en https://www.scrumguides.org/.

JSON.org. Json. Disponible en https://www.json.org/.

Riot Games. League of legends. PC, Disponible en https://euw.leagueoflegends.com/.

Steve Russel. Spacewar! PC, Disponible en https://www.computerhistory.org/pdp-1/c861483915ffc8a67b11b72de8ed403c/.

Supercell. Clash of clans. iOS, Android, Disponible en https://clashofclans.com/.

Taito Corporation. Space invaders. Arcade, Disponible en https://www.taito.com/steam/sie.

Titan Forge Games. Smite. PC, Xbox One, PS4, Nintendo Switch, Disponible en https://www.smitegame.com/.

Ubisoft. Rainbow six - siege. PC, Xbox One, PS4, Disponible en https://rainbow6.ubisoft.com/siege/es-mx/home/index.aspx.

Unity Technologies. Unity. Disponible en https://unity.com/.

Valve Corporation. Counter strike. PC, Xbox, Disponible en https://blog.counter-strike.net/.

Valve Corporation. Counter strike: Global offensive. PC, Xbox 360, Xbox One, PS3, Disponible en https://blog.counter-strike.net/.

Valve Corporation. Defense of the ancient 2. PC, Disponible en http://es.dota2.com/.

Wargaming. World of tanks. PC, Xbox 360, Xbox One, PS4, Disponible en https://worldoftanks.eu/.

Appendix A

Introduction

The world of the e-Sports, also know as electronic sports, is fairly new[1]. When we talk about e-Sports, we mean online or in-person video game championships in which players compete to be the best. The first competition was in 1972, at Stanford University, where students competed by playing the video game *Spacewar!* (Steve Russel, 1962). In it, two players pilot spaceships and must destroy the opposing player. The first public event was eight years later, in 1980, and received a lot of media attention, especially in the *Space Invaders* (Taito Corporation, 1978) competition, an arcade game in which a single player pilots a spaceship and must destroy the alien enemy ships. Players challenged each other for the highest score. Later, in the 1990s and with the introduction of the Internet, users were finally able to connect and compete from different parts of the world. Thanks to this, video games spread. Different companies, such as Nintendo, began to organise their own championships and, in 1997, the Red Annihilation tournament was organised for the game *Quake* (id Software, 1996). This is a first-person shooter where the use of 3D graphics was revolutionary. It was played in multiplayer mode, connecting via local area network or the Internet. This tournament gathered around 2000 players and is considered the first modern e-Sports championship. From that moment on, tournaments continued to be organized and the e-Sports won public. In 2018, around 3985 competitions were organized worldwide and around 161,8 million dollars in prizes were distributed[2].

It is clear that, despite their short history, e-Sports move large amounts of money and jobs. That is why, with this Final Year Project, we intend

[1] Medium.com (2018). The history and evolution of e-Sports [online] Available in: `ttps://medium.com/@BountieGaming/the-history-and-evolution-of-esports-8ab6c1cf3257`

[2] Esportsearnings.com (2018). Overall e-Sports stats for 2018 [online] Available in: `https://www.esportsearnings.com/history`

to enter into this world, specifically in the technological area related to the software applications of interest for this sector. Thanks to the initial research process, we detected a big gap when it comes to tools oriented to e-Sports coaches and that's why we see the need to provide them with specific tools for their work. This project has for main motivation to reduce part of that void.

For a couple of years now, e-Sports academies have started to proliferate. In them, teachers and coaches try to transmit their knowledge and strategies to their players but, as we have said before, this becomes difficult without the right tools. Therefore, we propose the development of an application with which teachers/professional coaches can devise and explain strategies of MOBA-type games to their students/players. MOBA games are multiplayer strategy games in which the player controls only one character, the member of a team that faces another in combat. Normally, the objective in this kind of game is to defeat the opposing team by destroying their base. We choose this genre of video games because it is extremely popular in e-Sports, their structure and main gameplay is very similar in all of them —so we can cover quite a few titles— and, in addition to that, we are presented with the opportunity to follow the guidelines of a real customer who works with MOBA games. We use *League of Legends* (Riot Games, 2009), as our client is very interested in having a tool that specifically covers this title, one of the most popular in this kind of competition. However, our tool is designed to be extendable to any other game of the same genre, to meet the needs of other coaches specializing in titles.

A.1. Report's structure

This project's report consists of the following chapters:

1. The current section where we present our proposal, its origin and the reasons why we have decided to embark on the development of this project.

2. Analysis of the current situation of our tool's area, its competition in the market, and a reference to the interview with José de Matías, director of the training centre The Global ESports Academy.

3. Explanation of the main objectives of the project and specification of the requirements of the application.

4. Description of the different techniques and resources we have used for the design and development of our entire project.

5. MOBA Tactician, the details of the application's design, the implementation, and the development of this system.

6. Explanation of the tests and analysis of the results carried out with users on MOBA Tactician, the application developed.

7. Concise analysis of the fulfilment of the objectives and requirements obtained in the third chapter, along with a reflection on the future lines of work that would allow this system to continue to be extended.

Appendix B

Conclusions and future work

In this chapter, the conclusions of our project are exposed, and also some future lines of work are outlined for subsequent projects that would want to continue the development and improvement of this system o propose similar developments.

We recall the goals proposed for this project in chapter 3 of this report:

- Identify the needs of e-Sports coaches.
- Create a tool, or a prototype of it, that meets those needs.
- Such a tool has to be collaborative, generic, and extendable to any other MOBA game.
- Test that the created prototype is interesting for the community and determine how to improve it, so that it can be used at a professional level.

Next, we will analyze if we have, indeed, fulfilled these goals during this project.

- **Identify the needs of e-Sports coaches:**

  This objective is defined once we discover that the need within the e-Sports tool market is not on the players' side, which was what it was initially thought. This is because they already exist numerous tools orientated to players' abilities development. Even in numerous cases, it is the game itself that provides the tools or alternatives for players to improve their techniques, their tactics, and develop their qualities. It was thanks to the first interview, that it was discovered that the need for help was on the coaches' side. Once we were focused on the coaches, José de Matías also helped identify what kind of tool was

needed. In this process, several ideas appeared which could be useful for coaches and teachers. Among these, two of them stood out: a tool capable of tracking the players' statistics to be able to keep an eye on their progress and activity, and an interactive map to plan strategies, the latter being the one that would be chosen.

Due to the above, it is considered that, regarding this objective, a great job has been done at recognizing the needs of e-Sports coaches and teachers. We have been able to identify a gap in the market of tools that help these coaches and teachers to perform their work more efficiently. As already mentioned, this has been achieved not only from the feedback obtained from the first interview we had with José de Matías but also from the study carried out in the Chapter 2, where the current market situation and the tools present in it, such as RiftKit and Wottactics, are analyzed.

- **Create a tool, or a prototype of it, that meets those needs:**

Concerning this objective, we have managed to create a tool that meets all the needs required by users of applications belonging to this niche market. Thanks to the study mentioned above, in chapter 2, all the needs that a user of our application could have, have been identified. From the recognition of these needs, in the design phase (headland 5.1 - "App Design") some functionalities are outlined to be able to satisfy these deficiencies with the help of the feedback obtained in the first interview. In the comparative table (see Figure 2.5), we can see all these features and whether the current market applications (RiftKit and Wottactics) have them.

Once we finished our app, its functionalities were compared with the ones analyzed in chapter 2. As we can see in the new table (see Figure 7.1), MOBA Tactician, not only meets all the functionalities that each tool has, but also those that they do not.

Features	RiftKit	Wottactic	MOBA Tactician
Basic functionalities of this type of app	✓	✓	✓
Works online	✓	✓	✓
Saves screenshots of the map	✗	✓	✓
Resets the map	✓	✓	✓
Permission control system	✗	✓	✓
Allows to choose different champions	✗	✓	✓
Allows to add different NPCs	✗	✓	✓
Gives information of NPCs statistics	✗	✗	✓
interactive map	✗	✗	✓
Stickers scale when you zoom	✗	✓	✓
Allow to change the trail color	✗	✓	✓
Elements of the map leave a trail	✗	✗	✓
Contains the latest game updates (i.e., newest champions)	✗	✗	✓
Work for several games	✗	✓	✓
Comfortable and intuitive controls	✗	✗	✓
bounds for camera and other elements work	✓	✗	✓
The map and the NPCs are updated according to the minute in the game	✗	✗	✓

Figure B.1: Comparative table of the analized tools with ours.

- **The tool has to be collaborative, generic and expandable to any other MOBA game:**

  This objective has also been completed, although it has not been possible to create enough content to illustrate it. In the implementation phase, explained in chapter 5, the followed process to make MOBA Tac-

tician a multi-user online app that can be extended for several games is developed. To make it an online tool, the Unitys net system called UnityNetworking is used. Moreover, to make MOBA Tactician work for different games, an extensions system was made (at headline 5.3 "Extensiones de la herramienta" explains how to create an extension), so the app can create a working session for specific games, from the resources found in these extensions.

- **Test that the created prototype is interesting for the community and determine how to improve it, so it can be used at a professional level:**

  This last objective has also been successfully achieved. It is through the second interview with José de Matías that we come up with the ideas explained in chapter 6. These ideas explain how MOBA Tactician is a tool that functionally meets all the requirements that a coach or teacher would need to be able to use it in a class or to train a team. Despite its proper functioning and good design, it is not visually appealing to users. Some visual details such as the interface or the realism of the map can make that, despite meeting all the needs of users, they may be reluctant to use it.

  With this idea, we establish that, although a valuable prototype has been developed as an approach to the world of tools for e-Sports, to get this tool to be used by teams and companies at a professional level, MOBA Tactician would have to be improved and taken better care of its visual and aesthetic elements.

B.1. Future work

As we just mentioned in the previous paragraph, MOBA Tactician is a tool that performs and meets the needs of the market to which it is orientated so that it can be used perfectly today. The drawback to having a future in this industry is the lack of aesthetic care, which would make it more pleasant to users who would feel "invited" to use it. In conclusion, it is considered that, most of the work that should be done in the tool in the future, would have to be focused on its aesthetic improvement.

The aspects that would have to be improved for MOBA Tactician to get a place in the market would be:

- Give 3D relief to the map to give it more depth and realism.
- Make the HUD elements (i.e., buttons, time bar, etc.) more aesthetic.
- Make each user register himself in the main menu with a unique name so, the teacher can identify them more comfortably from the permission

control panel.

- In some cases (i.e., minions), it would be more aesthetic and closer to real experience if in-game sprites were used instead of their icons.
- In the visual elements, the range of vision they provide would be useful to have.
- Readjust the size of the stickers.
- Add a feature that allows moving the camera with the mouse.

Apéndice C

Aportaciones individuales de los autores

Comenzamos este apartado notando que dado nuestro estilo de trabajo, las tareas que a continuación listamos no han sido realizadas únicamente por una persona. Si bien uno de los participantes ha trabajado más en algunas tareas que el otro, normalmente trabajamos juntos y nos vamos pasando las tareas cuando así lo requerimos.

Queremos decir con esto que consideramos que ambos participantes de este proyecto han trabajado equitativamente y con la misma dedicación que el otro.

C.1. Lucía Carrión García

- Transcribir la primera entrevista realizada a José de Matías.
- Escribir el Capítulo 1: Introducción.
- Traducir el Anexo A: Introduction.
- Analizar las dos herramientas del Capítulo 2: Riftkit y Wottactics.
- Escribir el Capítulo 2: Estado de la cuestión (excepto el apartado 2.1.1. League of Legends).
- Escribir el Capítulo 3: Objetivos y especificación.
- Escribir el Capítulo 4: Metodología y herramientas.
- Escribir el resumen del trabajo.
- Crear las imágenes del diseño por Photoshop.

- Escribir el apartado "Diseño" del Capítulo 5: MOBA Tactiacian.
- Programar la lectura dinámica de los menús por JSON (script *MenuManager*).
- Crear todos los diferentes JSON de los menús.
- Hacer que los stickers cojan la imagen de la opción que se ha seleccionado.
- Crear el diccionario de menús y los diferentes tipos que pueden ser los menús.
- Programar la creación de submenús (excepto su activación).
- Hacer que los stickers dejen un rastro.
- Hacer que los rastros se activen, desactiven y borren, tanto en el servidor como en el cliente.
- Incluir la librería HSV y hacer que funcione para cambiar el color de los rastros tanto en el servidor como en el cliente.
- Crear los dos submenús de búsqueda: creación de campeones y cambiar de imagen a los campeones (tanto su JSON como su funcionalidad en el código).
- Hacer que los stickers de los campeones puedan cambiar sus imágenes a la de otro campeón.
- Recopilar las imágenes de los diferentes personajes y NPCs que utilizamos para las stickers normales.
- Modificar y recortar con la herramienta Photoshop el mapa y los NPCs y objetos que utilizamos para las stickers estáticas.
- Crear la barra del tiempo con su campo de input, coordinarlos y hacer que funcionen tanto en servidor como en cliente.
- Crear la funcionalidad de multispawn (que se generen varias stickers a la vez).
- Hacer que se pueda escoger el equipo de los personajes en los menús (cambiando los JSON y scripts correspondientes).
- Hacer que los stickers tengan. si es necesario, un marco que los diferencie por equipo, tanto en el servidor como en el cliente.
- Crear y programar la opción de tomar capturas de pantalla.

- Hacer el primer sistema para dar y quitar control a los clientes desde el servidor (este sistema acaba siendo modificado).
- Hacer que el mapa se ajuste a la pantalla, y el collider del mapa se ajuste a este.
- Comentar y limpiar código.
- Hacer que los stickers puedan tener diferentes menús de edición.
- Crear la UI del Network Manager y su script (NetworkManagerCustom).
- Cambiar todos los métodos RPCs de mover stickers, cambiar sus imágenes, cambiar el color de su rastro, ver u ocultar rutas, a un sistema de Hooks y SyncVars.
- Hacer que la imagen de un cliente que se ha desconectado desaparezca de la lista de clientes conectados.
- Escribir la mitad del apartado "Implementación" del Capítulo 5: MOBA Tactician, con la parte del proceso de implementación que he realizado.

C.2. Jeremy J. Ruzicka Peinado

- Transcribir la segunda entrevista realizada a José de Matías.
- Escribir el apartado 2.1.1. League of Legends del capítulo 2
- Escribir el capítulo 5: MOBA Tactician. Excepto el apartado 2.1.1. Diseño y la parte realizada por Lucía del 5.2. Implementación.
- Escribir el capítulo 6: Análisis.
- Escribir el capítulo 7: Conclusiones y trabajo futuro.
- Traducir el capítulo 7: Conclusions and future work. 7conclusiones.
- Traducir el resumen del trabajo.
- Crear la tabla de comparación de las herramientas del capítulo 2 (véase Figura 2.5).
- Diseñar el framework de interacción de la aplicación.
- Diseñar la interfaz de la aplicación.
- Implementación de los primeros prototipos de la aplicación.

- Incluir la librería ContextMenu y hacerla funcionar con los menús básicos (AddMenu y EditMenu).
- Programar que los submenús se activen.
- Poder crear y eliminar stickers.
- Implementación de la primera versión de SwapSprite para que las pegatinas estáticas puedas cambiar su apariencia entre dos imágenes, cuando haces clic sobre ellas.
- Hacer que las stickers estáticas estén siempre coordinadas tanto en servidor como en clientes.
- Programar la lectura dinámica y creación de las stickers estáticas por JSON (script *GameManager*).
- Crear todos los JSONs de las sitckers estáticas.
- Extracción de todas las imágenes necesarias para hacer las sitckers estáticas directamente del juego (*League of Legends*).
- Colocar todas las stickers estáticas en su posición.
- Creación e implementación del *poptag*.
- Hacer que la información del *poptag*, este actualizada con el minuto marcado en la barra de tiempo.
- Implementación del movimiento de la cámara, que le permite moverse en las ocho direcciones.
- Crear la funcionalidad *zoom in* y *zoom out* de la cámara.
- Implementación del sistema que evita que la cámara se salga de los margenes del mapa.
- Implementación de ajustes que permiten a usuarios unirse a la sesión en cualquier momento y que esta este actualizada.
- Hacer el sistema final para dar y quitar control a los clientes desde el servidor.
- Realización de ajustes en el código, que evitan a los usuarios sin permiso puedan hacer cambios en la sesión.
- Cambiar los métodos RPCs relacionados con las stickers estáticas, a un sistema de Hooks y SyncVars.
- Crear el panel de control de permisos y que este organice los botones para el usuario del servidor.

- Implementar el escalado de las sickers dinátimas cuando se hace *zoom* en el mapa, haciendo que las sickers dinámicas sean *listeners* de la cámara.
- Reimplementación del componente SwapSprite para que, las stickers estáticas en lugar de poder tomar la apariencia de únicamente dos imágenes, puedan tomar la apariencia de una lista de imágenes.
- Hacer a las stickers estáticas *listeners* de la barra de tiempo para que estas puedan cambiar su aspecto dependiendo del minuto marcado por la barra.
- Crear y programar la opción de resetear el mapa.
- Comentar y limpiar código.

Apéndice D

Transcripciones de las entrevistas a José de Matías

D.1. Primera entrevista

D.1.1. Datos

Fecha: 4 de octubre de 2018

Nombre: José de Matías - Director de The Global ESports Academy / Entrenador de League of Legends

D.1.2. Transcripción

Jeremy: Bueno, primero. ¿Hace cuánto más o menos comenzó como tu involucración en el mundo de los e-Sports y en el deporte electrónico competitivo?

José: Vale. A ver, yo acabo mi segunda carrera, que es la de ciencias de la actividad física y el deporte, en Valencia. La termino en 2007-2008. No recuerdo ahora mismo la fecha exacta, pero 2007-2008. Y tengo un profesor de carrera de universidad que estaba enamorado de la aplicación deportiva que hacen en los países nórdicos. Y me surge la oportunidad, también un poco yo me dejo arrastrar por mi profesor, de ir a Estocolmo durante seis meses a estudiar un poco cómo funciona el mundo del deporte allí. Pero no del deporte electrónico. Del deporte. A secas. En 2008, ¿había videojuegos? Sí. ¿Había videojuegos competitivos? Sí, pero todavía ni siquiera se les llamaba e-Sports. Y yo me marcho a Estocolmo a estudiar el deporte pero el finés y- Bueno, el idioma que hablan por allí tiene muchas cosas parecidas con el inglés pero en el resto no se parece en nada. Y en uno de esos e-mails pues ellos hablan una mezcla de inglés con finés y cuando escriben la palabra é-Sports'pues yo, como no la conocía, entiendo que hablamos de deportes

normales y corrientes pero que le han colado ahí una é'pues no sé muy bien por qué. Y me voy, y cuando llego ahí a la casa donde voy a pasar seis meses, descubro que estoy en una casa en la que está un equipo profesional, que en aquella época era VirtusPro, un equipo profesional de CS:GO, —de hecho bastante bueno—, entrenando. Entrenando en ordenadores, claro. Y que ellos estaban interesados, pues claro era un toma y recibe ¿no? vamos a decir. Ellos me permitían estar allí aprendiendo su deporte entre comillas y yo lo que hacía era... Ellos ya hacían unas horas de gimnasio de entrenamiento y como había acabado mi carrera de ciencias de la actividad física y el deporte, les llevaba todo lo que era el apartado de entrenador personal, dinámicas de grupo incluso, etcétera. Claro, cuando llego allí y veo a cinco tíos rubios enormes viendo- jugando a CS:GO lo primero que pienso es: "Me han engañado, o algo pasa aquí..Entonces el entrenador, se llamaba Mike, me explicó un poco pues que era todo esto, y me explicó: "No, no. Yo no te he engañado. Te puse en el correo e-Sports."Yo dije: "no tengo ni idea de lo que me estás hablando", y me lo explicó. Me explicó: "Pues mira esto aquí se juega mucho, incluso sale en la televisión, allí SPN lo lleva emitiendo muchos años, y somos jugadores de CS:GO". Digo: "No, no, no. Si yo el CS:GO sé lo que es pero, pero es un videojuego". Dice: "No, en nuestro caso es un deporte. En nuestro caso entrenamos muy duro, entrenamos cinco días a la semana, tenemos un régimen estricto de dieta, tenemos un régimen estricto de ejercicio. Porque cuanto más seamos capaces de aguantar en el ordenador más frescos estamos, más frescos estamos entonces estar en forma es muy importante para nosotros y por eso nos interesaba que vinieras pero nosotros pensábamos que sabias a donde venias". Digo: "No tenía ni idea pero me gusta la idea, de hecho para mi trabajo de final de carrera, de licenciatura, esto me parece la bomba". Y llamo a mi profesor, obviamente, a la universidad. Le cuento la historia, al principio no estaba muy convencido Üh... es que eso deporte tal". dice: "Pero estás en los países nórdicos ya, no te voy a hacer venir así que no te preocupes, yo sé que has escogido esto un poco guiado por mi persona y es un poco culpa mía así que quédate allí y hazlo sobre eso que además yo veo que a ti te convence". Pues yo he jugado a videojuegos toda la vida, y he hecho deporte. Pasa el tiempo, hago mi trabajo de fin de carrera y la cuestión es que me sale muy bien, de hecho me ponen una matrícula en la universidad y el profesor alucina con lo que le cuento que he visto allí. Entonces yo ya cuando vengo aquí a España tengo el mono de decir, y ¿por qué no tengo eso yo aquí en España? Entonces fue cuando empiezo a colaborar con cosas como Campus Party en Valencia en aquella época -no sé si os sonará, que a día de hoy es la GameJam, pero antes era la CampusParty-, en las primeras 'jameséspañolas empiezo a colaborar con ellas, me empiezo a meter un poco más en el mundo este de videojuegos y competición. Que yo ya estaba metido en videojuegos y competición pero no a este- hasta este punto. Y acabo fundando un club. Fundo el club, empiezo a hacer mis primero estudios sobre videojuegos, que tengo ya varios trabajos

publicados al respecto. El club asciende a lo que es la primera división española etcétera y ya es cuando ya más o menos me inicio de verdad en este mundo. Y desde ahí hasta ahora he pasado por el Valencia y por muchísimos, muchísimos clubes deportivos. Y de hecho en la universidad de Valencia escreo que soy el que más años lleva estudiándolo, y no voy a decir de España por si se me escapa alguien, pero estoy casi seguro de que si no soy el que más años lleva deportes electrónicos desde un punto de vista académico, soy el segundo, vamos.

Lucía: La academia, ¿cuántos años lleva?

José: ¿Esta?

Lucía: Sí.

José: Lleva... seis meses. La academia lleva seis meses, de hecho yo tengo mi plaza en Valencia y he cogido un excedencia para venir a trabajar aquí. Porque la oferta era muy buena y porque el proyecto es muy interesante. Esta academia lo que pretende es educar personas a través de los deportes electrónicos. Porque los deportes electrónicos tienen una serie de valores muy importantes como el trabajo en equipo, te obligan a comunicarte con otras personas y eso lleva a resolución de conflictos, liderazgo en algunos caso si resulta eres tú el que tiene que tomar la iniciativa. También es saber, vamos a decir, aceptar órdenes de alguien, debate, comunicación espontánea... Hay un montón de cosas muy importantes en los deportes electrónicas que la gente que ha fundado esto cree firmemente que puede educar, y yo también lo creo, porque de hecho yo te digo que después de muchos años estudiándolo me parece que son una herramienta increíble. Y me convenció el proyecto. Sí es verdad que yo llevaba unos años en la universidad que estaba un poco... Sí, llevo mucho tiempo investigando esto pero llega un momento en el que me queda poco que investigar de ello. Me apetecía dejar de, por así decirlo, escribir y ponerme a hacer, y esto era un poco ponerme a hacer. Y aquí he podido poner en práctica los exer-games, que son los videojuegos que te ayudan a hacer ejercicio, y demostrar que se puede perder peso con ellos. Quitando un poco el mito de encima de que todos los videojuegos fomentan la obesidad o que si juegas a videojuegos vas a acabar siendo una persona con problemas cardíacos, etcétera. Esto es un poco, va a ser un poco mi laboratorio personal ahora mismo.

Lucía: Pues está genial.

Jeremy: ¿Y con qué videojuegos has trabajado en un ámbito más profesional?

José: A ver, desde el ámbito profesional deportivo, vamos a decir. Yo he tenido Call of Duty, que lo tuve en wSystem y de hecho quedamos segundos de España en la SuperLiga, bueno, SuperLiga en aquel momento era el IVP creo que se llamaba los [] si no me equivoco. CS:GO, hemos sido campeones de la TLP, hemos quedado también terceros de España. Era un equipazo, tenia jugadores aquí como Draguno, Torpe, jugadores de CS:GO muy buenos.

League of Legends. League of Legends he entrenado equipos masculinos en Super Liga Orange quedando segundos. Ahí ya no he entrenado, he hecho de mánager de equipos con Pepinero, con Adri, con jugadores LCS. Fui uno de los primeros en tener un equipo femenino y ganamos la SL femenina europea. He tenido equipo de Clash Royal, es el último equipo que he tenido en el Valencia, que quedamos segundos también. Nos ganó Gigants en la final. Y luego a parte de eso, como director del club wSystem, he tenido prácticamente todos los videojuegos, o sea los más locos que se te puedan ocurrir. He tenido Street Fighter, he tenido Tekken. He tenido Racing, Gran Turismo, y juegos de coches. Prácticamente de todo. DOTA también he tenido. He tenido prácticamente de todo pero, pero eso es a nivel competitivo. A nivel personal. Yo empecé jugando Diablo con mi padre, y Diablo en aquella época desarrolló una plataforma de competición que se llamaba BattleNet. En BattleNet yo era el segundo, tercero, dependiendo, mejor en PvP. Ese fue mi primer contacto de competición, te hablo que tendría yo 18 años a lo mejor. Sabes, hace ya muchos, muchos años. Luego, empecé a competir en Doom y Quake y ya de Doom y Quake pasé a World of Warcraft y de War of Warcraft empecé con el League of Legends. El League of Legends soy diamante actualmente, que para una persona de 36 años, la gente se suele sorprender. Pero es verdad que lo juego desde que salió. Llevaré 10 años jugando al LOL casi, si no llega. 9, 10 años. Desde que salió. Entonces luego a nivel educativo, lo que es educativo exclusivamente he trabajado sobretodo con el League of Legends, Fornite, y Minecraft. Por ejemplo, el Minecraft para educar es la leche. No sé si sabéis que hay un tío, que está loquísimo en mi opinión, que con el Minecraft y el 'redstone'—no se si sabéis ni lo que es, esto que pinta una especie de circuito rojo en el suelo—, ha programado un emulador de GameBoy. ¿No lo habíais visto?

Lucía: No.

José: Buscadlo en Internet. O sea, es, haciendo operadores lógicos binarios, de encendido/apagado ha hecho un emulador de GameBoy. O sea, es un trabajo tan— Es como hacer las pirámides casi. Es un trabajo tan bestial que yo cuando lo vi aluciné.

Jeremy: Vale. Ahora vamos a preguntar cosas más relacionadas con la academia.

José: Muy bien.

Jeremy: Cuando llegan alumnos nuevos que se quieren meter en el mundo de los e-Sports, ¿con qué intención suelen venir a la academia?

José: Depende de la edad sobretodo. Entre los 14 y los 16 años, ahí hay un punto intermedio en el que es un mezcla entre ambas, pero por debajo de los 14 siempre vienen a divertirse. Esa es su principal motivación. Y por encima de los 16 siempre vienen a ser mejores. Porque para divertirse pues pueden jugar en su casa. Entonces luego tienes ese bypass entre los 14 y los 16 en el que unos vienen para una cosa y otros vienen para otra. Que ese es el

problema que tenemos, que solemos hacer una evaluación inicial de ese tipo de persona para ver si su motivación es una u otra para ponerlo en un grupo o en otro. Pero básicamente si tuviéramos que definir las motivaciones, son esas dos: unos divertirse, otros mejorar.

Lucía: Entre los que vienen para mejorar, ¿hay alguno que específicamente os haya dicho que quiere meterse en el mundo profesional?

José: Todos.

Lucía: ¿Todos?

José: Todos quieren meterse en el mundo profesional. Pero es nuestra obligación decirles la verdad. O sea, esto es como en el mundo del fútbol: ¿cuántos llegan a ser futbolistas de primera división?

Lucía: Muy pocos.

José: Menos del 1 por ciento de hecho. Y la posibilidad de que seas tú, pues está ahí, sí, pero es tan remota que yo no te puedo garantizar que lo vayas a conseguir. Cuando vienen aquí les decimos un poco lo mismo. O sea, que te conviertas en jugador profesional de e-Sports es una posibilidad muy remota que va a depender que: 1) Que seas muy, muy, muy, muy, muy bueno. 2) Que trabajes mucho muchísimo. Y 3) Que encima tengas suerte. Porque es que hay gente muy buena, que trabaja mucho pero no tiene suerte y se lesiona o pasa cualquier cosa y así no llegan. Yo lo he visto, en este mundo, pasó con Revend hace no mucho. Un chaval que lleva jugando al League of Legends montón de tiempo, hace vídeos en YouTube, hace prácticamente de todo. Se pone a jugar y en el segundo año se lesiona. Pues compañero mala suerte, lo que hay. Les intentamos eso, enseñar que si vienes aquí no te vamos a dar garantías de que seas un jugador profesional. De lo que te vamos a dar garantías es de que vas a entrar con un nivel y que vas a salir con otro mejor. Pero no te puedo garantizar que te vayas a convertir en profesional porque es que te estoy engañando y además me estoy condenando a mí mismo a fracasar.

Jeremy: Vale, ahora, por ejemplo, que hemos estado hablando del League of Legends. Cuando vosotros veis a un jugador que entra nuevo en la academia, vosotros decís: ese tío va a ser un AD Carry, o ese tío va a ser buen Support. Por ejemplo en el fútbol sabes si va a ser delantero o va a ser portero. ¿Podéis saber qué tipo de jugador va a ser?

Lucía: ¿Ya les clasificáis o...?

José: Te explico, te explico. Hay dos maneras de determinar. Porque, por ejemplo, algo muy importante en el mundo del deporte es la motivación que tú tengas. O sea, si a ti no te gusta ser defensa, a lo mejor eres un defensa genial, pero si no te gusta ser defensa, ni vas a entrenar para ser defensa ni vas a estar motivado para ser defensa, y aunque seas muy bueno, puede ser contraproducente. Entonces lo que hay que hacer es un estudio en primer lugar de sus motivaciones y luego en segundo lugar de sus características como jugador. Y a partir de ahí le puedes recomendar. A ver, si coinciden,

estupendo, tienes un tío que es un delantero nato, o en el caso del League of Legends, un AD Carry nato. "No, es que a mí me gustan los AD Carries."Y además es que ves que el tío farmea por minuto, tiene un posicionamiento que una de las cosas más importante de los AD Carries en team fighter de las mejores. Es que este tío es AD Carry. O esa, no cabe ninguna duda. Pero el problema es cuando, no es que este tío se posiciona estupendo, farmea por minuto pero cuando le dices, juégate un AD Carry te dice: "Pero qué dices, pero si eso me da asco, no quiero jugar eso. No quiero, me aburro". Y le pones a jugar y justo cuando le haces jugar un AD Carry por obligación empieza a desmadrarse, o se queja de todo y acaba siendo un problema para sus compañeros porque está amargado durante la partida. O sea, tienes que valorar las dos cosas. No es tan sencillo como una operación matemática en la que le dices a la gente a ver... esto, esto, esto, tu eres AD Carry. No es algo tan fácil. Tiene que darse las circunstancias de que, 1) por un lado, él quiera, 2), a él le motive, y 3), valga para.

Jeremy: ¿Y cómo de fácil es ver talento en un jugador simplemente viendo sus partidas?

José: Relativamente fácil. O sea, no es demasiado complicado porque, a ver. Lo que sí es verdad es que tienen que ser partidas competitivas o como mínimo de lo que es el SoloCube. No partidas normales con amigos. Quiero decir, SoloCube, estás compitiendo por subir en tu posición mundial y en competitivo estás jugándote un trofeo. Porque ahí se dan vamos a decir que se da todo el caldo de cultivo idóneo para evaluar al jugador. Lo primero, hay presión que muchos jugadores son buenísimos pero cuando los pones sentados en un escenario con un montón de gente delante se convierten en malísimos porque se ponen nerviosos y fallan absolutamente todo. Incluso he visto jugadores que el primer día que han entrenado de cara al público, las manos temblaban. Claro, pues no juegas igual. Lo primero es eso, la presión. Lo segundo, ¿contra qué nivel de rivales se enfrenta? Porque no es lo mismo enfrentarte contra jugadores nuevos y que el tío se salga, porque claro yo también me saldría, que contra niveles competitivos sobretodo o personas de SoloCube en Challenger como él. Si ves que el nivel es equiparable y que además funciona, vamos a decir. Perfecto. Luego también hay que evaluar la toma de decisiones durante la partida. Si ves que el tío es un loco que se tira contra todo que ve, y que va a matar a todo el mundo, te das cuenta de que no es un jugador de mente fría, pues no es lo más interesante. Ahí ya hay que valorar si es tan bueno que aun tirándose a loco acaba con todos, dices, ostras, me interesa. Pero si de vez en cuando toma una mala decisión y eso acarrea un problema para su equipo, tienes que valorar si es algo que se puede corregir o no. En líneas generales, es totalmente el campo de pruebas, es partidas de competición, partidas de SoloCube. Además, es que no tienes otro sitio donde verlos. Así que sí, es muy importante.

Jeremy: Vale, ahora, hablando de juegos específicos que tratáis aquí en

la academia en los que los jugadores sean peores o les cueste más mejorar...

José: Donde más cuesta mejorar es en League of Legends. League of Legends es prácticamente— estudiar League of Legends al completo es casi una carrera universitaria. Hablamos de 148 campeones, multiplicado por 5 habilidades: una pasiva, tres normales y un ultimate. 800 stats que tiene el videojuegos pasando por regeneración de vida, de maná, vida, escudos, armaduras, resistencia mágica —que además hay una fórmula específica para calcular, si tienes tanta resistencia mágica, cuánto daño reduces mágico. Hay otra fórmula para calcular, si tienes tanta armadura, cuánto daño reduces físico. Luego cada habilidad tiene escala con un porcentaje o de poder, de habilidad o de daño de ataque, tienes el robo de vida, la generación de vida, tienes 50.000 stats multiplicados por 141 campeones, cada uno con sus vidas, con sus armas... Que escalan encima sus stats de manera diferente en unos y otros según si sube nivel 2, 3, 4, 5, 6, 7 y 8. Añádele que esos stats también se aplican a los monstruos del propio mapa, añádele la estrategia dentro del propio mapa, la visión. O sea, es un carrera universitaria. O sea, realmente aprender a jugar al LOL te lleva, mínimo tres años. Bien, yo no conozco a nadie que se haya puesto a jugar al LOL y sepa todo el LOL en un año. A lo mejor si es muy muy muy bueno mecánicamente hablando— bueno, la palabra correcta es técnicamente hablando. Mecánica nos la hemos inventado un poco porque las mecánicas de los juegos, vosotros que estudiáis videojuegos sabéis que es algo un poco diferente. Técnicamente hablando, es decir, en ejecución puede llegar a ser diamante, pero llegar a ser diamante sin saber realmente todo del videojuego. Es un juego muy complejo. Luego Fortnite es un juego muy simple, que quizá ahí radica su éxito, es un juego en el que puedes jugar tres meses y en tres meses lo tienes todo más o menos controlado, porque hay creo que son 20 armas más o menos, en seguida las tienes clasificadas por colores, si son mejores o peores, creo que son gris, verde, azul, morado, naranja, creo recordar o algo así.

Jeremy: Sí.

José: El mapa más o menos es siempre el mismo, es cierto que siempre están cambiando cosas pero más o menos es siempre el mismo. Entonces claro, es mucho más sencillo. Entonces donde más cuesta mejorar es en el LOL, después del LOL en lo que damos aquí en clase en el CS:GO, y después en el resto, esos dos son quizás los dos más complicados.

Jeremy: Y ahora hablando de habilidades y cualidades de los jugadores, ¿cuáles son las que más cuestan mejorar?

José: A ver, por regla general, la que más cuesta mejorar... a ver... Diría que los primeros pasos de un jugador totalmente nuevo, lo que más cuesta mejorar es la actitud, cuando el jugador es totalmente nuevo. Pero cuando el jugador ya ha avanzado suficiente, es decir, ha cogido los niveles mínimos de coordinación óculo-manual para jugar, lo que más cuesta mejorar es esos últimos pasos hacia la perfección de la coordinación óculo-manual son los

que más cuestan. Porque de hecho esos últimos pasos son extremadamente complicados. La diferencia entre farmear, por ejemplo, el número estándar que se estipula del farmeo en League of Legends es 10 minions por minutos, conseguir el 11, el número 11 por minuto se puede conseguir pero cuesta una auténtica barbaridad porque ya no es solo cuestión de lo bien que lo haces, sino de dónde sacas el minion número 11, porque no existe, no viene en la oleada. O sea, cada 10 minutos vienen 10 minions en total, ¿de dónde sacas el 11? Pues ya tienes que saber en qué momento puedes ir a la jungla de tu compañero o del rival a conseguir el minion número 11 y sobre todo con el cuidado de no encontrarte el otro jungler y que te mate, en fin, esos niveles son muy complicados. Y ahí ya normalmente lo que haces es, con los jugadores más profesionales trabajas más la actitud, que si va muy mal en la partida no tiltee, que llamamos nosotros, que es que empiezan a venirse abajo y empiezan a cometer errores absurdos, porque en ese estado de frustración dejas de pensar lo que estás haciendo, sino que vas como en piloto automático y vienen los errores. Te centras más en la actitud. Al principio es muy difícil mejorar la actitud pero ya cuando hablas con jugadores profesionales es muy difícil mejorar la excelencia, vamos a decir, de ejecución.

Jeremy: Y¿qué es lo más difícil para que un jugador pase de ser un jugador bueno y destacado a poder llegar ser profesional? Ese último paso.

José: A ver yo creo que en ese último paso lo más importante son varios factores, o sea, no podría atribuirlo a uno solo. Yo creo que 1) tienes que tener suerte, es decir, que un equipo esté buscando alguien en tu posición y tú seas lo suficientemente bueno como para cubrirla y te den la oportunidad y ese día que te dan la oportunidad te salgan bien las cosas. Entonces ahí hay un factor suerte que se encadena. Y luego en segundo lugar diría que para dar el salto a profesional —a profesional de alto nivel te estoy hablando, a lo mejor una LSC europea, no a nivel español, que es más regional—, además, tienes que ser entre comillas un privilegiado desde el punto de vista del nivel mecánico —bueno, técnico— de jugar a videojuegos, o sea tienes que ser una persona que en general videojuego que coges, videojuego que destrozas. Esa gente normalmente, los chavales que yo he tenido, por ejemplo, Pepinero, es que es un tío que juego al que juega, juego en el que es el mejor. O sea, Pepinero para que te hagas una idea, lo bannearon del League of Legends tres semanas, se instaló el CS:GO sin haber jugado en su vida y en tres semanas era Global Elite en CS:GO que es el rango máximo, para que te hagas un idea. O sea, tienes que ser entre comillas un privilegiado. Lo mismo te digo, Pepinero entre partido y partido competitivo de League of Legends tenia instalado el Osur. ¿Lo conocéis?

Jeremy: Sí.

José: Bueno pues se ponía el Osur, no a dificultad máxima, no. Se descargaba canciones que eran todavía más difíciles que dificultad máxima y, entre partida y partida, se las hacía. Porque además son gente que normalmente

el estar quieto lo llevan bastante mal. Si de hecho algunos tienen diagnosticado el trastorno este de poli-hiperactividad desde pequeños. Muchos de ellos. Y claro, el ser hiperactivos en muchas ocasiones les sirve, porque al ser hiperactivo estás jugando y constantemente practicando, y al final de todo se traduce en más horas de juego, más perfección a la hora de jugar.

Jeremy: Esto ya a parte de ser una pregunta hacia ti como entrenador y profesor, también como jugador: ¿cuál crees que es la forma más eficiente de mejorar en un videojuego tanto, qué filosofía seguís o que técnica usáis para entrenar?

José: La filosofía más eficiente para mejorar en un juego, sino a cualquier cosa en la vida, es el feedback. Es decir, o lo haces tú ese feedback o que te lo haga alguien, pero constantemente tienes que recibir y analizar el feedback de cada cosa que hayas hecho: juegas una partida de League of Legends, un feedback, si no te lo ha hecho alguien, coge tu la partida y la ves, sacas el feedback y lo analizas ¿Qué puedo mejorar? Esto, esto, esto. ¿Qué he hecho aquí? Esto, esto, esto. ¿Cómo puedo hacerlo mejor? Esto, esto, esto. Entonces, ¿qué pasa? Muchas veces es verdad que es mucho trabajo jugar, analizar a la vez, entonces siempre lo ideal es tener un preparador o un entrenador que te ayude. ¿Por qué? Porque tú acabas la partida y tienes el feedback hecho. O sea, el feedback es este, esto es lo que has hecho mal en el minuto tal en el minuto cual, en el minuto nosecuantos. Por eso la figura del entrenador en los deportes electrónicos es tan importante. Es que es la clave, hasta en el deporte. En deporte, aunque parezca mentira, hay un entrenador de 100 metros. Muchos piensan: "¿Para qué quiero un entrenador de 100 si es correr como un zopenco 100 metros todo lo que puedas? ¿Para qué quiero un entrenador ahí?" Pues porque el entrenador graba en vídeo tu carrera y te dice: "Esta zancada podías haberlo hecho un centímetro más larga y hubieras recortado nosecuantas milésimas de segundo". Eso tú después de correr, coge el vídeo, analízalo, saca que esta zancada es más corta, esta zancada es más larga, es complicado. Bueno, te supone perder mucho tiempo. Entonces el feedback es, en todo, en el deporte y en los deportes electrónicos será prácticamente lo más importante. Es la filosofía clave para mejorar, y por supuesto no creerte que ya no puedes mejorar más.

Jeremy: Esto ya nos lo has comentado antes un poco, más o menos, pero, ¿usáis algún tipo de software especial para mejorar cosas específicamente o normalmente trabajáis directamente sobre el juego en sí?

José: La mayoría, el 95 por ciento del trabajo son los juegos. Además el juego también hay que decir que te trae una herramienta de práctica. El LOL tiene una herramienta en la que puedes poner una especie de espantapájaros a los que les puedes hacer cosas y no mueren nunca y puedes practicar con ellos e incluso tus habilidades no tienen *cooldown*. Entonces puedes trabajar muy bien en la herramienta de práctica. Pero ese 5 por ciento que resta es mucha información estadística, vamos a decir, a través de web, que es lo que

os he enseñado del OP.GG, de ver las partidas del rival y de *scouting*, que se suele llamar. Tanto para calentar como para mejorar las habilidades técnicas de los jugadores, el Aimbot que os explicaba antes, el mapa este especial de CS:GO o el League of Legends, la aplicación de coordinación óculo-manual de ir apretando teclas. Tanto como calentamiento como para cuando a lo mejor estás un poquito más flojo, porque los jugadores de deportes electrónicos tienen estados de forma como los futbolistas. A veces están mejor, a veces están peor. Entonces si están peor, pues a lo mejor ponerles a practicar esto les ayuda a recobrar, pero poco más. Me atrevería a decir que prácticamente nada más.

Jeremy: Ahora, hablando de controles: ¿normalmente qué tipo de dispositivos usáis en la academia?

José: Todos.

Jeremy: ¿Para todos los videojuegos?

José: Usamos Nintendo Switch, Xbox, PlayStation y ordenadores, y no tenemos dispositivos móviles para, por ejemplo, el Clash Royale porque obviamente la gente ya viene con su propio dispositivo móvil, no vamos a tener aquí 50 móviles para repartir. Pero lo utilizamos todo.

Jeremy: ¿Y notas que, por ejemplo, a la gente se le de mejor usar unos dispositivos u otros?

José: No. Cada uno está acostumbrado a usar lo que usa en casa. Lo que sí que es verdad es que todos se dan cuenta de que el ratón es infinitamente más preciso que el mando de la Play, de la Xbox o que el teléfono. Todos. En seguida. En cuanto cogen un ratón dicen: .Eh, es que no me gustan", pero en cuanto lo cogen 20 minutos porque les has indicado: "Hoy tienes que hacer 20 minutos de ratón, pruébalo", para apuntar no hay color. O sea, el ratón es infinitamente más preciso que el mando de la Play o que el mando de la Xbox. De hecho, en competición de Fortnite, que es un juego curioso porque se juntan en competición los de Xbox, los de PlayStation, y los de ordenador y los de móvil. Porque tienen de todo y todos van al mismo sitio a competir. Los de PlayStation y Xbox tienen un ratón y un teclado que conectan a la PlayStation y a la Xbox para competir porque es que sino con el mando se comen un turrón. No ganan, vamos, sería imposible que ganaran.

Jeremy: Cuando llega gente a la academia, ¿vosotros les soléis inculcar o recomendar unos controles específicos o les dejáis a ellos que se manejen como quieran?

José: Sí, normalmente la primera clase que damos de casi todos los videojuegos es presentación profesor alumnos y configuración del juego. Entonces, nosotros recomendamos, por ejemplo, en el caso del League of Legends nosotros sí recomendamos el estándar. El League of Legends, por ejemplo, sí está muy estandarizado que todos usamos más o menos lo mismo, pero por ejemplo en caso de CS:GO no es así. En CS:GO cada jugador tiene su mirilla, su CPIs colocado de una manera, la sensibilidad de una manera específica...

Entonces, en el propio programa este que os decía, en el Aimbot te sale las fotos de los jugadores profesionales y cuando disparas en su foto se te pone automáticamente su configuración y puedes probarla. Entonces lo que hacemos el primer día en CS:GO es decirle: "Pruébalas todas y decide cuál te gusta, y a partir de ahora esta es la que vas a usar siempre". Pero es que depende mucho del juego, unos sí tienen unos controles estándar y otros, no.

Jeremy: Vale, y esto lo hemos comentado antes un poco pero, por ejemplo, si llega un alumno que es zurdo y ¿le intentáis amoldar los controles un poco o que haga como un diestro o qué?

Lucía: Claro, ¿o le ponéis en la esquina como has dicho, o que intenten jugar como diestro o algo así?

José: Intentamos que juegue como zurdo siempre. Si es zurdo, que juegue como zurdo. Lo de la diestra y la siniestra es ya de otra época. Se puede jugar perfectamente siendo zurdo, hay jugadores profesionales que son zurdos y no han tenido ningún problema más allá de que es verdad que el zurdo por el diseño —no tanto de los ratones sino de los teclado— ocupa más espacio en la mesa que los diestros. Entonces, si tienen un compañero a los lados, al ocupar más espacio, pueden estar restando espacio a otras personas y eso desde el punto de vista competitivo no es interesante. Entonces lo que se suele hacer con los zurdos es, se les coloca en los laterales y se les añade una pequeña mesa supletoria por si necesitan un espacio extra. Es verdad que algunos zurdos han aprendido a base de que juegas con otra gente y se quejan de que les ocupan el espacio, han aprendido a reducir su espacio al mismo espacio que usa todo el mundo pero es cierto que normalmente los jugadores zurdos siempre ocupan unos centímetros de más, porque el teclado está preparado para estar al lado izquierdo y un zurdo se lo pone en el derecho, entonces para utilizar las teclas que se suelen utilizar tiene que desplazar más el teclado hacia fuera y eso es lo que termina molestando. Pero vamos normalmente el jugador zurdo aquí puede aprender siendo zurdo e incluso nos adaptamos a cualquier cosa que necesite. O sea, no nos suele parecer ningún problema. De hecho algunas veces se les recomienda que cambien las teclas por si es mejor para él. Pero si por ejemplo dice: "No es mejor para mí, estoy más cómodo así". A fin de cuentas lo que interesa es rendir, el rendimiento, es decir, de nada sirve que intentes jugar con la diestra si con la diestra eres malísimo y con la zurda eres buenísimo. Pues si con la zurda eres buenísimo, juega con la zurda. no es ningún problema, porque lo que importa es la ejecución.

Lucía: ¿También tenéis ratones para zurdos?

José: Tenemos ratones especiales para zurdos, correcto. Tenemos un ratón que la forma es totalmente simétrica y lo único que tiene que hacer es cambiar los botones. Si no me equivoco, los zurdos os ponéis el clic a la derecha-

Jeremy/Lucía: Sí.

José: Y el clic izquierdo, el que para mí es clic derecho, para vosotros es

clic izquierdo, entonces lo único que hay que hacer es cambiar la configuración y ya está. El ratón es exactamente igual para si lo usa un diestro como si lo usa un zurdo. Como este, por ejemplo [nos enseña un ratón hp].

Jeremy: Es simétrico, ¿no?

José: Correcto. Es totalmente simétrico.

Jeremy: Vale, ya estamos acabando.

José: De hecho este está aquí porque lo gastó el otro día un zurdo.

Lucía: Ya que estamos hablando del hardware que utilizáis, crees que importa mucho que... Entiendo que necesitas un buen PC que corra bien el juego, pero pantalla, ordenador... ¿Crees que es importante que sea muy, muy bueno para poder llegar a jugar bien a videojuegos?

José: No solo que corra bien el juego, excelente, además. Os explico, que el ordenador pueda mover bien el juego a, como mínimo, 60 fotogramas por segundo es fundamental. Si el ordenador no mueve el juego en 60 fotogramas por segundo todo el rato, incluso en los momentos de mayor exigencia de una partida, que hay 50.000 tíos peleando, hay luces por todos lados... Si ahí los fotogramas descienden de 60, el ordenador no sirve. Porque es clave poder identificar en todo momento qué está pasando, y si hay un salto, por pequeño que sea, y te has perdido medio segundo de pelea o de lo que sea, puede suponer la diferencia entre ganar o perder un campeonato del mundo y haber perdido 8 millones de dólares. Entonces es fundamental. Eso respecto a lo que hay dentro de la máquina. Respecto a lo que hay fuera: tener un ratón con muchos DPIs, que te permita una configuración lo más fina y precisa posible, también es importante. Sobre todo que sea lo más ergonómico posible, porque las manos después de muchas horas jugando, duelen. Lo mismo pasa con el teclado: tener un teclado que te transmita las sensaciones del videojuego de la mejor manera posible. Por ejemplo, probad estos teclados [Presiona una tecla de un teclado mecánico y se oyen clics]. ¿Oís cómo son las teclas?

Lucía: Sí.

José: Son teclas que te transmiten. Pruebas un teclado de membrana, que se llaman, y no hace ningún ruido. Tú cuando pulsas no sabes. Mira, esto es un teclado de membrana [Presiona teclas del teclado de membrana]. Yo no sé en qué momento realmente la tecla está de verdad pulsando. [Presiona teclas del teclado mecánico], Aquí sí, aquí llega un momento que hace clic.

Jeremy: ¿Qué switches usáis?

José: Pues estos realmente no lo sé, te lo tendría que decir el informático que es el que los ha decidido, pero normalmente utilizamos siempre teclado mecánico. Eso sí, de membrana jamás. El tipo de switch es el que no conozco. Porque claro en League of Legends no es tan importante pero, por ejemplo, en CS:GO es clave. Entonces, si quieres luego le preguntamos al informático y que os lo diga. Pero sobretodo lo importante, independientemente del switch, por lo que oigo de mis jugadores, es que la tecla te diga en qué momento

está pulsando. Porque fíjate que las teclas estas, las pulsas [presiona teclas del teclado mecánico], las puedes pulsar sin que estés pulsando, o sea ahora mismo la estoy pulsando pero no está haciendo nada, y llega un momento en el que hace clic. Esa es la clave. Que el clic te esté indicando: .ªhora estoy actuando", porque eso hace que seas mucho más preciso a la hora de jugar. Y luego algo que muy poca gente conoce es, los cascos son hiper importantes, porque por ejemplo en la mayoría de shooters los pasos del rival suenan diferentes si lo tienes arriba, abajo, a la izquierda o a la derecha. Está hecho así a posta para que los jugadores profesionales puedan extraer al máximo el rendimiento de la partida y saber si tienen un tío arriba o abajo. De hecho muchas veces las grandes jugadas son porque él está escuchando, escucha que tiene uno arriba y dispara al suelo— bueno al techo, que es el suelo de la siguiente planta— y acaba matando a alguien. Y toda la gente: .ªla, ¿cómo lo ha hecho? ¿Cómo lo ha sabido?"Lo sabe porque el tío está escuchando. Entonces, que los cascos, 1) insonoricen mucho y 2) sean muy nítidos, la emisión de sonido sea muy nítida, es fundamental. Y la otra cosa que es total y absolutamente fundamental es que el monitor sea de 144Hz de refresco. Porque, por ejemplo, en League of Legends, Lee Sin, que es un de los campeones que más se ha jugado a lo largo de la historia en los mundiales, justo antes de lanzarte una Q, que es una habilidad que lanza como un proyectil y que luego se lanza él, hace un breve movimiento con el hombro que en un monitor que no sea de 144 Hz no percibes. Si no percibes ese movimiento, la Q te da porque en el momento en el que sabes que te están lanzando la Q es tarde para esquivarla. En un monitor de 144Hz, si estás atento, lo ves y te da tiempo a esquivarlo. Lo mismo pasa también en CS:GO. En el CS:GO si tú te pones a dar vueltas a lo loco, no ves en un monitor normal. Ves todo borroso, no ves nada. En un monitor de 144Hz, que son estos por ejemplo, tú te pones a dar vueltas a lo loco y, aunque estés dando vueltas, lo ves todo nítido. No tienes ningún problema. Es decir, un jugador de CSGo de hecho, si no es un monitor de 144Hz no puede ni jugar. Un profesional te dice que no juega, directamente. Y luego, en Fortnite el monitor de 144Hz es importante también por eso, por lo nítido mientras te mueves, pero además en Fortnite es muy importante que tengan sistemas de contraste específicos. Fortnite es un juego con tantos colorines, que hace que sea difícil identificar a un tío que se asoma por algún lado. Porque son colorines y colorines, no identificas.. Pero si tienes sistemas de contraste especiales, sí que es más sencillo identificarlo, y la diferencia entre identificar a un tío a una distancia de 300 metros o identificarlo cuando ya lo tienes a 200, es la diferencia entre vivir o morir, que en un torneo supone mucho dinero. Ya te digo, estos ordenadores todos llevan sistema de contraste especial, y todos llevan 144Hz. De hecho, para eso no puedes conectarlos por HDMI, no sé si lo sabéis, se conectan por DisplayPort. Desde el punto de vista técnico se exprime todo lo posible. De hecho, con la conexión a Internet tenemos el mismo problema: si tu conexión a internet es mala, en casa lo

tienes complicado, porque el ping es fundamental. Normalmente jugar por encima —en España, te hablo—, en España jugar por encima de 50 de ping ya vas mal. Y, por ejemplo, en Alemania jugar por encima de 30 de ping,ya vas mal. Y en Corea, jugar por encima de 10 de ping vas mal. En Estados Unidos no conozco exactamente el ping con el que os movéis pero no debe ser mal ping tampoco porque tenéis los servidores ahí mismo en Estados Unidos, o sea que... Esas son todas las especificaciones técnicas que solemos utilizar.

Jeremy: Vale, ya queda poco.

José: Tranquilo, tranquilo, no me molestáis. He cogido la mañana para atenderos, no os preocupéis.

Lucía: Gracias.

Jeremy: Tanto en equipos profesionales como aquí, en la academia, aspectos psicológicos y anímicos de los jugadores, ¿hacéis algo al respecto? ¿Cómo los tratáis? ¿La importancia que les dais?

José: Los trata sobretodo el psicólogo y no es mi campo. Te puedo decir lo que sé de hablar con él y de haber trabajado con varios a lo largo del tiempo. Como, por ejemplo, os recomiendo que habléis con una chica que se llama Anxane. No sé si os suena. Si queréis os doy incluso su contacto en redes sociales por si queréis preguntarle algo o mandarle un correo con preguntas, sería interesante. Pero vamos, nosotros trabajamos sobretodo con ellos el "tilt". No sé si sabéis lo que es.

Jeremy: Sí.

José: Que básicamente es una sensación de... Para que lo entendáis rápido, ¿practicáis algún tipo de deporte?

Jeremy: Sí, yo juego al béisbol.

José: Béisbol.

Lucía: Natación.

José: Natación, vale. ¿Nunca os ha pasado que llegáis a un sitio, por ejemplo, en natación dices: "Hoy voy a hacerlo que te cagas de bien, lo voy a hacer muy bien hoy, estoy segura de que me va a salir", o vas a una competición y dices: "Voy a ganarz todavía no has empezado pero tú dices: "Voy a ganarz luego estás en la competición y ganas? Y te sale. Y dices: "Joe, es que soy buenísimo". ¿Sabes? Vale, pues el *tilt* es todo lo contrario, es: "Hoy la voy a liar. Hoy, no. Hoy no voy a funcionar. Hoy no voy a funcionar. Hoy no voy a funcionar". Y aunque el otro sea peor que tú, no funcionas, y pierdes. Eso es un estado que sucede en el deporte y que intentamos evitar en el deporte. ¿Cómo? Pues lo primero es un psicólogo intenta hacer que ese estado que aparece siempre en todos los deportistas, aparezca 1) lo más tarde posible, y 2) lo mas suave posible. O sea, que no sea un *tilt* muy fuerte. Y luego tenemos el *choke*, que solemos llamar también, que no sé qué significará en inglés, la verdad, porque no me he parado nunca a estudiarlo. Pero el *choke*, ¿qué es? Sería el equivalente al miedo escénico. Llegas a un evento importante, te puede la presión y haces un desastre.

Lucía: Es atragantarse, literalmente.

José: Tal cual. Haces un desastre. A lo mejor eres buenísimo, haces un torneo estupendo pero llega la final y dices: .Es que este equipo que durante todo el año parecía que iba a comerse el mundo ha llegado a la final, ¡se han convertido en cinco malos!"Y suelen decirte los jugadores: "He chokeado. He Chokeado."Lo que pasa es que, sí, miedo escénico o como lo que has llamado . Entonces incluso dentro del propio miedo identificamos dos, que es: el miedo a perder, que hace que no quieras jugar. Como no quieres perder, normalmente lo que no quieres es competir o no quieres jugar, no quieres exponerte. Y también existe el miedo a ganar.

Lucía: ¿Cómo puede ser eso?

José: Que ese se oye menos. El miedo a ganar es: eres mejor que el otro, pero el otro también es bastante bueno, y piensas: "Yo no puedo ganar a este tío, este tío es buenísimo, no puedo ganarle". Y a lo mejor eres mejor, pero: .Es que este tío es muy bueno y lo respeto mucho, y no sé". Y pierdes. Y pierdes, pero no pierdes porque seas peor que el otro, pierdes porque tienes miedo a ganarle, porque no te ves capaz de ser mejor que el otro aunque lo seas ya. Entonces hay que trabajar las dos cosas, normalmente casi siempre nos movemos en ese en ese círculo de cosas. Pero seguro que si habláis con Aranxa os dirá alguna más, porque ese sí que es su campo, yo os hablo, no tanto desde el punto de vista de haberlo estudiado, sino desde el punto de vista de la experiencia, lo que hemos trabajado. Sobretodo el miedo a ganar pasa mucho, sobretodo un equipo que, imagínate, ha ganado en los dos últimos campeonatos, y al que ya has ganado fase de grupos tres veces. Llega la final y se combinan el choke con el miedo a ganar y dices: .Es que estoy muy nervioso y estos ya han ganado otros campeonatos, no me veo capaz de ganarles en una final". Pero si ya les has ganado tres veces en la fase de grupos, esto es un partido más, ¿qué más te da? Pues si la mente te dice que no, es que no. Es así. Es que no.

Lucía: Vosotros en la academia, ¿tenéis psicólogo?

José: Tenemos psicólogo. Tenemos psicólogo, tenemos nutricionista y tenemos preparador físico.

Lucía: Tenéis aquí de todo.

José: Sí.

Lucía: ¿Cualquier alumno tiene acceso a ellos?

José: No. Solo porque, a ver, no podemos. Entre comillas, solo tenemos un psicólogo, solo tenemos un nutricionista y solo tenemos un preparador físico. No podemos cargarlos de trabajo. Entonces, ellos lo que hacen es: 1) Nos pasan ciertos test que nosotros pasamos habitualmente en clase, normalmente al principio de clase. Lo primero que hacen es rellenar los tests, él los evalúa y él decide: "Pues este muchacho me gustaría tener una sesión con él". Se llama al padre por teléfono: .Oiga, ¿podría traer a su hijo antes? No, no puedo traerlo antes". "Bueno, pues, va a perder una clase, si a us-

ted le parece bien, que queremos que haga una sesión con nuestro psicólogo, porque nuestro psicólogo quiere trabajar con él, nosequé". Eso es una de las cosas que puede suceder. Puede suceder que el propio profesor diga: "Tengo un problema con este muchacho, está toda la clase gritando, o se frustra mucho, creo que debería de tener una sesión con el psicólogo". Y entonces pues volvemos al proceso anterior, llamamos al padre, etcétera, etcétera, etcétera. Normalmente, siempre es una de esas dos cosas, y rara vez —aunque alguna vez ha pasado, no en esta academia sino que me ha pasado como docente— el padre es el que te pide una sesión con el psicólogo o con el nutricionista o con lo que sea. Por ejemplo, tenemos dos alumnos que tienen sobrepeso y los padres nos dijeron: "¿Tenéis un nutricionista?Sí". "Podríais intentar hacer que mi hijo coma mejor y que me sea más fácil que coma verduras, etcétera". "Sí, claro. Vamos a mandarlo a hablar con el nutricionista". Hablan con el nutricionista, y tuvieron su sesión con el nutricionista y estamos intentando mejorar en ese aspecto. No sabemos si lo conseguiremos pero por lo menos la intención es conseguirlo.

Jeremy: La última pregunta que ya nos has respondido antes, pero algún tipo de software que crees que te podría ayudar, como nos has comentado, lo del mapa, por ejemplo, ahí nos has comentado como una versión mejorada.

José: Una versión actualizada y mejorada del mapa para la estrategias del League of Legends sería útil.

Jeremy: Que cosas que, por ejemplo, hemos visto en este mapa, te faltarían.

José: Me faltaría que, cuando yo muevo un personaje, deje un rastro. Me faltaría que pudiera añadir minions, es decir, lo bichillos estos que salen en el LOL. Me faltaría que Dragón, Nashor y los campamentos de la jungla, etcétera, pudiera seleccionarlos y me diera información sobre los campamentos, e incluso eliminarlos. Y ya lo que sería increíble sería que arriba tuviera una especie de barra de tiempo que fuera desde el minuto 1 hasta el minuto 60 y que, conforme yo lo fuera moviendo— por ejemplo, sabéis que llega un momento en el que no sale un dragón sino que sale el Dragon Anciano. Pues que si yo estoy en ese minuto de la partida a partir del minuto veintipico, creo que es, o treinta no lo recuerdo ahora— me lo cambie automáticamente, se cambie uno por otro para que pueda trabajar. El Heraldo está hasta cierto minuto, luego desaparece y aparece el Nashor, pues que yo pueda con esa barra decidir en qué minuto de la partida estoy. Poder romper la torres que, por ejemplo, en esto no se rompen, siempre están puestas. Decir pues esta torre la quiero rota, esta también esta también y esta también. Eso sería súper útil, de hecho si lo hacéis me interesa mucho. Y el que os he dicho de seguimiento de mis jugadores y que me haga un resumen de lo que han hecho durante el mes también me parece otra herramienta súper necesaria y muy buena. Las dos herramientas creo que hacen mucha falta en League of Legends. Luego, para cualquier shooter cualquier herramienta que te haga

practicar la puntería puede ser útil. Porque en los shooters, a fin de cuentas, lo más importante es practicar la puntería, pero no solo la puntería en parado, sino también puntería en movimiento, puntería en salto, puntería...

Jeremy: También nos has comentado para practicar el "last hit".

José: Correcto. El juego de practicar el "last hit", que sería el equivalente de practicar la puntería del CS:GO y todos los shooters, pero al LOL, que es pues una serie de bichos que vayan perdiendo vida y que tengas que darles el golpe justo antes de que mueran y te cuenten una serie de puntuaciones. Sabéis que una de las cosas más difíciles del LOL es "hitear" bajo torre. Lo que pasa es que el tema del "last hit"— hacer una herramienta para el "last hit" te digo cual es mi problema: la contrapartida es que ya podemos practicarlo en el LOL porque ha salido la herramienta de práctica que tú lo que puedes hacer es activar la torre que necesites, parar a tus minions con unos espantapájaros y que se vayan acumulando ahí, y vienen los minions de la otra oleada directamente contra tu torre y puedes "last hitear" bajo tu torre que era lo que nos faltaba. Incluso puedes "last hitear" en la línea normal y corriente sin rival pues, más o menos, realmente la herramienta para practicar el "last hit"—

Jeremy: Eso está cubierto, ¿no?

José: Es quizá lo menos necesario. Si la haces y, de alguna manera, puedes trabajar algo que no se puede trabajar mediante el juego, genial. Pero es más complicado. Quizá lo que aportaría esa herramienta es inmediatez, porque para poder practicar el "last hit" tienes que preparar la partida antes. O sea, entras en una personalizada, quitas todos los bots, activas los minions, bloqueas el paso de los tuyos. En fin, pierdes bastante tiempo hasta que realmente puedes empezar a practicar pero cuando te pones a practicar realmente es bastante eficiente. Pero sobre todo la herramienta esta del mapa, pero con lo que te digo, poder quitar y poner torres, poder avanzar en la línea de tiempo. Eso seria una pasada. Que los personajes cuando yo los muevo dejen un rastro de por dónde los he ido moviendo para saber la ruta que han ido siguiendo. Eso sería una auténtica pasada. Y lo del seguimiento. Yo creo que esas dos.

Jeremy: En la del seguimiento, cuando quieres el resumen de la información, ¿qué es lo que más te importa que esté en este resumen?

José: Victorias, derrotas y porcentaje del Win-Ratio. Luego ese Win-Ratio desglosado y qué campeones ha jugado y cuánto Win-Ratio tiene con cada uno de ellos. Los valores de oro por partida de media, minions por partida de media, kills por partida de media, veces que se muere de media por partida, wards que pone de media por partida. Es decir, sobre todo todas las medias posibles por partido me interesan. Porque es a fin de cuentas lo que yo quiero trabajar con ellos, es decir, si tu media de kills es muy baja y eres el que se supone Asesino, pues tío como Asesino no lo estas haciendo bien, hay que mejorar eso, hay ser un poco más valiente. O sea, son datos

que me pueden servir para ver si estás cumpliendo tu función. Si eres el AD Carry, tío tienes que farmear muy bien y tienes que hacer grandes cantidades de daño. El daño medio por partida pues es importante, o sea, si tu acabas todas las partidas con mil de daño por partida cuando un AD Carry tiene que acabar con mínimo 15-20 mil pues tío no estás jugando bien de AD Carry. ¿Cuántas veces te has muerto? Normalmente, un jugador profesional no debería morirse más de 5 veces por partida de media, nunca. De hecho, 5 ya es cuestionable. Si me sale que la media es 6 estás muriéndote mucho, eres demasiado arriesgado— risky. O sea, eres demasiado loco. Pues básicamente esa herramienta también es muy útil. Yo, mi opinión, si queréis que os dé mi opinión sincera de las dos más útiles, tanto la del mapa como la del manejo de los datos de tus jugadores creo que son las dos más interesantes.

Lucía: Y una última pregunta: has dicho, por ejemplo, que el LOL es el más difícil—

José: A ver, hay juegos más difíciles que el LOL, pero no los damos en la academia

Lucía: Vale. Que puedes tardar como tres años en sabértelo todo. Solamente lleváis seis meses con la academia, pero ¿ya veis una progresión en los alumnos que tenéis?

José: A ver, ves una mejoría en cada clase incluso, porque aprenden cosas que no saben. El jugador medio de League of Legends desconoce el 40 por ciento del League of Legends, y sigue jugando al League of Legends, de hecho, y sigue aprendiendo pero muy poquito. A lo mejor aprenden un 0,2 cada día. Claro, llega un momento en el que, para aprender los conceptos avanzados del juego, necesitas a gente que ha estudiado los conceptos avanzados del juego. Mucha gente no sabe lo que es un Elo Boost en League of Legends o cómo se controlan las oleadas de los minions. Se creen que lo importante es pelear, que lo importante es yo gano porque peleo con el otro y le gano y entonces gano la partida. Luego vienen los problemas de cerrar partida: cuál es el orden prioritario de tirar las torres —si lo puedes elegir, claro, si no puedes elegir, pues la que puedas tirar, pero si lo puedes elegir porque vas por delante hay un orden que está demostrado que es mejor para ganar una partida. En qué momento ir a por un objetivo, cómo "guardear" una zona para tener todos los flancos cubiertos, cuáles son los "matches" entre los campeones, si este es más fuerte que este, en qué momento —porque además puede ser este más fuerte que este en nivel 1 y vas subiendo de nivel —2, 3, 4—, pero en el 5 este de repente ahora es más fuerte que tú y tienes que no pelear con él hasta el 6, 7, 8, 9, que tú vuelves a ser más fuerte que él. Claro, son tantas cosas las que tiene el League of Legends que es normal que el usuario medio que juega al League of Legends, no nos olvidemos, normalmente para pasarlo bien con los amigos, todas esas cosas 1) le dan igual y 2) no se molestan en buscarlas y 3) tampoco tiene nadie que se las cuente. Entonces siempre ves mejoría cuando viene aquí la gente. Además solo cuando prueban un monitor de 144Hz ya

notan la diferencia, dicen: .Anda, yo no sabía que existían estos monitores". Sí, señor, existen estos monitores.

Lucía: Ya, lo deben de flipar.

José:. Lo mismo pasa en el LOL. También, cuando mueves la cámara para mover algo, hay un momento desde que la mueves hasta que paras, que lo ves todo borroso, que no ves todo perfectamente nítido. En los monitores de 144Hz todo el transcurso se ve perfecto, independientemente de la velocidad a la que se mueva, porque la tasa de refresco es muy rápida. Entonces realmente tú estás moviendo la cámara de aquí a aquí y realmente estás haciendo muchos refrescos, pero cuando tienes un monitor de 60 pues es que estas haciendo menos de la mitad. Entonces, ¿cómo soluciona eso la pantalla? Con una mezcla de la posición uno y la posición dos y, como es un salto muy grande, encima pues le da una sensación a tus ojos de borrosidad y no puedes percibir bien todo lo que estás viendo. En el LOL, ya digo, es menos importante, pero en el CS:GO no puedes jugar sin un monitor de 144Hz.

Jeremy: ¿Algo más?

Lucía: A mí no se me ocurre nada más.

Jeremy: Pues ya estaría.

D.2. Segunda entrevista

D.2.1. Datos

Fecha: 3 de septiembre de 2019

Nombre: José de Matías - Director de The Global ESports Academy / Entrenador de League of Legends

D.2.2. Transcripción

Lucía: En la primera entrevista que te hicimos lo que queríamos ver era dónde nos podíamos meter en el tema e-Sports, y nos diste la idea de hacer herramientas dirigidas a entrenadores.

José: Muy bien.

Lucía: Entonces lo que se nos ocurrió, un poco por tu aportación, fue hacer una pizarra interactiva para el *League of Legends* que estuviese más completa de lo que pudieseis tener ahora mismo.

José: Ah, muy bien.

Lucía: Entonces hemos hecho una fase de investigación, hemos visto las herramientas que hay en el mercado, y con eso más tus aportaciones hemos creado esto, que está en beta.

José: Lo sé, lo sé, me lo imagino. Pero está muy bien.

Lucía: Entonces ahora te explicamos cómo va y lo que queremos de ti es que nos des tu opinión y también te haremos un par de preguntas un poco más específicas. Jeremy lo va a manejar.

[Abrimos dos instancias de la aplicación. Ambas están en el menú principal.]

José: Uno va a ser el cliente, bueno, el servidor, mejor dicho.

Jeremy: La idea es que el profesor o entrenador haría de servidor y cada uno de los alumnos o miembros del equipo se añadiría como cliente.

José: Vale, ¿y se añadiría automáticamente?

Lucía: Sí.

[Señalamos la lista de clientes a la derecha de la pantalla del servidor]

Lucía: Y aquí tienes la lista de los alumnos que se han unido.

José: Vale, ¿y entiendo que se añaden a las máquinas que están en red?

Jeremy: En red.

Lucía: En fases futuras, si seguimos con la implementación, nos gustaría hacer ya sí más pruebas en red porque ahora mismo lo estamos haciendo en el mismo ordenador, pero lo hemos hecho con Unity y tenemos la esperanza de que funcione. Entonces, las pantallas están coordinadas pero la interfaz es individual. Ahora, por ejemplo, si te pones sobre este personaje, sobre este NPC [Ponemos el ratón sobre un NPC de la jungla de la pantalla del cliente y aparece su información], te salen sus estadísticas, pero a este no le salen, le

sale solamente al único cliente. Entonces, ahora mismo por ejemplo, el cliente realmente, los alumnos no pueden hacer nada más que ver las *poptags*, las estadísticas y todo eso. No pueden hacer nada hasta que el profesor les de permiso.

José: ¿El profesor qué puede hacer?

Lucía: El profesor puede hacer más o menos de todo. Puede por ejemplo cambiar las torres. [Clicamos sobre una de las torres y su imagen cambia de no destruida a destruida]

José: De destruidas a no destruidas.

Lucía: Lo mismo pasa con los inhibidores.

[Clicamos sobre uno de los inhibidores y su imagen cambia de no destruido a destruido. Se actualiza en ambas pantallas]

Lucía: También puede cambiar los NPCs de la jungla, los puedes hacer desaparecer.

[Clicamos sobre uno de las criaturas de la jungla y su imagen cambia de estar presente a no estar. Se actualiza en ambas pantallas]

[Clicamos sobre el Barón Nashor y su imagen cambia de estar presente a no estar. Se actualiza en ambas pantallas]

José: Puedes quitar el Nashor.

Lucía: Sí.

[Clicamos varias veces sobre el dragón y su imagen cambia por los cuatro diferentes dragones y desaparecer. Se actualiza en ambas pantallas]

José: Dragón lo mismo.

Lucía: Puedes cambiarlos según lo que sea.

José: Vale.

Jeremy: Puedes hacer zoom, meterte hacia dentro y se actualiza.

[Hacemos zoom dentro del mapa. Se actualiza en ambas pantallas]

José: Muy bien.

[Nos movemos por el mapa utilizando las flechas del teclado. Se actualiza en ambas pantallas]

Jeremy: Y con las flechas puedes moverte.

José: Por si quieres entrenar en un sitio más específico y se coordina para que todos los alumnos estén a lo tuyo.

José: Vale. Y veo que lo del fondo [el mapa] es una imagen.

Jeremy: Sí.

José: ¿Plana?

Lucía: Sí.

José: Yo lo que hice cuando hice la mía, por si os sirve, para que tenga una sensación mucho más realista, por si queréis hacerlo, Unity tiene un gameObject que es *Terrain.* terreno. Entonces, coges un *terrain*, le pones una imagen y puedes levantar las paredes tú. No hace falta que las levante mucho. Un poco. Pero solo el efecto de sombra que te produce parece que

estás de verdad en la grieta y queda bastante mejor. Porque se nota que es plana. Pero de todas maneras este es un tema de diseño, que es una tontería. Solo os lo digo.

Lucía: No, si temas estéticos también nos interesan. Cualquier cosa que nos ayude a que se entienda mejor el mapa o lo que sea. Entonces ahora, puedes añadir personajes y cosas que te interesen. Por ejemplo puedes añadir los minions, el que te interese.

[Abrimos el menú de añadir, seleccionamos el equipo rojo de los minions y se abre el submenú de los minions. Seleccionamos el minion melée rojo. Aparece en ambas pantallas]

José: Meleé del lado rojo, Meleé del lado azul.

[Arrastramos el esbirro por el mapa. Se actualiza en ambas pantallas]

José: Puedes arrastrarlo.

Lucía: Puedes arrastrarlo. Y si te interesa, por ejemplo, puedes añadir un campeón, el que tú quieras.

[Abrimos el menú de añadir, seleccionamos el equipo rojo de los campeones y se abre el submenú de los campeones. Usamos la rueda del ratón para bajar por el submenú]

Lucía: Puedes hacer *scroll* por el menú. Escoges el que quieras.

[Seleccionamos un campeón aleatorio. Aparece en ambas pantallas]

Lucía: Están un poco escalados, pero bueno.

José: Vale.

Lucía: Y sí, lo puedes mover. Si quieres que te aparezcan las rutas.

[Clicamos el campeón con el botón derecho del ratón y aparece su menú de edición]

Lucía: Le das a enseñar ruta y entonces ya la dibuja. Como defecto, todas salen blancas. Esto se soluciona.

[Clicamos el campeón con el botón derecho del ratón y aparece su menú de edición]

Lucía: Le das a cambiar color y le pones el color que quieras y se cambiar también es todos.

[Seleccionamos la opción de cambiar de color, se abre la interfaz de los colores, seleccionamos el rojo. Se actualiza el color de la ruta en ambas pantallas]

José: Vale.

Lucía: Los campeones también se diferencian por el borde que les rodea para que se vea de qué equipo es.

José: Borde rojo, borde azul.

Lucía: Después, por ejemplo.

[Clicamos en la lista de clientes al cliente, su imagen pasa de rojo a verde]

Lucía: Les puedes dar permiso al cliente.

José: Vale, permiso para que ellos pongan cosas.

Lucía: Y para que lo muevan y tal. Por si quieres hacer la clase más interactiva y quieres preguntarle cualquier cosa a alguien individual, lo puedes hacer.

José: Sí, o decirle: .Es que deberías moverlo...A ver, a qué te refieres, te doy permiso, muévelo."Vale.

[Clicamos en la lista de clientes al cliente, su imagen pasa de verde a rojo]

Lucía: Y se lo quitas. Así, sin más.

José: Perfecto.

Lucía: Y después, por ejemplo, también puedes guardar "pantallazo" de lo que hay en la pantalla ahora mismo.

José: Hacer un *screenshot*, ¿no?

Lucía: Sí. Para si después las estrategias quieres que los chavales se las estudien o, hacerlas desde casa y luego enseñar o cualquier cosa.

José: Algo importante. ¿Puedes poner wards?

Lucía: Si.

José: Vale.

[Ponemos un ward en el mapa]

José: Vale perfecto.

José: Otra cosa importante. Necesitaría que los wards marcaran el radio de visión que dan. No se si visteis la que hice yo. No se si os la llegue a enseñar.

Lucía: La abriste pero no la recordamos.

José: Venga te la enseño rápido. Esta hecha en Unity también. ¿Vale? Venga.

[Abre su aplicación (la que utiliza en la academia)]

José: Es que, eso es de las cosas mas importantes, lo del rango de visión. Bueno escribe algo ahí, lo que sea.

[Jeremy escribe algo para iniciar sesión en su aplicación]

José: El rango de visión es importante porque si yo pongo un ward quiero saber mas o menos donde me van a ver. Esto es el rango de visión de los wards antiguos, ahora dan un poquito más de rango. Entonces eso te da mucho juego porque tu ponías un ward y sabias realmente si te iba a cubrir toda una entrada o, si ponen el ward típico de aquí pues, si nos pegamos a la pared podemos evitarlo y cosas de ese estilo que eran importantes. Entonces si vosotros le añadís a los wards ese tipo de "circulito" alrededor, que más o menos te marque la distancia, es muy importante.

Lucía: Vale.

José: Por lo demás de momento lo que veo esta muy bien, Es verdad que los campeones te salen enormes y habría que hacerlos mas pequeños, pero está muy bien.

[Risas]

Lucía: los campeones si te acercas o cualquier cosa, se escalan para que

no te ocupen toda la pantalla.

José: [Afirma].

Lucía: Aparte puedes hacer "reset" del mapa, se borra todo.

José: Se borra todo y vuelves a empezar. ¿No?

Lucía: Si. Entonces puedes volver a añadir cualquier cosa. Después también tienes aquí una barra del tiempo que sirve para poder actualizar un poco en el mapa estadísticas o, que este NPC se cambien. Entonces, le das por ejemplo... [mientras cambia del Herando De La Grieta al Barón Nashor].

Jeremy: Le das al minuto ahí [señalando la barra de tiempo].

José: ¿Son minutos?

Lucía: Sí.

José: Sale el Nashor. ¿no? [Cuando llega al minuto veinte en la barra de tiempo].

Lucía y Jeremy: Sí.

Lucía: Tenemos aquí hasta el treinta y cinto. [Cuando llega al minuto treinta y cinco en la barra de tiempo].

José: El dragón anciano.

Lucía: Y también se van cambiado sus estadísticas.

José: Las estadísticas de los bicho de la jungla.

Jeremy: Las estadísticas hemos puesto los puntos de vida que tienen porque no sabíamos muy bien que estadísticas serias las más importantes.

José: Las más importantes son la vida y el daño. Porque las otras creo que son fijas y no hace falta que las programes ni nada, que serían el tiempo que tardan el salir, que es un tiempo fijo. El dragón creo que es cada cuatro minutos o cada cinco, no recuerdo ahora. El Nashor pues cada cuatro, cinco o seis, no recuerdo ahora.

Lucía: ¿Algo más? [Pregunta a Jeremy].

José: Vale queréis que os de *feedback* entiendo. ¿No?

Lucía: Sí.

José: Primero lo de los wards, muy importante. Intentad que se pueda ver el margen o rango de visión del ward, que es vital. Aunque no pongáis el icono, aunque solo salga el circulo me valdría, porque es lo mas útil. A mi saber que tienen un ward en un determinado sitio no me sirve de tanto como saber hasta donde les va a dar visión. ¿Vale? Luego los campeones obviamente mas pequeños. Ahora son muy grandes. Como tema de diseño, el mapa se me queda demasiado plano, si lo pudierais hacer en 3D, es que no cuesta nada en realidad. Con el tema del terreno, es que lo haces en un momento. Es coger el ratón y pintarrajear por encima de las "esquinitas". Es que le da otro aire, sobre todo cuando haces zoom, no parece tan plano. Yo aquí ahora hago *zoom* y noto que es una imagen, aparte tengo que hacer mucho *zoom*, sin embargo cuando yo lo pongo con el terreno puedo no hacer tanto y queda más natural. Aunque parezca una tontería pero, el hecho de

que todo tenga el mínimo relieve le da otro aire. Cosas como estas [mientras muestra el *terrain* de su aplicación] dan, la sensación de que el terreno esta levantado de verdad. Eso visualmente me parece mejor. Luego un botón para solo quitan los wards de un equipo. Quitar los wards del equipo rojo para volver a ponerlos y quitar los wards del equipo azul para volver a ponerlos. Vale, ¿por qué? Porque entiendo que este wards lo podre quita (Quita un ward del mapa). Pero si he puesto cuarenta wards de una estrategia del equipo rival y tengo que quitarlos uno a uno es una liada y, si quiero conservar los del otro equipo y le doy a "reset" tengo que volver a empezar. Entonces quitar solo los de un equipo es importante. También es un muy cómodo. Por lo demás, la aplicación esta muy bien. Le faltan detalles de diseño, más bonito, más feo, unas cosas u otras, los botones los haría más bonitos y cosas así. Pero a mi por lo menos, personalmente me parece que esta muy bien.

Jeremy: Una cosa que nos comentaste es, que te parecía interesante la idea de poder añadir una oleada (de mininions) de golpe. entonces hemos hecho que tu puedas añadir toda la oleada de golpe.

[Creamos una oleade de minions]

José: Ah, muy bien.

Jeremy: No hace falta añadir minions sueltos.

José: Ah, muy bien.

Lucía: De vez en cuando tiene que aparecer un super-minion pero se puede añadir a mano.

José: Vale, estupendo a mi me parece genial. A ver, puntos de mejora son sobre todo de diseño. Mejorar los botones, quizás mejorar los minions, no son muy intuitivos cuando tienes seis cuadrados ahí puestos. Habría que intentar conseguir el *render* del minion aun que sea el png sin fondo. Te va a quedar más o menos lo más realista posible para, que le de otro aire a la aplicación. Con eso, con los wards y con los campeones, suficiente. Yo tengo una versión en la que cuando ponías el campeón se ponía el campeón en modelo en 3d, cuando Riot te lo facilitaba. ahora ya están más cerrados que cerrado. Pero en general me parece que esta muy bien. ¿Hasta cuantas personas soporta que se conecten a la red?

Jeremy: No lo hemos llegado a testear pero deberían ser todas las que te permita Unity.

Lucía: Si hasta que pete.

José: Normalmente por encima de veinte petaba creo recordar.

Lucía: Un equipo, que es para lo que lo teníamos pensado o una clase pequeña...

José: que son 8 o así sobra. Yo he llegado a tener veintiséis conectados o cosas así y ya empezaba a dar problemas.

[En este momento empezamos a preguntar las preguntas que traíamos preparadas]

Lucía: Vale, por ejemplo te queríamos preguntar sobre los controles de

la aplicación en si, que los hemos hecho sobretodo para que se pueda manejar casi todo con el ratón para que, no haya barras de herramientas y cosas así.

José: Cuanto más gastes el ratón y menos las teclas mejor.

Lucía: Lo único que haces con las teclas es...

José: Mover la cámara.

Lucía: Entonces, ¿eso te parece intuitivo?

José: A mi personalmente me gusta más hacer clic derecho y arrastrar, pero es algo ya más personal que otra cosa. En el juego como nos movemos con los bordes, cuando mueves el ratón al borde el juego se desplaza. Que yo recuerde además te puedo decir como se hace eso. Creo que tienes que poner en Unity un comando que hace que el *mouse* este restringido a la ventana y ahí te marca la diferencia de cuando el *mouse* esta fuera o dentro. Entonces sabes si el ratón esta fuera por la derecha o fuera por la izquierda entonces, puedes desplazar. Es "mouse.Confine" o algo así creo recordar, es "confine" seguro. Así es como se mueve en en el "lol". Así o minimapa y haces clic en un sitio en concreto. Eso es lo más intuitivo porque es lo que cualquier jugador de *League of Legends* va a entender. Yo puse que hacías clic derecho arrastrabas y te movías. Como vosotros tenéis un menú con clic derecho a lo mejor no es lo más intuitivo, pero clic normal y arrastrar creo que a vosotros os podría servir o, rueda del ratón y arrastrar. No necesitas quitar las teclas, las puedes dejar y tener las dos cosas. Por lo demás con el *zoom* la rueda del ratón me parece lo suyo. De movimiento creo que no nada más.

Lucía: Lo siguiente es que te parece la idea de poder dar y quitar el control a los diferentes usuario.

José: Me parece genial, yo en la mía los que veían no podían hacer nada, solo podían ver. En la vuestra pueden ver y si en un momento les digo: "oye tío explícame eso que me estas diciendo, que no me entero". Le das permiso y el mueve y deshace y quita y tal. Esta muy bien, eso me gusta.

Jeremy: Lo que si es que el "reset" lo hemos restringido solo al profesor.

Lucía: Sí, no vaya a ser que alguien te fastidie la sesión o cualquier cosa.

José: Sí, sí, sí, me parece bien.

Lucía: Una cosa que a mi me surgió la duda, el botón de "save"por ejemplo si que es individual. Te gustaría que fuese solo del profesor y en el momento que a ti te interesa...

José: A ver lo que más me gustaría es que pudieras habilitar el botón de "save" para que lo vean o no. Por ejemplo, si estoy dando una clase de una estrategia que no quiero que se la guarden pues preferiría decirles que no quiero habilitarles el botón de "save", pero si estoy dando una estrategia que no me importa que se la guarden o incluso que saquen *screams* y que las suban a sus redes sociales etc.., pues que se habilite el botón de "save". Es que hay ocasiones en las que si te interesa que el contenido que estás explicando salga y otras en las que no. Pero os digo una cosa, que hay gente que hace trampa igual, te hace un imprimir pantalla y ya. Entonces no se

si merece la pena que perdáis el tiempo en hacer eso, yo creo que no. Pero si encontráis una manera de poder evitarlo a toda costa, pues genial pero no creo, lo veo complicado. He visto aplicaciones que han llegado al absurdo de que si apretabas imprimir pantalla se ponía delante una lámina negra. Entonces al apretar imprimir pantalla te salía una imagen en negro, pero que sucede, que te hace çomviazoz te comes una mierda igual, o te ponen un programa que este grabando la pantalla o te lo "streamean" o te usan el OBS. Te lo pueden sacar de mil maneras así que no os preocupéis.

Lucía: Pero me parece buena idea, ya que la aplicación esta pensada para que la utilicéis en clase. Que es un entorno bastante controlado entonces...

José: Ya pero un "giazo" por ejemplo que en muchos ordenadores ya lo ha instalado algunos alumnos te hacen "clic clic" y ya. Yo no perdería el tiempo en hacer eso, personalmente.

Lucía: También nos interesa tu opinión sobre la idea del mapa interactivo como lo de hemos hecho de poder destruir torres, poder quitar los NPCs y cosas así.

José: Si, mira. la idea genial, poder destruir las torres me parece muy bueno. Insisto lo único que me falla un poco son sobre todo aspectos de diseño vale que hacen que la aplicación tengan un poco más de peso o de empaque o que digas "me gusta usarla". Ahora mismo visualmente esta justa, esta para funcionar pero, visualmente no está pulida. Entonces quizás no invita todavía a usarla del todo. Por ejemplo, aspectos que yo cambiaría, si yo paso el ratón por encima de una torre, que la torre se me ilumine como diciéndome que ahí puedo hacer algo. Porque ahora mismo si tu no me dices que puedo hacer clic en una torre y desmontarla, no lo sabría. Entonces esos pequeños detalles hacen que te den ganas de utilizar una aplicación. Que los minions no sean cuadros flotando es importante, parece una tontería pero son cosas importantes. Pero por lo demás, lo que es la ejecución, la funcionalidad de la aplicación, en lineas generales cumple con lo que necesitaría.

Lucía: Después la barra barra del tiempo. ¿Te parece que cumple su función?

José: La barra del tiempo es lo único que os falta por terminar del todo, por que hay datos que faltan. Porque solo la vida de los monstruos de la jungla no es suficiente, solo cuando sale el nashor no es suficiente, solo cuando sale el dragón no es suficiente. Me interesa el daño de los monstruos de la jungla, es muy importante. A lo mejor un gráfico de cuanto oro hemos ganado de forma pasiva en el "lol", que no me acuerdo cuanto se gana ahora en el "lol", no se si son dos o tres de oro cada décima de segundo o algo así. Pero que tu cuando aumente el tiempo por minuto, te salga el oro que has ganado pasivo o algo así. Datos que son interesantes para una partida, si me interesan. Quizás la linea del tiempo es lo único que aun os falta un poco de ejecución.

Jeremy: Osea que cuando tu marcas un minuto concreto quieres recibir

más información de ese minuto.

José: Correcto. Sobre todo con monstruos de jungla etc.. Porque ¿que es lo que te interesa del Nashor? ¿que sale en el minuto veinte? Todos sabemos que sale al minuto veinte. Quizás no todos sabemos la vida que tiene el Nashor, pero tu aquí me pones "tanda vida mas 140 por minuto". No quiero saber eso, quiero que me digas cuanta vida tiene, quiero que me calcules y me lo pongas. El daño que tiene igual. Por ejemplo me pregunta un alumno:"cuanto daño tiene el Nashor al minuto veintisiete?" y yo venir aquí y poner el minuto veintisiete y le digo: "pues tiene tanto". Eso es información útil y hace que la herramienta sea útil. Cuando quiero hacer una ruta de jungla: "no se si con "x" campeón puedo hacer una ruta completa al empezar porque mi campeón tiene 400 de vida mas dos pociones en total unos 900 de vida". Pues yo quiero poder poner los minutos en los que se va a ir moviendo en cada campamento, minuto uno vas a estar aquí, en el dos aquí, en el tres aquí y podemos ir sumando los daños y podemos ver si podemos hacer la ruta o no hacerla. Ya te digo, la barra del tiempo es lo que puede que más falte por pulir, el resto está muy bien. O sea que yo pueda cambiar los dragones está muy bien. Aunque también es verdad que los dragones no todos tienen los mismos *stats*, cada uno es diferente.

Jeremy: ¿Tienen diferente vida cada uno?

José: Y cada uno tiene diferente daño y, también escalan con el tiempo. Hay la cosa ya se va complicando pero bueno eso ya si lo podéis poner es ya para nota. Pero si no lo ponéis no pasa nada, con que pongáis una media o simplemente los datos generales de los cuatro a la vez, vale.

Lucía: Vale. Una pregunta. por ejemplo en el Nashor ahora mismo te estamos poniendo la fórmula, te interesa que mantengamos la fórmula o simplemente ponemos la vida ya calculada.

José: Es una buena pregunta. Cuando tenéis este "rect" aquí blanco, entiendo que aquí caben muchas cosas. [Refiriendose al panel del *poptag* de las stickers estáticas].

Jeremy: Lo ideal sería que se escalase en proporción a la cantidad de texto que tienes.

José: Vale. a ver, ¿me interesa la formula? sí. Pero yo personalmente pienso que es mas útil ir la dato concreto, porque normalmente... A ver, sí, si me pones las dos cosas sería lo mejor. Simplemente poner que "vida base más 140 por minuto" y abajo en paréntesis la vida en ese minuto quizás seria lo suyo. Eso seria lo mejor. En realidad teniendo el minuto y sabiendo que hay que sumarle 140 es facilísimo la verdad.

Lucía: Vale.

José: Incluso, entiendo que esta variable la tenéis ya programada.

Lucía: ¿El minuto?

José: Claro. ¿Esta variable ya te la guarda el programa, entiendo?

Lucía: Sí.

José: Porque con esto habéis calculado que salga esto. Así que tenéis que haber hecho un comparativo de: "*if* esto más mayor que "nosecuantos" sale el Nashor". ¿No?

Lucía: [afirma].

José: Vale, pues coges el minuto, lo multiplicas por 140, le metes un "toString" y fuera, y lo enchufas como texto ahí y a correr. Eso creo yo que sería fácil. No sería muy difícil. ¿Que más?

Lucía: Hemos hecho la aplicación de una forma un poco general para poder meter otros juegos.

José: Ah muy bien. Poder cambiar el fondo, cambiar cosas...

Lucía: Poder cambiar el texto, las imágenes también...

Jeremy: La aplicación en Unity no esta hecha con el *League of Legends*. Todo lo del "lol" está en una carpeta a parte.

José: [afirma].

Jeremy: Entonces cuando se empieza el juego coge un JSON lo lee y coge todos los recursos del la carpeta.

José: Ah, muy guay.

Lucía: Entonces la idea sería...

José: ¿Lo hacéis con un "resourcesLoad" o algo así?

Lucía: Si, cuando quieres cargar cualquier cosa, haces eso. La idea sería en el menú primero que hemos visto, tener una especie de *scroll* con los paquetes...

José: Y elegir League of legends, CSGO, ¿Fortnite? Si en Fortnite se podría usar para las rutas del mapa. Es interesante también. No se me había ocurrido pero es muy interesante.

Lucía: Lo hemos pensado especialmente para juegos tipo MOBA.

José: Vale. DOTA por ejemplo.

Lucía y Jeremy: Si.

José: Vale. Por ejemplo hacer Fortnite os seria muy fácil ¿vale?. Os lo digo para que lo sepáis porque en Fortnite no necesitaríais los campeones y todo eso te lo quitas de encima. lo único que tienes que cargar es un "muñequito" en el mapa, que lo puedas mover y vaya dejando la ruta. Eso es lo único que necesita un entrenador de Fornite, lo tenéis superfácil. O sea lo que tenéis hecho ya de los campeones, pero solo un campeón que es tu personaje y a moverlo por los sitios y las rutas y como mucho si queréis un "circulito" que te diga donde está la tormenta. ¡Ya está!

Lucía: Vale. ¿Se te ocurre alguna otra preguntas? [Preguntando a Jeremy].

Jeremy: No. ¿Se han terminado ya? [Preguntando a Lucía refiriéndose a las preguntas preparadas].

[Nos pregunta sobre la entrega del tfg]

Lucía: Sí, bueno ya decirte que cada uno de los personajes y NPCs puede

tener su propio menú.

José: Y apagarle la ruta, cambiar el color...

Lucía: Puedes poner el que a ti te interese. [refiriéndose al color].

José: ¡Incluso puedes cambiar el campeón!

Lucía: Sí.

José: ¿Directamente?

[cambiamos de campeón directamente a través del menú de edición]

José: Ah mira y lo puedes cambiar directamente uno por otro. Mira yo eso tampoco lo tenía. Tenías que darle a suprimir, quitarlo y crear otro.

Jeremy: También pensamos que podría ser cómodo para no tener que estar abriendo submenus, tener atajos de teclado.

José: Hombre, los atajo siempre vienen bien pero yo por ejemplo para la pizarra no los gastaría. Yo la pizarra la utilizo sobra todo para enseñar rutas, no necesito el teclado para arrastrar el "muñecote", pero habrá mucha gente que los encontrara muy útiles seguro. Es decir, si los pones, no te digo yo que un día no lo use, por ejemplo un atajo que sea meter una oleada donde tengo le ratón por ejemplo çtrl + oz *¡pam!* ya la tienes o control mas algo que este cerquita y *¡pam!* ya tengo los minions ahí puestos. De hecho una cosa genial seria que los minions no estén superpuesto uno encima del otro y que ya salgan orientados a la calle. Tres y tres, tres melée y tres magos y el *cannon* siempre iría en medio, dependiendo del minuto en el que estemos. Pero ya es complicaros mucho y ya son cosas de entrenadores el saber en que minuto sale un minion *cannon* y no de personas que están desarrollando una aplicación. Porque también tiene su "wasa", al principio salen cada x oleadas, y cuando llega x minuto salen cada menos tiempo y cuando llega x minuto salen cada menos tiempo y cuando llega x minuto salen cada menos tiempo y luego llega un momento en el que salen siempre.

Lucía: Vi una tabla explicativa y dije: "esto no es factible".

José: Es que el *league of legends* tiene muchas cosas es un juego muy complejo. Si os dais cuenta a lo que refiero siempre es... ¿Puedo ampliar uno de los dos? [Refiriendose a las ventanas de la aplicación].

lucía y Jeremy: Sí.

[No lo consigue abrir y continuamos de todas formas]

José: Por ejemplo los dragones, como habéis cogido un png y no tiene borde. Aun que creo que los podíais haber adaptado un poquito mejor al *pit*, visualmente queda bien, que es a lo que voy. Porque ahora mismo me chirría visualmente porque ver esto visualmente, no es lo suyo. pero bueno la ejecución esta bien y funcional es muy funcional y en lineas generales es útil. Sobre todo lo de la ruta.

[José intenta crear un campeón]

José: Ah vale aquí esta .ªddchamp". Ah y elijo el color. No me acordaba. Vamos a elegir uno del equipo azul y vamos a tener "Morgana"por ejemplo.

Hago clic derecho y le digo que me muestre la ruta. ¿No?

Lucía y Jeremy: Sí.

José: vale entonces ya en cuanto la muevo me pinta la "rutill". Luego clic derecho, puedo cambiar el color de la ruta al que me de la gana, perfecto y borrar la ruta, perfecto. Si borro al campeón se borra la ruta entiendo. ¿No?

Lucía y Jeremy: Sí.

[José borra el campeón junto a su ruta]

José: La ejecución esta muy bien. Ya te digo le falta pulirlo gráficamente.

Lucía: [Hablando a Jeremy] Si no te se ocurre nada más.

Jeremy: No se me ocurre nada.

Lucía: Vale pues ya la última pregunta sería: opinión realista. ¿Esto un entrenador de e-Sports lo utilizaría? ¿Le sería útil?

José: Muchos de ellos sí, pero no todos, porque cada uno tiene su manera de trabajar. pero creo que sí. Obviamente, ¿sabéis que pasa? Que os van a pedir que visualmente sea atractiva. Le llevas esto a un entrenador y te va a decir que estéticamente esto esta muy justo. Prefiero gastar una que tengo menos funcionalidades pero que visualmente me hace trabajar un poco mejor. Porque pasamos muchas horas utilizando estos rollos. Hay una que se llama Riftkit que creo que os la pase.

Lucía: Sí.

José: Que eso es lo más sencillo, de lo más sencillo, de lo más sencillo y hay gente que la utiliza porque no tiene nada que te chirrié. Hay otras como la mía que hay gente que el modelado este de que son "ciculitos" no le gusta y prefiere gastar el Riftkit. Hay gente que gasta la mía, hay gente que se ha hecho la suya propia. Hay gente que gasta el Photoshop no te digo más, que tiene varias capas y en las capas va moviendo las cosas, que dices: vamos tío pero no me seas cutre. Así que yo creo que si podría ser útil. Ahora bien, no creo que estarías dispuestos a pagar mucho por algo así porque ya hay mucho gratuito. Entonces esto en el momento en el que esto se quisiese monetizar se tendría que monetizar con algún tipo de publicidad y si eres *premium* pagando uno o dos euros, no te comes publicidad. Algo muy rollo Twitch que es a lo que estamos acostumbrados los "gamers". En plan, yo la puedo utilizar gratis y me como anuncios para que el que haya creado esto gane dinero, pero si no me quiero comer anuncios me subscribo por 2 o 3 euros. Eso quizá sería en mi opinión la política de monetizarlo. Pero para que la gente este dispuesta a gastarse dinero es esto tendríais que dejarla muy pulida visualmente.

Jeremy: Vale.

Lucía: Pues yo creo que ya está.

Lucía y Jeremy: La verdad, creemos que nos ha ayudado bastante.

José: Me alegro, hombre.

Lucía: Y bueno, pues ya está.