Universidad de Valladolid

Escuela de Ingeniería Informática

Trabajo fin de grado

Grado en Ingeniería Informática
(Mención en Ingeniería de Software)

APP Gestión de partidos y jugadas de un equipo de balonmano

Autor:
D. Miguel Sánchez Rueda

Universidad de Valladolid

Escuela de Ingeniería Informática

Trabajo fin de grado

Grado en Ingeniería Informática
(Mención en Ingeniería de Software)

APP Gestión de partidos y jugadas de un equipo de balonmano

Autor:
D. Miguel Sánchez Rueda
Tutora:
Dña. María Margarita Gonzalo Tasis

Agradecimientos

A mi tutora Margarita, por su trabajo y apoyo.

A mis compañeros de carrera, con quienes aprendí el valor de trabajar en equipo.

A mi familia y amigos, en especial a mis padres por apoyarme en todo momento.

Gracias a todos

Resumen

El objetivo de este proyecto consiste en desarrollar una aplicación para sistemas Android que sirva de apoyo para gestionar partidos de balonmano. Destinada a entrenadores y sus ayudantes, de todas las categorías superiores a infantil.

En la actualidad no existe una solución integral para que los entrenadores de balonmano consigan llevar las tácticas de su equipo a un nuevo nivel. Se busca, mediante la digitalización y uso de repeticiones dar más facilidades a los entrenadores para que puedan tener a su vez más tiempo durante el partido.

El proyecto se ha desarrollado empleando el motor de videojuegos Unity y el leguaje C#, aplicando la metodología RUP.

Abstract

The purpose of this project is to develop a software for Android devices which helps to prepare handaball games. Intended for coaches and their assistants, of all categories above infant.

Currently there is no comprehensive solution for handball coaches to take their team's tactics to a brand new level. It is sought, through digitization and the use of repetitions, to give more facilities to the coaches so that they can have more time during the game.

The project has been developed using Unity, a videogame engine, and C# as programming language, following RUP metodology.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Como introducción a este Trabajo Fin de Grado, en adelante TFG, abordaré ciertos aspectos de relevancia para entender el mismo, como pueden ser los objetivos, las alternativas actuales al problema abordado o la estructura empleada.

1.1. Contexto

En la actualidad los entrenadores de balonmano no disponen de una solución software integral, completa, que les permita llevar las tácticas de juego al siguiente nivel. Debido a esto, una gran mayoría no emplea la informática, esto implica desventajas como: volatilidad y falta de repeticiones, algo clave para el aprendizaje. Aunque existan algunas aproximaciones, no resultan ser suficientes.

1.2. Motivación

La motivación para elegir este tema fue por tanto implementar un software capaz de ayudar en estas tareas a los entrenadores, simplificando su trabajo durante el partido, permitiéndoles así seguirlo de cerca, y gestionarlo de manera más eficiente. Partiendo de la idea de la tutora, se establecen una serie de requisitos mínimos a lograr, los cuales se van complementado al analizar las diversas aplicaciones actuales, ampliando el alcance del proyecto.

Todo ello en el marco de aprendizaje de una tecnología actual de desarrollo (Unity), lo que es útil en mi carrera profesional, ya que la programación de videojuegos es una rama que tenía interés en explorar, y esta ocasión era una buena oportunidad.

1.3. Objetivos

El objetivo principal es desarrollar una App de gestión de partidos de balonmano, la cual permita diseñar y mostrar jugadas dinámicas (en las cuales los jugadores y el balón se desplazan por el terreno de juego), así como dibujar en una pizarra convencional. Se valorarán mejoras en la misma a medida que se consiguen los objetivos, siguiendo en todo momento la metodología RUP.

Las posibles tecnologías eran Kotlin, Ionic y Unity, decantándome por esta última. El sistema objetivo es Android, pero se procurará dar servicio a los sistemas MacOS y Windows, ya que Unity lo permite (no habiendo mayor problema que asegurarse de que todo funciona en los diversos sistemas).

1.3.1. Objetivos personales

- **Gestión de proyectos.** Aprender de primera mano a gestionar un proyecto completo, desde cero, con metodologías ágiles.
- **Proyecto software al completo.** Participar personalmente en todas las etapas de un proyecto software: especificación de requisitos, creación de diagramas UML, diseño de la interfaz de usuario y su implementación, pruebas y creación de memoria con manuales. Todo ello aplicando las teorías y principios de la ingeniería de software, garantizando la calidad y mantenibilidad del código, sin dejar de lado la eficiencia.
- **Identificar y analizar necesidades y problemas.** Ser capaz de valorar las necesidades de un proyecto, especificando los requisitos necesarios teniendo en cuenta las diversas limitaciones de tiempo y coste. A su vez, ser capaz de encontrar soluciones adecuadas a la teoría y sobreponerse a las adversidades no contempladas en el análisis de riesgos, así como los problemas de implementación.
- **Conocimiento técnico.** Formarme en C# y Unity [50], un potente motor de videojuegos, que tiene soporte de compilación en diferentes plataformas (Android, WebGL [43], iOS, Windows, las principales consolas del momento, y muchos otros).

1.4. Estudio de aplicaciones en el mercado

Inicialmente, se estudió el Reglamento de partidos y competiciones [38]. A continuación se exploran las tecnologías que permitan realizar los objetivos.

Posteriormente se analizan las aplicaciones presentes en el mercado, por lo que se realiza una búsqueda en pro de detectar fortalezas y debilidades en el trabajo que otros ya han realizado, tratando de evitar así los mismos errores. A su vez, algunas de las aplicaciones presentan funcionalidades no tenidas en cuenta, que se podrían implementar en caso de tener tiempo.

1.4.1. TacticalPad Futsal & Handball [40].

Esta aplicación Android (figura 1.1), permite usarse para balonmano y fútbol sala. Presenta una versión de pago que implementa funcionalidades muy necesarias.

Puntos a favor:

- Permite la creación de jugadas estáticas, con jugadores de ambos equipos dispuestos por el terreno de juego, diferentes líneas para indicar acciones, y elementos de juego como balones, o utensilios de entrenamiento como conos, aros o banderines.
- Permite personalizar los equipos, con un nombre, color e imagen. También se puede añadir el nombre y posición de cada jugador (algo útil, pero que sin mayores funcionalidades, no sirve de mucho).

Puntos en contra:

- Al no tener ninguna explicación, el aprendizaje recae en el usuario, y no es demasiado intuitivo.
- Sólo permite el uso de jugadas dinámicas si te suscribes a la app (26€/ año).
- La traducción a español es automática, no siendo correcta en algunos puntos, pudiendo confundir al usuario.

Figura 1.1: TacticalPad Futsal & Handball

1.4.2. Field Hockey Dood [32].

También para dispositivos Android, es una app (figura 1.2), orientada a hockey hierba, pero se puede adaptar a las necesidades del balonmano, por el "parecido" del campo (respecto a las otras opciones del desarrollador: fútbol, baloncesto, vóleibol, etc.), es una app similar a la idea involucrada en este TFG.

Puntos a favor:

- Permite situar jugadores, balones, texto, líneas y conos en el campo y distribuirlos en éste.
- Se pueden crear jugadas añadiendo instantes de tiempo (situación actual de la pantalla), estas transiciones tienen un tiempo de paso y pueden visualizarse mientras se componen las jugadas.
- Permite guardar formaciones, jugadas y vídeos de las mismas (en el dispositivo).

Puntos en contra:

- Solicita permiso para grabar audio cuando éste no se emplea en ningún momento, por lo que puede incurrir en falta de privacidad si se usa sin consentimiento.
- Al encontrarse en fase experimental, la grabación de jugadas falla a veces.

- No tiene ninguna forma de tutorial, por lo que pese a ser fácil de usar, los usuarios pueden experimentar problemas, como puede apreciarse en alguna de las reseñas en [32].
- La traducción a español no es nativa.

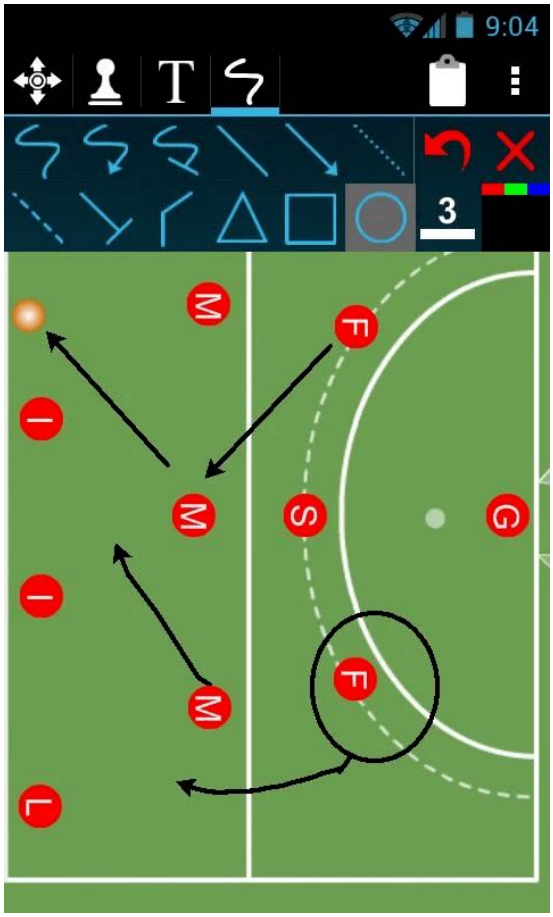

Figura 1.2: Field Hockey Dood

1.4.3. Pizarra Táctica: Balonmano [4].

En términos generales, es la aplicación Android (figura 1.3), que resulta más usable y parecida a la idea del TFG.

Puntos a favor:

- Su interfaz es muy usable y resulta intuitiva de manejar.
- Permite la creación de jugadas con jugadores de ambos equipos, flechas indicativas, elementos de entrenamiento, etc. tanto estáticas como dinámicas, pudiendo almacenarlas con un nombre y descripción.
- Las jugadas dinámicas se pueden recorrer de una manera muy simple con un slider, teniendo en todo momento posibilidad de repasar el flujo.
- Tiene una sección FAQ, donde se detalla parte del funcionamiento, así como un enlace a un vídeo en YouTube [3] que explica la creación de jugadas dinámicas.

Puntos en contra:

- Presenta publicidad mínimamente intrusiva, interrumpiendo de vez en cuando al usuario, sobre todo al regresar a la App.
- Ofrece una versión de pago (único, de 5,49€) que incluye la gestión de equipos, diversas flechas, y la opción de compartir jugadas. Sin estas opciones, la aplicación pierde bastante potencial (sobre todo, por las flechas).

Figura 1.3: Pizarra Táctica: Balonmano

1.4.4. Basketball Playbook Designer [10].

Esta aplicación web (figura 1.4), de gestión de baloncesto (y otros: fútbol, fútbol americano y lacrosse) está orientada al manejo de equipos, formaciones, y creación de jugadas.

Puntos a favor:

- Es muy completa, añadiendo frente a las otras alternativas una gestión mínima de los equipos, sin estadísitcas. (No contemplado inicialmente, sería algo que aportaría mucho a la App, en caso de tener tiempo).
- Su organización permite una gran usabilidad.

Puntos en contra:

- La gestión del equipo/ jugadores queda un poco pobre al no contabilizar estadísiticas de los partidos.
- Los jugadores solo se pueden añadir al equipo por invitación a un email, lo cual limita las pruebas y complica las incorporaciones.
- Se solicita un registro por parte del usuario, incluso para la versión de prueba.
- Tras la versión de prueba se solicita el pago de una suscripción mensual empezando desde los 7$ o anual empezando en los 70$.

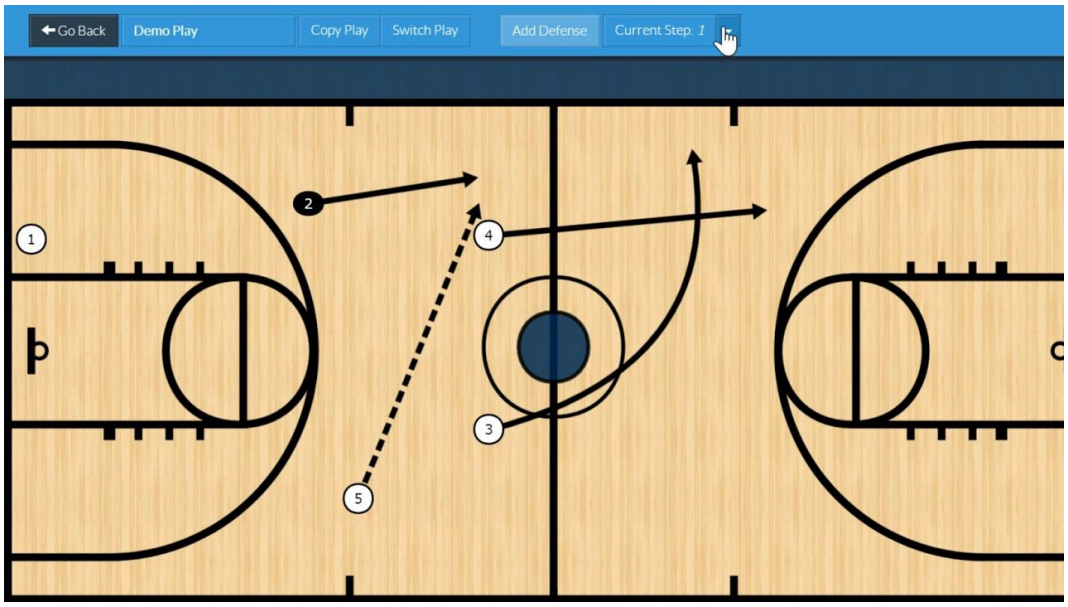

Figura 1.4: Basketball Playbook Designer

1.5. Consulta de TFGs similares

Como orientación procedo también a buscar algunos TFG que puedan considerarse similares, en la web UVaDoc de la Universidad de Valladolid. Realmente no encontré ninguno muy similar en concepto, pero sí alguno interesante teniendo en cuenta la tecnología, como son:

- **Creación de una plataforma para la gamificación de una asignatura.** [Enlace]. Desarrollo de un juego empleando Unity, con diferentes pantallas y niveles, de los que se almacenan récords de los usuarios.
- **Desarrollo de un juego 2D para Android usando Unity.** [Enlace]. Este juego Unity parte de una base similar, introducirse en la programación de videojuegos para un posible desarrollo profesional. No abarcando el 3D, como es la idea de éste TFG.
- **Estudio y aplicación de un juego en red multijugador para dispositivos móviles.** [Enlace]. En comparación a los anteriores, resultó de gran interés, pese a no estar relacionado, comprobar cómo se enfoca un juego al multijugador en red.

1.6. Estructura de la memoria

La memoria de este trabajo sigue la "Guía del alumno del TFG (Estructura, Formato, Documentación)" [21] expuesta en la página web del centro docente. A continuación, a modo de resumen, el contenido de la estructura es el siguiente:

- En la sección 2 , Planificación, se detalla la planificación de desarrollo del proyecto, la metodología elegida, y el análisis de riesgos y costes, así como el detalle de horas empleadas.
- En la sección 3, Análisis, se engloba la descripción de actores, la especificación de requisitos y la ley de protección de datos. Así como todo lo relacionado con casos de uso y el modelo de dominio.
- En la sección 4, Diseño, donde aparecen la arquitectura del sistema, patrones empleados, el diseño de la interfaz de usuario y diagramas de diseño: de secuencia entre otros.
- En la sección 5, Implementación, donde se destacarán las herramientas empleadas así como detalles sobre la protección de datos.
- En la sección 6, Pruebas, donde se especificarán las pruebas de caja negra llevadas a cabo y la evaluación de usabilidad de la interfaz por parte de usuarios reales.
- En la sección 7, Conclusiones, se expondrán las deducciones derivadas del TFG así como las posibles funcionalidades que podrían añadirse en un futuro.
- En el Apéndice I: Manual de instalación, se detalla todo lo necesario para la instalación del entorno de desarrollo y en dispositivos compatibles.
- Finalmente, el Apéndice II: Manual de uso, donde se expone paso a paso, cada una de las funcionalidades de la aplicación.

Capítulo 2

Planificación

En este capítulo se analizará la metodología a seguir para el desarrollo del TFG, así como los riesgos y costes que supone. Con todo, se realizará una planificación inicial que actuará como guión a seguir durante el resto del trabajo. También se tomará un registro de las horas empleadas, con la finalidad de detectar desvíos en la planificación y aprender de mis errores.

2.1. Plan de desarrollo

Este TFG, tiene asignada una carga de 12 créditos ECTS (European Credit Transfer and Accumulation System), según el plan de estudios actual [54]. Con una correspondencia horaria de 25 horas/ crédito, se obtienen 300 horas de trabajo individual. De estas 300, y según [54] 200 corresponden al trabajo autónomo por parte del estudiante, 75 a la elaboración de la memoria del trabajo, 20 a tutorías, y 5 para evaluación y defensa del mismo.

Tomando esto en consideración, la memoria ha sido redactada de manera continuada desde el inicio del trabajo, con una mayor carga en esta tarea al inicio, por la adecuación del formato, y poder así trabajar tomando nota de las referencias empleadas. De la misma manera, se han realizado las tutorías de manera regular, durante todo el proceso, resolviendo dudas y guiándome (por medio de videollamadas y correos electrónicos, adaptado así a la nueva situación por la pandemia).

Como **metodología** de trabajo se aplica RUP (Proceso Racional unificado), ha sido el más empleado en software, y se debe a sus principales fortalezas:

- **Adaptación al cliente.** Se adapta el proceso de manera que permita interactuar con el cliente. El alcance del proyecto, los requisitos que cumplir, así como otras regulaciones o procesos de negocio se establecerán de manera estrecha con el cliente (la tutora, en este caso).
- **Valor iterativo.** Se producen versiones a lo largo de la vida del proyecto, que permiten al cliente comprobar el avance del mismo Además permite detectar si la dirección es acorde con lo establecido en el primer punto, o si es necesario modificar ciertos aspectos.
- **Abstracción lógica.** Se defiende el empleo de elementos software reutilizables y constatados como son los patrones de diseño, modelos UML o frameworks. De esta manera los miembros del equipo disponen de una idea completa del proyecto que los ayuda a implementar de acorde a los objetivos.

- **Asegurar la calidad.** Los controles de calidad son realizados en todas las etapas del proyecto y no solo al final, evitando así desavenencias futuras. En este TFG se cuenta con las revisiones de la tutora, en cada una de las etapas del proyecto.
- **Planificación de recursos.** Al analizar los requisitos, tener en cuenta las restricciones y ser conscientes de lo que implica garantizar la calidad, se puede planificar el proyecto para garantizar que no haya retrasos por requisitos contradictorios, o la disputa de recursos finitos de manera simultánea, no perdiendo tiempo.
- **Colaboración.** Aunque no sea el punto principal en este TFG, esta metodología facilita la comunicación entre los miembros del equipo, que podrán coordinar el análisis, diseño, implementación y pruebas, siendo así más eficientes.

La metodología RUP basa su ciclo de vida en las fases que dictamina el **proceso unificado de negocio** [1]. Se trata de un desarrollo iterativo e incremental, que consta de 4 fases:

- **Inicio.** El comienzo es la fase más corta de todas, en la cual se establece una visión preliminar del proyecto, una planificación de tareas, y el presupuesto asociado a todo el trabajo. También se analiza si el proyecto es factible, teniendo en cuenta el presupuesto y la dificultad técnica (pudiendo comparar con aplicaciones similares existentes, así como con el consejo de la tutora). También es el momento en el que se decide entre comprar o desarrollar el software; en este caso, evidentemente, se desarrolla (empleando las bibliotecas que ya hayan sido implementadas y sean de utilidad).
- **Elaboración o Análisis.** Durante esta etapa, se identifican la mayoría de los requisitos del sistema. Y también se toman en cuenta los riesgos potenciales que podrían afectar a priori al proyecto.

  A su vez es común realizar los diagramas de casos de uso[1] y de clases[2]. Por último se establece la arquitectura del sistema[3] lo que permite planificar la siguiente fase, teniendo en cuenta siempre los riesgos e información recopilada durante estas dos fases.

  Es decir, esta fase se compone, esencialmente, del análisis y diseño de la aplicación.
- **Construcción.** Siendo la más extensa del proceso, es donde se implementan las funcionalidades propuestas en la fase de elaboración. Al realizarse por etapas, cada una de ellas debe ser completamente funcional (para los objetivos establecidos), por lo que cada etapa implementa nuevas funcionalidades y/o corrije fallos detectados, tanto de funcionalidad como de usabilidad. Para ello se suelen escribir tests [57] completos, evitando así errores desde fases tempranas. Debido a que la tecnología de desarrollo empleada es Unity, que permite la exportación a muy diversos sistemas operativos, se probará cada evolución en todos los sistemas soporte: Android, macOS y Windows.
- **Transición y pruebas.** En esta última fase, se trata de encontrar errores en las funcionalidades, pudiendo para ello mostrar el sistema a usuarios, obteniendo retroalimentación para mejorar el resultado final, por lo que es ventajoso realizarlo desde fases tempranas del desarrollo. También se incluye en este momento enseñar a los usuarios a utilizar el sistema correctamente.

Este proceso, ha sido aplicado por tanto, añadiendo requisitos en el capítulo 3, Análisis, a medida que el sistema iba avanzando en funcionalidad. Estos avances se pueden apreciar de manera detallada en la sección 2.6, Seguimiento, donde se comentan algunos sucesos de interés.

[1] Descripción de una actividad que puede realizar el usuario en el sistema.
[2] Diagrama UML que en diseño orientado al objeto, permite modelar relaciones entre las entidades implementadas.
[3] Conjunto de patrones, objetivos y restricciones que establece la estructuración del software.

A su vez, se presenta (figura 2.1), un diagrama de Gantt [55] [56] (una herramienta clave en la gestión de proyectos, que presenta una visión general de las tareas que componen el mismo y que permite a los miembros del equipo ser conscientes de plazos de entrega y estado actual de un proyecto).En dicha figura, se muestra el proceso del proyecto, detallando sólo las fases. Más adelante, en el subcapítulo 2.5, Planificación inicial, se presentarán 4 diagramas de Gantt, uno por fase, más detallados.

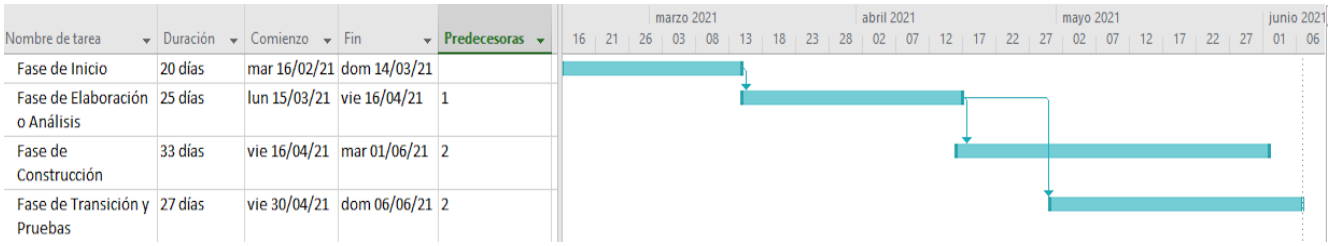

Figura 2.1: Diagrama de Gantt de las fases del proyecto.

2.2. Restricciones

Se aplican una serie de restricciones al proyecto, que acotan su alcance.

- **Requisitos de distribución.** La aplicación será soportada obligatoriamente en el sistema operativo Android, sin ningún fallo. A su vez, se intentará que ningún bug imposibilite su uso en las demás plataformas a las que se quiere dar acceso de manera secundaria: macOS y Windows.
- **Contemplación de riesgos.** Se realizará de manera continuada el análisis y seguimiento de riesgos, apartado 2.3.
- **Almacenamiento de datos.** Los datos de la aplicación serán almacenados en el propio dispositivo, evitando así la necesidad de una conexión a internet, y acelerando la interfaz de usuario.
- **Trabajo.** El proyecto no deberá exceder de las 300 horas hombre, respetando así su asignación de 12 ECTS.
- **Fecha de entrega.** La fecha temprana de entrega será, como mínimo, el 31 de mayo de 2021, día en el que termino la última asignatura del grado (las prácticas en empresa). Contando unos días a mayores para su evaluación.
- **Tiempo.** El proyecto debe finalizar en el mes de junio de 2021, teniendo tiempo suficiente para su defensa en la convocatoria ordinaria de TFG.

2.3. Análisis de riesgos

Según se puede inferir de [20] el riesgo es un evento, o serie de eventos, que pueden o no ocurrir. En caso de que ocurran, tendrá un efecto en los objetivos del proyecto, ya sea positivo (una oportunidad) o negativo (una amenaza). Es por tanto la posibilidad de exponerse a consecuencias adversas, o eventos futuros adversos. Los riesgos no son, por tanto, problemas actuales. En esta sección trataré los riesgos principales que pueden acontecer durante el desarrollo de este trabajo. Como pude aprender durante la asignatura Planificación y Gestión de Proyectos [55] es una tarea imprescindible, puesto que el análisis de riesgos permite tener en cuenta aspectos externos o implícitos al trabajo, que pueden condicionar el éxito o fracaso de cualquier proyecto, y condicionar en gran medida a la planificación del mismo. Tras la tarea de identificación y análisis de éstos, no debemos dejarlos olvidados, hay que

monitorizarlos teniendo en cuenta cómo avanzan, y si aparecen nuevos, o los existentes no han sido priorizados correctamente. Es, por tanto, una tarea iterativa durante el trabajo, por lo que se deben revisar y actualizar si fuera necesario.

Los riesgos y su análisis brindan conocimiento acerca de las tareas que se realizan, y para no sufrirlos, es de vital importancia realizar una lista de errores con la cual tenerlos localizados; para esto, fue realmente útil seguir la lista [9].

Existen diferentes tipos de riesgos, que son:

- **Riesgos conocidos.** Fruto de evaluar el proyecto y su alcance.
- **Riesgos predecibles.** Aquellos que la experiencia dicta que pueden volver a ocurrir.
- **Riesgos impredecibles.** Circunstancias como la pandemia, imposibles de predecir.

Por tanto, nos debemos centrar en los riesgos reconocibles, teniendo en cuenta el impacto que supondrán al proyecto. Los riesgos deben evaluarse durante todas las etapas de un proyecto: identificación, análisis, control y monitorización.

Los riesgos son tenidos en cuenta pero no siempre se pueden evitar, también se puede proteger del mismo procurando que afecte en menor medida en caso de producirse, o transferir el riesgo para que otros lo asuman. En cualquier caso, también puede llegar a aceptarse un riesgo, ya que si el esfuerzo y los costes implicados en evitarlo, superan los que generaría en caso de que se dé, no tendría sentido, y se acepta que pueda ocurrir.

A continuación, expongo los riesgos previstos en este trabajo. Con un identificador para referirse a ellos si fuera necesario, una descripción del mismo, la probabilidad de que éste ocurra y su impacto en caso de que se dé. Por otra parte, y derivado de la probabilidad-impacto, se toman en cuenta una serie de acciones para mitigar (tratar de reducir la probabilidad de ocurrencia del riesgo), así como de contingencia (reducir, en caso de ocurrencia, los efectos en la planificación).

ID	Categoría	Descripción del riesgo	Probabilidad	Impacto	Acciones de Mitigación	Acciones de Contingencia
R01	Planificación y control	Empeoramiento de la pandemia mundial por COVID-19.	Medio	Medio	No procede.	Adecuarse a una presentación telemática caso de que no se permita de forma presencial.
R02	Planificación y control	Inexperiencia en la gestión de proyectos.	Alta	Bajo	1.- Mantener una buena comunicación con la tutora. 2.- Actualizar la lista de riesgos, y tenerla siempre presente.	No procede.

ID	Categoría	Descripción del riesgo	Probabilidad	Impacto	Acciones de Mitigación	Acciones de Contingencia
R03	Planificación y control	Estimación inadecuada de los recursos necesarios.	Media	Medio	Tratar de aprovechar el tiempo al máximo y empezar con las actividades lo antes posible para mantener alta la flotabilidad[4].	Replanificar e informar a la tutora.
R04	Planificación y control	Inexperiencia a la hora de abordar una tarea novedosa.	Media	Medio	Dedicar parte del tiempo asignado a investigar el aspecto desconocido, aprovechando la flotabilidad.	No procede.
R05	Requisitos	Cambio en los requisitos.	Media	Alto	Aplicar la metodología ágil en el desarrollo software para reducir el impacto que supondrían los cambios.	En la reunión semanal, revisar el trabajo realizado y planificar una depuración del software, así como la implementación de los cambios.
R06	Requisitos	Los requisitos establecidos no son correctos.	Baja	Medio	En las reuniones con la tutora, confirmar que el desarrollo es acorde a los requisitos.	Revisión de requisitos e inclusión de cambios en la planificación.
R07	Personal	Ampliación del horario de trabajo/ Cambio de trabajo.	Baja	Medio	Llevar al día el TFG y el trabajo.	Reestructurar el horario y la planificación para tener en cuenta la nueva situación y adaptarse al cambio lo antes posible.
R08	Personal	Contracción de la COVID-19.	Media	Alto	1.- Seguir a rajatabla las recomendaciones sanitarias de las autoridades en cada momento. 2.- No realizar acciones que favorezcan el contagio.	1.- Seguir las recomendaciones sanitarias, manteniendo el ritmo de trabajo que sea posible. 2.- Al recuperarse y retomar el ritmo habitual, replanificar en función del tiempo perdido.

[4]La cantidad de tiempo que puede ser retrasada una tarea sin que cause un retraso al proyecto.

ID	Categoría	Descripción del riesgo	Probabilidad	Impacto	Acciones de Mitigación	Acciones de Contingencia
R09	Personal	Contracción de otras enfermedades.	Media	Medio	1.- Evitar exposiciones a grupos de contagio y las actividades de riesgo.	1.- Seguir las recomendaciones sanitarias, manteniendo el ritmo de trabajo que sea posible. 2.- Al recuperarse y retomar el ritmo habitual, replanificar en función del tiempo perdido.
R10	Equipo	Falta de disponibilidad de los miembros del equipo de trabajo.	Baja	Medio	No procede.	Mayor trabajo y organización autónoma.
R11	Técnico	Avería del ordenador.	Baja	Medio	Realizar un mantenimiento y uso del equipo adecuado.	1.- Si fuera necesario, adquirir un equipo nuevo. 2.- Instalación de los programas de desarrollo necesarios. 3.- Reestructurar la planificación en base al tiempo discurrido entre la avería y la puesta en marcha.
R12	Técnico	Cortes del suministro eléctrico o de internet.	Baja	Bajo	Realizar copias de seguridad accesibles.	Realizar tareas que no requieran consumo eléctrico/internet durante la caída del servicio.
R13	Técnico	Caída del sistema de almacenamiento externo.	Baja	Medio	1.- Tener copias locales actualizadas del trabajo en progreso, mediante GitLab. 2.- Tener un sistema alternativo de desarrollo que pueda sustituir al sistema caído con una copia de seguridad periódica.	Trabajar en local con el sistema alternativo hasta que se restablezca el almacenamiento externo.

ID	Categoría	Descripción del riesgo	Probabilidad	Impacto	Acciones de Mitigación	Acciones de Contingencia
R14	Complejidad	Complejidad del proyecto.	Baja	Medio	Dividir las tareas complejas en un mayor número de subtareas simples y repartirlas en múltiples iteraciones.	No procede.
R15	Complejidad tecnológica	Uso de una tecnología con la que no se ha trabajado anteriormente.	Media	Medio	Tener presentes tecnologías que puedan usarse en ámbitos similares.	No procede.

Tabla 2.1: Desglose de riesgos.

2.3.1. Observaciones sobre riesgos acontecidos

Desde el principio se manifiesta el riesgo R14 (Nivel alto de complejidad técnica), con la herramienta LaTeXlo cual retrasa el inicio del proyecto. Se siguieron tutoriales y se atajaba cada problema según aparecían a la hora de redactar la memoria.

Por otra parte también se manifestó el riesgo R15 (Uso de una tecnología con la que no se ha trabajado anteriormente), como es el caso de Unity, incrementando el tiempo de formación por tutoriales en pocas horas.

2.4. Análisis de costes

El enfoque para calcular el presupuesto del TFG ha sido el Real Decreto-ley 28/2020, la ley de teletrabajo española [6], en el artículo 12, establece que la empresa debe remunerar al trabajador por los gastos derivados del trabajo en casa, mediante el convenio colectivo del sector. Es decir, esta mecánica aporta bastante flexibilidad, (debiendo contemplar todos los recursos empleados por el trabajador).

En base a la realidad, emplearé datos de mi caso particular. Los gastos serán imputables, referidos a mi domicilio actual, y tomados según la media geométrica de los últimos 12 meses. Todos los costes han sido calculados con la totalidad de los decimales, lo que explicaría posibles pequeñas variaciones en el cómputo global. A continuación, aparece un desglose de los mismos:

- **Vivienda.** Además del precio del alquiler, se incluyen los gastos de comunidad (donde se incluye el agua corriente), el seguro del hogar, y el pago del IBI. Dando un valor acorde al mercado y ubicación, teniendo en cuenta el valor actual según [15] el coste es de 540€/ mes, según las horas de uso respecto al día completo, el coste por hora sería de 0,75€. Y en las 300 horas de duración, un total de 225€.
- **Consumo eléctrico.** La factura de electricidad, incluyendo los impuestos, tasas, KWh, ... asciende a 55€/ mes tomando por referencia los 12 meses del año 2020 y realizando su media. Se obtiene un coste de 0,075€/ hora. Lo que tras 300 horas, hace un total de 22,5€.

- **Factura de calefacción.** El gasto medio de las facturas de calefacción de los 12 meses del año 2020 es de 61€. Este cálculo en horas resulta en 0,084€/ hora, y finalmente en 25,41€en total.
- **Proveedor de internet.** La tarifa mensual de internet es de 48€/mes. Teniendo en cuenta que se usa 300 horas, el coste es de 19,98€(0,022€/ hora).
- **Amortización del equipo informático.** El equipo utilizado es un MacBook Pro 13"de 2015. Con un precio de compra en 2015 de 1650€, y un valor actual de mercado que ronda los 650€, tras 5 años, el gasto anual de amortización es de (1650€- 650€) / 5 años = 200€/año. Tras una pequeña conversión, equivale a 0,023€/ hora, que, al utilizarlo durante 300 horas, se obtiene un total de 6,85€.
- **Horas de trabajo.** Según [5] (página 28) el salario bruto de un analista programador es de 1.438,08€/mes[5] (cifra que incluye los impuestos correspondientes al trabajador: IRPF y SS, calculadas en [36] por un valor de 275,68€). Teniendo en cuenta que la jornada en dicho caso sería de 40 horas semanales (160 mensuales), obtenemos un salario de 8,99€/ hora. Al ser 300 horas de trabajo, el total asciende a 2.696,4€.

Tras el detalle en los cálculos, se presentan en la siguiente tabla (2.2), para facilitar al lector su interpretación.

Gasto	Coste / hora	Coste total (300h)
Vivienda	0,75€	225€
Consumo eléctrico	0,075€	22,5€
Calefacción	0,084€	25,41€
Internet	0,065€	19,98€
Amortización	0,023€	6,85€
Horas de trabajo	8,99€	2.696,4€
Total	**9,98€**	**2.996,14€**

Tabla 2.2: Presupuesto del proyecto

2.4.1. Costes finales

Debido a los diversos problemas comentado en el análisis de riesgos acontecidos, se emplean 36 horas más de lo establecido. Esto redunda en los costes finales del proyecto. Tomando el coste por hora de la tabla 2.2, es decir 9,98€/ hora, el coste final asciende a los **3.355,63€** , es decir, **359,53€** más que lo inicialmente presupuestado.

2.5. Planificación inicial

Teniendo en cuenta el calendario de depósito y defensa del TFG de este año [22], se establece como fecha de fin de proyecto el 6 de junio de 2021. Comenzando el 16 de febrero de 2021, da como resultado 16 semanas para la elaboración del proyecto. Esta fecha de fin ha sido estimada en base a varios motivos:

- **Prácticas en Empresa.** Tomando en consideración la gran posibilidad de continuar empleado tras la finalización del contrato de prácticas, lo idóneo sería poder incorporarse cuanto antes, una vez terminado este TFG.
- **Alcance del proyecto.** Debido a los requisitos iniciales del proyecto, sección 3.2, se estima que no habrá problemas en implementar la funcionalidad requerida, y otras a mayores, por lo que 300 horas serán suficientes.

[5]Cifras del último convenio, en 2018.

Como se puede apreciar en la tabla 2.3, las dos primeras semanas son mínimamente reducidas, y las demás constan de 20 horas semanales. Se ha optado por una cantidad de horas semanales siguiendo el consejo de mi tutora, ya que permite una mayor flexibilidad horaria en el día a día, tratando de evitar que algunos imprevistos no interfieran en gran medida.
Así como la última semana, destinada a la revisión de la memoria por completo, y la presentación para la defensa.

Finalmente, se presenta en la tabla 2.4 la distribución inicial de las 300 horas, siguiendo la guía [54].

Nº	Inicio	Final	Horas	Observaciones/ Incidencias
1	16/02/2021	23/02/2021	14	
2	23/02/2021	02/03/2021	18	
3	02/03/2021	09/03/2021	20	
4	09/03/2021	16/03/2021	20	
5	16/03/2021	23/03/2021	20	
6	23/03/2021	30/03/2021	20	
7	30/03/2021	06/04/2021	20	
8	06/04/2021	13/04/2021	20	
9	13/04/2021	20/04/2021	20	
10	20/04/2021	27/04/2021	20	
11	27/04/2021	04/05/2021	20	
12	04/05/2021	11/05/2021	20	
13	11/05/2021	18/05/2021	12	Riesgo R10.
14	18/05/2021	25/05/2021	20	
15	25/05/2021	01/06/2021	20	
16	01/06/2021	06/06/2021	16	Revisión final y preparación de defensa.
Fin	**06/06/2021**	**Total**	**300**	**El cómputo total es de 300 horas.**

Tabla 2.3: Planificación inicial.

Sección	Horas	Porcentaje
Investigación, documentación e implementación.	200	66,7 %
Redacción de la memoria y adecuación a la plantilla.	75	25 %
Tutorías	20	6,6 %
Evaluación y defensa	5	1,6 %
Total	**300**	**100 %**

Tabla 2.4: Distribución inicial por secciones [54].

2.5.1. Plan de trabajo

A continuación, se detallan las tareas a abordar según cada fase comentada en el apartado 2.1. De igual manera se presenta en cada fase un diagrama de Gantt (figuras 2.2, 2.3, 2.4 y 2.5), más detallado que el expuesto en el plan de desarrollo. Éstos se han rediseñado, puesto que los diagramas generados con MS Project, no permitían visualizar de una manera óptima las tareas.

Fase de Inicio

En las tablas sucesivas a la tabla 2.5 se describen en detalle las tareas de esta fase. Se realizará la plantilla de la memoria así como toda la tarea de análisis de planificación.

Se estima como fecha de comienzo el 16/02/2021 y se espera finalizar el 14/03/2021.

Nombre de la tarea	Duración
T-01 Toma de contacto inicial con el problema.	5 horas
T-02 Análisis de riesgos.	2,5 horas
T-03 Análisis de costes.	2,5 horas
T-04 Búsqueda de aplicaciones similares.	2 horas
T-05 Formación en Unity y C#.	20 horas
T-06 Preparación del entorno de trabajo.	3 horas
T-07 Plan de trabajo.	3 horas
T-08 Redacción de la memoria.	20 horas
T-09 Tutorías online y mails.	5 horas
Total fase de Inicio.	**63 horas**

Tabla 2.5: Plan de trabajo de la fase de Inicio.

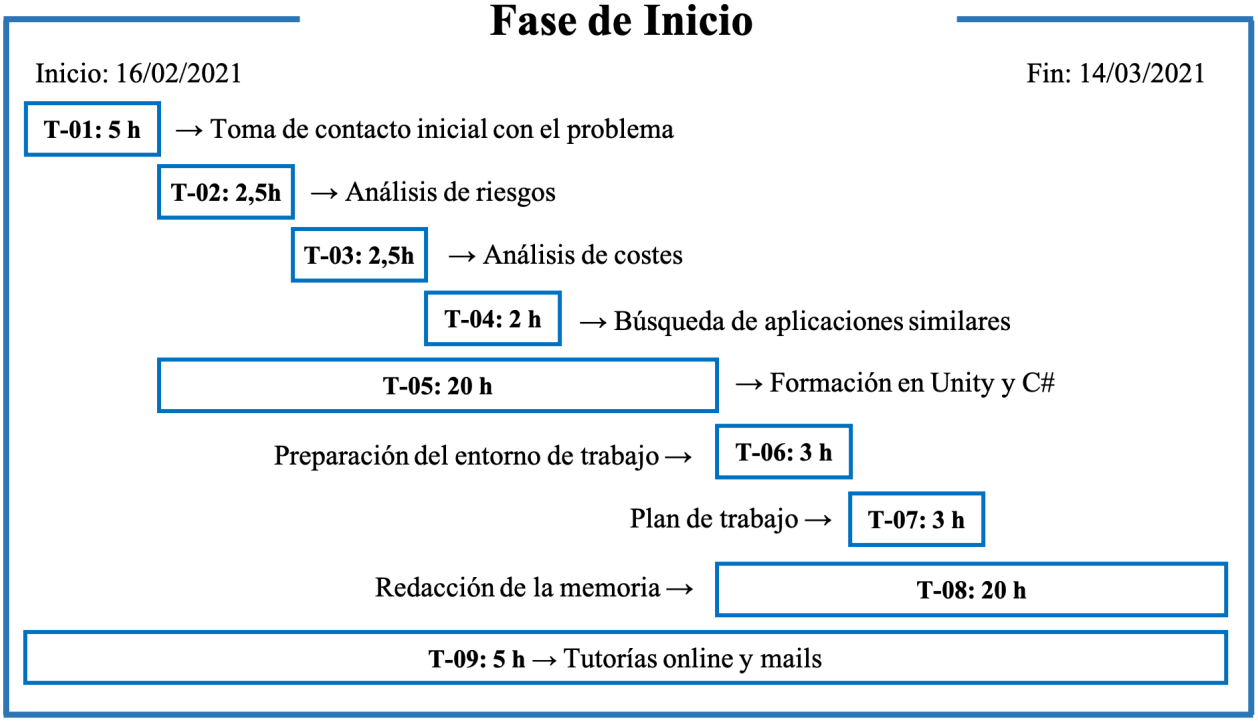

Figura 2.2: Diagrama de Gantt de la fase Inicio.

Fase de Elaboración o Análisis

En las tablas sucesivas a la tabla 2.15 se describen en detalle las tareas de esta fase. El análisis y diseño forman una parte clave del desarrollo software, pues es lo que dictamina cómo será la implementación, guiándola en todo aquello que pueda establecerse desde un inicio.

T-01	Toma de contacto inicial con el problema.
Descripción	Lograr una idea general de las funcionalidades que dispondrá la aplicación, complementando con las siguientes acciones. Búsqueda y lectura de los reglamentos de juego de balonmano. Visionado de 2 partidos de balonmano para detectar posibles funcionalidades.

Tabla 2.6: T-01: Toma de contacto inicial con el problema.

T-02	Análisis de riesgos.
Descripción	Detectar y analizar los posibles riesgos que incumben a este proyecto. Incluyendo la cuantificación de los mismos así como una descripción y el plan de contingencia en caso de que se produzca el riesgo.

Tabla 2.7: T-02: Análisis de riesgos.

T-03	Análisis de costes.
Descripción	Consiste en detectar los costes asociados al proyecto, calculando una estimación de los gastos que supondría un proyecto de estas características, en mi caso particular.

Tabla 2.8: T-03: Análisis de costes.

T-04	Búsqueda de aplicaciones similares.
Descripción	Realizar una búsqueda de aplicaciones existentes que atajen un paradigma similar. Se busca detectar fortalezas y debilidades en el trabajo previamente efectuado por otros, para intentar evitar fallos en el proyecto.

Tabla 2.9: T-04: Búsqueda de aplicaciones similares.

T-05	Formación en Unity y C#.
Descripción	Consiste en el visionado de vídeos, lecturas de manuales en páginas web y realización de ejemplos resueltos o tutoriales que ayuden a comprender la nueva tecnología en la que se implementará el proyecto.

Tabla 2.10: T-05: Formación en Unity y C#.

T-06	Preparación del entorno de trabajo.
Descripción	Instalación de Unity Hub, Unity y Visual Studio Code, así como todos los preparativos para desarrollar la aplicación. Preparación del repositorio remoto.

Tabla 2.11: T-06: Preparación del entorno de trabajo.

T-07	Plan de trabajo.
Descripción	Realizar un listado de tareas a desarrollar durante todo el proyecto (este apartado), incluyendo una estimación temporal para cada una de ellas, estableciendo el máximo en 300 horas de trabajo.

Tabla 2.12: T-07: Plan de trabajo.

Se estima como fecha de comienzo el 15/03/2021 y se espera finalizar el 15/04/2021.

T-08	**Redacción de la memoria.**
Descripción	Inicialmente, se elaborará la plantilla acorde a los requerimientos de este TFG con la herramienta LaTeX. Posteriormente, se plasmarán los conocimientos que conforman el proyecto, véanse diagramas, requisitos, manuales,... Se debe recoger todo aquello que se realiza en una fase. Abarca todo el proyecto.

Tabla 2.13: T-08: Redacción de la memoria.

T-09	**Tutorías online y mails.**
Descripción	Puesta en común de los avances del proyecto, así como la exploración de soluciones para los problemas encontrados. Abarca todo el proyecto.

Tabla 2.14: T-09: Tutorías online y mails.

Nombre de la tarea	**Duración**
T-10 Establecimiento de requisitos.	2 horas
T-11 Modelo de casos de uso.	5 horas
T-12 Especificación de casos de uso.	8 horas
T-13 Modelo de dominio.	2 horas
T-14 Diagramas de proceso/ flujo.	5 horas
T-15 Diseño de la Interfaz de Usuario.	12 horas
T-16 Investigación de arquitecturas en Unity.	3 horas
T-17 Arquitectura de la aplicación.	7 horas
T-08 Redacción de la memoria.	20 horas
T-09 Tutorías online y mails.	6 horas
Total fase de Elaboración o Análisis.	**70 horas**

Tabla 2.15: Plan de trabajo de la fase de Elaboración o Análisis.

T-10	**Establecimiento de requisitos.**
Descripción	Elaboración de los requisitos funcionales, no funcionales y funcionales de información, que dictaminan aquello de lo que debe ser capaz la aplicación.

Tabla 2.16: T-10: Establecimiento de requisitos.

T-11	**Modelo de casos de uso.**
Descripción	Consiste en determinar los casos de uso que tendrá la aplicación en función de los requisitos. Éstos han de plasmarse en el diagrama de casos de uso.

Tabla 2.17: T-11: Modelo de casos de uso.

T-12	**Especificación de casos de uso.**
Descripción	Descomposición de los casos de uso en detalle. Se expondrá cada uno de éstos, en una tabla que plasma la secuencia normal, así como las extensiones para otros casos particulares, o errores.

Tabla 2.18: T-12: Especificación de casos de uso.

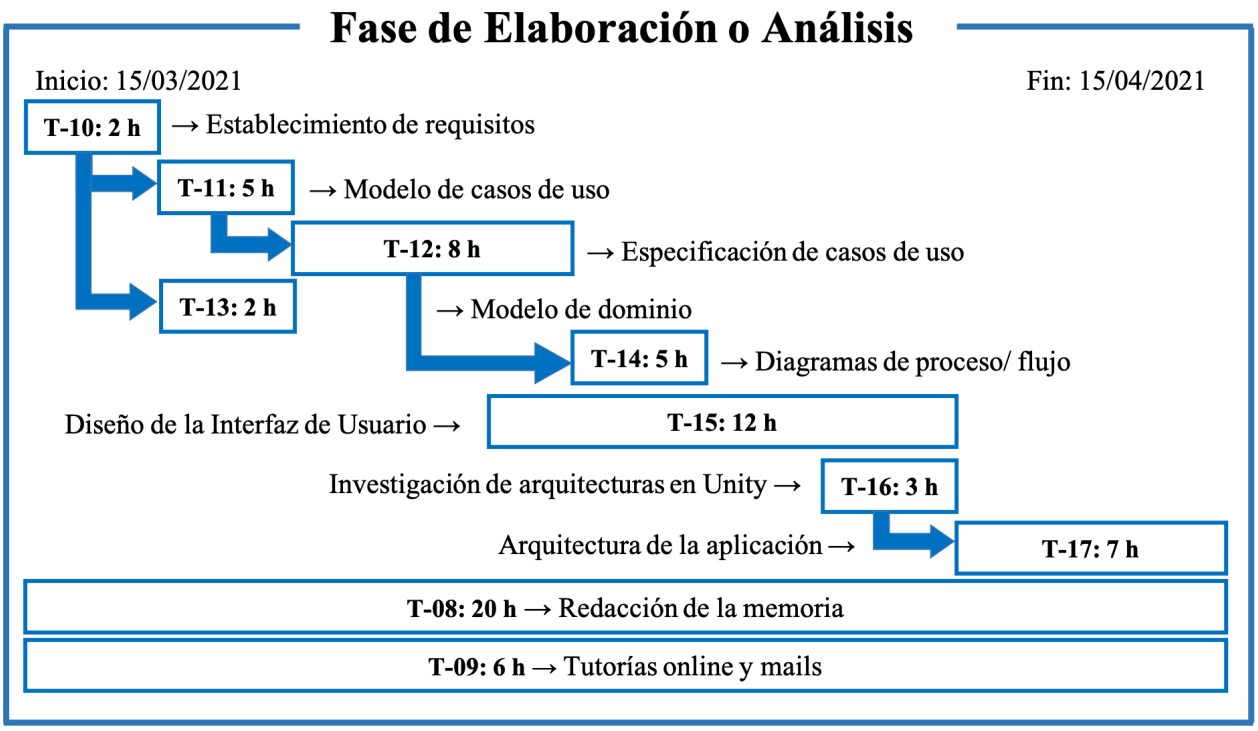

Figura 2.3: Diagrama de Gantt de la fase de Elaboración o Análisis.

T-13	**Modelo de dominio.**
Descripción	Elaboración del diagrama de clases del dominio, incluyendo sus atributos y relaciones entre clases.

Tabla 2.19: T-:13 Modelo de dominio.

T-14	**Diagramas de proceso/ flujo.**
Descripción	Especificación del flujo de los diferentes casos de uso, mediante diagramas de proceso detallados.

Tabla 2.20: T-14: Diagramas de proceso/ flujo.

T-15	**Diseño de la Interfaz de Usuario.**
Descripción	Como ayuda a la implementación de las vistas en la aplicación, se realizarán unos prototipos digitales que especificarán los detalles de las mismas.

Tabla 2.21: T-15: Diseño de la Interfaz de Usuario.

T-16	**Investigación de arquitecturas en Unity.**
Descripción	Recabar información acerca de las posibles arquitecturas aplicables en Unity, para elegir aquella que mejor se adapte al proyecto.

Tabla 2.22: T-16: Investigación de arquitecturas en Unity.

T-17	**Arquitectura de la aplicación.**
Descripción	Realizar los diagramas correspondientes a la arquitectura de la aplicación, planificando así la estructura del código.

Tabla 2.23: T-17: Arquitectura de la aplicación.

Fase de Construcción

En esta fase de construcción o implementación, se logra la funcionalidad establecida en análisis y conceptualizada en diseño. En las tablas sucesivas a la tabla 2.24 se describen en detalle las tareas de esta fase.

Se estima como fecha de comienzo el 16/04/2021 y se espera finalizar el 01/06/2021.

Nombre de la tarea	Duración
T-18 Implementación del menú principal.	5 horas
T-19 Implementación de la escena jugadas.	25 horas
T-20 Implementación de la escena listar jugadas.	10 horas
T-21 Implementación de la escena pizarra.	20 horas
T-22 Implementación de la escena plantilla.	15 horas
T-23 Implementación de la escena ajustes.	10 horas
T-24 Implementación del cambio de idiomas.	10 horas
T-08 Redacción de la memoria.	10 horas
T-09 Tutorías online y mails.	4 horas
Total fase de Construcción.	**109 horas**

Tabla 2.24: Plan de trabajo de la fase de Construcción.

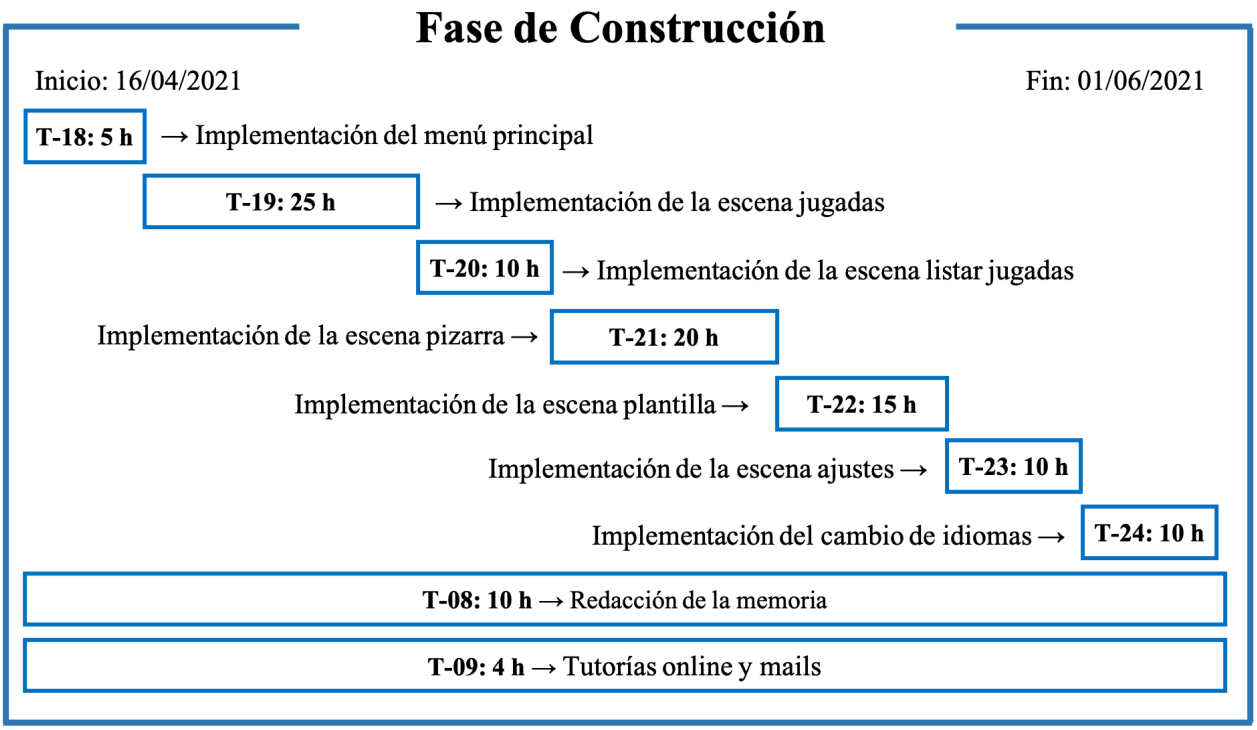

Figura 2.4: Diagrama de Gantt de la fase Construcción.

T-18	Implementación del menú principal.
Descripción	Elaboración de la funcionalidad requerida para acceder a las demás pantallas de la app.

Tabla 2.25: T-18: Implementación del menú principal.

T-19	Implementación de la escena jugadas.
Descripción	Confección de las funcionalidades asociadas a jugadas, como son su creación, guardado y carga de las mismas, así como poder reproducir una animación con sus pasos.

Tabla 2.26: T-19: Implementación de la escena jugadas.

T-20	Implementación de la escena listar jugadas.
Descripción	Realización de la escena que permite el listado de jugadas guardadas así como la eliminación parcial o completa de éstas.

Tabla 2.27: T-20: Implementación de la escena listar jugadas.

T-21	Implementación de la escena pizarra.
Descripción	Creación de una pizarra interactiva, que permita al usuario dibujar tácticas y borrar parcial o completamente su progreso, pudiendo guardar éste como una captura de pantalla.

Tabla 2.28: T-21: Implementación de la escena pizarra.

T-22	Implementación de la escena plantilla.
Descripción	Realización de la escena que permite listar a los jugadores presentes en el sistema, permitiendo además el alta, baja y modificación de éstos.

Tabla 2.29: T-22: Implementación de la escena plantilla.

T-23	Implementación de la escena ajustes.
Descripción	Implementar una escena que permita la modificación del color de la equipación de ambos equipos.

Tabla 2.30: T-23: Implementación de la escena ajustes.

T-24	Implementación del cambio de idiomas.
Descripción	Inclusión de un menú desplegable que modifique el idioma mostrado en la aplicación eligiendo entre: español, inglés, francés y alemán.

Tabla 2.31: T-24: Implementación del cambio de idiomas.

Fase de Transición y Pruebas

La fase de pruebas es muy importante y va de la mano con la fase de construcción, ya que debe probarse todo al terminar de implementar. De igual manera que es la última fase, pues es cuando se muestra la aplicación a más usuarios, encontrando así puntos de mejora que incorporar. Se incluye también la comprobación de la memoria y la preparación de la defensa.

En las tablas sucesivas a la tabla 2.32 se describen en detalle las tareas de esta fase.

Nombre de la tarea	Duración
T-25 Pruebas personales de funcionalidad.	20 horas
T-26 Pruebas finales de funcionalidad con usuarios.	5 horas
T-27 Elaboración del manual de instalación.	5 horas
T-28 Elaboración del manual de usuario.	3 horas
T-29 Presentación y preparación de la defensa.	5 horas
T-30 Revisión de la memoria.	15 horas
T-09 Tutorías online y mails.	5 horas
Total fase de Transición y Pruebas.	**58 horas**

Tabla 2.32: Plan de trabajo de la fase de Transición y Pruebas.

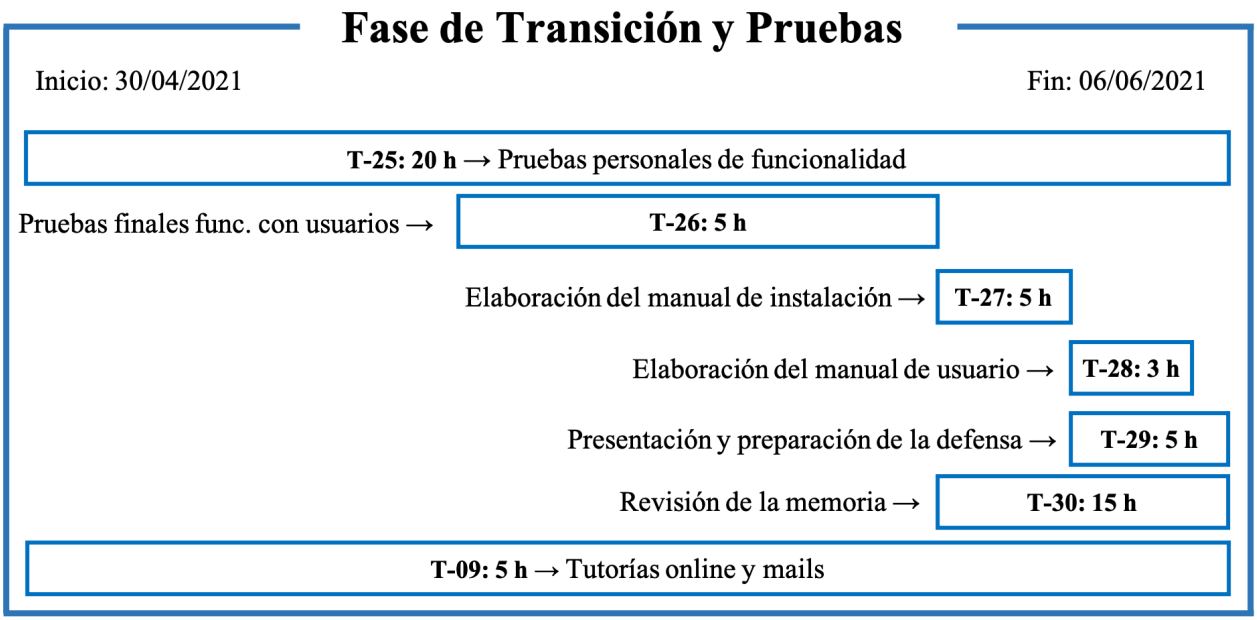

Figura 2.5: Diagrama de Gantt de la fase Transición y Pruebas.

T-25	Pruebas personales de funcionalidad.
Descripción	Comprobación individual de las funcionalidades del proyecto. En caso de encontrar fallos, se modificará la implementación hasta asegurar la cobertura del caso de uso.

Tabla 2.33: T-25: Pruebas personales de funcionalidad.

T-26	Pruebas finales de funcionalidad con usuarios.
Descripción	Pruebas de usabilidad con usuarios externos al proyecto.

Tabla 2.34: T-26: Pruebas finales de funcionalidad con usuarios.

T-27	Elaboración del manual de instalación.
Descripción	Redacción de una guía que permita a cualquier usuario conocer el método de instalación de la app en su dispositivo.

Tabla 2.35: T-27: Elaboración del manual de instalación.

La asignación de horas por fases queda, entonces, como se representa en la tabla 2.39.

T-28	**Elaboración del manual de usuario.**
Descripción	Confección de un manual del uso de la aplicación, que sirva como guía completa de las funcionalidades logradas a cualquier usuario, independientemente de su nivel.

Tabla 2.36: T-28: Elaboración del manual de usuario.

T-29	**Presentación y preparación de la defensa.**
Descripción	Elaboración de una presentación digital que exponga los puntos principales del proyecto, de cara a la defensa de este TFG.

Tabla 2.37: T-29: Presentación y preparación de la defensa.

T-30	**Revisión de la memoria.**
Descripción	Búsqueda de errores ortográficos, técnicos y de estilo. Redacción final de aquellos apartados susceptibles de mejora.

Tabla 2.38: T-30: Revisión de la memoria.

Fase	Horas
Fase de Inicio	63
Fase de Elaboración o Análisis	70
Fase de Construcción	109
Fase de Transición y Pruebas	58
Total	**300**

Tabla 2.39: Distribución temporal inicial por fases.

2.6. Seguimiento

En este subcapítulo se detallarán los tiempos reales empleados para cada tarea de cada fase del proyecto, en comparación con los previstos en la planificación inicial 2.5. Se obtendrán conclusiones acerca de los motivos de las desviaciones.

2.6.1. Planificación real

En comparación al desglose de horas semanales de la planificación inicial, tabla 2.3 se expone la tabla 2.40. Del mismo modo que se puede comparar la dedicación final de horas, tabla 2.41 con respecto a la distribución inicial de la tabla 2.4.

Nº	Inicio	Final	Horas	Observaciones/ Incidencias
1	16/02/2021	23/02/2021	14	
2	23/02/2021	02/03/2021	18	
3	02/03/2021	09/03/2021	0	Riesgo R10.
4	09/03/2021	16/03/2021	15	Riesgo R07
5	16/03/2021	23/03/2021	20	
6	23/03/2021	30/03/2021	20	
7	30/03/2021	06/04/2021	10	Riesgo R09.
8	06/04/2021	13/04/2021	20	
9	13/04/2021	20/04/2021	10	Riesgo R09.
10	20/04/2021	27/04/2021	12	Riesgo R09.
11	27/04/2021	04/05/2021	20	
12	04/05/2021	11/05/2021	20	
13	11/05/2021	18/05/2021	12	Riesgo R10.
14	18/05/2021	25/05/2021	20	
15	25/05/2021	01/06/2021	35	Finalización de prácticas en empresa.
16	01/06/2021	06/06/2021	35	
17	07/06/2021	13/06/2021	30	
18	14/06/2021	20/06/2021	25	Revisión final y preparación de defensa.
Fin	**20/06/2021**	**Total**	**336**	**Se excede en 36 horas la planificación inicial.**

Tabla 2.40: Planificación real.

Sección	Horas	Porcentaje
Investigación, documentación e implementación.	231	77 %
Redacción de la memoria y adecuación a la plantilla.	79	26,33 %
Tutorías	20	6,66 %
Evaluación y defensa	6	2 %
Total	**336**	**112 %**

Tabla 2.41: Distribución real por secciones (porcentajes en base a 300 horas).

2.6.2. Plan de trabajo real

Fase de Inicio

Se estimó como fecha de comienzo el 16/02/2021 y de fin el 14/03/2021. Finalmente esta fase estuvo comprendida entre el 16/02/2021 y el 26/03/2021.

Nombre de la tarea	Duración	Duración real
T-01 Toma de contacto inicial con el problema.	5 horas	5 horas
T-02 Análisis de riesgos.	2,5 horas	3 horas
T-03 Análisis de costes.	2,5 horas	2 horas
T-04 Búsqueda de aplicaciones similares.	2 horas	2 horas
T-05 Formación en Unity y C#.	20 horas	20 horas
T-06 Preparación del entorno de trabajo.	3 horas	2 horas
T-07 Plan de trabajo.	3 horas	3 horas
T-08 Redacción de la memoria.	20 horas	28 horas
T-09 Tutorías online y mails.	5 horas	5 horas
Total fase de Inicio.	**63 horas**	**70 horas**

Tabla 2.42: Plan de trabajo real de la fase de Inicio.

Fase de Elaboración o Análisis

Se estimó como fecha de comienzo el 15/03/2021 y de fin el 15/04/2021. Finalmente esta fase estuvo comprendida entre el 27/03/2021 y el 23/04/2021.

Nombre de la tarea	Duración	Duración real
T-10 Establecimiento de requisitos.	2 horas	2 horas
T-11 Modelo de casos de uso.	5 horas	5 horas
T-12 Especificación de casos de uso.	8 horas	7 horas
T-13 Modelo de dominio.	2 horas	2 horas
T-14 Diagramas de proceso/ flujo.	5 horas	6 horas
T-15 Diseño de la Interfaz de Usuario.	12 horas	12 horas
T-16 Investigación de arquitecturas en Unity.	3 horas	3 horas
T-17 Arquitectura de la aplicación.	7 horas	8 horas
T-08 Redacción de la memoria.	20 horas	18 horas
T-09 Tutorías online y mails.	6 horas	4 horas
Total fase de Elaboración o Análisis.	**70 horas**	**67 horas**

Tabla 2.43: Plan de trabajo real de la fase de Elaboración o Análisis.

Fase de Construcción

Se estimó como fecha de comienzo el 16/04/2021 y de fin el 01/06/2021. Finalmente esta fase estuvo comprendida entre el 24/04/2021 y el 08/06/2021.

Nombre de la tarea	Duración	Duración real
T-18 Implementación del menú principal.	5 horas	5 horas
T-19 Implementación de la escena jugadas.	25 horas	41 horas
T-20 Implementación de la escena listar jugadas.	10 horas	10 horas
T-21 Implementación de la escena pizarra.	20 horas	14 horas
T-22 Implementación de la escena plantilla.	15 horas	20 horas
T-23 Implementación de la escena ajustes.	10 horas	16 horas
T-24 Implementación del cambio de idiomas.	10 horas	13 horas
T-08 Redacción de la memoria.	10 horas	8 horas
T-09 Tutorías online y mails.	4 horas	4 horas
Total fase de Construcción.	**109 horas**	**131 horas**

Tabla 2.44: Plan de trabajo real de la fase de Construcción.

Fase de Transición y Pruebas

Se finaliza el TFG 2 semanas más tarde de lo planificado, el 20/06/2021.

Nombre de la tarea	Duración	Duración real
T-25 Pruebas personales de funcionalidad.	20 horas	23 horas
T-26 Pruebas finales de funcionalidad con usuarios.	5 horas	6 horas
T-27 Elaboración del manual de instalación.	5 horas	5 horas
T-28 Elaboración del manual de usuario.	3 horas	3 horas
T-29 Presentación y preparación de la defensa.	5 horas	7 horas
T-30 Revisión de la memoria.	15 horas	17 horas
T-09 Tutorías online y mails.	5 horas	7 horas
Total fase de Transición y Pruebas.	**58 horas**	**68 horas**

Tabla 2.45: Plan de trabajo real de la fase de Transición y Pruebas.

2.6.3. Desviaciones en la planificación

En la tabla 2.46 se aprecia con respecto a la tabla 2.39, cómo el número de horas empleadas ha sido superior a lo planificado.

Fase	Horas	Horas reales
Fase de Inicio	63	70
Fase de Elaboración o Análisis	70	67
Fase de Construcción	109	131
Fase de Transición y Pruebas	58	68
Total	**300**	**336**

Tabla 2.46: Distribución temporal real por fases.

Pero, ¿a qué se ha debido en cada fase? A continuación expongo las conclusiones que he podido extraer, ya que reflexionando será menos probable que vuelva a ocurrir en proyectos futuros.

Fase de Inicio

En esta fase destaca el tiempo de elaboración de la plantilla, ya que no tenía mucha experiencia con LaTeX y tuve que aprender casi de cero. Por contra, se consiguió una plantilla que hubo que modificar mínimamente, por lo que los tiempos de redacción se redujeron en las siguientes fases. Tomando esto en consideración, no debería repetirse, al haber ganado experiencia en el manejo de esta tecnología.

Fase de Elaboración o Análisis

La fase de análisis no tuvo desviaciones, debido a que es la parte más trabajada a lo largo de la carrera.

Fase de Construcción

El desconocimiento de Unity, pese a la realización de diversos tutoriales [12] [7] al inicio juega un papel determinante en el tiempo de implementación. La escena jugadas, al ser la primera en realizarse, es la que más tiempo lleva, no tanto por complejidad, si no por ser la primera toma de contacto. La gestión de la plantilla también resulta un tanto más compleja debido a tener que comprobar y sobreescribir los jugadores tanto en creación como edición.

Por otra parte, cabe destacar cómo la pizarra llevó menos tiempo debido al asset comentado en el apartado 5.5.2. Esta solución casi completa al paradigma, permitió reducir tiempos en esta fase, al tener únicamente que adaptar el código.

Se puede concluir que la cantidad de horas necesarias debió ser mayor, por el hecho de tratarse de una nueva tecnología, algo a tener en consideración en el aspecto laboral.

Fase de Transición y Pruebas

Finalmente el tiempo de pruebas es mayor debido a la incorporación de nuevas funcionalidades en pro de completar la app. También se cuenta con más usuarios de prueba, lo que lleva implícito más tiempo de explicación, resolución de dudas y apunte de sugerencias.

De igual manera se emplea más tiempo del establecido en las revisiones de la memoria (completando alguna sección, como esta) y la presentación, de cara a la defensa.

Capítulo 3

Análisis

En este capítulo se detalla la aplicación a desarrollar con las tareas de análisis, la parte troncal de cualquier desarrollo software, que ello conlleva: actores, requisitos, el tratamiento de los datos, los casos de uso a cubrir, el modelo de dominio y el flujo de la aplicación.

3.1. Actores y roles de la aplicación

Esta aplicación no implementa diferencias entre administrador, desarrollador y usuario, puesto que se enfoca como una app de usuario final, la cual pueda ser usada por completo de manera sencilla y auto contenida, sin conexión a internet tras su descarga.

Debido a esto, puede apreciarse que únicamente existe un rol en la aplicación, el usuario, el cual tiene acceso a la funcionalidad completa.

Este rol puede ser desempeñado por dos actores, el entrenador y su asistente (aunque realmente no se necesitan conocimientos avanzados de balonmano para usarla, por lo que cualquier persona puede desempeñar el rol de usuario). De ahora en adelante me referiré con entrenador (englobando a su asistente), al rol de usuario.

3.2. Requisitos

A continuación se detallan los requisitos del proyecto en función de los objetivos establecidos, y siguiendo como guía el libro de Sommerville utilizado previamente en la carrera [41]. Se han añadido nuevos requisitos no contemplados al inicio del proyecto, por disponer de más tiempo para desarrollar.

3.2.1. Requisitos Funcionales

Parafraseando la definición de [37] los requisitos funcionales son la "Definición de los servicios que el sistema debe proporcionar, cómo debe reaccionar a una entrada particular, y cómo debe comportarse ante situaciones particulares". Es decir, las funcionalidades que debe ser capaz de realizar la aplicación, las cuales aparecen en la tabla 3.1.

ID	Descripción
RF01	**Jugadas.** El sistema debe permitir añadir nuevas jugadas al sistema.
RF02	**Jugadas II.** El sistema permitirá mostrar todas las jugadas presentes en el sistema y reproducirlas.
RF03	**Jugadas III.** El sistema debe permitir eliminar jugadas. Para poder eliminar una jugada el sistema solicitará confirmación al usuario.
RF04	**Dibujo en pizarra digital.** El sistema debe permitir al usuario dibujar libremente en una pizarra digital que tenga por fondo un campo de balonmano reglamentario. Tendrá a su disposición más de un color, un borrador, la posibilidad de cambiar el grosor de estos y la opción de reiniciar la pizarra.
RF05	**Capturas de pantalla.** El sistema permitirá al usuario guardar capturas de pantalla de sus dibujos en la pizarra libre.
RF06	**Plantilla.** El sistema permitirá al usuario añadir jugadores a la plantilla del equipo.
RF07	**Plantilla II.** El sistema permitirá mostrar al usuario los jugadores que conforman su plantilla.
RF08	**Plantilla III.** El sistema deberá permitir al usuario modificar a los jugadores desde el listado de plantilla.
RF09	**Plantilla IV.** El sistema debe permitir eliminar jugadores. Para poder eliminar un jugador el sistema solicitará confirmación al usuario.
RF10	**Ajustes.** El sistema deberá permitir al usuario seleccionar los colores de los jugadores de su equipo y del equipo rival.
RF11	**Idiomas.** El sistema debe permitir elegir el idioma en que se muestra la app, que seleccione el usuario de entre los posibles.

Tabla 3.1: Requisitos funcionales.

3.2.2. Requisitos No Funcionales

Los Requisitos No Funcionales o RNF, tabla 3.2 definen propiedades del sistema, cuán fiable debe ser, su velocidad, sus necesidades de almacenamiento, etc.

ID	Descripción
RNF01	**Adaptabilidad.** El sistema debe asegurar que toda la aplicación se visualiza correctamente en cualquier pantalla con relación de aspecto 16:9.
RNF02	**Codificación de caracteres.** El sistema deberá utilizar el formato UTF-8, y no mostrar problemas tipográficos.
RNF03	**Usabilidad.** El sistema debe ser muy usable, y tener una curva de aprendizaje casi plana. Se debe enfocar al perfil de un entrenador de entre 20 y 65 años, que tiene conocimientos generales de informática.
RNF04	**Responsividad.** El sistema debe ser altamente responsivo, no mostrando pantallas de carga superiores a los 2 segundos.
RNF05	**Compatibilidad.** El sistema debe poder ejecutarse en cualquier dispositivo Android compatible, siendo la versión 4.4 la mínima para su funcionamiento. De igual manera deberá garantizar su funcionamiento en Windows 10 y MacOS 10.15, o superiores.
RNF06	**Conectividad.** Una vez descargada, la aplicación no hará uso de internet.
RNF07	**Fiabilidad.** El sistema deberá estar disponible las 24 horas del día, todos los días del año.
RNF08	**Ajustes de color.** El sistema deberá ser capaz de mostrar una previsualización en la elección de color, desde los ajustes.
RNF09	**Multilenguaje.** El sistema deberá funcionar completamente en los idiomas: español, inglés, francés y alemán. Los comentarios añadidos por el usuario no serán traducidos.
RNF10	**Multilenguaje II.** El sistema deberá mostrar por defecto el lenguaje español, y posteriormente, aquel lenguaje que fuera seleccionado.
RNF11	**Compatibilidad de ficheros.** El sistema deberá permitir exportar e importar fácilmente los ficheros generados, que deben ser 100 % compatibles entre los sistemas operativos.

Tabla 3.2: Requisitos no funcionales.

3.2.3. Requisitos Funcionales de Información

Los Requisitos Funcionales de Información o RFI, en la tabla 3.3, describen requisitos funcionales, que declaran la información que es almacenada en el sistema, para posibilitar el funcionamiento.

ID	Descripción
RFI01	**Jugadas.** El sistema guardará información sobre la posición de los jugadores de ambos equipos, así como el avance por el campo de los mismos, y la ubicación del balón reglamentario, para cada jugada existente.
RFI02	**Jugadores.** El sistema almacena el nombre, dorsal y posición de los jugadores de la plantilla.
RFI03	**Capturas de pantalla.** El sistema almacena las capturas de pantalla de la pizarra de dibujo libre.
RFI04	**Privacidad.** El sistema no almacenará datos personales, potencialmente sensibles de ningún usuario, no vulnerando así ni su privacidad ni la LO PDPGDD, Ley Orgánica 3/2018 de Protección de Datos Personales y Garantía de los Derechos Digitales.

Tabla 3.3: Requisitos funcionales de información.

3.3. Protección de datos

La sociedad actual, cada vez más informatizada y conectada, hace ya unos años que se encuentra sumida en un gran problema: la exposición de sus datos personales al resto del mundo, quedando a merced de posibles atacantes.

Por ello, y como se estudia en la asignatura Profesión y Sociedad [19], es muy importante asegurar la integridad de los datos de los usuarios, a la par que debemos informarles sobre qué datos son recogidos. Actualmente, en nuestro país, ésto se rige por la LO PDPGDD, Ley Orgánica 3/2018 de Protección de Datos Personales y Garantía de los Derechos Digitales, así como por el RGPD, Reglamento General de Protección de Datos.

Se debe informar al usuario de aquellos datos que serán almacenados (proporcionados por ellos mismos) y de aquellos procesados o generados en base al registro de uso u otras acciones. El usuario, debe dar consentimiento inequívoco y explícito para el tratamiento de sus datos, así como del conocimiento de ello. Se debe plasmar la privacidad y uso que se darán a los datos personales.

No olvidemos que no todos los datos son iguales, hay datos especialmente sensibles, aquellos que pueden identificar a una persona como son: su nombre completo, su dirección, el DNI, datos económicos o laborales, valores biométricos, su localización... todos éstos y muchos otros, requieren especial cuidado en su trato. El verdadero peligro reside en que pueden ser cruzados, haciendo así identificable a una persona, creando un perfil digital que vulnere su privacidad.

Por ello, los datos han de ser anonimizados por las empresas, en caso de querer realizar estadísticas o análisis para su negocio. Y en ningún caso deberían venderse en bruto, es decir, no se pueden ceder datos a otras empresas sin el consentimiento explícito del usuario, puesto que redundaría en la pérdida del control, y sería más fácil realizar un perfil de la persona, atentando así contra la normativa vigente.

La mejor solución para garantizar la privacidad, es pedir la menor cantidad de datos necesaria. Razonando sobre esto, varias decisiones fueron tomadas en el desarrollo de esta aplicación:

- **Registro.** No se solicitará al usuario un registro, puesto que no supone ventaja alguna en el uso. Esto evita registrar cualquier dato personal, por nimio que sea.
- **Datos de jugadores.** El único dato personal que se solicita en toda la aplicación es el nombre de los jugadores del equipo. Cabe recalcar que el nombre, por sí solo, no permite identificar a una persona, ni siquiera pese a ir acompañado de una posición de juego y un dorsal. Además de ser insuficientes para trazar un perfil, son fácilmente reconocibles por cualquiera que asista a un partido. Y por si fuera poco, podrían usarse apodos en lugar del nombre real.
- **Almacenamiento interno.** Los datos generados por la aplicación se almacenan únicamente en el dispositivo; prescindir de una copia de seguridad en la nube tiene sus desventajas, pero por otra parte lo hace independiente de las brechas de seguridad que tengan dichos sistemas, y permite cumplir el RNF-06 (funcionamiento sin conexión a internet).

En definitiva, esta aplicación no solicita datos personales críticos, y no se utilizan de ninguna manera. Pese a que se pueden compartir fácilmente, al no estar encriptados, su pérdida no supondría delito alguno, por lo expuesto anteriormente. El entrenador, así como sus jugadores, pueden estar plenamente tranquilos acerca de su privacidad en la app EntrenaBal.

3.4. Casos de uso

3.4.1. Diagrama de casos de uso

Tras los requisitos, y consultando [26], se realiza el diagrama de casos de uso, que como se menciona en el apartado 5.2.2, expone de manera gráfica y simple la interacción entre el usuario (el entrenador en este caso) y el sistema/ aplicación. Este diagrama, figura 3.1, presenta por tanto al actor y las acciones que puede realizar (contenidas en elipses), siendo los "extend" casos de uso adicionales, a los que se pueden acceder desde el caso de uso que las engloba. Esto es así ya que amplían la funcionalidad del caso general y pueden llevarse a cabo o no.

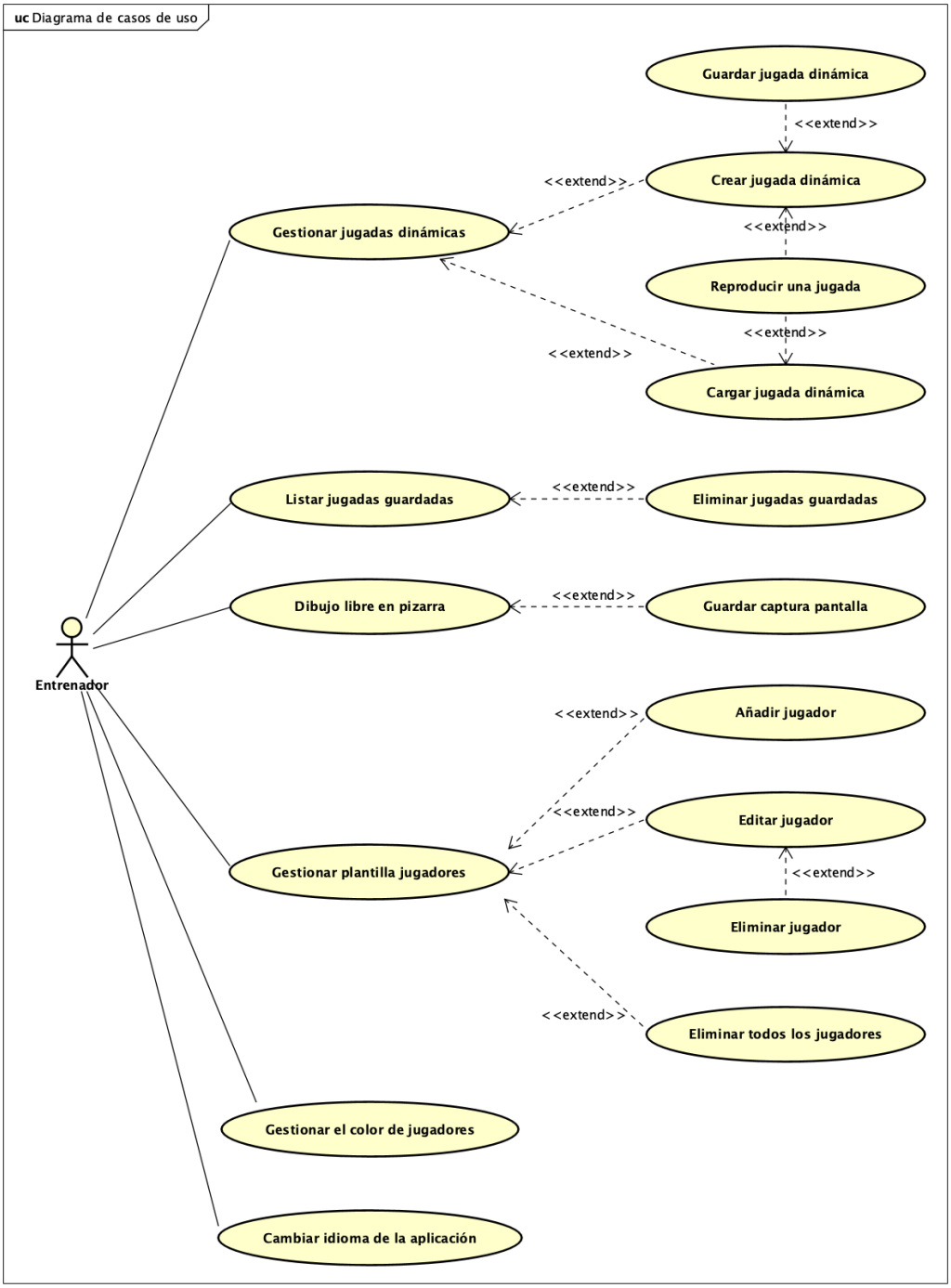

Figura 3.1: Diagrama de casos de uso.

3.4.2. Especificación de los casos de uso

Acto seguido, se detallan todos los casos de uso. Estos escenarios, detallan el flujo **normal** de acciones entre el usuario y el sistema, indicando el paso en que se encuentra, quién realiza la acción y en qué consiste la misma.

Seguidamente, se presentan las extensiones al flujo habitual, las posibles variantes en la ejecución o cancelaciones prematuras del caso de uso concreto, identificadas por el número de la secuencia principal (con una letra correlativa) y la condición que debe cumplirse para que se dé. El usuario generalmente es referido como actor, ya que es quien actúa (quien comienza todo caso de uso).

Caso de Uso CU-01: Gestionar jugadas dinámicas		
Precondición		Haber iniciado la aplicación y encontrarse en el menú principal.
Paso	Control	Descripción
1	Usuario	Indica que quiere gestionar jugadas dinámicas.
2	SISTEMA	Solicita al actor que elija entre los casos de uso “Crear jugada dinámica” y “Cargar jugada dinámica”.
3	Usuario	El actor indica el caso de uso que quiere realizar.
4	SISTEMA	Según la elección, navega al caso de uso correspondiente. El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario solicita regresar al menú.	
	SISTEMA	Informa al actor de la cancelación (regresando al menú).
	salir	Se abandona el caso de uso, quedando sin efecto.
3.b	Si el usuario solicita realizar el caso de uso “Crear jugada dinámica”.	
	SISTEMA	Navega al caso de uso “Crear jugada dinámica”. El caso de uso termina con éxito..
3.c	Si el usuario solicita realizar el caso de uso “Cargar jugada dinámica”.	
	SISTEMA	Navega al caso de uso “Cargar jugada dinámica”. El caso de uso termina con éxito..
Postcondición		Se permiten las operaciones crear, cargar y reproducir jugadas.

Tabla 3.4: Escenario del Caso de Uso CU-01: Gestionar jugadas dinámicas.

Caso de Uso CU-02: Crear jugada dinámica		
Precondición		Haber iniciado la aplicación y encontrarse en la escena Jugadas.
Paso	Control	Descripción
1	Usuario	Indica que quiere crear una jugada dinámica.
2	SISTEMA	Solicita al actor que añada un nuevo paso para la jugada.
3	Usuario	Especifica los movimientos de un paso, creando éste pulsando un botón.
4	SISTEMA	Solicita al actor que visualice la jugada para observar el progreso.
5	Usuario	Especifica que quiere visualizar el progreso pulsando un botón.
6	SISTEMA	Solicita al actor que guarde la jugada una vez satisfecho.
7	SISTEMA	Ejecuta el caso de uso “Reproducir una jugada”.
8	Usuario	Especifica que quiere guardar la jugada pulsando un botón.
9	SISTEMA	Navega al caso de uso “Guardar jugada dinámica”. El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a 5.a 8.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Solicita al actor confirmación para cancelar el proceso.
	Usuario	Confirma que quiere cancelar el proceso.
	SISTEMA	Informa al actor de la cancelación (volviendo al menú previo).
	salir	Se abandona el caso de uso, quedando sin efecto.
3.b	Si el usuario quiere añadir otro paso a la jugada.	
	SISTEMA	Solicita al actor que añada un nuevo paso para la jugada.
	Control	Se retorna el control al paso 3.
5.b	Si el usuario no quiere visualizar el progreso de la jugada.	
	Control	Se retorna el control al paso 6.
5.c	Si el usuario quiere añadir un paso tras visualizar el avance de la jugada.	
	SISTEMA	Solicita al actor que añada un nuevo paso para la jugada.
	Control	Se retorna el control al paso 3.
8.b	Si el usuario quiere seguir añadiendo pasos.	
	Control	Se retorna el control al paso 3.
8.c	Si el usuario no quiere guardar la jugada.	
	control	Se retorna el control al paso 3.
Postcondición		Se habrá creado una jugada que puede ser reproducida y/o guardada.

Tabla 3.5: Escenario del Caso de Uso CU-02: Crear jugada dinámica.

Caso de Uso CU-03: Guardar jugada dinámica		
Precondición		Se debe realizar el CU-02 completo.
Paso	Control	Descripción
1	Usuario	Indica que quiere guardar los pasos establecidos como una nueva jugada.
2	SISTEMA	Solicita al actor que especifique el nombre de la jugada.
3	Usuario	Indica el nombre de la jugada.
4	SISTEMA	Registra la nueva jugada con el nombre asociado y los pasos que la conforman.
5	SISTEMA	El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario solicita cancelar la operación.	
5.a	SISTEMA	Informa al actor de la cancelación (volviendo al menú previo).
7.a	salir	Se abandona el caso de uso, quedando sin efecto.
3.b	Si el nombre identificativo de la jugada ya existía.	
	SISTEMA	Informa al actor del suceso, solicitando un nuevo nombre.
	Control	Se retorna el control al paso 3.
3.c	Si el nombre identificativo de la jugada es nulo.	
	SISTEMA	Informa al actor del suceso, solicitando un nuevo nombre.
	Control	Se retorna el control al paso 3.
Postcondición		Se habrá guardado la jugada creada en el CU-02.

Tabla 3.6: Escenario del Caso de Uso CU-03: Guardar jugada dinámica.

Caso de Uso CU-04: Reproducir una jugada		
Precondición		Disponer de una jugada, ya sea creándola según el CU-02 o cargándola según el CU-05.
Paso	Control	Descripción
1	Usuario	Indica que quiere reproducir una jugada previamente guardada o también al crearse.
2	SISTEMA	Solicita al actor que pulse el botón de reproducir.
3	Usuario	El usuario pulsa el botón reproducir.
4	SISTEMA	Procede a ejecutar la animación de pasos. El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Informa al actor de la cancelación (volviendo al menú previo).
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		Se habrá reproducido la animación de la jugada elegida.

Tabla 3.7: Escenario del Caso de Uso CU-04: Reproducir una jugada.

Caso de Uso CU-05: Cargar jugada dinámica		
Precondición		Haber creado antes una jugada, siguiendo el CU-02. Encontrarse en la escena jugadas.
Paso	Control	Descripción
1	Usuario	Indica que quiere cargar una jugada previamente guardada.
2	SISTEMA	Muestra el listado de las jugadas almacenadas.
3	SISTEMA	Solicita al actor que elija una de las jugadas almacenadas.
4	Usuario	Indica cuál de las jugadas disponibles quiere cargar.
5	SISTEMA	Carga los pasos de la jugada elegida para poder reproducirlos. El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
4.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Informa al actor de la cancelación (regresando a la pantalla previa).
	salir	Se abandona el caso de uso, quedando sin efecto.
4.b	Si no hay jugadas guardadas.	
	Usuario	Cancela la acción.
	SISTEMA	Informa al actor de la cancelación (regresando a la pantalla previa).
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		Se habrán cargado los pasos de la jugada seleccionada, permitiendo su edición, reproducción y guardado.

Tabla 3.8: Escenario del Caso de Uso CU-05: Cargar jugada dinámica.

Caso de Uso CU-06: Listar jugadas guardadas		
Precondición		Haber iniciado la aplicación y encontrarse en el menú principal.
Paso	Control	Descripción
1	Usuario	Indica que quiere visualizar en una tabla todas las jugadas almacenadas.
2	SISTEMA	Navega a la escena Listar jugadas. El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
1.a	Si el usuario no quiere navegar.	
	SISTEMA	Informa al actor de la cancelación (No navegando a la escena Listar jugadas).
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		Se muestran por pantalla todas las jugadas presentes en el sistema.

Tabla 3.9: Escenario del Caso de Uso CU-06: Listar jugadas guardadas.

Caso de Uso CU-07: Eliminar jugadas guardadas		
Precondición		Disponer de al menos una jugada almacenada en el sistema, siguiendo el CU-03.
Paso	Control	Descripción
1	Usuario	Selecciona las jugadas guardadas a eliminar.
2	SISTEMA	Solicita al actor una confirmación para eliminar las jugadas.
3	Usuario	Indica que está conforme con el borrado de las jugadas seleccionadas.
4	SISTEMA	Elimina las jugadas del almacenamiento.
5	SISTEMA	Recarga el listado de las jugadas guardadas, actualizado. El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
1.a 3.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Informa al actor de la cancelación (regresando al menú).
	salir	Se abandona el caso de uso, quedando sin efecto.
3.b	El usuario no muestra su conformidad.	
	Control	Se retorna el control al paso 5.
4.a	Si el usuario solicitó el borrado de TODAS las jugadas presentes.	
	SISTEMA	Solicita una confirmación extra al actor para eliminar las jugadas.
	Usuario	Indica que está conforme con el borrado de las jugadas seleccionadas.
	Control	Se retorna el control al paso 4.
Postcondición		Se habrán eliminado del sistema las jugadas que seleccione, no apareciendo éstas en la tabla.

Tabla 3.10: Escenario del Caso de Uso CU-07: Eliminar jugadas guardadas.

Caso de Uso CU-08: Dibujo libre en pizarra		
Precondición		Haber iniciado la aplicación y encontrarse en la escena Pizarra.
Paso	Control	Descripción
1	Usuario	Indica que quiere dibujar una jugada en la pizarra.
2	Usuario	Dibuja los elementos que cree oportunos hasta finalizar.
3	SISTEMA	Recuerda al actor que puede realizar una captura de pantalla del resultado.
4	Usuario	Indica que quiere realizar una captura de pantalla.
5	SISTEMA	Navega al caso de uso “Guardar captura de pantalla”.
6	SISTEMA	El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
2.a 4.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Informa al actor de la cancelación.
	salir	Se abandona el caso de uso, quedando sin efecto.
2.b	Si el usuario solicita borrar el progreso.	
	SISTEMA	Solicita al actor confirmación para borrar el progreso.
	Usuario	Confirma que quiere borrar el progreso.
	SISTEMA	Vuelve al menú anterior actualizando la pizarra.
	Control	Se retorna el control al paso 2.
4.b	Si el usuario no quiere realizar una captura.	
	Control	Se retorna el control al paso 6.
Postcondición		El usuario habrá podido pintar a su gusto, pudiendo guardar una captura si realiza el CU-09.

Tabla 3.11: Escenario del Caso de Uso CU-08: Dibujo libre en pizarra.

Caso de Uso CU-09: Guardar captura pantalla		
Precondición		Haber realizado el CU-08 hasta el paso 2.
Paso	Control	Descripción
1	Usuario	Indica que quiere guardar una captura de pantalla.
2	SISTEMA	Solicita al actor que pulse el botón “Guardar captura”.
3	Usuario	Pulsa el botón “Guardar captura”.
4	SISTEMA	Guarda una captura de pantalla de la pizarra.
5	SISTEMA	Muestra un mensaje informativo al usuario. El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario no solicita tomar la captura de pantalla.	
	SISTEMA	Informa al actor de la cancelación, no tomando la captura, ni mostrando mensaje alguno.
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		El sistema contendrá la captura de pantalla que corresponde a los dibujos del CU-08.

Tabla 3.12: Escenario del Caso de Uso CU-09: Guardar captura pantalla.

Caso de Uso CU-10: Gestionar plantilla jugadores		
Precondición		Haber iniciado la aplicación y encontrarse en el menú principal.
Paso	Control	Descripción
1	Usuario	Indica que quiere gestionar la plantilla de jugadores.
2	SISTEMA	Solicita al actor que elija entre los casos de uso "Añadir jugador", "Editar jugador" y "Eliminar todos los jugadores".
3	Usuario	El actor indica el caso de uso que quiere realizar.
4	SISTEMA	Según la elección, navega al caso de uso correspondiente. El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario solicita regresar al menú.	
	SISTEMA	Informa al actor de la cancelación (regresando al menú).
	salir	Se abandona el caso de uso, quedando sin efecto.
3.b	Si el usuario solicita realizar el caso de uso "Añadir jugador".	
	SISTEMA	Navega al caso de uso "Añadir jugador". El caso de uso termina con éxito..
3.c	Si el usuario solicita realizar el caso de uso "Editar jugador".	
	SISTEMA	Navega al caso de uso "Editar jugador". El caso de uso termina con éxito..
3.d	Si el usuario solicita realizar el caso de uso "Eliminar todos los jugadores".	
	SISTEMA	Navega al caso de uso "Eliminar todos los jugadores". El caso de uso termina con éxito..
Postcondición		Se muestran las acciones añadir, editar y eliminar todos los jugadores al usuario, permitiendo la correcta navegación a cada uno de ellos.

Tabla 3.13: Escenario del Caso de Uso CU-10: Gestionar plantilla jugadores.

Caso de Uso CU-11: Añadir jugador		
Precondición		Haber iniciado la aplicación y encontrarse en la escena Plantilla.
Paso	Control	Descripción
1	Usuario	Indica que quiere añadir un jugador a la plantilla.
2	SISTEMA	Solicita al actor que pulse el botón “Añadir jugador”.
3	Usuario	Pulsa el botón “Añadir jugador”.
4	SISTEMA	Solicita al actor que especifique el dorsal del jugador.
5	Usuario	Indica el dorsal del jugador.
6	SISTEMA	Solicita al actor que seleccione la posición del jugador.
7	Usuario	Selecciona la posición del jugador.
8	SISTEMA	Solicita al actor que especifique el nombre del jugador.
9	Usuario	Indica el nombre del jugador.
10	SISTEMA	Solicita al actor que proporcione una descripción del jugador.
11	Usuario	Proporciona una descripción del jugador.
12	SISTEMA	Solicita al actor que pulse el botón “Guardar” cuando esté conforme.
13	Usuario	Pulsa el botón “guardar”.
14	SISTEMA	Registra el nuevo jugador con el dorsal asociado como nombre de fichero.
15	SISTEMA	El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a, 5.a 7.a, 9.a 11.a, 13.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Solicita al actor confirmación para cancelar el proceso.
	Usuario	Confirma que quiere cancelar el proceso.
	SISTEMA	Informa al actor de la cancelación (volviendo al menú previo).
	salir	Se abandona el caso de uso, quedando sin efecto.
11.b	Si el usuario no quiere proporcionar una descripción.	
	Control	Se retorna el control al paso 12.
14.a	Si no se ha proporcionado un dorsal válido, un nombre y una posición.	
	SISTEMA	Informa al actor del suceso, instándole a que complete todos los campos obligatorios.
	Control	Se retorna el control al paso 4.
14.b	Si el dorsal del jugador ya está registrado.	
	SISTEMA	Informa al actor del suceso, instándole a que introduzca otro diferente.
	Control	Se retorna el control al paso 4.
Postcondición		El sistema cuenta con un nuevo jugador, cuyos datos corresponden con los introducidos.

Tabla 3.14: Escenario del Caso de Uso CU-11: Añadir jugador.

Caso de Uso CU-12: Editar jugador		
Precondición		Disponer de al menos un jugador en el sistema. Encontrarse en la escena plantilla.
Paso	Control	Descripción
1	Usuario	Indica que quiere editar un jugador de la plantilla.
2	SISTEMA	Solicita al actor que pulse el botón “Editar jugador” en la fila correspondiente.
3	Usuario	Pulsa el botón “Editar jugador” de la fila correspondiente.
4	SISTEMA	Solicita al actor que especifique el nuevo dorsal del jugador.
5	Usuario	Indica el nuevo dorsal del jugador.
6	SISTEMA	Solicita al actor que seleccione la nueva posición del jugador.
7	Usuario	Selecciona la nueva posición del jugador.
8	SISTEMA	Solicita al actor que especifique el nuevo nombre del jugador.
9	Usuario	Indica el nuevo nombre del jugador.
10	SISTEMA	Solicita al actor que proporcione una nueva descripción del jugador.
11	Usuario	Proporciona una nueva descripción del jugador.
12	SISTEMA	Solicita al actor que pulse el botón “Guardar” cuando esté conforme.
13	Usuario	Pulsa el botón “guardar”.
14	SISTEMA	Registra el nuevo jugador con el dorsal asociado como nombre de fichero.
15	SISTEMA	El caso de uso termina con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a, 5.a 7.a, 9.a 11.a, 13.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Solicita al actor confirmación para cancelar el proceso.
	Usuario	Confirma que quiere cancelar el proceso.
	SISTEMA	Informa al actor de la cancelación (volviendo al menú previo).
	salir	Se abandona el caso de uso, quedando sin efecto.
3.b, 5.b 7.b, 9.b 11.b, 13.b	Si el usuario solicita borrar al jugador.	
	SISTEMA	Solicita al actor confirmación para borrar al jugador.
	Usuario	Confirma que quiere cancelar el proceso.
	SISTEMA	Navega al caso de uso “Eliminar jugador”.
5.c	Si el usuario no quiere proporcionar un nuevo dorsal.	
	Control	Se retorna el control al paso 6.
7.c	Si el usuario no quiere proporcionar una nueva posición.	
	Control	Se retorna el control al paso 8.
9.c	Si el usuario no quiere proporcionar un nuevo nombre.	
	Control	Se retorna el control al paso 10.
11.c	Si el usuario no quiere proporcionar una descripción.	
	Control	Se retorna el control al paso 12.
14.a	Si no se ha proporcionado un dorsal válido, un nombre y una posición.	
	SISTEMA	Informa al actor del suceso, instándole a que complete todos los campos obligatorios.
	Control	Se retorna el control al paso 4.
14.b	Si el dorsal del jugador ya está registrado.	
	SISTEMA	Informa al actor del suceso, instándole a que introduzca otro diferente.
	Control	Se retorna el control al paso 4.
Postcondición		El jugador seleccionado del sistema modifica sus datos con los nuevos introducidos, no duplicándose ni eliminando otro jugador con mismo dorsal.

Tabla 3.15: Escenario del Caso de Uso CU-12: Editar jugador.

Caso de Uso CU-13: Eliminar jugador		
Precondición		Disponer de al menos un jugador en el sistema, el cual se está editando.
Paso	Control	Descripción
1	Usuario	Indica que quiere eliminar al jugador que está siendo editado.
2	SISTEMA	Solicita al actor que pulse el botón “Borrar jugador”.
3	Usuario	Pulsa el botón “Borrar jugador”.
4	SISTEMA	Solicita al actor una confirmación para eliminar las jugadas.
5	Usuario	Indica que está conforme con el borrado del jugador.
6	SISTEMA	Elimina al jugador del almacenamiento.
7	SISTEMA	Recarga el listado de jugadores, actualizado. El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
1.a 3.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Informa al actor de la cancelación.
	salir	Se abandona el caso de uso, quedando sin efecto.
5.a	El usuario no muestra su conformidad.	
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		El jugador seleccionado es eliminado del sistema y ya no aparece en la tabla de plantilla.

Tabla 3.16: Escenario del Caso de Uso CU-13: Eliminar jugador.

Caso de Uso CU-14: Eliminar a todos los jugadores		
Precondición		Disponer de al menos un jugador en el sistema, realizando el CU-11.
Paso	Control	Descripción
1	Usuario	Indica que quiere eliminar todos los jugadores de la plantilla.
2	SISTEMA	Solicita al actor que pulse el botón “Borrar TODO”.
3	Usuario	Pulsa el botón “Borrar TODO”.
4	SISTEMA	Solicita al actor una confirmación para eliminar a todos los jugadores.
5	Usuario	Indica que está conforme con el borrado de todos los jugadores.
6	SISTEMA	Solicita al actor una confirmación extra para eliminar finalmente a todos los jugadores.
7	Usuario	Indica que está seguro del borrado de todos los jugadores.
8	SISTEMA	Elimina todos los jugadores del almacenamiento.
9	SISTEMA	Recarga el listado de jugadores, actualizado. El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
1.a 3.a	Si el usuario solicita cancelar la operación.	
	SISTEMA	Informa al actor de la cancelación (regresando al menú).
	salir	Se abandona el caso de uso, quedando sin efecto.
5.a	El usuario no muestra su conformidad.	
7.a	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		Todos los jugadores son eliminados del sistema y ya no aparece ninguno en la tabla de plantilla..

Tabla 3.17: Escenario del Caso de Uso CU-14: Eliminar a todos los jugadores.

Caso de Uso CU-15: Gestionar el color de los jugadores		
Precondición		Haber iniciado la aplicación y encontrarse en la escena Ajustes.
Paso	Control	Descripción
1	Usuario	Indica que quiere gestionar el color de las equipaciones.
2	SISTEMA	Solicita al actor que modifique los colores a su gusto.
3	Usuario	Modifica los colores de ambos equipos hasta que está conforme.
4	SISTEMA	Solicita al actor que pulse el botón “Guardar cambios”.
5	Usuario	Pulsa el botón “Guardar cambios”.
6	SISTEMA	Guarda los nuevos colores en el sistema. El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario solicita canelar el proceso.	
	SISTEMA	Informa al actor de la cancelación, regresando al menú principal.
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		El color de las equipaciones de los jugadores es modificado, pudiendo ver los cambios en la visualización previa de ajustes, y en la escena jugadas. Los colores se mantienen hasta el siguiente cambio.

Tabla 3.18: Escenario del Caso de Uso CU-15: Gestionar el color de los jugadores.

Caso de Uso CU-16: Cambiar idioma de la aplicación		
Precondición		Haber iniciado la aplicación y encontrarse en el menú principal.
Paso	Control	Descripción
1	Usuario	Indica que quiere cambiar el idioma de la aplicación.
2	SISTEMA	Solicita al actor que seleccione el nuevo lenguaje en el que mostrar la aplicación.
3	Usuario	Selecciona el lenguaje que quiere desde el desplegable.
4	SISTEMA	Actualiza el idioma de la aplicación.
6	SISTEMA	Guarda la nueva elección de idioma en el sistema. El caso de uso finaliza con éxito.
Extensiones		
Paso	Control	Descripción de la extensión
3.a	Si el usuario solicita canelar el proceso.	
	SISTEMA	Informa al actor de la cancelación, ocultando el desplegable sin cambiar el idioma.
	salir	Se abandona el caso de uso, quedando sin efecto.
3.b	Si el usuario selecciona el lenguaje actual.	
	SISTEMA	Informa al actor de la cancelación, ocultando el desplegable sin cambiar el idioma.
	salir	Se abandona el caso de uso, quedando sin efecto.
Postcondición		El lenguaje mostrado de todos y cada uno de los textos de la app es modificado al lenguaje elegido. El idioma se mantiene hasta el siguiente cambio.

Tabla 3.19: Escenario del Caso de Uso CU-16: Cambiar idioma de la aplicación.

3.5. Modelo de dominio

En la figura 3.2, se muestra el modelo de dominio. En éste, se encuentran las clases identificadas a nivel de análisis y las relaciones entre ellas.

El modelo de dominio es una representación conceptual de las clases que conforman un software en específico. Se describen las entidades que las conforman, así como sus atributos y las relaciones con otras entidades.

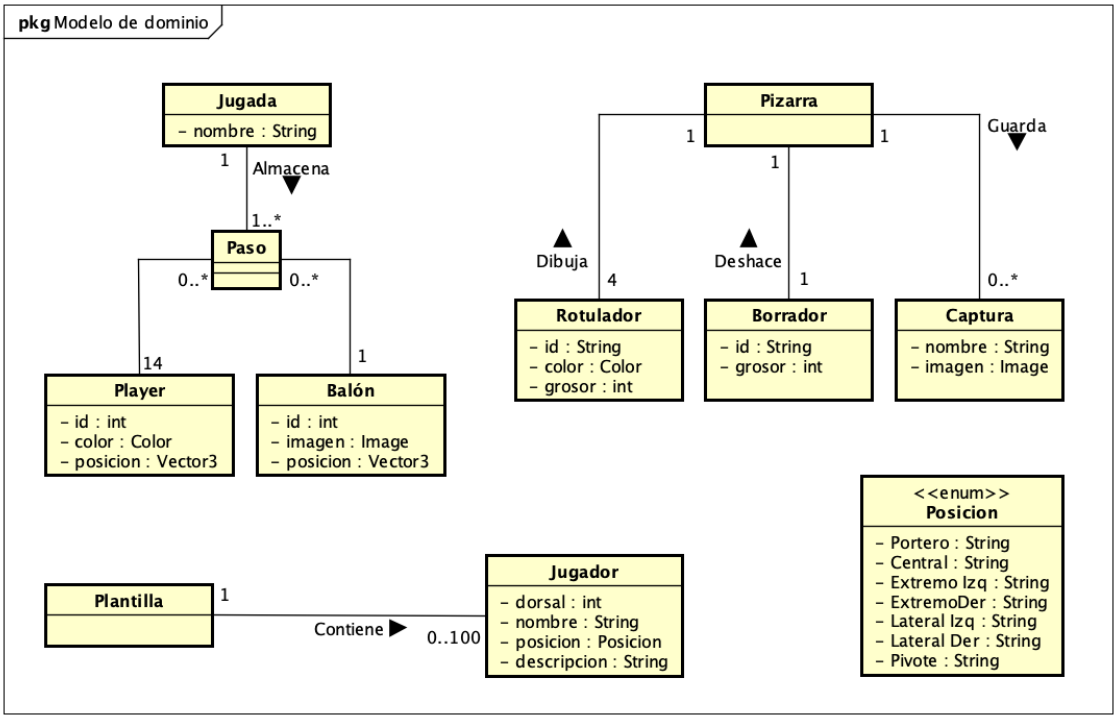

Figura 3.2: Modelo de Dominio.

3.6. Flujo de la App

Con la finalidad de tener una visión clara del flujo de acciones que se pueden realizar en la App, se presenta la figura 3.4, el diagrama de proceso (desde el menú principal). Como ya comenté en el subcapítulo 5.2.2, éstos permiten visualizar mediante un diagrama de actividades los procesos elementales de negocio, es decir, plasmar los requisitos como acciones que realizan los usuarios. Para ayuda del lector, se incluye una pequeña leyenda, figura 3.3, que se tomará como base para explicar los conceptos.

En la figura 3.3 se pueden observar 5 elementos que conforman el diagrama de proceso, y cada uno presenta una nota rosa correspondiente a su nombre, tomándolo así como guía.

- **1. Nodo inicial.** Desde el cuál parte el flujo, es la acción que inicia el sistema.
- **2. Nodo final.** Independientemente de las tareas desarrolladas, todo flujo debe terminar, y debe hacerlo en este símbolo.
- **3. Acción.** Las acciones son un conjunto de operaciones que una vez se inician, terminan. Son aquellas que brindan funcionalidad al sistema, y en la figura 3.4 se diferencian por colores para facilitar el seguimiento: rojo para salir de la aplicación, naranja para indicar que nos encontramos en el menú, morado para navegar hacia este, azul para los casos de uso principales, verde para los casos de uso que los extienden y amarillo (el único por defecto) para el resto de las acciones.
- **4. Nodo de unión o decisión.** Estos nodos cumplen dos funcionalidades. O bien se usan para decidir qué opción de entre las posibles (la pregunta es planteada en la flecha que llega al nodo) y así continuar, o bien

para recoger flechas que llegados a ese punto (nodo) continuarían de una misma manera, simplificando así el diagrama disminuyendo el número de líneas presentes.

- **5. Flecha de flujo** Unen actividades y nodos, permitiendo y aclarando, la navegación por el sistema.

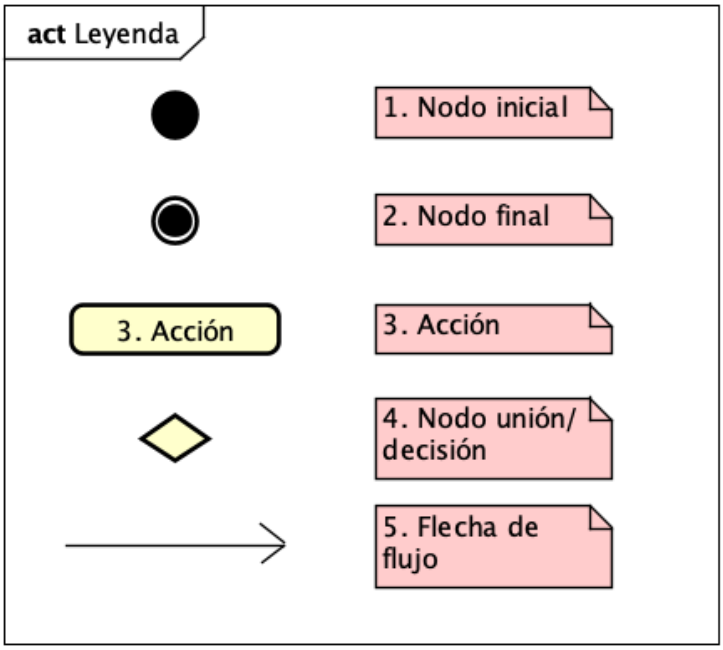

Figura 3.3: Leyenda del diagrama de proceso.

Finalmente, se muestra el diagrama del menú, figura 3.4, seguido de los diagramas de los demás casos de uso de la aplicación, accesibles desde éste.

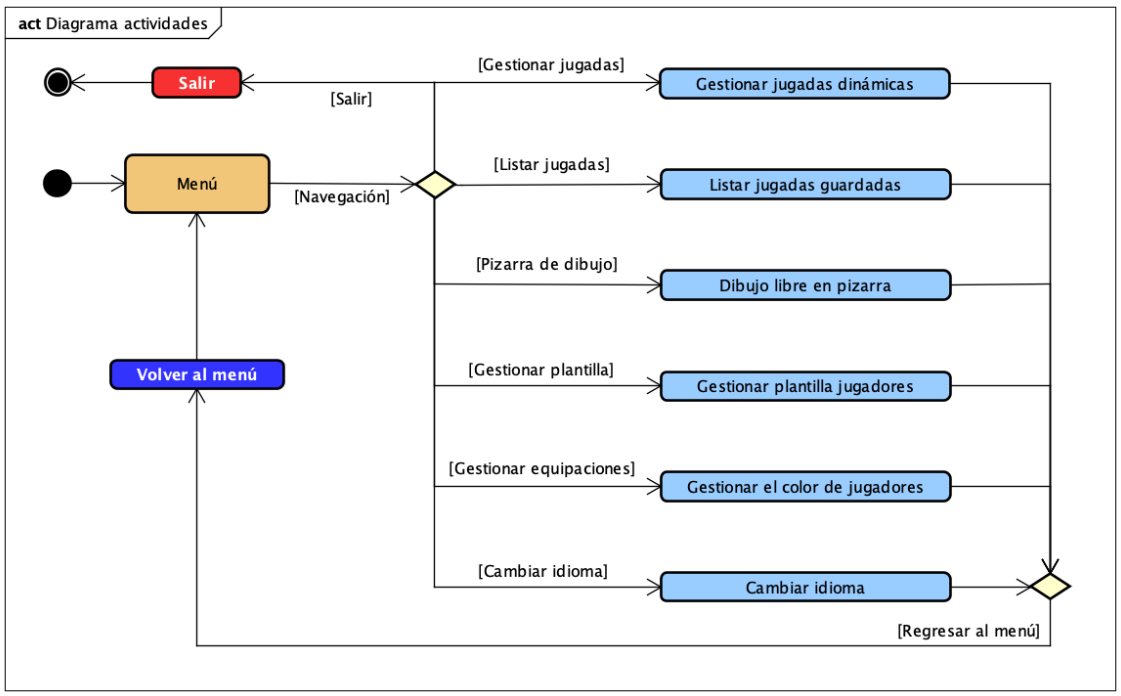

Figura 3.4: Diagrama de proceso desde el menú principal.

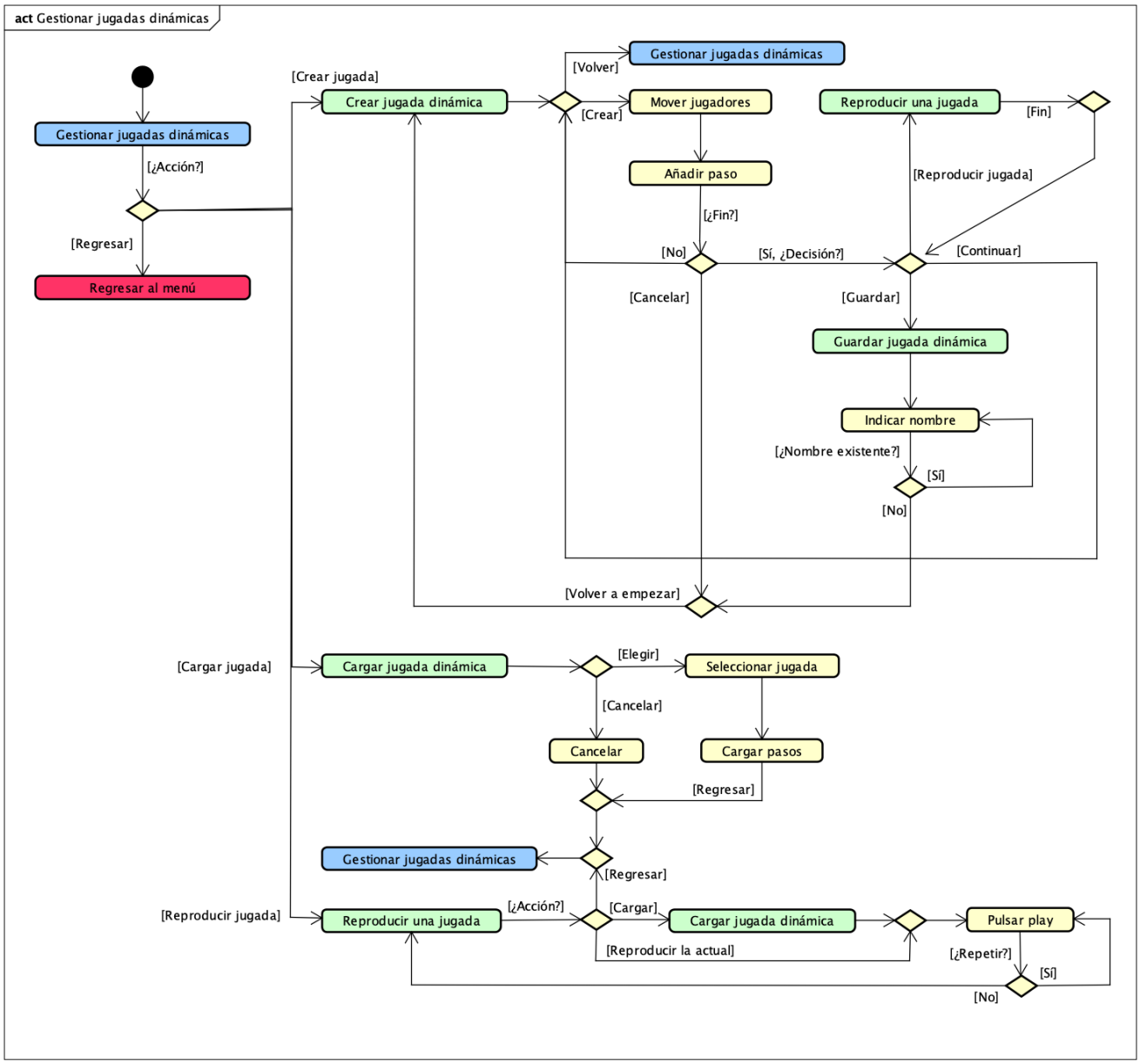

Figura 3.5: Diagrama de proceso del caso de uso: Gestionar jugadas dinámicas

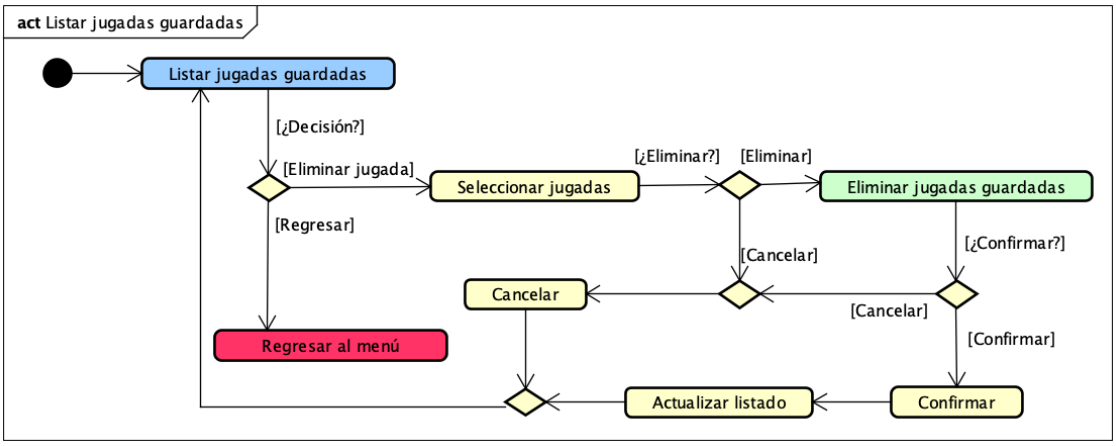

Figura 3.6: Diagrama de proceso del caso de uso: Listar jugadas guardadas

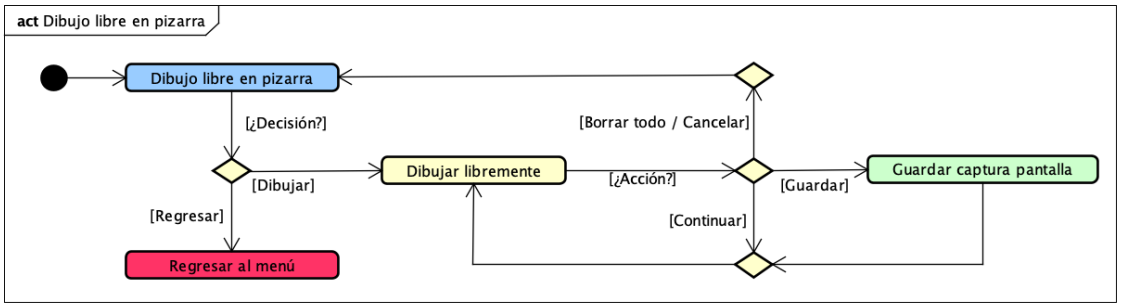

Figura 3.7: Diagrama de proceso del caso de uso: Dibujo libre en pizarra

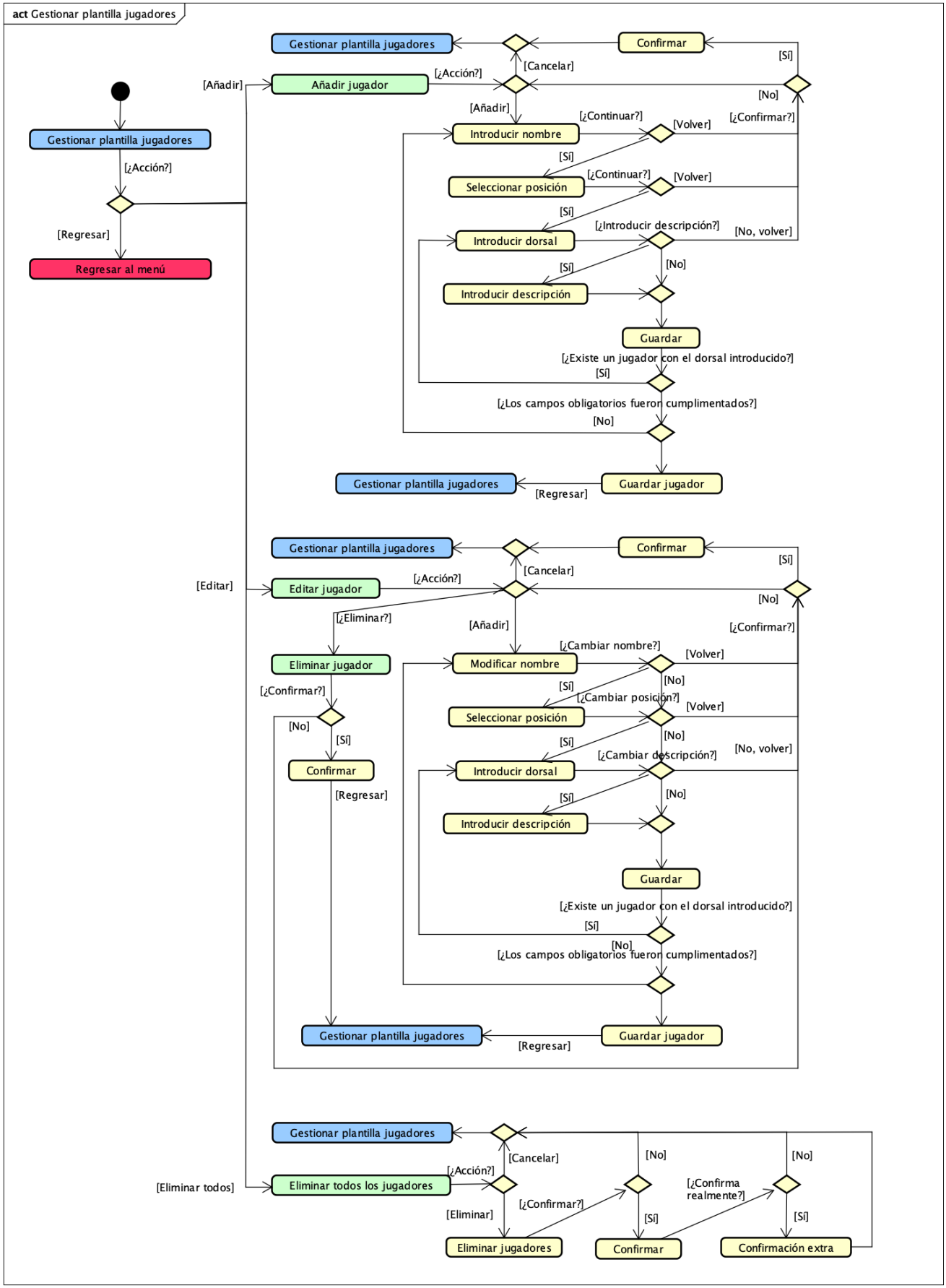

Figura 3.8: Diagrama de proceso del caso de uso: Gestionar plantilla jugadores

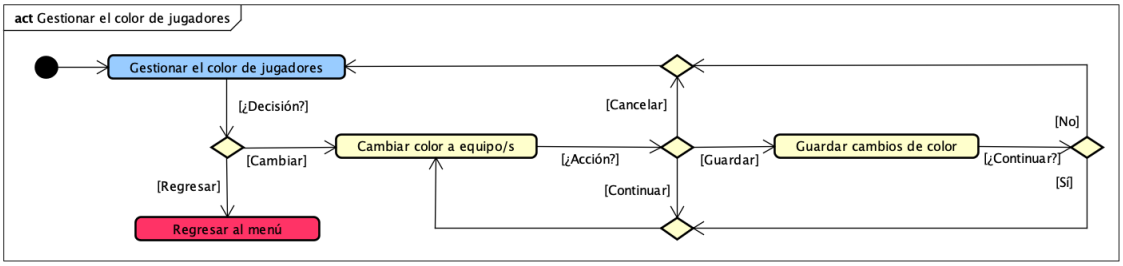

Figura 3.9: Diagrama de proceso del caso de uso: Gestionar el color de jugadores

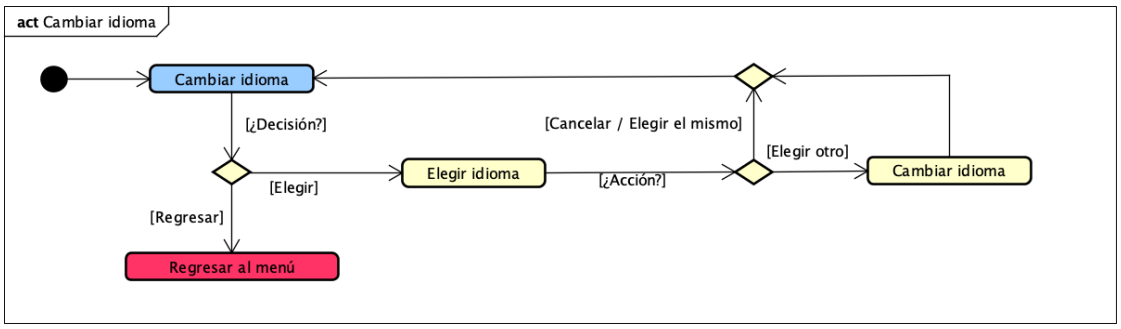

Figura 3.10: Diagrama de proceso del caso de uso: Cambiar idioma

Capítulo 4

Diseño

Tras el Análisis, sección 3, se establece el Diseño de la aplicación. En este capítulo por tanto se detallarán decisiones de diseño software así como la arquitectura del sistema, y los prototipos de la interfaz de usuario.

4.1. Arquitectura

Para este proyecto, que no requiere internet, se emplea la arquitectura de tres niveles 4.1. No habiendo despliegue, cliente ni servidor al no ser necesarios.

Se organiza por tanto el código en tres capas, repartiendo claramente las funciones:

- **Presentación.** Destinada a la interfaz de usuario.
- **Negocio.** Donde se encuentra el modelo de negocio.
- **Persistencia.** Donde se establece el almacenamiento.

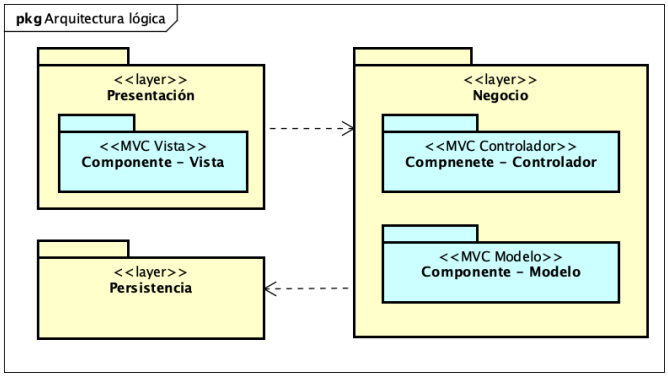

Figura 4.1: Arquitectura lógica.

4.2. Patrones

Los patrones de diseño representan soluciones generales a problemas repetitivos de software. Son una serie de consejos y principios que ayudan a organizar el código para que sea mantenible, ofreciendo a su vez una mayor calidad final. Se exponen a continuación los empleados.

4.2.1. Patrón arquitectónico MVC

Un patrón arquitectónico o arquetipo, tiene por finalidad estructurar el software, resolviendo problemas propios del desarrollo siguiendo unos criterios comunes para facilitar la tarea, y propiciar la mantenibilidad del código.

En este proyecto, me decanto por usar el patrón MVC (Modelo Vista Controlador), visto por primera vez durante el grado, en la asignatura Interacción Persona Computadora [18], es un patrón muy usado en la actualidad pese a haber surgido en los años 70. De hecho, este factor habla mucho y bien sobre su valía, y es más que válido para desarrollar en conjunto con Unity.

Se caracteriza por separar los datos, la interfaz y la lógica, de una aplicación en componentes, que a modo de resumen son los siguientes:

- **Modelo.** Contiene los datos de la aplicación, su persistencia, y la lógica que se puede aplicar para modificarlos.
- **Vista.** Se encarga de la interfaz y la interacción con el usuario, incluye los aspectos visuales que permiten realizar cambios en el sistema.
- **Controlador.** Gestiona la información que puede circular entre el Modelo y la Vista, haciendo de intérprete de los datos para adecuarlos a ambas.

El resultado de separar estas responsabilidades, como expone [18] es el desacoplamiento entre el modelo y la vista, en los cuales se pueden introducir cambios de manera independiente. De esta forma, es posible reutilizar el modelo con diferentes vistas, las cuales pueden tener más de un modelo, siempre y cuando implementen su funcionalidad.

Los eventos y su interacción entre los componentes se puede apreciar en la figura 4.2, que marca el flujo a seguir para interactuar entre los mismos.

Modelo

Se debe encargar de:

- Definir la funcionalidad que puede realizar el sistema.
- Acceder y gestionar los datos.
- En caso de ser MVC activo, notificará a la vista de sus cambios, para que ésta se actualice automáticamente. Por tanto debe llevar un control de sus vistas.

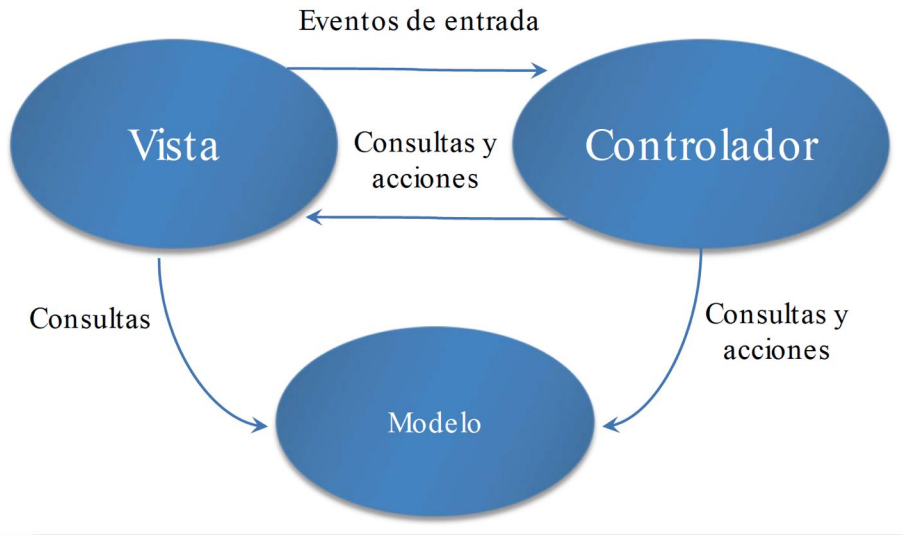

Figura 4.2: Eventos en el patrón MVC. Fuente: [18].

Vista

La vista es responsable de:

- Mostrar los datos presentes en el modelo.
- Tener asociado un controlador.
- Permitir las actualizaciones de la misma, garantizando su correcta visualización.

Controlador

Por su parte, el controlador debe:

- Gestionar los cambios producidos por el usuario.
- Tratar los eventos, ya sea actualizando las vistas o solicitar datos al modelo.

Estructura de control MVC

En general, el uso de MVC genera el siguiente flujo iterativo de acciones:

1. El usuario realiza una acción en la vista, modificándola, por ejemplo, moviendo los jugadores de posición, añadiendo un paso, o guardando una jugada.
2. El controlador recibe la notificación de la acción realizada, y la gestiona.
3. El controlador accede al modelo, y lo actualiza o modifica si es necesario.
4. La vista recupera los nuevos datos del modelo instada por el controlador, reflejando de esta manera los cambios en la interfaz.

4.2.2. Patrón Experto

Este patrón indica que sólo se debe asignar la responsabilidad a aquellas clases que tengan la información necesaria para manejarla. Permite:

- Encapsular la información.
- Bajo acoplamiento: Los módulos interactúan entre sí por medio de pequeñas comunicaciones, sin necesidad de conocer la implementación del otro módulo. Caracteriza a los sistemas robustos y fácilmente mantenibles.
- Alta cohesión: La medida en la que una clase realiza únicamente la tarea para la cual fue diseñada, no involucrándose en otras acciones. Es por tanto reutilizable.

4.2.3. Patrón Observador

Este patrón se aplica para el cambio de idiomas de la aplicación. Se basa en una dependencia 1..* de modo que cuando el objeto 1 se vea modificado (el lenguaje seleccionado) automáticamente los n textos de la aplicación se actualicen automáticamente.

4.2.4. GRASP: Polimorfismo

Los patrones GRASP en la Programación Orientada al Objeto, son los patrones generales para la asignación de responsabilidades. Entre ellos se encuentran los ya comentados Alta cohesión y Bajo acoplamiento. Aunque más que patrones son buenas prácticas, quisiera incluirlo en este apartado.

Entre ellos se encuentra el Polimorfismo, lo que permite enviar mensajes a las diferentes instancias de una misma clase. Los PreFabs de Unity, cumplen la función habitual del polimorfismo, esto se ve reflejado, por ejemplo, en los jugadores de cada equipo cuando se les induce a moverse por el campo al reproducir una jugada.

4.3. Diseño detallado de paquetes

A continuación se expone el diseño de paquetes detallado que conforma la aplicación final.

Primeramente, la figura 4.3, muestra la estructura de código típica de Unity (empleada en este proyecto), la cual no afecta a la arquitectura. Se hace hincapié en el paquete Assets, pues es el principal, donde radica la aplicación como tal. Los detalles de a qué corresponde cada carpeta de dicho paquete pueden consultarse en la sección 5.5.1.

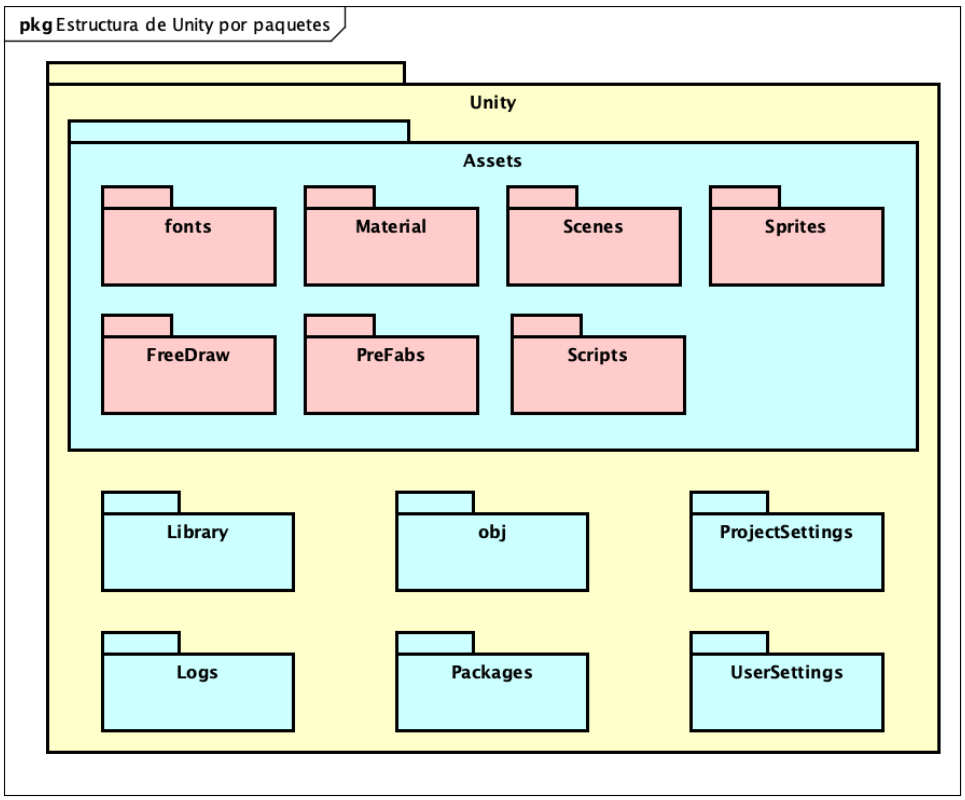

Figura 4.3: Estructura de Unity por paquetes.

Las siguientes figuras, muestran los paquetes en detalle, más importantes de la figura 4.3.

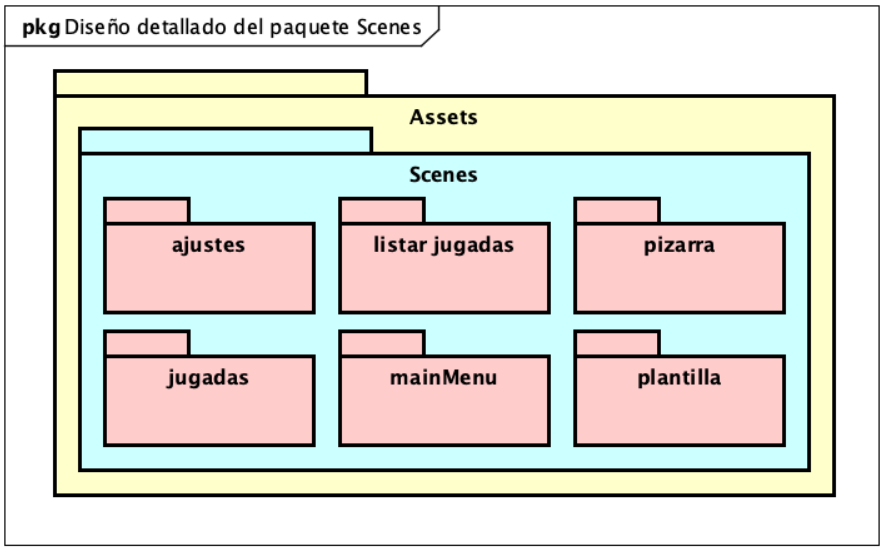

Figura 4.4: Diseño detallado del paquete Scenes.

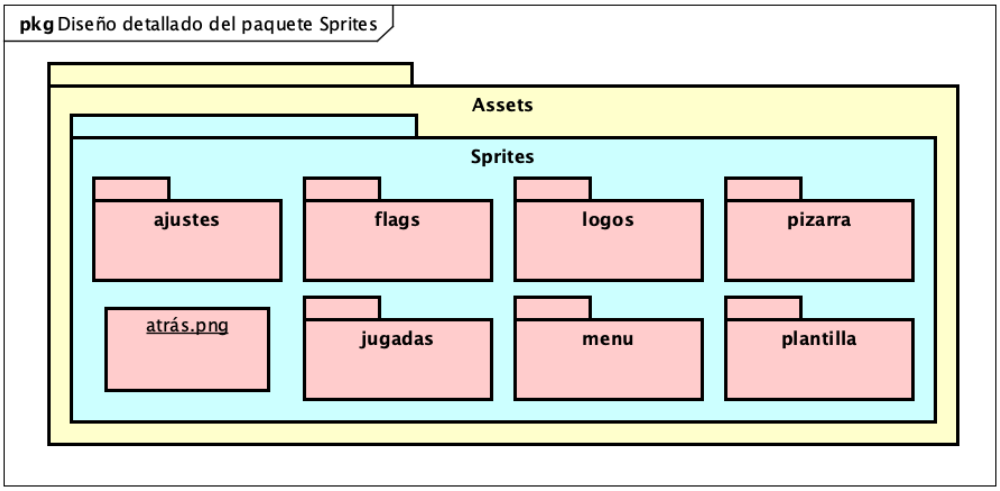

Figura 4.5: Diseño detallado del paquete Sprites.

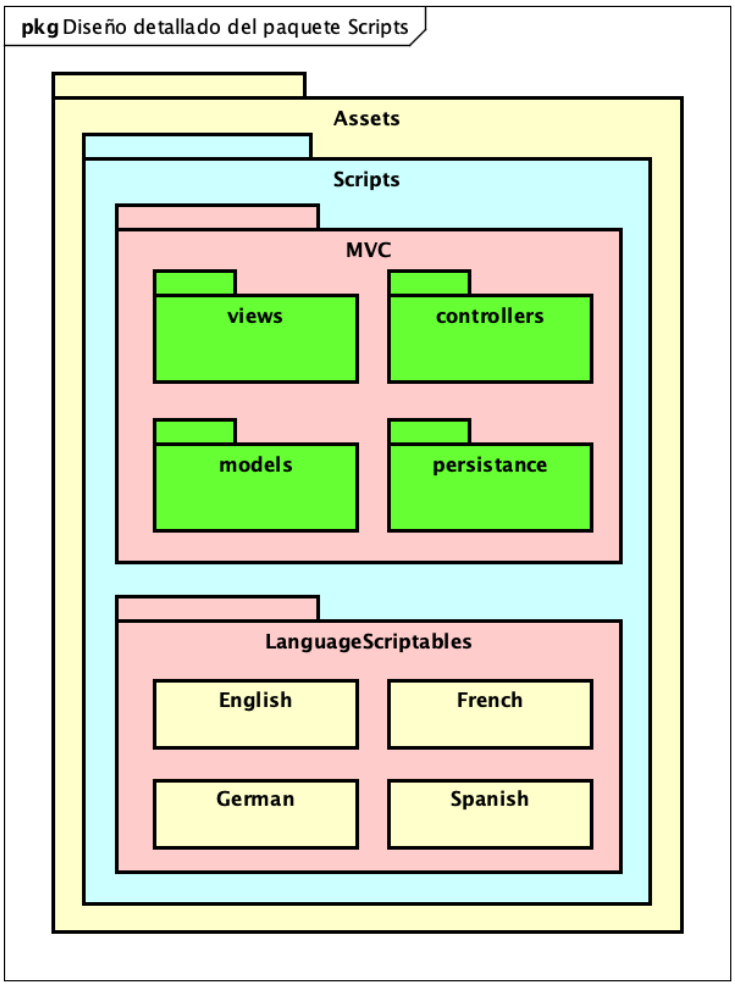

Figura 4.6: Diseño detallado del paquete Scripts.

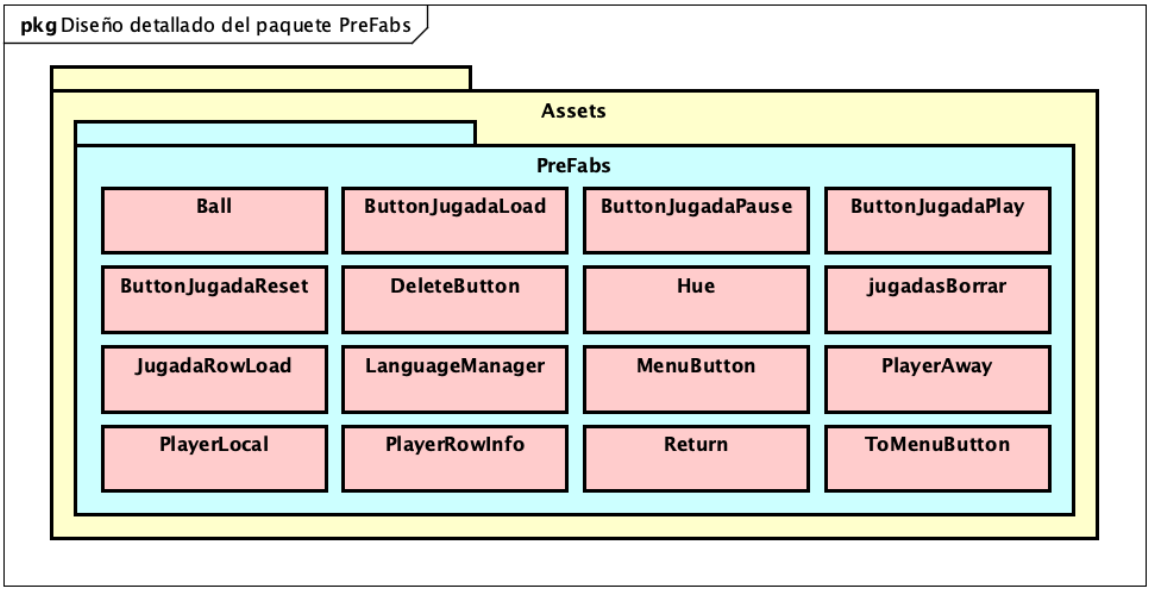

Figura 4.7: Diseño detallado del paquete PreFabs.

4.4. Diseño de la Base de Datos

Tras analizar los objetivos a cumplir, y formarme en Unity, rápidamente me percato de que no es necesario crear una base de datos para este proyecto, y que de hecho, sería contraproducente.

Las bases de datos en Unity suelen implementarse con SQLite (aunque la mayoría de juegos simples no las usan, como pude apreciar en mi búsqueda [34]), lo cual no suponía un problema puesto que es abordado en la carrera. Pero se toma esta decisión, de prescindir de base de datos debido a las siguientes razones:

- **Complejidad innecesaria.** Las acciones que requerirían almacenamiento persistente, teniendo en cuenta que no hay registro, son el guardado de jugadas, capturas de pantalla y jugadores. Debido a que no se trata información privada de los usuarios, no se requiere de una seguridad extra que aporta una base de datos.
  Los elementos que son almacenados son fácilmente exportables como GameObjects que son, y por tanto, su guardado es prácticamente inmediato. Las ediciones de jugadores se pueden realizar sobre el propio archivo, por lo que se simplifica el tratamiento de los ficheros.
  A fin de cuentas, encontrar una solución completamente válida para un problema, que lleva menos tiempo y simplifica todos los procesos, es admirada en cualquier empresa software.

- **Mantenibilidad.** Mantener una base de datos con el tiempo es una tarea que conlleva cierta dedicación. Teniendo en cuenta que para las acciones actuales la presencia de una base de datos no aporta grandes ventajas, es mejor mantener el software simple, prescindiendo de la gestión de una base de datos.

- **Compartir archivos.** Al no disponer de una base de datos, y sí de ficheros binarios corrientes, la compartición de jugadas y jugadores entre dispositivos es sumamente simple para el usuario final, no implicando más operaciones por su parte que copiar y pegar archivos.

4.5. Diseño de la Interfaz de Usuario

Este proyecto Unity tiene por finalidad su ejecución en Android, pero también contempla su uso en otros sistemas operativos, como son Windows y macOS. En cualquier caso, el rasgo principal que debe caracterizar a la app es la simplicidad de uso, una interfaz sencilla que permita a todos los usuarios, independientemente de sus aptitudes informáticas (**usabilidad**), manejar sin problemas todos los casos de uso a implementar.

Esta sección se basa en gran medida en la asignatura Interacción Persona Computadora [18], que aporta muchos de los conceptos clave necesarios, ya que de la interacción entre la persona y la máquina puede depender el éxito o fracaso de cualquier aplicación.

Por tanto, la interfaz se puede definir como aquella parte de la app que permite al usuario interactuar con la computadora, y a ésta informar de los cambios que se producen en el modelo 4.1. En la interfaz también se tiene en consideración al hardware, que en este caso sería o bien la pantalla táctil de una tablet (con su control táctil para todos los elementos, inlcuído el teclado) o ratón y teclado convencional si se dispone de la posibilidad inalámbrica, o bien, se ejecuta desde un ordenador.

El interés en el diseño de interfaces de usuario es muy alto, debido a que juega un papel fundamental en el ciclo de vida de una aplicación (el 50 %, según un estudio de Myers & Rosson en 1992). No es una cuestión trivial, puesto que las personas somos impredecibles, y acertar con un diseño agradable y acorde a la mayoría, no es ni mucho menos sencillo. Para ello se han de seguir unas **guías de diseño** que ayuden a enfocar correctamente el diseño de la interfaz, y siendo necesarias las **pruebas con usuarios reales** (no técnicos) para detectar cuanto antes dificultades de uso.

En los siguientes subcapítulos trataré de ampliar estos conceptos, exponiendo además el diseño de la interfaz de usuario del proyecto.

4.5.1. Usabilidad

Comentada anteriormente, la aplicación debe ser usable, que es la capacidad de un producto para alcanzar los objetivos marcados, en un contexto específico, con criterios de:

- **Eficacia.** Cumplir con los requisitos, de manera precisa.
- **Eficiencia.** Lograr los objetivos con una calidad notable, con respecto a los recursos empleados.
- **Satisfacción.** El usuario debe encontrarse cómodo ante el uso de la aplicación.

Para ello, no debe suponer un gran esfuerzo de aprendizaje para el usuario, manteniendo lo más simple posible cada aspecto. Esto se puede medir en función de si el usuario es capaz de deducir todas las funcionalidades de un sistema, simplemente explorando. Y de la misma manera, mantener al mínimo la necesidad de recuerdo, es decir, que el usuario pueda permanecer grandes periodos de tiempo sin utilizar la aplicación y que esto no repercuta en ser capaz de recuperar su nivel. La eficiencia, en éste ámbito, trata sobre lograr el mínimo esfuerzo a dedicar por parte del usuario.

Cabe recordar que todos podemos errar como usuarios, el tratamiento de errores humanos juega un papel fundamental en cualquier sistema informático. Por ejemplo, si el usuario quiere regresar al menú sin haber guardado una jugada, debe mostrarse un aviso por pantalla que le informe de ello, quedando en su mano salir sin guardar, pero habiendo evitado el despiste. Se trata de evitar riesgos, pese a que pueda entorpecer el uso normal de la aplicación. La tolerancia al error estará en mano del usuario, ya que será quien tenga siempre la última palabra.

Finalmente, la satisfacción por parte del usuario es clave, si no se siente a gusto, no seguirá usando la aplicación, por buena que sea. Es problemático, ya que no es algo tan objetivo como poder realizar todas las tareas o hacerlas eficientemente y sin posibilidad de errores; lo que es agradecido por algunos, puede ser aborrecido por otros, por lo que deben evitarse extremos, y medir la satisfacción mediante cuestionarios. Toda retroalimentación es bien recibida, y debe ser analizada para avanzar en el camino correcto.

Con todo ello, podemos señalar a la interfaz como la encargada de mostrar el modelo del sistema (un modelo completo y correcto del funcionamiento) de una forma adecuada al usuario, para que lo entienda correctamente. Para lograrlo, se emplean los **principios de diseño**, recomendaciones generales de diseño basados en la experiencia, y en la forma de pensar, valorando especialmente:

- **Comprensión intuitiva.** La percepción de la interfaz guía al usuario, en base a su experiencia previa. Un ejemplo informático sería el uso de logos bien conocidos, como el de una papelera para eliminar ficheros, una barra para visualizar elementos de una lista, o los iconos de un reproductor de vídeo, todos comparten el mismo patrón; ser sumamente conocidos y extendidos, de manera que sean fácilmente reconocibles.
- **Visibilidad.** Todo aquello que se pueda controlar debe quedar al alcance del usuario. Aquello no presente, como puede ser el manejo mediante gestos, no resulta intuitivo al usuario, y por tanto, requiere de un esfuerzo adicional para su aprendizaje. El usuario debe tener claro qué puede hacer, y cuál es el cambio que produce en cada momento, por lo que se ha de mantener informado al usuario para que sea consciente de la actividad que está realizando y a cuales podría avanzar.
- **Retroalimentación.** Las acciones realizadas por el usuario deben mostrar un efecto visible, e inmediato (o al menos, un indicador de carga si la espera es mayor a 1 segundo). Generalmente será información visual, pero también puede ser sonora o táctil. En este TFG las respuestas serán visuales, ya que la información sonora (pese a ser una ayuda para personas con discapacidades visuales), generalmente es un inconveniente: entornos ruidosos que impiden su apreciación, situaciones en las que el ruido ambiente no es permitido, personas con dificultades auditivas...
- **Consistencia.** Aquellos aspectos similares deben actuar de manera similar, y aquello que sea diferente, no debe parecerse. Esto es un claro ejemplo de las acciones destructivas (borrado de ficheros, pérdida del progreso) o de navegación en la aplicación, en las primeras, el color rojo nos indica alerta y el verde seguridad; mientras que en la navegación por la interfaz, es aconsejable emplear un mismo indicador, posicionado siempre en la misma zona (superior izquierda, preferiblemente).

El usuario se comunica con la máquina mediante dispositivos apuntadores (en este caso el control táctil o mediante teclado y ratón). Conociendo cuales son los métodos de los que dispone el usuario, se puede diseñar de una manera más acorde. En este caso, el dispositivo objetivo es una tablet común de 10.1", un dispositivo relativamente pequeño, por lo que hay que prestar suficiente espacio entre acciones, para evitar pulsaciones erróneas, de la misma manera que se deben agrupar las funcionalidades similares (minimizando el movimiento), y alejando aquellas acciones que pueden ser peligrosas como el borrado de elementos.

Dada la naturaleza del proyecto, los menús desplegables no parecen ser necesarios, y atendiendo a la ley de Fitts se descartan finalmente debido a la complejidad que representan, siendo más lentos (en comparación a otros métodos), y no olvidando la alta probabilidad de equivocarse eligiendo otro submenú.

Volviendo a los errores, se trata de poner especial atención a los lapsus y deslices, es decir, evitar que el usuario se equivoque limitando las acciones peligrosas, ya que los errores de concepto que pueda tener el usuario son más difíciles de corregir. Deshabilitando funcionalidades (o más bien, los botones que las permiten) cuando no estén disponibles, disuade errores y guía en cierta parte al usuario, esto se ve claramente con las jugadas dinámicas, al no poder reproducirlas mientras no haya pasos.

Otra medida tomada en cuenta fue respetar el trabajo del usuario, procurando no borrar la información introducida mientras no lo solicite, como es el caso del guardado temporal de campos de texto, en la creación de un jugador en plantilla, mientras permanezca en dicha escena.

4.5.2. Guías de diseño

Son una serie de ideas que se pueden aplicar a un producto para facilitar su diseño, marcando los pasos que se pueden y no se pueden dar. No hay que olvidar que son plantillas generalistas, y que no resuelven todos los problemas. Algunas de las consideraciones tenidas en cuenta son:

- **Importancia de los colores.** No es recomendable mostrar demasiados colores en una sola pantalla, y si por algún casual deben aparecer, no han de aparecer muy juntos. Por otra parte deben mantenerse tonos similares en toda la aplicación.
- **Colores "conocidos".** Es recomendable el uso del color rojo para tareas de alerta, cancelar o salir, y el verde para confirmaciones y acciones no destructivas. Estos colores son reconocidos por cualquier persona en la sociedad ya que son reconocibles desde otros ámbitos, como pueden ser los semáforos.
- **Texto.** La importancia del texto es crucial. Debe mostrarse información escrita con un tamaño adecuado de letra, y no debe ser muy extenso, disponer de más de 60 caracteres de ancho, es considerada una mala práctica.
- **Uso de menús.** Los menús no debe ser muy extensos siguiendo la ley Hick-Hyman. Si se presentan demasiadas opciones u ocupan un espacio excesivo, es muy probable que abrumen al usuario.
- **Manejo de acciones.** Aquellas acciones peligrosas, tales como los borrados, deben permanecer alejadas de otras para evitar así llevarlas a cabo por un lapsus.
- **Misma disposición.** Recientemente, con la llegada de los smartphones, los cuales incorporan acelerómetros que le permiten estar al tanto de su orientación, adecuando la visualización para que el usuario nunca vea algo del revés. Sin embargo, esto genera problemas en el diseño de interfaces. En este caso, puesto que la rotación automática no tiene mucho sentido (y menos cuando se puede usar la aplicación en dispositivos sin estas características), se prescinde de ella mostrándose siempre en la misma posición (horizontal) y en una disposición en particular para respetar las fundas comunes de tablet que tengan un atril.

4.5.3. Diseño de la Interfaz de Usuario

En este subcapítulo se expondrán, por escenas/ funcionalidades, los prototipos diseñados como base para la implementación de la interfaz, destacando algunas de las decisiones consideradas. Como prototipos, no se corresponden totalmente con la versión final implementada.

Se debe tener en cuenta que el dispositivo objetivo es una tablet Android de 10.1", las más comunes. Los prototipos son diseñados de manera digital porque permite rectificar más fácilmente, y las imágenes usadas podían llegar a usarse en la implementación, como ocurrió.

Menú principal

El menú inicial trataba ser simple y mostrar los elementos de manera clara y ordenada. Es sustituido debido a que el diseño no se asemeja a los estándares actuales, más modernos con logos simples.

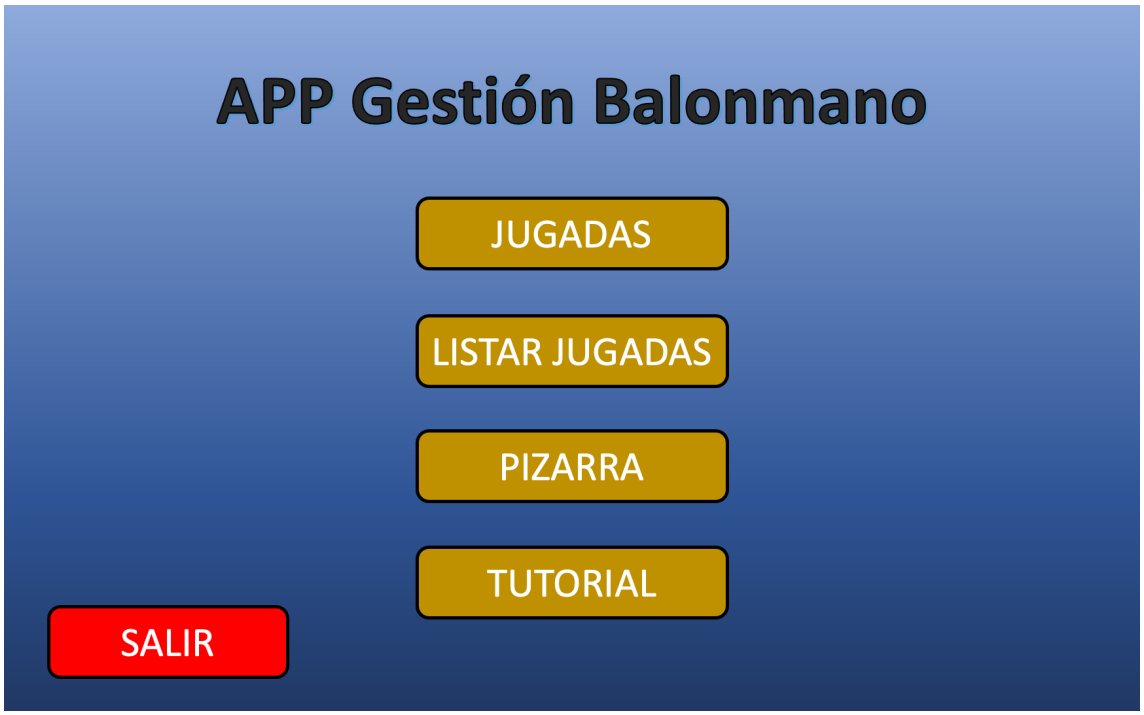

Figura 4.8: Prototipo inicial del menú principal.

Por otra parte quisiera reseñar el diseño del icono de la app, que será una aspecto visual sólo apreciable en el dispositivo, antes de lanzar la app.

Figura 4.9: Prototipo de icono de la aplicación.

Jugadas dinámicas

El punto central de las jugadas dinámicas, figura 4.10, es lograr una correcta visualización de los múltiples botones, que se disponen por los bordes, y poder movilizar correctamente los jugadores. La disposición inicial de los jugadores en el campo es por tanto un aspecto clave, ya que no tiene mucho sentido que todos partan desde la zona de banquillo. El uso habitual involucrará a la totalidad de los jugadores, si estos están dispuestos en el campo en una formación general, involucra menos movimientos, o más cortos, por parte del usuario.

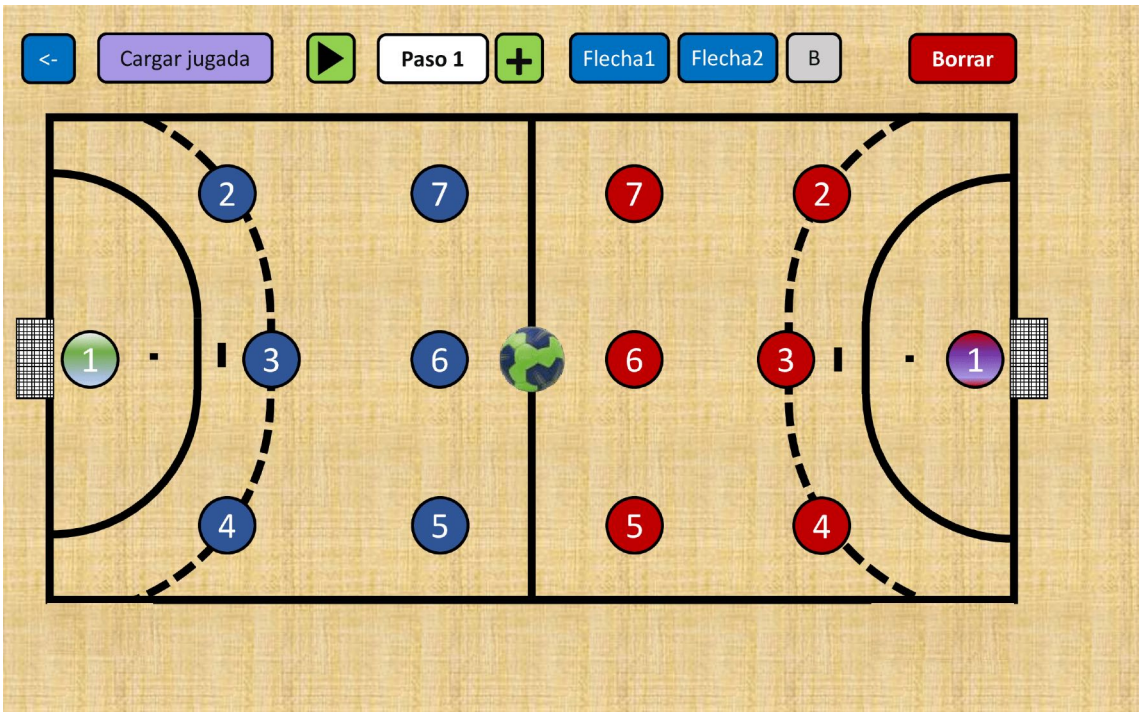

Figura 4.10: Prototipo de jugadas dinámicas.

El diseño de la imagen que representa el balón (figura 4.11), tiene por finalidad su correcta visualización en el terreno de juego, no confundiéndose con los jugadores. Para ello, se elige una imagen de balón genérico con dos colores, en forma de espiral, ya que el color contrasta notablemente con el del terreno de juego. Como es un elemento clave del juego, se representa con un tamaño ligeramente mayor que el de los jugadores, para que pueda apreciarse en todo momento.

Figura 4.11: Prototipo del balón de juego.

Pizarra

El prototipo de la pizarra libre, figura 4.12, ha sido el único que no ha sufrido modificaciones en su implementación final. Se predisponen los elementos de dibujo al lateral derecho dejando así más area de dibujo. El botón de Borrar TODO estará protegido, por lo que no ocurrirán lapsus pese a encontrarse cerca de los útiles de dibujo.

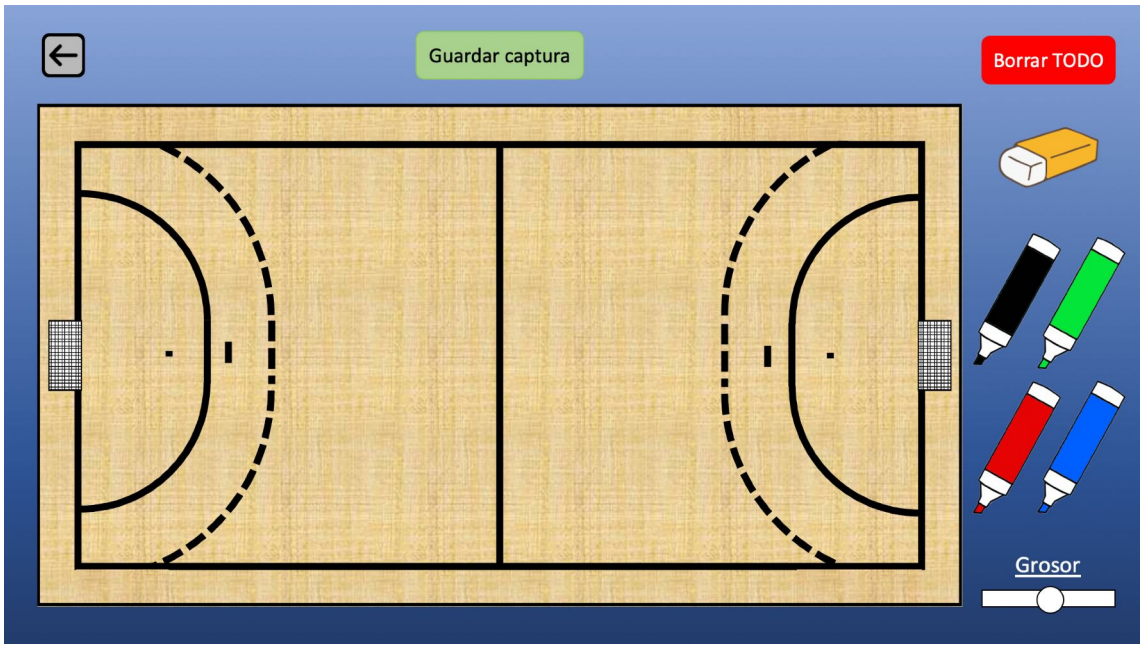

Figura 4.12: Prototipo de la pizarra de dibujo libre.

En la figura 4.13, se puede apreciar una de las pantallas de aviso al usuario, en este caso pregunta por borrar todo aquello que se dibujase. El resto de pantallas de aviso son muy similares, por lo que no expondré todas ellas debido al gran número de estas.

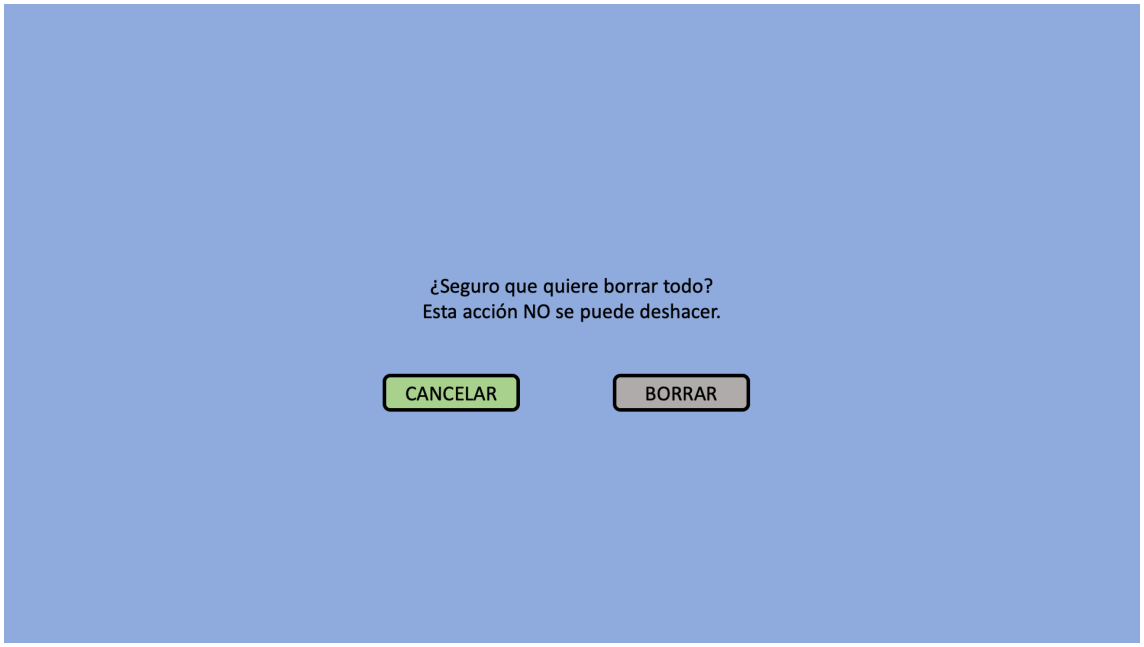

Figura 4.13: Prototipo de aviso de borrado en pizarra.

Lista de jugadas

Para poder eliminar las jugadas que se hayan diseñado, ya sea por error u otras consideraciones, se presentan en forma de listado como puede apreciarse en la figura 4.14. Con una cantidad mínima de botones se logra seleccionar sólo aquellas que se quieren borrar, todas, o desmarcarlas. Borrando así la cantidad deseada por el usuario.

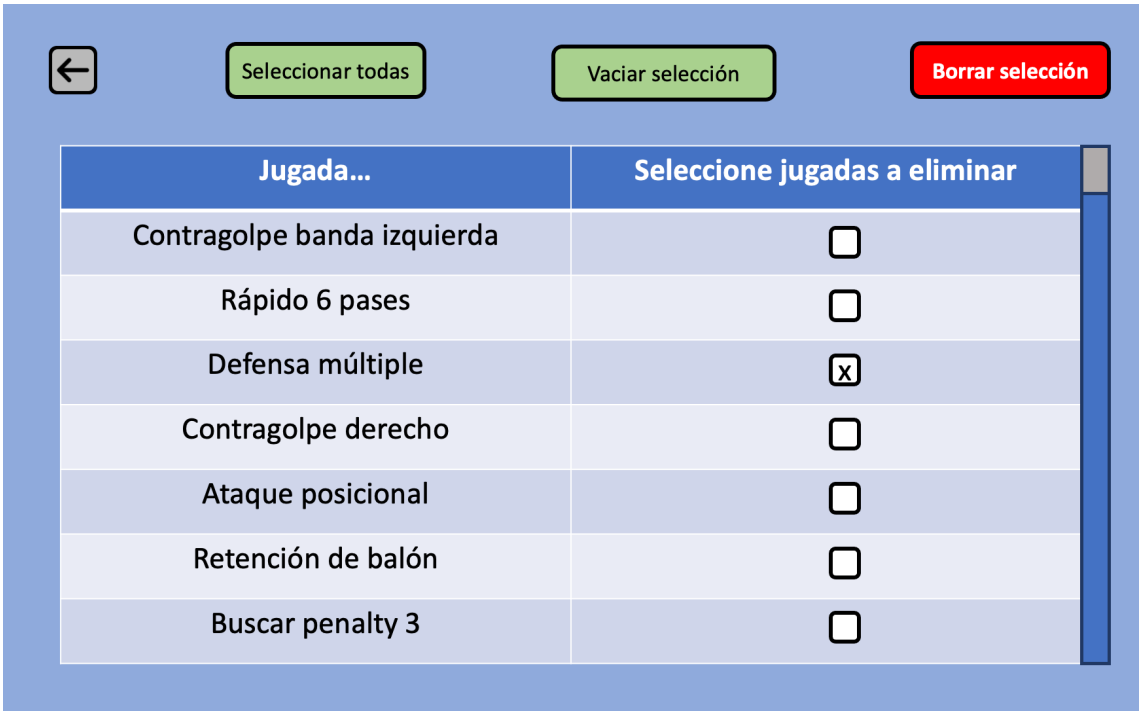

Figura 4.14: Prototipo de un listado de jugadas.

Plantilla de jugadores

De manera análoga, en forma de tabla se presentan los jugadores de la plantilla del club, figura 4.15, con los campos mínimos que ayuden a identificarlo, primando el campo de Apuntes, pues es lo realmente útil para el entrenador, en conjunto con la posición que desempeña.

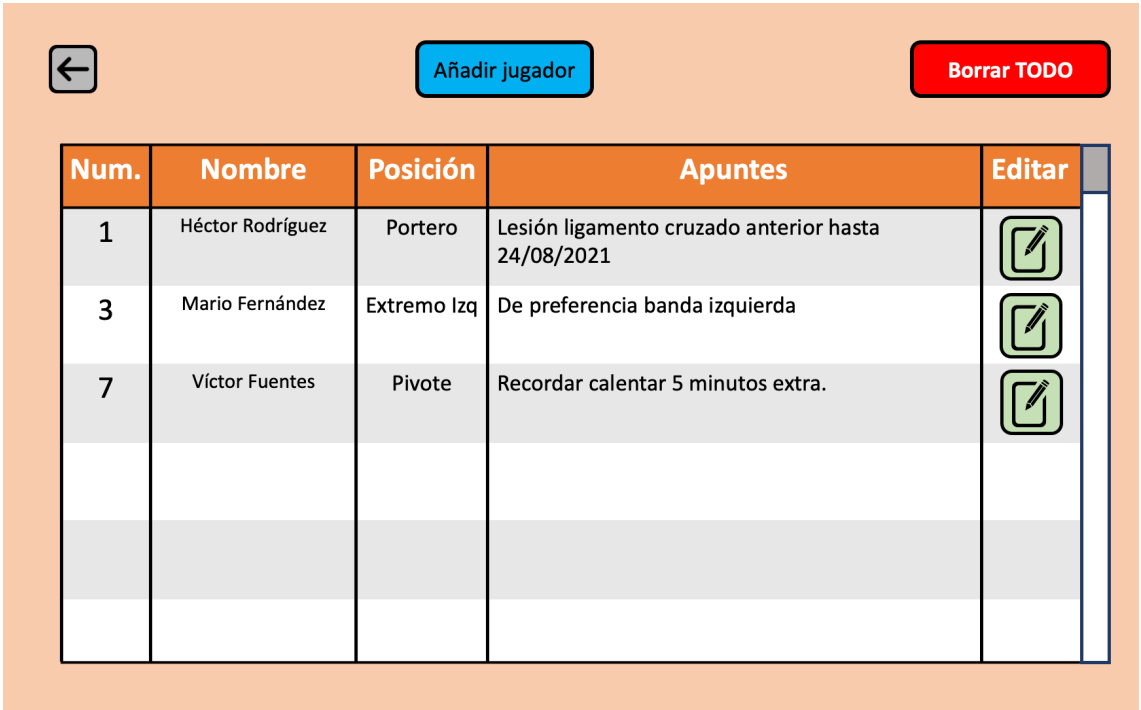

Figura 4.15: Prototipo de un listado de jugadores.

En la figura 4.16, se aprecia el diseño del menú de creación y edición de los jugadores. Al crearse no se puede borrar, lógicamente, por lo que el botón estará deshabilitado. El espacio entre los botones se establece para los mensajes de error que puedan surgir, como por ejemplo, no poder guardar un jugador debido a que ya existe otro con dicho número.

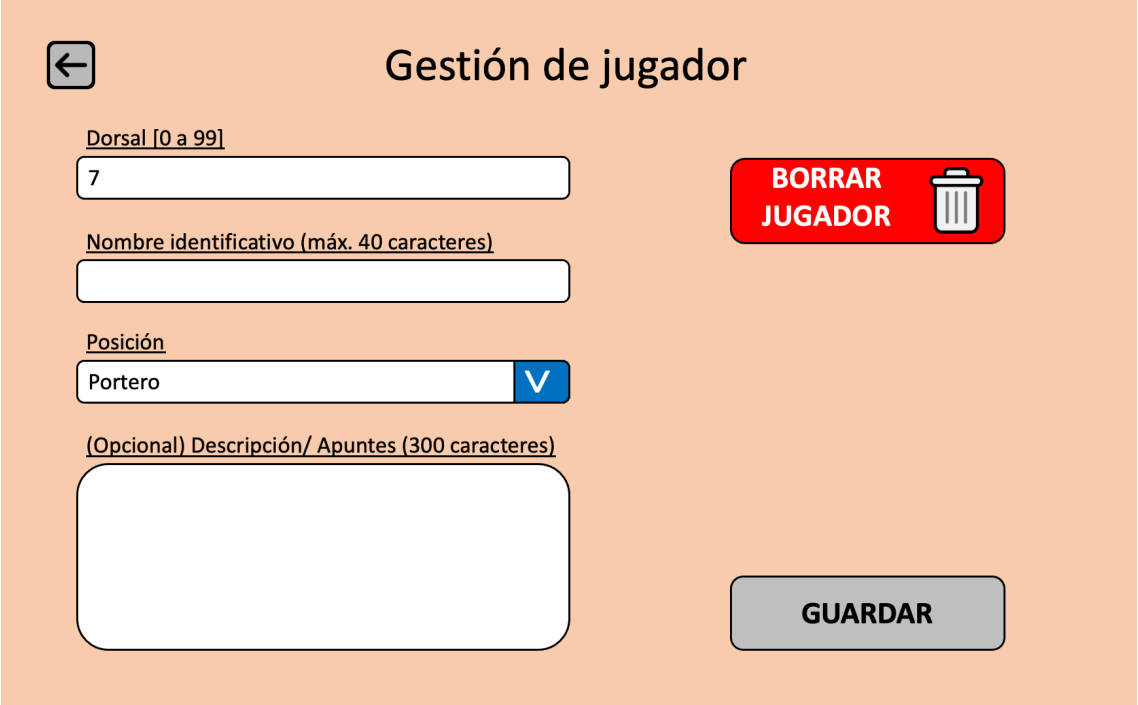

Figura 4.16: Prototipo de creación/ edición de un jugador.

Ajustes

El prototipo de ajustes inicial, figura 4.17, muestra 8 equipaciones posibles tanto para el equipo propio como para el rival, con una visualización previa de cómo se verían en el terreno de juego, y junto al balón. Con esta opción, si por ejemplo el azul era elegido para el equipo propio, quedaría imposibilitado para el rival, no pudiendo coincidir nunca.

Imágenes implementadas

Algunas de las imágenes empleadas en el prototipado, fueron descargadas desde repositorios de imágenes gratuitas, las cuales fueron aplicadas también a la interfaz. A continuación listo los enlaces a las mismas, para dar el crédito que merecen sus autores:

- Balón para el logo
- Icono de menú: Ajustes
- Icono de menú: Plantilla
- Icono de menú: Salir
- Banderas de países para idiomas

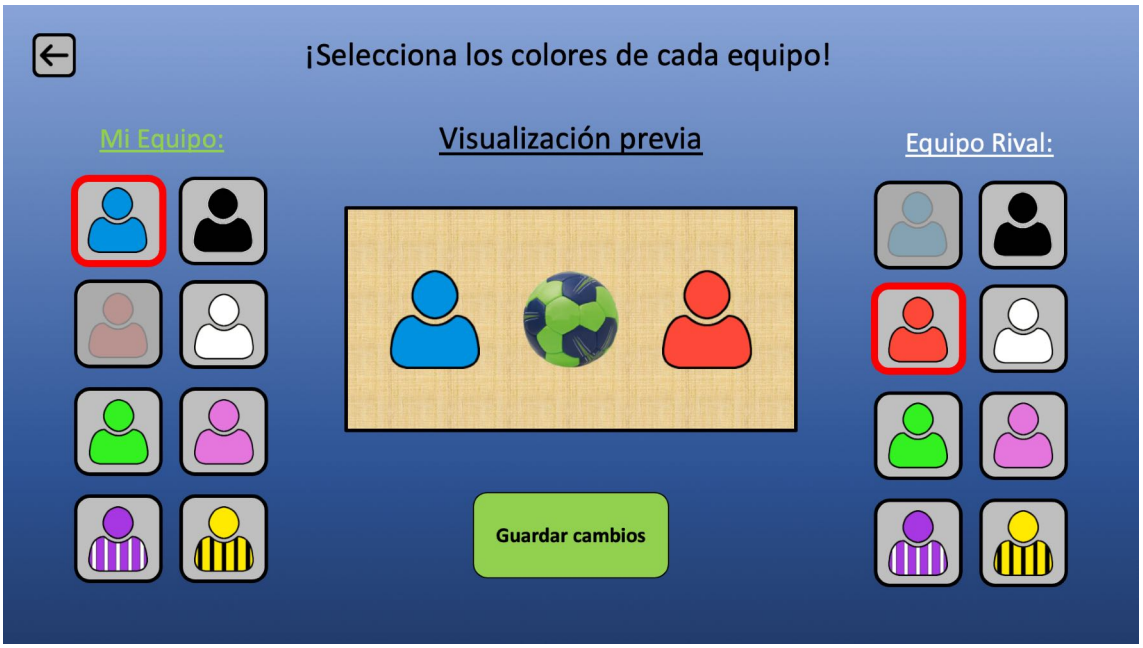

Figura 4.17: Prototipo de los ajustes de color de equipo.

- Balón de juego
- Rotuladores
- Borrador
- Edición de jugador en plantilla
- Papelera en gestión de jugador
- Ruleta de colores RGB

Evaluación de prototipos con usuarios

Como parte de las pruebas de usabilidad, se encuentra el análisis de los prototipos que serán implementados finalmente en la app. Analizando los siguientes puntos, se rediseñan los prototipos pudiendo así aplicarlos directamente al crear las interfaces.

- **Menú principal.** El menú propuesto en el diseño de interfaz de usuario luce antiguo, no es acorde a la época actual donde los logos sobresimplificados son preferidos.
- **Concordancia de colores y estilos.** Los colores de fondo de las diferentes pantallas han de ser los mismos, así como los colores empleados en botones con acciones similares. De igual manera se debe aplicar una fuente y tamaño unificado, ya sean títulos
- **Indicar rotulador en pizarra.** El usuario no tendría ninguna manera de distinguir cuál es el rotulador seleccionado, hasta que pinte algo. Se incluirá algún elemento gráfico que indique el seleccionado.
- **Colores en ajustes.** Tras revisar la propuesta de 8 colores fijos para los equipos, se decide que se realice mediante una ruleta RGB. Por sugerencia de la tutora: mayor versatilidad y margen de maniobra para personas con discapacidades visuales como el daltonismo.
- **Disposición de elementos en pantalla.** Reorganización de los componentes mostrados en la aplicación, como en la edición de jugador en plantilla, debido al mal aprovechamiento del espacio.

4.6. Realización en diseño de casos de uso

En esta subsección se detalla la comunicación entre las diferentes capas para realizar un caso de uso concreto.

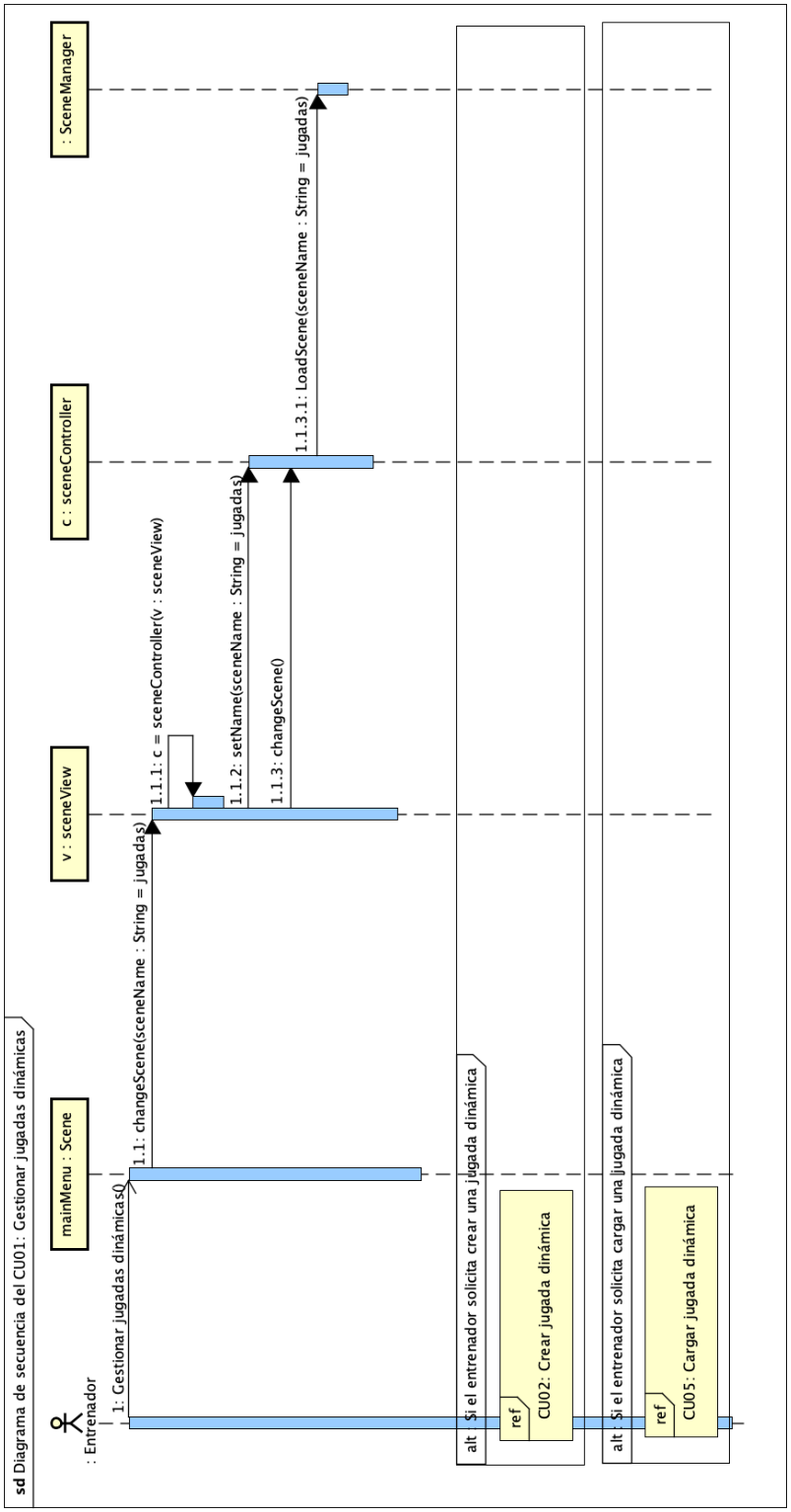

Figura 4.18: Realización en diseño del CU01: Gestionar jugadas dinámicas.

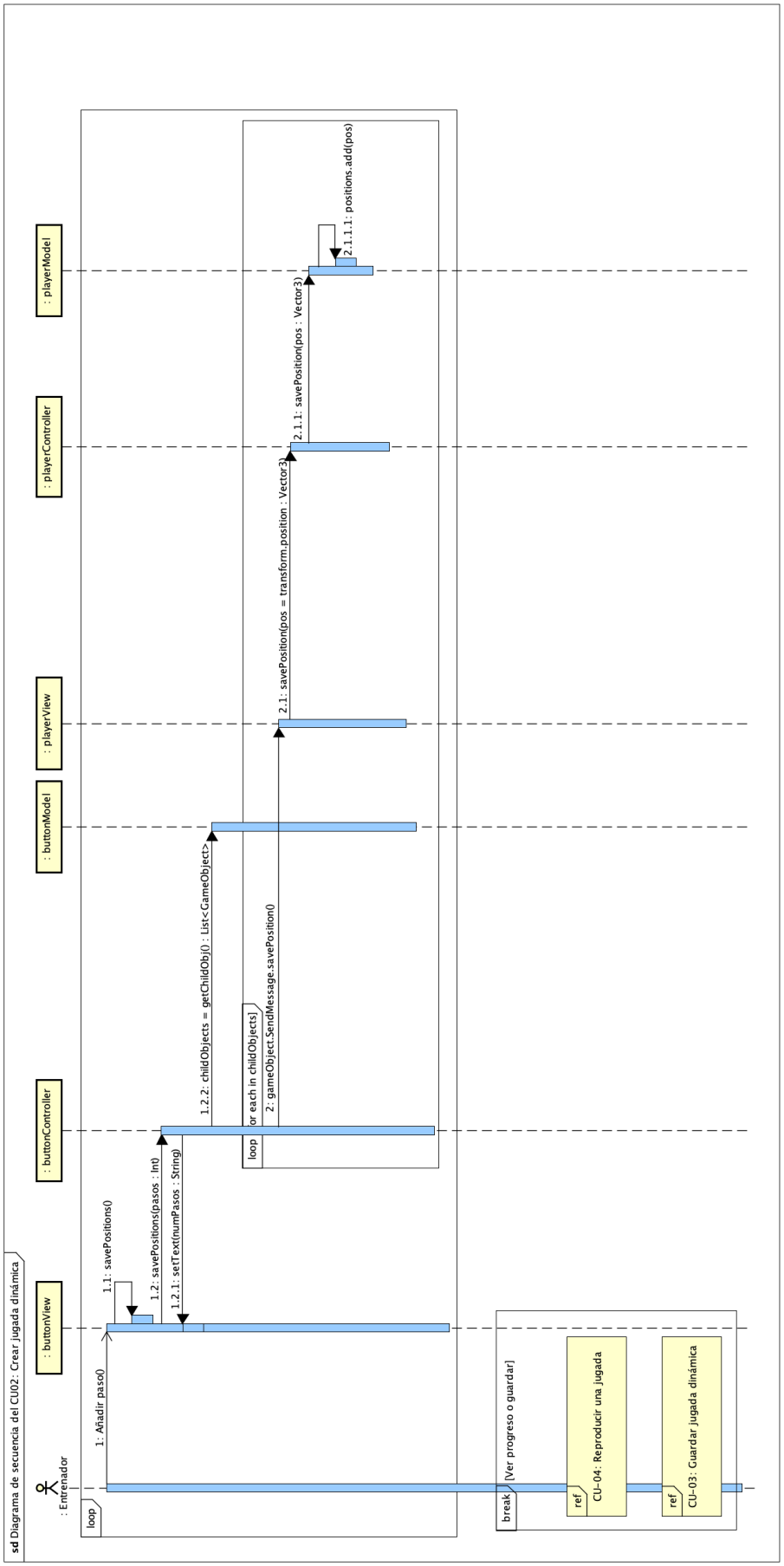

Figura 4.19: Realización en diseño del CU02: Crear jugada dinámica.

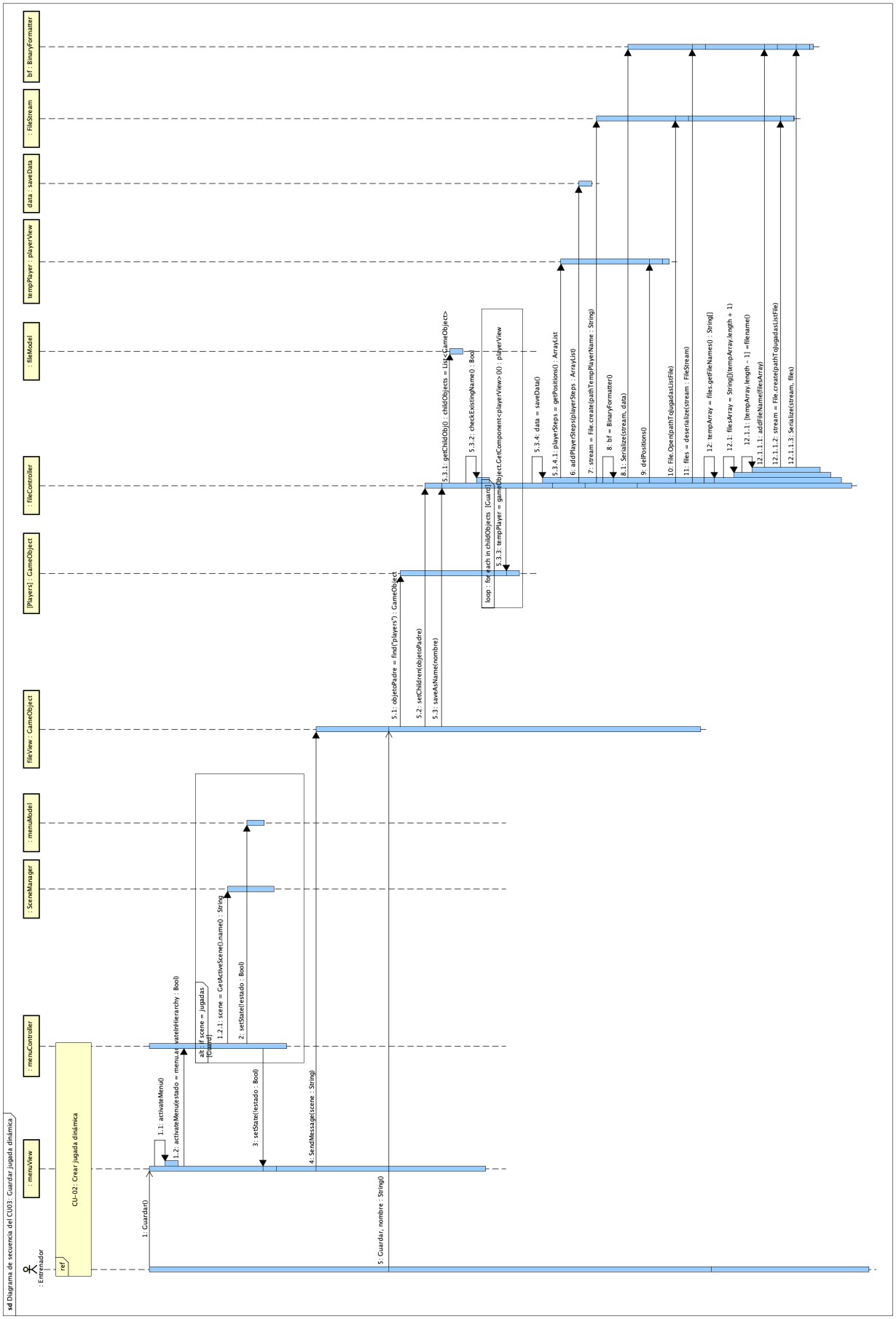

Figura 4.20: Realización en diseño del CU03: Guardar jugada dinámica.

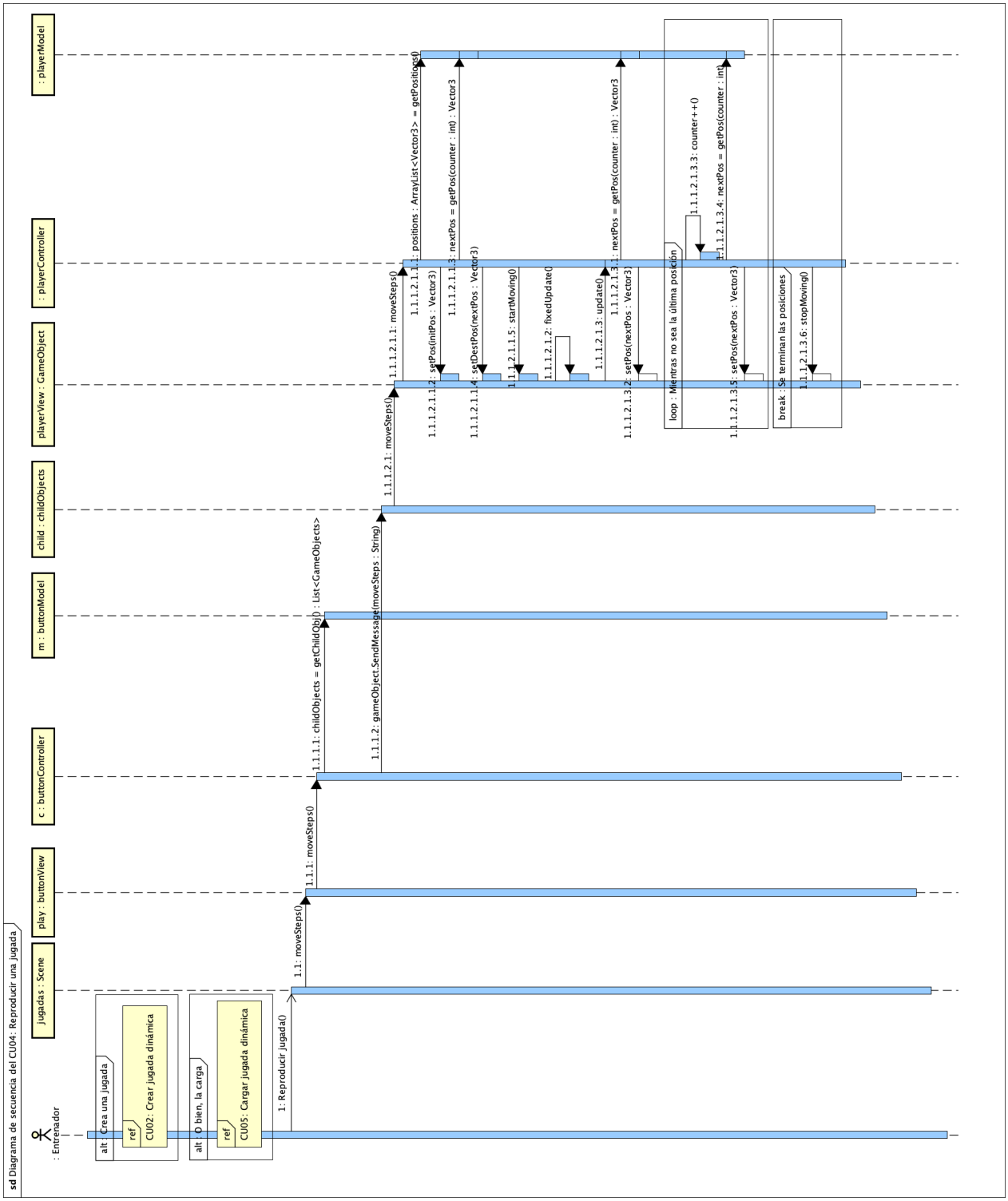

Figura 4.21: Realización en diseño del CU04: Reproducir una jugada.

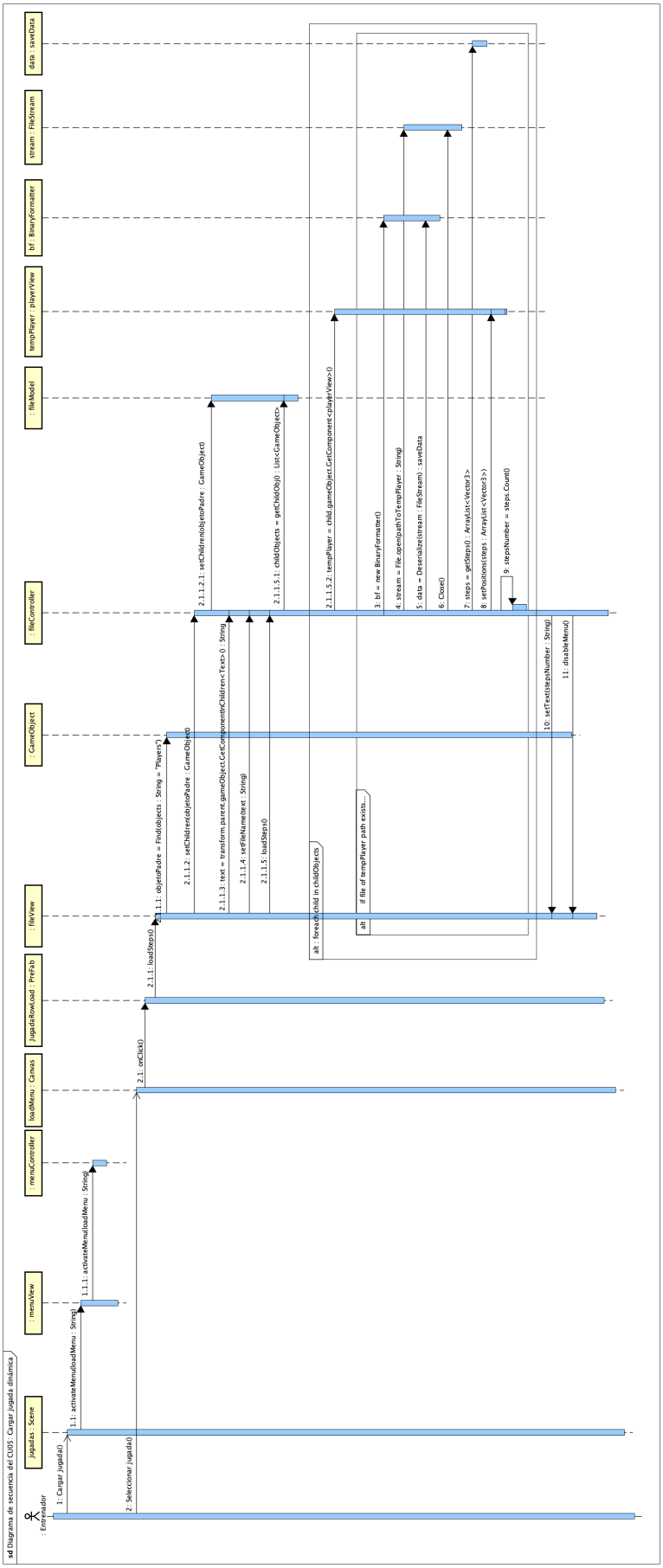

Figura 4.22: Realización en diseño del CU05: Cargar jugada dinámica.

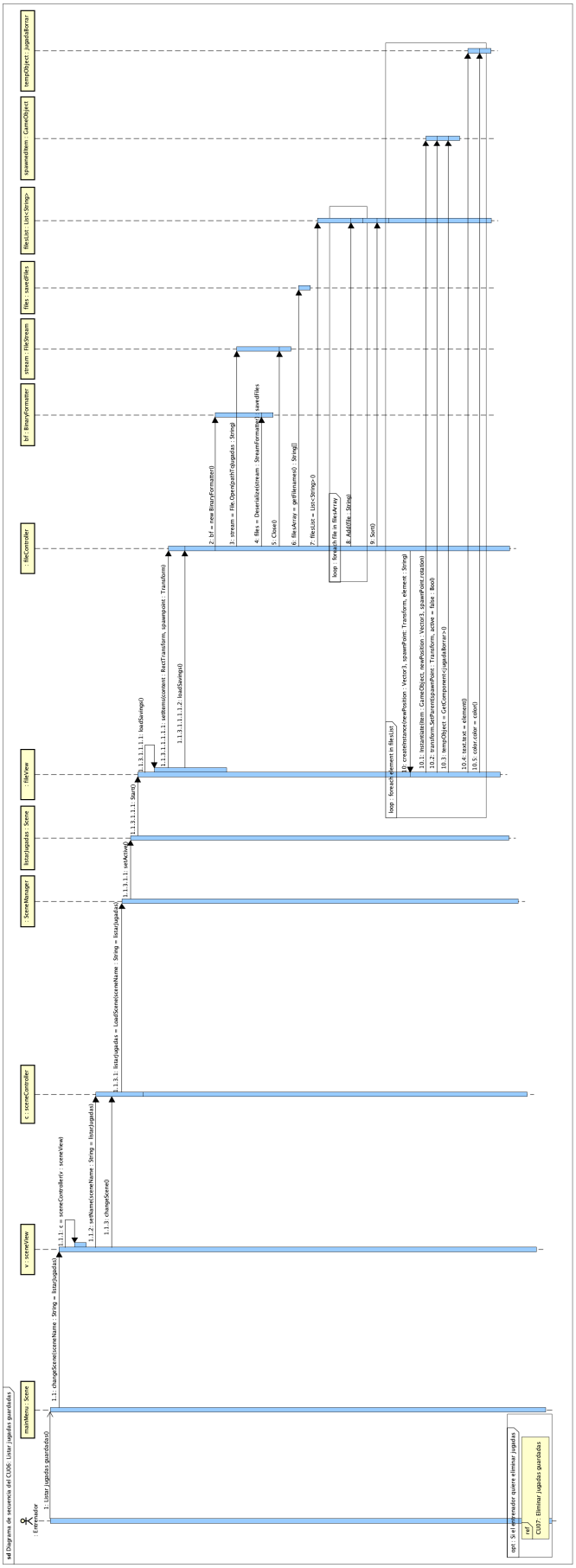

Figura 4.23: Realización en diseño del CU06: Listar jugadas guardadas.

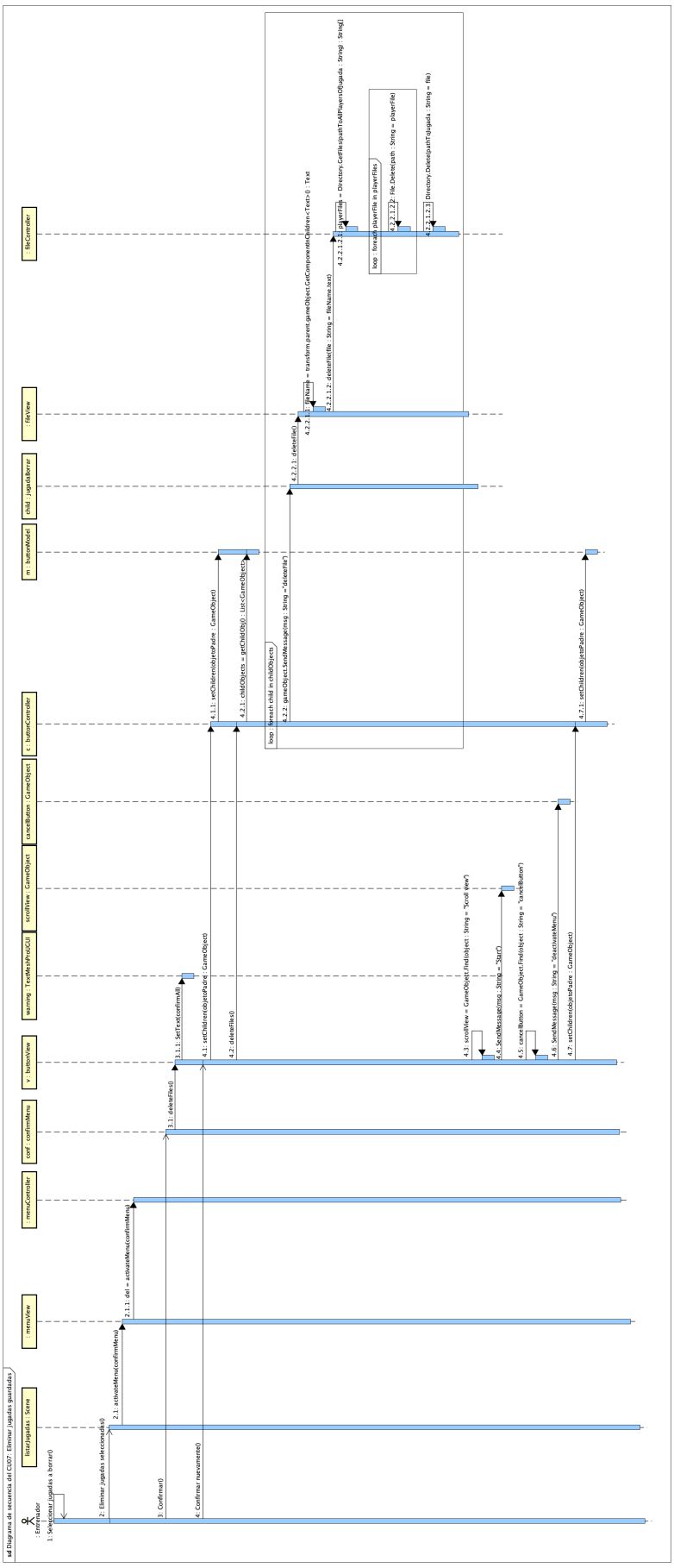

Figura 4.24: Realización en diseño del CU07: Eliminar jugadas guardadas.

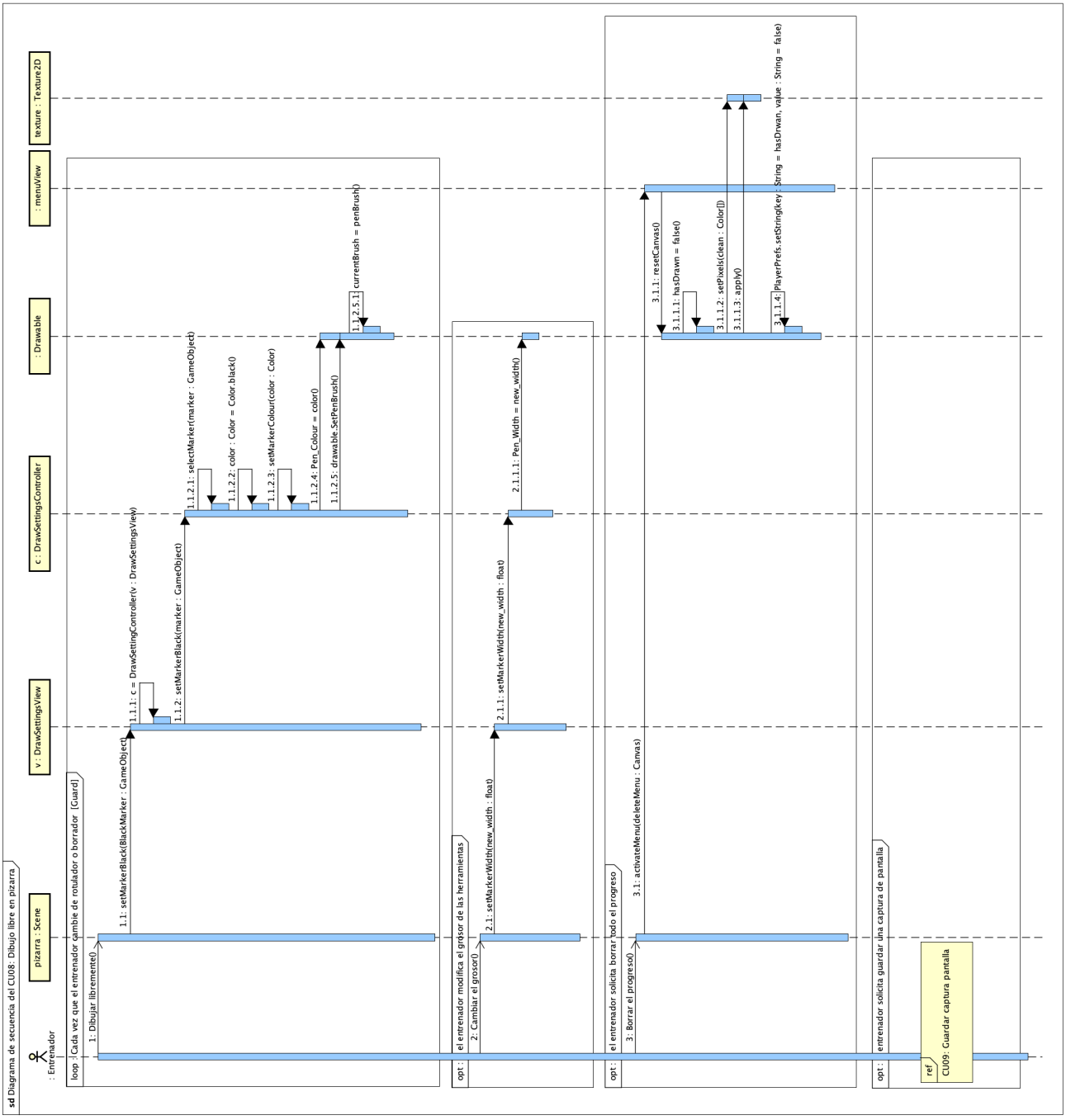

Figura 4.25: Realización en diseño del CU08: Dibujo libre en pizarra.

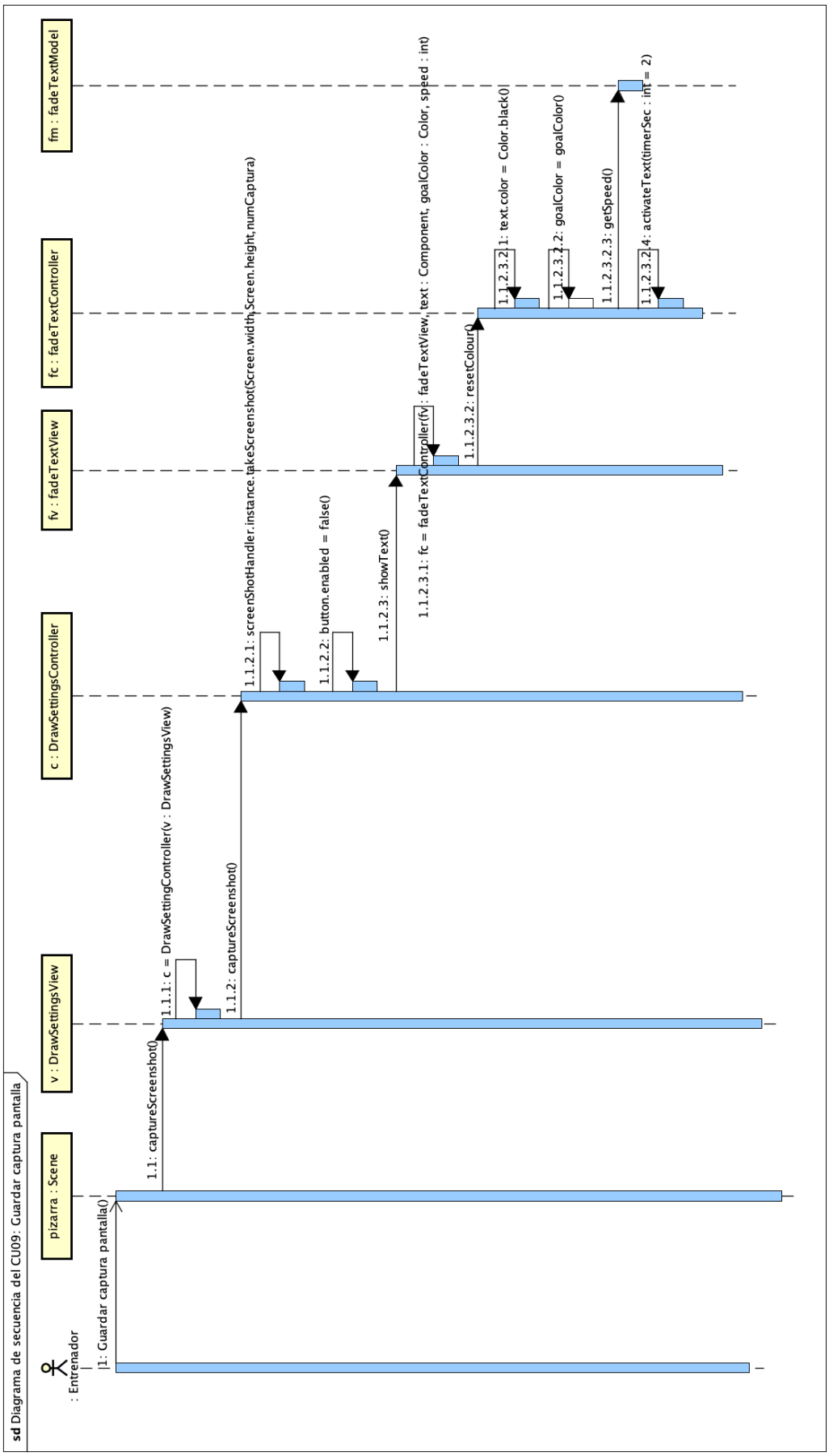

Figura 4.26: Realización en diseño del CU09: Guardar captura pantalla.

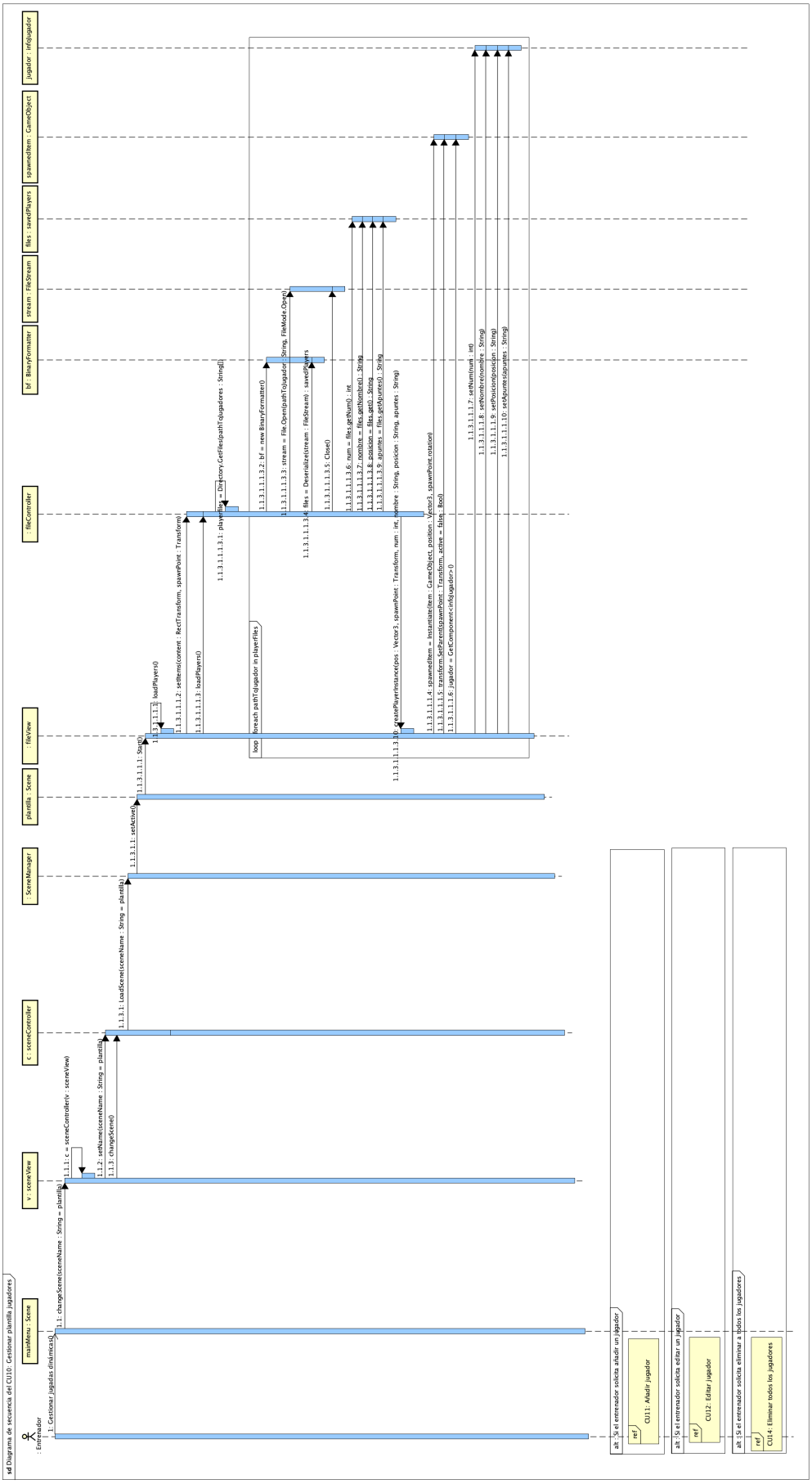

Figura 4.27: Realización en diseño del CU10: Gestionar plantilla jugadores.

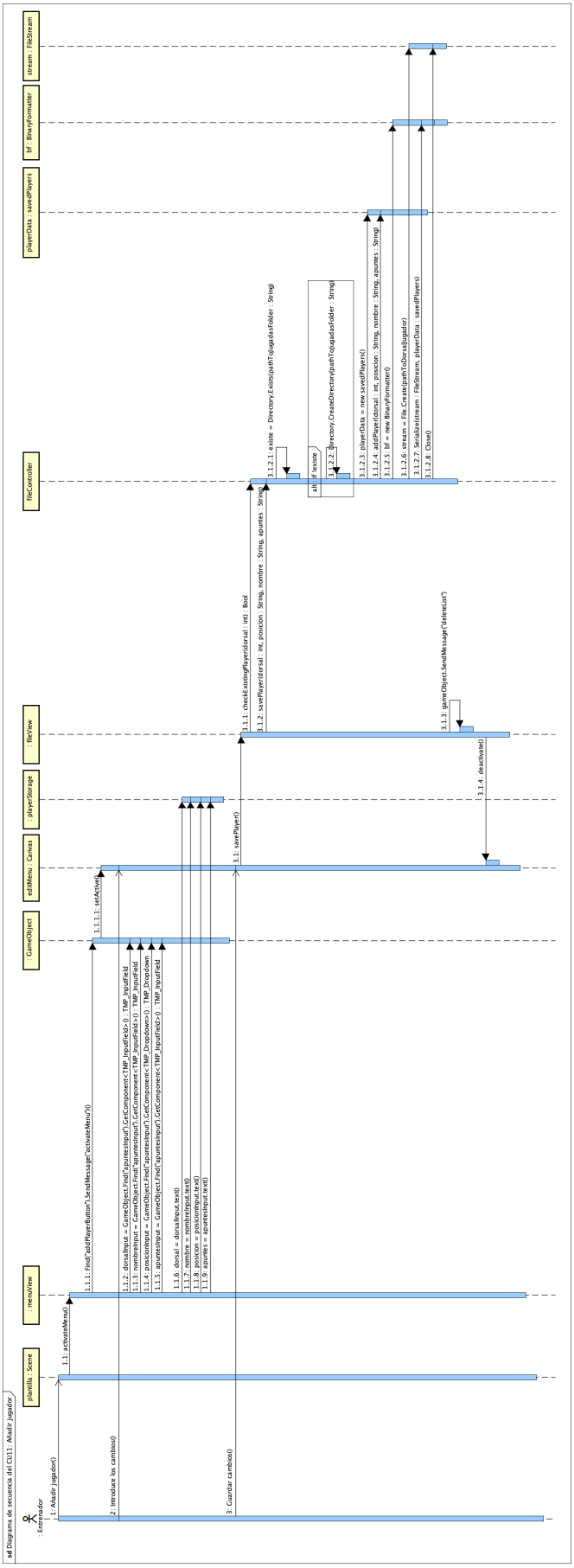

Figura 4.28: Realización en diseño del CU11: Añadir jugador.

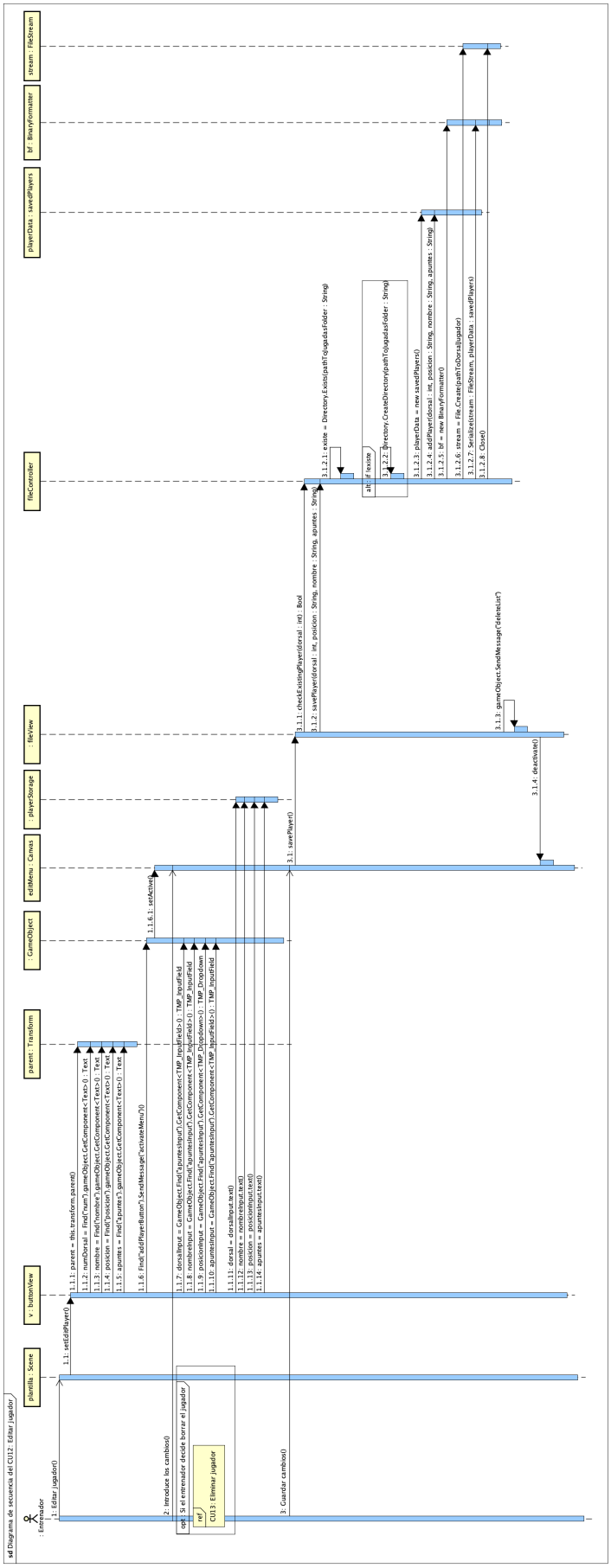

Figura 4.29: Realización en diseño del CU12: Editar jugador.

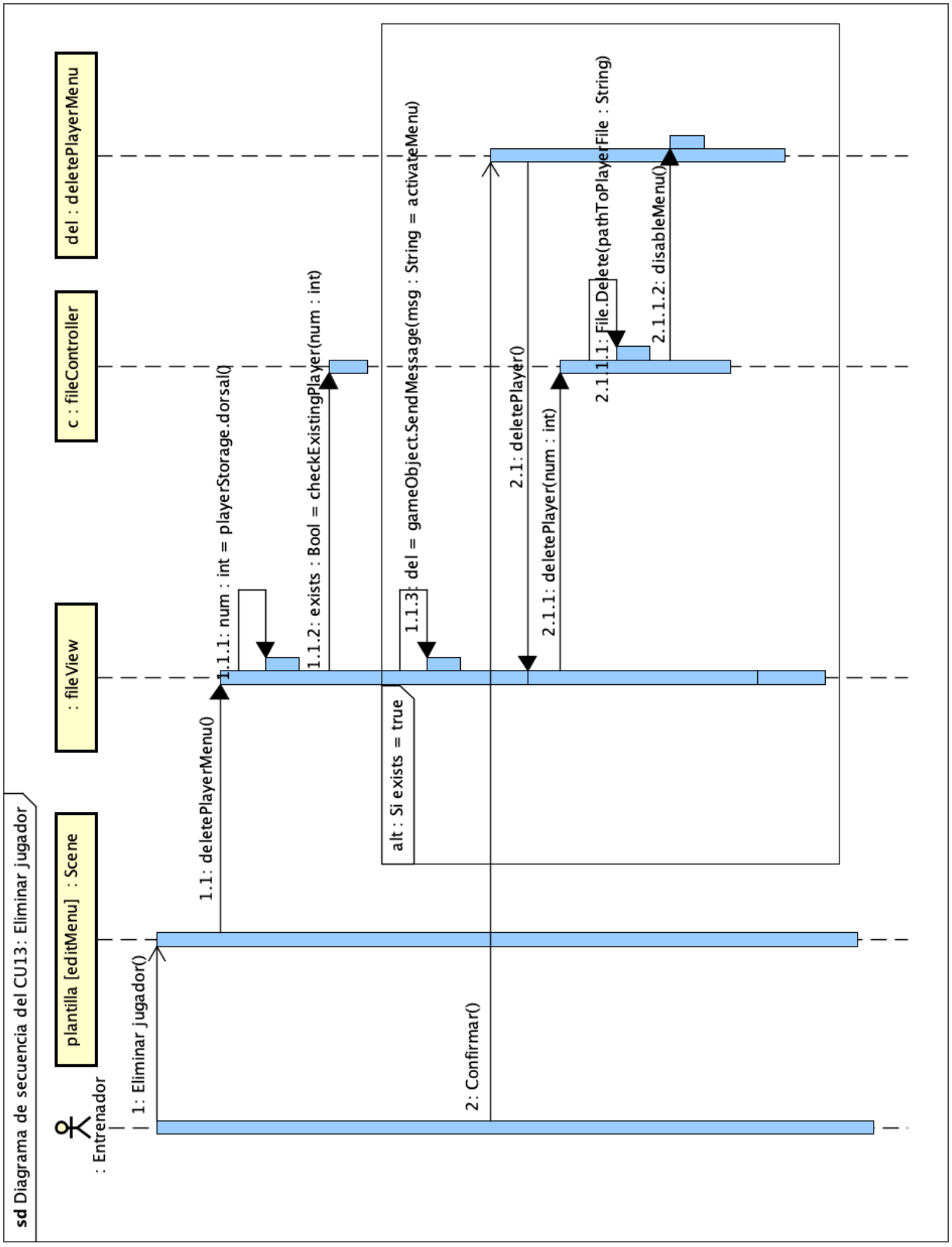

Figura 4.30: Realización en diseño del CU13: Eliminar jugador.

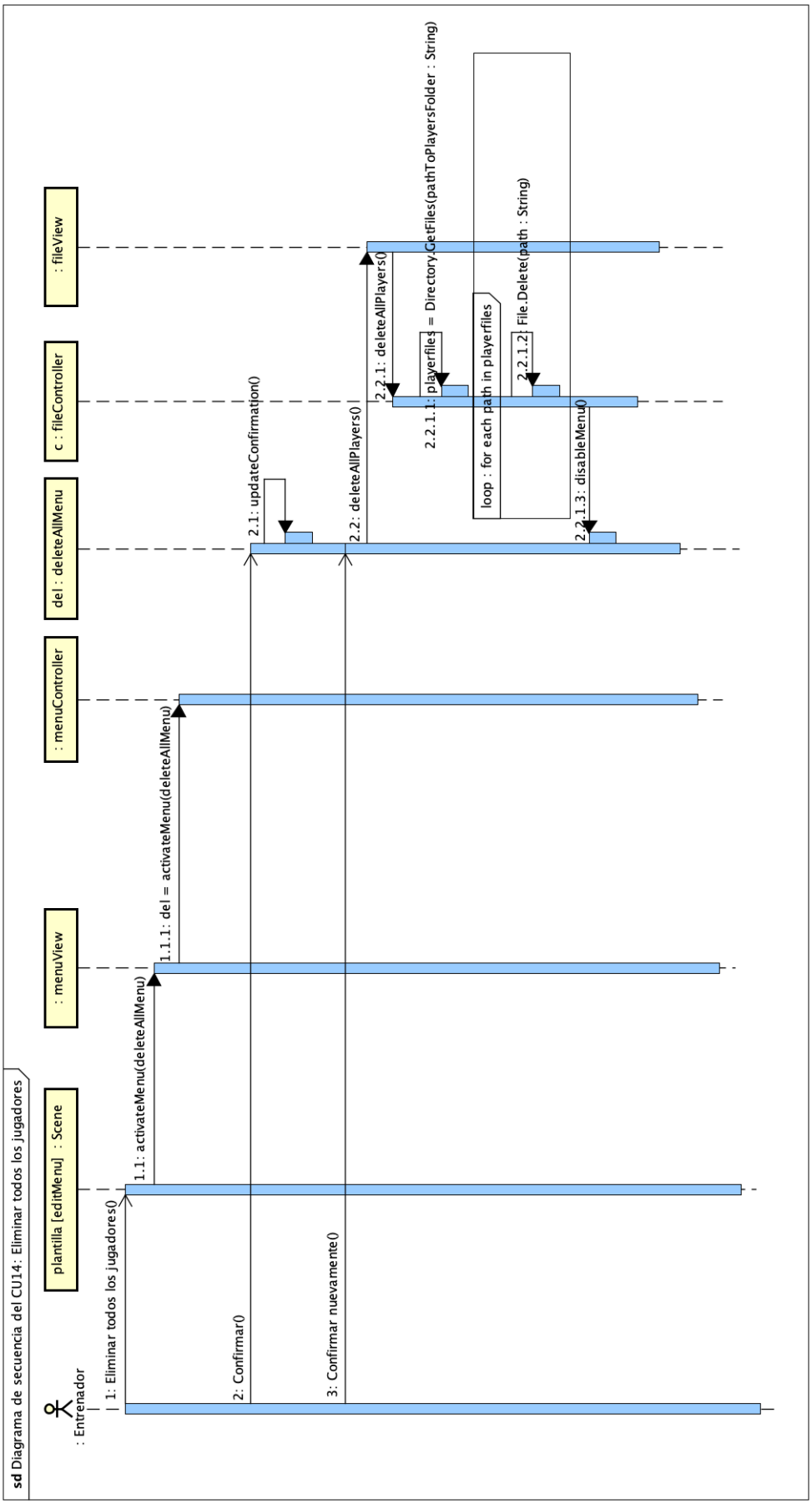

Figura 4.31: Realización en diseño del CU14: Eliminar a todos los jugadores.

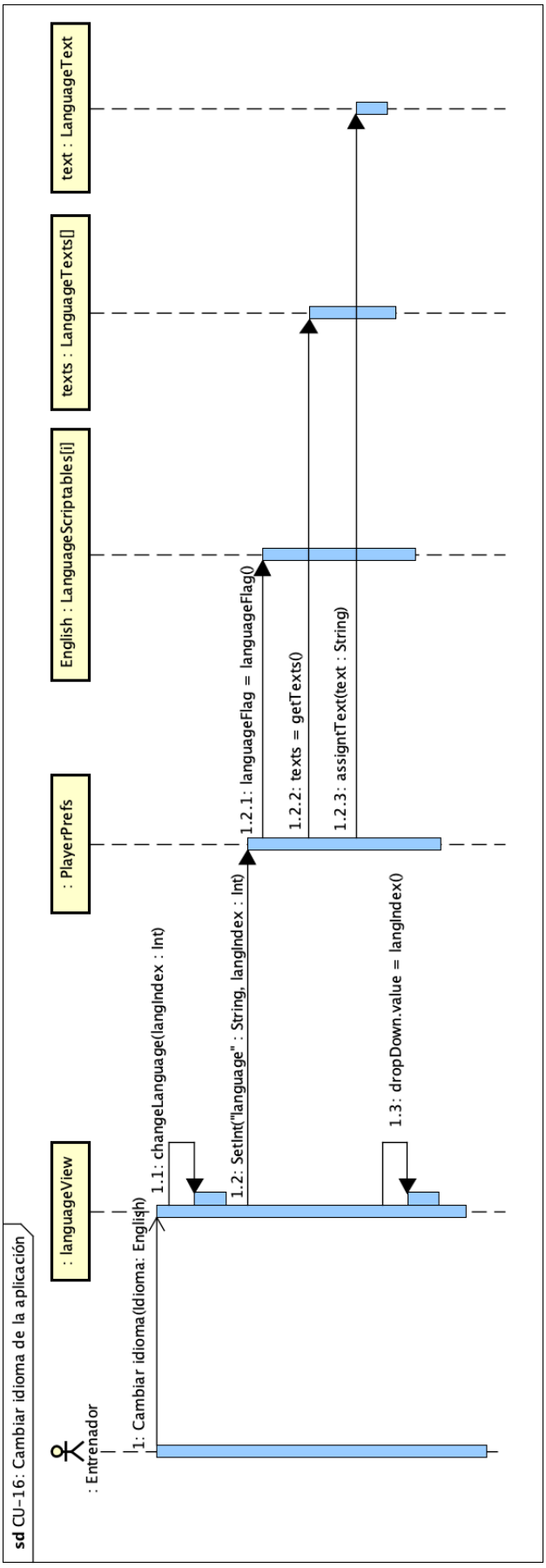

Figura 4.32: Realización en diseño del CU16: Cambiar idioma de la aplicación.

Capítulo 5

Implementación

En este capítulo se tratará todo aquello relacionado con la implementación de la aplicación software que sea destacable. Así pues encontraremos las herramientas empleadas, pequeños comentarios sobre el software y cómo se asegura la protección de datos de los usuarios.

5.1. Tecnologías utilizadas

Tras sentar las bases de los objetivos a lograr con la app, capítulo **??**, lo más importante era elegir las tecnologías que permitiesen llevar a cabo los mismos, debido a que no estaba claramente definido qué debía usarse, mientras se pudiese conseguir con un nivel de calidad acorde.

Primero se realiza una búsqueda, dentro de las tecnologías disponibles (Kotlin, Ionic y Unity), que mejor se ajustase al paradigma concreto, tarea fundamental bajo mi punto de vista, ya que si bien un paradigma se puede resolver de diversas maneras, elegir la mejor puede suponer importantes ventajas.

5.1.1. Unity

Unity [50] es uno de los motores de videojuegos[1] más utilizados en la actualidad, debido a la gran profesionalidad que puede lograrse en el producto final, prueba de ello son juegos actuales como "Fall Guys: Ultimate Knockout" [28] o "Monument Valley" [53], y muchos de los juegos móviles actuales. Es realmente una plataforma muy potente que permite lograr casi cualquier meta en este ámbito, y cuenta con una gran comunidad de usuarios y soporte.

Desde Unity Hub [52], la herramienta de gestión de Unity, se pueden gestionar las versiones del motor de videojuego instaladas en el equipo, así como los archivos que dan soporte a los sistemas objetivo de desarrollo, siendo muy sencillo de manejar. Se programa mediante un editor gráfico visual y mediante el uso de scripts[2], y presenta utilidades [27] realmente interesantes:

- **Motor gráfico para renderizar gráficos 2D y 3D.** Unity da la oportunidad de crear juegos tanto en 2D como en 3D, si bien los 3D son más llamativos, hay aplicaciones como ésta que tienen más sentido en 2D, y

[1] Software con rutinas de programación que permite diseñar y crear un entorno interactivo o videojuego.
[2] Código fuente de un lenguaje interpretado, sencillo, y que se ejecuta de manera secuencial.

casos de juegos tremendamente exitosos en la actualidad, como "Among Us" [23].

- **Motor físico que simule las leyes de la física.** Permite emular diversos materiales, así como físicas y mecánicas de juego, que brindan realismo y oportunidades a las creaciones que se pueden realizar.
- **Animaciones.** Uno de los elementos principales a utilizar, para el movimiento de los jugadores, pese a ser básico.
- **Sonidos.** Dada la naturaleza de un partido, donde puede haber bullicio, el apartado auditivo no se usará. En otros ámbitos podría usarse en combinación con las físicas de juego, para lograr una mayor inmersión.
- **Inteligencia Artificial.** Queda lejos del alcance de este proyecto, pero puede ser de gran utilidad a la hora de desarrollar la inteligencia/ comportamiento de los enemigos en los modos offline.
- **...y un largo etcétera.** Evidentemente, dado el tiempo disponible y naturaleza del proyecto, no se hará un uso completo de las herramientas que aporta Unity, ni siquiera los proyectos más ambiciosos pueden hacer un uso completo de la cantidad de soluciones que aporta.

Otro de los puntos fuertes de Unity radica en la cantidad de sistemas compatibles [51] en los que se puede ejecutar una aplicación, previa exportación:

- **Ordenadores:** Windows, Linux y macOS.
- **Móviles:** iOS y Android.
- **Consolas:** Play Station 4 y 5; Xbox One y sus modelos S, X, series S y series X; Nintendo Switch; Google Stadia.
- **Sistemas de realidad aumentada:** Oculus Rift, Magic Leap y Microsoft HoloLens (realidad mixta).
- **Televisiones inteligentes:** AndroidTV y tvOS.
- **WebGL** [43].

Dada esta versatilidad, se consigue que la aplicación objetivo de este proyecto, pueda ejecutarse en casi cualquier plataforma (todas aquellas en las que se haya probado), logrando así un gran acceso entre los potenciales usuarios, sea cual sea su dispositivo.

En conjunto, y tras investigar posibles implementaciones, hacen de Unity (a mi parecer) la mejor opción para resolver el paradigma planteado de jugadas dinámicas (logrando un producto final con la calidad requerida, eliminando complejidad innecesaria). La elección de la tecnología marca el éxito del proyecto así como la posibilidad de desarrollarlo finalmente o no, por lo que no era algo que tomar a la ligera, y para lo que se han dedicado algunas horas de investigación explorando las posibles soluciones.

Además, la programación de videojuegos tiene un papel cada vez más importante en la sociedad como avalan multitud de análisis financieros, [39]. Si bien el proyecto no es un juego como tal (algo lúdico), las mecánicas a emplear, se identifican claramente en este ámbito.

5.1.2. C#

C#, también conocida como C-sharp [29], es un lenguaje similar a Java, desarrollado por Microsoft para su plataforma .NET, siendo el lenguaje principal de ésta. De entre los lenguajes admitidos por Unity [8], C# es prácticamente la única opción, ampliamente admitida por la comunidad, aparte de resultar familiar para cualquiera con conocimientos previos de programación orientada al objeto.

También tiene sentido desde el punto de vista técnico, ya que Unity usa Mono [33], una plataforma de desarrollo de .NET. Por si fuera poco, la totalidad de las bibliotecas de Unity están escritas en C#, e incluso Unity lo considera como la única opción.

Algunas de las alternativas serían:

- **JavaScript.**
- **Boo.**
- **IronPython.**
- **Lua.**
- **C/C++.**
- **Rust.**

Tomando todo esto en consideración, C# es claramente el lenguaje usado en este proyecto Unity. Además, presenta una curva de aprendizaje menor que la de otros lenguajes, como Java o Kotlin, y es fácilmente escalable y mantenible, lo que siempre es un punto a favor a la hora de programar.

5.2. Herramientas empleadas

5.2.1. Visual Studio Code

Para poder programar en C# y editar sus scripts, es necesario un editor de código fuente, ya que no es posible desde Unity Hub. Visual Studio Code es la opción desarrollada por Microsoft, y va de la mano con Unity y C# debido a su componente .NET.

Sus ventajas son muchas, como se exponen en [31]...

- **Coloreado de sintaxis.**
- **Pareado de paréntesis.**
- **IntelliSense (función de autocompletado [42]).**
- **Snippets (métodos funcionales).**
- **CodeLens (ayuda de entendimiento del código [11]).**

- **Acciones de código y sugerencias.**
- **Navegación entre elementos.**

Además Unity permite la integración nativa con este editor, por lo que desde la gestión de las escenas, se pueden abrir directamente los ficheros para editarlos en el acto, ya que es el programa por defecto.

5.2.2. Astah

Astah [2], es una herramienta de diseño de software que permite plasmar ideas en diferentes diagramas UML. Los diagramas creados se pueden consultar en los capítulos 3 (Análisis) y 4 (Diseño), siendo algunos de los creados:

- **Diagrama de casos de uso.** Presenta de manera simplificada la relación de los actores con el sistema y las acciones (casos de uso) que pueden realizar en el mismo. Suele tener un carácter informativo para el cliente, y se realiza de la mano de los requisitos. A su vez, es común detallar los casos de uso, especificando el comportamiento del sistema durante el desempeño de los mismos, tomando en consideración tanto los casos de éxito como los de fracaso.
- **Diagrama de proceso de negocio.** También conocidos como diagramas de flujo (y con cierta relación a los diagramas de actividades), se emplean en la gestión de proyectos para visualizar el flujo de acciones que pueden llevarse a cabo de manera sencilla. Se identifican por tanto a los actores (en este caso, sólo existe el entrenador), los límites del sistema y las interconexiones de este.
- **Diagramas de secuencia.** Modela las interacciones que ocurren entre los diferentes objetos del sistema en función de las acciones que realice el usuario. Se muestra con carácter temporal el flujo de la aplicación para cada caso de uso 3.4. Todos los realizados son genéricos y no de instancia (donde se describe un escenario específico) ya que resulta de mayor interés, pudiendo aplicar cualquier caso en particular.

5.2.3. LaTeX

LaTeXes un sistema para componer textos. Permite la integración de referencias de una manera profesional, algo realmente necesario en un trabajo de esta índole. Su alta calidad tipográfica, lo hace destacar en los artículos científicos de todo el mundo, donde se considera en parte un estándar.

Otra de las ventajas que presenta es la posibilidad de independencia entre el contenido y el formato, puesto que cambiando unas pocas líneas, se puede cambiar el aspecto, numeración, tipografía, etc. presentes en todo el documento. Así como el estilo bibliográfico empleado, el cual puede modificarse únicamente mediante el identificador del mismo.

Por otra parte, representa una curva de aprendizaje inicial mayor que la de otros editores como Word, pero que se va aplanando de manera más rápida durante su uso, haciendo que para aquellos con experiencia, LaTeXresulte muy sencillo de usar, aportando así, además, un plus en cuanto a rapidez en la redacción.

5.2.4. MS Project

Este programa de Microsoft [30] que usamos en la asignatura Planificación y Gestión de Proyectos [55], gracias a la licencia de la escuela, tiene por objetivo ayudar en la gestión de proyectos. En mi caso, ha sido utilizado para

la creación de diagramas de Gantt, explicados en la sección 2, pero permite muchas otras funcionalidades como las líneas base, que son capturas de la situación puntual del proyecto, de manera que se pueden comparar entre ellas para evaluar el progreso.

Pese a todo, algunas funcionalidades como las líneas base, no las llegué a usar, dado que hay un seguimiento cercano y continuado del proyecto, el cual se detalla cada semana, por lo que se tiene una visión clara de éste y de su avance.

5.3. Entorno IDE

Para la implementación se ha hecho uso de Unity Hub, que es la herramienta de gestión de versiones del software. Se opta por descargar una versión estable de desarrollo Unity, para evitar en lo posible errores no corregidos, y seguir las recomendaciones de la comunidad, la versión elegida es la 2020.3.3f1. Siendo la versión final empleada la 2020.3.11f1, este cambio se debe a la resolución de ciertos fallos que se solucionaban actualizando el software. Inicialmente se descarga el soporte para Android, Windows y macOS.

La configuración del entorno fue realmente sencilla, comenzando un proyecto desde cero el cual sincronicé vía terminal, con un proyecto desde mi cuenta de estudiante de GitLab, para mitigar el riesgo R13 (tabla 2.1), y porque en el desarrollo de software, es una buena práctica emplear un repositorio. El proyecto inicial consta de una escena donde se desarrollarán las jugadas dinámicas, la actividad central de la App.

A su vez se emplea Visual Studio Code, para la gestión de los ficheros C#. Su uso es muy sencillo debido a que permite importar un proyecto completo de Unity. Siendo éste el mismo que se actualiza en el repositorio remoto de git, cada vez que se lanza el programa se mantiene actualizado, aunque se cambie de rama. Por si fuera poco, Unity permite su inclusión en el IDE, por lo que cuando hemos terminado de diseñar la interfaz de usuario podemos asignar directamente los scripts, y modificarlos abriéndolos desde Unity, sin tener que realizar búsqueda alguna.

Por otra parte, y para completar la información sobre el entorno de desarrollo, no puede faltar los dispositivos usados tanto para la implementación como para las pruebas (detalladas en el subcapítulo 6.1). Éstos fueron, con sus características:

MacBook Pro 13”	Implementación, desarrollo de memoria y pruebas.
Sistema operativo	macOS Big Sur (11).
Procesador	Intel i5-5257U. 2 núcleos, 2.7GHz.
Memoria RAM	8GB DDR3.
Pantalla	24”, 2560x1440 píxeles. Formato 16:9.

Tabla 5.1: Entorno de desarrollo: MacBook Pro.

Tablet Android	Dispositivo de pruebas principal.
Sistema operativo	Android 10 .
Procesador	Ocho núcleos, 1.6GHz.
Memoria RAM	4GB.
Pantalla	10.1”, 1920x1080 píxeles. Formato 16:9.

Tabla 5.2: Entorno de desarrollo: Tablet Android.

PC Windows básico	Dispositivo de pruebas antiguo, de 32 bits.
Sistema operativo	Windows 10, 32 bits.
Procesador	Intel Core Duo. Dos núcleos, 1.4 GHz.
Memoria RAM	2GB.
Pantalla	19", 1366x768 píxeles. Formato 16:9.

Tabla 5.3: Entorno de desarrollo: PC Windows básico.

PC Windows medio	Ordenador de prestaciones comunes, de 64 bits.
Sistema operativo	Windows 10, 64 bits.
Procesador	Intel i5-4570. Cuatro núcleos, 3.2 GHz.
Memoria RAM	8GB.
Pantalla	21", 1920x1080 píxeles. Formato 16:9.

Tabla 5.4: Entorno de desarrollo: PC Windows medio.

5.4. CVS Control de Versiones

5.4.1. GitLab

GitLab [17] es una aplicación completa destinada a desarrolladores software, en concreto para los ciclos de desarrollo (creación de los componentes software a partir de los diseños ideados previamente) y despliegue (hacer disponible el software para su uso, tras las pruebas).

Además, la escuela nos permite hacer uso de esta plataforma con cuentas de funcionalidad "Premium", siendo ésto una gran ventaja sobre la versión normal (mejor trabajo entre equipos, mayor capacidad de control y minutos de pruebas, entre otras).

Cuenta con numerosas herramientas para facilitar el desarrollo y gestión de software:

- **Planificación.** Mediante issues (tareas o asuntos que resolver) se pueden organizar las funcionalidades a alcanzar, dividiendo así en pequeñas tareas el trabajo completo que debe ser alcanzado. Se establece un nombre identificativo para la tarea, así como una descripción y la posibilidad de agregar a un miembro del equipo como encargado de la misma.

  También se incluye un tablero Kanban [24] (figura 5.1), con el que se pueden vislumbrar fácilmente las tareas que faltan por hacer, aquellas que están en progreso, que fueron finalizadas, en fase de revisión, o aprobadas (y cualquiera que se quiera añadir por parte del usuario, para su mejor gestión personal). Las issues pueden tener una fecha estimada de fin, facilitando el visionado de las que están prontas a vencer, y recibiendo avisos por aquellas que deberían estar completadas pero que no se lograron a tiempo.

- **Control de versiones.** La creación de repositorios[3] es uno de los aspectos más conocidos, ya que permite recuperar versiones del software específicas. Esto es útil en caso de tener que regresar a un punto anterior por cualquier motivo, generalmente el dejar de lado una funcionalidad por ser inalcanzable.

  Este control se lleva a cabo mediante:

  - **Ramas.** Cada rama, permite a los miembros del equipo trabajar en sus tareas sin interferir en el trabajo de otros. Es muy común tener una jerarquía donde existe una rama **master** donde se vierten las versiones del software, se usa por tanto, exclusivamente para los entregables. También la rama **develop**

[3]Espacio digital centralizado donde se almacena, mantiene y distribuyen archivos informáticos.

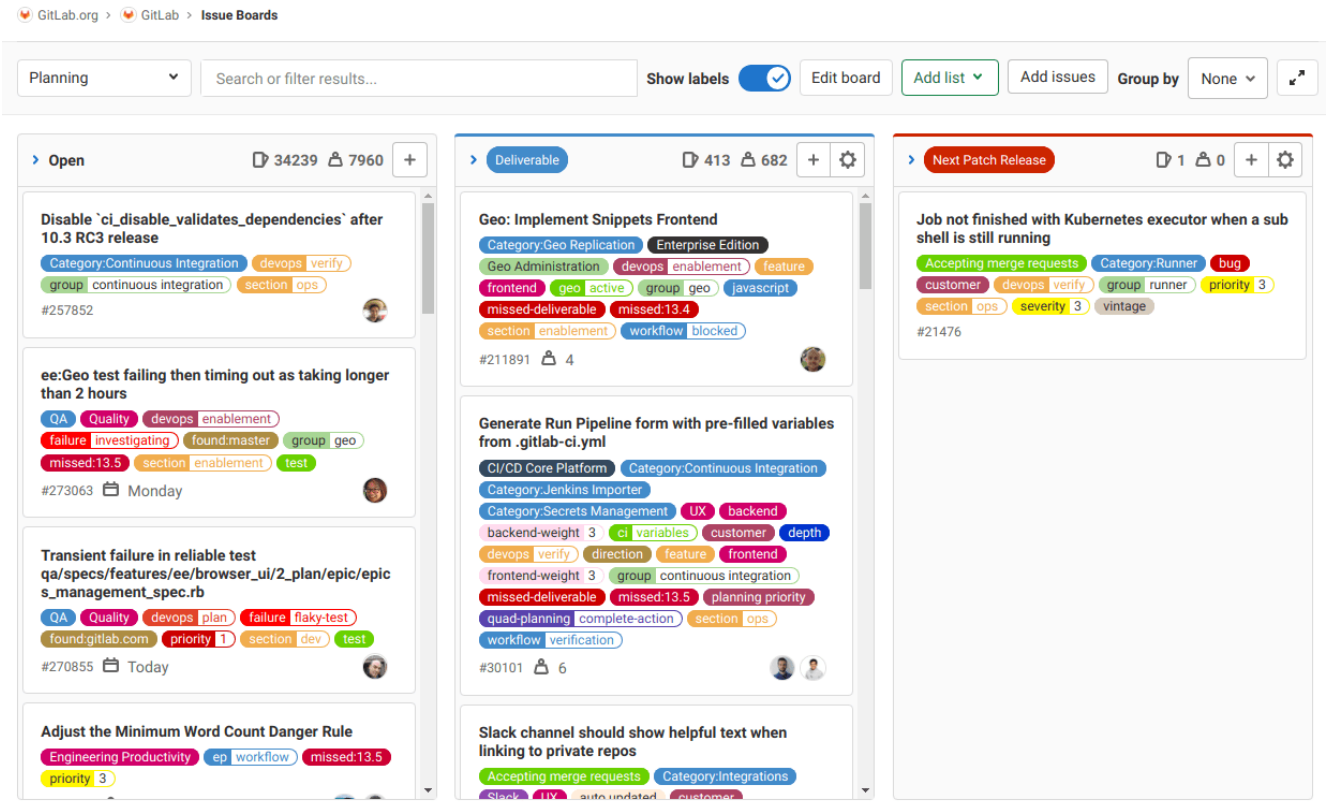

Figura 5.1: Ejemplo de tablero de Issues. Fuente: [16].

la considerada rama común, donde es volcado el trabajo del resto de ramas para su puesta en común. Desde develop, surgen las ramas de **feature's** (del inglés característica o funcionalidad), pequeñas tareas que aportan al global de la aplicación.

- **Resolución de conflictos.** Pese a la naturaleza de las ramas, en ocasiones se debe desarrollar en paralelo alguna parte del sistema, y por tanto, puede haber conflictos entre las diferentes aportaciones de código. Esto es revisado fácilmente desde GitLab, donde nos avisa de éstos a la hora de hacer un merge request, marcando en rojo las partes de código involucradas.
- **Commits.** Mediante esta operación, se sincronizan los cambios implementados en local por el desarrollador, incorporándolos al repositorio. Es un estándar acompañar al commit con un mensaje declarativo de aquello que se ha logrado implementar, y no realizarlo en caso de tener errores de compilación del código. Estos cambios son almacenados en la rama de desarrollo que se esté utilizando.
- **Merge requests.** Tras al menos un commit, se puede realizar la operación de fusión a otra rama, generalmente a **develop**. Resultando así en la combinación de los ficheros modificados por el programador, con nueva funcionalidad, con lo ya presente en la rama destino.

En definitiva, GitLab es una excelente herramienta para la gestión del código.

5.4.2. Copias locales

Como método alternativo de seguridad, aparte de la copia local, emplearé dos discos duros, donde guardaré todas las versiones del proyecto como zip; y la versión actual de cada momento, como directorio completo. Atendiendo así al riesgo R13 de la tabla 2.1.

5.5. Aspectos relevantes sobre la implementación

5.5.1. Unity, conceptos clave/básicos aprendidos

Quisiera exponer debido a las diferencias que supone el desarrollo en Unity en comparación a otros lenguajes de programación más típicos del grado, y de la mano del manual de Unity [44], algunos de los conceptos claves empleados con esta tecnología.

Assets

Son la representación de cualquier item que puede ser usado en el proyecto, y que son archivos importados en Unity como imágenes, vídeos, música, texturas (propias o generales), controladores de animaciones, modelos 3D, etc [45].

En cualquier caso, el archivo original permanece intacto, puesto que se crea una copia como una representación en el juego. Si posteriormente se realizan modificaciones en los Assets, éstas pueden aplicarse actualizándolos fácilmente.

A su vez puede encontrarse en la **Asset Store** una gran cantidad de éstos ya sean de manera gratuita o de pago, para ayudar al programador a desarrollar a partir del trabajo previo de otros. Esta idea es la misma que vimos en la asignatura Diseño Basado en Componentes y Servicios [25] donde una extensa tienda de componentes permite reutilizar código al programador, para tardar menos tiempo, usar menos recursos, y lograr un mejor resultado final.

Escenas

Son Assets que contienen la aplicación o parte de ella, ya que es donde se sitúa el contenido. Las escenas permiten crear niveles de un juego o separar partes de una aplicación. Para crear una escena debe hacerse desde una plantilla, las cuales proporcionan una preconfiguración, ayudando de esta manera al programador simplificando su tarea al presentar todos los elementos que necesita para empezar. De hecho, se pueden crear plantillas propias (que son almacenadas como Assets), teniendo así el programador un resultado personalizado para lo que necesite.

Toda escena en Unity debe tener una cámara para poder visualizar correctamente los elementos presentes en el **Canvas**, por convención, la primera cámara habilitada es conocida como Main Camera.

Las escenas son, por tanto, la manera de separar las diferentes vistas que tiene una aplicación. Una será el menú principal, desde el cual se puede regresar al sistema, y las demás, diferentes escenas que aportan la funcionalidad por medio de los diferentes objetos presentes en ellas y los **scripts** que los modifican. Así que para navegar entre escenas solo hace falta introducir una transición, y tener en cuenta la situación de la escena actual.

Canvas

En el canvas [46] se sitúan todos los elementos de interfaz de usuario, es un **GameObject** con un **Component** de tipo Canvas, del cual parten todos los elementos de interfaz de usuario, que son sus hijos. Por medio del sistema de eventos se comunica con el sistema de mensajes.

Los objetos presentes en el canvas siguen una jerarquía, donde el primer hijo es posicionado primero, y se sigue el orden natural, pudiendo sobreponerse unos a otros, por lo que para reordenarlos basta con cambiar el orden de la jerarquía (pudiendo realizarse de manera manual o vía script, mediante los métodos de **Transform Component**).

Un canvas se puede renderizar para ser empleado en una pantalla, cámara, o en un mundo (lo que podría ser el escenario de un videojuego 3D). Los cambios en una pantalla, como puede ser el tamaño, o entre pantallas (resolución) son tenidos en cuenta aquí y debe asegurarse que funcionará de manera automática. Emplear un Canvas como cámara es similar a una pantalla, con la salvedad de la distancia que incorpora la cámara. Ésta es la encargada de dar la perspectiva, el campo de visión, el ángulo con el que se interpreta..., por lo que los ajustes modificarán la manera de contemplar el escenario. Y en caso de emplear el espacio de mundo, el canvas se comporta como cualquier otro objeto, enfocándose en aquello que puede apreciarse.

Component

Un componente incluye de manera autocontenida instrucciones de código y que implementan cierta funcionalidad. Los componentes pueden ser usados por los objetos (GameObjects) para reunir funcionalidades y crear las aplicaciones por medio de pequeños bloques de código. Por ejemplo, para poder hacer clic en un objeto, se añade el componente Selectable, y se registra un capturador de eventos, el cual avisará a la lógica del programa para actuar conforme a ello, ya sea una pulsación de botón, arrastrar un jugador, o pintar una línea sobre el tablero.

La idea se basa en crear un sistema conectado por partes más pequeñas de software, reduciendo el tiempo a emplear. A su vez, permite ajustarse a las buenas prácticas de programación, como es la reutilización, modularidad y una mayor calidad en el producto final.

GameObject

[47] Son los objetos fundamentales de Unity y representan los escenarios y aquello que se incluya en éstos, actuando como contenedores de los **Componentes** que se encargan de la funcionalidad.

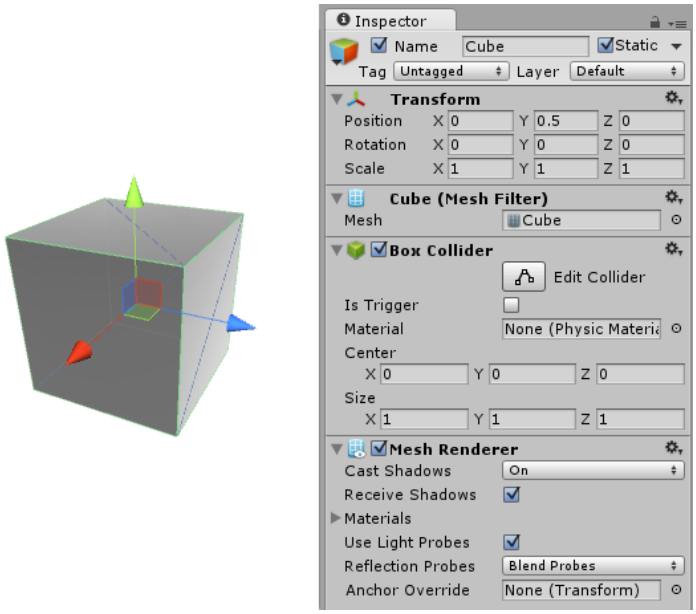

Figura 5.2: GameObject que representa a un cubo. Fuente: [47]

[49] Los sprites son objetos 2D, y un tipo de Asset que puedes importar al proyecto. Por ejemplo las imágenes al trabajar en 2D como es el caso, son Sprites de manera automática.

Prefabs

El sistema de Prefabs [48] permite crear, configurar y **almacenar** un **GameObject** por completo (componentes, propiedades y GameObjects hijos) en un Asset reusable. Es una plantilla desde la cual se pueden crear instancias para los objetos de una escena. En caso de querer disponer en una escena de varios GameObjects "iguales", se debería realizar con un Prefab, en nuestro caso para los jugadores, por ejemplo. La diferencia reside en que, de esta manera, todas las copias quedan sincronizadas, y permiten realizar cambios de manera conjunta, al aplicarlos al prefab.

Los Prefabs permiten crear una jerarquía, de tal manera que se puedan conseguir objetos más complejos, pero que sean modificables de manera sencilla desde los diferentes niveles. Y, evidentemente, no todas las instancias son idénticas pudiendo modificar sus propiedades (o siguiendo el ejemplo, posiciones), pudiendo llegar a crear variantes de los prefabs.

Ejemplos de Prefabs serían los jugadores, los botones de movimiento rápido, los rotuladores de la pizarra, los botones atrás y todas las filas que son mostradas en cualquier parte de la app.

Scripts

Los scripts (pequeños fragmentos de código interpretado por otros programas, y no por el procesador como es el caso de los programas compilados) permiten crear **componentes propios** para controlar la aplicación, manejar eventos que se produzcan (como la interacción del usuario) o modificar las propiedades de Componentes. Éstos pueden ser aplicados a los GameObjects para modificar su comportamiento. Programados en C# (ver sección 5.1.2) se ejecutan en la aplicación todo el tiempo que esté activa, la primera vez con la función Start() y una vez por frame con Update().

Son, por tanto, la parte central de cualquier aplicación Unity, pues es lo que la hace responsiva. En los Scripts encontraremos por ejemplo: los cambios de escena, el manejo de eventos que realice el usuario, como son los movimientos de jugadores en la creación de jugadas dinámicas y también los cambios necesarios para guardar y eliminar jugadas, entre otros.

5.5.2. Desarrollo de la pizarra

Para la funcionalidad de la pizarra (o escena en Unity) se ha empleado un Asset descargado [35] desde el Asset Store, gratuito, que presentaba prácticamente la totalidad de funcionalidad requerida. Puede ser consultada desde la página web del Asset Store, todos los derechos a su creador.

Realmente fue enriquecedor poder encajar la funcionalidad provista en la aplicación. El hecho de que resolviese el problema no evitó que tuviera que adaptar el código a la estructura del proyecto. Además de cambios visuales, se corrigieron mínimas imperfecciones, y se agregaron otras funcionalidades, como son las capturas de pantalla y las comprobaciones de seguridad para evitar despistes.

En la Universidad, habíamos tratado previamente con componentes, pero siempre los habíamos creado nosotros, o los profesores proporcionaban algo similar a una API, la cual podíamos usar de manera sencilla. Personalmente, me alegro de haberlo encontrado y haber podido aplicar los conceptos estudiados para integrar una solución real, presente en el mercado al software que estaba desarrollando, ya que es una práctica habitual en el mundo laboral, donde constantemente debes trabajar con APIs y componentes prefabricados. Ha sido, en definitiva una gran y enriquecedora experiencia.

5.6. ¿Cómo se asegura la protección de datos de los usuarios?

La importancia de la protección de datos fue tratada en el subcapítulo 3.3 en el que además se expone cómo EntrenaBal no solicita prácticamente ningún dato a los usuarios. Los datos solicitados son meramente de los jugadores, y son: su nombre (o apodo), su dorsal, y la posición en la que juegan. Ninguno de estos datos es crítico y no se comparten con terceros ni se analizan.

Pese a todo, podría decirse que los datos están seguros, ya que:

- No se utilizan servicios de terceros, por lo que no hay que aceptar la política de privacidad de otras aplicaciones o empresas.
- Todo el almacenamiento es local, por lo que la seguridad e integridad de éste, depende únicamente del usuario. Si el método de seguridad o contraseña de su dispositivo es vulnerable, o directamente no tiene, es responsabilidad únicamente del usuario.
- Y finalmente, no supone ningún riesgo, ya que los datos involucrados son públicos, considerando que su disposición en el campo determina su posición, y el número y el dorsal aparecen en la camiseta de juego.

Capítulo 6

Pruebas

En este capítulo se exponen las pruebas realizadas para asegurar la calidad de la aplicación así como la comprobación de todas las funcionalidades presentes en la app. De igual manera, se exponen las observaciones tras la exposición de la aplicación a los usuarios.

6.1. Pruebas de caja negra

Las pruebas de código son una parte fundamental en cualquier desarrollo software. Permiten asegurar la calidad del producto final así como comprobar que se están cumpliendo todos los requisitos y objetivos.

Las pruebas se clasifican habitualmente en dos grandes grupos:

- **Pruebas de caja blanca.** Evalúan el código "internamente", es decir comprueban la corrección de la lógica implementada. El mayor exponente de éstas son los test unitarios, los cuales consisten en evaluar los diferentes métodos o partes de código de manera aislada, asegurando que todos funcionan de manera correcta. Se asegura que nada rompe el sistema, es decir, que ninguna acción modifica el correcto funcionamiento.
- **Pruebas de caja negra por caso de uso.** Por otra parte, este tipo de pruebas se basan en comprobar la funcionalidad sin ver el código. Se evalúa de manera externa el correcto funcionamiento del software, para cada uno de los casos de uso; la salida obtenida debe ser igual a la esperada, para todos los casos.

También resulta interesante evaluar la interfaz gráfica, al ser una aplicación que se basa en este aspecto, pero es casi imposible realizarlas de manera automática. Por ende, suelen realizarse con usuarios de prueba, que evalúan la aplicación como se expone en el siguiente subcapítulo.

Si bien se han realizado pruebas de caja blanca a cada método, a continuación se exponen únicamente las pruebas de caja negra manuales, a petición expresa de mi tutora. Cabe destacar en este punto, el alto componente gráfico de Unity. La batería de pruebas detalladas de caja negra, es la siguiente [1]:

[1] CP-01, por ejemplo, equivale a Caso de Prueba 1.

6.1.1. Jugadas

CP-01	Crear una jugada dinámica.
Descripción	Desde la ventana de Jugadas se añaden pasos formando una jugada.
Entrada	Se disponen los jugadores en el terreno de juego, y por cada posición se añade un paso mediante el botón "+".
Resultado esperado	Una jugada que se puede reproducir y guardar correctamente. También se pueden añadir más pasos tras reproducir.
Resultado	Correcto.

Tabla 6.1: Descripción del CP-01: Crear una jugada dinámica.

CP-02	Guardar una jugada dinámica.
Descripción	Añadir una jugada dinámica al sistema.
Entrada	La jugada creada en el CP-01. Desde la escena Jugadas se selecciona "Guardar" y en el formulario se introduce un nombre de jugada no existente en el sistema: ▪ Nombre: Ataque posicional. Finalmente se selecciona "Confirmar".
Resultado esperado	La jugada es añadida y es visible desde la escena Jugadas al seleccionar "Cargar jugada" y desde la escena Listar jugadas.
Resultado	Correcto.

Tabla 6.2: Descripción del CP-02: Guardar una jugada dinámica.

CP-03	Cargar una jugada dinámica.
Descripción	Se carga una jugada dinámica que fue guardad previamente.
Entrada	Jugada guardada en el CP-03. Desde la escena Jugadas, se selecciona "Cargar jugada" y se selecciona "Cargar" desde la fila correspondiente a la jugada que tiene por nombre "Ataque posicional".
Resultado esperado	La jugada es cargada, el número de pasos así lo indica. Se puede reproducir sin problemas y se visualiza la animación de pasos.
Resultado	Correcto.

Tabla 6.3: Descripción del CP-03: Cargar una jugada dinámica.

CP-04	Reproducir una jugada cargada.
Descripción	Se reproducen los pasos de una jugada cargada.
Entrada	Se parte de la jugada cargada del CP-03. Desde la escena Jugadas, y con una jugada cargada, se pulsa el botón play.
Resultado esperado	La animación comienza, mostrando los pasos almacenados de la jugada correctamente.
Resultado	Correcto.

Tabla 6.4: Descripción del CP-04: Reproducir una jugada cargada.

CP-05	Reproducir una jugada en creación.
Descripción	Se reproducen los pasos de una jugada que está siendo creada.
Entrada	Se parte de la jugada que está siendo creada en el CP-01. Desde la escena Jugadas, y con una jugada que está en proceso de creación, se pulsa el botón play.
Resultado esperado	La animación comienza, mostrando los pasos almacenados de la jugada correctamente.
Resultado	Correcto.

Tabla 6.5: Descripción del CP-05: Reproducir una jugada en creación.

CP-06	Movimiento rápido de jugadores.
Descripción	Movimiento de los jugadores de ambos equipos.
Entrada	Desde la escena Jugadas, al crear una jugada se pulsan los botones de movimiento rápido de ambos equipos.
Resultado esperado	Se observa que los jugadores de ambos equipos (según el botón) van del campo, a la zona de banquillo, y de ésta a la posición inicial.
Resultado	Correcto.

Tabla 6.6: Descripción del CP-06: Movimiento rápido de jugadores.

CP-07	Borrar pasos al crear una jugada
Descripción	Se borran los pasos que fueron añadidos a una jugada.
Entrada	Jugada con pasos añadidos como en el caso del CP-01. Se selecciona “Borrar”. En la pantalla de aviso se selecciona “Si”.
Resultado esperado	El contador de pasos vuelve a 1. Ahora no se puede reproducir, y no se puede borrar hasta que se añadan más pasos.
Resultado	Correcto.

Tabla 6.7: Descripción del CP-07: Borrar pasos al crear una jugada.

CP-08	Volver a menú desde Jugadas
Descripción	Desde la escena Jugadas, sin guardar, se vuelve al menú.
Entrada	Jugada con pasos añadidos como en el caso del CP-01. Se selecciona “Borrar”. En la pantalla de aviso se selecciona “Salir”.
Resultado esperado	Se regresa al menú principal de la aplicación.
Resultado	Correcto.

Tabla 6.8: Descripción del CP-08: Volver a menú desde Jugadas.

6.1.2. Listar jugadas

CP-09	Listar jugadas existentes
Descripción	Desde la escena Listar jugadas se visualizan las jugadas almacenadas.
Entrada	Se selecciona la escena Listar jugadas.
Resultado esperado	Se muestran todas las escenas creadas con anterioridad, que son: ▪ Ataque posicional. ▪ Defensa rápida. ▪ Contragolpe derecho.
Resultado	Correcto.

Tabla 6.9: Descripción del CP-09: Listar jugadas existentes.

CP-10	Eliminar una jugada
Descripción	Eliminar una jugada guardada desde la escena Listar jugadas.
Entrada	Se parte del CP-09, y sus tres jugadas desplegadas. Se marca la jugada "Defensa rápida" y se selecciona "Borrar selección". Finalmente se selecciona "Borrar" en la pantalla de confirmación.
Resultado esperado	La jugada "Defensa rápida" desaparece del listado, quedando en éste únicamente: ▪ Ataque posicional. ▪ Contragolpe derecho.
Resultado	Correcto.

Tabla 6.10: Descripción del CP-10: Eliminar una jugada.

CP-11	Eliminar todas las jugadas
Descripción	Desde la escena Listar jugadas, se eliminan todas las jugadas almacenadas.
Entrada	Vista de las dos jugadas dispuestas en tabla como en el CP-10. Se selecciona "Seleccionar todas" y "Borrar selección" a continuación. Finalmente, en cada una de las dos pantallas de confirmación se selecciona "Borrar".
Resultado esperado	El listado se actualiza y ya no se muestra ninguna jugada guardada.
Resultado	Correcto.

Tabla 6.11: Descripción del CP-11: Eliminar todas las jugadas.

6.1.3. Dibujo en pizarra

CP-12	Dibujo de tácticas en pizarra
Descripción	Dibujar gráficamente en la escena Pizarra.
Entrada	Se selecciona la escena Pizarra. Se seleccionan los rotuladores y el grosor deseados y se dibuja una táctica.
Resultado esperado	Representación visual de los trazos de colores, dibujados mediante el control táctil.
Resultado	Correcto.

Tabla 6.12: Descripción del CP-12: Dibujo de tácticas en pizarra.

CP-13	Almacenar captura de pantalla
Descripción	Guardado de una captura de pantalla en el sistema.
Entrada	Se parte de un dibujo realizado previamente como el del CP-12. Se selecciona “Guardar captura”.
Resultado esperado	Se observa un mensaje informativo en la parte superior que dice “Captura guardada”. Se puede localizar, en la carpeta correspondiente de capturas, que la última captura corresponde al dibujo delineado por nosotros mismos.
Resultado	Correcto.

Tabla 6.13: Descripción del CP-13: Almacenar captura de pantalla.

CP-14	Borrado de trazos
Descripción	Se eliminan los trazos dibujados en Pizarra.
Entrada	Se parte de un dibujo realizado previamente como el del CP-12. Se selecciona “Borrar TODO”, y posteriormente, “Borrar”.
Resultado esperado	Las líneas previamente dibujadas desaparecen de la pizarra.
Resultado	Correcto.

Tabla 6.14: Descripción del CP-14: Borrado de trazos.

CP-15	Regreso al menú sin guardar la pizarra
Descripción	Desde la escena Pizarra se intenta volver al menú sin guardar captura.
Entrada	Se parte de un dibujo realizado previamente como el del CP-12. Se selecciona “Volver” y posteriormente “Salir”.
Resultado esperado	Se regresa al menú sin haber guardado una captura. Se puede comprobar que no existe una captura de pantalla con los trazos previamente dibujados.
Resultado	Correcto.

Tabla 6.15: Descripción del CP-15: Regreso al menú sin guardar la pizarra.

6.1.4. Plantilla

CP-16	Visualizar los jugadores de la plantilla
Descripción	Desde la escena Plantilla se visualizan los almacenados en el sistema.
Entrada	Se selecciona la escena .
Resultado esperado	Se muestran todos los jugadores creados con anterioridad, que son: (Formato: Dorsal; Posición; Nombre.) ▪ 9; Central; Raul. ▪ 28; Extremo Der; Aleix.
Resultado	Correcto.

Tabla 6.16: Descripción del CP-16: Visualizar los jugadores de la plantilla.

CP-17	Añadir un jugador a la plantilla
Descripción	Añadir un nuevo jugador al sistema desde la escena Plantilla.
Entrada	Se selecciona la escena Plantilla, y a continuación "Añadir jugador". Se cumplimentan los campos obligatorios: dorsal, posición y número con los siguientes valores: ▪ 10; Pivote; Luis. Se selecciona "Guardar".
Resultado esperado	Se regresa a la tabla de la plantilla y se observa que hay un nuevo jugador, con los datos introducidos. Quedando así la lista actual con los siguientes jugadores: (Formato: Dorsal; Posición; Nombre.) ▪ 9; Central; Raul. ▪ 28; Extremo Der; Aleix. ▪ 10; Pivote; Luis.
Resultado	Correcto.

Tabla 6.17: Descripción del CP-17: Añadir un jugador a la plantilla.

CP-18	Editar un jugador de la plantilla
Descripción	Editar los datos de un jugador del sistema, desde la escena Plantilla.
Entrada	Se selecciona la escena Plantilla, y a continuación "Editar jugador" en la fila del jugador número 10. Se cambia el dorsal 10 por el nuevo dorsal 13, se cambia el nombre por José Luis y la posición a Portero. Se selecciona "Guardar".
Resultado esperado	Se regresa a la tabla de la plantilla y se observa que hay un cambio en el último jugador, con los datos actualizados. Quedando así la lista actual con los siguientes jugadores: (Formato: Dorsal; Posición; Nombre.) ▪ 9; Central; Raul. ▪ 28; Extremo Der; Aleix. ▪ 13; Portero; José Luis.
Resultado	Correcto.

Tabla 6.18: Descripción del CP-18: Editar un jugador de la plantilla.

CP-19	Mensaje de error al crear un jugador
Descripción	Editar incorrectamente un jugador del sistema, desde la escena Plantilla.
Entrada	Se selecciona la escena Plantilla, y a continuación "Editar jugador" en la fila del jugador número 13. Se cambia el dorsal 13 por el nuevo dorsal 9. Se selecciona "Guardar".
Resultado esperado	Se observa que aparece un mensaje informativo que indica que ya existe un jugador con dicho dorsal, instando a cambiarlo.
Resultado	Correcto.

Tabla 6.19: Descripción del CP-19: Mensaje de error al crear un jugador.

CP-20	Mensaje de aviso al editar un jugador
Descripción	Salir sin guardar al editar un jugador del sistema.
Entrada	Se selecciona la escena Plantilla, y a continuación "Editar jugador" en la fila del jugador número 13. Se cambia el dorsal 13 por el nuevo dorsal 90. Se selecciona "Salir". Se muestra una pantalla de confirmación, que impide salir sin guardar los cambios, se selecciona "Volver".
Resultado esperado	Se reresa a la plantilla y se observa que los jugadores de la tabla no han sufrido modificación alguna.
Resultado	Correcto.

Tabla 6.20: Descripción del CP-20: Mensaje de aviso al editar un jugador.

CP-21	Eliminar un jugador de la plantilla
Descripción	Eliminar un jugador del sistema, desde la escena Plantilla.
Entrada	Se selecciona la escena Plantilla, y a continuación “Editar jugador” en la fila del jugador número 13. Se selecciona “Borrar jugador”. En la pantalla de confirmación se selecciona “Borrar”.
Resultado esperado	Se regresa a la tabla de la plantilla y se observa que ya no aparece el último jugador, José Luis, quien tenía el dorsal 9. Quedando así la lista actual con los siguientes jugadores: (Formato: Dorsal; Posición; Nombre.) ▪ 9; Central; Raul. ▪ 28; Extremo Der; Aleix.
Resultado	Correcto.

Tabla 6.21: Descripción del CP-21: Eliminar un jugador de la plantilla.

CP-22	Eliminar a todos los jugadores de la plantilla
Descripción	Eliminar a todos los jugadores del sistema, desde la escena Plantilla.
Entrada	Se selecciona la escena Plantilla, y a continuación “Borrar TODO”. Se selecciona “Borrar” en la primera pantalla de confirmación. En la segunda pantalla de confirmación, también se selecciona “Borrar”.
Resultado esperado	Se regresa a la tabla de la plantilla y se observa que ya no aparece ningún jugador.
Resultado	Correcto.

Tabla 6.22: Descripción del CP-22: Eliminar a todos los jugadores de la plantilla.

6.1.5. Ajustes

CP-23	Cambiar el color de los equipos
Descripción	Desde la escena Ajustes se modifica el color de las equipaciones de ambos equipos.
Entrada	Se selecciona la escena Ajustes, y a continuación se modifica el color del equipo local a verde, y el del equipo rival a morado. Se selecciona “Guardar cambios”.
Resultado esperado	Se observan los cambios en la “Visualización previa”. Se navega a la escena Jugadas y se observa que el color también fue modificado aquí.
Resultado	Correcto.

Tabla 6.23: Descripción del CP-23: Cambiar el color de los equipos.

CP-24	Cambiar el color de los equipos sin guardar
Descripción	Desde la escena Ajustes se modifica, sin efecto, el color de las equipaciones de ambos equipos.
Entrada	Se selecciona la escena Ajustes, y a continuación se modifica el color del equipo local a rosa, y el del equipo rival a blanco. Se selecciona “Salir”.
Resultado esperado	Se observan los cambios en la “Visualización previa”. Se navega a la escena Jugadas y se observa que el color no se ha visto alterado.
Resultado	Correcto.

Tabla 6.24: Descripción del CP-24: Cambiar el color de los equipos sin guardar.

6.1.6. Selección de idioma

CP-25	Cambiar el idioma de la aplicación
Descripción	Desde la escena Menú se modifica el idioma de la aplicación.
Entrada	Se selecciona el menú desplegable y desde éste, se selecciona el idioma "English".
Resultado esperado	Se observa que el idioma de los textos en todas las escenas han cambiado de español a inglés.
Resultado	Correcto.

Tabla 6.25: Descripción del CP-25: Cambiar el idioma de la aplicación.

CP-26	Cambiar el idioma de la aplicación al actual
Descripción	Desde la escena Menú se modifica el idioma de la aplicación, al mismo.
Entrada	Se selecciona el menú desplegable y desde éste, se selecciona el idioma "Español".
Resultado esperado	Se observa que el idioma de los textos en todas las escenas no se ven modificados y que siguen en español.
Resultado	Correcto.

Tabla 6.26: Descripción del CP-26: Cambiar el idioma de la aplicación al actual.

6.2. Pruebas con usuarios

Uno de los objetivos claves era lograr que la aplicación fuese muy usable, cuya importancia se declaró en el subcapítulo 4.5.1. Para ello, lo mejor es poder disponer de la ayuda de usuarios reales que prueben la aplicación final y comenten sus dificultades en el uso, posibles mejoras, y puntos fuertes encontrados.

Estas pruebas se han realizado con la tutora principalmente, aunque también con familia y amigos, e incluso con una entrenadora de balonmano real. Se han realizado desde etapas tempranas, en concreto, cada vez que se lograba implementar con éxito alguna funcionalidad ya que detectar en una etapa temprana los fallos es crucial para evitar repetirlos y no continuarlos.

Las conclusiones extraídas fueron tenidas en cuenta en el desarrollo, siendo alguna de ellas:

- **Choque entre jugadores.** En una primera versión, los jugadores presentaban barreras con otros jugadores y el balón, de forma que se podían producir pequeños movimientos no deseados, y no se podían juntar como barrera. Se eliminan las fuerzas por lo que ahora ningún elemento choca, y el balón queda siempre por cima.
- **Movimiento rápido de jugadores.** Por sugerencia de un usuario, al intentar realizar una jugada sin el equipo rival, se solicita una opción para mover todos los jugadores rivales. Se aplica un botón con esta funcionalidad para ambos equipos.
- **Rotulador de pizarra.** Por defecto el rotulador negro era el inicial, y el selector de rotulador que indica cuál está seleccionado también es negro, por lo que se veía mal. Además, no se distingue mucho el negro de las líneas del campo, por lo que se decide empezar por defecto, con el color azul.
- **Volver atrás con botón.** Para mayor versatilidad se decide implementar que la tecla escape o bien el botón atrás de android tomen la funcionalidad del botón atrás.

Capítulo 7

Conclusiones y líneas de trabajo futuras

Una vez finalizado el proyecto, se pueden extraer valiosas conclusiones acerca del mismo así como posibles mejoras implementar en un futuro.

7.1. Conclusiones

Todos los objetivos establecidos en el subcapítulo 1.3 han sido cumplidos de manera satisfactoria. A nivel de implementación de hecho, se lograron más objetivos que los inicialmente propuestos. A grandes rasgos estos han sido:

- **Gestión de proyectos.** Familiarizarme con todo el proceso de desarrollo software, si bien no en una profundidad extrema, he podido ampliar mi punto de vista, y conocimientos, así como en el uso de GitLab, y en la gestión de riesgos y tiempos.
- **Planificación.** La planificación es un elemento clave, y este TFG me ha dado pautas para mi futuro laboral. El tiempo excedido en la planificación inicial ha sido fruto de la inexperiencia en la gestión, y en la novedad que supuso C# y el manejo de Unity. En todo caso, considero valiosa la experiencia y tendré en cuenta mis fallos para evitarlos en el futuro.
- **Despliegue en Android.** Se ha logrado crear una aplicación de gestión de balonmano para Android (y también en Windows y Mac), que facilita al entrenador crear tácticas y enseñarlas a sus jugadores.
- **Unity.** He aprendido algunas de las funcionalidades que brinda este motor de videojuegos, aunque no hayan sido empleadas, comola cantidad de soporte para juegos 3D de las que dispone.
- **Conocimiento del balonmano.** Este trabajo también me ha hecho darme cuenta de los muchos aspectos a considerar en cualquier deporte, y en particular, en el balonmano. Destacar cómo el saber más acerca de un tema, puede llevar a un equipo de desarrollo a incrementar su planificación inicial.

7.2. Líneas de trabajo futuras

A continuación, expongo algunas funcionalidades que podrían implementarse en la aplicación con la finalidad de hacerla más completa y útil. Debido a la atención que requieren algunas de ellas, podría resultar más interesante la creación de otra aplicación, que permita a un segundo entrenador o ayudante, estar al tanto.

- **Control del partido.** Mediante un botón flotante que puede estar presente en cada sección de la aplicación, se permite registrar la actividad de un partido, y las acciones que ocurren en el mismo por medio de diferentes acciones. Además de mostrar un cronómetro con el tiempo jugado, las acciones se añadirían teniendo esto en cuenta, por medio de botones simples, que lanzasen preguntas al entrenador, como por ejemplo:
  - **Goles.** Tras pulsar el botón se pregunta qué equipo realiza el gol, en caso de que sea el propio, aparecerá un desplegable con los jugadores, y finalmente permitirá diferenciar si fue normal, por falta o por penalti.
  - **Faltas o amonestaciones.** De manera análoga preguntaría si el equipo la ha recibido o la ha provocado, preguntando en todo caso por el jugador involucrado, y el tipo de amonestación: ninguna, verbal, amarilla o roja.
  - **Tiros, fintas, paradas, cambios...** Todo aquello que pudiera implementarse sin ocupar un espacio excesivo en pantalla.
- **Formaciones para jugadas.** Una de las opciones que quedaron en el tintero en cuanto a las jugadas, fue permitir al entrenador elegir la formación inicial de los jugadores, e incluso crear las suyas propias. Se podría elegir desde un menú desplegable en la propia escena, o desde ajustes, para no saturar.
- **Recolección de estadísticas.** En estrecha relación con el punto anterior, la recolección de los datos de un partido se pueden recolectar y mostrar en diversos gráficos para ayudar al entrenador a obtener conclusiones. El análisis estadístico de los deportes es desde hace décadas, sinónimo de éxito.
- **Sugerencias tácticas.** En una propuesta más exótica, cabría la posibilidad de, mediante machine learning, proponer al entrenador cambios en la plantilla inicial, o asociaciones de jugadores con los que el equipo gana más, por ejemplo.

Bibliografía

[1] J. Alrow e I. Neustadt, *UML 2 and the Unified Process*, 2.ª ed. Addison-Wesley Professional, 2005.

[18] M. Gonzalo, Y. Crespo, C. Hernández y A. Martínez, *Apuntes de Interacción Persona Computadora*. Universidad de Valladolid, 2018, Curso 2018-2019.

[19] C. Hernández, M. Barrio y H. Ortega, *Apuntes de Profesión y Sociedad*. Universidad de Valladolid, 2020, Curso 2020-2021.

[20] B. Hughes y M. Cotterell, *Software Project Management*, 5.ª ed. McGraw Hill, 2009.

[25] M. Á. Laguna, *Apuntes de Desarrollo Basado en Componentes y Servicios*. Universidad de Valladolid, 2020, Curso 2020-2021.

[26] C. Larman, *UML y Patrones. Introducción al Análisis y Diseño Orientado a Objetos y al Proceso Unificado*, 2.ª ed. Prentice Hall, 2004.

[37] F. Prieto, *Apuntes de Fundamentos de Ingeniería de Software*. Universidad de Valladolid, 2018, Curso 2018-2019. Tema: 2.

[41] I. Sommerville, *Ingeniería del software*, 9.ª ed. Pearson, 2011, Capítulo 2.

[55] J. M. Vegas, *Apuntes de Planificación y Gestión de Proyectos*. Universidad de Valladolid, 2020, Curso 2020-2021.

Webgrafía

[2] Astah. (2021). «Astah Professional Software.» Accedido: 2021-03-07, dirección: https://astah.net/products/astah-professional/.

[3] BLUELINDEN. (2016). «Vídeo demostrativo de creación de jugadas de la App: Pizarra Táctica.» Accedido: 2021-02-27, dirección: https://www.youtube.com/watch?v=cL9qELNLS6o&t=4s&ab_channel=BluelindenApps.

[4] BLUELINDEN. (2019). «(Aplicación Android) Pizarra Táctica: Balonmano.» Accedido: 2021-02-27, dirección: https://play.google.com/store/apps/details?id=com.bluelinden.coachboardhandball.

[5] BOE. (2018). «XVII Convenio colectivo estatal de empresas de consultoría y estudios de mercado y de la opinión pública.» Accedido: 2021-03-06, dirección: https://www.boe.es/boe/dias/2018/03/06/pdfs/BOE-A-2018-3156.pdf.

[6] BOE. (2020). «Real Decreto-ley 28/2020, de 22 de septiembre, de trabajo a distancia.» Accedido: 2021-03-06, dirección: https://www.boe.es/diario_boe/txt.php?id=BOE-A-2020-11043.

[7] Brackeys. (2017). «Tutoriales: Unity Begginer Tutorials.» Accedido: 2021-03-16, dirección: https://www.youtube.com/playlist?list=PLPV2KyIb3jR5QFsefuO2RlAgWEz6EvVi6.

[8] I. Buckley. (2019). «Lenguajes de desarrollo para Unity.» Accedido: 2021-03-10, dirección: https://www.makeuseof.com/tag/unity-game-development-languages/.

[9] C. B. Bueno y F. O. G. Rubio. (2015). «Gestión de Proyectos Software.» Accedido: 2021-03-04, dirección: https://ocw.unican.es/pluginfile.php/274/course/section/194/GP-t5.pdf.

[10] CoachYouths. (2021). «Basketball Playbook Designer.» Accedido: 2021-02-27, dirección: https://www.basketballplaybookdesigner.com/.

[11] CodeProject. (2014). «What is CodeLens?» Accedido: 2021-03-11, dirección: https://www.codeproject.com/Articles/794766/What-is-CodeLens.

[12] H. Costa. (2017). «Tutoriales: Juego Plataformas 2D en Unity 5.» Accedido: 2021-03-11, dirección: https://www.youtube.com/playlist?list=PLiplYDjUMtti5bWJ1Ugystr-vUg8B5EuP.

[13] ElAndroideFeliz. (2020). «Cómo instalar aplicaciones en APK en un móvil Android.» Accedido: 2021-06-12, dirección: https://www.xatakandroid.com/tutoriales/como-instalar-aplicaciones-en-apk-en-un-movil-android.

[14] ElAndroideFeliz. (2020). «Cómo instalar archivos o aplicaciones APK en Android.» Accedido: 2021-06-12, dirección: https://elandroidefeliz.com/instalar-aplicaciones-archivos-apk-android/#:~:text=Abre%20el%20el%20men%C3%BA%20de,%2D%3E%20Instalar%20aplicaciones%20desconocidas%C2%AB..

[15] Fotocasa. (2021). «Análisis del coste del alquiler.» Accedido: 2021-03-06, dirección: https://www.fotocasa.es/es/alquiler/pisos/valladolid-provincia/todas-las-zonas/l.

[16] GitLab. (2021). «Issue Boards.» Accedido: 2021-03-09, dirección: https://docs.gitlab.com/ee/user/project/issue_board.html.

[17] GitLab. (2021). «What is GitLab?» Accedido: 2021-03-09, dirección: https://about.gitlab.com/what-is-gitlab/.

[21] E. de Ingeniería Informática de Valladolid. (2017). «Guía del alumno del TFG (Estructura, Formato, Documentación).» Accedido: 2021-02-18, dirección: https://www.inf.uva.es/wp-content/uploads/2013/01/00-GuiaAlumnoTFG_2017.pdf.

[22] E. de Ingeniería Informática de Valladolid. (2021). «Calendario de depósito y defensa de TFG 2020-2021.» Accedido: 2021-02-26, dirección: https://www.inf.uva.es/wp-content/uploads/2014/09/CalendarioPresentacionTFG.pdf.

[23] InnerSloth. (2021). «Among Us.» Accedido: 2021-03-10, dirección: https://innersloth.com/gameAmongUs.php.

[24] Kanbanize. (2020). «¿Qué es un tablero Kanban?» Accedido: 2021-03-07, dirección: https://kanbanize.com/es/recursos-de-kanban/primeros-pasos/que-es-tablero-kanban.

[27] masterD. (2020). «Unity ¿Qué es y para qué sirve?» Accedido: 2021-03-08, dirección: https://www.masterd.es/blog/que-es-unity-3d-tutorial/.

[28] Mediatonic. (2021). «Fall Guys: Ultimate Knockout.» Accedido: 2021-03-10, dirección: https://fallguys.com/.

[29] Microsoft. (2021). «Documentación de C#.» Accedido: 2021-03-10, dirección: https://docs.microsoft.com/es-es/dotnet/csharp/.

[30] Microsoft. (2021). «Gestión de proyectos con MS Project.» Accedido: 2021-03-07, dirección: https://www.microsoft.com/es-es/microsoft-365/project/project-management-software.

[31] Microsoft. (2021). «Visual Studio Code and Unity.» Accedido: 2021-03-11, dirección: https://code.visualstudio.com/docs/other/unity.

[32] mola. (2017). «(Aplicación Android) Field Hockey Dood.» Accedido: 2021-02-27, dirección: https://play.google.com/store/apps/details?id=air.FieldHockeyDoodFree.

[33] MonoProject. (2021). «Cross Platform Mono for .NET.» Accedido: 2021-03-08, dirección: https://www.mono-project.com/.

[34] F. Mortals. (2015). «Is Database Really Necessary in Unity.» Accedido: 2021-04-24, dirección: https://gamedev.stackexchange.com/questions/109080/is-database-necessary.

[35] F. Mortals. (2019). «Free Draw Unity Asset.» Accedido: 2021-05-16, dirección: https://assetstore.unity.com/packages/tools/painting/free-draw-simple-drawing-on-sprites-2d-textures-113131.

[36] E. Pais. (2021). «Calculadora de salario neto.» Accedido: 2021-03-06, dirección: https://cincodias.elpais.com/herramientas/calculadora-sueldo-neto/#tabla_resultados.

[38] RFEBM. (2020). «Reglamento de partidos y competiciones.» Accedido: 2021-02-16, dirección: https://www.rfebm.com/sites/default/files/documentos/rpc_modificacion_asamblea_-_web.pdf.

[39] S. Rodríguez. (2020). «Los videojuegos facturarán 159 mil millones de dólares en 2020.» Accedido: 2021-02-29, dirección: https://comunicacionmarketing.es/comunicacion/10/07/2020/los-videojuegos-facturaran-159-mil-millones-de-dolares-en-2020/15290.html.

[40] G. Software. (2021). «(Aplicación Android) TacticalPad Futsal Handball.» Accedido: 2021-02-27, dirección: https://play.google.com/store/apps/details?id=com.clansoft.tacticalpadfutsallite.

[42] Techopedia. (2015). «What is Intellisense?» Accedido: 2021-03-11, dirección: https://www.techopedia.com/definition/24580/intellisense.

[43] TechTarget. (2019). «What is WebGL?» Accedido: 2021-02-28, dirección: https://whatis.techtarget.com/definition/WebGL.

[44] Unity. (2020). «Unity MANUAL.» Accedido: 2021-03-11, dirección: https://docs.unity3d.com/Manual/index.html.

[45] Unity. (2020). «Unity Manual: Asets Workflow.» Accedido: 2021-03-11, dirección: https://docs.unity3d.com/Manual/AssetWorkflow.html.

[46] Unity. (2020). «Unity Manual: Canvas.» Accedido: 2021-03-11, dirección: https://docs.unity3d.com/2020.1/Documentation/Manual/UICanvas.html.

[47] Unity. (2020). «Unity Manual: GameObject.» Accedido: 2021-03-11, dirección: https://docs.unity3d.com/560/Documentation/Manual/class-GameObject.html.

[48] Unity. (2020). «Unity Manual: Prefabs.» Accedido: 2021-03-11, dirección: https://docs.unity3d.com/Manual/Prefabs.html.

[49] Unity. (2020). «Unity Manual: Sprites.» Accedido: 2021-03-11, dirección: https://docs.unity3d.com/Manual/Sprites.html.

[50] Unity. (2021). «Motor de videojuegos Unity.» Accedido: 2021-03-08, dirección: https://unity.com/es.

[51] Unity. (2021). «Plataformas objetivo de Unity.» Accedido: 2021-03-10, dirección: https://unity.com/es/features/multiplatform.

[52] Unity. (2021). «Unity Hub.» Accedido: 2021-03-10, dirección: https://docs.unity3d.com/Manual/GettingStartedInstallingHub.html.

[53] Ustwogames. (2021). «Monument Valley.» Accedido: 2021-03-10, dirección: https://www.monumentvalleygame.com/mv2.

[54] U. de Valladolid. (2021). «Guía docente Trabajo de Fin de Grado (mención en Ingeniería de Software).» Accedido: 2021-02-23, dirección: https://albergueweb1.uva.es/guia_docente/uploads/2020/545/46976/1/Documento.pdf.

[56] C. Villanueva. (2018). «¿Qué es y para qué sirve un diagrama de Gantt?» Accedido: 2021-02-24, dirección: https://blog.teamleader.es/diagrama-de-gantt.

[57] I. Villaumbrales. (2019). «Testing, la importancia sobre la fase de test de software.» Accedido: 2021-02-24, dirección: https://www.hiberus.com/crecemos-contigo/testing-fase-de-testeo-de-software/.

Apéndice A: Manual de instalación

En este manual, se describe una breve guía de instalación según el sistema operativo. Teniendo en cuenta los requisitos (por sistema operativo) que se deben cumplir, así como explicaciones detalladas del proceso, se busca que el usuario final no tenga problemas en la instalación de la aplicación. Puede encontrar los archivos de instalación para cada sistema desde el siguiente enlace.

Si quiere desplegar el proyecto en el IDE de Unity, puede descargar la carpeta UnityProject desde el siguiente enlace. Si no sabe cómo, en dicho enlace encontrará el fichero README.md, donde se especifican los pasos.

A.1. Instalación en sistemas Android

A.1.1. Requisitos

- Versión mínima: Android 4.4.
- Almacenamiento interno, al menos 50 MB.

A.1.2. Proceso de instalación en Android

Conocer la versión de Android [13] [14] le permitirá seguir esta guía, de manera más acorde a su dispositivo. La versión mínima soportada es la 4.4, si es menor, no podrá ejecutarla. Si desconoce cuál es su versión, puede consultarla con los siguientes pasos (reflejados en la figura 1):

- Abrir la aplicación Ajustes.
- Pulsar en Acerca del teléfono/dispositivo (normalmente, al final).
- Podrá visualizarlo en “Versión de Android”.

Si bien se ha realizado un esfuerzo por elaborar una guía general, ha de saber que puede haber variaciones en función de la capa de personalización del fabricante, por lo que los pasos indicados pueden variar.

También ha de descargar el fichero, desde la carpeta ejecutables del repositorio, EntrenaBal.apk. Los ficheros .apk son los paquetes de instalación de cualquier aplicación Android, y contienen todos los datos que la forman. Puede tenerlo en su sistema de ficheros, o descargarlo desde internet y proceder con su instalación. Al tratarse de ficheros ejecutables, el sistema nos avisa para protegernos, una apk no ha sido revisada en busca de malware, y

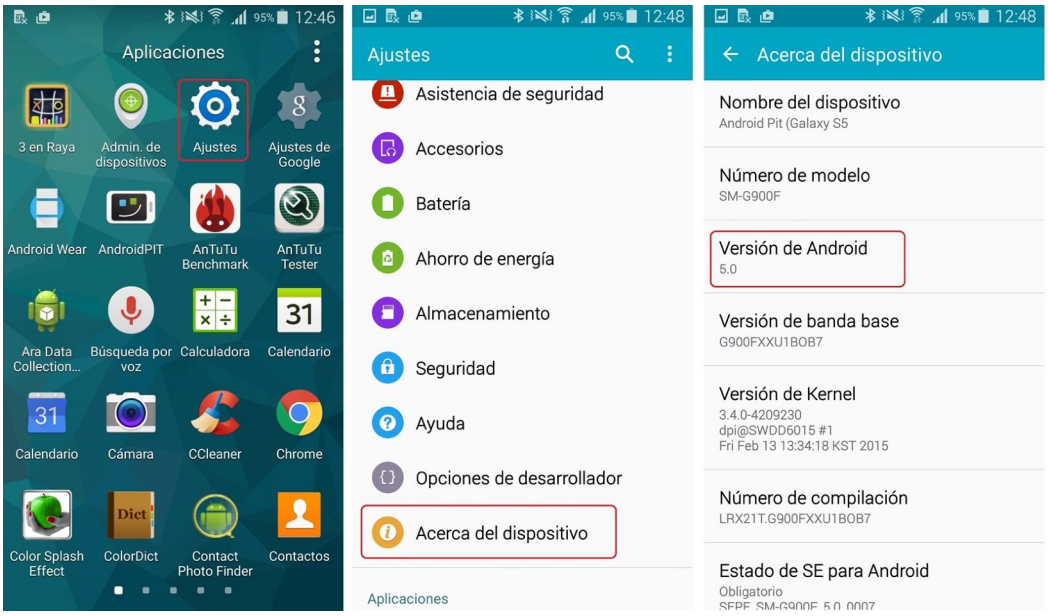

Figura 1: Consulta de versión Android. Fuente: NextPit.es

hay que tener cuidado de la fiabilidad de aquellas que descarguemos.

Finalmente, reseñar que no puede instalar esta aplicación desde el Google Play Store (como estará habituado), puesto que no ha sido publicada en el mismo, por carecer de la licencia de desarrollador requerida para ello.

- **Android 7, o inferior**

  En caso de que la versión de Android de su dispositivo sea 7, o anterior (4.4, 5.1, ...), para habilitar la instalación de este tipo de apps debemos seguir los siguientes pasos (figura 2):

  - Abrir la aplicación Ajustes (o Configuración).
  - Pulsar en Seguridad.
  - Activar la pestaña "Fuentes desconocidas".

  Una vez hecho esto, podremos abrir e instalar cualquier archivo .apk en nuestro dispositivo (figura 3). Una vez en la ubicación de la descarga, pulsamos sobre este y preguntará por instalar la aplicación.

- **Android 8, o superior**

  Desde Android 8 en adelante, se debe dar permiso a cada app que se quiera instalar, en contraposición a las versiones anteriores, buscando así una mayor seguridad para el usuario.

  Por tanto, en Android 10, 9 y 8, los permisos para instalar APKs se otorgan de manera individual. Si descargamos los archivos APK desde el navegador, entonces tendremos que darle permisos especiales al navegador (como puede ser Google Chrome) para que pueda instalar este tipo de archivos en el dispositivo.

  Los pasos (figura 4) a seguir serían:

  - Abrir la aplicación Ajustes.
  - Seleccionar Aplicaciones y notificaciones.
  - Pulsar en Acceso especial de aplicaciones.

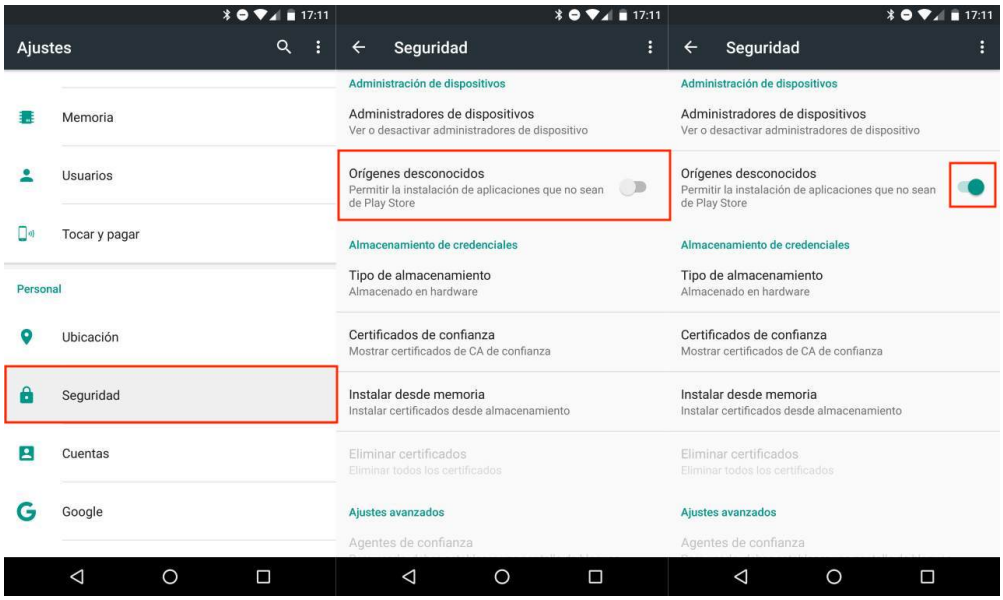

Figura 2: Permisos de instalación, Android 7 o inferior. Fuente: [14]

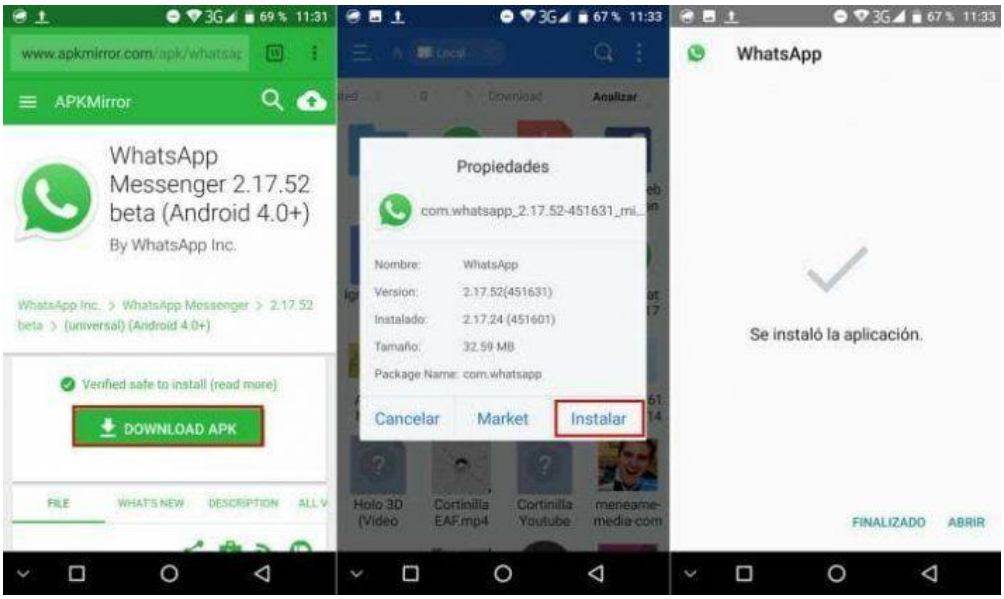

Figura 3: Instalación de una apk, Android 7 o inferior. Fuente: [14]

- Activar la pestaña “Instalar aplicaciones desconocidas”.

Esto nos llevará a un listado (figura 5) donde veremos todas las aplicaciones que tenemos instaladas en el dispositivo. Seleccione el navegador (o la aplicación desde la cual vaya a instalarla) y marque la opción “Autorizar descargas de esta fuente”. Esto mismo puede llevarse a cabo con más de una aplicación, por lo que podemos tener varias fuentes de instalación.

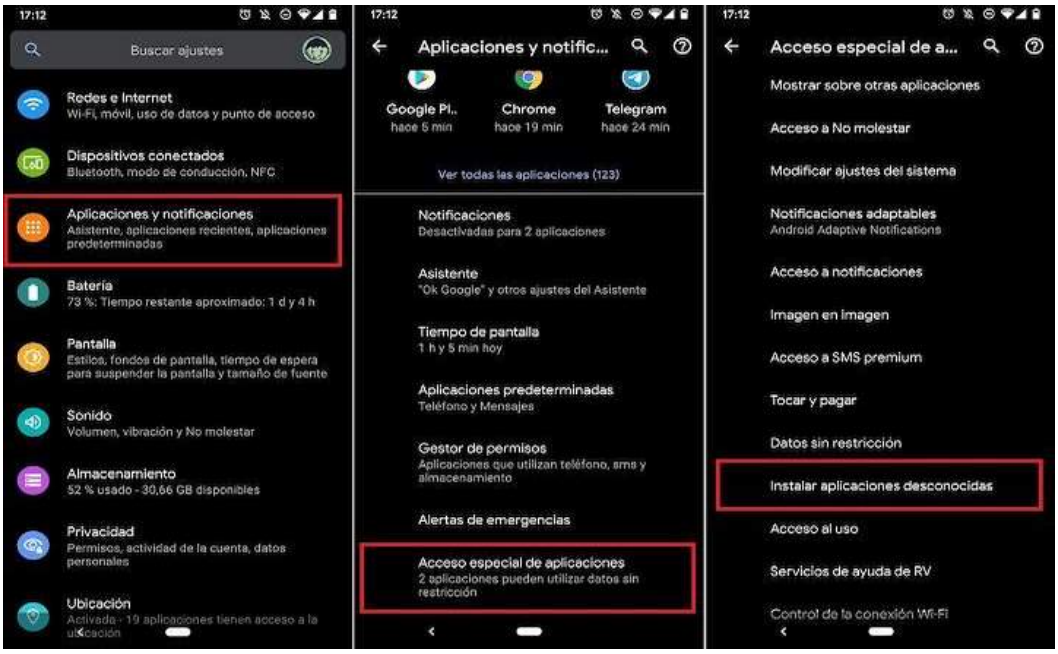

Figura 4: Permisos de instalación, Android 8 o superior. Fuente: [14]

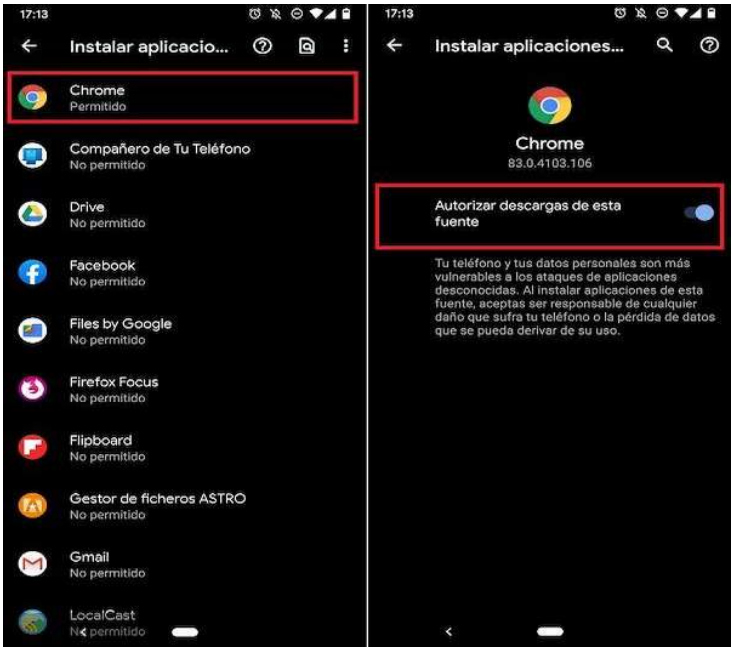

Figura 5: Instalación de una apk, Android 8 o superior. Fuente: [14]

A.2. Windows

A.2.1. Requisitos

- Versión mínima, Windows 10.
- Almacenamiento interno, al menos 70 MB.

A.2.2. Proceso de instalación en Windows

La versión mínima de funcionamiento de esta aplicación es Windows 10 (si tiene la versión 8, 7, XP, 98... no podrá ejecutarla). Si no conoce la arquitectura de su procesador, 32 o 64 bits, puede seguir los pasos que aparecen en esta página web.

Una vez conozca la arquitectura de su Windows (64 bits, lo más probable), debe descargar desde el repositorio, en la carpeta Ejecutables, el fichero acorde a su máquina: Windows32bits.zip o Windows64bits.zip. Deberá descomprimirlo, si no sabe cómo, puede seguir esta guía.

Indistintamente, encontrará dentro de la carpeta extraída, el fichero EntrenaBal.exe el cual se ejecutará inmediatamente al hacer doble click, o haciendo click derecho con el ratón - Abrir. En función de su configuración de seguridad, puede que aparezca un mensaje de aviso del firewall de Windows, esto se debe a que Windows no reconoce al desarrollador, pero puede aceptar sin miedo.

A.3. Instalación en sistemas macOS

A.3.1. Requisitos

- Versión mínima, macOS 10.15 Catalina.
- Procesador Intel de 64 bits de arquitectura (el nuevo Apple Silicon m1 NO es compatible).
- Almacenamiento interno, al menos 70 MB.

A.3.2. Proceso de instalación en Mac

Debido a que la aplicación no ha sido publicada en la App Store de Mac, ni proviene de un desarrollador conocido, deberá cambiar la configuración de seguridad de su mac para poder ejecutarla. A continuación, se exponen los pasos seguros para ello.

Abra la aplicación Terminal, puede hacerlo escribiendo el nombre en spotlight (command + barra espaciadora) o desde la carpeta Aplicaciones. Una vez allí, escriba, sin comillas: “sudo spctl –master-disable” , acto seguido, introduzca la contraseña del ordenador.

Proceda a descargar la carpeta MacOS contenida en la carpeta Ejecutables del repositorio, para poder continuar. Desde Finder (el explorador de archivos), podrá encontrar en la carpeta descargada el fichero EntrenaBal.app, simplemente haga clic derecho y seleccione “Abrir”, inmediatamente se lanzará la aplicación.

Por seguridad, una vez revisada la aplicación, le aconsejo que revierta la opción, no me hago responsable de daños causados por la instalación posterior de otras aplicaciones. Para ello abra la aplicación Preferencias de sistema - Seguridad y privacidad - General - Abra el candado que aparece en la esquina inferior izquierda - Introduzca su contraseña - Seleccione la opción “App Store y desarrolladores identificados” - Cierre el candado.

Apéndice B: Manual de uso

En los siguientes subcapítulos se expondrán todas las funcionalidades de la app paso a paso con la finalidad de guiar a los usuarios. Se adjuntarán capturas de pantalla de la aplicación final (hay diferencias con los prototipos) para situar al usuario cuando se le especifiquen las acciones a realizar. En casi cualquier pantalla podrá pulsar la tecla Esc, si se encuentra en Windows o Mac, o bien la flecha atrás de Android, para navegar hacia atrás o salir de la app.

B.1. Menú principal e idiomas

Nada más arrancar la aplicación se nos presenta la pantalla de menú principal, figura 6. Desde el menú puede navegar al resto de funcionalidades de la app (siguientes subcapítulos), además de otras funcionalidades.

Figura 6: EntrenaBal: Menú principal

En la esquina superior derecha se puede observar un menú desplegable. He aquí la funcionalidad de cambio de idioma de la aplicación. Si pulsa sobre el desplegable podrá elegir el idioma en el que se muestra la app, pudiendo elegir actualmente entre: español (por defecto), inglés, francés y alemán. Su elección se aplicará al instante y será recordada mientras no lo cambie.

También puede salir de la aplicación pulsando en el botón rojo “Salir” (esquina inferior derecha).

B.2. Jugadas

Desde el menú principal, figura 6, se puede acceder, haciendo clic en “Jugadas”, a la pantalla de creación y visualización de jugadas dinámicas. Éstas pueden visualizarse en cualquier momento pulsando el botón play que aparece abajo, ya sea mientras se crea una jugada o tras cargar una que previamente hayamos guardado.

Una vez dentro, figura 7, puede observar un campo de balonmano y 7 jugadores por equipo, además de un balón. Puede disponer estos elementos donde quiera arrastrando con clic izquierdo o con el dedo, en el caso de los dispositivos de pantalla táctil. Una vez los jugadores se encuentren en la siguiente etapa de la jugada, pulse el botón “+” situado arriba en el centro, esto añadirá un paso a la jugada, pudiendo añadir tantos como quiera.

Figura 7: EntrenaBal: Jugadas dinámicas

Como podrá observar, a los laterales se encuentran unos botones (azul y rojo), figura 8, que permiten mover rápidamente a todos los jugadores de cada equipo fuera del campo. Esto puede ser de utilidad en caso de querer realizar una jugada sin rivales, o si involucra a pocos jugadores, no teniendo por tanto, que mover manualmente a muchos jugadores.

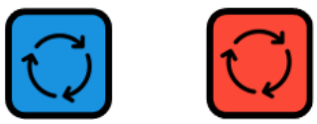

Figura 8: EntrenaBal: Botones de movimiento de equipo

En caso de que se equivoque al añadir pasos o quiera comenzar de cero, puede borrarlos pulsando el botón rojo “Borrar”, se le mostrará una pantalla como la figura 9, para evitar borrados por accidente.

Una vez considere finalizada su jugada, puede guardarla pulsando el botón blanco “Guardar”, desde el cual,

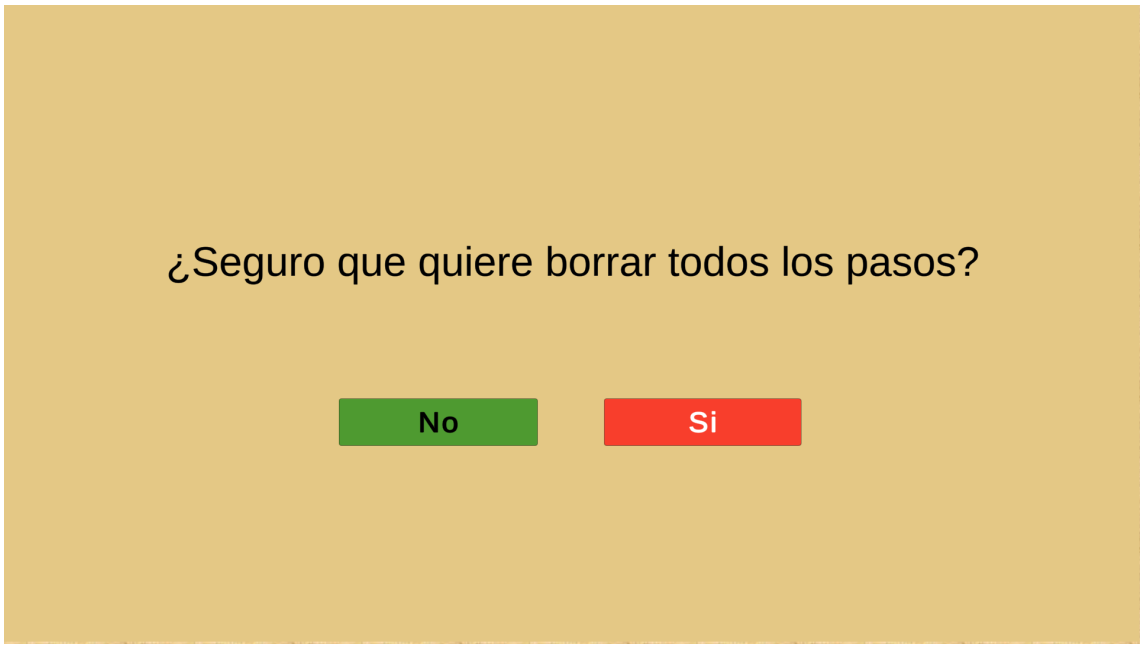

Figura 9: EntrenaBal: Borrado de pasos

figura 10, se solicitará un nombre identificativo para guardar su jugada. Como mínimo tendrá 1 carácter y como máximo, 25. Si ya existe una jugada con el nombre especificado NO podrá guardar, apareciendo un mensaje informativo en la figura 10.

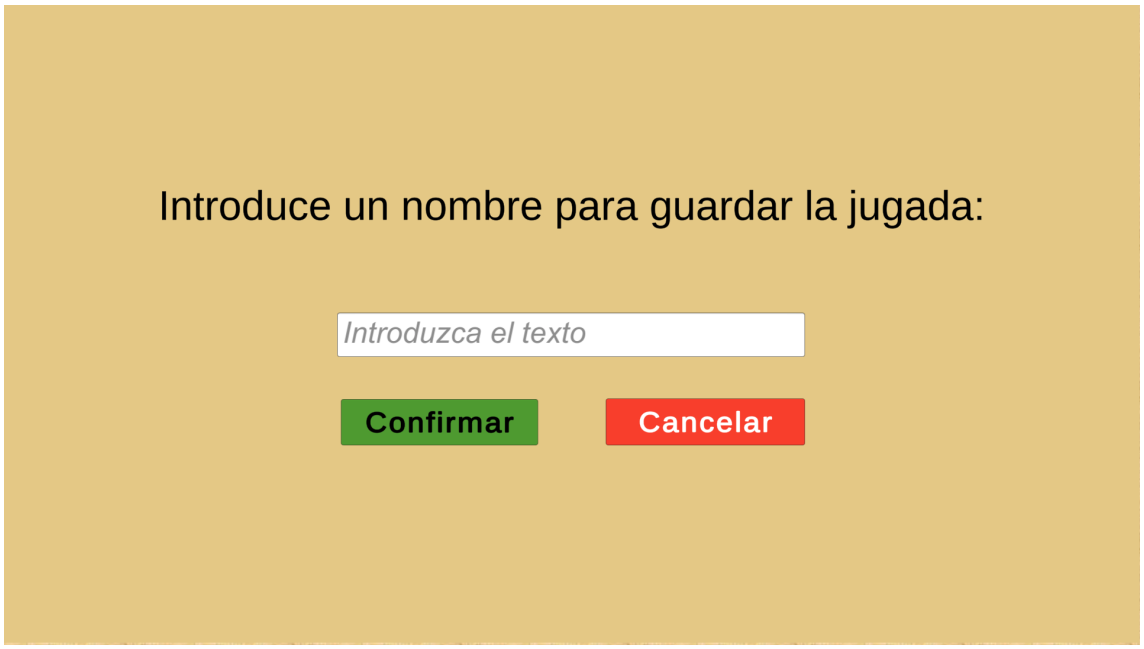

Figura 10: EntrenaBal: Guardar jugada

De manera similar, podrá cargar una jugada previamente guardada pulsando el botón verde “Cargar jugada”, lo que mostrará el listado de jugadas guardadas (en la figura 11 puede ver un ejemplo) en el dispositivo, permitiéndole seleccionar la que quiera pulsando en el botón verde “Cargar” de la correspondiente fila.

Una vez cargada, puede reproducirla pulsando el botón verde play; cada vez que lo pulse, la animación volverá a comenzar. **Recalcar** que tras cargar una jugada, pueden añadirse pasos y guardarla como una nueva.

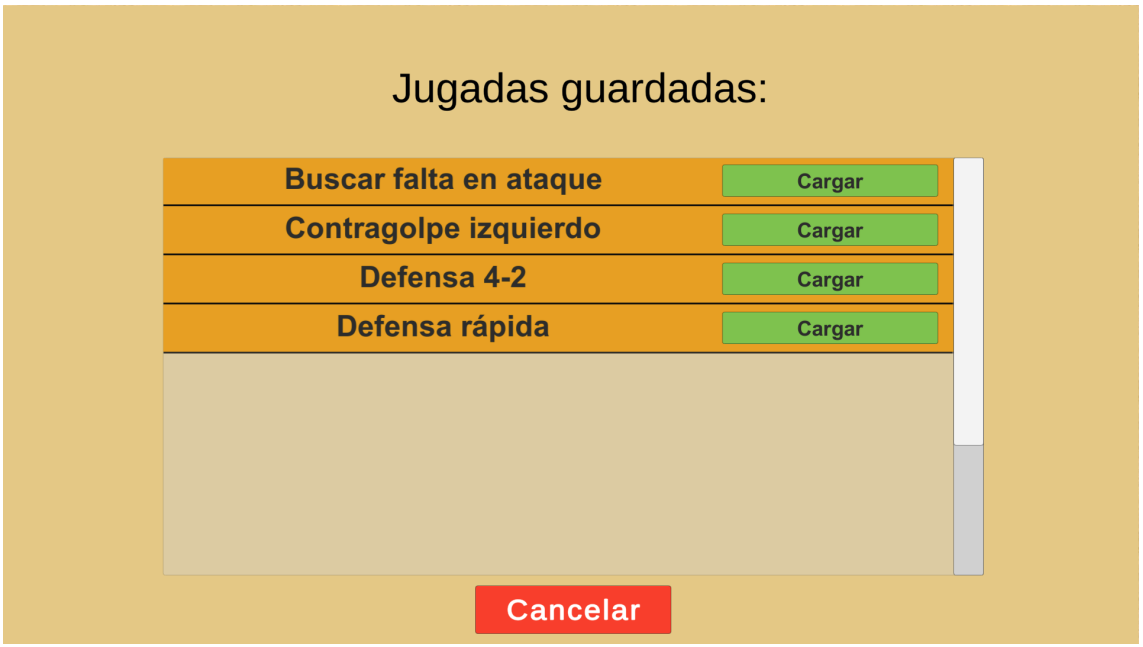

Figura 11: EntrenaBal: Cargar una jugada

Si tiene pasos añadidos y pulsa el botón gris con una flecha (esquina superior izquierda) o la función de escape comentada previamente, aparecerá una pantalla de aviso, figura 12, para prevenir que pierda el trabajo realizado, por un despiste.

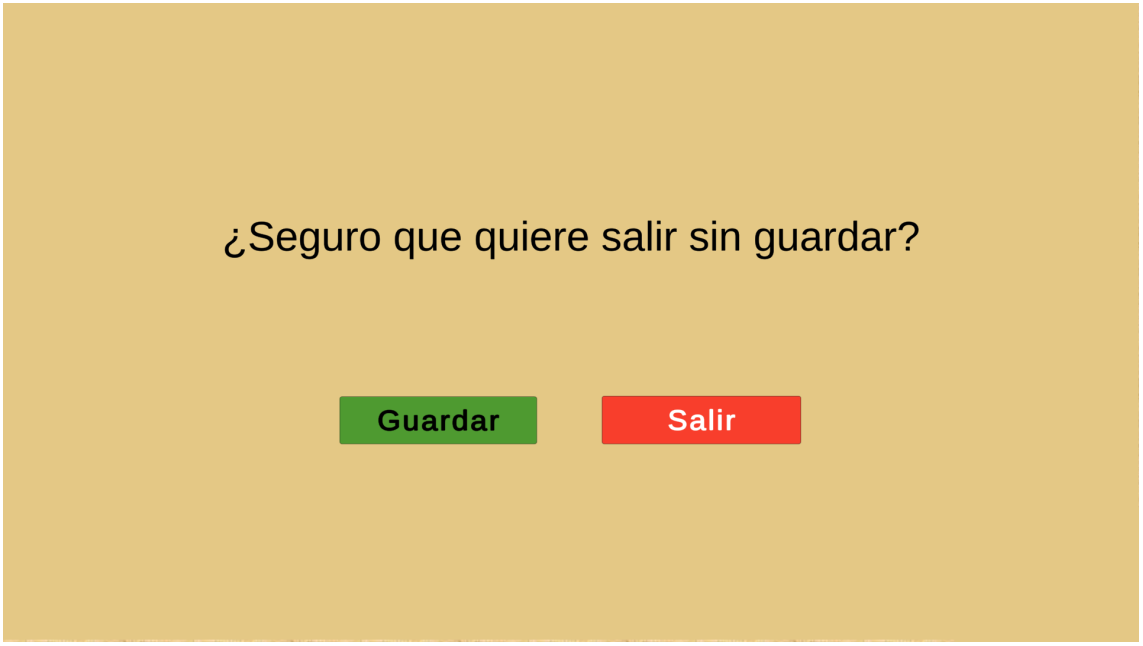

Figura 12: EntrenaBal: Salir sin guardar

B.3. Listar jugadas

Desde el menú principal, figura 6, se puede acceder a “Listar jugadas”, donde se muestra una tabla que contiene las jugadas almacenadas en el dispositivo como se puede apreciar en la figura 13. En la columna derecha aparecen

casillas seleccionables con la finalidad de borrar aquellas marcadas desde el botón rojo “Borrar selección”, estas quedarán marcadas con un tick, como es el caso de “Defensa rápida”. También se cuenta con los botones verdes “Seleccionar todas” y “Vaciar selección”, para realizar las acciones que su propio nombre indica, facilitando así el trabajo.

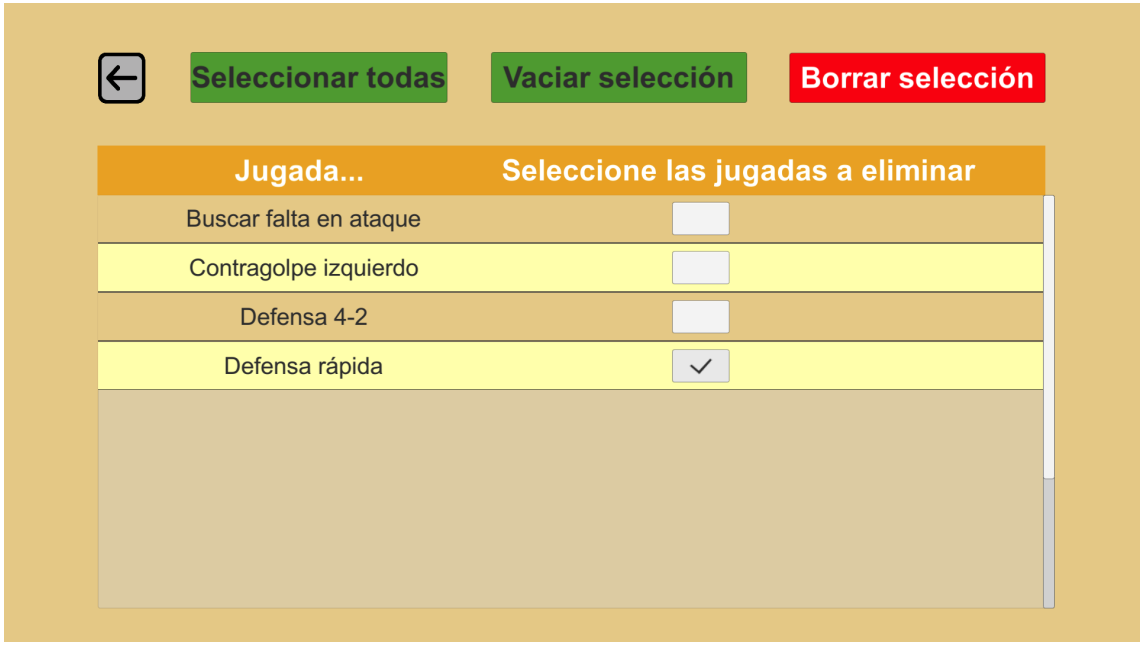

Figura 13: EntrenaBal: Listar jugadas

Con la/s jugada/s a borrar seleccionada/s, se puede pulsar el botón rojo “Borrar selección” el cual mostrará un mensaje de aviso, como el de la figura 14, para evitar borrar jugadas por error. En caso de que se hayan seleccionado todas las jugadas, se mostrarán dos mensajes de error debido al alto impacto que puede suponer.

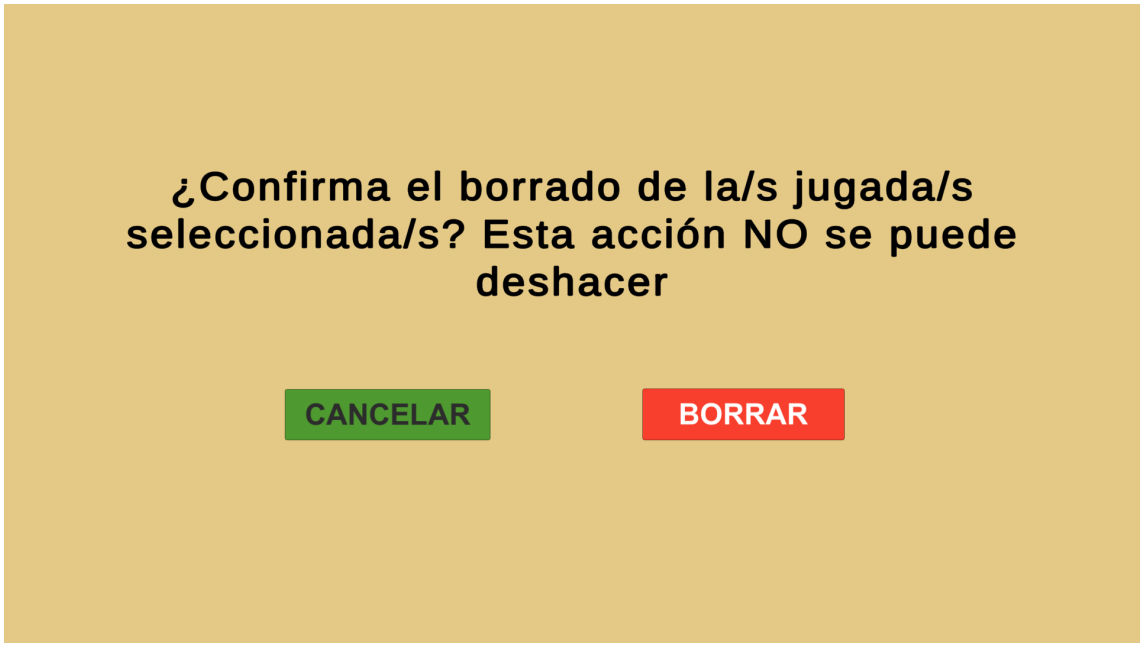

Figura 14: EntrenaBal: Borrar jugada/s

En cualquier momento puede pulsar el botón gris con una flecha (esquina superior izquierda) o la función de escape comentada previamente para regresar al menú principal.

B.4. Pizarra

Desde el menú principal, figura 6, se puede acceder, haciendo clic en "Pizarra", a una pizarra de dibujo completa. En el margen derecho de la figura 15, se pueden observar 4 rotuladores de colores: azul (por defecto), negro, rojo y verde que permiten al usuario distinguir colores para acciones, jugadores o equipos. Por encima de éstos se encuentra una goma que permite borrar las líneas trazadas. Para modificar el grosor de éstos elementos, se encuentra en la esquina inferior derecha un selector, donde cuanto más a la izquierda más fino será el trazo y más grueso cuanto más a la derecha.

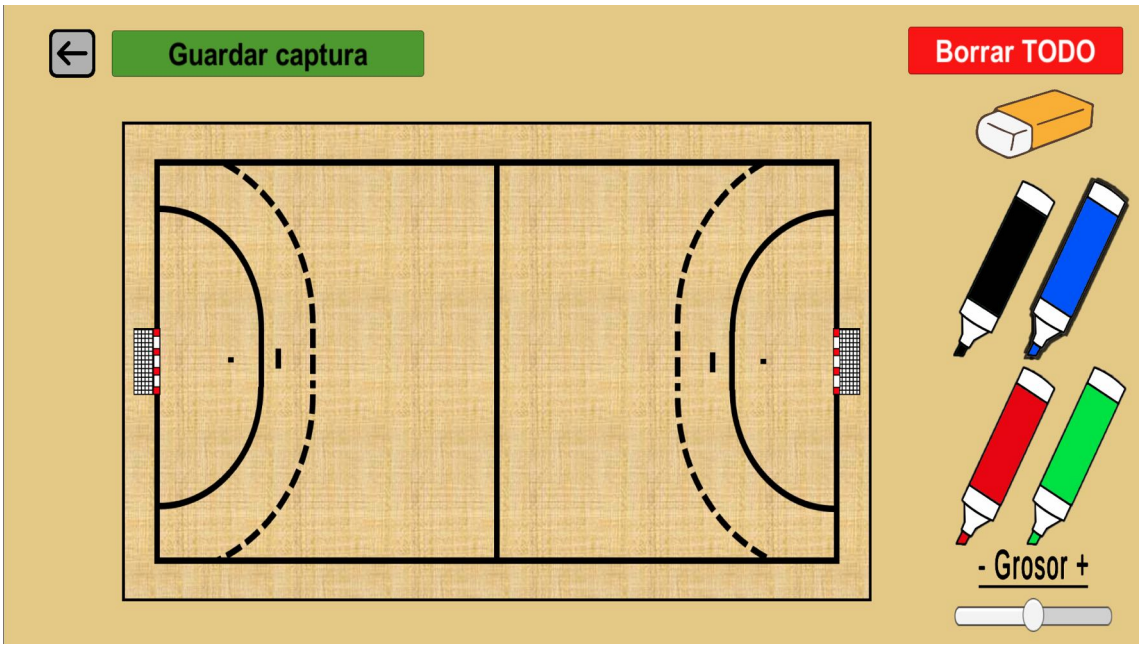

Figura 15: EntrenaBal: Pizarra interactiva

Debido a que el borrado mediante goma puede resultar tedioso, o insuficiente por la rapidez de un partido, se presenta en la esquina superior derecha el botón rojo "Borrar TODO", el cual sólo se puede pulsar si se ha dibujado algo, y que mostrará una pantalla de confirmación. Si en dicha pantalla, figura 16 se selecciona "BORRAR" se regresa a la pizarra y no aparecerá ningún dibujo.

Por otra parte, se puede pulsar el botón verde "Guardar captura" el cual salvará en el espacio interno del dispositivo una captura de pantalla de la pizarra, y muestra un mensaje informativo entre los dos botones superiores. En caso de tener algo dibujado (sin haber guardado captura) y pulsar el botón gris con una flecha (esquina superior izquierda) o la función de escape comentada previamente, se mostrará una ventana informativa para evitar perder el dibujo por error, como se puede apreciar en la figura 17.

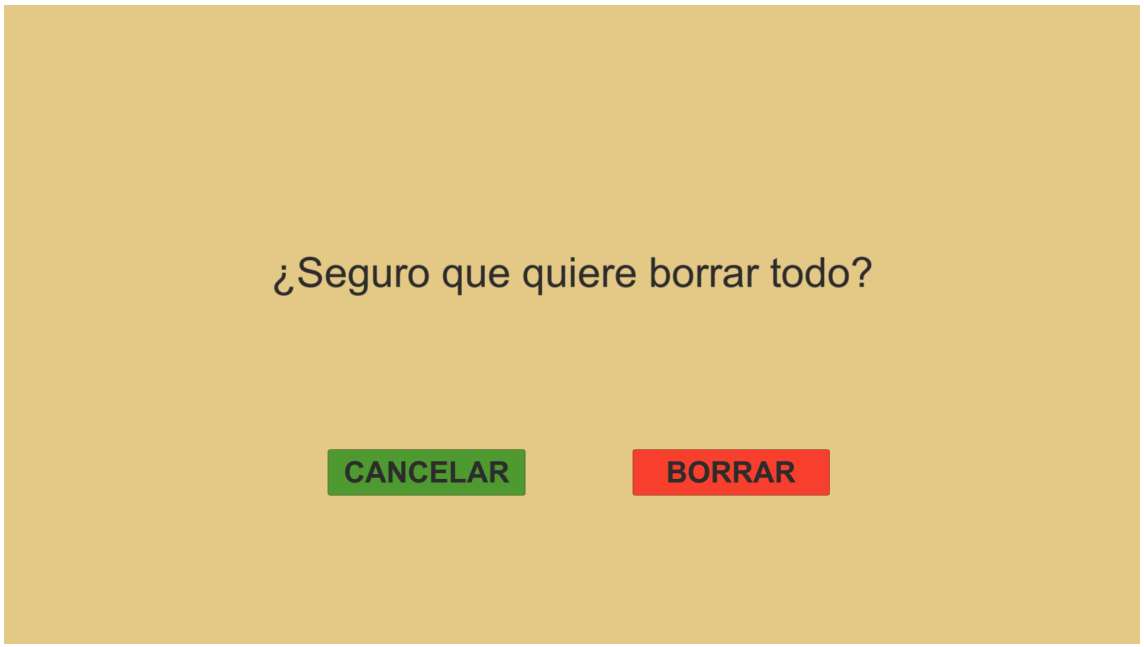

Figura 16: EntrenaBal: Borrar dibujos

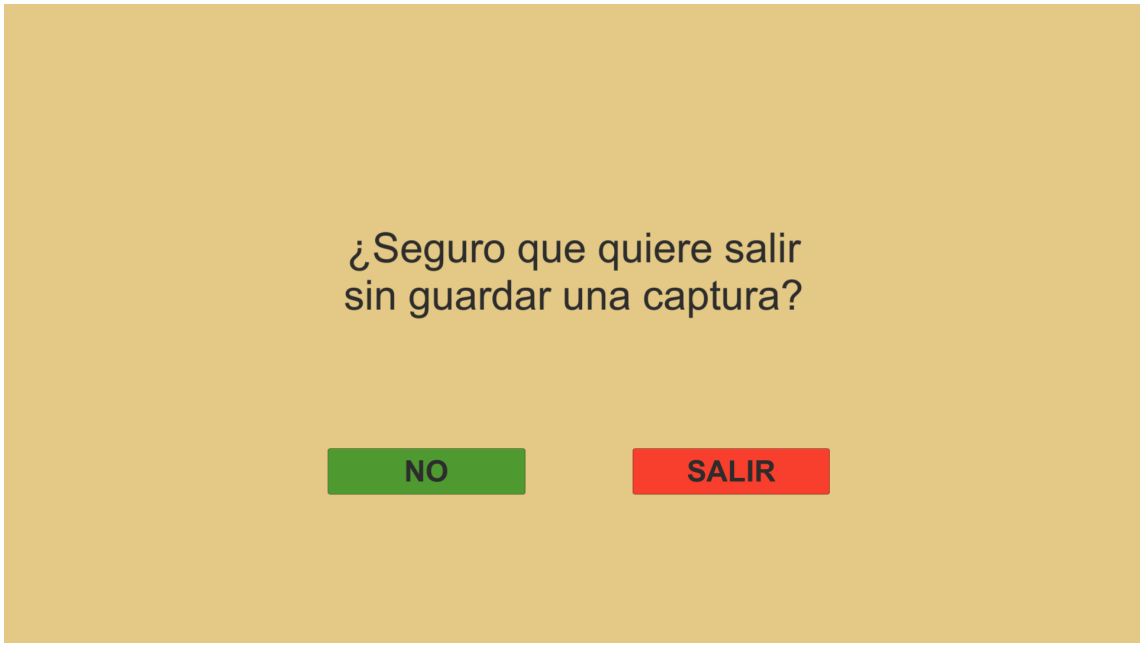

Figura 17: EntrenaBal: Salir sin guardar captura

B.5. Plantilla

Desde el menú principal, figura 6, se puede acceder, haciendo clic en "Plantilla", a una pantalla donde se muestran en forma de tabla, figura 18, los jugadores que hayan sido añadidos y estén almacenados en el dispositivo. Tiene por finalidad permitir al entrenador una mínima gestión de sus jugadores mostrando la información más básica y permitiendo añadir un comentario a cada uno, que puede ser usado, por ejemplo, para tiempos de recuperación de una lesión o para establecer cambios.

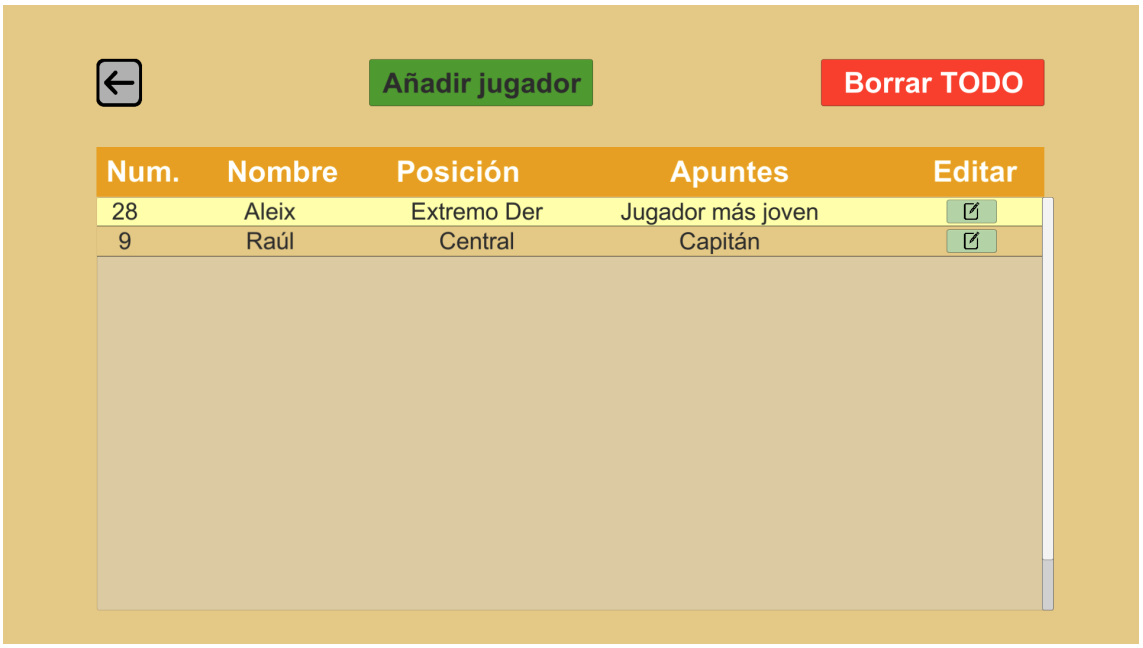

Figura 18: EntrenaBal: Plantilla de jugadores

Desde esta pantalla podemos, mediante el botón verde "Añadir jugador" incorporar un nuevo jugador a las filas del equipo. De manera similar, podremos editar los existentes pulsando en el botón verde de la fila correspondiente. En ambos casos se mostrará una pantalla como la figura 19 donde no podrá guardar hasta que cumplimente los 3 campos obligatorios. En caso de que elija un dorsal ya asignado, se mostrará un mensaje informativo entre los botones "Borrar jugador" y "Guardar" y tampoco se guardará.

Al añadir un jugador el botón de "Borrar jugador" no estará disponible ya que carece de sentido. Al editar un jugador esta función sí estará disponible, y, como se puede observar en la figura 20, lanzará una pantalla de aviso para no borrar accidentalmente un jugador.

En caso de estar editando un jugador y querer regresar al listado, mediante la flecha gris de arriba a la izquierda, se mostrará una pantalla de aviso (figura 21) para no perder los cambios de manera accidental.

Si desde la pantalla que se aprecia en la figura 18 pulsa el botón rojo "Borrar TODO" se procederá a mostrar dos pantallas de aviso como la figura 22, debido al impacto que puede suponer esta acción de cara al entrenador.

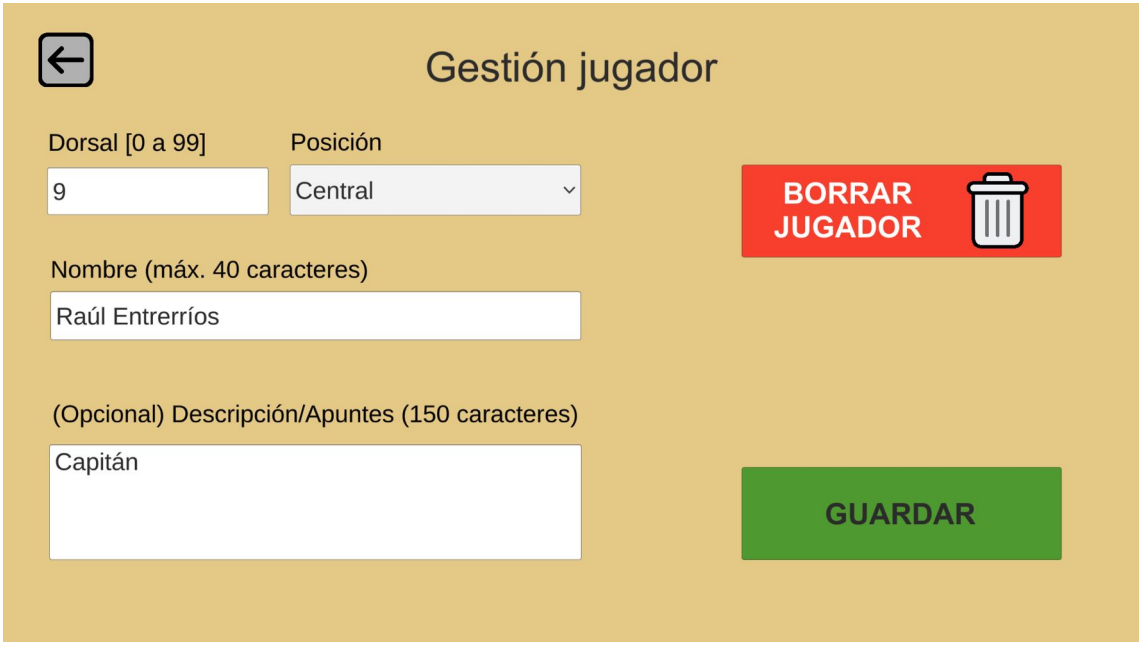

Figura 19: EntrenaBal: Añadir o editar un jugador

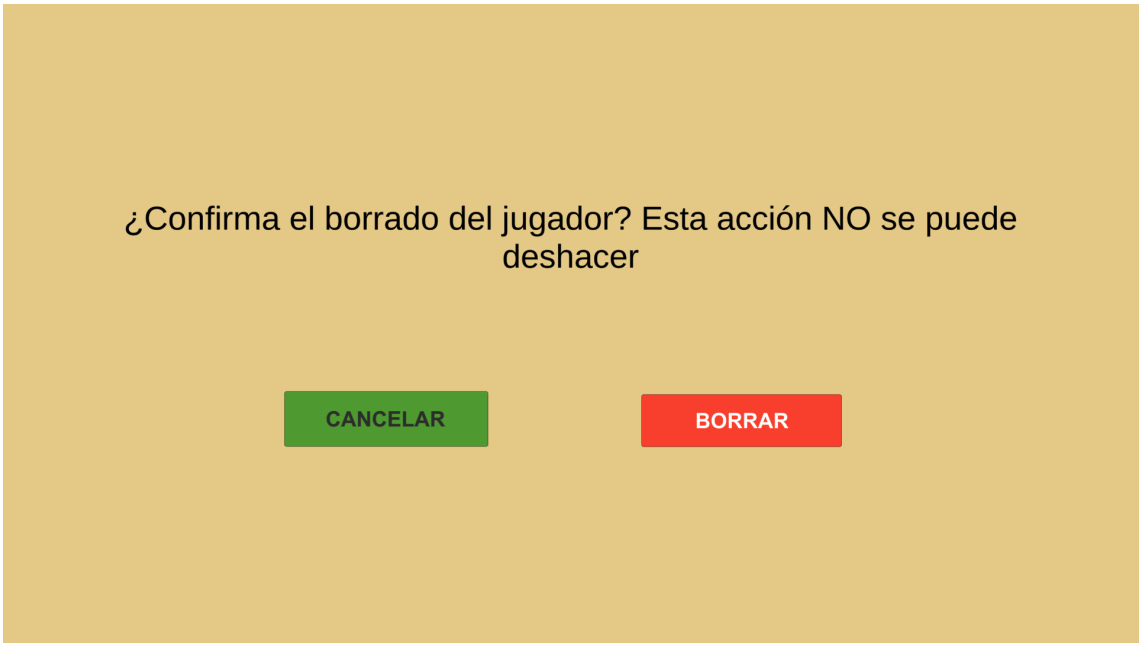

Figura 20: EntrenaBal: Borrar un jugador

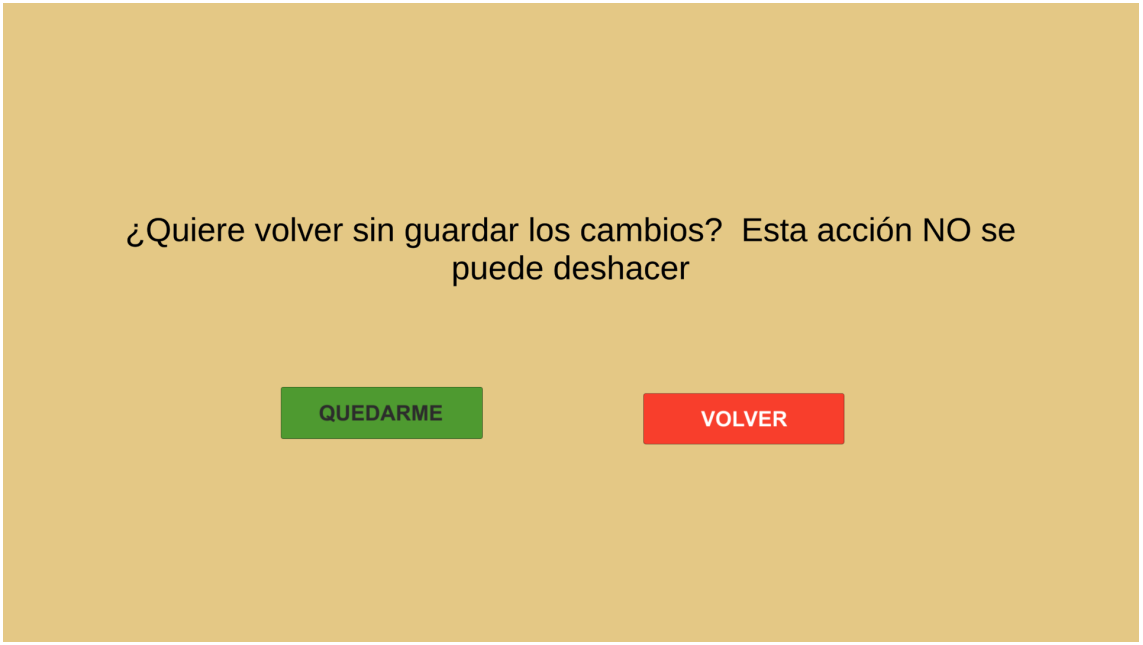

Figura 21: EntrenaBal: Mensaje de aviso al editar

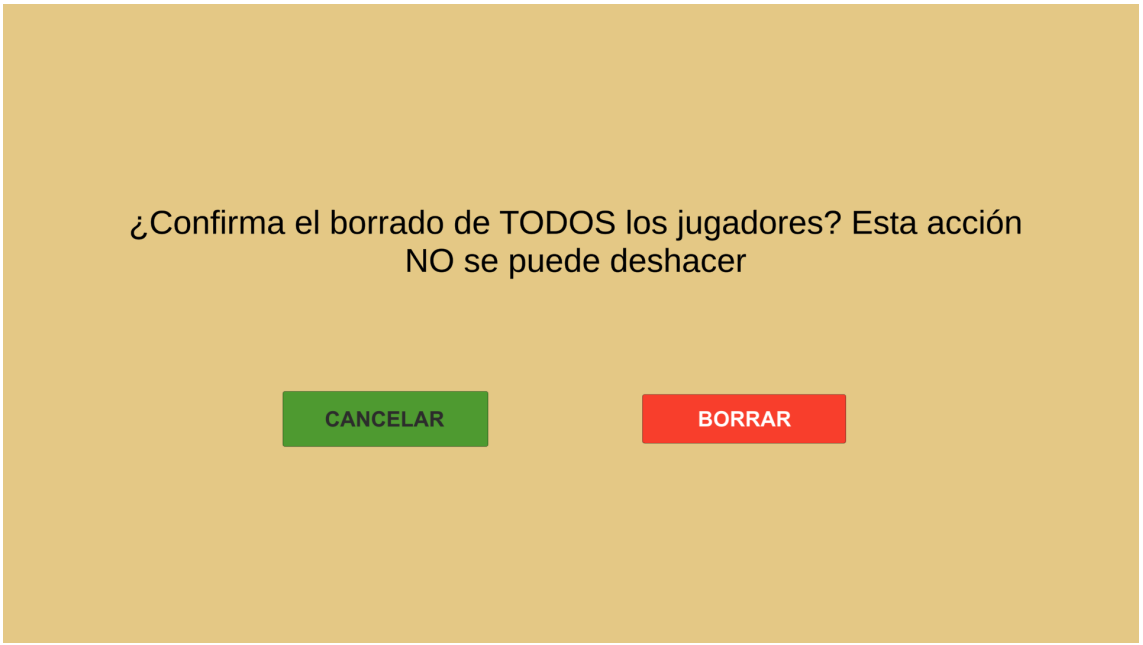

Figura 22: EntrenaBal: Borrado de todos los jugadores

B.6. Ajustes

Finalmente, si clicamos en "Ajustes" desde el menú principal, figura 6, accederemos a los ajustes de la aplicación (figura 23), donde se puede cambiar el color de la equipación del equipo propio y del adversario.

Este cambio se lleva a cabo seleccionando un color en la ruleta RGB y desplazando el cursor de luminosidad, haciendo asi más oscuro (llegando al negro) o más claro (llegando al blanco) el color final. En la "Visualización previa" se van actualizando los cambios, de tal manera que el usuario sea consciente del resultado.

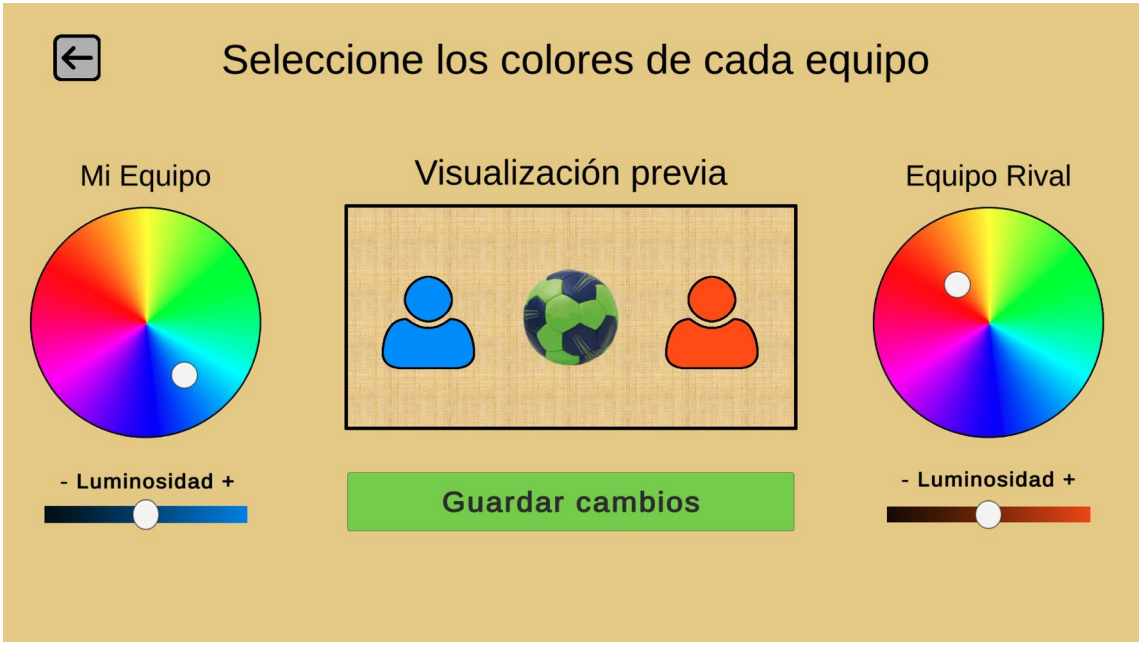

Figura 23: EntrenaBal: Ajustes

Una vez esté conforme con los colores, pulse el botón verde "Guardar cambios". Éstos serán aplicados y podrá verlos en los jugadores de cada equipo, y los botones de movimiento rápido, como se aprecia en la figura 24 en contraposición a los colores iniciales de la figura 7. Estos cambios aplican también a las jugadas previamente guardadas.

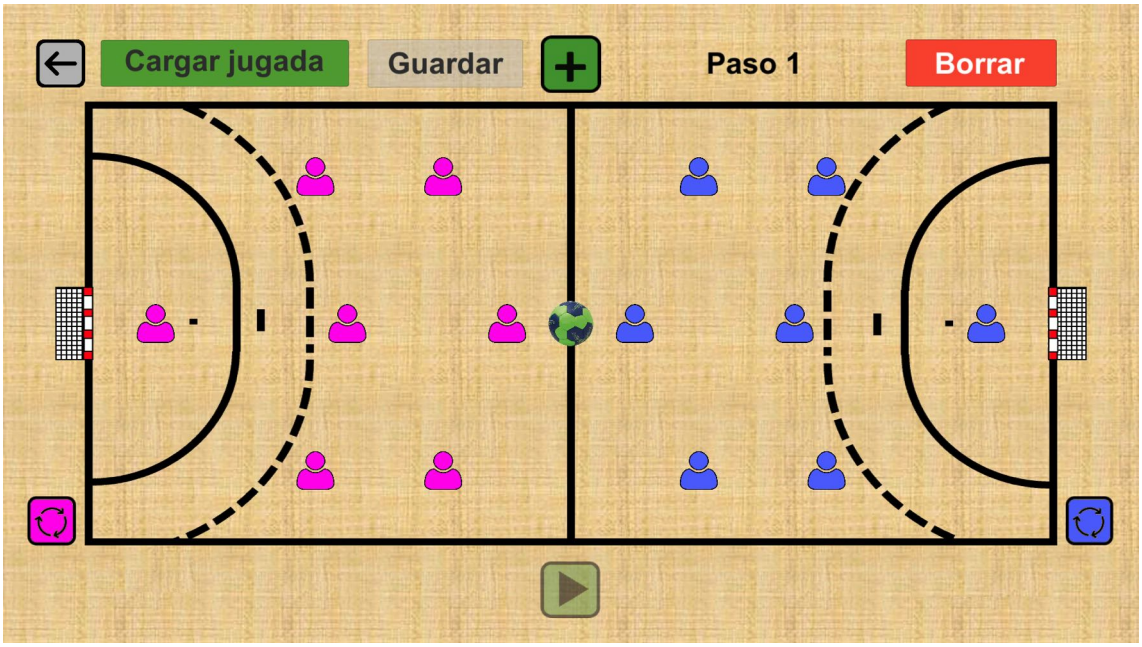

Figura 24: EntrenaBal: Cambio de color

B.7. Acceso directo a archivos

Como hemos visto, la aplicación permite crear jugadas y jugadores, así como almacenar capturas de pantalla en el dispositivo. Las imágenes pueden resultar de utilidad al usuario para pensar en futuras tácticas, o analizar por qué no funcionó la estrategia. Mientras que tanto las jugadas como los jugadores, pueden ser compartidos entre los diferentes sistemas operativos, pudiendo así trabajar en el ordenador y exportarlo a la tablet.

Debido a que Unity almacena estos datos generados por la aplicación en carpetas poco conocidas para un usuario medio, se adjuntan los accesos directos para facilitar esta tarea. Según el sistema operativo se encuentran (en su correspondiente carpeta):

- **Windows.** Son los archivos Capturas.bat, Jugadas.bat y Jugadores.bat. Haciendo doble clic le llevarán a la carpeta donde se almacena cada uno de estos elementos generados en la app. También puede acceder siguiendo las rutas:
  - **Capturas.** C:\Users\[su usuario]\AppData\LocalLow\MiguelSanchezRueda\EntrenaBal\capturas.
  - **Jugadas.** C:\Users\[su usuario]\AppData\LocalLow\MiguelSanchezRueda\EntrenaBal\jugadas.
  - **Jugadores.** C:\Users\[su usuario]\AppData\LocalLow\MiguelSanchezRueda\EntrenaBal\jugadores.
- **Mac.** Capturas, Jugadas y Jugadores son ficheros ejecutables de Unix que hacen de acceso directo. Si lo usa se ejecutará una ventana de terminal en segundo plano, no pasa nada, es el proceso del acceso directo. También puede acceder siguiendo las rutas:
  - **Capturas.** /Users/[su nombre de usuario]/Library/Application Support/MiguelSanchezRueda/EntrenaBal/capturas.
  - **Jugadas.** /Users/[su nombre de usuario]/Library/Application Support/MiguelSanchezRueda/EntrenaBal/jugadas.

- **Jugadores.** /Users/[su nombre de usuario]/Library/Application Support/MiguelSanchezRueda/EntrenaBal/jugadores.

- **Android.** En este sistema operativo no se pueden crear accesos directos genéricos como ocurría en el resto. Si quiere tener acceso a las carpetas deberá tener un gestor de archivos (muchas versiones de Android incluyen uno por defecto). Si quiere disponer de un acceso directo deberá crearlo usted mismo, pudiendo tomar como referencia esta guía. La ruta sería: /storage/emulated/0/Android/data/com.MiguelSanchezRueda.EntrenaBal/files.

Los ficheros .gd contienen los datos que forman las jugadas y jugadores, copiándolos en destino, y pegándolos en origen, podrá visualizar lo mismo entre dispositivos.