Universidad del País Vasco
Euskal Herriko Unibertsitatea

Facultad de Informática

Grado de Ingeniería Informática

# Proyecto Fin de Grado

Ingeniería del Software

Desarrollo para Comunidad Web de Videojuegos

Videojuego 3D Online Multijugador Multiplataforma

Iker Boyra Sarachaga

Junio 2016

# Agradecimientos

Ante todo, quiero dar las gracias a mi compañero de proyecto Héctor Antruejo Escalante por su gran trabajo y dedicación y por su amistad a lo largo de todos estos años.

Agradezco también a nuestro profesor José Ángel Vadillo Zorita, no solo por sus enseñanzas, sino por unirnos a Héctor y a mí por primera vez en un mismo grupo de trabajo. Aquello marcó el inicio de nuestra amistad y probablemente fue el origen de este proyecto.

Muchísimas gracias a nuestro profesor y tutor José Miguel Blanco Arbe por todo su apoyo, sus lecciones y sus buenos consejos, y por ser siempre capaz de mostrarnos nuevos enfoques.

Muchas gracias a Rebeca Fernández Armengol por aguantarnos a Héctor y a mí y por el gran apoyo que me da siempre que lo necesito.

Gracias a mi familia por confiar en mí y mostrarme siempre sus apoyo.

A mis amigos, por estar siempre ahí y por ayudarme con cualquier problema que me surja.

A todos mis compañeros de clase, por compartir conmigo estos magníficos años de universidad.

Muchas gracias a todos.

# Resumen

A raíz de la iniciativa empresarial IHEGames, surjen dos proyectos de fin de grado paralelos con el objetivo de desarrollar unas primeras versiones de los sistemas que forman el núcleo de esta iniciativa empresarial.

En este proyecto de fin de grado se desarrolla el juego K-Tan para las plataformas web y Android. K-Tan es un juego 3D online multijugador multiplataforma basado en un juego de mesa y consiste en el primero de los juegos de IHEGames. En su desarrollo se establecen las bases para la creación y el desarrollo de los futuros juegos de IHEGames.

Mientras que en el segundo proyecto, desarrollado en paralelo, se implementan algunos de los servicios web de IHEGames de los que K-Tan hace uso y el Portal D20, un portal web de juegos donde se aloja la versión web de K-Tan.

# Índice

# Figuras

# Tablas

# 1

# Introducción

En este capítulo se describe, en líneas generales, el contenido del documento. Por un lado se expone la relación y dependencias que existen entre este proyecto y otro de los proyectos de fin de grado desarrollados en paralelo. Por otro lado, se describe la estructura de esta memoria y el contenido de sus capítulos. Finalmente, se detalla el formato que se ha utilizado a lo largo del documento para resaltar las partes comunes de ambos proyectos.

## 1.1 Introducción del proyecto

A raíz de la iniciativa empresarial IHEGames, surgen dos proyectos de fin de grado paralelos con el objetivo de desarrollar unas primeras versiones de los sistemas que forman el núcleo de esta iniciativa empresarial.

Por un lado, se lleva a cabo este proyecto de fin de grado, en el cual se desarrolla K-Tan para las plataformas web y Android. K-Tan consiste en un juego 3D online multijugador multiplataforma basado en un juego de mesa y es el primero de los juegos de IHEGames. En su desarrollo se establecen también las bases para la creación y el desarrollo de los futuros juegos de IHEGames.

Por otro lado, en el proyecto de fin de grado *Desarrollo para Comunidad Web de Videojuegos: Portal Web de Videojuegos Online Multijugador*[1] desarrollado en paralelo por Héctor Antruejo Escalante, se implementan algunos de los servicios web de IHEGames de los que K-Tan hace uso y el Portal D20, un portal web de juegos donde se aloja la versión web de K-Tan.

Tras este capítulo introductorio, se describen las motivaciones que han dado pie a la iniciativa empresarial IHEGames así como en qué consiste y cuáles son sus objetivos para con estos dos proyectos paralelos. Asimismo, se describen muchas de las pautas y tecnologías que ha marcado la propia iniciativa empresarial para el desarrollo de ambos proyectos.

Una vez descritos los antecedentes que proporcionan la base del sentido y la relación entre ambos proyectos, se describen los objetivos concretos de este proyecto así como los límites de su alcance. Estos proporcionan una idea general del tamaño y las partes del desarrollo realizado. A su vez, se presenta también un capítulo con el ciclo de vida del proyecto. En este capítulo se describe el ciclo de vida incremental que ha establecido IHEGames para la elaboración de ambos proyectos. Se explican las diversas fases y etapas de las que se hace uso y se especifica qué parte de ese ciclo de vida de proyecto engloba este proyecto de fin de grado.

Una vez presentadas las bases del proyecto, se describe la arquitectura general cliente-servidor de K-Tan y sus comunicaciones con los servicios de IHEGames y el propio Portal D20. A su vez se describen la interfaces de los servicios de IHEGames que se han desarrollado en el proyecto paralelo y que se han integrado con K-Tan.

Para entender algunos de los términos y conceptos que se hallan a lo largo de esta memoria se reserva un capítulo en el cual se explica el reglamento de juego de K-Tan, basado en el juego de mesa *Los Colonos de Catan*. A partir de este reglamento de juego y de otras especificaciones dadas por IHEGames, se identifican los actores y los casos de uso, así como los requisitos funcionales y no funcionales que se han desarrollado a lo largo de este proyecto.

A continuación se detalla el análisis y el diseño realizado, separando las tres capas de modelo de datos, lógica de negocio e interfaz de usuario. Asimismo, se describe la implementación realizada principalmente con Visual Studio y Unity y se describen los Asset o recursos que se han utilizado para el desarrollo de K-Tan.

Por otro lado, se detallan los casos de prueba realizados así como las pruebas con usuarios que se han realizado en una de las etapas finales de este proyecto. Asimismo, se describen también algunos de los pasos más significativos que se han llevado a cabo en la puesta en producción que ha sido necesaria para poder llevar a cabo dichas pruebas con usuarios.

Finalmente, se realiza una reflexión sobre la gestión llevada a cabo a lo largo de este proyecto. En este capítulo se describen principalmente la gestión del alcance, los costes y las adquisiciones, así como la gestión de las comunicaciones y del tiempo que han sido fundamentales para coordinar y llevar a cabo satisfactoriamente ambos proyectos de IHEGames.

Para terminar, se desarrollan unas conclusiones personales sobre el trabajo en equipo y el desarrollo con la herramienta Unity, y se presentan unas reflexiones de los futuros pasos de K-Tan y los sistemas de IHEGames.

## 1.2 Formato del documento

Este proyecto se realiza en paralelo a otro proyecto de fin de grado con el que tiene grandes dependencias. Por ello es inevitable que algunos capítulos de esta memoria contengan partes comunes que se comparten en las memorias de ambos proyectos.

Por otro lado, ambos proyectos de fin de grado se presentan en el mismo periodo. Para facilitar al lector la comprensión del documento y el saber si lo que lee es algo común de las dos memorias o algo específico de ésta, se han presentado las partes comunes con un color de letra más claro y grisáceo, mientras que las partes específicas de este proyecto se mantienen en color negro.

# 2

# Antecedentes

En este capítulo se resumen las motivaciones personales que han dado pie a la iniciativa empresarial IHEGames.

Asimismo, se detallan los sistemas principales que forman el núcleo del sistema de IHEGames. Finalmente, se describen cuáles son sus objetivos para con los dos proyectos de fin de grado que surgen a partir de esta iniciativa y se detallan muchas de las pautas y tecnologías que ha marcado la propia iniciativa empresarial para el desarrollo de ambos proyectos.

## 2.1 IHEGames

A lo largo del año 2014, Héctor Antruejo e Iker Boyra deciden unirse con la idea de aplicar sus conocimientos profesionales y académicos adquiridos a lo largo del grado de informática para crear la iniciativa empresarial IHEGames.

Tras haber trabajado conjuntamente en varios proyectos desde que sus caminos se cruzaron en Ingeniería del Software, y debido a su afán compartido por los videojuegos, deciden enfocar la iniciativa empresarial IHEGames al desarrollo de videojuegos online.

A comienzos del 2015 cuando la iniciativa aun está comenzando a tomar forma, las ventas de los videojuegos son casi equiparables a las ventas en grandes industrias como podría ser la del cine, tal y como recoge la Asociación Española de Videojuegos[2], la cual asegura que: *"La industria del videojuego ha mantenido en los últimos años su posición como principal industria de ocio audiovisual e interactivo, con una cuota de mercado muy superior a la del cine y la música".*

La iniciativa IHEGames pretende crear, entre otros, un portal de juegos denominado D20 y desarrollar los juegos que alojara en él.

El portal D20 ofrecerá a los jugadores de todo la posibilidad de satisfacer sus necesidades de juego. El portal D20 se basa en una página web que dispondrá de una variedad de juegos de calidad que cuiden sus gráficos, aspecto visual y jugabilidad. A su vez, dichos juegos podrán ser descargables y jugables desde otras plataformas ajenas al portal, como por ejemplo, dispositivos móviles.

El portal D20 no es un mero portal web. D20 aspira a convertirse en una gran comunidad de jugadores de todo el mundo, una comunidad en la que los usuarios podrán interactuar y jugar con el resto de usuarios a la selecta variedad de juegos que tendrán a su disposición.

El ámbito de los videojuegos es muy extenso como para abarcarlo completamente, por lo que se decide priorizar el lanzamiento de videojuegos por categorías.

El sector de los juegos de mesa ha crecido significativamente en la última década. Este sector, que hasta hace poco se centraba en juegos para niños o para la familia, ha ampliado sus objetivos a jugadores más experimentados dando como resultado una enorme variedad de juegos de mesa de todo tipo y dificultad. Aunque este sector sigue siendo desconocido para gran parte de la sociedad las cifras de ventas de este tipo de juegos han aumentado considerablemente tal y como se muestra en esta noticia de El Mundo[3]: *“Dos juegos que ya se pueden considerar dos fenómenos mundiales con 20 millones de copias vendidas (Los Colonos de Catán) y 10 millones de copias vendidas (Carcassonne).”*

Dada la carencia detectada, IHEGames decide aprovecharla y centrarse en sus inicios en el desarrollo de juegos de mesa (juegos de mesa, tablero, cartas, casino) de tipo multijugador.

## 2.2 Arquitectura de IHEGames

IHEGames establece siete sistemas principales, que compondrán la arquitectura base del portal D20. Estos sistemas consisten en el propio portal D20, los juegos que se alojan en D20 y los servicios comunes que dan soporte tanto al portal como a los propios juegos.

En la siguiente figura se muestra la arquitectura general que componen los sistemas base de IHEGames y las dependencias que existen entre ellos:

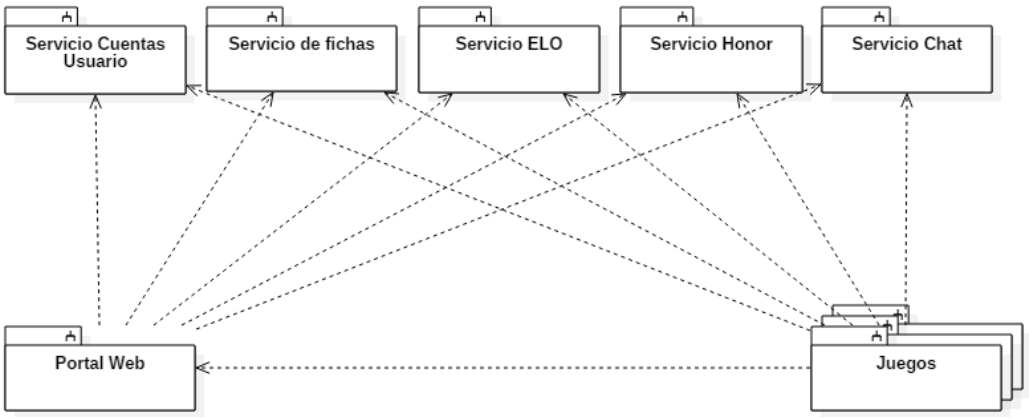


Figura 1: Sistemas de IHEGames

Todas las comunicaciones entre estos sistemas, se realizan mediante conexiones suficientemente seguras, de manera que se pueda garantizar la autenticidad y la privacidad de los datos transmitidos.

### 2.2.1 Servicio de cuentas de usuario

Esta aplicación almacena las Credenciales de los usuarios de IHEGames y es el sistema de gestión de usuarios que utilizan el resto de sistemas de IHEGames. Este servicio es de acceso exclusivo de las aplicaciones de IHEGames.

Por otro lado almacena los datos básicos de perfil de los usuarios y sus avatares.

El servicio de cuentas limita la creación y modificación de cuentas de usuario a algunas aplicaciones exclusivas, permitiendo al resto únicamente obtener los datos de las cuentas y autentificar a los usuarios.

El servicio de cuentas permite que la autentificación de los usuarios se realice mediante usuario y contraseña, o mediante otros proveedores de cuentas como Google, Facebook y Twitter.

### 2.2.2 Servicio de fichas

Esta aplicación almacena y gestiona las fichas de los usuarios de IHEGames. Este servicio es de acceso exclusivo de las aplicaciones de IHEGames.

Las fichas consisten en una moneda virtual que los usuarios pueden apostar en las partidas y utilizarlas para tener acceso a funcionalidades premium. Estas fichas se pueden obtener mediante pagos con dinero real o como bonificaciones otorgadas a los usuarios en su uso de las aplicaciones de IHEGames.

### 2.2.3 Servicio de Elo

Esta aplicación es la encargada de gestionar tanto la puntuación Elo como el nivel y la Experiencia de los usuarios en los diversos juegos de IHEGames.

El sistema de puntuación Elo[4] es un método matemático, basado en cálculo estadístico, para calcular la habilidad relativa de los jugadores en juegos como el ajedrez. La puntuación Elo de un jugador aumenta con las victorias y disminuye con las derrotas. La cantidad que aumenta o disminuye depende de su puntuación Elo y la de sus rivales.

La Experiencia de un usuario simboliza su experiencia de juego y la cantidad de partidas de un juego concreto en las que ha participado. El nivel de usuario se calcula dependiendo de su experiencia mediante otra fórmula matemática.

El servicio calcula y almacena una puntuación Elo, la Experiencia y el nivel para cada usuario en cada juego. Por otro lado, el servicio da la posibilidad de permitir que aplicaciones ajenas a IHEGames, con sus propios usuarios, sean capaces de consumir este servicio, siempre y cuando estén autorizadas para poder hacer uso de este servicio.

### 2.2.4 Servicio de Honor

Esta aplicación es la encargada de gestionar el Honor de los usuarios en los diversos juegos de IHEGames.

El Honor de un usuario es un concepto utilizado por IHEGames que simboliza la conducta o comportamiento general que ha demostrado en partidas anteriores y permite al resto de usuarios prever el tipo de jugador que es.

El Honor de un usuario se calcula mediante una fórmula matemática en base a los votos positivos o negativos recibidos, los abandonos y los retrasos de las últimas partidas en un

juego concreto. Los votos positivos suben el Honor de un usuario. Por el contrario, los votos negativos, los abandonos y los retrasos disminuyen su Honor.

El servicio calcula y almacena unos datos de Honor para cada usuario en cada juego. Y como el servicio de Elo, este servicio también da la posibilidad de permitir que aplicaciones ajenas a IHEGames, con sus propios usuarios, sean capaces de consumir este servicio, siempre y cuando estén autorizadas para poder hacer uso de este servicio.

### 2.2.5 Servicio de Chat

Esta aplicación permite que los usuarios de IHEGames se puedan comunicar entre ellos mediante el envío de mensajes. Este servicio está limitado a los usuarios de IHEGames, pudiendo solo estos acceder al servicio.

Este servicio permite enviar mensajes a los usuarios de una misma aplicación o a usuarios en otras aplicaciones. Por otro lado el servicio permite enviar mensajes privados a usuarios concretos o a los usuarios de la misma partida.

### 2.2.6 Portal de juegos D20

Aplicación web que expone los juegos desarrollados por IHEGames y ofrece a sus usuarios un entorno para interactuar y jugar entre ellos.

En este portal D20 los usuarios pueden registrarse o modificar sus datos de cuenta, perfil y avatar. Los usuarios tienen también la posibilidad de añadir a otros usuarios a su lista de amigos. Los usuarios identificados pueden visualizar y acceder a los juegos alojados en el portal D20, así como visualizar su puntuación Elo, Experiencia nivel y Honor en los diversos juegos.

El portal D20 ofrece la posibilidad de visualizar su contenido en múltiples idiomas. También contiene un panel de administración, desde el cual se gestionan las noticias que se pueden publicar en el portal.

### 2.2.7 Juegos

Los juegos de IHEGames permiten a sus usuarios jugar partidas contra otros usuarios a través de internet en tiempo real. Los juegos son multiplataforma, estos están alojados en el portal D20 para jugarlos en un navegador y están también disponibles en Google Play y App Store para descargarlos y jugar en dispositivos Android e IOS respectivamente.

Los usuarios pueden crear nuevas partidas e invitar a otros usuarios a entrar a ellas. Los usuarios pueden visualizar todas las partidas disponibles y unirse a las partidas creadas por otros usuarios. Los usuarios pueden también visualizar partidas en curso uniéndose a ellas como espectador.

Al crear una partida los usuarios pueden configurar unos parámetros

- El número de jugadores: determina el número de jugadores que tiene la partida.
- Espectadores: determina si se permite que otros usuarios se unan como espectadores a la partida.
- Tipo de acceso: limita el acceso de otros jugadores a la partida a cualquiera, a amigos o a invitados.
- Partida competitiva: determina si el resultado de la partida influirá en las puntuaciones Elo de los jugadores.
- Limitar Elo: determina si solo los jugadores de una misma categoría de Elo o superior puedan acceder a la partida. La categoría de Elo de un usuario depende de su puntuación Elo. Las categorías existentes son "sin categoría", bronce, plata, oro, diamante y platino.
- Limitar Honor: determina si se restringe el acceso a los jugadores no honorables. Los jugadores no honorables son aquellos usuarios que tienen un valor de Honor inferior a un valor determinado.
- Fichas: Determina el número de fichas que apuestan los jugadores en la partida.

Al igual que el portal D20, los juegos ofrecen la posibilidad de visualizar su contenido en múltiples idiomas

## 2.3 Objetivos de IHEGames

En septiembre de 2015 los promotores de IHEGames plantean una etapa previa a la construcción de IHEGames como entidad empresarial. En esta etapa, IHEGames pretende para noviembre de 2016 tener en funcionamiento la arquitectura base descrita en el apartado anterior con un primer juego en funcionamiento.

Dado el periodo académico de los promotores de IHEGames, se establece una sub-etapa hasta junio de 2016. A lo largo de este periodo se plantea desarrollar, en dos proyectos de fin de grado, unos prototipos funcionales de los servicios de cuentas de usuarios, Elo y Honor, así como el desarrollo del Portal D20 y del primer juego de IHEGames.

Como primer juego, IHEGames toma la determinación de desarrollar el juego K-Tan, basado la versión básica del juego de mesa "Colonos de Catan". Con este primer juego se establece la base para el desarrollo de los futuros juegos de IHEGames.

Otro de los objetivos fundamentales en esta etapa consiste en desarrollar y perfeccionar una cultura organizativa que consolide la base de un adecuado trabajo en equipo y faciliten la gestión, tanto de este proyecto como de proyectos futuros.

Adicionalmente se pretende lograr una eficaz comunicación y visibilidad entre los distintos proyectos de IHEGames y sus equipos de trabajo.

## 2.4 Tecnologías de IHEGames

IHEGames establece y marca una serie de tecnologías concretas para el desarrollo y la elaboración de sus proyectos.

### 2.4.1 Unity

IHEGames establece Unity como su principal plataforma de desarrollo de juegos, que serán desarrollados con tecnología 3D.

Unity es un motor de videojuego multiplataforma creado por Unity Technologies. El scripting de Unity viene a través de Mono. El script se basa en Mono, la implementación de código abierto de .NET Framework. Los programadores pueden utilizar UnityScript (un lenguaje personalizado inspirado en la sintaxis ECMAScript), C# o Boo (que tiene una sintaxis inspirada en Python).

Unity está disponible como plataforma de desarrollo para Microsoft Windows, OS X y Linux, y permite crear juegos para IOS, Android, Windows Phone, Tizen, Windows, Windows Store Apps, Mac, Linux/Steam OS,WebGL, Play Station, Xbox, Wii, Nintendo 3DS, Oculus Rift, Google CarBoard, Steam Vr, Gear Vr, Microsoft Hololens, Android Tv, Samsumg Smart TV, TvOS.

IHEGames establece que las versiones web de sus juegos se compilen en WebGL. WebGL está desarrollada por Mozilla Foundation y consiste en un estándar para motores gráficos de juegos en 3D para navegadores.

WebGL permite mostrar gráficos en 3D acelerados por hardware (GPU) en páginas web, sin la necesidad de plug-ins en cualquier plataforma que soporte OpenGL 2.0 u OpenGL ES 2.0. Técnicamente es un API para javascript que permite usar la implementación nativa de OpenGL ES 2.0 que será incorporada en los navegadores. WebGL es gestionado por el consorcio de tecnología sin ánimo de lucro Khronos Group .

Unity cuenta con Unity Asset Store que es un recurso disponible en el editor de Unity. Los usuarios de Unity pueden acceder a una colección de paquetes de Asset en una amplia gama de categorías, incluyendo modelos 3D, texturas y materiales, sistemas de partículas, música y efectos de sonido, tutoriales y proyectos, paquetes de scripts, extensiones para el editor y servicios en línea.

### 2.4.2 ASP.NET

IHEGames establece ASP.NET como framework principal para el desarrollo de sus servicios y aplicaciones web. ASP.NET es un framework para aplicaciones web desarrollado y

comercializado por Microsoft. Es usado por programadores y diseñadores para construir sitios web dinámicos, aplicaciones web y servicios web XML.

Apareció en enero de 2002 con la versión 1.0 del .NET Framework, y es la tecnología sucesora de la tecnología Active Server Pages (ASP). ASP.NET está construido sobre el Common Language Runtime, permitiendo a los programadores escribir código ASP.NET usando cualquier lenguaje admitido por el .NET Framework.

### 2.4.3 Visual Studio 2015

IHEGames estable Visual Studio 2015 como su principal entorno de desarrollo para aplicaciones ASP.NET

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE) para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como C++, C#, Visual Basic .NET, F#, Java, Python, Ruby, PHP; al igual que entornos de desarrollo web como ASP.NET MVC, Django, etc., a lo cual sumarle las nuevas capacidades online bajo Windows Azure en forma del editor Monaco.

Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, consolas, etc.

Los usuarios de Visual Studio pueden utilizar su extensión integrada Nuget. Nuget es un gestor de paquetes gratuito y de software libre diseñado para la plataforma de desarrollo de Microsoft.

### 2.4.4 C#

IHEGames establece C# como lenguaje principal de desarrollo tanto en Unity como en las aplicaciones realizadas con Visual Studio.

C# es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA (ECMA-334) e ISO (ISO/IEC 23270). C# es uno de los lenguajes de programación diseñados para la infraestructura de lenguaje común.

Su sintaxis básica deriva de C/C++ y utiliza el modelo de objetos de la plataforma .NET, similar al de Java, aunque incluye mejoras derivadas de otros lenguajes.

Por otro lado, IHEGames establece el uso del inglés para las definiciones de los nombres de variables, propiedades, clase, métodos, etc. Por ello, todos los diagramas mostrados en esta memoria que hacen referencia a elementos de código o al propio código mantendrán su nomenclatura anglosajona.

### 2.4.5 WCF

IHEGames establece WCF como principal plataforma de mensajería de sus servicios web. Windows Communication Foundation o WCF, es la plataforma de mensajería que forma parte de la API de la Plataforma .NET 3.0.

Fue creada con el fin de permitir una programación rápida de sistemas distribuidos y el desarrollo de aplicaciones basadas en arquitecturas orientadas a servicios (también conocido como SOA), con una API simple; y que puede ejecutarse en una máquina local, una LAN, o sobre Internet en una forma segura.

### 2.4.6 ASP.NET MVC

IHEGames establece el patrón ASP.NET MVC para la elaboración del portal D20 y del resto de sus páginas web.

El ASP.NET MVC Framework es un framework de aplicaciones web que implementa el patrón modelo-vista-controlador (MVC). Basado en ASP.NET, permite a los desarrolladores de software construir una aplicación web como una composición de tres funciones: modelo, vista y controlador.

Uno de los módulos a destacar de ASP.NET MVC es ASP.NET Identity ASP.NET Identity es un sistema para la gestión de usuarios en aplicaciones web ASP.NET. Esto incluye la gestión de las Credenciales y almacenamiento de datos de los usuarios en las páginas web. Adicionalmente ASP.NET Identity permite la extensión de sus funciones de manera sencilla.

### 2.4.7 SSL/TLS

IHEGames estable como protocolo de comunicación el uso de SSL/TLS (Secure Socket Layer / Trasport Layer Security) en todas las comunicaciones de sus aplicaciones. Este protocolo proporciona autenticación y privacidad de la información entre extremos sobre Internet mediante el uso de criptografía.

Desde comienzos del siglo XXI el uso de HTTPS se ha ido popularizando en las aplicaciones de internet. Esto se ve reflejado con la iniciativa de Https Everywhere[5] promocionada por la Electronic Frontier Foundation y fundada en el año 2010, que fomenta el uso de conexiones cifradas HTTPS. Por otro lado buscadores como Google comienzan a premiar el uso de este protocolo, aumentando el SEO de las URL HTTPS, como indican en su blog de seguridad[6]: *"we also started giving a slight ranking boost to HTTPS URLs in search results last year".*

### 2.4.8 Diseño Adaptativo o Responsive

IHEGames establece como política de diseño el uso de interfaces con un diseño adaptativo o Responsive.

Es una filosofía de diseño y desarrollo cuyo objetivo es adaptar la apariencia de las aplicaciones al dispositivo que se esté utilizando para visualizarlas. Hoy día las aplicaciones se ejecutan y visualizan en multitud de dispositivos como tabletas, teléfonos inteligentes, libros electrónicos, portátiles, PCs, etcétera. Además, aún dentro de cada tipo, cada dispositivo tiene sus características concretas: tamaño de pantalla, resolución, potencia de CPU, sistema operativo o capacidad de memoria entre otras. Esta tecnología pretende que con un único diseño, se obtenga una visualización adecuada en cualquier dispositivo.

Esto se refleja en buscadores como el de Google que crea la iniciativa de Mobile Friendly[7] aumentando el valor SEO de páginas web con diseño Responsive y penalizando las páginas que no lo cumplan.

### 2.4.9 Mingle

IHEGames establece Mingle como herramienta para la gestión y el seguimiento y control de sus proyectos.

Mingle es una herramienta online para la gestión de proyectos que permite a compañías de cualquier tamaño implementar, gestionar y escalar metodologías ágiles.

La herramienta Mingle ha sido desarrollada por Thoughtwork Inc. y es utilizada también por empresas conocidas de gran envergadura y mucho peso en el ámbito tecnológico como son: Siemens o Cisco.

Mingle tiene a disposición de sus usuarios, plantillas basadas en metodologías ágiles para el desarrollo de software como son el Kanban, Agile o Scrum.

Mingle, permite trabajar con cartas conocidas como *Index Card* las cuales pueden simbolizar una tarea, *story*, *bug*, característica, y mucho más. Junto a las muchas características que las cartas pueden tener, las cartas contienen una descripción y un responsable.

Las cartas se sitúan en muros o *Card Walls*, donde se pueden organizar de múltiples maneras, ya sea por el tipo de carta, por la etapa en la que se encuentra, por las características que compartan, etc.

### 2.4.10 Mysql

IHEGames establece Mysql como sistema de gestión de bases de datos. Mysql es un sistema de gestión de bases de datos relacionales desarrollada bajo licencia dual GPL y Licencia comercial por Oracle Corporation. La base de datos relacional es un tipo de base de datos

que cumple con el modelo relacional el cual permite establecer interconexiones o relaciones entre los datos almacenados.

A la fecha de la realización de este proyecto, Mysql es una de las bases de datos más utilizadas, estando en el segundo puesto de bases de datos más utilizadas según la página web de db-engines.com[8]

# 3

# Objetivos

En este capítulo se desarrolla el objetivo principal de este proyecto, que consiste en la construcción de un primer prototipo funcional de K-Tan, así como el resto de objetivos que derivan a partir de éste. Por otro lado, se describe en que parte del sistema principal de IHEGames consiste este desarrollo de K-Tan y se detallan las dependencias que tiene con el resto de los sistemas de IHEGames. Finalmente, se detallan las inclusiones y exclusiones que definen los límites de su alcance y se define un EDT con los paquetes de trabajo principales que han guiado el desarrollo de este proyecto.

## 3.1 Descripción de los objetivos

El objetivo principal de este proyecto es desarrollar una primera versión funcional de K-Tan.

K-Tan consiste en un juego 3D online multijugador y multiplataforma que será el primero de los juegos de IHEGames. Este primer juego se basa en la versión básica del juego de mesa alemán *Los colonos de Catan*.

*Los Colonos de Catán* o *Los Descubridores de Catán*, es un juego de mesa de 2 a 4 jugadores inventado por *Klaus Teuber*. Perteneciente al nuevo estilo de juegos de mesa europeos, es probablemente el primer juego de mesa alemán que ha alcanzado popularidad fuera de Europa.

El objetivo de *Los colonos de Catán* es construir poblados, ciudades y carreteras sobre un tablero formado por terrenos hexagonales y que es distinto cada vez, mientras se van acumulando varios tipos de cartas y recursos. Todos estos elementos proporcionan distintas puntuaciones, ganando la partida el primer jugador que llega a los diez puntos.

Otro de los objetivos principales en el desarrollo de K-Tan, es diseñar y establecer una base reutilizable para el desarrollo de K-Tan y de los futuros juegos de IHEGames.

Por otro lado, se pretende probar el funcionamiento de la primera versión de la arquitectura que forman los sistemas principales de IHEGames, al integrear en K-Tan los servicios web funcionales de IHEGames y al desplegar una versión web de K-Tan en el Portal D20.

Adicionalmente a la versión web de K-Tan, se pretende desarrollar una versión del juego para dispositivos Android.

Una vez completado el desarrollo de la versión funcional de K-Tan, y su integración en la arquitectura de IHEGames, se pretende realizar unas pruebas con usuarios para validar la satisfacción de los usuarios y obtener los datos suficientes que permitan definir las futuras mejoras del juego.

En la siguiente figura se destaca el sistema desarrollado en este proyecto y las dependencias con el resto de sistemas de IHEGames:

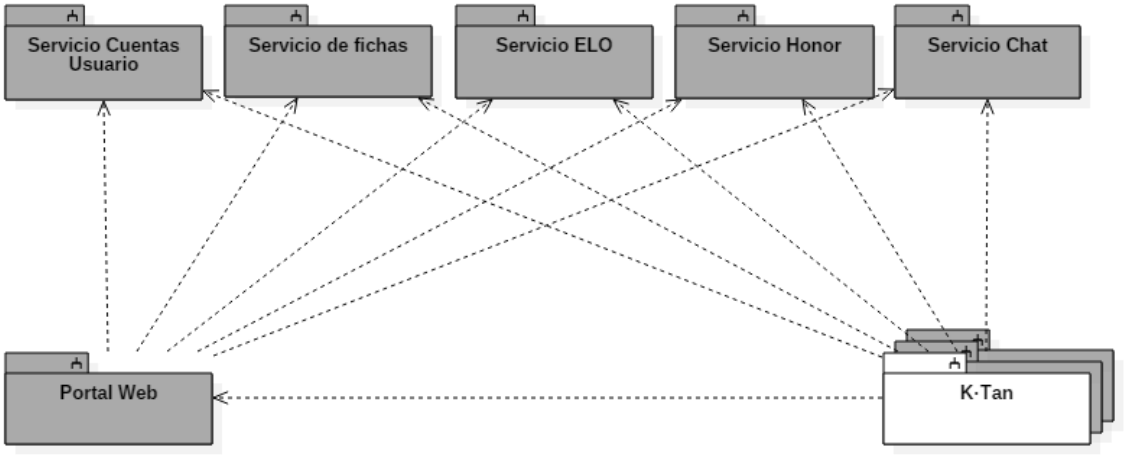


Figura 2: Sistemas desarrollados en el proyecto

## 3.2 Alcance

En este apartado se definen las inclusiones y exclusiones que definen el alcance del proyecto realizado.

### 3.2.1 Inclusiones

Las inclusiones de este proyecto están englobadas en el desarrollo de la primera versión funcional del juego de K-Tan. A continuación se describen las inclusiones concretas del proyecto:

- Análisis de los requisitos funcionales base de los juegos de IHEGames y específicos de K-Tan
- Análisis de los requisitos no funcionales base de los juegos de IHEGames y específicos de K-Tan
- Definición de los casos de uso base de los juegos de IHEGames y específicos de K-Tan
- Diseño de la arquitectura y las comunicaciones base de los juegos de IHEGames y específico de K-Tan
- Diseño e implementación del modelo de datos base de los juegos de IHEGames y específico de K-Tan
- Diseño e implementación de la base de datos para el almacenamiento de datos de K-Tan.
- Diseño e implementación de la lógica de negocio base de los juegos de IHEGames y específica de K-Tan
- Diseño e implementación adaptable de las interfaces de usuario base de los juegos de IHEGames y específicas de K-Tan

- Integración de K-Tan con el servicio de cuentas de usuario de IHEGames
- Integración de K-Tan con el servicio de Elo de IHEGames.
- Integración de K-Tan con el servicio de Honor de IHEGames
- Producción de una versión web del juego de K-Tan
- Producción de una versión Android del juego de K-Tan
- Integración del juego de K-Tan en el Portal D20
- Realización de pruebas de software para el juego K-Tan
- Realización de pruebas con usuarios para el juego K-Tan y obtención de feedback de los usuarios

A continuación se enlistan las funcionalidades que se desarrollan en esta primera versión de K-Tan:

- Funcionalidades generales de los juegos de IHEGames:
    - Identificación de usuarios mediante usuario y contraseña
    - Selección de idiomas (Inglés y Español)
    - Activación y desactivación de música y efectos de sonido
    - Crear y configurar partidas
    - Visualizar partidas creadas
    - Unirse a partidas creadas
    - Invitar a usuario a unirse a partida
    - Visualizar partidas
    - Jugar partidas
- Funcionalidades específicas de K-Tan basadas en el reglamento de la versión básica del juego de mesa *Los Colonos de Catan*[9] y recogidas en el capítulo *Reglamento de juego*.

### 3.2.2 Exclusiones

Se excluye del alcance del proyecto todo lo no mencionado en las inclusiones, de lo que se destaca:

- La implementación de las funcionalidades de cualquiera de las expansiones existentes del juego de mesa “Colonos de Catan”.
- El desarrollo del Portal D20 y los servicios web de IHEGames.
- La identificación de usuarios mediante otros proveedores de identificación de usuarios como Google, Facebook o Twitter.
- La puesta en explotación de K-Tan, así como las disposiciones legales y reglamentarias relacionadas con la puesta en explotación.
- El análisis y pruebas de rendimiento y disponibilidad del juego de K-Tan.
- La justificación de alternativas de las tecnologías escogidas para la realización de este proyecto.

## 3.3 EDT

A continuación se muestra un EDT con los paquetes de trabajo que han formado parte de este proyecto.

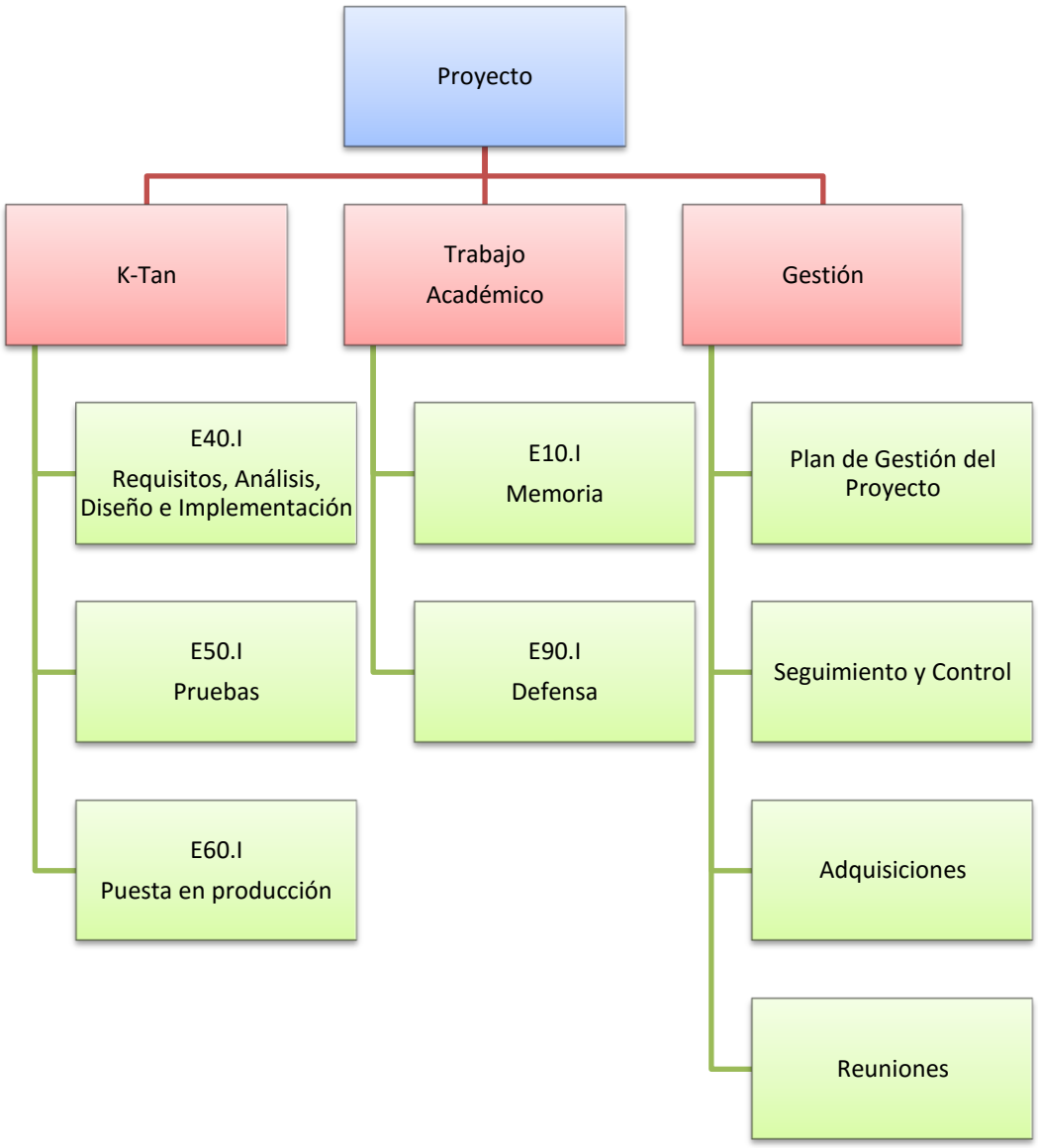


Figura 3: EDT del proyecto

# 4

# Ciclo de vida del proyecto

A lo largo de este capítulo se describe el ciclo de vida incremental que ha establecido IHEGames para el desarrollo de este proyecto y se detallan las diferentes fases y etapas en las que se ha descompuesto, así como qué parte de dicho ciclo de vida engloba este proyecto de fin de grado.

## 4.1 Descripción del ciclo de vida

El ciclo de vida de este proyecto está determinado por el ciclo de vida de los proyectos establecido por IHEGames, el cual establece una serie de fases a lo largo del desarrollo de un proyecto que se pueden repetir a lo largo de uno o varios de ciclos.

### 4.1.1 Fases en la etapa de desarrollo

El desarrollo de los proyectos de IHEGames consta de cuatro fases divididas a su vez en una o más iteraciones.

En la siguiente figura se recoge la estructura general de las fases en el desarrollo del proyecto.

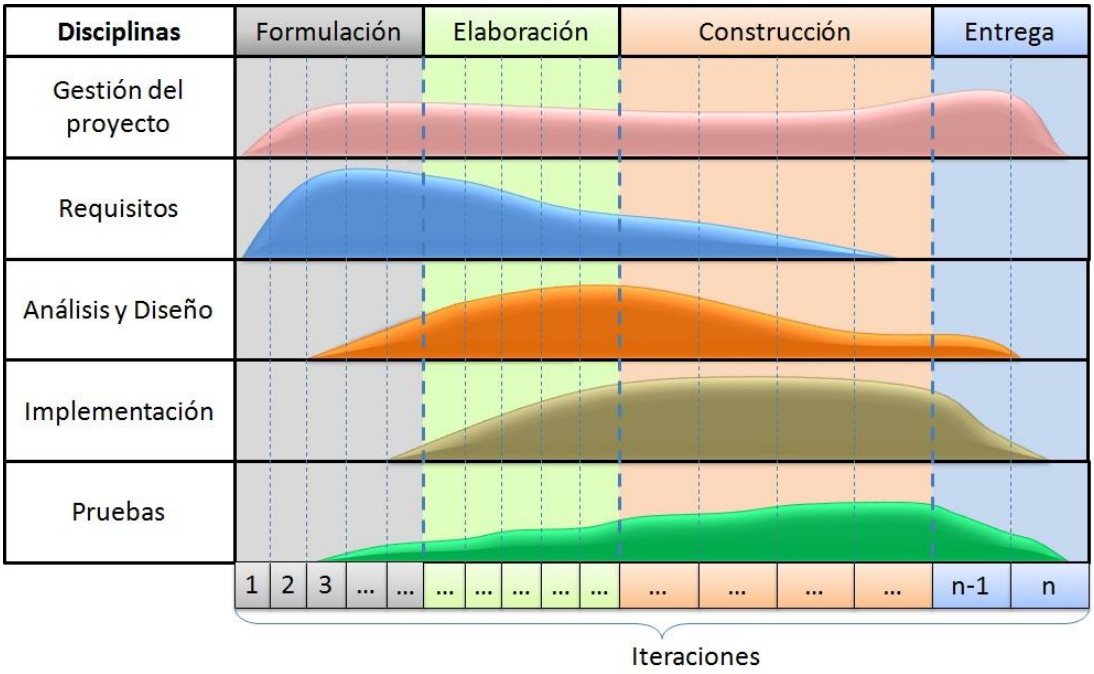


Figura 4: Fases en el desarrollo de los proyectos de IHEGames[1]

Cada fase dentro del desarrollo de los proyectos de IHEGames consta de unos objetivos y unos artefactos concretos que pueden llevarse a cabo dependiendo de la naturaleza del

[1] *Imagen obtenida de http://escritura.proyectolatin.org/gestion-de-proyectos-de-software/ejemplos-de-procesos/*

proyecto y que pueden refinarse en fases posteriores. A continuación se describe el objetivo de cada una de las fases y sus artefactos:

- Formulación: Esta fase tiene como propósito definir y acordar el alcance del proyecto con los patrocinadores, identificar los riesgos asociados al proyecto, proponer una visión muy general de la arquitectura de software y producir el plan de las fases y el de iteraciones posteriores. Se desarrollan una serie de artefactos en esta fase:
    - Diagrama de casos de uso
    - Especificación de requisitos.
- Elaboración: se realiza la especificación de los casos de uso que permiten definir la arquitectura base del sistema, el primer análisis del dominio del problema y se diseña la solución preliminar. Se desarrollan una serie de artefactos en esta fase:
    - Diagrama de clases
    - Modelo de entidad relación (E-R)
    - Diagramas de secuencias
- Construcción: el propósito de esta fase es completar la funcionalidad del sistema. Los artefactos de esta etapa son los siguientes:
    - Pruebas de los casos de uso desarrollado
- Entrega: El propósito de esta fase es asegurar que el software esté disponible para los usuarios finales

Cada fase puede realizarse en una o más iteraciones. A lo largo de cada iteración se realiza una serie de tareas consecutivas agrupadas en las siguientes disciplinas: gestión del proyecto, requisitos, análisis y diseño, implementación y pruebas.

### 4.1.2 Etapas en un ciclo del proyecto

IHEGames establece que este desarrollo de cuatro fases se puede repetir a lo largo de uno o más ciclos que se descomponen a su vez en varias etapas. El objetivo de cada ciclo consiste en desarrollar una nueva versión del producto a partir de nuevas especificaciones. A partir del primer ciclo, dichas especificaciones se definen teniendo en cuenta los resultados obtenidos en pruebas con usuarios reales.

En la siguiente figura se muestran las etapas que componen un ciclo de un proyecto.


Figura 5: Etapas a lo largo de un ciclo de los proyectos de IHEGames

Al inicio de cada proyecto se definen unas especificaciones iniciales para la primera versión del producto a desarrollar. Una vez desarrollada la primera versión se procede a la validación del producto mediante pruebas con usuarios reales. De las pruebas realizadas se recogen las valoraciones de los usuarios y se definen unas nuevas especificaciones para la siguiente versión. Este proceso se continúa hasta obtener la versión final deseada.

Este proyecto concreto ha consistido en un primer ciclo, en el cual se han desarrollado una primera versión de K-Tan a partir de unas especificaciones iniciales y se ha validado mediante pruebas con usuarios reales.

Dado que el ciclo de vida de los proyectos establecido por IHEGames no contempla la documentación académica que es inherente a este proyecto, se establece una cuarta etapa al final del ciclo (tras la validación mediante las pruebas con usuarios) para la redacción de la memoria y la defensa del proyecto.

# 5

# Arquitectura

A lo largo de este capítulo se describe la arquitectura general que se ha diseñado para integrar K-Tan y los futuros juegos de IHEGames junto con el resto de sistemas y servicios de IHEGames.

Por otro lado, se describen las interfaces que proporcionan los servicios web que se han integrado en K-Tan a lo largo de este proyecto, tales como los servicios de cuentas de usuario, Elo y Honor.

## 5.1 Arquitectura

Para la arquitectura de K-Tan, y a su vez para la de los futuros juegos de IHEGames, se establece una arquitectura cliente-servidor. Esta arquitectura cliente-servidor consta de dos tipos de clientes: clientes Android y clientes Web. Los clientes Android son Apps que se ejecutan independientemente desde un dispositivo Android. En cambio, los clientes Web son aplicaciones web accesibles desde el Portal D20 y que se ejecutan desde un navegador web.

En la siguiente figura se muestra la arquitectura general de K-Tan junto con el resto de sistemas de IHEGames, así como los protocolos mediante los que se comunican los diferentes módulos.

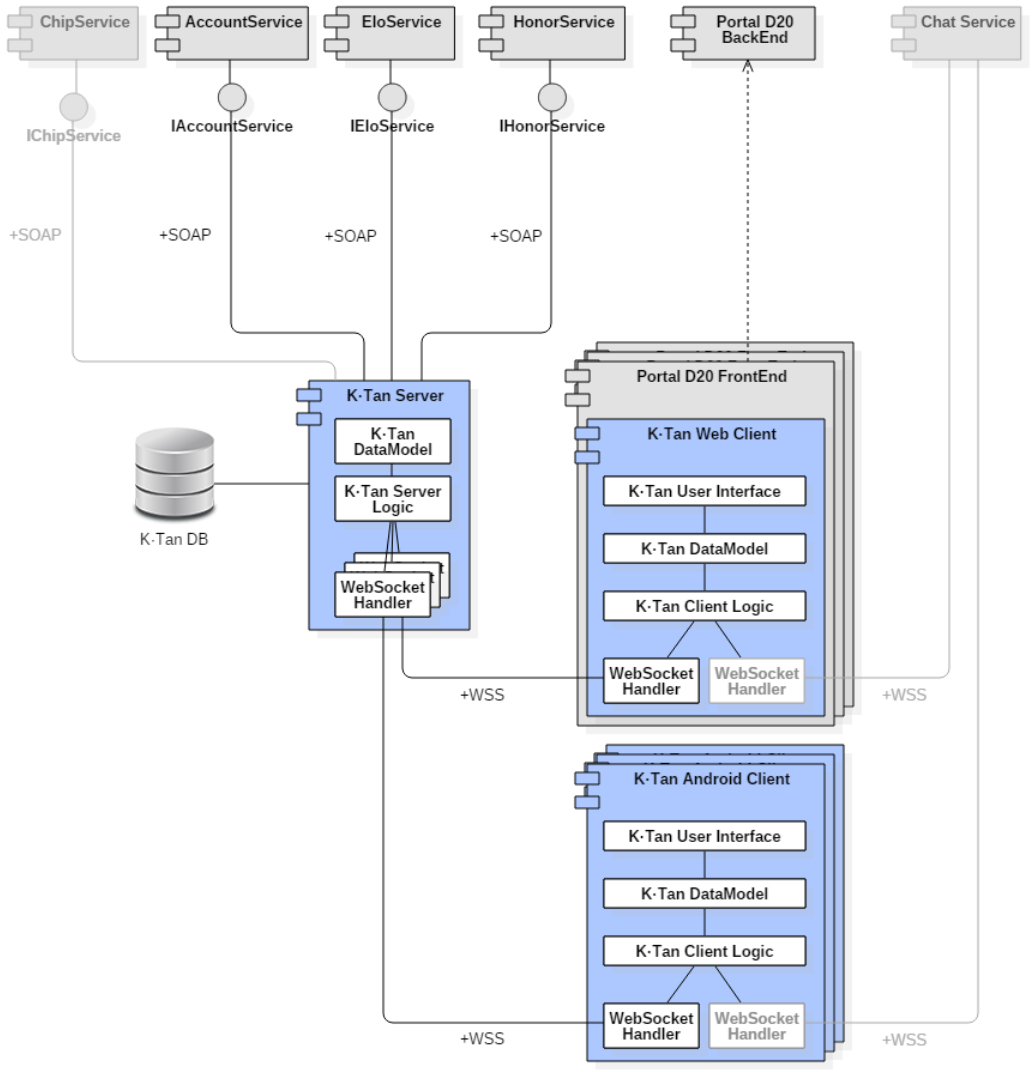


Figura 6: Arquitectura general de K-Tan

Dado que los clientes Web de K-Tan se ejecutan desde un navegador web, se han de tener en cuenta algunas de las limitaciones de conectividad que establecen los navegadores web; concretamente el hecho de que los navegadores no permiten realizar una conexión directa mediante sockets. Por lo tanto, se decide que las comunicaciones entre los clientes, tanto Android como Web, y el propio servidor del juego se realicen mediante el protocolo WSS o *WebSocket Secure*, ya que WSS está especialmente diseñado para ser implementada, entre otros, en navegadores y servidores web, y proporciona un canal de comunicación bidireccional sobre un único socket TCP.

Para la comunicación entre el servidor y los clientes de K-Tan se define un sencillo protocolo de envío y ejecución de acciones o `Action`. Estas acciones son clases que se serializan en formato JSON y se envían a través de WSS. Una vez en el destino, las acciones se deserializan, se verifican y se ejecuta su método `Execute`. Estas acciones, constituyen la interfaz de comunicación entre ambas partes y permiten una comunicación del estilo a llamadas de procedimientos remotos. Más adelante, en el capítulo de *Análisis y diseño* se concretan los detalles sobre estas acciones.

En principio, dado que al servicio de chat aspira a ser accesible no solo desde los propios juegos sino que también desde el propio Portal D20, se definen las comunicaciones con este servicio con el mismo protocolo WSS y se plantea, para su futura implementación, una comunicación idéntica a la definida entre el servidor y los clientes de K-Tan.

Por otro lado, se establece que la comunicación entre el servidor de K-Tan y el resto de servicios web de IHEGames se realice mediante el protocolo SOAP (Simple Object Access Protocol), del que hace uso el modelo WCF que implementan dichos servicios web.

## 5.2 Los servicios de IHEGames

Cada servicio de IHEGames proporciona una interfaz y un modelo de datos. Cada interfaz define las funcionalidades que se pueden invocar en cada uno de los servicios, mientras que el modelo de datos proporcionado establece la estructura de los datos intercambiados y que es fundamental para la comunicación con dichos servicios.

En los siguientes apartados se describe la interfaz y el modelo de datos de cada uno de los servicios de IHEGames que se ha integrado en este ciclo del proyecto.

### 5.2.1 Servicio de cuentas de usuario

El servicio de cuentas de usuario ofrece una interfaz muy extensa. Sin embargo, la mayoría de funcionalidades están restringidas para las aplicaciones como K-Tan que no requieren de estas otras funcionalidades y que por lo tanto no disponen de los permisos necesarios para invocarlas. Gracias a este servicio, K-Tan puede identificar a un usuario y obtener los datos de perfil asociados a dicho usuario.

La identificación de usuario en la versión actual de K-Tan se puede realizar mediante un email o nombre de usuario y una contraseña proporcionados por el propio usuario o, en el caso de la versión web de K-Tan, mediante un identificador de usuario y un Token, ambos obtenidos a través del Portal D20 donde el usuario se ha tenido que identificar previamente.

El servicio proporciona otro método adicional para identificar a un usuario mediante proveedores externos como Google, Facebook, o Twitter. Sin embargo, en esta primera versión de K-Tan no se implementa este método de identificación de los usuarios.

A continuación se detallan las funcionalidades que el servicio de cuentas de usuario ofrece para K-Tan, así como su modelo de datos:

```
    public interface IAccountService {
        User checkUserPassword(string email, string password);
        User checkUserGameToken(uint userID, string token);
        User checkUserOauth(string provider, string providerUserId,
string email, string userName);
    }
```

Figura 7: Interfaz del servicio de cuentas de usuario

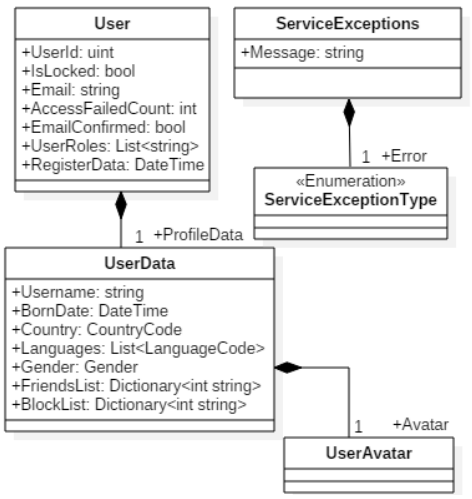


Figura 8: Modelo de datos del servicio de cuentas de usuario

### 5.2.2 Servicio de Honor

En el caso del servicio de Honor todas sus funcionalidades están disponibles para K-Tan. Mediante estas funcionalidades, K-Tan puede obtener los datos relacionados con el Honor de un usuario concreto o de varios usuarios. Por otro lado, K-Tan puede enviar un reporte con los resultados de una partida para que el servicio recalcule el Honor de los usuarios de la partida y devuelva los nuevos datos de cada uno.

Este servicio ofrece otras funcionalidades adicionales, como por ejemplo obtener un top de los 10 usuarios como mayor Honor. Sin embargo, en esta versión de K-Tan aún no se hace uso de ellas.

A continuación se detallan las funcionalidades que el servicio de Honor ofrece para K-Tan, así como su modelo de datos:

```
public interface IVGHonorService {
    List<Honor> addGameReport(GameReport newGameReport);
    Honor getUserHonor(uint userID);
    Honor getUserWeightedHonor(uint userID);
    List<Honor> getUsersHonor(List<uint> userList);
    List<Honor> getUsersWeightedHonor(List<uint> userList);
    List<Honor> getTopTenHonorableUsers();
    List<Honor> getTopTenWeightedHonorableUsers();
}
```

Figura 9: Interfaz del servicio de Honor

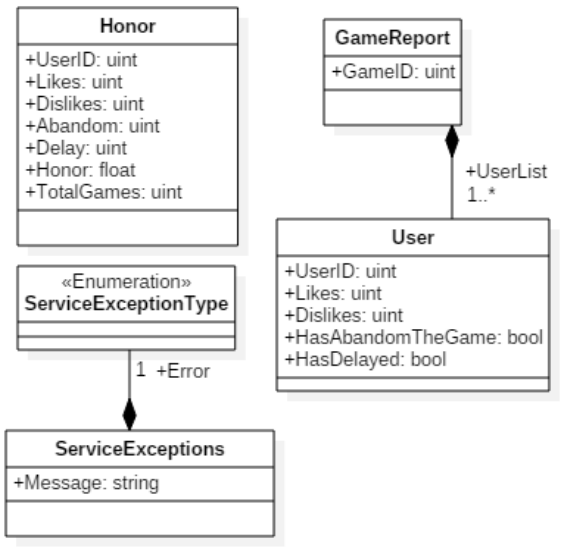


Figura 10: Modelo de datos del servicio de Honor

### 5.2.3 Servicio de Elo

Al igual que en el servicio de Honor, en el servicio de Elo todas sus funcionalidades están disponibles para K-Tan. Mediante estas funcionalidades, K-Tan puede obtener los datos relacionados con el Elo de un usuario concreto o de varios usuarios. Por otro lado, K-Tan puede enviar un reporte con los resultados de una partida para que el servicio recalcule el Elo de los usuarios de la partida y devuelva los nuevos datos de cada uno.

Este servicio ofrece otras funcionalidades adicionales, como por ejemplo obtener un reporte enviado Sin embargo, en esta versión de K-Tan aún no se hace uso de ellas.

A continuación se detallan las funcionalidades que el servicio de Elo ofrece para K-Tan, así como su modelo de datos:

```
public interface IELOService{
    UserELO getUserELO(uint userID);
    List<UserELO> getUsersELO(List<uint> usersIDS);
    List<UserELO> addGameReport(GameReport newGameReport);
    GameReport getGameReport(uint gameID);
}
```

Figura 11: Interfaz del servicio de Elo

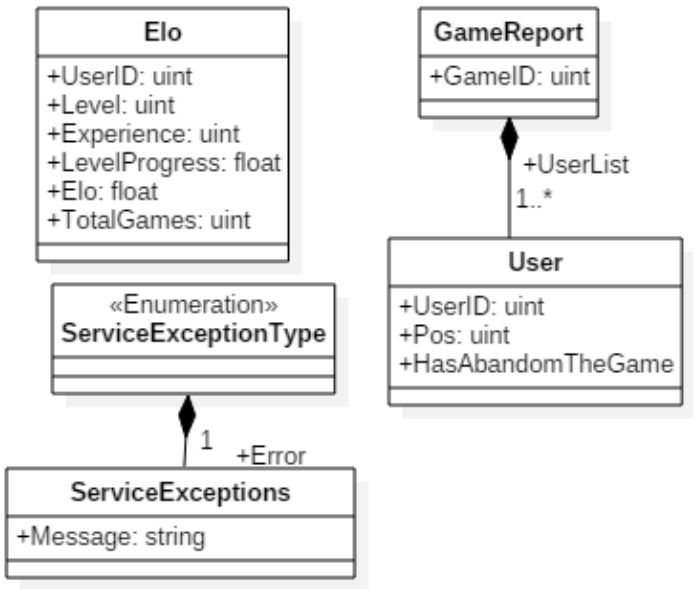


Figura 12: Modelo de datos del servicio de Elo

# 6

# Reglamento de juego

En este capítulo se explica el reglamento básico de K-Tan, extraído del reglamento de *Los Colonos de Catan*[9]. Se describen unas nociones básicas del juego así como los términos y conceptos más importantes que pueden surgir después a lo largo de esta memoria.

Inicialmente se describen algunos conceptos básicos para ayudar a entender el resto del reglamento y el objetivo principal en las partidas de K-Tan. Posteriormente, se detallan los diferentes componentes que se pueden hallar en el juego y finalmente, se describen las reglas generales del juego.

## 6.1 Conceptos básicos

K-Tan comparte muchos conceptos básicos que se pueden encontrar en los juego de mesa en general. Para aquellos que no estén demasiado familiarizados con estos conceptos básicos pueden tomar como referencia un juego de mesa quizás más conocido como el Monopoli. Al igual que éste, K-Tan también se desarrolla en un tablero. Igualmente, se desarrolla a lo largo de turnos en los que se lanzan dados. Y al igual que en el Monopoli los jugadores obtienen dinero para construir casas y hoteles, en K-Tan obtienen recursos con los cuales construir las carreteras, poblados y ciudades.

## 6.2 Objetivo

Las partidas de K-Tan son de 2 a 4 jugadores. El objetivo en una partida de K-Tan es construir poblados, ciudades y carreteras sobre un tablero formado por terrenos hexagonales y que es distinto en cada partida, mientras se van acumulando varios tipos de cartas y recursos. Todos estos elementos proporcionan distintas puntuaciones, ganando la partida el primer jugador que llega a los diez puntos de victoria.

## 6.3 Componentes

El tablero se compone por 19 terrenos hexagonales que se distribuyen al azar en una isla hexagonal. Hay 6 tipos de terreno. Los 5 tipos de terreno principales producen recursos y se les coloca, en base a su posición en la isla, una de las fichas numeradas que tienen valores del 2 al 12, excluyendo el 7. El último tipo de terreno, el desierto, no produce ningún recurso.

Los **recursos** existentes son la **madera**, la **arcilla**, la **lana**, los **cereales** y el **mineral**. Estos recursos se representan mediante cartas y sirven para construir y comprar los diferentes elementos que permiten obtener puntos.

Los elementos a construir consisten en **carreteras**, **poblados** y **ciudades**, cada uno con un coste concreto de recursos. Cada una de ellas se representa por una **figura** de juego concreta. Cada jugador dispone de 15 carreteras, 5 poblados y 4 ciudades y se distinguen entre las figuras del resto de jugadores por su color, rojo, blanco, naranja o azul.


Las carreteras se colocan los **caminos**, es decir, en las aristas de los terrenos hexagonales. Los poblados y ciudades se colocan en las **intersecciones** de los caminos, o lo que es lo mismo, los vértices de los terrenos.

Además de construir, los jugadores pueden comprar **cartas de desarrollo**. Existen 25 cartas de desarrollo y son de 5 tipos diferente: cartas de **caballero**, cartas de **punto**, cartas de **monopolio**, cartas de **construcción de carreteras** y cartas de **descubrimiento**.

Existen unos componentes adicionales en el tablero de juego. Hay 9 **puertos**, 4 de ellos genéricos y los otros cinco relacionados cada uno con uno de los recursos del juego. Los puertos se colocan al azar en caminos de costa concretos. Por otro lado, hay una figura del **ladrón**, que inicialmente se coloca sobre el terreno de desierto.

Hay dos cartas especiales, la **gran ruta comercial** y el **gran ejercito de caballería**. Cada una de estas cartas se puede asignar a un jugador que cumpla unas determinadas condiciones, proporcionándole dos puntos de victoria adicionales.

Por último, hay dos **dados** de 6 caras con valores del 1 al 6.

## 6.4 Reglas básicas

Una vez preparado el tablero y escogido al azar el orden y color de los jugadores, hay una primera fase de preparación. Tras concluir esta fase, la partida se desarrolla a lo largo de la fase de turnos.

### 6.4.1 Fase de preparación

Esta fase se compone de dos rondas. En cada ronda, todos los jugadores, siguiendo el orden establecido en la primera ronda y en orden inverso en la segunda, colocan un poblado y una carretera en el tablero. Los poblados deben colocarse siempre en una intersección vacía que no tenga ningún otro poblado en las intersecciones adyacentes. La carretera se coloca en un camino adyacente al poblado.

Al colocar el segundo poblado, cada jugador recibe un recurso de cada terreno adyacente a éste. Los recursos y cartas se mantienen ocultos del resto de jugadores.

### 6.4.2 Fase de turnos

Esta fase consiste en un número indeterminado de turnos. Comenzando por el jugador inicial, los jugadores realizan acciones a lo largo de un turno y cuando terminar, es el turno del siguiente jugador, siguiendo siempre el orden establecido al inicio de la partida.

Cada turno los jugadores pueden realizar unas acciones determinadas.

#### 6.4.2.1 Lanzar dados

El jugador debe lanzar una vez los dados al comienzo del turno. Los resultados de ambos dados se suman para obtener el resultado de la tirada. Los jugadores obtienen un recurso por cada poblado adyacente a un terreno con un número igual al resultado de la tirada, y dos recursos por cada ciudad adyacente a dichos terrenos. Los recursos obtenidos dependen del tipo de recurso que produce el terreno.

Si el resultado de los dados es un 7, primero, todos aquellos jugadores con más de 7 recursos en la mano deben descartarse inmediatamente de la mitad de sus recursos,

redondeando hacia abajo. Después, el jugador que ha lanzado los dados, debe mover la figura del ladrón y colocarlo sobre otro terreno distinto.

Un terreno que contenga al ladrón no produce ningún recurso aunque el resultado de los dados coincida con su número.

Una vez movido al ladrón, el jugador puede escoger a uno de los jugadores que tenga un poblado o ciudad adyacente al terreno donde se haya movido al ladrón y robarle un recurso al azar.

#### 6.4.2.2 Construir

Una vez lanzados los dados y resuelto su resultado, un jugador puede construir tanto como quiera. El jugador descarta los recursos necesarios coge una de las figuras correspondientes que tenga disponibles y la sitúa en un lugar válido del tablero.

El coste de las carreteras es de un recurso de madera y otro de arcilla, y deben construirse siempre en un camino vacío adyacente a otra carretera, poblado o ciudad propia.

El coste de los poblados es de un recurso de madera, uno de arcilla, uno de lana y otro de cereales. Los poblados deben construirse siempre en una intersección vacía, que no tenga ningún poblado o ciudad en las intersecciones adyacentes y que tenga en algún camino adyacente una carretera propia.

El coste de las ciudades es de tres recursos de mineral y dos de cereales. Las ciudades se construyen sobre un poblado. El jugador recupera la figura del poblado y situa una figura de ciudad en su lugar.

Un jugador no puede construir si no dispone de la figura necesaria para colocar en el tablero.

Además de construir, el jugador puede comprar cartas de desarrollo. Las cartas de desarrollo se barajan en un mazo que se sitúa boca abajo al inicio de la partida. Al comprar una carta de desarrollo, el jugador descarta los recursos necesarios y coge la primera carta del mazo.

El coste de las cartas de desarrollo es de un recurso de mineral, uno de lana y otro de cereales. Las cartas de desarrollo se mantienen ocultas del resto de jugadores.

#### 6.4.2.3 Desarrollo

Un jugador puede, incluso antes de tirar los dados, usar cada turno una de sus cartas de desarrollo. Las cartas compradas durante ese mismo turno no se pueden utilizar. Cada carta de desarrollo se resuelve de una manera determinada dependiendo del tipo de carta y una vez resuelta, ésta se descarta.

Caballero: el jugador mueve el ladrón a otro terreno y roba un recurso a un jugador que tenga un poblado o ciudad adyacente a dicho terreno.

Monopolio: el jugador escoge un tipo de recurso y el resto de jugadores deben darle todas las cartas de recurso que tengan de ese tipo.

Construcción de carreteras: el jugador puede construir dos carreteras inmediatamente sin coste alguno.

Descubrimiento: el jugador obtiene dos cartas de recurso a su elección.

Punto: estas cartas no se usan. Proporcionan un punto de victoria adicional, y solo se muestran al final de la partida.

#### 6.4.2.4 Intercambio

Una vez lanzados los dados y resuelto su resultado, el jugador puede realizar intercambios marítimos mediante los cuales realiza cambios directos de recursos. El jugador puede realizar tantos cambios como quiera por turno y se realizan a razón de 4:1, es decir, el jugador puede obtener un recurso cualquiera a cambio de descartarse de 4 cartas de recurso iguales.

En caso de que el jugador tenga un poblado o ciudad adyacente a uno de los puertos del tablero, el jugador puede realizar cambios a menor coste.

Si el jugador tiene un poblado o ciudad adyacente a un puerto genérico, el jugador podrá realizar cualquier cambio de recursos a razón de 3:1.

Si el jugador tiene un poblado o ciudad adyacente a un puerto de un tipo de recurso, el jugador podrá realizar cambios a razón de 2:1 si descarta el mismo tipo de recursos que el del puerto.

Además de realizar intercambios marítimos, el jugador puede negociar intercambios de recursos con otros jugadores. Los otros jugadores, a su vez, son libres de ofrecerle los otros intercambios. Si el jugador acepta el intercambio, ambos jugadores intercambian los recursos acordados. En un turno se pueden realizar tantos intercambios como se quiera, siempre que el jugador del turno actual sea uno de los dos involucrados.

#### 6.4.2.5 Fin del turno

Una vez el jugador haya lanzado los dados y resuelto su resultado, puede terminar su turno cuando quiera.

### 6.4.3 Fin de la partida

El juego termina cuando un jugador durante su turno alcanza los 10 o más puntos de victoria, siendo este jugador el ganador de la partida.

Cada poblado en el tablero proporciona al jugador 1 punto de victoria.

Cada ciudad en el tablero proporciona al jugador 2 puntos de victoria.

Cada carta de punto proporciona al jugador 1 punto de victoria.

Las cartas espaciales de Gran ruta comercial y Gran ejército de caballería proporcionan cada una 2 puntos de victoria adicionales.

La carta de Gran ruta comercial la obtiene aquel jugador que tenga la ruta más larga formada por al menos 5 carreteras consecutivas. Si otro jugador consigue una ruta más larga a lo largo de la partida, este nuevo jugador cogerá la carta de Gran ruta comercial.

La carta de Gran ejército de caballería la obtiene aquel jugador que más cartas de caballero haya usado, habiendo usado al menos 3. Si otro jugador consigue más cartas de caballero que éste, el nuevo jugador cogerá la carta de Gran ejército de caballería.

# 7

# Requisitos

A lo largo de este capítulo se describen los actores y los casos de uso identificados a partir del reglamento de K-Tan y de las especificaciones concretas definidas por IHEGames. A su vez se recogen los requisitos funcionales y no funcionales que se han identificado e implementado en el proyecto.

## 7.1 Casos de uso

En este apartado se describen, por un lado, los diferentes actores que interactúan en K-Tan, y por otro lado, los casos de uso identificados, tanto los casos de uso base de los juegos de IHEGames entre los que se encuentra K-Tan, como los casos de uso específicos de K-Tan.

### 7.1.1 Actores

Todos los actores que interactúan en K-Tan son usuarios del mismo. Sin embargo, se ha realizado una distinción de los usuarios en base a su situación en el juego.

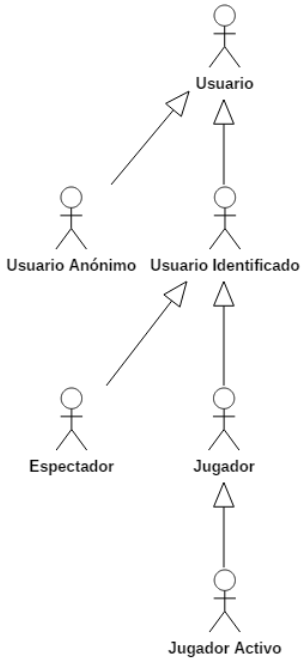


Figura 13: Actores de K-Tan

| | |
|---|---|
| Usuario | Cualquier usuario del juego |
| Usuario Anónimo | Usuario que aún no se ha identificado correctamente |
| Usuario Identificado | Usuario que se ha identificado correctamente |
| Espectador | Usuario identificado que está unido a una partida como espectador |
| Jugador | Usuario identificado que está unido a una partida como jugador |
| Jugador Activo | Jugador del cual es el turno para jugar. |

Tabla 1: Actores de K-Tan

### 7.1.2 Casos de uso base de los juegos de IHEGames

Por un lado, se han identificado los casos de uso base, no los específicos de K-Tan, sino los de cualquiera de los juegos de IHEGames. La identificación de estos casos de uso se ha realizado en base a las especificaciones dadas por IHEGames.

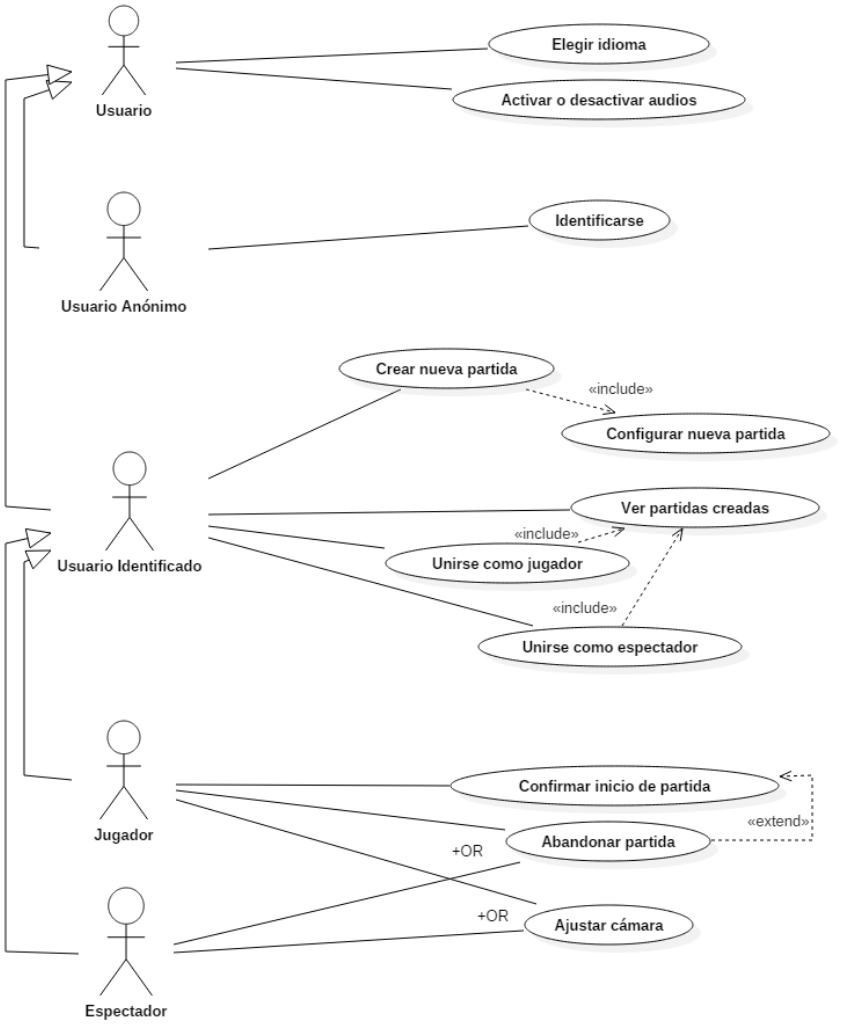


Figura 14: Casos de uso base de los juegos de IHEGames

En la siguiente tabla se detallan cada uno de los casos de uso base de los juegos de IHEGames desarrollados en este proyecto.

| Elegir idioma |
| --- |
| **Actores**: Usuario<br>**Precondiciones**: Ninguna<br>**Flujo básico**: El usuario escoge uno de los idiomas disponibles (Español o Inglés) y los textos de la interfaz de usuario cambian al idioma escogido<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Ninguno |
| **Activar o desactivar audios** |
| **Actores**: Usuario<br>**Precondiciones**: Ninguna<br>**Flujo básico**: El usuario puede seleccionar si activar o desactivar la música y los efectos audio<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Ninguno |
| **Identificarse** |
| **Actores**: Usuario anónimo<br>**Precondiciones**: Ninguna<br>**Flujo básico**: El usuario introduce el email o nombre de usuario y la contraseña. Si las Credenciales son correctas el usuario se identifica correctamente.<br>Si el usuario está de antemano unido como jugador a una partida iniciada, el usuario entra directamente a la partida.<br>**Flujos alternativos**: Si la contraseña o el email o nombre de usuario introducido son incorrectos el usuario recibe una notificación.<br>**Casos de uso relacionados**: Ninguno |
| **Crear nueva partida** |
| **Actores**: Usuario identificado<br>**Precondiciones**: El usuario no está unido a ninguna partida.<br>**Flujo básico**: El usuario selecciona la opción de crear una nueva partida. Tras configurar la partida, la partida se crea y el usuario se une como jugador a la partida creada automáticamente.<br>**Flujos alternativos**: Si el usuario cancela la configuración de la nueva partida, la creación de la nueva partida finaliza sin crear una nueva partida.<br>**Casos de uso relacionados**: Incluye el caso de uso *Configurar nueva partida* |

| Configurar nueva partida |
| --- |
| **Actores**: Usuario identificado<br>**Precondiciones**: El usuario no está unido a ninguna partida.<br>**Flujo básico**: El usuario selecciona el número de jugadores de la partida, el acceso (cualquiera/solo amigos/solo invitados), si se permiten espectadores, si es una partida competitiva, si se limita el acceso por Elo, si se limita el acceso por Honor, la velocidad de partida (Lenta/Normal/Rápida) y el número de fichas apostadas. El usuario confirma la configuración seleccionada y la partida se configura.<br>**Flujos alternativos**: El usuario puede cancelar la configuración de la nueva partida en cualquier momento.<br>**Casos de uso relacionados**: Incluida en el caso de uso *Crear nueva partida* |
| **Ver partidas creadas** |
| **Actores**: Usuario identificado<br>**Precondiciones**: El usuario no está unido a ninguna partida iniciada.<br>**Flujo básico**: El usuario puede visualizar todas las partidas existentes que no hayan finalizado. El usuario puede seleccionar cada una de ellas para visualizar la configuración de la partida así como los jugadores y espectadores unidos a la partida.<br>**Flujos alternativos**: Si la contraseña o el email o nombre de usuario introducido son incorrectos el usuario recibe una notificación.<br>**Casos de uso relacionados**: Incluida en los caso de uso Unirse como jugador y Unirse como espectador |
| **Unirse como jugador** |
| **Actores**: Usuario identificado<br>**Precondiciones**: El usuario no está unido a ninguna partida.<br>**Flujo básico**: El usuario selecciona una partida y la opción de unirse a ella como jugador. Si el usuario cumple con los requisitos para unirse a la partida y el número de jugadores de la partida no está completo, el usuario se une a la partida como jugador.<br>Si el número de jugadores de la partida se completa, los jugadores de la partida reciben una notificación para confirmar el inicio de la partida.<br>**Flujos alternativos**: Si el usuario no cumple con los requisitos para unirse a la partida o el número de jugadores de la partida está completo, el usuario no se une a la partida.<br>**Casos de uso relacionados**: Incluye el caso de uso *Ver partidas creadas*. |

| Unirse como espectador |
| --- |
| **Actores**: Usuario identificado<br>**Precondiciones**: El usuario no está unido a ninguna partida.<br>**Flujo básico**: El usuario selecciona una partida y la opción de unirse a ella como espectador. Si la partida permite espectadores, el usuario se une a la partida como espectador.<br>Cuando la partida se inicie, el usuario entrará automáticamente en la partida y podrá visualizar el proceso de la partida. Si la partida a la que se une ya está iniciada, el usuario entra directamente en la partida.<br>**Flujos alternativos**: Si la partida no permite espectadores, el usuario no se une a la partida.<br>**Casos de uso relacionados**: Incluye el caso de uso *Ver partidas creadas*. |
| **Confirmar inicio de partida** |
| **Actores**: Jugador<br>**Precondiciones**: El usuario recibe una notificación de confirmación de inicio de partida.<br>**Flujo básico**: El usuario selecciona la opción de confirmar inicio.<br>Cuando todos los jugadores confirman el inicio de partida, la partida se inicia y el usuario entra automáticamente en la partida.<br>**Flujos alternativos**: Si alguno de los jugadores abandona la partida, voluntariamente o automáticamente si transcurren 30 segundos sin confirmar el inicio de partida, la confirmación de inicio de partida se cancela.<br>**Casos de uso relacionados**: Extiende el caso de uso *Abandonar partida.* |
| **Abandonar partida** |
| **Actores**: Jugador o Espectador<br>**Precondiciones**: Ninguna<br>**Flujo básico**: El usuario selecciona la opción de abandonar partida.<br>Si la partida no ha iniciado o ya ha terminado, el usuario abandona la partida directamente.<br>Si la partida está iniciada y no ha terminado, el usuario recibe una notificación indicando que el abandono de la partida afectará a su Honor. Si el usuario confirma el abandono de la partida, el usuario abandona la partida.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Puede extender del caso de uso *Confirmar inicio de partida.* |
| **Ajustar cámara** |
| **Actores**: Jugador o Espectador<br>**Precondiciones**: La partida está iniciada<br>**Flujo básico**: El usuario puede seleccionar uno de los tiros de cámara disponibles para visualizar el tablero. El usuario puede también desplazar la cámara y ajustar el zoom de la cámara dentro de los límites disponibles.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Ninguno |

Tabla 2: Casos de uso base de los juegos de IHEGames

### 7.1.3 Casos de uso específicos de K-Tan

Por otro lado, a partir del reglamento de juego de K-Tan, se han identificado los siguientes casos de uso específicos del propio K-Tan.

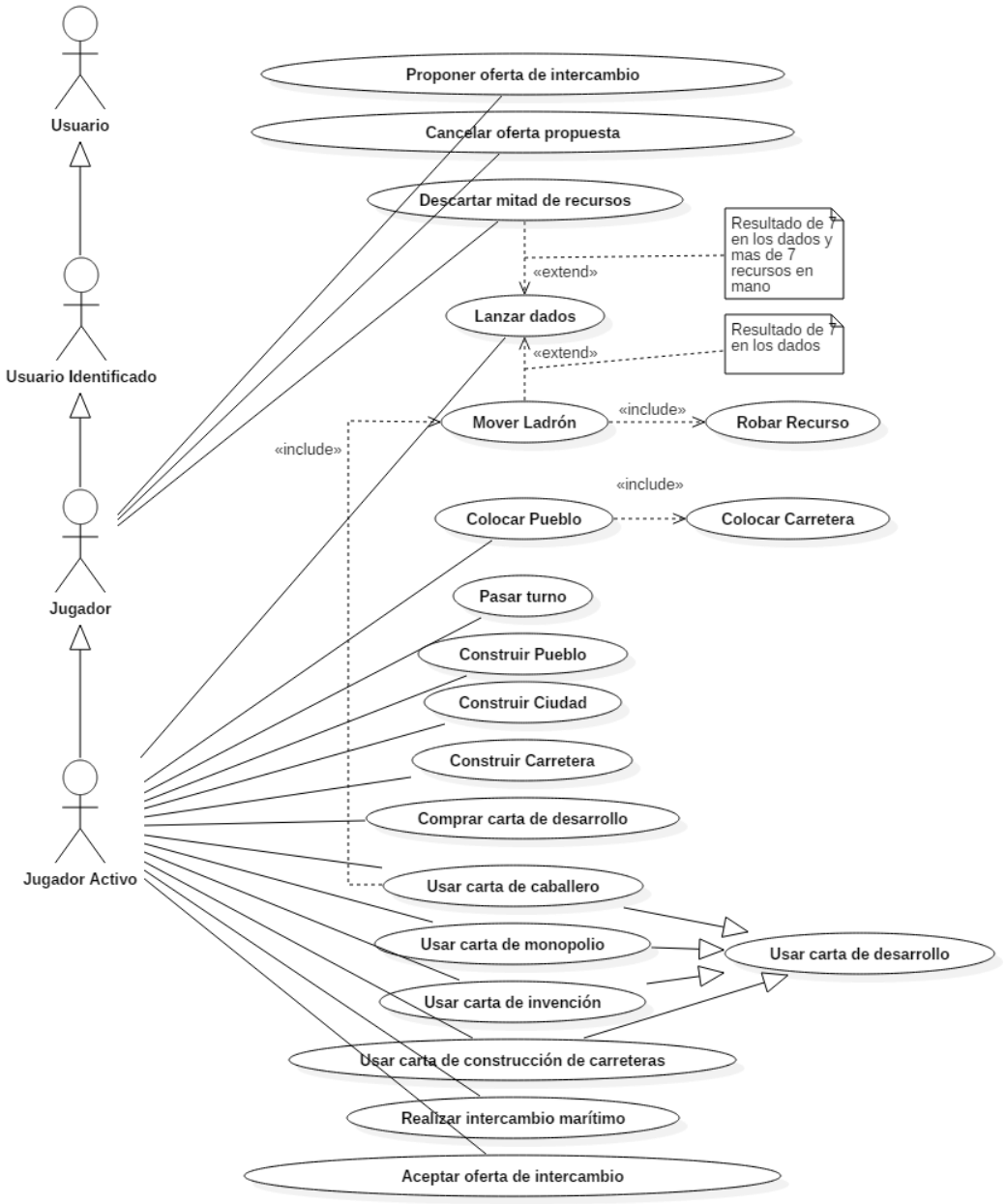


Figura 15: Casos de uso específicos de K-Tan

Se detallan también en la siguiente tabla cada uno de los casos de uso específicos de K-Tan:

| Proponer oferta de intercambio |
| --- |
| **Actores**: Jugador<br>**Precondiciones**: La partida está iniciada y se ha concluido la fase de preparación.<br>**Flujo básico**: El usuario selecciona la opción de proponer oferta y escoge que recursos ofrece de entre los recursos en su mano, y que recursos quiere a cambio. Tras confirmar la oferta, todos los jugadores visualizan el intercambio propuesto.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Ninguno |
| **Cancelar oferta propuesta** |
| **Actores**: Jugador<br>**Precondiciones**: La partida está iniciada y se ha concluido la fase de preparación y el jugador ha propuesto un intercambio.<br>**Flujo básico**: El usuario selecciona la opción de cancela la oferta. El intercambio propuesto se cancela y se deja de visualizar.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Ninguno |
| **Descartar mitad de recursos** |
| **Actores**: Jugador<br>**Precondiciones**: La partida está iniciada y se ha concluido la fase de preparación y el jugador debe descartarse de la mitad de los recursos.<br>**Flujo básico**: El usuario selecciona la mitad, redondeando hacia abajo, de los recursos a descartar de su mano. Tras confirmar el descarte, los recursos seleccionados se descartan.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Extiende del caso de uso *Lanzar dados*, si el resultado de los dados es de 7 y el usuario tiene más de 7 recursos en mano. |
| **Lanzar dados** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se ha concluido la fase de preparación y el jugador aún no ha lanzado los dados en su turno.<br>**Flujo básico**: El usuario selecciona la opción de lanzar los dados y los dados se lanzan para obtener un resultado. Todos los jugadores reciben, por cada poblado adyacente a una casilla de terreno con un número igual al resultado, un recurso del tipo indicado en la casilla de terreno. De igual manera, todos los jugadores reciben 2 recursos por cada ciudad adyacente a dichas casillas de terrenos.<br>Si el resultado obtenido en los dados es de 7, primero cada jugador con más de 7 recursos en mano debe descartarse de la mitad de los recursos. A continuación, el usuario podrá robar un recurso a un jugador tras mover el ladrón.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Extiende en los casos de uso *Mover Ladrón* si el resultado es de 7, y en *Descartar mitad de recursos* si además el usuario tiene más de 7 recursos en mano. |

| Mover ladrón |
| --- |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se ha concluido la fase de preparación y el jugador bien ha obtenido un 7 en los dados o bien ha usado una carta de caballero<br>**Flujo básico**: El usuario escoge una casilla de terreno distinta a la que se encuentra el ladrón. El ladrón se mueve a la casilla de terreno seleccionada.<br>**Flujos alternativos**: Ninguno.<br>**Casos de uso relacionados**: Incluye el caso de uso *Robar Recurso*. Se incluye en el caso de uso *Usar carta de caballero* y extiende de *Lanzar dados* si el resultado en los dados es de 7. |
| Robar Recurso |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se ha concluido la fase de preparación y se ha movido el ladrón.<br>**Flujo básico**: Tras haber movido al ladrón, el usuario selecciona a algún otro jugador que tenga uno o más poblados o ciudades alrededor de la casilla de terreno donde se haya movido el ladrón. El jugador seleccionado le entrega un recurso al azar de su mano.<br>**Flujos alternativos**: Si no hay poblados ni ciudades alrededor de la casilla de terreno al que se ha movido el ladrón o ninguno de los jugadores con algún poblado o ciudad adyacente tiene cartas de recurso, el usuario no roba ningún recurso.<br>**Casos de uso relacionados**: Se incluye en el caso de uso *Mover Ladrón*. |
| Colocar poblado |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación.<br>**Flujo básico**: El jugador selecciona una intersección vacía que no tenga ningún poblado en alguna de las intersecciones adyacentes a ésta. Se coloca un poblado en la intersección seleccionada y el jugador obtiene un punto de victoria. Tras colocar el poblado el jugador coloca una carretera.<br>Si es el segundo poblado colocado, el jugador recibe un recurso de cada casilla de terreno adyacente al poblado.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Incluye el caso de uso *Colocar carretera*. |
| Colocar carretera |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha colocado un poblado.<br>**Flujo básico**: Al colocar un poblado, el jugador selecciona un camino adyacente a dicho poblado. Se coloca una carretera en el camino seleccionado.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Se incluye en el caso de uso *Colocar poblado*. |

| Pasar turno |
| --- |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona la opción de pasar turno. El turno se pasa al siguiente jugador que será ahora el jugador activo.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Se incluye en el caso de uso *Colocar poblado*. |
| **Construir poblado** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona la opción de construir y escoge construir un poblado. A continuación, selecciona una intersección vacía conectada a una de sus carreteras y que no tenga ningún otro poblado o ciudad en ninguna de sus intersecciones adyacentes. Se coloca un poblado en la intersección seleccionada y el jugador pierde de la mano un recurso de madera, uno de arcilla, uno de cereales y otro de lana. Finalmente el jugador obtiene un punto de victoria.<br>**Flujos alternativos**: Si el jugador no dispone de los recursos suficientes, o no dispone de poblados disponibles o ninguna intersección cumple las características necesarias, no se coloca ningún poblado ni se pierde ningún recurso.<br>**Casos de uso relacionados**: Ninguno. |
| **Construir ciudad** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona la opción de construir y escoge construir una ciudad. A continuación, selecciona uno de sus poblados en el tablero. Se retira el poblado y se coloca una ciudad en su lugar y el jugador pierde de la mano tres recurso de piedra y dos de cereales. Finalmente el jugador obtiene un punto de victoria.<br>**Flujos alternativos**: Si el jugador no dispone de los recursos suficientes, o no dispone de ciudades disponibles o no tiene poblados en el tablero, no se coloca ninguna ciudad ni se retira ningún poblado ni se pierde ningún recurso.<br>**Casos de uso relacionados**: Ninguno. |

| Construir carretera |
| --- |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona la opción de construir y escoge construir una carretera. A continuación, selecciona un camino vacío conectado a uno de sus poblados, ciudades o carretera. Se coloca una carretera en el camino seleccionado y el jugador pierde de la mano un recurso de madera y otro de arcilla.<br>Si al construir la carretera, el jugador tiene una ruta formada por cinco o más carreteras consecutivas y es el jugador con la ruta más larga, obtiene la carta especial de Gran ruta comercial que le aporta dos puntos de victoria. Si la carta especial de Gran ruta comercial estaba en posesión de un jugador, ese jugador pierde los dos puntos de victoria correspondientes.<br>**Flujos alternativos**: Si el jugador no dispone de los recursos suficientes, o no dispone de carreteras disponibles o ningún camino cumple las características necesarias, no se coloca ninguna carretera ni se pierde ningún recurso.<br>**Casos de uso relacionados**: Ninguno. |
| **Comprar carta de desarrollo** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona la opción de construir y escoge comprar carta de desarrollo. El jugador recibe una carta de desarrollo del mazo de cartas de desarrollo y el jugador pierde de la mano un recurso de piedra, uno de cereales y otro de lana.<br>**Flujos alternativos**: Si el jugador no dispone de los recursos suficientes, o se ha terminado el mazo de cartas de desarrollo, no se recibe ninguna carta de desarrollo ni se pierde ningún recurso.<br>**Casos de uso relacionados**: Ninguno. |
| **Usar carta de desarrollo** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador no ha usado otra carta de desarrollo durante el turno.<br>**Flujo básico**: El jugador selecciona una de las cartas de desarrollo de su mano que no sea un punto de victoria. La carta se usa y se descarta.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: *Usar carta de caballero*, *Usar carta de monopolio*, *Usar carta de invención* y *Usar carta de construcción de carreteras* son especializaciones de este caso de uso. |

| Usar carta de caballero |
| --- |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador no ha usado otra carta de desarrollo durante el turno.<br>**Flujo básico**: Al usar una carta de caballero, el jugador mueve el ladrón y puede robar un recurso a otro jugador.<br>Si el jugador ha usa 3 o más cartas de caballero y es el jugador que más cartas de caballero ha usado, obtiene la carta especial de Gran ejercito de caballería que le aporta dos puntos de victoria. Si la carta especial de Gran ejercito de caballería estaba en posesión de un jugador, ese jugador pierde los dos puntos de victoria correspondientes.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Especialización de Usar carta de desarrollo. Incluye el caso de uso *Mover Ladrón*. |
| **Usar carta de monopolio** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador no ha usado otra carta de desarrollo durante el turno.<br>**Flujo básico**: Al usar una carta de monopolio, el jugador selecciona un tipo de recurso. El jugador obtiene todas las cartas de recurso del resto de jugadores del tipo seleccionado.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Especialización de Usar carta de desarrollo. |
| **Usar carta de construcción de carretera** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador no ha usado otra carta de desarrollo durante el turno.<br>**Flujo básico**: Al usar una carta de construcción de carretera, el jugador selecciona un primer camino vacío conectado a uno de sus poblados, ciudades o carretera. Se construye una carretera en el camino seleccionado sin coste alguno de recursos.<br>El jugador selecciona un segundo camino vacío conectado a uno de sus poblados, ciudades o carretera. Se construye una carretera en el camino seleccionado sin coste alguno de recursos.<br>Si al construir la carretera, el jugador tiene una ruta formada por cinco o más carreteras consecutivas y es el jugador con la ruta más larga, obtiene la carta especial de Gran ruta comercial que le aporta dos puntos de victoria. Si la carta especial de Gran ruta comercial estaba en posesión de un jugador, ese jugador pierde los dos puntos de victoria correspondientes.<br>**Flujos alternativos**: Si el jugador no dispone de carreteras disponibles o ningún camino cumple las características necesarias, no se coloca ninguna otra carretera<br>**Casos de uso relacionados**: Especialización de Usar carta de desarrollo. |

| Usar carta de descubrimiento |
| --- |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador no ha usado otra carta de desarrollo durante el turno.<br>**Flujo básico**: Al usar una carta de descubrimiento, el jugador selecciona dos recursos. El jugador obtiene los dos recursos seleccionados.<br>**Flujos alternativos**: Ninguno<br>**Casos de uso relacionados**: Especialización de Usar carta de desarrollo. |
| **Realizar intercambio marítimo** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona la opción de intercambio. A continuación, selecciona el tipo de recurso que entregará de entre los tipos de recurso de su mano, y los recursos que recibirá a cambio. Tras confirmar el intercambio, el jugador recibe los recursos acordados y pierde, por cada recurso recibido, cuatro recursos del tipo escogido.<br>Si el jugador tiene un poblado o ciudad adyacente a un puerto 3:1, el jugador pierde tres recursos del tipo escogido, en lugar de cuatro, por cada recurso recibido.<br>Si el jugador tiene un poblado o ciudad adyacente a un puerto 2:1 del tipo de recursos escogido, el jugador pierde dos recursos del tipo escogido, en lugar de cuatro, por cada recurso recibido.<br>**Flujos alternativos**: Si el jugador no dispone suficientes recursos del tipo escogido, no se produce ningún intercambio.<br>**Casos de uso relacionados**: Ninguno. |
| **Aceptar oferta de intercambio** |
| **Actores**: Jugador activo<br>**Precondiciones**: La partida está iniciada y se encuentra en la fase de preparación y el jugador ha lanzado los dados<br>**Flujo básico**: El jugador selecciona una oferta de intercambio de otro jugador y selecciona la opción de aceptarla. Cada uno de los dos jugadores entrega al otro los recursos indicados en la oferta de intercambio.<br>**Flujos alternativos**: Si el jugador no dispone de los recursos que se indican en la oferta, no se produce ningún intercambio.<br>**Casos de uso relacionados**: Ninguno. |

Tabla 3: Casos de uso específicos de K-Tan

## 7.2 Requisitos funcionales

Junto con los casos de uso se han identificado una serie de requisitos funcionales que se han organizado en dos grupos de requisitos. En la primera tabla se encuentran los requisitos base de cualquiera de los juegos de IHEGames, entre los que se incluye a K-Tan.

| NÚM. | PRIORIDAD | DESCRIPCIÓN |
|---|---|---|
| RF1 | 1 | El sistema debe establecer una conexión entre los clientes y el servidor de juego. |
| RF2 | 1 | El sistema debe autorizar las acciones realizadas por cada usuario. |
| RF3 | 3 | El sistema de almacenar los datos suficientes para recuperar una partida. |
| RF4 | 1 | El sistema debe sincronizar al momento, tanto en el servidor como en cada cliente de K-Tan, los cambios producidos por las acciones del juego llevadas a cabo por los usuarios. |
| RF5 | 5 | El sistema debe permitir a los usuarios la modificación del idioma de la interfaz. |
| RF6 | 5 | El sistema debe permitir la activación o desactivación de música y de efectos de audio. |
| RF7 | 1 | El sistema debe permitir la identificación por usuario/email y contraseña |
| RF8 | 1 | El sistema debe permitir la creación de nuevas partidas. |
| RF9 | 2 | El sistema debe permitir la configuración de las nuevas partidas antes de crearlas. |
| RF10 | 1 | El sistema proporciona las partidas creadas no finalizadas y sus detalles. |
| RF11 | 1 | El sistema proporciona los usuarios conectados y sus perfiles |
| RF12 | 1 | El sistema permite unirse a una partida como jugador. |
| RF13 | 4 | El sistema permite unirse a una partida como espectador. |
| RF14 | 1 | El sistema, al completarse el número de jugadores de una partida, debe mostrar a cada jugador de la partida una notificación de confirmación de inicio. |
| RF15 | 5 | El sistema debe hacer abandonar la partida a todo jugador que no confirme el inicio de partida antes de 30 segundos |
| RF16 | 3 | El sistema debe permitir abandonar una partida. Si la partida ya está iniciada, el sistema debe registrar el abandono para notificárselo al servicio de Honor una vez finalizada la partida. |
| RF17 | 5 | El sistema debe permitir cambiar el tiro de cámara de la partida así como desplazar o modificar el zoom. |
| RF18 | 1 | El sistema debe mostrar el estado de la partida a sus jugadores y espectadores. |
| RF19 | 1 | El sistema debe llevar un control de las fases de la partida así como del jugador activo en cada momento. |
| RF20 | 5 | Al finalizar una partida, el sistema debe notificar los resultados obtenidos de la partida al servicio de Honor de IHEGames |
| RF21 | 5 | Al finalizar una partida, el sistema debe notificar los resultados obtenidos de la partida al servicio de Elo de IHEGames |

Tabla 4: Requisitos funcionales base de los juegos de IHEGames

En esta segunda tabla se recogen los requisitos funcionales específicos de K-Tan.

| NÚM. | PRIORIDAD | DESCRIPCIÓN |
|---|---|---|
| RF22 | 4 | El sistema debe permitir proponer una oferta de intercambio. |
| RF23 | 4 | El sistema debe permitir cancelar una oferta de intercambio propuesta. |
| RF24 | 2 | El sistema debe permitir lanzar los dados y obtener un resultado aleatorio. |
| RF25 | 5 | Si el jugador no lanza los dados en el límite de tiempo establecido en la configuración de la partida, el sistema debe ser capaz de lanzar los dados automáticamente y registrar el retraso para notificárselo al servicio de Honor una vez finalizada la partida. |
| RF26 | 3 | El sistema debe obligar a descartar la mitad de los recursos a todos los jugadores con más de 7 recursos en mano si el resultado en los dados es de 7. |
| RF27 | 3 | El sistema debe permitir descartar la mitad de los recursos de la mano. |
| RF28 | 3 | El sistema debe permitir mover al ladrón. |
| RF29 | 3 | El sistema debe permitir escoger a un jugador, con al menos un poblado o ciudad junto a la casilla de terreno donde se halla el ladrón, para robarle un recurso al azar de la mano. |
| RF30 | 2 | El sistema debe permitir colocar un poblado en una intersección vacía sin ningún poblado o ciudad en sus intersecciones adyacentes. |
| RF31 | 5 | Si el jugador no coloca ningún poblado en el límite de tiempo establecido en la configuración de la partida, el sistema debe ser capaz de colocar uno aleatoriamente en uno de los posibles lugares y registrar el retraso para notificárselo al servicio de Honor una vez finalizada la partida. |
| RF32 | 3 | El sistema debe proporcionar al jugador un recurso de cada casilla de terreno adyacente a su segundo poblado colocado. |
| RF33 | 3 | El sistema debe aumentar la cuenta de puntos de victoria del jugador en uno al colocar un poblado. |
| RF34 | 2 | El sistema debe permitir colocar una carretera en un camino adyacente al último poblado colocado. |
| RF35 | 5 | Si el jugador no coloca ninguna carretera en el límite de tiempo establecido en la configuración de la partida, el sistema debe ser capaz de colocar una aleatoriamente en uno de los posibles lugares y registrar el retraso para notificárselo al servicio de Honor una vez finalizada la partida. |
| RF36 | 2 | El sistema debe permitir pasar el turno al siguiente jugador. |
| RF37 | 5 | Si tras haber lanzado los dados, el jugador no pasa turno en el límite de tiempo establecido en la configuración de la partida, el sistema debe ser capaz de pasar el turno automáticamente y registrar el retraso para notificárselo al servicio de Honor una vez finalizada la partida. |
| RF38 | 2 | El sistema debe permitir construir un poblado en una intersección vacía conectada a una carretera propia y sin ningún poblado o ciudad en sus intersecciones adyacentes y a cambio de un recurso de madera, uno de arcilla, uno de cereales y uno de lana. |

| | | |
|---|---|---|
| RF39 | 3 | El sistema debe aumentar la cuenta de puntos de victoria del jugador en uno al construir un poblado. |
| RF40 | 2 | El sistema debe permitir construir una ciudad en una intersección con un poblado propio a cambio de tres recursos de piedra y dos de cereales. |
| RF41 | 3 | El sistema debe aumentar la cuenta de puntos de victoria del jugador en uno al construir una ciudad. |
| RF42 | 2 | El sistema debe permitir construir una carretera en un camino vacío conectado a una carretera, poblado o ciudad propia a cambio de un recurso de madera y uno de arcilla. |
| RF43 | 3 | El sistema debe contabilizar las rutas de carreteras consecutivas y otorgar la carta especial de *Gran ruta comercial* al jugador con la ruta más larga y que sea de 5 o más carreteras. |
| RF44 | 3 | El sistema debe aumentar la cuenta de puntos de victoria en dos al recibir la carta especial de *Gran ruta comercial* y disminuir la cuenta al perder la ficha. |
| RF45 | 2 | El sistema debe permitir adquirir una carta de desarrollo al azar del mazo de cartas de desarrollo a cambio de un recurso de piedra, uno de cereales y uno de lana. |
| RF46 | 3 | El sistema debe permitir escoger y jugar una carta de desarrollo de la mano. |
| RF47 | 3 | El sistema debe permitir mover al ladrón al usar una carta de caballero. |
| RF48 | 3 | El sistema debe contabilizar las cartas de caballeros jugados de cada jugador y otorgar la carta especial de *Gran ejército de caballería* al jugador con el mayor número de caballeros jugados siempre que este número sea mayor o igual a 3. |
| RF49 | 3 | El sistema debe aumentar la cuenta de puntos de victoria en dos al recibir la carta especial de *Gran ejército de caballería* y disminuir la cuenta al perder la ficha. |
| RF50 | 3 | Al usar una carta de monopolio, el sistema debe permitir obtener del resto de jugadores todas las cartas de recurso del tipo escogido. |
| RF51 | 3 | Al usar una carta de descubrimiento, el sistema debe permitir escoger y obtener dos recursos. |
| RF52 | 3 | Al usar una carta de construcción de carreteras, el sistema debe permitir construir dos carreteras sin coste alguno de recursos. |
| RF53 | 3 | El sistema debe permitir hacer intercambios marítimos. |
| RF54 | 3 | El sistema debe calcular la cantidad de recursos que un jugador debe entregar en un intercambio marítimo, teniendo en cuenta los puertos adyacentes a sus poblados o ciudades. |
| RF55 | 4 | El sistema debe permitir aceptar una oferta de intercambio y realizar el intercambio de recursos. |

Tabla 5: Requisitos funcionales de K-Tan

## 7.3 Requisitos no funcionales

| NÚM. | PRIORIDAD | REQUISITO | DESCRIPCIÓN |
|---|---|---|---|
| RNF1 | 3 | USABILIDAD | El tiempo de aprendizaje de las reglas de juego para un usuario deberá ser menor a 10 minutos.<br>El tiempo de aprendizaje del uso de la interfaz para un usuario deberá ser menor a 1 minuto.<br><br>La tasa de errores cometidos por el usuario no deberá exceder del 1% de las transacciones totales ejecutadas con el sistema.<br><br>El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.<br><br>La aplicación debe poseer un diseño adaptativo a fin de garantizar la adecuada visualización en múltiples dispositivos. |
| RNF2 | 1 | SEGURIDAD | El ingreso al sistema está restringido a usuarios no identificados correctamente.<br><br>El sistema debe autorizar las acciones realizadas por cada usuario.<br><br>El sistema debe generar trazas de las interacciones de cada usuario con el sistema. |
| RNF3 | 2 | FIABILIDAD | El sistema debe contar con un repertorio de casos de prueba que respalde su fiabilidad y funcionamiento adecuado. |
| RNF4 | 1 | MULTIPLATAFORMA | El sistema deberá funcionar tanto en dispositivos Android como en navegadores web. |

# 8

# Análisis y diseño

A lo largo de este capítulo se describe el análisis y el diseño realizados en el modelo de datos, la lógica de negocios y las interfaces de usuario.

En cada uno de esos tres aspectos, se describen primero las consideraciones más importantes a tener en cuenta relativas al diseño llevado a cabo y que facilitan su compresión. Finalmente se describe el diseño llevado a cabo desglosándolo en dos partes principales; una que recoge la base que será común para cada juego de IHEGames, y otra que es específica de K-Tan.

## 8.1 Modelo de datos

### 8.1.1 Modelo Observer

Uno de los requisitos fundamentales que ha condicionado completamente el diseño del modelo de datos ha sido la necesidad de que los cambios ejecutados por un usuario se sincronicen al momento y que todos los clientes muestren los cambios acontecidos. Por los tanto, para poder mantener la separación entre la lógica de negocio y las interfaces de usuario, se ha recurrido a la estrategia de utilizar el patrón de diseño conocido como *Observer* u Observador.

Este patrón define una dependencia del tipo *uno-a-muchos* entre objetos, de manera que cuando uno de los objetos cambia su estado, notifica este cambio a todos los dependientes. Gracias a este patrón la interfaces de usuario pueden recibir notificaciones por parte del modelo de datos cuando este sea modificado por la lógica de negocio, y actualizarse para mostrar los cambios ocurridos.

Para hacer uso de este patrón, se definen dos clases "observables", `ObservableObject` para objetos simples y `ObservableCollection` para colecciones de objetos. A su vez de definen dos interfaces que implementarán las interfaces de usuario para poder hacer uso de este patrón de diseño, `IObjectObserver` e `ICollectionObserver`.

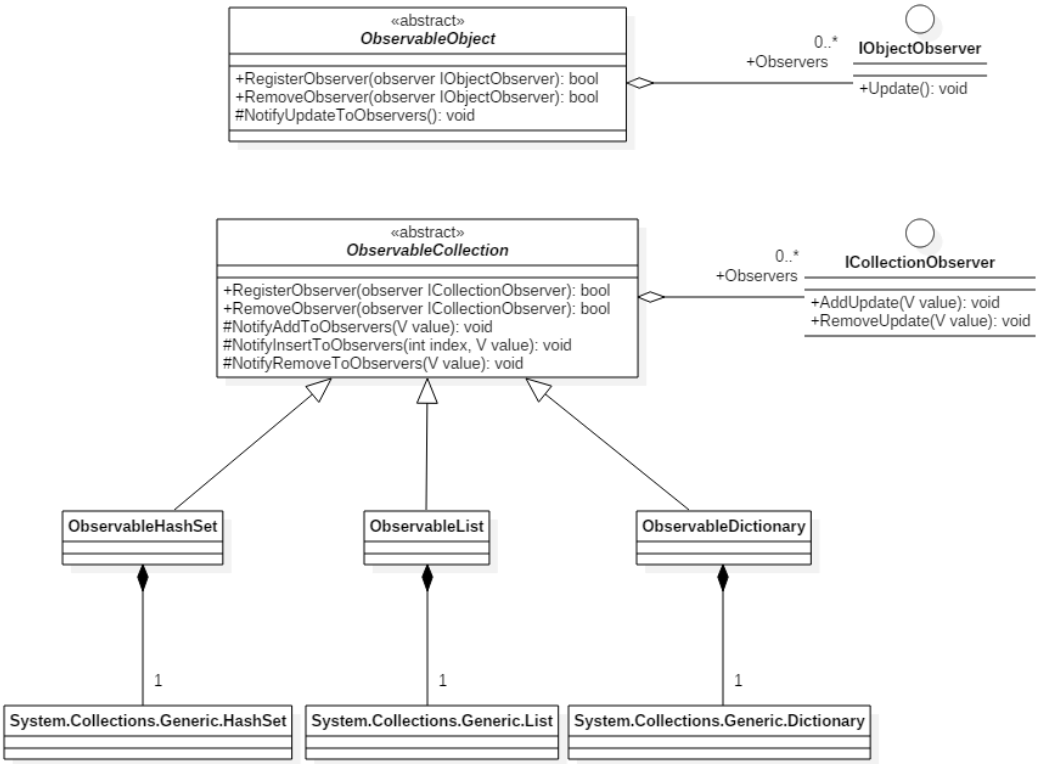


Figura 16: Modelo *Observer*

Como se puede apreciar en la figura, además de las clases e interfaces antes mencionadas, se han definido 3 colecciones concretas a partir de `ObservableCollection`: `ObservableList`, `ObservableHashSet` y `ObservableDictionary`. Cada una de ellas hace uso y mantiene las funcionalidades de las colecciones generales que proporciona C#, `List`, `HashSet` y `Dictionary` respectivamente; y notifican a sus observadores los cambios que ocurren en dichas colecciones.

Todas las clases y colecciones del modelo de datos de K-Tan susceptibles de cambio y que sean relevantes para las interfaces de usuario, extienden de las clases definidas en este modelo.

### 8.1.2 Partes del modelo de datos

Para simplificar la sincronización entre el servidor y los clientes de K-Tan, se ha diseñado un modelo común que tanto el servidor como los clientes utilizarán. Este modelo de datos general es el modelo base diseñado para todos los juegos de IHEGames, incluido K-Tan. El modelo se ha estructurado en diferentes partes. La mayoría de las partes del modelo son comunes tanto en el servidor como en los clientes, sin embargo, algunas partes son exclusivas del servidor o de los clientes.

Por otro lado, aunque no se ha implementado su integración con el servicio de chat de IHEGames, se ha procurado tenerlo en cuenta a lo largo de todo el diseño. En el siguiente diagrama se muestra la estructura del modelo de datos.

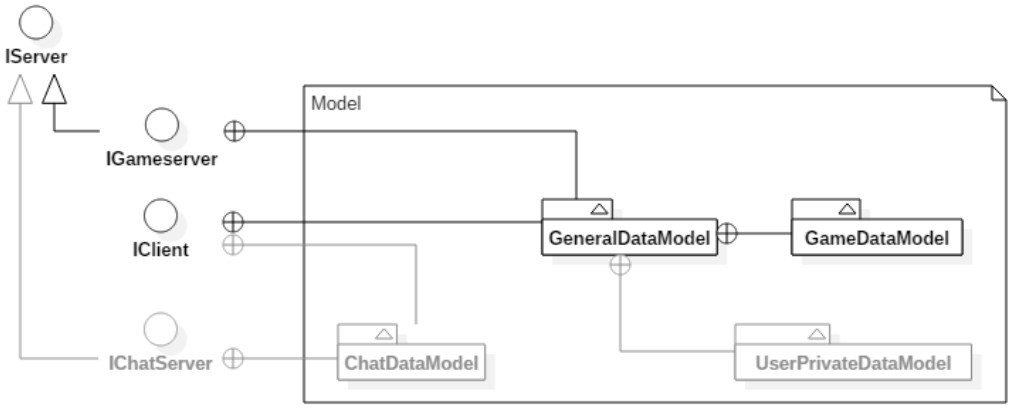


Figura 17: Estructura del modelo de datos de K-Tan

GeneralDataModel: contiene los datos de los usuarios y los datos generales de las partidas existentes. A su vez contiene las partes de `GameDataModel` y `PlayerPrivateDataModel`.

GameDataModel: contiene los datos y detalles concretos de las partidas existentes.

UserPrivateDataModel: contiene los datos privados de los usuarios, aquellos datos irrelevantes para el resto de usuarios (para funcionalidades en próximas versiones).

ChatDataModel: contiene los datos relativos al chat y a los mensajes enviados.

Cabe destacar que tanto la parte de `GameDataModel` como de `PlayerPrivateDataModel` (y también la de `ChatDataModel`), no se comparten completamente entre los clientes y el servidor. Mientras que el servidor contiene los datos completos de esas partes, cada cliente solo contiene los datos privados del propio usuario y los datos concretos de la partida a la que esté unido (así como los datos de chat relativos al usuario).

### 8.1.3 Modificando el modelo de datos

Adicionalmente, dado que este modelo base será reutilizado en los futuros juegos de IHEGames, se ha dedicado una especial atención a la visibilidad de los métodos y propiedades de los diferentes componentes del modelo, en especial a aquellos que modifican el propio modelo.

Mediante el modificador de acceso de C# `protected internal`, se limita la visibilidad de los métodos *set* del modelo de datos para que solo desde una clase del propio Ensamblado o a través de la herencia de clases se pueda acceder a dicho método. Por otro lado, para poder modificar el modelo de datos, se ha definido una jerarquía de acciones que se encuentran en el mismo Ensamblado que los modelos y que serán las responsables de realizar en ellos los cambios concretos. Estas acciones forman el puente entre el modelo de datos y la lógica de negocio.

Las acciones se han estructurado manera similar al modelo de datos, en base al ámbito de actuación de cada acción. A continuación se muestra la jerarquía básica de acciones.

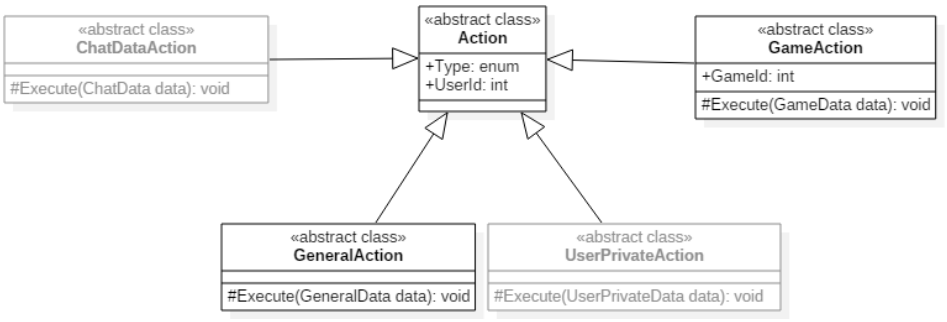


Figura 18: Base de la jerarquía de acciones

De estas acciones básicas se extienden otras acciones concretas que se describen en el próximo apartado *Lógica de negocio*. Cada una de las acciones, contiene el método abstracto `Execute` que se implementa en las acciones concretas. Este método está a su vez restringido por el modificador de acceso `protected internal`. Esto se debe a que, en general, las acciones de un mismo ámbito del modelo se tienen que resolver de manera atómica, puesto que una ejecución simultánea de varias acciones en el mismo ámbito del modelo podría acarrear un funcionamiento no deseado. Para evitar ese tipo de situaciones,

cada parte del modelo es la responsable de ejecutar atómicamente las acciones correspondientes, haciendo uso de los bloqueos pertinentes.

En la siguiente figura se muestra un diagrama de secuencia al ejecutar una acción desde el modelo `GeneralDataModel`:

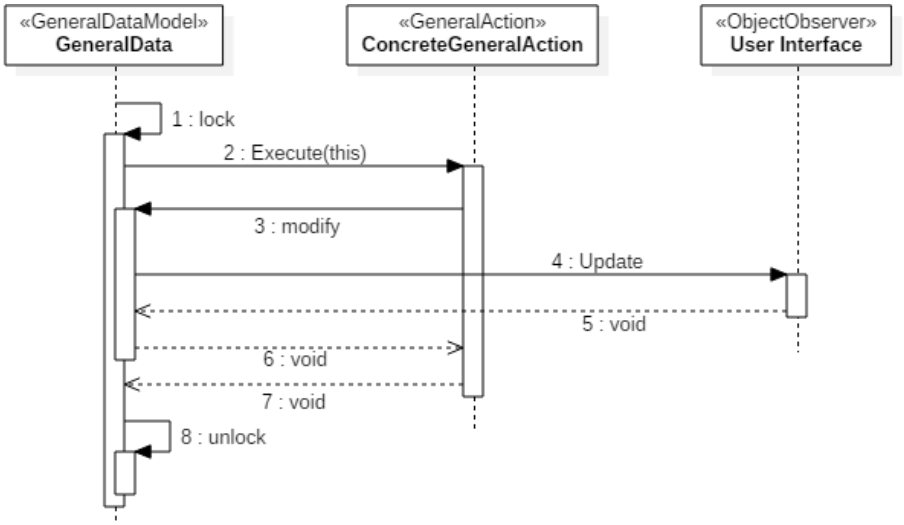


Figura 19: Diagrama de secuencia de modificación de modelo de datos

Las demás partes del modelo de datos ejecutan las acciones de manera idéntica. Sin embargo, cabe destacar que la parte de `GameDataModel`, una vez ha ejecutado satisfactoriamente la acción, la añade a una colección de acciones. De manera que, cuando un cliente solicita al servidor el estado actual de una partida, el cliente puede alcanzar dicho estado de la partida reproduciendo cada una de las acciones de la partida. Esto permite además que, para futuras funcionalidades del juego, se pueda reproducir la partida una vez finalizada, reiniciando la partida y ejecutando dichas funciones de nuevo.

Por otro lado, dado que se estima que en la mayor parte de las veces los usuarios acceden a una partida al inicio de ésta, cuando aún no se ha realizado ninguna acción, se prevé que con esta estrategia el tráfico de datos será mucho más reducido que si se enviase el estado completo de la partida.

### 8.1.4 Modelo de datos base de los juegos de IHEGames

Teniendo en cuenta todos los aspectos que hasta el momento se han descrito en este apartado, se diseña un modelo de datos base para los juegos de IHEGames, entre los que se encuentra K-Tan. En el siguiente diagrama se muestra dicho modelo de datos en el que se han encuadrado las cuatro partes del modelo de datos antes mencionadas y se han relacionado con las acciones y el patrón *Observer*:

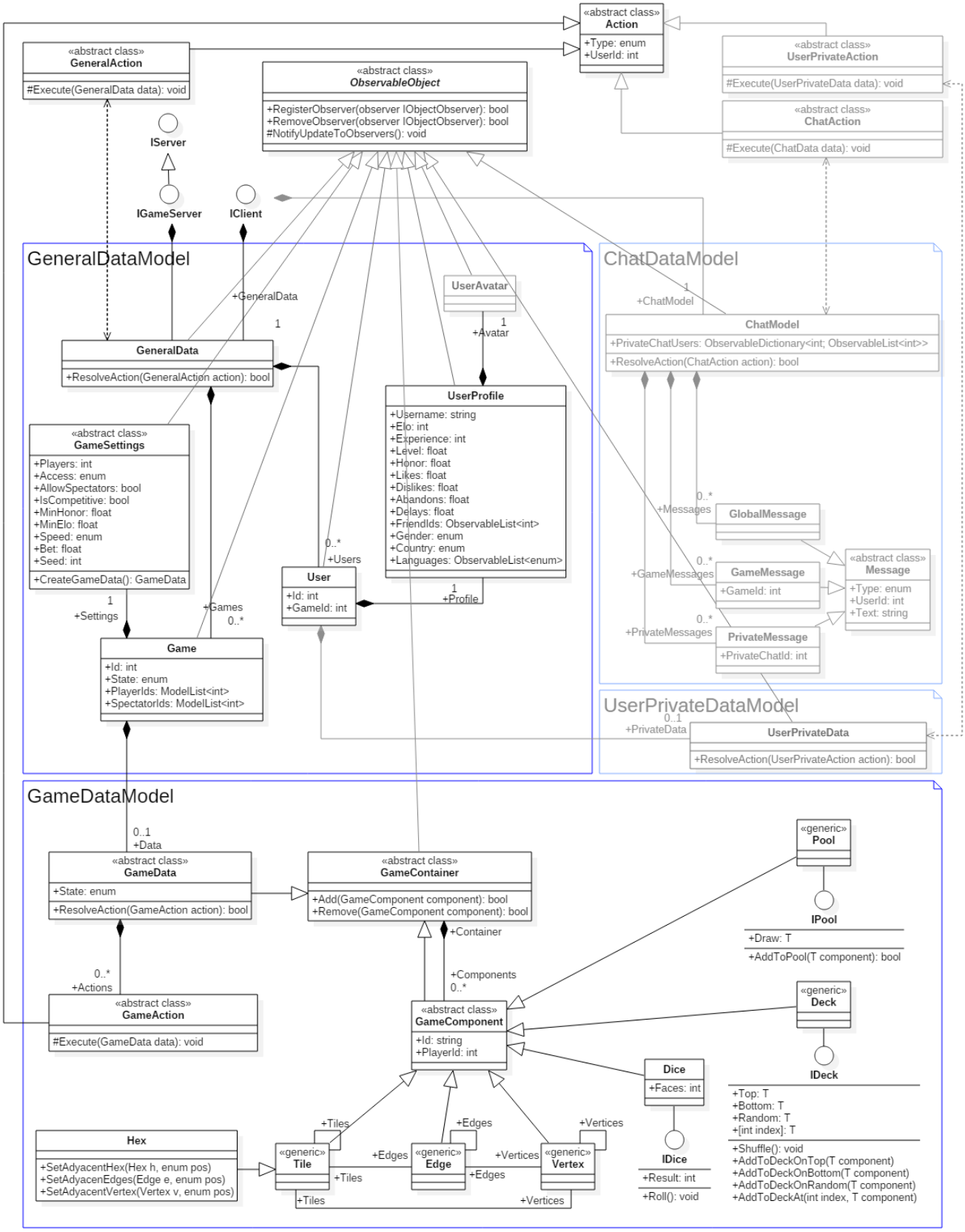


Figura 20: Modelo de datos base de los juegos de IHEGames

Este modelo de datos base define la base de los modelos de todos los juegos de IHEGames. En la parte de `GameDataModel`, se ha diseñado una estructura básica de componentes contenedores de otros componentes. Esto permite que cada juego diseñe e implemente sus propios componentes de juego. Aún así, en el análisis de los componentes de juego de K-Tan se han extraído y definido algunos componentes más concretos que podrían ser comunes en otros juegos, como es el caso de las clases `Pool`, `Deck`, `Dice`, `Tile`, `Edge`, `Vertex` y `Hex`. Cabe destacar que, este modelo se irá ampliando con cada desarrollo de juegos, puesto que, se espera que, en los análisis y diseños de los futuros juegos, se vayan incorporando nuevos componentes de juego genéricos.

Por otro lado, aunque no se puede apreciar en este modelo, todas las relaciones de composición y agregación entre las clases de este modelo, se implementan mediante el uso de las colecciones "observables" del modelo del patrón *Observer* antes descrito, tales como `ObservableList`, `ObservableHashSet` y `ObservableDictionary`.

Otro factor de suma importancia en este modelo es la semilla o `Seed` de la clase `GameSettings`. Tal y como se ha comentado, se quiere mantener una sincronización entre el modelo de datos del servidor y el de los clientes. Sin embargo, los juegos, suelen contener elementos de azar en sus partidas. Dado que la clase `GameSettings` es la responsable de crear las partidas a través de su método `CreateGameData()`, su semilla permite que dos instancias de `GameSettings` con la misma semilla creen una misma partida y que ante una secuencia idéntica de acciones la partida se desarrolle de la misma manera. Esta funcionalidad es la que principalmente permite, no solo mantener la sincronización de la partida entre cliente y servidor, sino recuperar el estado actual de la partida iniciando o creando de nuevo la partida a partir de la misma instancia de `GameSettings` y ejecutando de nuevo la misma secuencia de acciones.

### 8.1.5 Modelo de datos específico de catan

A partir del modelo base, cada juego puede extender la parte de `GameDataModel`, así como la clase `GameSettings` de la parte `GeneralDataModel`, formando un modelo de datos específico para cada juego.

Tras el análisis de K-Tan se definen unos elementos específicos de este juego. Estos elementos específicos, junto al modelo de datos base antes mostrado, forman el modelo de datos específico de K-Tan.

En la siguiente figura se muestra dicho modelo, en la cual se han destacado los elementos específicos de K-Tan:

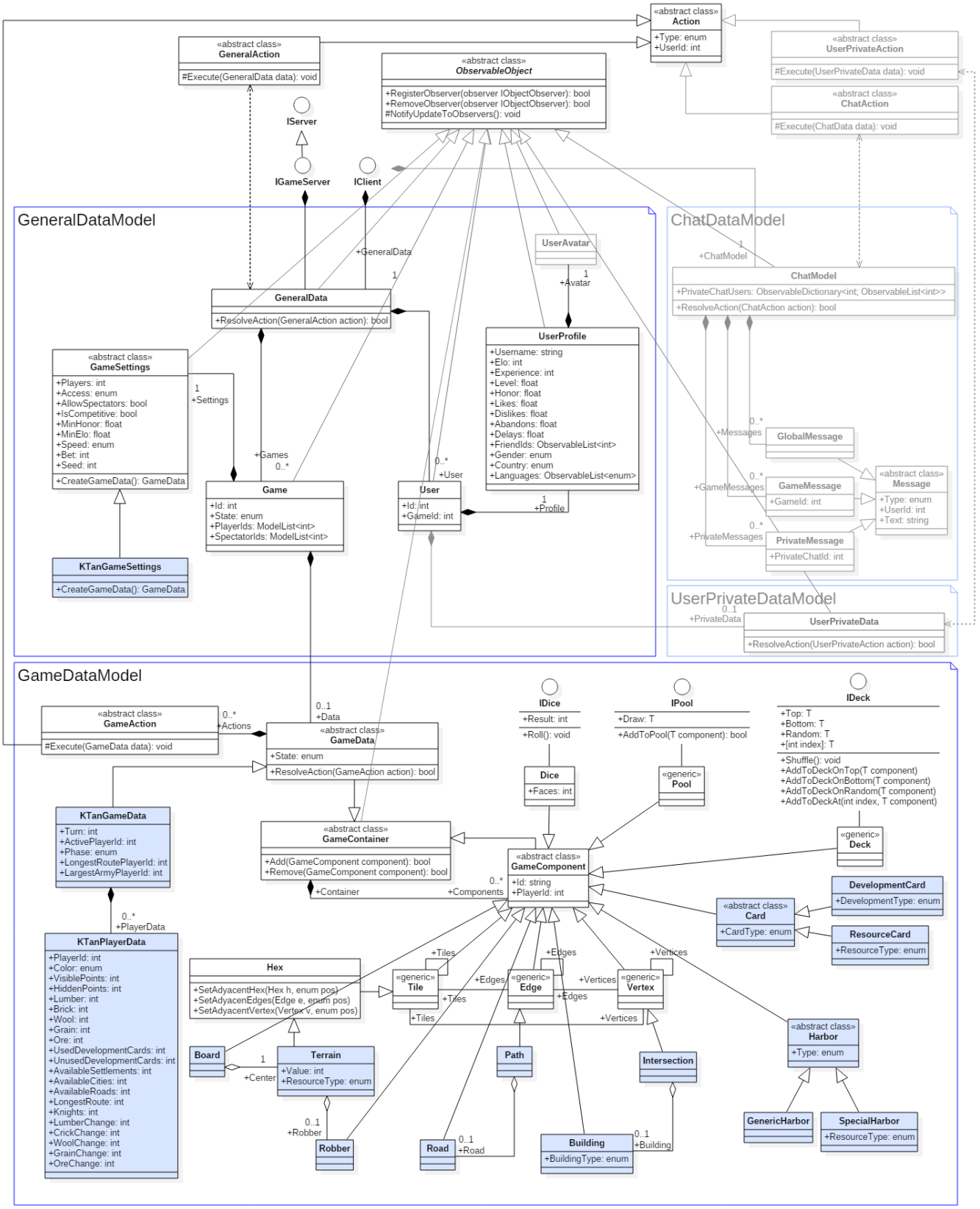


Figura 21: Modelo de datos específico de K-Tan

### 8.1.6 Esquema de la base de datos

Respecto al almacenamiento de datos, dado que los diferentes servicios de IHEGames se encargan de almacenar la información general de los usuarios, así como la relacionada con su Elo y Honor, se define un sencillo modelo relacional para el almacenamiento de las partidas. Mediante este modelo se pueden almacenar las partidas, sus jugadores y las acciones realizadas a lo largo de cada partida.

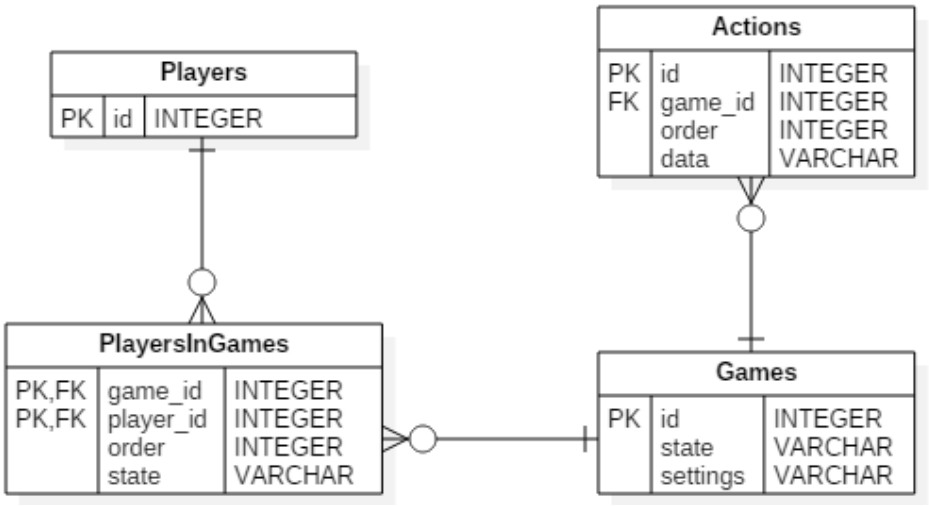


Figura 22: Esquema de la base de datos de K-Tan y los juegos de IHEGames

Cabe destacar que tanto `settings` en `Games`, como `data` en `Actions`, almacenan un objeto de configuración de partida (`GameSettings`) y un objeto de acción (`Action`) respectivamente, ambos en formato JSON. Esto permite que todos los juegos utilicen este mismo modelo para almacenar sus partidas, ya que a partir de la configuración de partida, sus jugadores y las acciones ejecutadas, el sistema es capaz de recuperar el estado de la partida.

Los campos `order` de `PlayersInGames` y `Actions`, almacenan el orden de entrada de los jugadores y el orden de la ejecución de las acciones respectivamente.

Los campos `state` de `PlayersInGames` y `Games`, almacenan el estado del jugador (`STARTED/FINISHED/ABANDONED`) y el estado del juego respectivamente (`NOT_STARTED /STARTING/STARTED/FINISHED/CANCELLED`).

## 8.2 Lógica de negocio

Tal y como se ha descrito en el apartado anterior *Modelo de datos*, se han definido una serie de acciones que permites modificar el modelo de datos de manera controlada. Por otro lado, se ha explicado en el capítulo de *Arquitectura*, que estas acciones constituyen también la base de las comunicaciones entre los clientes y el servidor de juego. Es por ello que la lógica de negocio consiste principalmente en el envío y tratamiento de dichas acciones así como en las propias acciones.

### 8.2.1 Envío de acciones

Tal y como se ha mostrado en el capítulo de *Arquitectura*, K-Tan y el resto de juegos de IHEGames, se separan en múltiples clientes y un servidor de juego. El método de comunicación entre estas partes consiste en el envío de acciones en formato JSON a través de WSS. Por ello, cada parte es capaz de serializar y deserializar en formato JSON cualquier acción.

Este envío de acciones, conlleva una gestión de los diferentes sockets. Los clientes solo necesitan gestionar un socket a través del cual se comunican con el servidor de juego (y otro para el servicio de chat). Sin embargo, el servidor requiere una gestión un poco más compleja ya que la comunicación con un cliente puede necesitar extenderse a otros clientes concretos. Por ello, en el servidor, cada socket se relación con un usuario cuando éste se identifica. De esta manera, además de realizar envíos a un socket concreto, el servidor puede trasmitir una acción a todos los sockets abiertos, a todos los sockets de los usuarios identificados, así como a todos los sockets de un usuario concreto

### 8.2.2 Tratamiento de las acciones

El tratamiento de acciones en los clientes de juego es sencillo, ya que se considera que los datos provenientes del servidor son de confianza. Por ello, la lógica de los clientes, se limita a recibir las acciones del servidor, y según el tipo base de acción pasársela a la parte del modelo de datos correspondiente para que éste la ejecute tal y como se explica en el apartado anterior *Modelo de datos*.

Por otro lado, debido a la interacción con el usuario, una interfaz de usuario es capaz de solicitar a la lógica de negocio del lado cliente la ejecución de una acción. Sin embargo, la arquitectura planteada para K-Tan y los juegos de IHEGames una arquitectura cliente-servidor, siendo el servidor un servidor autoritativo. Es decir, toda acción del usuario que se extienda más allá del ámbito de la parte cliente, debe ser antes autorizada por el servidor. Por ello, la lógica del lado cliente, al recibir una acción por parte de la interfaz de usuario, la remite directamente al servidor.

Al contrario que en el dado del cliente, en el lado servidor las acciones recibidas se validan antes de ejecutarlas. Dado que el proceso de ejecución de cada acción está ligado al tipo concreto de cada acción, el proceso de ejecución de una acción no se puede alterar y cualquier otro tipo de acción resultaría desconocida para el servidor y por lo tanto se detectaría en el proceso de deserialización del JSON recibido. Por ello, la lógica de negocio del servidor se limita a validar los datos generales de las acciones, como contrastar el usuario que realiza la acción con el socket a través del que se recibe la acción, y realiza una serie de comprobaciones básicas para comprobar que esa acción está autorizada a ejecutarse. El resto de comprobaciones se delegan a la ejecución de las acciones concretas. Al comienzo de su ejecución se validan los datos concretos de la propia acción y se realizan las comprobaciones de autorización oportunas.

Una vez ejecutada la acción el servidor remite la misma acción a los clientes oportunos, y finalmente almacena la acción en la base de datos. Por ejemplo, las acciones de tipo `GeneralAction` se remiten a todos los clientes asociados a los usuario identificados, mientras que las de tipo `GameAction` se remiten a los clientes de los usuarios unidos a la partida concreta en la que se ejecuta la acción.

A continuación de muestra una secuencia de los pasos que ocurren a partir de la interacción de un usuario con la interfaz.

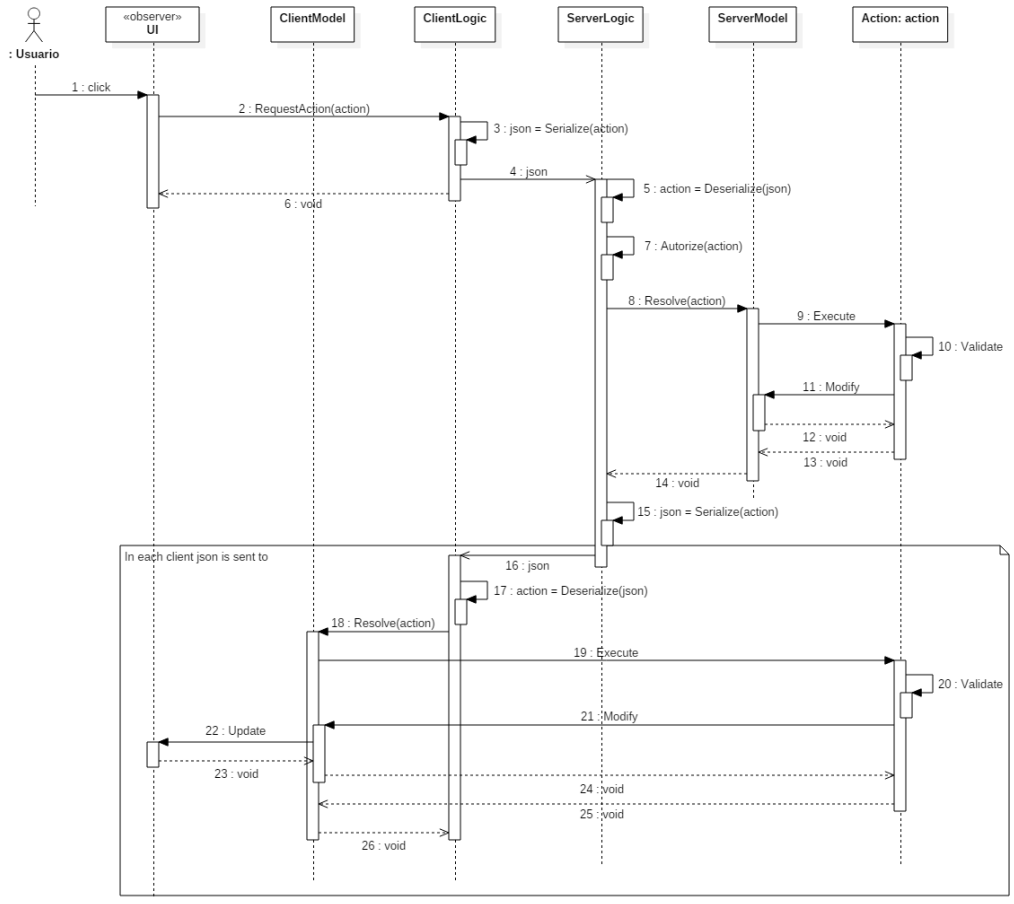


Figura 23: Diagrama de secuencia del envío y tratamiento de acciones

### 8.2.3 Las acciones

En el apartado de *Modelo de datos* se han descrito las diferentes acciones generales relacionadas con el modelo de datos. Sin embargo, se definen unas clases de acciones adicionales que se ejecutan directamente en la lógica de negocio correspondiente. Estas acciones se caracterizan porque además de modificar el modelo utilizan funcionalidades específicas de la lógica de negocio.

En la siguiente figura se muestran la jerarquía básica de acciones extendida, en la que se destacan estas acciones adicionales.

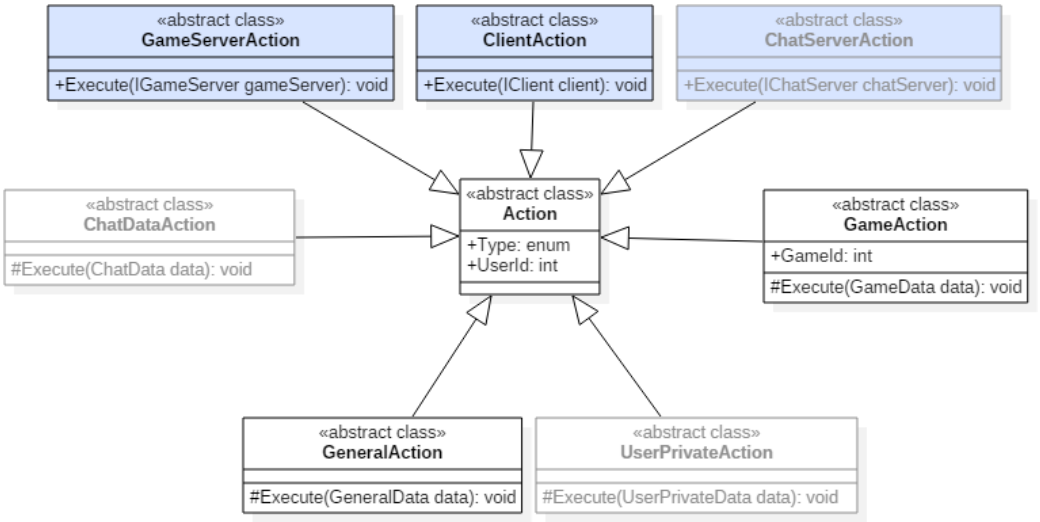


Figura 24: Base extendida de la jerarquía de acciones

A partir de estas acciones base, se definen acciones concretas que serán las encargadas de ejecutar la mayoría de las funcionalidades del juego. Cada una de las acciones tiene un método `Execute` mediante el cual se ejecuta la acción. En la ejecución de este método, antes de aplicar la acción o grupo de acciones correspondientes, y tal y como se ha especificado en el sub-apartado anterior, la acción debe validar los datos concretos que contenga y verificar el estado del modelo de datos. Si todo es correcto, la ejecución termina por aplicar las acciones establecidas. En caso contrario la ejecución de la acción se detiene y se finaliza notificando el error, que se extiende hasta la lógica de negocio donde se registra.

A continuación se muestra la jerarquía completa de las acciones concretas que se han definido y que son comunes para K-Tan y el resto de juegos de IHEGames.

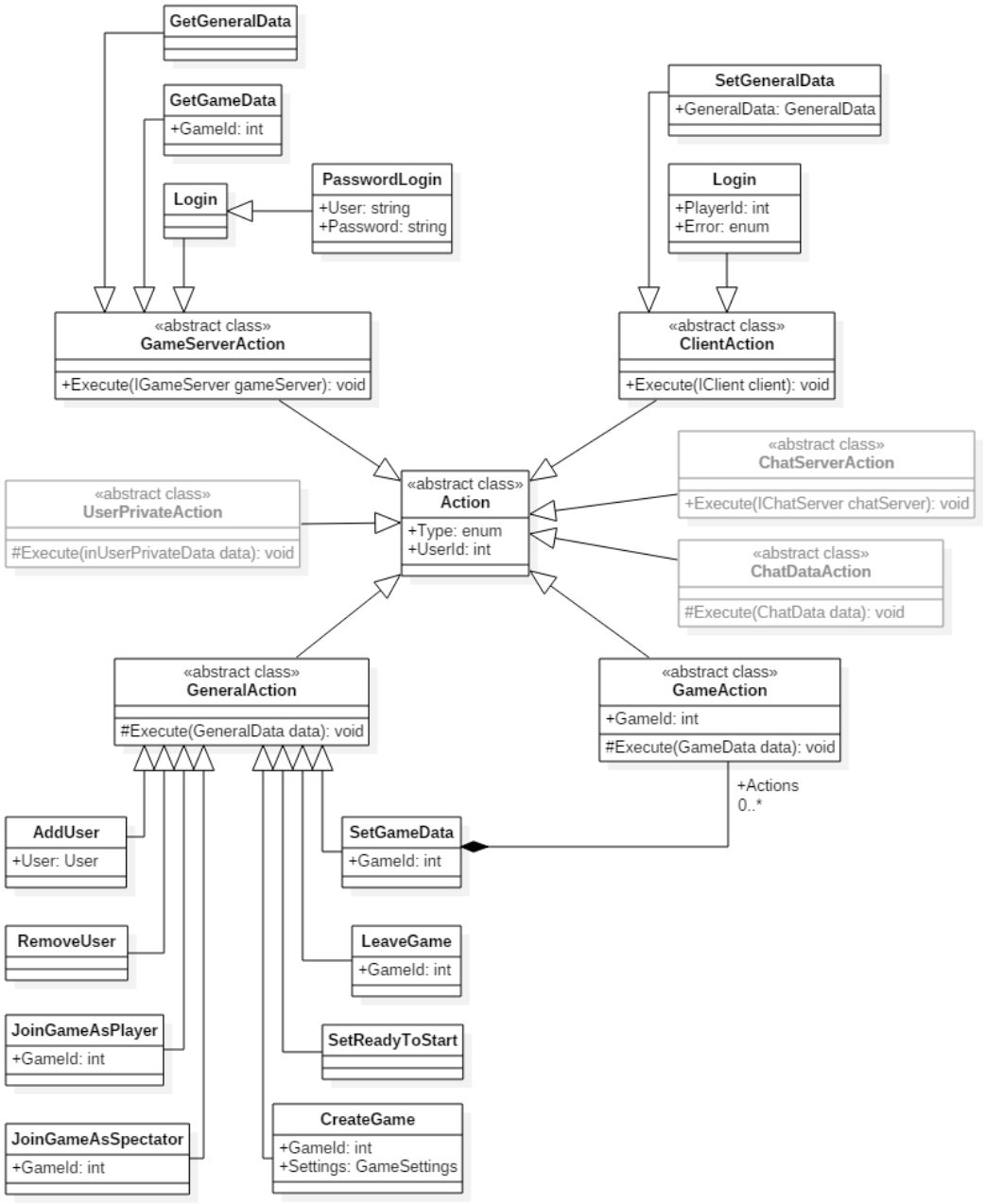


Figura 25: Jerarquía de las acciones base de los juegos de IHEGames

Como se puede apreciar, en esta jerarquía no se define ninguna acción concreta de tipo `GameAction` puesto que, en principio, estas acciones consisten en las acciones específicas de cada juego.

Cabe destacar el hecho de que al serializar acciones como `SetGeneralData` que contienen una parte del modelo de datos que a su vez contiene más partes, solo se serializa la estructura de la parte principal, sin incluir el resto de las partes. De esta manera se mantiene

la diferenciación entre las distintas partes y no se comparten con los clientes los datos que sean irrelevantes para dichos clientes.

En la siguiente figura, se muestra la jerarquía de acciones específica de K-Tan, la cual se forma a partir de la anterior jerarquía base de acciones junto con las acciones específicas de K-Tan, las cuales se han resaltado en el diagrama.

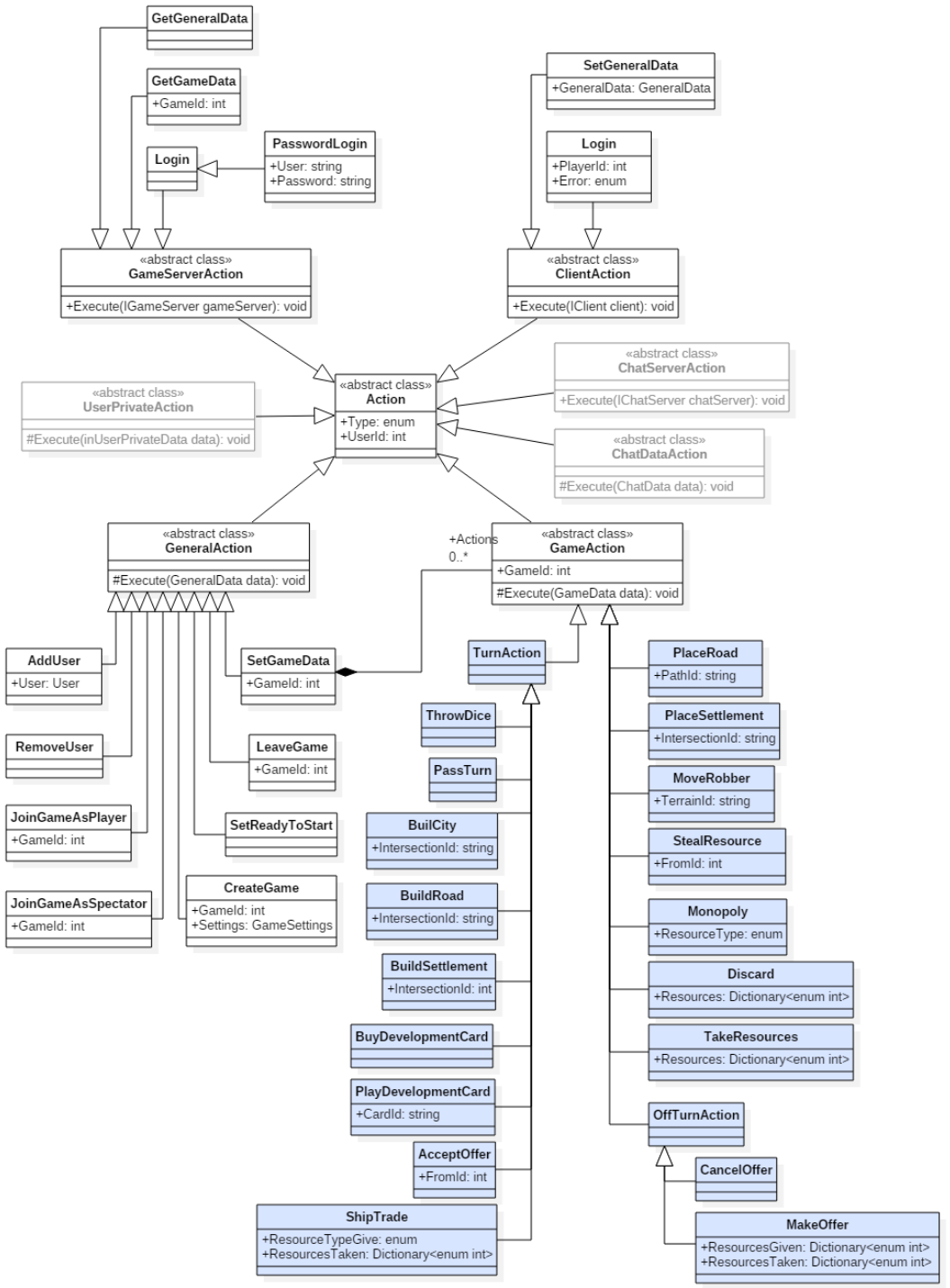


Figura 26: Jerarquía de acciones específica de K-Tan

### 8.2.4 Procesos paralelos

Junto con las características anteriormente descritas, la lógica del servidor puede iniciar procesos paralelos en respuesta a la ejecución exitosa de determinadas acciones. Gracias al patrón *Observer* sobre el cual se construye el modelo de datos, esos procesos son capaces de reaccionar ante determinadas situaciones y cambios en el modelo, así como ante el paso del tiempo. Junto con la capacidad que tienen estos procesos de solicitar la ejecución de acciones a la lógica del servidor, son la base para que el sistema pueda realizar determinadas acciones automáticamente y desarrollar una IA que pueda interactuar en el juego.

Mediante estos procesos, se han desarrollado las funcionalidades que consisten en acciones automáticas del sistema que se realizan tras un periodo de tiempo determinado:

Cuando se crea una partida, la lógica del servidor, además de insertar la partida en la base de datos, lanza un proceso paralelo que monitoriza esta partida. Este proceso, se encarga de actualizar el estado de los jugadores y de la partida en la base de datos, así como de realizar las acciones automáticas cuando un jugador consume el tiempo máximo establecido para realizar dichas acciones. Cuando la partida finaliza, este proceso es también el encargado de notificar su resultado a los servicios de Honor y Elo.

## 8.3 Interfaces de usuario

En el diseño de las interfaces de usuario se definen 3 interfaces principales: una interfaz inicial para la identificación del usuario, una interfaz general donde se muestran las partidas existentes y una interfaz de la partida.

En un principio, se realizan unos diseños de las distribuciones básicas de los elementos de dichas interfaces y de los distintos controles de interacción. Estos diseños se realizan teniendo en consideración la adaptabilidad de las interfaces a las diferentes proporciones de los dispositivos desde los que se podrán visualizar. Para ello se consideran las proporciones de 16:9, 4:3 y 3:2.

A continuación se muestran algunos de los diseños realizados:

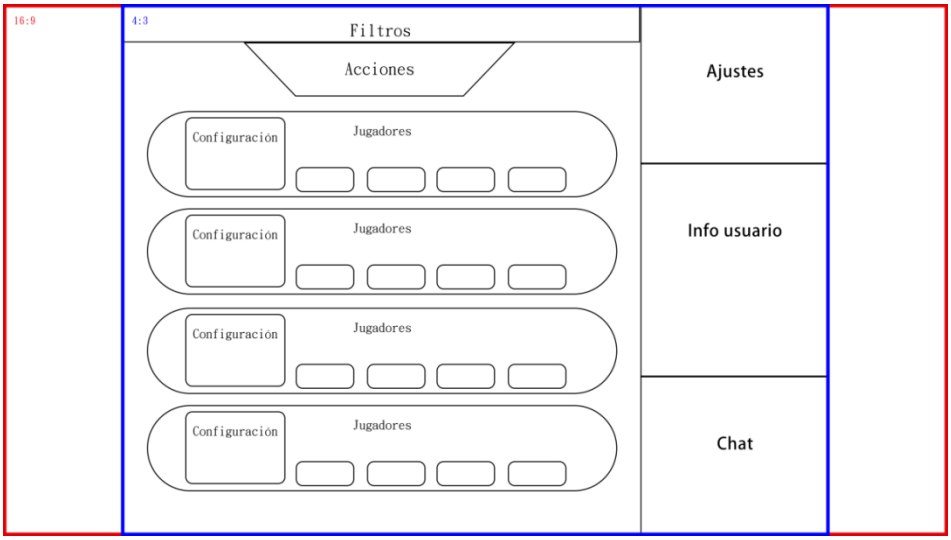


Figura 27: Diseño de la distribución de la interfaz general a 4:3

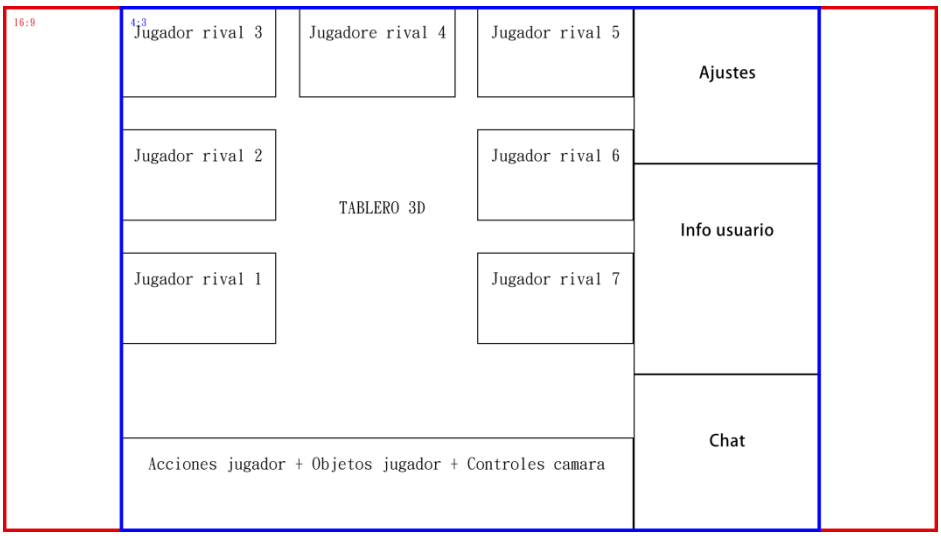


Figura 28: Diseño de la distribución de la interfaz de partida a 4

En el diseño de estas interfaces, se ha procurado mantener una escala adecuada de los diferentes componentes para lograr una adecuada usabilidad en dispositivos móviles y tabletas con pantallas más reducidas. A partir de estos diseños, se han implementado las interfaces definitivas del juego.

A continuación se muestran dichas interfaces en pantallas de diferentes proporciones:


Figura 29: Interfaz de usuario: Identificación de usuario


Figura 30: Interfaz de usuario: Partidas existentes


Figura 31: Interfaz de usuario: Partida de K-Tan

Cabe destacar, que las dos primeras interfaces, serían comunes o similares para todos los juegos de IHEGames, siendo la última la específica de cada juego.

A continuación se muestran algunas secuencias de interacciones con la interfaz.

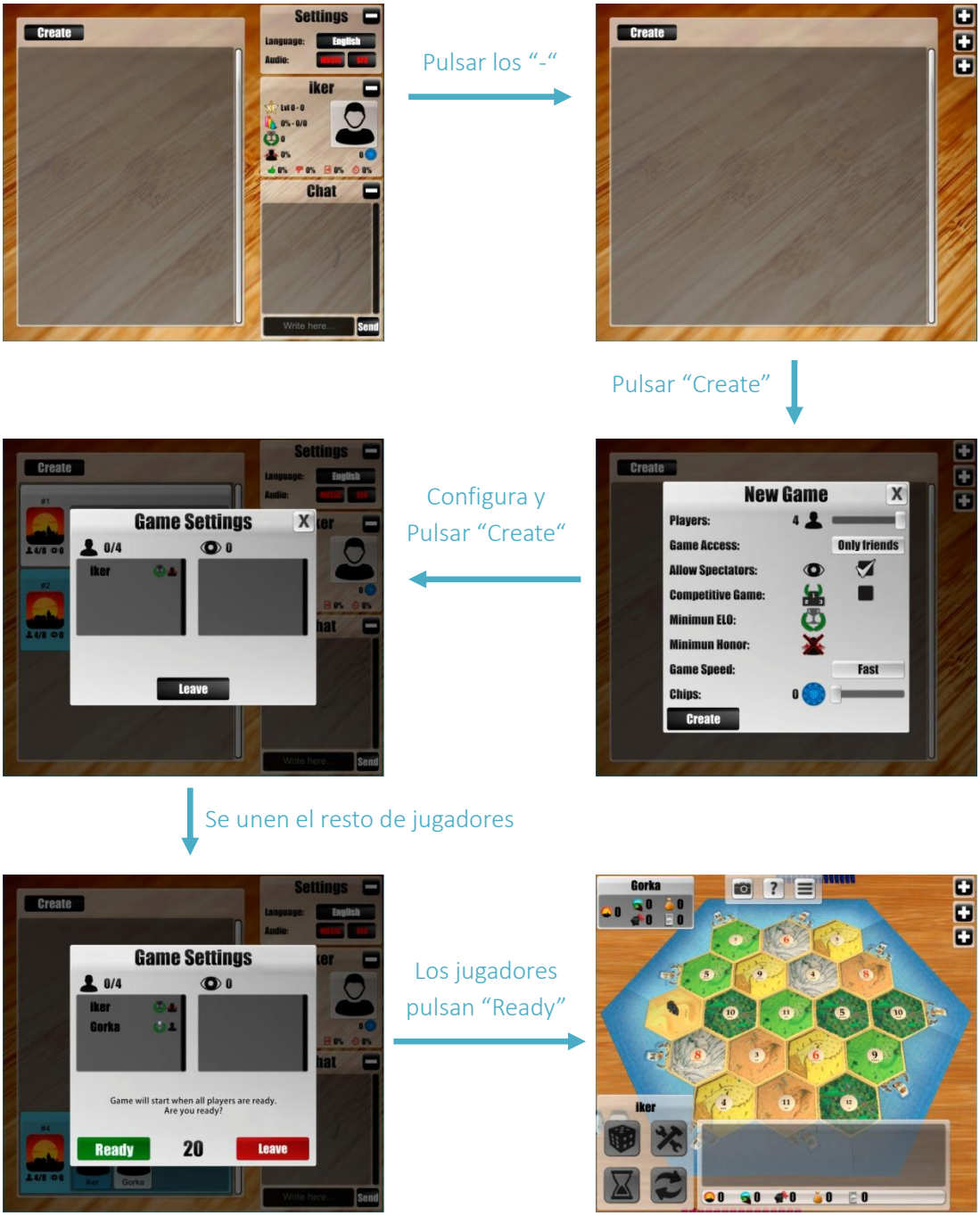


Figura 32: Secuencia de ejemplo para crear e iniciar partidas

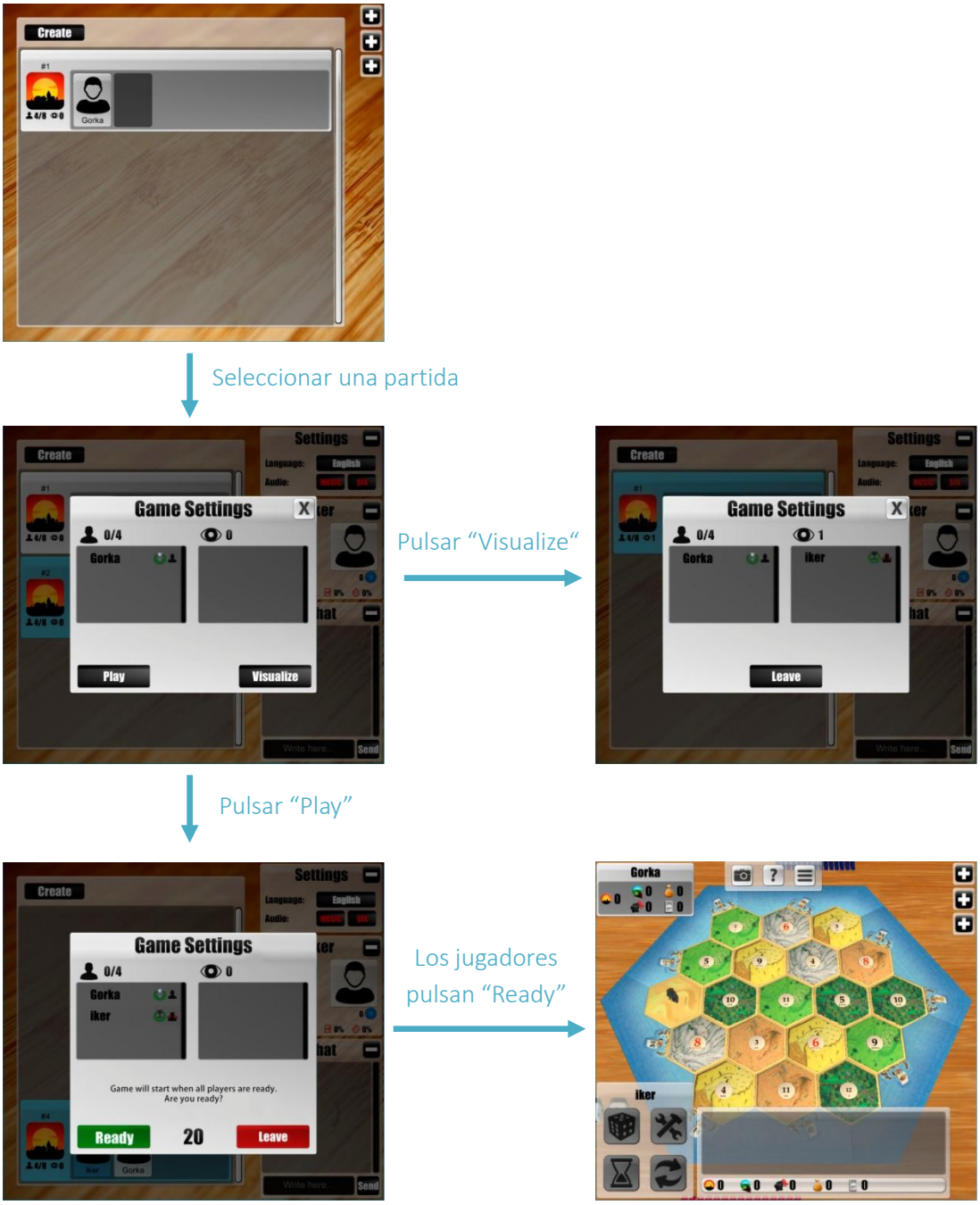


Figura 33: Secuencia para unirse a partidas

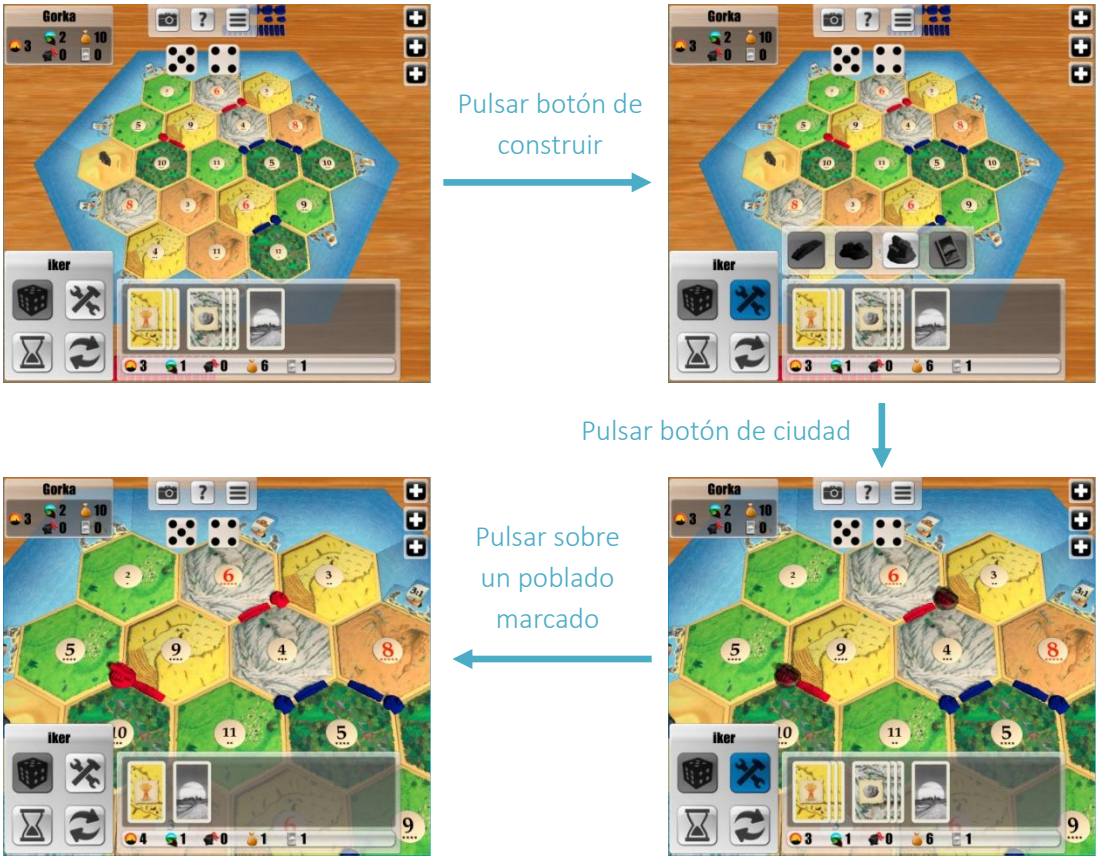


Figura 34: Secuencia para construir una ciudad


Figura 35: Secuencia para lanzar un dado

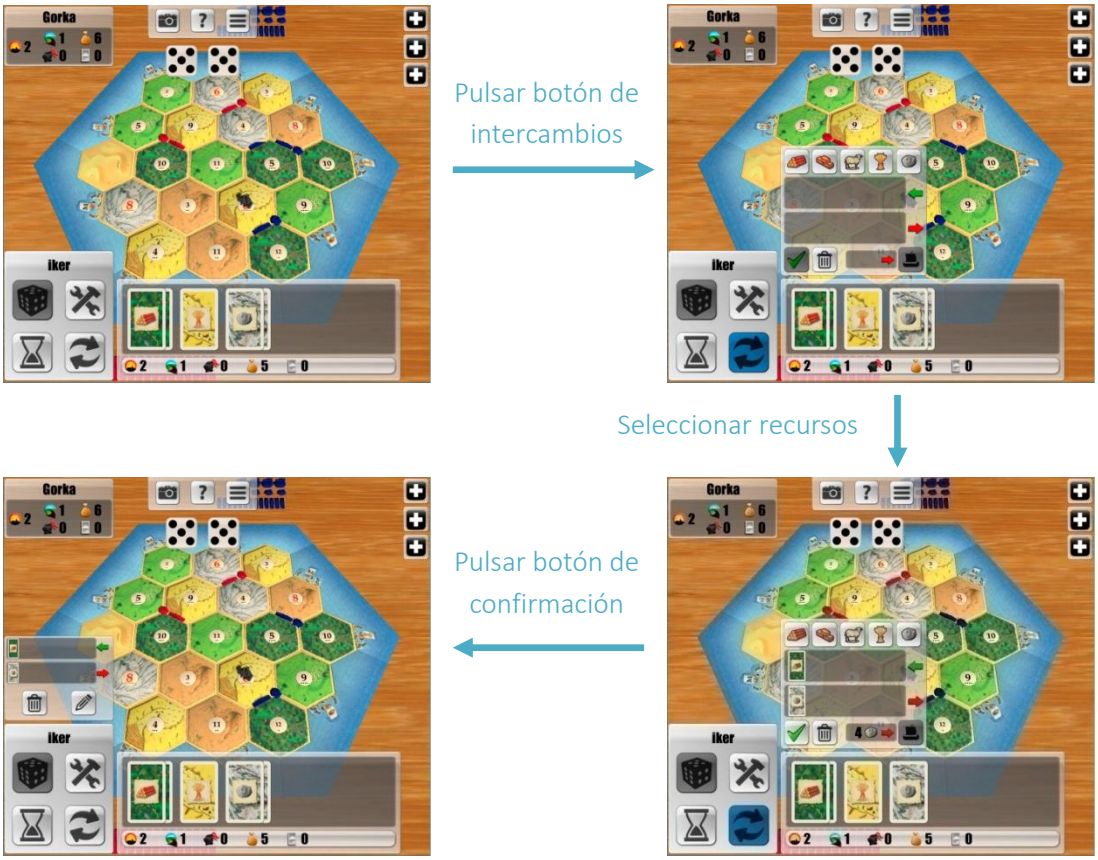


Figura 36: Secuencia para ofrecer un intercambio a otros jugadores.


Figura 37: Secuencia para visualizar las cartas de desarrollo

# 9

# Implementación

En este capítulo se describen la estructura que se ha llevado a cabo a la hora de implementar este proyecto, y se muestran algunas métricas que reflejan la magnitud del trabajo realizado. Finalmente se detallan los Asset y las herramientas que se han utilizado para llevar a cabo la implementación de este proyecto.

## 9.1 Estructuración de la implementación

La implementación del servidor de K-Tan se ha realizado mediante la herramienta de desarrollo Visual Studio 2015. Mientras que los clientes de K-Tan se han implementado en Unity.

Puesto que tanto la parte cliente como la parte de servidor comparten el modelo de datos y algunas bases de la lógica de negocio, dichas implementaciones se han realizado en Visual Studio y se han reutilizado en Unity para la parte cliente. Sin embargo, dado que el scripting en Unity se basa en Mono, la implementación de código abierto de .NET Framework, ha sido necesario limitar las versiones .NET de las partes comunes desarrolladas en Visual Studio. Limitando la versión de los ensamblados que se han utilizado en Unity a la versión.NET 3.5 es suficiente para que éstos sean compatibles en Unity. Todo esto ha supuesto una estructuración de ensamblados para organizar las partes que son comunes entre cliente y servidor, así como las que son comunes para todos los juegos de IHEGames y lo que es específico de K-Tan.

En la siguiente figura se muestra la relación entre los propios ensamblados creados en Visual Studio y la implementación del lado cliente en Unity:

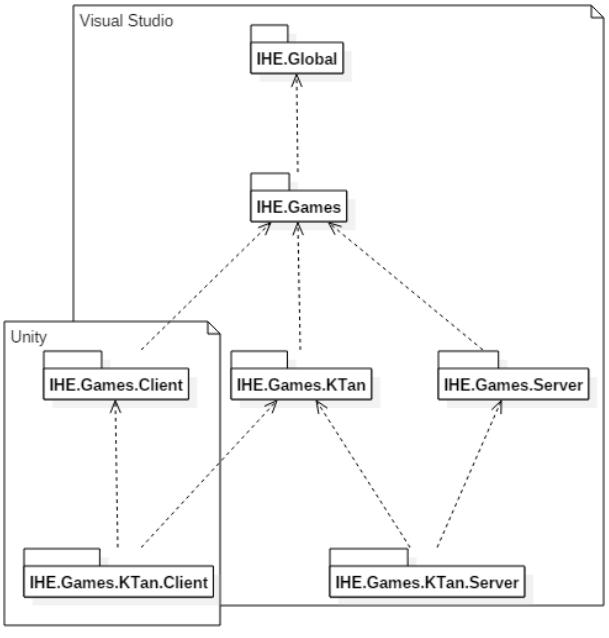


Figura 38: Dependencias de los ensamblados desarrollados

IHE.Global: librería de clases generales para todos los sistemas de IHEGames (Versión .NET 3.5).

**IHE.Games**: contiene el modelo de datos base y las acciones base de los juegos de IHEGames (Versión.NET 3.5).

**IHE.Games.KTan**: contiene el modelo de datos y las acciones específicas de KTan (Versión.NET 3.5).

**IHE.Games.Client**: contiene la lógica de negocio base del lado cliente de los juegos de IHEGames (Versión.NET Mono).

**IHE.Games.Server**: contiene la lógica de negocio base del lado servidor de los juegos de IHEGames (Versión.NET 4.5.2).

**IHE.Games.KTan.Server**: implementa el lado servidor específico de K-Tan (Versión.NET 4.5.2).

**IHE.Games.KTan.Client**: implementa el lado cliente específico de K-Tan (Versión.NET Mono).

A continuación se muestra, a modo de ejemplo, las estructuras de algunos de los ensamblados antes mencionados:

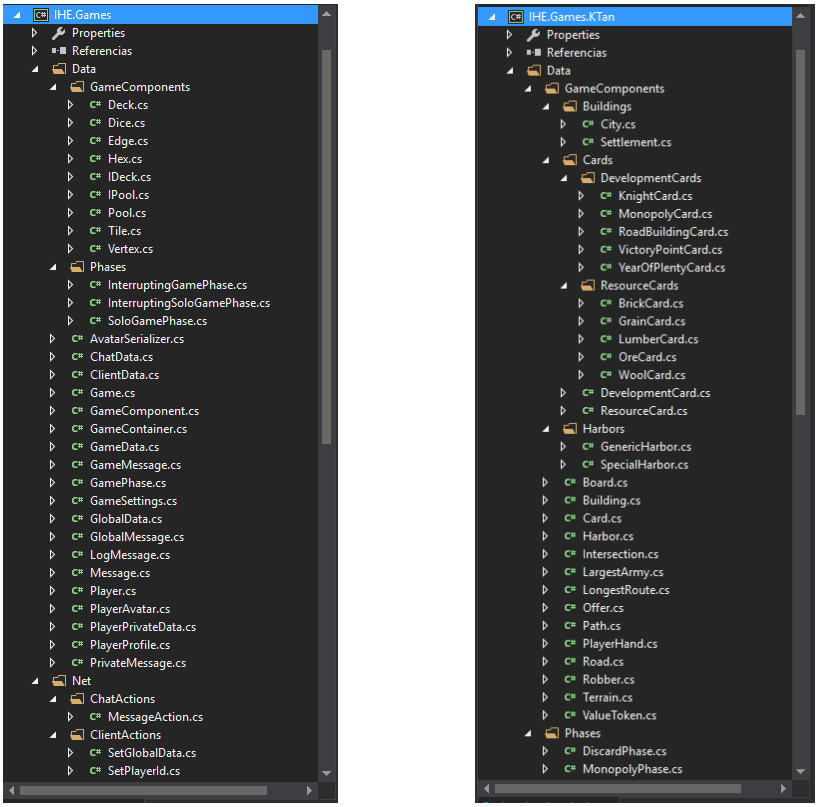


Figura 39: Estructura de IHE.Games e IHE.Games.Ktan

Por otro lado, la implementación de las interfaces de usuario en Unity se ha distribuido en 6 partes o, como se denominan en Unity, escenas. Las escenas siguen un flujo concreto en cada una se desarrollan unas funcionalidades concretas.


Figura 40: Secuencia de las escenas

Intro: Se muestra una pantalla de inicio. En esta escena se establece la conexión con el servidor de K-Tan. Una vez establecida, se avanza a la siguiente escena.

Login: Se muestra la interfaz de identificación de usuarios. Si el usuario se identifica correctamente se avanza a la siguiente escena.

LoadGameData: Se muestra una pantalla de carga en la que se obtienen los datos generales del juego (los usuarios conectados y las partidas existentes). Una vez obtenidos los datos se avanza a la siguiente escena.

Games: Se muestra la interfaz de partidas existentes. Si el usuario está unido o se une a una partida iniciada, o la partida a la que se ha unido inicia, se avanza a la siguiente escena.

LoadGameData: Se muestra una pantalla de carga en la que se obtienen los datos de la partida (la configuración de la partida y las acciones realizadas en ella). Una vez obtenidos los datos se avanza a la siguiente escena.

Game: Se muestra la interfaz de partida. Si la partida termina o el usuario abandona la partida se vuelve a la escena *Games.*

En cada escena se establece una jerarquía de objetos denominados `GameObject`,a los cuales se les asocia scripts denominados `Component.` Estos scripts son clases que extienden de una clase `MonoBehaviour` proporcionada por Unity y que determinan las funcionalidades de cada `GameObject`.

A continuación se muestran, a modo de ejemplo, las jerarquías de objetos de algunas de las escenas implementadas en Unity.


Figura 41: Jerarquía de objetos de las escenas Games y Game

## 9.2 Métricas de la implementación realizada

A lo largo del desarrollo del proyecto se han implementado cerca de 4000 líneas de código en el lado servidor y 3000 líneas de código en el lado cliente, distribuidas respectivamente en 163 y 73 clases e interfaces.

A continuación se muestra una tabla con las métricas obtenidas para cada uno de los ensamblados desarrollados:

| Ensamblado | Líneas de código | Clases e interfaces |
|---|---|---|
| IHE.Global | 283 | 6 |
| IHE.Games | 1771 | 83 |
| IHE.Games.KTan | 1554 | 68 |
| IHE.Games.Server | 335 | 5 |
| IHE.Games.KTan.Server | 6 | 1 |
| SUBTOTALES | **3949** | **163** |
| IHE.Games.Client | 1212 | 41 |
| IHE.Games.KTan.Client | 1769 | 32 |
| SUBTOTALES | **2981** | **73** |
| TOTALES | **6930** | **236** |

Tabla 6: Métricas de la implementación realizada

## 9.3 Asset utilizados

Para el desarrollo del juego de K-Tan se han utilizado una serie de Asset adquiridos en el Unity Asset Store.

### 9.3.1 JSON .NET For Unity

Este Asset ofrece herramientas y funcionalidades que permite entre otros, serializar objetos a texto en formato JSON. No solo es completamente compatible para WebGL, sino que además ofrece una serie de funcionalidades indispensables para la implementación de este proyecto.

La herramienta, ofrece la posibilidad serializar y volver a deserializar un objeto manteniendo las referencias entre objetos así como sus tipos, sin importar la complejidad de la estructura de datos ni el polimorfismo de clases.

Por otro lado, permite configurar algunos parámetros que permiten reducir la longitud del JSON, tales como ignorar valores null o valores por defecto.

Otra de las funcionalidades más relevantes, son las etiquetas que ofrece. La etiqueta [JsonIgnore] se puede marcar una variable o propiedad de una clase para que el serializador la ignore. Así mismo, mediante la etiqueta [JsonProperty] se puede marcar una variable o propiedad de una clase para que el serializador la incluya en el JSON independientemente del modificador de acceso de dicha variable o propiedad. Estas dos etiquetas, han permitido implementar el modelo de datos y el protocolo de comunicación entre el cliente y el servidor tal y como se ha definido en el cápitulo de *Análisis y diseño*.

Adicionalmente, ofrece otras etiquetas como [JsonContructor] para indicar al deserializador que constructora de la clase utilizar al deserializar el objeto.

### 9.3.2 Simple Web Sockets for Unity WebGL

Este Asset de Unity ofrece las funcionalidades para implementar conexiones de websockets, tanto al desarrollar para WebGL como para el resto de plataformas disponibles en Unity. Gracias a este Asset se ha podido implementar las conexiones y el envío de datos entre los clientes y el servidor de K-Tan.

### 9.3.3 Localization package

Este Asset de Unity ha simplificado la implementación del juego en varios idiomas.

Para hacer uso de este Asset, primero se deben insertar los textos de la aplicación en una hoja de cálculos de Google. En cada fila se escribe cada uno de los textos, de manera que la primera columna contenga un código que identifique a ese texto y el resto de columnas contengan el propio texto en cada uno de los idiomas de la aplicación.

Mediante este Asset se puede, de manera sencilla, obtener los textos desde la hoja de cálculo y volcarlos en unos documentos XML, que luego el propio Asset utiliza para obtener cada uno de los textos. Para modificar el idioma y obtener los textos, basta con hacer uso de las funciones que trae el propio Asset entre las que se destacan *Language.Get(string key)* y *Language.SwitchLanguage(LanguageCode code)*.

### 9.3.4 NGUI: Next-Gen UI

Este Asset de Unity, disponible en UnityAssetStore, ofrece funcionalidades adicionales de las que provee Unity para el desarrollo de interfaces de usuario. En especial, cabe destacar que permite establecer anclajes entre los componentes de la interfaz así como otras funcionalidades que han permitido implementar unas interfaces de usuario adaptables a la gran mayoría de proporciones y resoluciones del mercado.

Por otro lado, permite crear y trabajar con los atlas, que consisten en una colección de imágenes que se agrupan en una única imagen. Esto ayuda a mejorar el rendimiento de la aplicación.

### 9.3.5 TexturePacker Importer

Este Asset de Unity permite reducir la cantidad de materiales utilizados en los elementos 3D y mejorar así el rendimiento del juego. Para ello, este Asset ofrece funcionalidades para trabajar con atlas de texturas y utilizarlos para los materiales de los objetos 3D. Dichos atlas, se pueden crear mediante la aplicación TexturePacker de la misma desarrolladora que este Asset, CodeAndWeb.

### 9.3.6 Master Audio: AAA Sound

Este Asset deUnity proporciona funcionalidades para gestionar y reproducir listas de audios y efectos sonoros. Permite configurar las transiciones entre audios y ajustar una gran variedad de parámetros de audio.

## 9.4 Otras herramientas utilizadas

Tal y como se ha comentado, principalmente se ha hecho uso de Unity y de Visual Studio para la implementación de este proyecto. Sin embargo, en este proyecto se han generado una serie de elementos gráficos, tanto elementos 2D como elementos 3D, que se han utilizado en las interfaces de usuario.

La herramienta utilizada para la generación de los elementos 3D ha sido Blender. Los elementos gráficos 2D, en cambio se han desarrollado en Adobe Photoshop CC 2015.

Por otro lado, ya se ha mencionado también el uso de la herramienta TexturePacker para agrupar las texturas utilizadas para los elementos 3D en un solo atlas.

# 10

# Pruebas

En este capítulo se describe la sistemática de pruebas que se ha llevado a cabo para asegurar la calidad del producto desarrollado y se detallan algunos de los casos de prueba realizados. Por otro lado, se describen las pruebas con usuarios reales realizadas en la última etapa de este proyecto para validar la usabilidad de la aplicación y la satisfacción de los usuarios.

## 10.1 Sistemática de pruebas

La sistemática de las pruebas realizadas ha consistido en diseñar y realizar pruebas estructurales, o de caja blanca para cada una de las acciones o clases `Action` implementadas. Estas pruebas han consistido en un total de 152 casos de prueba recogidos en 31 pruebas de unidad desarrolladas y ejecutadas en VisualStudio 2015.

Al igual que los elementos del modelo de datos y las acciones se han estructurado en distintos ensamblados, las pruebas implementadas también se han organizado de una manera similar, creando un paquete de pruebas unitarias por cada uno de los ensamblados de las acciones testadas y una prueba de unidad con los diferentes casos de prueba para cada acción.

Para cada caso de prueba, se define un estado concreto del modelo de datos y se comprueba el funcionamiento de la ejecución de la acción o el estado resultante del modelo de datos tras su ejecución.

A continuación se muestran, a modo de ejemplo, los 8 casos de prueba realizados para la acción `BuildSettlement` definidos en la prueba de unidad `BuildSettlement_Tests` del Ensamblado `IHE.Games.KTan.Tests`:

| | | |
|---|---|---|
| **Número de prueba** | BuildSettlement_Test01 | |
| **Descripción** | La partida se encuentra en la fase de preparación y se ejecuta la acción | |
| **Métodos/Herramientas** | Pruebas de unidad en Visual Studio | |
| **Prioridad** | ☐Alta ☒Media ☐Baja | |
| **Resultado esperado** | Durante la ejecución de la acción se lanza una excepción `NonExecutableAction` | |
| **Resultados obtenido** | ☒Satisfactorio | ☐Insatisfactorio |

| | | |
|---|---|---|
| **Número de prueba** | BuildSettlement_Test02 | |
| **Descripción** | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien no sea el turno. | |
| **Métodos/Herramientas** | Pruebas de unidad en Visual Studio | |
| **Prioridad** | ☐Alta ☒Media ☐Baja | |
| **Resultado esperado** | Durante la ejecución de la acción se lanza una excepción `NonExecutableAction` | |
| **Resultados obtenido** | ☒Satisfactorio | ☐Insatisfactorio |

| Número de prueba | BuildSettlement_Test03 | |
|---|---|---|
| Descripción | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien sea el turno, indicando la construcción en una intersección inexistente. | |
| Métodos/Herramientas | Pruebas de unidad en Visual Studio | |
| Prioridad | ☐Alta ☒Media ☐Baja | |
| Resultado esperado | Durante la ejecución de la acción se lanza una excepción `ActionException` | |
| Resultados obtenido | ☒Satisfactorio | ☐Insatisfactorio |

| Número de prueba | BuildSettlement_Test04 | |
|---|---|---|
| Descripción | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien sea el turno, indicando la construcción en una intersección no vacía. | |
| Métodos/Herramientas | Pruebas de unidad en Visual Studio | |
| Prioridad | ☐Alta ☒Media ☐Baja | |
| Resultado esperado | Durante la ejecución de la acción se lanza una excepción `NonExecutableAction` | |
| Resultados obtenido | ☒Satisfactorio | ☐Insatisfactorio |

| Número de prueba | BuildSettlement_Test05 | |
|---|---|---|
| Descripción | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien sea el turno, indicando la construcción en una intersección vacía pero con otro poblado en una de sus intersecciones adyacentes. | |
| Métodos/Herramientas | Pruebas de unidad en Visual Studio | |
| Prioridad | ☐Alta ☒Media ☐Baja | |
| Resultado esperado | Durante la ejecución de la acción se lanza una excepción `NonExecutableAction` | |
| Resultados obtenido | ☒Satisfactorio | ☐Insatisfactorio |

| Número de prueba | BuildSettlement_Test06 | |
|---|---|---|
| Descripción | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien sea el turno, indicando la construcción en una intersección vacía sin otros poblados o ciudades en sus intersecciones adyacentes. El jugador no dispone de los recursos suficientes. | |
| Métodos/Herramientas | Pruebas de unidad en Visual Studio | |
| Prioridad | ☐Alta ☒Media ☐Baja | |
| Resultado esperado | Durante la ejecución de la acción se lanza una excepción `NonExecutableAction` | |
| Resultados obtenido | ☒Satisfactorio | ☐Insatisfactorio |

| Número de prueba | BuildSettlement_Test07 | |
|---|---|---|
| Descripción | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien sea el turno, indicando la construcción en una intersección vacía sin otros poblados o ciudades en sus intersecciones adyacentes. El jugador dispone de los recursos necesarios, pero no tiene poblados disponibles para situar en el tablero. | |
| Métodos/Herramientas | Pruebas de unidad en Visual Studio | |
| Prioridad | ☐Alta ☒Media ☐Baja | |
| Resultado esperado | Durante la ejecución de la acción se lanza una excepción `NonExecutableAction` | |
| Resultados obtenido | ☒Satisfactorio | ☐Insatisfactorio |

| Número de prueba | BuildSettlement_Test07 | |
|---|---|---|
| Descripción | La partida se encuentra en la fase de turnos y se ejecuta la acción por parte de un jugador de quien sea el turno, indicando la construcción en una intersección vacía sin otros poblados o ciudades en sus intersecciones adyacentes. El jugador dispone de un poblado para situar en el tablero y de los recursos necesarios. | |
| Métodos/Herramientas | Pruebas de unidad en Visual Studio | |
| Prioridad | ☐Alta ☒Media ☐Baja | |
| Resultado esperado | El jugador tiene un recurso de madera, uno de arcilla uno de cereales y uno de lana menos que al inicio de la ejecución. Se ha situado un poblado en la intersección indicada y el jugador dispone de un poblado menos para colocar en el tablero. | |
| Resultados obtenido | ☒Satisfactorio | ☐Insatisfactorio |

Tabla 7: Casos de prueba de la acción BuildSettlement

## 10.2 Pruebas con usuarios

En la última etapa del ciclo de este proyecto se han desarrollado unas pruebas con usuarios para validar la usabilidad de las aplicaciones desarrolladas para IHEGames y verificar la satisfacción de los usuarios.

### 10.2.1 Tipos de usuarios

IHEGames establece una serie de usuarios tipo para la realización de las pruebas con usuarios, en base al espectro de dedicación de dicho usuarios en los juegos. Se distinguen las siguientes cuatro categorías de usuarios.

Casual: Se denomina así a los jugadores que no suelen emplear mucho tiempo en un juego dado, caracterizándose, en muchos casos, por jugar a muchos juegos, pero durante poco tiempo o en intervalos irregulares. Además, estos jugadores no suelen estar comprometidos propiamente a conseguir todos los objetivos posibles en un juego. Un gran grupo de ellos no tienen interés en mejorar sus habilidades ya que lo consideran solo un pasatiempo.

Regular: Muchas personas no consiguen encajar en la categoría de jugador *Hardcore*, pero a su vez tampoco en jugadores casuales, para estos se asigna el término de jugadores regulares el cual se refiere a un jugador que juega de manera habitual, tiene ciertos conocimientos de los videojuegos, pero no busca un gran reto como los jugadores *Hardcore*; son competitivos pero no tienen interés en ser los mejores.

Hardcore: Se caracteriza por ser un jugador que dedica muchas horas al día a jugar videojuegos y que busca mejorar constantemente y tener puntuaciones máximas. Estos jugadores no quedan satisfechos con terminar un videojuego de manera habitual, y buscan siempre conseguir todo lo que se pueda en la mayor dificultad posible. Estos jugadores son verdaderamente aficionados a los videojuegos, no buscan solo un medio de entretenimiento sino un reto o una aventura.

Profesional: Se caracteriza por ser un jugador con habilidades extraordinarias para jugar, y por ello, es considerado un jugador experto por la comunidad de los videojuegos. Un gran grupo de estos jugadores logran ganar dinero por sus habilidades, integrarse a un equipo y participar en torneos.

Para las pruebas realizadas, se han buscado jugadores de las 3 primeras categorías, ya que IHEGames tiene estimado que con estos tres tipos de usuario se cubre más del 90% de los consumidores de videojuegos.

### 10.2.2 Tipos de dispositivos

Para las pruebas realizadas se han utilizado dispositivos con diferentes tamaños y resoluciones de pantalla para comprobar la usabilidad de las aplicaciones y sus interfaces. Por ello, las pruebas se han llevado a cabo tanto en PCs y portátiles, como en tabletas y móviles.

Tal y como está definido en su alcance, el juego K-Tan se ha desarrollado para su ejecución en dispositivos Android y en la web a través del Portal D20; debido a esto, los dispositivos móviles y tabletas utilizadas en estas pruebas disponen del sistema operativo Android.

A continuación se listan los dispositivos utilizados en las pruebas:

| Núm. | Tipo de dispositivo | Pantalla | |
|---|---|---|---|
| | | Tamaño | Resolución |
| D1 | Móvil Android | 5“ | 480x854 |
| D2 | Móvil Android | 5“ | 1280 x 720 |
| D3 | Tableta Android | 7“ | 1024x600 |
| D4 | Tableta Android | 10.1” | 1280x800 |
| D5 | Portátil | 14” | 1366x768 |
| D6 | Portátil | 15.6” | 1920x1080 |
| D7 | PC | 24” | 1680x1050 |
| D8 | PC | 24” | 1920x1080 |

Tabla 8: Dispositivos utilizados en las pruebas con usuarios

### 10.2.3 Pruebas realizadas

Para estas pruebas se ha reunido a 8 usuarios, distribuidos entre las 3 categorías de usuario descritas en el apartado *Tipos de usuario*. Concretamente han sido 3 jugadores casuales, 3 jugadores regulares y 2 jugadores *hardcore*, entre los cuales 1 de los jugadores casuales, 2 de los regulares y uno de los *hardcore* conocían de antemano las reglas de *Los colonos de Catan* en el cual está basado el juego K-Tan. Dado que las pruebas se han realizado simultáneamente con los 8 usuarios, cada uno ha hecho uso de uno de los 8 dispositivos descritos en el apartado anterior *Tipos de dispositivos*.

A lo largo de estas pruebas cerradas, los usuarios han tenido libertad de explorar las distintas funcionalidades a su gusto, pero han sido guiados a través de una serie de pasos que se han dividido en dos etapas. En la primera etapa, los usuarios han podido probar algunas de las funcionalidades del Portal D20:

- Los usuarios han realizado los pasos necesarios para completar su registro en el Portal D20 de IHEGames. Los registros de las cuentas se han realizado tanto creando una cuenta mediante usuario y contraseña como a través de otro proveedor de cuentas de usuario externo como Google, Facebook o Twitter.
- Los usuarios han podido completar sus perfiles y modificar los datos de sus cuentas.
- Los usuarios han explorado las secciones y funcionalidades del Portal D20.
- Finalmente, los usuarios han rellenado una encuesta sobre su experiencia en el Portal D20.

En la segunda etapa los usuarios han podido probar la aplicación de K-Tan:

- Los usuarios que han accedido a través de la App de Android de K-Tan, se han identificado.
- Los jugadores han explorado las funcionalidades de la interfaz general de K-Tan.
- Se han creado dos partidas simultáneas de 4 jugadores .Los 8 jugadores se han distribuido en estas dos partidas en base a si conocían de antemano las reglas del juego.
- Se han explicado las reglas de juego a los jugadores que no las conocían.
- Los jugadores han explorado las funcionalidades de la interfaz de partida.
- Los jugadores de la primera partida en concluir, se han unido a la otra partida como espectadores.
- Tras la finalización de las dos partidas, los usuarios han rellenado una segunda encuesta sobre su experiencia con K-Tan.

### 10.2.4 Resultados

Las encuestas realizadas a los usuarios se han realizado a mediante la aplicación online de Google Forms. Esta aplicación web permite generar formularios de una forma fácil y sencilla mediante las herramientas de que dispone. Y ofrece la posibilidad de responder a los formularios a través de un enlace. Por otro lado, almacena todas las respuestas enviadas y genera gráficas a partir de dichas respuestas, facilitando la compresión y análisis de los resultados.

A continuación se muestran algunas de las gráficas obtenidas en las encuestas y referentes a las características de los usuarios comentadas a lo largo de este capítulo (se reservan el resto de los resultados obtenidos):

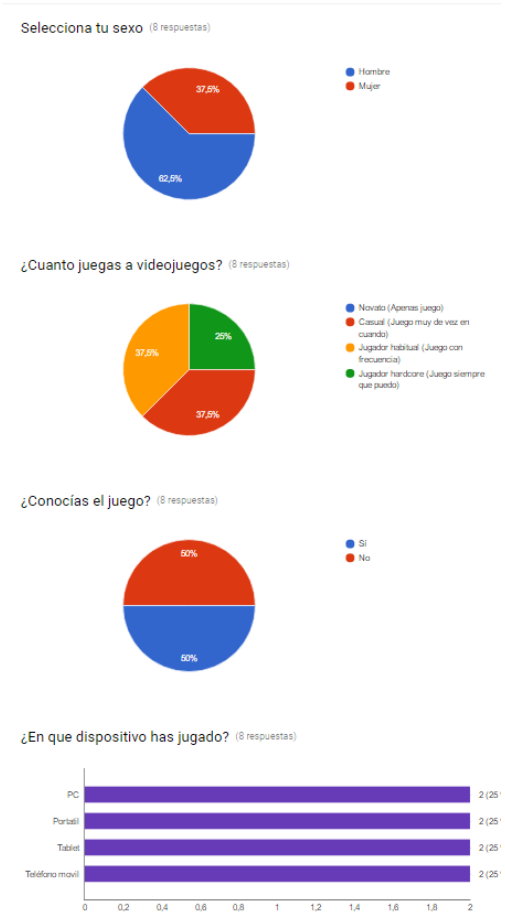


Figura 42: Gráficas de las respuestas de las encuestas

A partir de las valoraciones recogidas en estas encuestas, IHEGames inferirá las nuevas especificaciones para las siguientes versiones del Portal D20 y K-Tan.

# 11

# Puesta en producción

A lo largo de este capítulo, se describen algunos de los pasos más significativos que se han llevado a cabo en la puesta en producción que ha sido necesaria para poder llevar a cabo las pruebas con usuarios.

Para empezar, se describe la configuración que ha sido necesaria para integrar los sistemas de IHEGames con K-Tan. A continuación se describen los auto-certificados que se han generado y que han sido necesarios para que las comunicaciones fueses seguras. Finalmente, se desglosan los pasos que han sido necesarios para poder desplegar la versión Android y la versión web de K-Tan, una en dispositivos Android y la otra en el Portal D20.

## 11.1 Conectar con los servicios de IHEGames

Visual Studio permite una integración sencilla de los servicios web WCF. Proporciona una sencilla interfaz de usuario mediante la cual se puede localizar un servicio web y añadir su referencia al proyecto. Al hacer eso, Visual Studio genera automáticamente la interfaz con los métodos y el modelo de datos que proporciona el servicio.

Una vez añadidas las referencias de los servicios de cuentas de usuario, Elo y Honor, Visual Studio genera una configuración básica para las comunicaciones con estos servicios. Por ello, ha sido necesario realizar las configuraciones necesarias para la comunicación en base a las especificaciones de seguridad concretas de los servicios de IHEGames.

A continuación se muestra un extracto del archivo XML de configuración en el cual se han definido dichas configuraciones de las comunicaciones con los servicios.

```
    <wsHttpBinding>
        <binding name="WSHttpBinding_IELOService">
            <security mode="TransportWithMessageCredential">
                <transport clientCredentialType="Certificate" />
                <message clientCredentialType="UserName" />
            </security>
        </binding>
        <binding name="WSHttpBinding_IVGHonorService">
            <security mode="TransportWithMessageCredential">
                <transport clientCredentialType="Certificate" />
                <message clientCredentialType="UserName" />
            </security>
        </binding>
        <binding name="WSHttpBinding_IAccountService">
            <security mode="TransportWithMessageCredential">
                <transport clientCredentialType="None" />
                <message clientCredentialType="UserName" />
            </security>
        </binding>

    </wsHttpBinding>
```

Figura 43: Configuración de la comunicación con los servicios de IHEGames s

## 11.2 Certificados del servidor

Para poder establecer las conexiones a través de SSL/TLS ha sido necesario crear una serie de certificados en la puesta en producción. Dado que en este ciclo del proyecto, no está incluida la puesta en explotación de K-Tan, no se ha recurrido a una entidad certificadora reconocida.

En su lugar, se ha creado un certificado auto-firmado de una autoridad certificadora propia, mediante el cual se ha firmado el certificado del servidor de K-Tan.

Para poder conectarse con el servidor, ha sido necesario que los dispositivos donde se han ejecutado los clientes de K-Tan tuviesen instalado el certificado creado de la autoridad certificadora para que puedan considerar el certificado del servidor de K-Tan como un certificado de confianza.

A su vez, dado el servidor de K-Tan actúa como cliente para los servicios de IHEGames, ha sido necesario que la máquina en la que se ejecutaba el servidor de K-Tan tuviese instalado el certificado correspondiente para poder establecer unas comunicaciones seguras con los servicios.

## 11.3 Creando una App de K-Tan para Android

Una de las mayores ventajas de Unity3D es que permite desarrollar juegos para muchas plataformas con realmente muy poco trabajo extra.

Para poder compilar para cada dispositivo es necesario algún trabajo extra mínimo. En el caso de Android, ha sido necesario descargar e integrar el apk de Google para Android y mediante el cual Unity es capaz de compilar la App de K-Tan.

## 11.4 Integrando K-Tan el Portal Web

Al jugar a K-Tan desde el portal D20, para evitar que el usuario tenga que identificarse tanto en el portal como en K-Tan, se establece un protocolo para que el juego pueda obtener los datos necesarios para identificar al usuario sin necesidad de que éste proporcione ningún tipo de credencial.

Una vez se carga y se inicia el juego en el navegador, el propio juego utiliza una función de javascript que hace una petición AJAX al portal D20. Éste, que ya ha tenido que identificar al usuario previamente y tiene los permisos necesarios, invoca el servicio de cuentas de usuario para solicitar un Token. El portal D20 devuelve el Token solicitado junto con la id del usuario. Mediante estos datos, el juego es capaz de identificar al usuario y obtener su información a través del servicio de cuentas.

Este Token, es de un solo uso y además de tiempo limitado, tras el cual se caduca y queda invalidado.

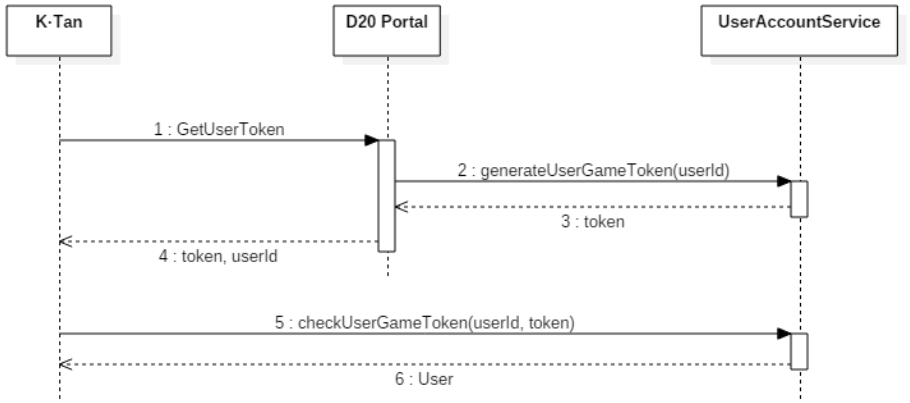


Figura 44: Diagrama de secuencia de la identificación de usuarios mediante tokens

Gracias a la directiva **#if** de C#, se le puede indicar al compilador qué partes de código compilar y cuáles no, dependiendo de la plataforma para la cual se compila el proyecto de Unity. De esta manera, se puede mantener las mismas escenas y elementos en el juego pero especificando en la escena *Login*, mencionada en el capítulo *Implementación*, que la identificación del usuario se haga de esta otra manera, sin mostrar la interfaz de identificación de usuario.

# 12

# Gestión

En este capítulo se describen, las áreas más relevantes en lo referente a la gestión llevada cabo a lo largo de todo este proyecto.

Dada las dependencias directas que existen entre el juego K-Tan, desarrollado en este proyecto, y el Portal D20 y los servicios de IHEGames, desarrollados en otro proyecto en paralelo, la gestión del tiempo y de las comunicaciones es fundamental a lo largo de este proyecto.

Además de la gestión de las comunicaciones y del tiempo, en este capítulo, también se desarrolla la gestión del alcance, los costes y las adquisiciones, puesto que han sido también de gran relevancia para este proyecto.

## 12.1 Gestión del alcance

Debido a las dependencias que existen entre este proyecto y el proyecto en el que se desarrolla el Portal D20 y los servicios de IHEGames, se establece al inicio del proyecto que todo cambio en el alcance del proyecto se debe realizar con la aprobación de IHEGames y de los responsables de ambos proyectos. Sin embargo, el alcance de este proyecto no ha sufrido grandes modificaciones a lo largo del proyecto.

Cabe destacar que, tal y como se indica en el apartado de *Gestión de costes*, la mayoría de los paquetes de trabajo han sufrido un sobrecoste con respecto a las estimaciones realizadas. A pesar de ello, se decide asumir el coste añadido y no excluir nada del alcance original.

Por otro lado, tras la realización del prototipo de la parte servidor de K-Tan, se comprueba que su diseño puede ser adecuado para realizar el servicio de chat. Debido a esto, se plantea junto con los promotores de IHEGames la inclusión del desarrollo del servicio de chat al alcance de este proyecto. Sin embargo, los costes en dedicaciones que conlleva la realización de dicho servicio, añadidos a los sobrecostes antes mencionados, superarían con creces el coste de dedicaciones asumido para la realización de este proyecto. Por lo tanto, finalmente se decide posponer su desarrollo a proyectos futuros.

## 12.2 Gestión de costes

Dada la naturaleza académica de este proyecto, se establecen como costes las horas dedicadas en la realización de los distintos paquetes de trabajo de este proyecto. Por otro lado, los costes económicos que ha habido debido algunas de las adquisiciones realizadas en este proyecto se detallan en el apartado de *Gestión de adquisiciones*.

A continuación, se recogen las horas necesarias estimadas al inicio del proyecto y las horas reales de dedicación que han sido necesarias, así como los desvíos resultantes entre dichos valores.

| BLOQUE DE TRABAJO | TAREA | DEDICACIONES | | |
|---|---|---|---|---|
| | | Estimadas | Reales | Desvíos |
| ANÁLISIS, DISEÑO E IMPLEMEN-TACIÓN | I+D | 16h | 30h | +14h |
| | Requisitos | 12h | 11h 30’ | -30’ |
| | Casos de uso | 4h | 4h | 0 |
| | Diseño de la arquitectura y las comunicaciones | 12h | 8h | -4h |
| | Diseño del modelo de datos | 12h | 14h | +2h |
| | Diseño de la lógica de negocios | 12h | 15h | +3h |
| | Diseño de la interfaz de usuario | 12h | 13h 30’ | +1h 30’ |
| | Implementación del modelo de datos base | 16h | 13h | -3h |
| | Implementación de la lógica de negocios base | 16h | 23h | +7h |
| | Implementación del modelo de datos específico de K-Tan | 16h | 12h | -4h |
| | Implementación de la lógica de negocios específica de K-Tan | 16h | 15h | -1h |
| | Implementación de las interfaces de usuario | 80h | 105h | +25h |
| | Generación del material gráfico de las interfaces de usuario | 16h | 10h 30’ | -5h 30’ |
| | **SUBTOTALES** | **240** | **274h 30’** | **+34h 30’** |
| PRUEBAS | Generación de pruebas de unidad | 20h | 23h | +3h |
| | Corrección de errores | 20h | 35h | +15h |
| | **SUBTOTALES** | **40h** | **58h** | **+18h** |
| PUESTA EN PRODUCCIÓN | Despliegue web | 8h | 7h | -1h |
| | Despliegue Android | 2h | 2h | 0 |
| | Pruebas con usuarios reales | 4h | 4h | 0 |
| | Recogida de datos de usuarios | 2h | 3h | +1h |
| | **SUBTOTALES** | **16h** | **16h** | **0** |
| GESTIÓN | Planificación | 8h | 6h | -2 |
| | Adquisiciones | 8h | 13h | +5h |
| | Seguimiento y Control | 4h | 3h 30’* | -30’* |
| | Reuniones | 12h | 14h* | +2h* |
| | **SUBTOTALES** | **32h** | **36h 30’** | **+4h 30’** |
| TRABAJO ACADÉMICO | Memoria | 60h | 80h* | **+20h*** |
| | Defensa | 10h | 10h* | **0h*** |
| | **SUBTOTALES** | **70h** | **90h** | **+20h** |
| **TOTALES** | | **398** | **475h** | **+77h** |

Tabla 9: Costes de tiempo del proyecto

* *Valores estimados dado que la tarea no ha concluido*

Como se puede apreciar ha habido en la mayoría de paquetes de trabajo un sobrecoste con respecto a las estimaciones de tiempo realizadas al inicio del proyecto. A pesar de ello, el sobrecoste general del proyecto ha sido de un 19% sobre las horas totales estimadas, un sobrecoste asumible dada la naturaleza y características de este proyecto.

Cabe destacar que, a partir de estas dedicaciones se puede presuponer que el desarrollo de nuevos juegos, teniendo ya la base que se ha construido en este proyecto, se podría llegar a realizar a un ritmo de 1-2 meses por juego.

## 12.3 Gestión de las comunicaciones

Se observa al inicio, tanto en del proyecto de K-Tan como en del de los servicios de IHEGames y el Portal D20, que ambos proyectos comparten los mismos interesados. En la siguiente tabla se detallan todos los interesados identificados:

| INTERESADOS | |
|---|---|
| Responsable del proyecto de K-Tan | Iker Boyra Sarachaga |
| Responsable del proyecto de los servicios IHEGames y del portal D20 | Héctor Antruejo Escalante |
| Promotores de IHEGames | Héctor Antruejo Escalante<br>Iker Boyra Sarachaga |
| Tutor del Proyecto de Fin de Grado | José Miguel Blanco |

Tabla 10: Interesados del proyecto

A continuación se indica la autoridad (“Poder”: Bajo/Medio/Alto) y el nivel de preocupación (“Interés”: Bajo/Medio/Alto) con respecto a los resultados de este proyecto de cada uno de los interesados identificados para cada uno de los bloques principales de este proyecto.

| | PRODUCTO | | ACADÉMICO | | GESTIÓN | |
|---|---|---|---|---|---|---|
| | INTERÉS | PODER | INTERÉS | PODER | INTERÉS | PODER |
| Responsable del proyecto de K-Tan | Alto | Alto | Alto | Alto | Alto | Alto |
| Responsable del proyecto de los servicios IHEGames y del portal D20 | Medio | Medio | Bajo | Bajo | Alto | Medio |
| Promotores de IHEGames | Alto | Alto | Bajo | Bajo | Alto | Alto |
| Tutor del Proyecto de Fin de Grado | Medio | Bajo | Alto | Alto | Medio | Bajo |

Tabla 11: Interés y poder de los interesados del proyecto

Dada las dependencias que existen entre ambos proyectos, y el interés y el poder que se deriva de dicha dependencia, es importante mantener una buena comunicación entre los responsables de ambos proyectos, e indirectamente con los promotores de IHEGames.

Para mantener una buena comunicación y visibilidad de los proyectos, se definen dos líneas de comunicación principales.

Mediante la herramienta online de Mingle, ambos responsables registran diariamente el progreso y el estado de sus respectivos proyectos. De esta manera, cada responsable es capaz de visionar en cualquier momento el avance y características de las fases, iteraciones o historias ("Story") de cualquiera de los dos proyectos, entre las que se destaca:

- Estado completado/en curso/pendiente
- Fecha de inicio y de finalización
- Estimación de horas inicial
- Estimación de horas hasta su final
- Coste de horas dedicadas

Por otro lado, se establecen reuniones periódicas entre ambos responsables, tanto presenciales como vía Skype, para compartir los aspectos relevantes de ambos proyectos entre ambos responsables e indirectamente a los promotores de IHEGames.

Adicionalmente, se establece un sistema de información compartido mediante Google Drive para poder compartir la documentación relevante entre ambos responsables de proyecto.

Respecto a la comunicación con el tutor del Proyecto de Fin de Grado, ésta se realiza mediante reuniones acordadas previamente mediante mensajes de correo electrónico. Gestión de adquisiciones

## 12.4 Gestión del tiempo

El tiempo a lo largo de este proyecto se ha gestionado principalmente en base al ciclo de vida del proyecto teniendo en cuenta el calendario académico y las fechas en las que se han previsto causas externas que limiten el tiempo de dedicación disponible para la realización del proyecto.

En la siguiente figura se muestra la distribución preestablecida de los distintos paquetes de trabajo a lo largo del tiempo así como las transiciones entre las fases del ciclo de vida del proyecto.

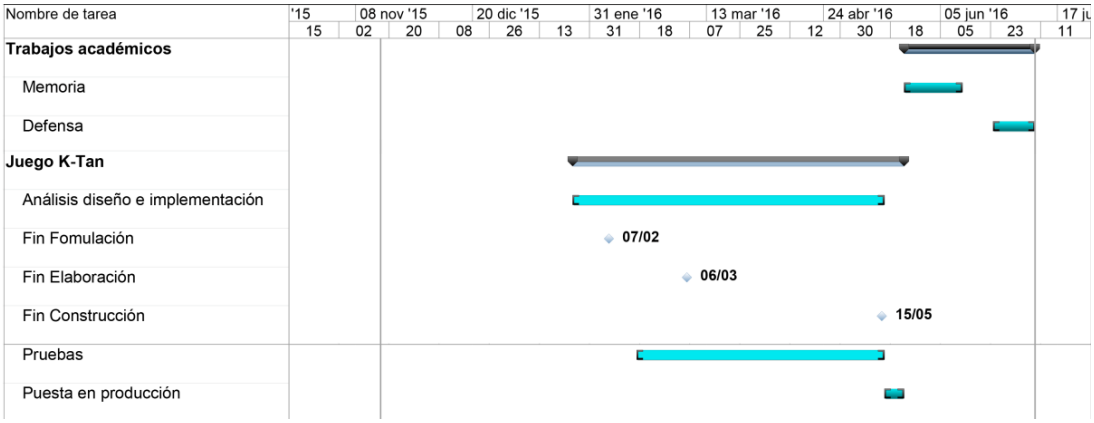


Figura 45: Diagrama de barras del proyecto

Dado que el juego K-Tan depende directamente, tanto de los servicios de IHEGames como del propio portal D20, ha sido necesario gestionar el tiempo conjuntamente con el proyecto paralelo responsable de estos sistemas.

En la siguiente figura, se muestra la distribución de los distintos paquetes de trabajo pertenecientes al proyecto paralelo en el que se desarrollan los servicios de IHEGames y el Portal D20.

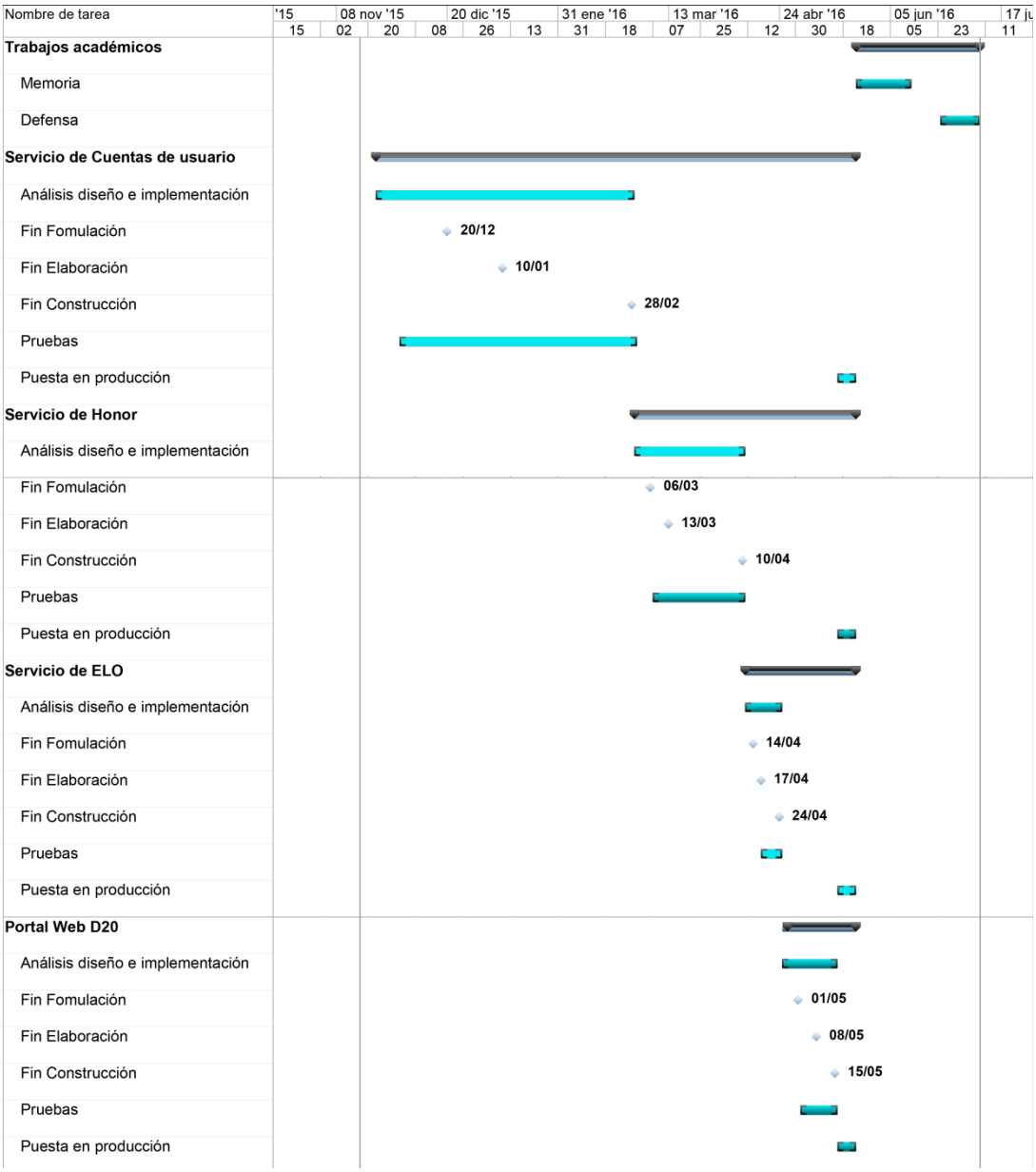


Figura 46: Diagrama de barras del proyecto de los servicios IHEGames y el Portal D20

Junto a la distribución de las actividades, se establecen los hitos más significativos de ambos proyectos. Estos hitos se muestran en la siguiente figura.

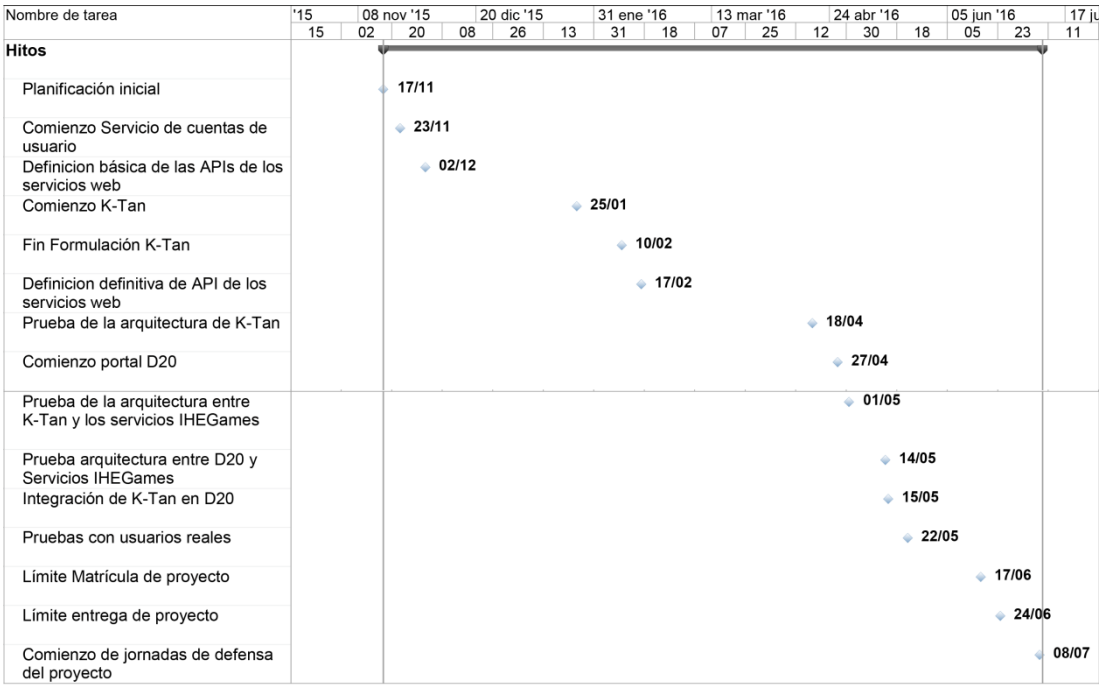


Figura 47: Hitos del proyecto

Tanto la distribución de las actividades como los hitos, se preestablecen teniendo en cuenta las dependencias que existen entre los sistemas desarrollados en cada proyecto y los costes de tiempo estimados de cada paquete de trabajo. A pesar de ello se establecen las fechas finales con márgenes de tiempo suficientes para intentar garantizar la finalización de las actividades y evitar retrasar las actividades dependientes en caso de sobrecostes o eventos inesperados.

Como se muestra en el apartado de Gestión de Costes, los paquetes de trabajo de este proyecto han sufrido, en general, un sobrecoste con respecto a las estimaciones realizadas. A pesar de ello, los márgenes de las fechas han permitido respetar los hitos del proyecto.

## 12.5 Gestión de las adquisiciones

A lo largo de este proyecto, se han utilizado una serie de Asset de Unity. Algunos de estos Asset ya estaban accesibles desde el inicio de este proyecto, puesto que ya eran parte de los recursos disponibles de IHEGames. Sin embargo, ha sido necesario adquirir otros desde la plataforma de Unity Asset Store.

En lo que respecta a las adquisiciones, los nuevos Asset forman a pasar parte de los activos de IHEGames. Dado que el posible coste de los Asset lo cubre IHEGames, ha sido necesaria su aprobación a la hora de adquirirlos.

La mayoría de los Asset adquiridos están accesibles vía Unity Asset Store de manera gratuita. Sin embargo, se ha visto adecuado adquirir el Asset "JSON .NET for Unity", a pesar de no ser gratuito. Tras una previa valoración, IHEGames ha aceptado cubrir el coste de este nuevo Asset e incorporarlo a sus activos disponibles.

A continuación se muestra una tabla con los costes de las adquisiciones del proyecto:

| CONCEPTO | COSTE |
|---|---:|
| TexturePacker Importer | 0.00€ |
| Simple Web Sockets for Unity WebGL | 0.00€ |
| JSON .NET For Unity | 27.02€ |
| **SUBTOTAL** | **27.02€** |

Tabla 12: Costes económicos del proyecto

# 13

# Conclusiones

En este capítulo, se desarrollan unas conclusiones personales sobre el trabajo en equipo y el desarrollo con la herramienta Unity, y se presentan unas reflexiones de los futuros pasos de K-Tan y los sistemas de IHEGames.

## 13.1 Conclusiones del trabajo en equipo

Uno de los puntos que considero más característicos en este proyecto es el hecho de haber trabajado en paralelo a otro proyecto del cual se depende en gran medida y que ha marcado el camino recorrido a lo largo de ambos desarrollos.

Como puede parecer en un principio, trabajar en paralelo ofrece ciertas ventajas, en especial en lo que respecta a plazos. Sin embargo, uno no siempre es consciente de los enormes costes añadidos que pueden representar la gestión y la organización de uno o más proyectos donde se tengan que coordinar más de una persona. Se puede tender a menospreciar la dificultad que representa esta gestión de equipo, al compararla tal vez con la gestión personal. Pero el haber experimentado la gestión en equipo en un proyecto de cierta envergadura como éste no me hace más que confirmar que una buena organización en equipo es siempre más costosa de lo que parece, y que por raro que parezca, una mala gestión o una gestión nula conlleva un coste aún mayor. A su vez, estoy convencido que estos costes crecen de manera exponencial a medida que aumenta el tamaño del equipo de trabajo.

A raíz de este proyecto, me llevo conmigo muchas pequeñas lecciones y reflexiones que, en general, pueden resultar innegables y considerarse de sentido común. Sin embargo, a la hora de la verdad, uno no llega a ser consciente de ellas e ignora la importancia que pueden tener para el proyecto, sobre todo si no se tiene suficiente experiencia trabajando en equipo. Y es que al trabajar en equipo, acciones tan sencillas como establecer desde un inicio unas pautas comunes, un mismo marco de trabajo para todos los integrantes de un equipo, puede suponer un ahorro sustancial, no solo en tiempo sino también en futuros cambios y quebraderos de cabeza.

Tanto mi compañero como yo sentimos que nos hemos coordinado suficientemente bien, pero somos conscientes de cuanto, en lo referente a la gestión, debemos ir mejorando con la práctica y la experiencia, en especial si aspiramos a que nuestro equipo de trabajo crezca con el tiempo. También somos conscientes de algunos de los aspectos que nos han funcionado y tenemos ciertas ideas de cómo refinarlos para que funcionen mejor en un futuro.

Por destacar alguna, hemos comprobado lo efectiva que puede resultar una visibilidad del trabajo y el progreso de cada miembro del equipo y de los proyectos relacionados. Esto no solo aporta información relevante para cada proyecto en todo momento sino que además, con un enfoque adecuado, puede proporcionar un estímulo moral que ayude a uno mismo a avanzar a ritmo del equipo.

Por otro lado, considero que un trabajo en equipo como este representa un cierre adecuado para esta etapa académica que se aproxima a su final. Este proyecto nos ha permitido poner en práctica no solo muchos de los conocimientos técnicos adquiridos a lo largo del grado en informática y en especial de la ingeniería del software, sino también muchas de las

herramientas que quizás en un proyecto para uno mismo no resultan tan necesarias pero que en un trabajo en equipo, y orientado a crecer en el futuro a manos de otros desarrolladores, resultan fundamentales.

## 13.2 Conclusiones sobre Unity

Ésta no es la primera vez que he desarrollado con Unity. Anteriormente, he realizado pequeños proyectos con este motor de juegos que me han permitido aprender a utilizar esta herramienta y asimilar el modo de desarrollar en Unity. Sin esta experiencia previa con Unity, el coste de dedicaciones de este proyecto se habría visto incrementado en enorme medida. Y es que no solo habría que tener en cuenta el tiempo de aprendizaje de la herramienta, sino también que muchas de las decisiones y diseños iniciales así como la elección de los Asset, parten de la experiencia previa con Unity.

Con esto no quiero desalentar a ningún lector que esté interesado en probar a desarrollar en Unity, sino al contrario, puesto que Unity ofrece un gran abanico de ventajas para el desarrollo de aplicaciones y en especial de juegos 3D. Sin embargo, quisiera destacar que por ser una herramienta especializada para el desarrollo de juegos 3D, el desarrollo en Unity es algo distinto al desarrollo en muchas de las herramientas de desarrollo software más comunes y es fundamental entender el funcionamiento de su motor de juego. Además, requiere un conocimiento adicional del desarrollo de juegos y de los entornos 3D que distan mucho del desarrollo de interfaces tradicionales en dos dimensiones.

Sin embargo, una vez se adquieren los conocimientos básicos, Unity puede resultar una herramienta muy útil para el desarrollo de cualquier aplicación. Una de sus principales ventajas es la posibilidad de desarrollar para un enorme abanico de las plataformas existentes en el mercado. Permite compilar un mismo proyecto a distintas plataformas de manera sencilla. Si no se hace uso de funcionalidades específicas de una plataforma concreta y se tiene en cuenta las posibles limitaciones de las plataformas para las que se desarrolla, en general, solo son necesarias unas pocas configuraciones para compilar a cada plataforma. Por ejemplo, para algunas plataformas como Windows o WebGL, se puede realizar la compilación directamente. Otras como Android requieren la descarga del Apk de Android de Google para que Unity pueda hacer uso de él al compilar el proyecto. IOS puede resultar una de las más complicadas, ya que requiere de un Mac y el pago de una licencia de desarrollador de IOS para realizar una compilación final a partir del XCode que se compila con Unity.

En el mundo actual de las aplicaciones multiplataforma, Unity puede resultar un gran aliado y en especial si se pretende explotar su capacidad para realizar juegos 3D.

## 13.3 Líneas futuras

En esta primera versión de K-Tan, el objetivo ha sido desarrollar un prototipo funcional del juego. Tal y como la propia palabra “prototipo” indica, K-Tan aún puede crecer y mejorarse. Me siento orgulloso de las funcionalidades desarrolladas y considero que se han alcanzado los objetivos establecidos para este ciclo del proyecto. Sin embargo, en el transcurso del desarrollo de K-Tan, me he convencido de que, ahora que se dispone de las funcionalidades básicas del juego, el próximo ciclo de este proyecto debe centrarse en los aspectos relativos a la interacción persona-computador. Las pruebas con usuarios realizadas no hacen más que confirmarme que éste puede ser el siguiente paso a seguir si se pretende lograr una completa satisfacción de los usuarios de K-Tan.

Por otra parte, K-Tan forma parte del núcleo principal de IHEGames, junto con el resto de los sistemas desarrollados en el otro proyecto paralelo. Este núcleo, es por ahora una pequeña base con la que empezar, pero con cada nuevo enfoque surgen nuevos sistemas y servicios que podrían formar parte de los sistemas de IHEGames y cuyo desarrollo e integración habrá que valorar en un futuro próximo.

Aún nos queda mucho trabajo por hacer, muchos errores de los que aprender y mucho que pulir, pero en general, como responsable de este proyecto y promotor de IHEGames, me alegra decir que me siento satisfecho con el trabajo realizado y el resultado que se ha obtenido en estos dos proyectos. No solo por el sistema desarrollado, sino también por lo que estos dos proyectos en sí representan, pues marcan el final de una etapa y, a su vez, el comienzo de otra mayor.

# Referencias


[1] Desarrollo para Comunidad Web de Videojuegos: Portal Web de Web de Videojuegos Online Multijugador. *Héctor Antruejo Escalante*. Junio 2016.
(Vid. Pág. 2)

[2] Asociación Española de Videojuegos. [Online] 19 de septiembre de 2015.
http://www.aevi.org.es/la-industria-del-videojuego/en-el-mundo/
(Vid. Pág. 6)

[3] Noticia el mundo juegos de mesa. [Online] 26 de Mayo de 2016.
http://www.elmundo.es/economia/2015/06/19/5583de0946163fc21e8b4572.html
(Vid. Pág. 6)

[4] Sistema de puntuación Elo. Wikipedia [Online] 26 de Mayo de 2016.
https://es.wikipedia.org/wiki/Sistema_de_puntuaci%C3%B3n_Elo
(Vid. Pág. 8)

[5] Https Everywhere. [Online] 26 de Mayo de 2016.
https://www.eff.org/HTTPS-EVERYWHERE
(Vid. Pág. 13)

[6] Google Security Blog. [Online] 26 de Mayo de 2015.
https://security.googleblog.com/2015/12/indexing-https-pages-by-default.html
(Vid. Pág. 13)

[7] Google Mobile Friendly. [Online] 26 de Mayo de 2016.
https://support.google.com/adsense/answer/6196932?hl=en
(Vid. Pág. 14)

[8] DB Engines Ranking. [Online] 26 de Mayo de 2016.
http://db-engines.com/en/ranking
(Vid. Pág.15)

[9] Reglamento de *Los Colonos de Catan*. [Online] 11 de Junio de 2016.
http://wp.devir.es/wp-content/uploads/2014/04/Catan-ManualDeLosColonos-Reglas.pdf
(Vid. Pág. 20, 33)

# Bibliografía


Unity3D. [Online] 24 de mayo de 2016.
https://unity3d.com/es.

Unity documentation [Online] 24 de mayo 2016
http://docs.unity3d.com/Manual/index.html

Unity Asset Store. [Online] 24 de mayo de 2016.
https://unity3d.com/es.

Mingle [Online] 24 de Mayo de 2016
https://www.thoughtworks.com/mingle/

W3C, [Online] 24 de Mayo de 2016.
http://www.w3c.es/

Microsoft Documentation. [Online] 24 de Mayo de 2016.
https://msdn.microsoft.com/en-us/library/ms123401.aspx

OWASP [Online] 24 de Mayo de 2016
https://www.owasp.org/index.php/Main_Page

Catan rules [Online] 24 de Mayo de 2016.
http://www.catan.com/en/download/?SoC_rv_Rules_091907.pdf

JSON.NET. [Online] 24 de Mayo de 2016.
http://www.newtonsoft.com/json

Google. Material Design . [Online] 24 de Mayo de 2016.
https://material.google.com/

# Glosario

**Asset**: hace referencia a un recurso o paquete de recursos de Unity, los cuales pueden ser modelos 3D, texturas y materiales, sistemas de partículas, música y efectos de sonido, tutoriales y proyectos, paquetes de scripts, extensiones para el editor y servicios en línea. . (Vid. Pág. 3, 11, 81, 86, 87, 88, 109, 113)

**Credenciales**: consisten en el uso de nombre de usuario o dirección de correo y contraseña para comprobar que un usuario es quien dice ser. (Vid. Pág. 7, 13, 44)

**Elo**: es un método matemático, basado en cálculo estadístico, para calcular la habilidad relativa de los jugadores de juegos como el ajedrez. (Vid. Pág. 8, 9, 10, 32, 45, 67)

**Experiencia**: la experiencia de un usuario simboliza su experiencia de juego y la cantidad de partidas en las que ha participado. (Vid. Pág.8, 9)

**Honor**: el Honor de un usuario simboliza su conducta o comportamiento general que ha demostrado en partidas anteriores y permite al resto de jugadores prever el tipo de jugador que es. (Vid. Pág. 8, 10, 31, 45, 46, 67)

**JSON**: acrónimo de JavaScript Object Notation, es un formato de texto ligero para el intercambio de datos. (Vid. Pág. 29, 67, 68, 69, 86, 87)

**SOAP**: siglas de Simple Object Access Protocol. Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. (Vid. Pág. 29)

**Token**: consiste en una secuencia alfanumérica, generado de forma aleatoria, que puede ser de una validez temporal. (Vid. Pág. 30, 99, 99)

**Ensamblado**: en Microsoft .NET framework, un ensamblado es principalmente una biblioteca de código compilado para ser utilizado en instalaciones, versionamiento y seguridad. (Vid. Pág. 62, 90)