UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Desarrollo de juego de cartas francés "Belote" en aplicación móvil comercial "PlayJoy"

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Rico Alfonso, Sergio

Tutor/a: Mollá Vayá, Ramón Pascual

CURSO ACADÉMICO: 2021/2022

Resumen

La industria del videojuego supone la industria de entretenimiento más lucrativa actualmente, pasando a industrias como el cine o la música. Este sector está distribuido en diferentes plataformas, siendo los teléfonos móviles la mayor, tanto en número de jugadores como en dinero generado. Es por este motivo que muchas empresas han centrado su actividad a este ámbito y plataforma, como es el caso de Mundijuegos, Plato o NoSpoon con su proyecto *Playjoy*, empresa que aloja el desarrollo de este proyecto. Todos estos proyectos constan de las llamadas aplicaciones sociales, uniendo los juegos con cualidades sociales como listas de amigos y chats privados y públicos. Con el objetivo de aumentar su base de jugadores y explorar un mercado francés no explotado hasta el momento, en NoSpoon se busca desarrollar la belote, juego tradicional francés.

Para el desarrollo de este juego, se buscó generalizar las fases de este proceso para ofrecer una visión más transparente del progreso a todo el equipo y mejorar el proceso de creación de campañas de marketing. Se plantearon entonces estas nuevas fases con objetivos marcados para representar mejor el progreso del juego.

La aplicación cuenta ya con siete juegos previos a la incorporación de la belote. Estos juegos marcan las bases del diseño gráfico del nuevo juego junto con sus funcionalidades. La distribución vertical de los gráficos para un uso de la aplicación más natural y la reconexión fluida en cualquier momento de la partida, son los dos rasgos obligatorios traídos de los otros juegos. Todos estos también cuentan con una inteligencia artificial competente para estos, pero en el caso de la belote, se buscó dar un paso adelante en este aspecto y se buscó diseñar e implementar una nueva IA mejorando a las anteriores. Esta busca ser más sencilla de entender al igual que de mejorar en un futuro, por lo que se diseñó una IA estática que utiliza funciones heurísticas para valorar la situación actual y tomar la mejor decisión posible.

Por lo tanto, este proyecto desarrollado dentro de la empresa NoSpoon y alojado en su aplicación *Playjoy* tiene como objetivos mejorar la organización de los desarrollos de futuros juegos, diseñar e implementar una nueva inteligencia artificial que cumpla con los requisitos de competencia necesarios para la belote, y un objetivo final de introducir la aplicación al mercado francés, sector todavía no explorado por esta.

Palabras clave: Kotlin, juego de cartas, Belote, metodologías ágiles, roadmaps, sprints.

Abstract

The video game industry is currently the most lucrative entertainment industry, surpassing industries such as film and music. This sector is distributed in different platforms, being cell phones the largest, both in number of players and money generated. It is for this reason that many companies have focused their activity to this area and platform, as is the case of Mundijuegos, Plato or NoSpoon with its project Playjoy, company that hosts the development of this project. All these initiatives consist of so-called social applications, combining games with social functionalities such as lists of friends and private and public chats. In order to increase its player base and explore a hitherto untapped French market, NoSpoon is looking to develop belote, a traditional French game.

For the development of this game, we sought to generalize the phases of this process to offer a more transparent view of the progress to the whole team and improve the process of creating marketing campaigns. These new phases were then proposed with marked objectives to better represent the progress of the game.

The application now has seven games prior to the addition of the belote. These games set the basis for the graphic design of the new game along with its functionalities. The vertical distribution of the graphics for a more natural use of the application and the fluid reconnection at any time of the game, are the two mandatory features brought from the other games. All of them also have a competent artificial intelligence enough for their case, but in the case of belote, it was sought to take a step forward in this aspect and design and implement a new AI improving the previous ones. This seeks to be easier to understand as well as to improve in the future, so we designed a static AI that uses heuristic functions to assess the current situation and make the best possible decision.

Therefore, this project developed within the company NoSpoon and hosted in its Playjoy application aims to improve the organization of future game developments, design and implement a new artificial intelligence that meets the competence requirements needed for the belote, and a final goal of introducing the application to the French market, a sector not yet explored by it.

Keywords: Kotlin, card games, Belote, agile methodologies, roadmaps, sprints.

Dedicatoria

Este trabajo está dedicado a mis padres y familia, por haberme apoyado incondicionalmente y siempre luchar por mi felicidad.

En especial está dedicado a mi madre, que espero que esté feliz y orgullosa allí donde se encuentre. Gracias por confiar siempre en mí y nunca dudar, espero que puedas descansar ahora. Te quiero.

Agradecimientos

Primero de todo agradecer a Ramón por tutorizarme durante la escritura de este documento, ofreciéndome el mejor consejo posible y animándome a continuar con la escritura.

Gracias a Patricio Letelier por la ayuda para comenzar las prácticas de empresa que han desembocado en este proyecto y mi actual lugar de trabajo regular.

Gracias a NoSpoon, Daniel González, Daniel Blázquez, Adrián Guevara y el resto del equipo por compartir sus conocimientos durante estos meses de desarrollo. También por darme esta increíble oportunidad de trabajar en un proyecto tan interesante.

También agradecer a mis padres por haberme dado la increíble oportunidad de cursar este grado, apoyarme durante su curso y nunca haber dejado de confiar en mi y mis capacidades. Agradezco a mi pareja, quien me ha ayudado a centrar mis energías a la escritura de esta memoria y poder llevarla a cabo.

Tabla de contenidos

Índice de figuras

Índice de tablas

1. Introducción

1.1 Motivación

Desde pequeño, siempre me apasionaron los videojuegos, de cualquier tipo y en cualquier medio, fue por este motivo y por mi gusto por la tecnología en general que decidí cursar la carrera de Ingeniería Informática. A medida que avanzaba por los cursos del grado, fui descubriendo mi gusto al desarrollo software en cualquier forma, por lo que decidí elegir la rama de Ingeniería del Software como especialización. Después, ya en el último año contacté con uno de mis profesores para buscar prácticas en alguna empresa local, a poder ser de desarrollo de videojuegos. Entonces fue cuando comencé a trabajar con NoSpoon, en su proyecto *Playjoy*, una aplicación recopilatoria de juegos tradicionales para móvil.

Trabajando con esta empresa afiancé mi gusto por el desarrollo software y desarrollo de videojuegos. Ya que el trabajo que realizaba dentro de la empresa entraba dentro de las características para ser proyecto de fin de grado, lo elegí como tal. Además, durante el desarrollo de dicho juego, la belote, adquirí un gran número de conocimientos sobre desarrollo ágil y como trabajar en un entorno profesional de trabajo. Otro aspecto que recalcar del desarrollo del juego fue la organización e implementación de la inteligencia artificial, para la cual utilicé los conocimientos adquiridos en el programa de intercambio en el extranjero que cursé el año anterior.

1.2 Análisis del mercado de los videojuegos

Actualmente no es sorpresa ver que la industria del videojuego sobrepasa con gran diferencia a otras industrias del mundo del entretenimiento como el cine y la música (1). Lo que no es sabido a gran escala es cuán grande esta diferencia es en realidad. Cuando se habla de ingresos y personas involucradas en los desarrollos, los videojuegos no han dejado de crecer durante los últimos años.

A nivel global, la industria en conjunto ha generado durante 2021 un total de 175,800 millones de dólares, 16,500 millones más que 2020 (2) (3). De este total, un 45% (79,000 millones de dólares) (2) fueron generados exclusivamente por videojuegos para smartphones, resultando en un aumento del 4.7% respecto a 2020 (3), el cual a su vez supuso un notable aumento del 15.8% respecto a 2019 (4). A nivel nacional, la industria generó un total de 1,747 millones de euros en 2020 (5) (18% más que 2019 (1)). Ya en 2019, donde generó un total de 1,479 millones de euros, la industria del videojuego era líder en el mundo del entretenimiento español. Superó a la industria del cine y la música conjuntamente, con 558.5 millones de diferencia (1).

También cabe recalcar el aumento en el número de jugadores globales. Durante 2020 y 2021, ha habido un aumento constante del 5.3% resultando en la increíble cifra de 3,000 millones de jugadores en 2021 (2).

A modo de nota, estos incrementos vienen marcados por varios factores, como la mayor facilidad de acceso al mundo online o la mayor asequibilidad de smartphones potentes; pero el factor más influyente ha sido la situación especial que el mundo ha estado sufriendo estos dos últimos años debido al COVID.

1.3 Análisis de mercado de videojuegos tradicionales en línea

El gran aumento de jugadores durante estos últimos ha introducido a nuevos públicos, los cuales no estaban tan familiarizados con los videojuegos. Este público es principalmente de naturaleza casual, por lo que se ven inclinados a consumir juegos rápidos, sencillos y de fácil acceso, generando el aumento antes descrito de la industria del videojuego móvil.

Es por esta familia de jugadores casuales que muchas compañías han optado por revitalizar juegos de mesa lanzando versiones digitales, tanto móviles como versiones para consolas u ordenadores. Es el caso de juegos como el *Monopoly Plus*[1] y *UNO*[2], desarrollados por Ubisoft u otros juegos más tradicionales como el parchís o el tute, adaptados por Tangelo Games[3] en su página web Mundijuegos.com[4]. Estas hacen uso del juego en línea para eliminar uno de los mayores obstáculos que estos juegos presentan en su forma natural, la necesidad de estar todos los jugadores presentes. Esta adición fue especialmente llamativa durante la pandemia, permitiendo a esta familia de jugadores a disfrutar con amigos juegos con los que ya están familiarizados.

Estos juegos abarcan desde el famoso UNO o *Monopoly*[5], hasta juegos de cartas tradicionales como el chinchón[6] o la brisca[7]. Aunque juegos como los mencionados UNO y *Monopoly* requieren de sus respectivas licencias, los más tradicionales no las requieren en su gran mayoría. Así, las puertas quedan abiertas para PYMES y *startups* que deseen entrar en el mercado desarrollando juegos tradicionales sin requerir de costosas licencias Este es el caso de empresas como Plato Team Inc. con su aplicación *Plato*[8] o Tangelo Games con su web y aplicación móvil *Mundijuegos*[9]. Además, viendo el gran abanico de posibilidades que se abre al tener disponible los juegos tradicionales, no es sorpresa que muchos de los proyectos emprendidos busquen recopilar un gran número de juegos en una sola aplicación. De esta forma los clientes encontrarán un núcleo de juegos

[1] https://www.ubisoft.com/es-es/game/monopoly/monopoly
[2] https://www.ubisoft.com/en-gb/game/uno/uno
[3] https://tangelogames.com/
[4] https://www.mundigames.com/
[5] https://monopoly.hasbro.com/es-essbro
[6] https://www.ludoteka.com/clasika/chinchon.html
[7] https://www.ludoteka.com/clasika/brisca.html
[8] https://www.platoapp.com/
[9] https://www.mundigames.com/

conocidos, pudiendo saltar entre ellos. Esto ayuda mucho a mantener a los jugadores interesados y poder incluir a diferentes grupos de jugadores con una misma aplicación.

1.4 NoSpoon y su proyecto Playjoy

Entre las empresas que cuentan con proyectos de dichas características se encuentra NoSpoon[10] y su aplicación móvil *Playjoy*[11]. Esta empresa nacida en 2016 y centrada en el desarrollo de videojuegos cuenta con diversos proyectos dentro del mundo, a la vez que en otros ámbitos. Gracias a proyectos pasados como la página web *Mundijuegos.com*, la empresa contaba con un elevado conocimiento y muchos materiales para el desarrollo de su aplicación *Playjoy*. Esta se trata de uno de estos "núcleos" de juegos tradicionales digitalizados en línea, donde se pueden encontrar juegos como la brisca, el dominó, el parchís o el chinchón entre otros. Para más información sobre la empresa, refiérase al anexo 1.

Esta aplicación no solo se centra en ser una recopilación de juegos tradicionales, sino que también busca ser un medio social donde amigos puedan interactuar chateando o jugando entre ellos y puedan conocer a nuevos en partidas online o chateando en grupos comunes. Con estos dos pilares, la aplicación consigue ofrecer un ecosistema social y de entretenimiento donde los jugadores se sientan cómodos, hagan amistades y jueguen a los juegos que ya conocen.

Además, la aplicación cuenta con más funcionalidades enfocadas a la retención de sus usuarios, como son un sistema de apuestas obligatorias para jugar las partidas, la capacidad de subir niveles consiguiendo así recompensas y unas tablas de posicionamiento mensuales, donde se publican los jugadores más victoriosos de los juegos. En el momento del desarrollo de la belote, la aplicación cuenta con un total de ocho juegos: dominó, chúpate dos, parchís, chinchón, brisca, tute, durak y bingo, cada uno con sus tablas de puntos donde los jugadores pueden competir por ser el más victorioso.

En el sistema de apuestas antes mencionado reside la fuente de ingresos de la aplicación, ya que son necesarias para jugar, y durante el cálculo del bote total, cierto porcentaje es eliminado. De esta forma, tanto ganando como perdiendo, ese porcentaje eliminado deja de estar en manos de los jugadores, creando así un déficit de monedas y haciendo que eventualmente los usuarios se queden sin monedas y necesiten conseguir más de alguna de las dos maneras principales. Como primera opción, todos los jugadores reciben diariamente una cantidad fija de monedas; como segunda opción, los jugadores tienen disponibles ciertos paquetes de monedas en la tienda de la aplicación, pudiendo ser comprados por dinero real. En el anexo 2 se explora con más detalle la aplicación y sus sistemas.

[10] http://nospoonlab.com/
[11] https://playjoy.com/en/

1.5 Objetivos de la empresa a partir de la inclusión del belote

En cuanto a sus jugadores, la mayoría de estos provienen de países hispanohablantes, seguidos de los ingleses, portugueses y rusos. Con el objetivo de ampliar y diversificar su población, la empresa busca desarrollar el juego de cartas francés tradicional llamado belote. Este no solo abrirá las puertas a un público francés, también llamará la atención de otros muchos países europeos como Grecia, Bulgaria, Croacia, e incluso países más lejanos como Arabia Saudí. Puesto que el principal público objetivo es francés, tanto el juego como la aplicación tendrán que ser adaptados a este idioma.

Además, la empresa busca con este juego conseguir generalizar el ciclo de desarrollo de los juegos, permitiendo ofrecer mejores estimaciones y una metodología más ágil en un futuro. Debido a tratarse de una empresa emergente y su naturaleza ágil y cercana, este problema nunca se había tomado a consideración, pero los objetivos de expansión y crecimiento requieren de una organización bien establecida.

Por último, también se busca mejorar la calidad de las inteligencias artificiales y sus capacidades con la inclusión de este nuevo juego. Los *bots* actuales cuentan con una guía de actuación preestablecida, lo cual dificulta sus posibles modificaciones para responder a nuevas estrategias de juego o incrementos en nivel de complejidad.

1.6 Objetivos

Los objetivos de este trabajo girarán en torno al desarrollo de un nuevo juego dentro de la aplicación *Playjoy* y la organización del proceso. Dichos objetivos son:

1. Desarrollar juego de cartas tradicional belote dentro de la aplicación móvil *Playjoy*.
2. Incrementar el número de jugadores en el mercado francés, expandiendo los jugadores y potenciales inversores del proyecto.
3. Estudiar y generalizar el proceso de desarrollo de los videojuegos dentro del proyecto, determinando las fases que dividen a este. De esta forma la empresa podrá adquirir unas mejores estimaciones con futuros juegos.
4. Desarrollar una inteligencia artificial competente y adaptable que controle los *bots* de las partidas.

1.7 Impacto esperado

Mediante el desarrollo de este nuevo juego, se espera entrar en nuevos países, aumentando y diversificando la cantidad de jugadores que la disfruten. Además, se utilizará como punto de venta para buscar nuevos contratos con empresas de dichos países, pudiendo aplicar a nuevas promociones y posibles ventajas para los jugadores.

En cuanto al proceso de desarrollar juegos, se busca poder generalizarlo mediante el estudio del juego tratado en este trabajo. De esta manera el desarrollo de futuros juegos

se vería mejorado, siendo este más ágil y fácil de organizar, ya que este proceso contaría con fases bien marcadas, cuyos objetivos son conocidos y permiten una previsión de fechas más acertada.

En los juegos, cuando un jugador abandona una partida, ya sea de forma definitiva o momentáneamente es "poseído" por un *bot*, el cual será manejado por una inteligencia artificial. Estos actualmente están programados mediante órdenes según el estado de la partida, puramente reactivos y difíciles de modificar ya que requieren de cambios a la estructura del código para jugar con diferentes estrategias.

Por ese motivo, con el belote se busca mejorar este aspecto utilizando heurísticas que evalúen cada una de las posibilidades y escoger la más positiva. De esta manera solo hará falta modificar dicha heurística para mejorar o solo cambiar la forma de actuar de dicho *bot*. Al tratarse de un juego por equipos, esta ha de poseer unos conocimientos tácticos mínimos. Con estos, podrá aportar a la partida de forma positiva para su equipo, a la vez que suponer un reto para los oponentes.

1.8 Estructura

En este apartado se comentará la estructura de la memoria realizada, organizada en un total de siete capítulos. Se recorrerán brevemente los contenidos de cada uno de estos.

El primer capítulo se compone de la introducción, como ya se ha visto. Se comenzará con las motivaciones personales y económicas detrás de la decisión del desarrollo de la belote. Se presentará la empresa que alojará el desarrollo del proyecto al igual que la aplicación donde se ofrecerá acceso al mismo. Se presentarán los objetivos a cumplir con el desarrollo del proyecto, centrándose en los ítems a conseguir por la empresa.

En el segundo capítulo se explorará el estado actual de los videojuegos móviles, recorriendo brevemente la historia de estos. Se irá focalizando la redacción hasta llegar a las aplicaciones móviles de juegos sociales, donde reside la aplicación *Playjoy*, que ofrecerá el juego desarrollado. Por este motivo se expondrá el estado actual de la aplicación, indicando número de jugadores actuales y su principal procedencia.

Continuando con la memoria, en el tercer capítulo se planteará una propuesta al juego a desarrollar. Comenzando por la exposición de la competencia y su estudio mediante un diagrama DAFO, se generarán los requisitos a cumplir por la implementación a desarrollar. Por último, se recorrerán las nuevas fases comunes y su coste temporal esperado.

Ya con la propuesta bien definida, en el cuarto capitulo se centrará en el diseño de la implementación comenzando por el estudio de las tecnologías disponibles, su proceso de selección y una breve explicación de sus características. Finalizando este capítulo se encontrará la explicación de la estructura de las diferentes partes de la aplicación y la comunicación entre ellas.

En el quinto capítulo se comentará el desarrollo del juego, pasando por las mayores dificultades encontradas, su impacto en el progreso de las fases propuestas y explicando el proceso de las pruebas realizadas durante el trabajo. Este impacto se estudiará exponiendo las fechas de finalización y comienzo reales de cada fase y, por lo tanto, los cambios a las fechas planificadas originalmente para el proyecto entero.

El sexto capítulo contendrá una explicación completa de la inteligencia artificial realizada para la belote. Este capítulo contendrá una estructura tradicional para la solución de un problema, comenzando por un estudio completo del estado actual de las inteligencias artificiales en el resto de los juegos, continuando por una propuesta de la solución, un diseño, desarrollo y las pruebas realizadas a esta.

Como penúltimo capítulo se encontrará la conclusión del proyecto, comenzando con el repaso la aplicación de los conocimientos adquiridos durante la carrera al proyecto. Continúa el capítulo con el impacto de la explotación de la belote durante los últimos más de seis meses, mostrando su impacto en los jugadores de la aplicación. Se valorarán las fases propuestas para el desarrollo al igual que la IA desarrollada. Finalizando con el trabajo futuro a realizar dentro de la empresa.

Cerrando la memoria se encuentra el octavo capítulo conteniendo las fuentes bibliográficas utilizadas en las referencias que aparecen a lo largo de la redacción.

Como adición a la memoria se encuentran cuatro anexos adicionales, ofreciendo más información de la empresa y la aplicación donde reside el juego desarrollado. Se ofrecerán las reglas del juego desarrollado con el objetivo de facilitar el entendimiento de las partes de la memoria referentes a las partidas del juego. El último anexo muestra y explora los gráficos comunes en todos los juegos de la aplicación.

1.9 Convenciones

En este documento, los títulos o nombres de proyectos y aplicaciones serán escritos en cursiva, contengan o no palabras extranjeras. A su vez, se marcarán entre comillas los nombres de las clases dentro del código mencionadas en el texto.

Finalmente, las referencias a las fuentes consultadas se realizarán siguiendo la norma ISO 690.

2. Estado del arte de juegos móviles

2.1 Videojuegos móviles

Desde su nacimiento, los videojuegos han estado ligados a máquinas de entretenimiento no móviles, como son ordenadores personales o consolas. En 1962, Steve Rusell junto al MIT crearon el llamado *Spacewar!* (Figura 1), este no llegó a comercializarse puesto que solo podía jugarse en unidades centrales de computación, no disponibles para la gran mayoría de usuarios de la época (6). Ya en los 70, esta realidad cambió y la primera videoconsola disponible para los consumidores vio la luz, la *Odyssey* de Magnavox. La primera versión de esta fue lanzada solo con el juego *Pong*, más tarde fue actualizada para aceptar ROM programables en forma de cartuchos (7).

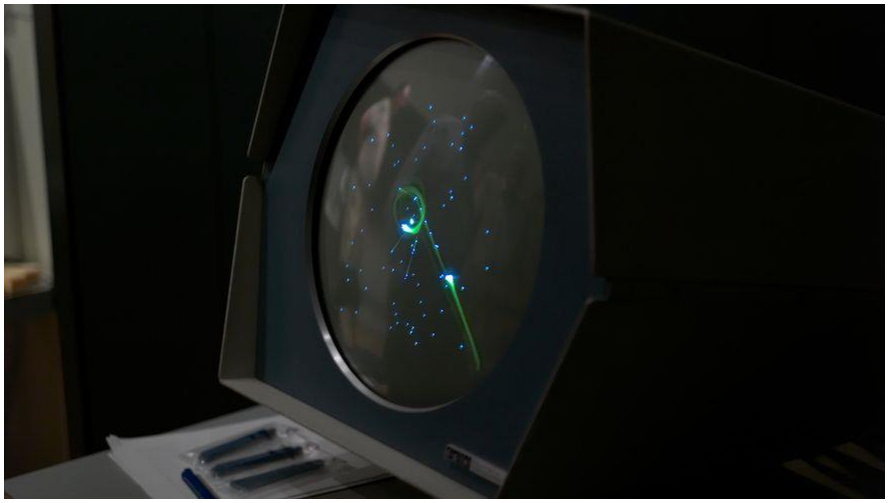

Figura 1. Imagen de Spacewar!

A mediados de los 70 y con la llegada de los microprocesadores, la industria creció sin límite, dando resultado a la primera época dorada de los videojuegos, ofreciendo nuevas consolas a los consumidores e incluso nuevas formas de crear juegos propios gracias a la llegada de los ordenadores personales. Este nuevo mundo de creación fue el que originó la caída de esta época dorada en 1983, ya que el mercado desbordó de juegos de poca calidad y una consecuente pérdida del interés popular (8).

Figura 2. Fotografía de la Nintendo Entertainment System

Fue en este momento cuando la empresa nipona Nintendo salvó la industria introduciendo la *Nintendo Entertainment System* (Figura 2)Figura 2. Fotografía de la Nintendo Entertainment System al mundo occidental, con su gran catálogo de juegos propios de alta calidad (8). Con este nuevo resurgir de la industria y el continuo aumento en la potencia computacional viable para ser vendida, durante los 90 llegaron a nuestras manos las primeras videoconsolas portátiles, destacando la *Nintendo GameBoy* y la *Sega Game Gear* (9). Estas consolas fueron evolucionando, llegando a su última generación, donde se encuentran consolas como la *PSVita* de Sony o la *Nintendo 3Ds* (Figura 3).

Figura 3. Fotografía de una Nintendo 3DS

A pesar de su inicial éxito y novedad, las consolas portátiles han visto su declive debido al aumento en la accesibilidad y potencia en los actuales smartphones. Llegando a ser comparables o incluso a superar a las consolas en potencia disponible (10). Cabe también destacar el nuevo mercado de consolas híbridas, pudiendo ser usadas tanto de forma portátil como conectadas a un monitor de forma estática. Se encuentran productos como la *Nintendo Switch* o el *Steam Deck* (Figura 4). Es por este motivo que la industria del videojuego móvil es ahora la más grande tanto en número de jugadores (2,200 millones) como en dinero generado en todo el mercado.

Figura 4. Fotografía de un Steam Deck de Valve

2.1.1 Situación actual de los videojuegos móviles

Durante la época donde videoconsolas y portátiles convivían, se podía observar una alta diferencia en la calidad gráfica de los videojuegos, está siendo aún más notable al comparar con los dispositivos móviles. Esta, junto a la tendencia de hacer juegos cada vez más complejos y llamativos gráficamente convirtió a las videoconsolas en las mejores opciones, pero últimamente esta tendencia ha cambiado. Gracias a la creciente facilidad a la hora de desarrollar videojuegos, se han ido explorando nuevas alternativas a un apartado gráfico complejo y difícil de conseguir por dispositivos móviles y consolas portátiles.

Esta nueva tendencia y el incremento de la potencia encontrada en los *smartphones* ha permitido a estos dispositivos a conquistar el mercado de los videojuegos. Los videojuegos móviles son actualmente los reyes de la industria, llegando a manejar un 52% (90,600 millones de dólares) (2) del total generado por la industria de los videojuegos. Y no es solo en el apartado monetario donde reinan, también es la plataforma con más número de jugadores, llegando a ser un total de 2,200 millones mundialmente (11).

2.2 Adaptaciones digitales de juegos tradicionales

La digitalización de juegos tradicionales de mesa siempre ha estado presente en el mundo de los videojuegos. Anteriormente, estas adaptaciones estaban presentes tanto en aplicaciones móviles como en aplicaciones web. Las versiones web además ofrecían una experiencia online permitiendo la interacción entre los jugadores, mientras que este aspecto era inexistente en los móviles del momento. Este aspecto online y social que añadía el acceso a internet fue el detonante para abrir un nuevo mercado en auge como eran las adaptaciones digitales de juegos tradicionales con capacidad de socializar con gente de cualquier parte del globo.

Un ejemplo de estas webs es la página *mundijuegos.com* (Figura 5), la cual se centra en la recopilación de juegos sociales, incluyendo un sistema de amigos, apuestas, cosméticos, eventos y rankings competitivos. Posee un total de más de 60 juegos, pasando por el bingo, póker, chinchón, parchís, etc., donde los jugadores pueden crear partidas propias con amigos, conocer a nueva gente y no tanto jugar como socializar. De este modo nace el concepto de los juegos sociales.

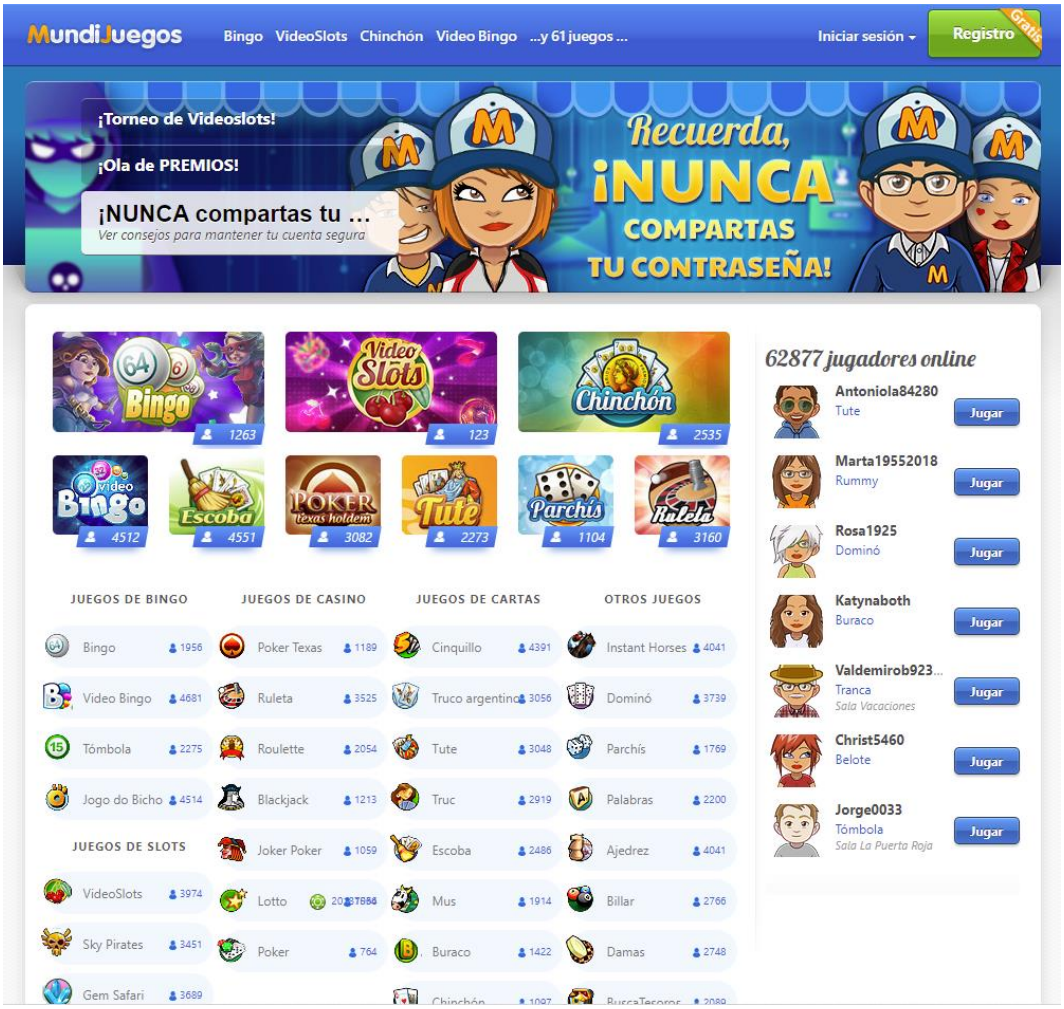

Figura 5. Página de aterrizaje de mundijuegos.com

Estas aplicaciones web fueron perdiendo parte de su público a medida que sus jugadores obtenían smartphones con los cuales eran capaces de jugar a los mismos juegos de manera más dinámica y en cualquier sitio. Debido a esta mayor facilidad de acceso y muchos de sus consumidores siendo jugadores casuales, comenzó una migración a aplicaciones del mismo estilo disponibles en sus dispositivos móviles.

2.2.1 Aplicaciones móviles de juegos sociales

Dentro de aplicaciones móviles de juegos sociales se encuentra una gran diversidad. Debido al elevado número de empresas que apuestan por este tipo de software, el mercado cuenta con aplicaciones más complejas y completas. Estas han de llegar al máximo número de jugadores posibles, manteniéndoles participativos dentro del

sistema aumentando así su público fiel. Existen diferentes factores que ayudan a obtener buenos resultados en este aspecto, destacando las recopilaciones de diversos juegos o el nivel de interacción social que las aplicaciones ofrecen.

En cuanto al primer factor, en el mercado se encuentran aplicaciones que recopilan diferente número de juegos sociales, como las ya mencionadas *Plato* (Figura 6) y *Mundijuegos*. Además, de esta manera también aumentan su recepción, ya que sus jugadores no verán la necesidad de cambiar de plataforma para jugar a otros juegos. Se puede observar en la imagen 1 parte de los juegos disponibles en *Plato*.

El segundo factor puede llegar a ser mucho más relevante que el primero, cabe recordar que estas aplicaciones ofrecen los llamados juegos sociales, por lo tanto, gran parte de su atractivo reside en su parte social. Utilizando la aplicación *Plato* (Figura 6) como ejemplo. Esta aplicación se puede considerar social debido a que soporta grupos de chat (2) y torneos temporales (3). Estos dos rasgos aumentan la competitividad entre los jugadores, incrementando de esta forma las interacciones entre jugadores y mejorando el ambiente social dentro de la aplicación.

Figura 6. Características sociales de la aplicación Plato

Otro factor que aumenta la competitividad entre los jugadores y está presente dentro de la gran mayoría de aplicaciones del estilo es la existencia de una economía interna. Esta consiste en la existencia de una moneda propia de la aplicación, necesaria para jugar partidas, realizando apuestas dependiendo de las cuales el equipo ganador obtiene un mayor o menor número de premio. Estas apuestas aumentan la competitividad entre jugadores, y las monedas pueden en muchos casos ser gastadas en otros juegos no competitivos o comprando cosméticos en la mayoría de los casos. Como ejemplo de esta característica, en la Figura 7 se muestra el menú de selección de apuesta en la aplicación

Mundijuegos, donde una vez elegida la modalidad de juego, se ha de elegir el nivel de monedas que apostar. En este caso se ha apostado una cantidad media de monedas.

Figura 7. Selección de apuesta en la aplicación Mundijuegos

2.2.2 Situación actual de la aplicación *Playjoy*

Playjoy se trata de la aplicación desarrollada por NoSpoon recopilatoria de juegos sociales en la cual se encuentra el juego desarrollado en este proyecto. Cuenta con un total de 10 juegos sociales, como el chinchón, chúpate dos, parchís o brisca. El programa cuenta también con un sistema de chats grupales, abiertos para todos los usuarios, al igual que chats privados y un sistema de rankings los cuales son anunciados públicamente una vez al mes. También cuenta con una economía interna, como se ha comentado anteriormente. Para más información sobre la empresa, refiérase al anexo 1, para información sobre la aplicación refiérase al anexo 2.

Playjoy ya se encuentra en diferentes plataformas en el mercado, contando ya con una base de más de 600 mil usuarios. La mayoría de estos procedentes de países hispanohablantes, sumando un total cercano al 65% de la base de jugadores. Esto se debe al alto número de juegos españoles incorporados a la vez que una publicidad casi exclusivamente en castellano.

La aplicación se puede acceder desde su página web, permitiendo acceder a su catálogo de juegos, reglas y jugar gratis, sin apuestas: PlayJoy - Juegos online clásicos para móvil y web. También se puede descargar su versión móvil, la cual permite acceder a la totalidad de sus funcionalidades y características: PlayJoy - Multiplayer games - Apps on Google Play.

El objetivo de la empresa es llegar al mayor número de jugadores, continuando la estrategia actual, se proponen el desarrollo de dos nuevos juegos populares en nuevos mercados. Estos son la belote y su variante, la coinché, siendo este el orden por el que serán introducidos a la aplicación. Con ellos se espera aumentar la cantidad de jugadores franceses en el sistema, aumentando su base y el interés multicultural de la aplicación.

En el momento del desarrollo de la belote, el sistema cuenta con <1% de jugadores franceses.

3. Propuesta de implementación

En este capítulo serán exploradas los riesgos y oportunidades que ofrecen del mercado actual de las implementaciones de la belote mediante un diagrama DAFO, extrayendo los requisitos a cumplir por la propuesta hecha por parte de la empresa y exponiendo las soluciones para estos requisitos. Además, se explicarán las diferentes fases sugeridas por la empresa con el objetivo de unificar y agilizar el desarrollo de futuros juegos.

3.1 Otras implementaciones de la belote

En el mercado de las aplicaciones de juegos sociales, se encuentran diferentes implementaciones de la belote correspondientes a los diferentes tipos de estas vistos anteriormente. Por lo tanto, se analizarán en este apartado los potenciales competidores de nuestro producto, se enfocará tanto en la interfaz como en la experiencia de uso que estas aplicaciones ofrecen. Primero se verán los casos donde la aplicación solo ofrece la belote y sus variaciones y luego un caso donde que se trata de un juego más dentro de los ofertados por la aplicación, siendo competidor directo de *Playjoy*.

En todos los ejemplos, se analizarán las tres situaciones principales del juego: la fase de selección de triunfo, la fase de juego y como los puntos son mostrados al final de cada ronda. Se analizarán los elementos más relevantes de cada fase, al igual que su impacto en la experiencia de uso. También se tendrá en cuenta la experiencia de chat con el resto de los jugadores, crucial en el entorno de aplicaciones sociales.

Belote.com

Como primer caso está *Belote.com* (Figura 8), desarrollada por GameDuell [12]. En esta versión se apostó por una organización horizontal de la interfaz, lo cual favorece ciertos aspectos de la interfaz del juego, pero empeora la experiencia de chat dentro de la partida.

Durante toda la partida, el juego muestra en cada lado de la pantalla los cuatro perfiles de los jugadores, utilizando en lado inferior para el jugador principal, únicamente mostrando el tiempo de turno del jugador inferior. También se puede observar en la parte inferior un botón que lleva al menú principal del juego a la izquierda y a la derecha dos botones para comunicarse con el resto de los jugadores, uno de textos y otro de emoticonos. Aquí se observa el primer problema de orientación horizontal, ya que esta impide abrir un teclado de móvil completo sin tapar toda la pantalla, quedando la comunicación reducida a frases preestablecidas (4) o emoticonos limitados.

[12] https://www.gameduell.fr/gd/belote.html

Durante la fase de elección de triunfo (1), el juego muestra un dialogo grande sobre la mesa, el cual muestra la carta que marca el triunfo al lado de las opciones de coger triunfo o pasar y rechazar. En esta solución queda muy claro cuál es la carta de triunfo y, con el correcto conocimiento previo, también quedan claras las funcionalidades de cada uno de los botones visibles, pero puede resultar un poco confuso si no estas familiarizado con las reglas.

En la fase de juego (2), el dialogo desaparece y se mostrarán en el centro de la mesa las cartas lanzadas, resaltando las cartas de triunfo con un ligero tono amarillo. Además, cada perfil tiene una cantidad de cartas que refleja cuantas cartas todavía tiene en la mano, mostrando las cartas propias boca a arriba, más espaciadas y ordenadas en orden descendente para poder verlas claramente. Después, en ambas esquinas superiores muestra las cartas de la jugada anterior (izquierda) y los puntos de cada pareja en la ronda actual y los totales de la partida (derecha).

Se encuentra un par de problemas durante esta fase, primero la dificultad para ver cuantas cartas tiene el resto de los jugadores, pues no están separadas si no acumuladas. A pesar de esto, queda reducido ya que en todo momento todos los jugadores poseen las mismas cartas, por lo que basta con contar las propias para conocerlo. Durante esta fase es crucial saber quién ha sido el jugador que ha aceptado el palo de triunfo y cuál ha sido este, en esta implementación no queda muy claro pues, pese a aparecer en el perfil de jugador que ha aceptado, el icono del palo resulta muy pequeño a la vista (se puede observar en el perfil superior en las imágenes 2, 3 y 4).

Por último, al final de cada ronda (3) se muestra un texto indicando si el contrato ha sido completado o no, al igual que mostrando los puntos que el equipo del jugador ha marcado. Junto con la comunicación, aquí reside el principal problema de esta implementación, pues el juego realiza muchas cuentas las cuales muchos jugadores quieren comprobar por su cuenta. Es cierto que se ve reducido el problema con la cuenta mostrada en la esquina superior derecha, la cual se actualiza mano a mano, pero otras cuentas como los anuncios o las diez últimas quedan ocultas al jugador.

Figura 8. Capturas de la aplicación Belote.com

Belote Multiplayer

El segundo caso se trata de la aplicación llamada *Belote Multiplayer* (Figura 9), desarrollada por IsCool Entertainment[13]. De nuevo, está implementación se presenta con una organización horizontal, compartiendo las ventajas y problemas con *Belote.com*.

Comenzando con los elementos comunes durante toda la partida, se observan los perfiles de los cuatro jugadores, mostrando el tiempo de turno de cada uno mediante la barra donde se sitúa su nombre. En la parte inferior derecha encontramos dos botones de comunicación, mostrando el mismo problema que la implementación anterior, pudiendo ofrecer solamente opciones limitadas de mensajes y emoticonos (4).

Pasando a la fase de selección de triunfo (1), se observa un dialogo con la carta de selección en medio de este y dos botones para aceptar o no el triunfo y la carta. Esta implementación mejora sobre la anterior ya que añade un texto indicando la acción a realizar, ayudando así a jugadores menos familiarizados con el juego, y dejando más clara la función de cada botón dentro del contexto de la acción a realizar.

Luego, durante la fase juego (2) se muestran en el centro las cartas lanzadas durante esa ronda, marcando las cartas del palo de triunfo con un tono amarillo. En cada mano de los jugadores se observan las cartas que tienen a su disposición, mostrando las del jugador inferior boca arriba, más espaciadas y en orden ascendente. Después, en las esquinas superiores se muestran los puntos totales de cada pareja respecto al total (izquierda) y las cartas de la jugada anterior (derecha).

Se encuentran durante esta fase los dos mismos problemas que la aplicación anterior, las cartas de los otros jugadores quedan poco claras a simple vista y el palo de triunfo junto a su elector resultan un poco complicados de encontrar, ya que en este caso se encuentra muy lejos de la acción principal, en la parte superior del icono del jugador que elige. Además, también resulta problemática la forma en la que las cartas son lanzadas al centro de la mesa, ya que pueden crear dudas de quien las ha lanzado (2).

Al final de cada ronda (3) se muestra si la pareja del jugador ha ganado o no la ronda, pero otra vez no se muestran los cálculos necesarios para llegar a esta conclusión de puntos. Además, esta implementación no muestra si el contrato ha sido completado o no a pesar de ser esta condición la cual marca si se ha ganado o no la ronda.

[13] https://www.iscoolentertainment.com/

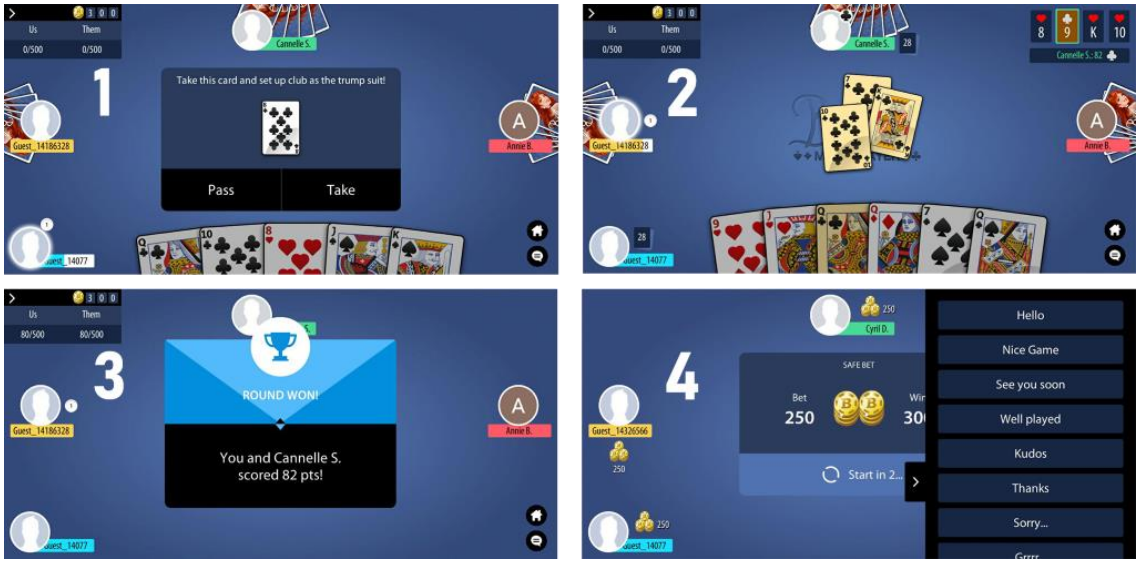

Figura 9. Capturas de la aplicación Belote Multiplayer

VIP Belote

Como último caso de aplicaciones únicas, se presenta *VIP Belote* (Figura 10), por Casualino Games[14]. A diferencia de las dos anteriores, esta aplicación presenta una organización vertical, siendo más natural al uso de un teléfono móvil.

Comenzando por los elementos comunes, esta implementación muestra como el resto los perfiles en cada uno de los lados de la pantalla, excepto cuando se abre el cuadro de comunicación, donde los perfiles se mueven a las esquinas (4). Los tiempos de turno se muestran en los rectángulos que alojan los nombres. En la esquina izquierda encontramos un botón para los ajustes dentro de la partida y la derecha los ya comunes botones de comunicación. Aquí encontramos una ventaja de la disposición vertical, ya que, a pesar de ser textos predefinidos de nuevo, la implementación ofrece un mayor número de opciones y estas son más complejas (4).

Durante la fase de selección de triunfo (1), esta implementación opta por un dialogo similar al primer caso, donde solo se muestra la carta de selección, el palo a elegir y un botón para pasar el turno. Otra vez, con un dialogo sin texto que indique la acción a realizar, la situación puede resultar extraña para los jugadores menos experimentados.

Durante la fase de juego (2) se observan de nuevo las manos de los jugadores de forma acumulada, dificultando saber el número de cartas por jugar, al igual que se muestran las cartas centrales apiladas, lo cual puede también resultar complicado de ver en ciertos casos. En las esquinas superiores se pueden ver las cartas de la jugada anterior (derecha) y los puntos actuales de la ronda y totales de ambas parejas (izquierda), debajo de los cuales se encuentra un gráfico de ayuda, el cual puede resultar de mucha ayuda para los jugadores menos experimentados. En este caso el palo de triunfo se encuentra en dos lugares diferentes, primero debajo de los puntos de ronda junto al elector, el cual resulta

[14] https://casualino.com/

fácil de ver, y en su propio perfil, mostrando la inicial del palo al lado del nombre del jugador.

Por último, al finalizar la ronda (3), se muestra un gráfico con un desglose de puntos un poco más completo que los anteriores, mostrando los puntos totales y los puntos conseguidos por el anuncio de belote. Pese a mostrar también los puntos de belote, este desglose sigue siendo bastante vago y falto de información.

Figura 10. Capturas de la aplicación Belote VIP

Mundijuegos

Como último caso está *Mundijuegos* (Figura 11), siendo el único caso de aplicación recopilatoria en incorporar el belote en su lista de juegos ofertados. El juego apuesta por una distribución horizontal de nuevo. Cabe notar que, debido a la falta de jugadores en esta plataforma, no se pudo encontrar una partida llena, teniendo que hacer las capturas en una partida con dos jugadores únicamente.

Los elementos comunes de esta implementación se observan en los perfiles de los jugadores, los cuales se muestran en cada uno de los lados de la pantalla de nuevo. En las esquinas superiores, hay un botón para desplegar el menú superior (izquierda), mostrando las opciones para abandonar partida, o jugar a juegos como el bingo o tragaperras, las cuales están incorporadas en dicho menú. A la derecha está el botón de chat, el cual abre en estos lados de la pantalla un chat con todos los jugadores simultáneos del juego, el cual permite hablar de forma natural y de cualquier tema en cualquier momento. A pesar de esta ventaja, debido a su disposición horizontal, una vez abierto el teclado toda la pantalla queda cubierta con él, permitiendo mostrar solo un mensaje en el chat, nulificando casi de forma completa la funcionalidad del chat (4).

En la fase de selección de triunfo, se muestra la carta de selección sobre la mesa, detrás del dialogo. Este dialogo indica la acción a realizar como pregunta, dando la opción de aceptar o rechazar, quedando así clara la finalidad de este. El único problema encontrado

ha sido la dificultad relativa para distinguir la carta que aparece como la carta de selección y no como una carta normal puesta en medio de la mesa.

Continuando con la fase de juego (2), se observa en la parte superior izquierda, bajo el nombre de "prize", el botín de monedas que el jugador ganador de la partida recibirá. En la parte inferior derecha se muestran las cartas de la jugada anterior. A diferencia de las otras implementaciones, esta muestra de forma más clara las cartas disponibles de cada jugador, al igual que el palo de triunfo, el cual se indica con el palo de triunfo sobre el perfil del jugador elector. Las cartas lanzadas al centro de la mesa tienen como novedad su valor debajo de ellas, ayudando de nuevo a jugadores menos experimentados. Cabe destacar que, a diferencia del resto de implementaciones, las cartas del jugador inferior no se encuentran con ningún orden, lo cual resulta curioso ya que empeora mucho su comprensión.

Al final de cada ronda (3) se presenta un dialogo indicando el ganador de la ronda y un desglose de todos los puntos ganados en la ronda. Este desglose se compone de los puntos totales de ronda, los puntos ganados en las bazas, los puntos de anuncios, los premios especiales (diez de últimas o capote) y el total de puntos.

Figura 11. Capturas de la aplicación Mundijuegos

3.1.1 Tabla resumen ejecutivo

Una vez vista la potencial competencia del producto en profundidad, se mostrará una tabla resumen de sus características. Estas están centradas en el apartado gráfico y funcional, ya que son las únicas características comparables desde un punto de vista del consumidor. Se dividirá la tabla en positivos y negativos de cada implementación para posteriormente realizar un diseño de la implementación propia maximizando los positivos.

Comenzando por los puntos negativos más problemáticos, se pueden identificar algunas características comunes presentes en las implementaciones. Primero de todo, muchas

opciones apuestan por un chat con mensajes predefinidos, lo cual soluciona problemas de mensajes incorrectos por parte de los usuarios, pero impide que estos puedan escribir libremente durante las partidas, eliminando en gran parte el factor social de las aplicaciones. Luego, durante las fases de selección de triunfo, algunas opciones pecan de ser demasiado simples, complicando la vida a los jugadores menos experimentados. Durante la fase de juego se observa en la mayoría de las implementaciones una presentación de las cartas en la mesa apiladas, haciendo más complicado saber que jugador ha lanzado cada una. También muchos de los casos fallan en mostrar los palos de triunfo elegidos y su selector, mostrando un icono demasiado pequeño en ocasiones. Por último, al final de las rondas, la mayoría de los diálogos mostrando los resultados resultan extremadamente sencillos, cuando el juego presenta un conteo de puntos demasiado complejo como para esta solución.

Pasando a los puntos positivos vistos en los ejemplos, se observa que con una organización vertical del juego se puede tener un sistema de dialogo más funcional, el cual, incluso siendo limitado por frases predefinidas, permite más opciones al igual que más complejas. Durante la fase de selección de triunfo, las implementaciones que optan por un dialogo de selección con una frase superior indicando la acción a tomar resultan más claros para todo tipo de jugadores, ayudando a los menos experimentados sin perjudicar la experiencia de los que ya conocen el juego. Luego, en la fase de juego, todos los ejemplos muestran las cartas de la jugada anterior, creando así un requisito a cumplir por la interfaz. También resulta un requisito la incorporación de algún tipo de gráfico de ayuda, mostrando el orden de las cartas si son o no de triunfo. La disposición de las cartas en la mesa también se ve como punto positivo, pues mostrar claramente quien ha lanzado cada carta ayuda a la experiencia de jugador. Por último, el ejemplo de *Mundijuegos* se observar el mejor dialogo de final de ronda, mostrando un resumen de todos los tipos de puntos ganados por cada pareja.

Implementación	Negativos	Positivos
Belote.com	- Chat muy limitado. - Diálogo de triunfo poco claro. - Número de cartas de otros jugadores poco claro. - Palo de triunfo y jugador selector poco claros. - Diálogo de final de ronda demasiado simple.	+ Cartas de la jugada anterior + Cartas de triunfo marcadas
Belote Multiplayer	- Chat muy limitado. - Número de cartas de otros jugadores poco claro. - Cartas en la mesa demasiado juntas. - Palo de triunfo y jugador selector poco claros. - Diálogo de final de ronda demasiado simple.	+ Dialogo de triunfo claro. + Cartas de la jugada anterior
VIP Belote	- Chat no libre. - Diálogo de triunfo poco claro. - Número de cartas de otros jugadores poco claro. - Cartas en la mesa demasiado juntas. - Palo de triunfo y jugador selector poco claros. - Diálogo de final de ronda demasiado simple.	+ Más y mejores opciones de chat. + Gráfico de ayuda con el orden y puntos de las cartas. + Cartas de la jugada anterior
Mundijuegos	- Chat inutilizado por la disposición horizontal. - Carta de selección de triunfo poco visible.	+ Chat libre. + Dialogo de triunfo claro. + Valor de las cartas indicado. + Cartas de la jugada anterior. + Cartas en la mesa separadas.

		+ Número de cartas de los otros jugadores claro. + Dialogo de final de ronda completo.

Tabla 1. Resumen del análisis a otras implementaciones

3.2 Análisis DAFO

El análisis mediante un diagrama DAFO (Figura 12) (debilidades, amenazas, fortalezas y oportunidades) es una herramienta que facilita y ayuda en el proceso de estrategia y desarrollo de soluciones a ciertos problemas. Este diagrama está formado por cuatro cuadrantes organizados en dos columnas (análisis interno y externo) y dos filas (positivos y negativos), mediante las cuales se exponen los resultados de un estudio de mercado y un estudio interno del equipo. Los cuadrantes negativos (debilidades y amenazas) buscan prever el peor caso en el que encontrarse, teniendo en cuenta las mayores debilidades internas junto con las amenazas del mercado externo. La parte positiva del diagrama, en oposición, busca mostrar las mayores posibilidades del proyecto, mostrando los resultados que apoyan un éxito del juego en este caso (12).

Comenzando por los mayores negativos que se presentan dentro de la empresa, en este se encuentra el desconocimiento por parte del desarrollador principal de las herramientas a utilizar, al igual que una falta de organización preestablecida para el desarrollo de los juegos, las cuales, junto al tamaño reducido de desarrollo pueden resultar en retrasos para la salida del juego. También la empresa intentará adentrarse en un mercado francés no explorado desde la salida de su antiguo proyecto de la belote años atrás. Fuera del entorno de desarrollo, como amenazas se encuentran la existencia de aplicaciones dedicadas a la belote, cuyos usuarios son fieles por características propias como torneos o diferentes modos de juego. Por el otro lado, las demás aplicaciones recopilatorias en el mercado están más establecidas y cuentan con equipos más grandes. Por último, el mercado de juegos de cartas digitales es muy abundante, lo cual suma a la complejidad de atraer a nuevos jugadores.

En los positivos, internamente la empresa posee muchos conocimientos en el desarrollo de juegos, incluso teniendo acceso a código fuente de implementaciones anteriores de la belote. Una fortaleza encontrada en el tamaño reducido del equipo se encuentra en el conocimiento por parte de todas las partes de realizar sesiones de testeo, ofreciendo diferentes puntos de vista durante estas. Ya que los juegos se realizan con bases de gráficos y código común, resulta sencillo dar comienzo a un nuevo proyecto como este. Fuera de la empresa, las mayores oportunidades se encuentran en el reducido número de aplicaciones recopilatorias que ofrecen la belote, las interfaces poco eficientes y unos jugadores de la belote acostumbrados a aplicaciones dedicadas, pudiendo ser captadas por las características de nuestra aplicación recopilatoria.

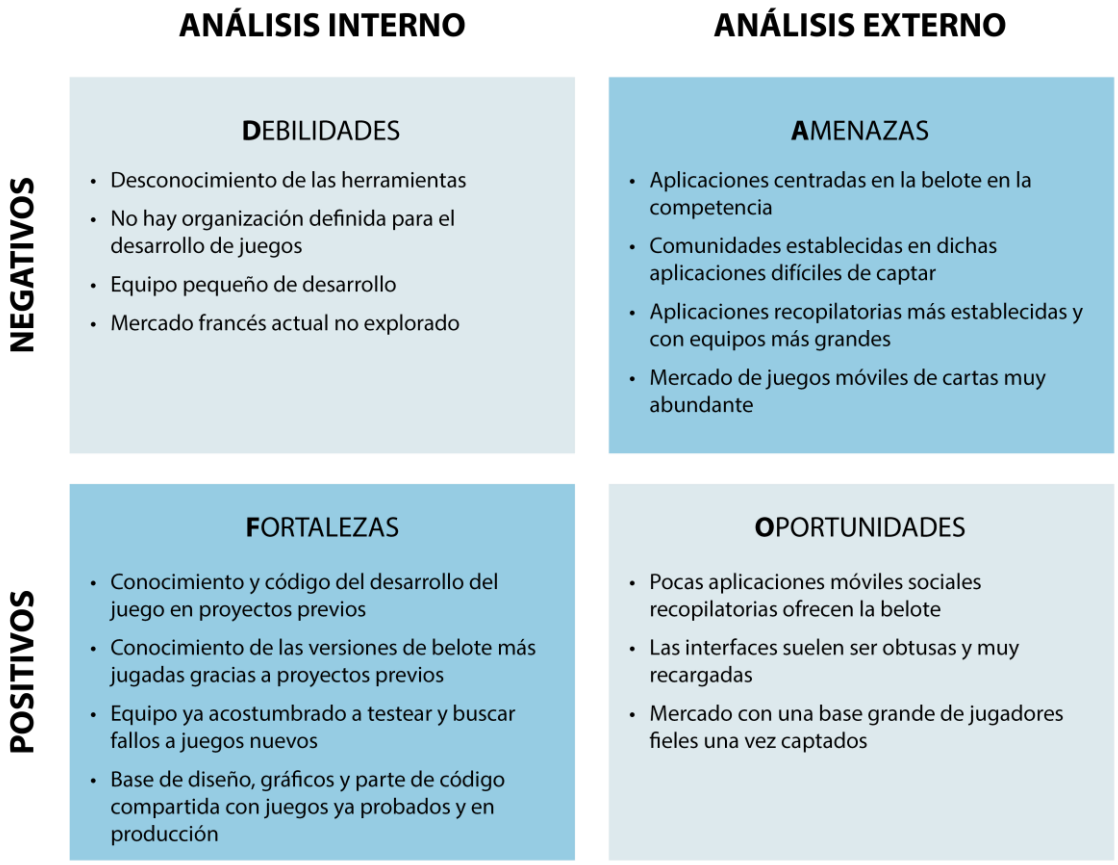

Figura 12. Diagrama DAFO del proyecto

3.3 Especificación de requisitos del juego

Una vez realizado el diagrama DAFO, se expondrán los principales requisitos a cumplir para poder así ofrecer una implementación de la belote más atractiva hacia el público general. Estos requisitos explorarán principalmente la parte gráfica y de experiencia de usuario. Son los siguientes:

- **Interfaz simple y clara**: nuestra implementación de la belote deberá seguir con la línea estética establecida por el resto de los juegos de la aplicación, ofreciendo unos gráficos simples y claros. Además, debido al valor cambiante de las cartas según el palo de triunfo, se incorporará una ayuda con el orden de valor dentro del juego.
- **Juego vertical**: como ha sido demostrado observado en el estado del arte, muchas implementaciones optan por una implementación horizontal, lo cual choca con parte de la naturaleza de los juegos de mesa. Estos juegos y, por ende, las aplicaciones que los ofrecen son altamente sociales, ofreciendo características como un chat durante la partida, al igual al encontrado en nuestra aplicación. Es por este motivo que la verticalidad del juego es de elevada importancia,

ofreciendo una experiencia común con el resto de la aplicación y una experiencia de chat más natural para todos los jugadores interesados.

- **Reconexión**: durante las partidas, puede ocurrir que los jugadores sufran de problemas de reconexión o tengan que utilizar diferentes aplicaciones en su móvil. Por lo tanto, el juego deberá ofrecer una experiencia sin interrupciones y lo más fluida posible, evitando obligar al jugador a hacer acciones que resulten innecesarias. Esto es, cuando un jugador abandona la partida momentáneamente y vuelve, la aplicación por su cuenta se encargará de reconectarle sin incidencias, devolviéndole al estado actual de la partida.
- **Posesión por parte de los bots**: el requisito anterior ataca el punto de vista del jugador que abandona la partida, este en cambio se centra en el punto de vista de los jugadores todavía dentro de la partida cuando esto ocurre. De nuevo, cuando un jugador abandona la partida, definitiva o momentáneamente, se ha de ofrecer una experiencia sin interrupciones, donde los jugadores fuera de la partida sean "poseídos" por una inteligencia artificial que juegue por ellos. De esta forma, los jugadores todavía dentro de la aplicación tendrán una experiencia de juego muy similar a la realidad.
- **Inteligencia artificial competente**: de vital importancia en la belote, se deberá ofrecer una inteligencia artificial altamente competente. Esto es por dos motivos principales: al tratarse de un juego por parejas la IA deberá poder ayudar a ganar la partida a su compañero humano; el otro motivo reside en el requisito anterior, donde se asegurar que la IA no perjudicará la partida que el jugador anterior estaba jugando, pudiendo volver a esta y no estar en desventaja.

3.3.1 Soluciones propuestas por la empresa

A los requisitos antes mencionados, la empresa buscó diferentes soluciones. Estas soluciones propuestas siguen los estándares establecidos por el resto de la aplicación, a excepción del requisito referido a la inteligencia artificial, para el cual se propuso una solución propia. Dichas soluciones serán expuestas a continuación:

Interfaz simple, clara y vertical

Para la elaboración de esta interfaz, se enumeraron las funcionalidades y la información que esta interfaz debería acomodar. Muchas de estas son comunes a todos los juegos que se encuentran en la aplicación, por lo que su implementación es común y no propia de la belote. De este modo, solo serán exploradas las decisiones propias de la belote, para las cuales se tomaron parte durante el desarrollo de este proyecto. Para más información sobre los gráficos comunes en todos los juegos, referirse al anexo 4.

Viendo el diseño general de la pantalla de juego en la Figura 13, se observan tres zonas principales: la mesa central (1), las manos de los otros jugadores (2) y nuestra propia mano (3). En la mesa central serán puestas las cartas jugadas por los jugadores, cada puesto haciendo referencia a la mano que tiene enfrente, además las cartas jugadas en la

baza anterior serán mostradas en la esquina superior derecha. En las manos de los oponentes aparecerán el número de cartas que poseen cara abajo, en caso de un oponente elegir el palo de triunfo, se añadirá a su mano el icono de este palo. Por último, en nuestra mano estarán las cartas en nuestra posesión cara arriba, al igual que encima de estas una barra indicando el tiempo restante de turno, los puntos actuales sobre los necesarios para ganar la partida y el icono del palo de triunfo en caso de elegirlo nosotros. Esta organización de elementos es común durante toda la partida y varían sus contenidos según la fase donde se encuentre esta. Se realizará ahora un análisis de los dos elementos con más detalles, siendo la mesa central y la mano del jugador.

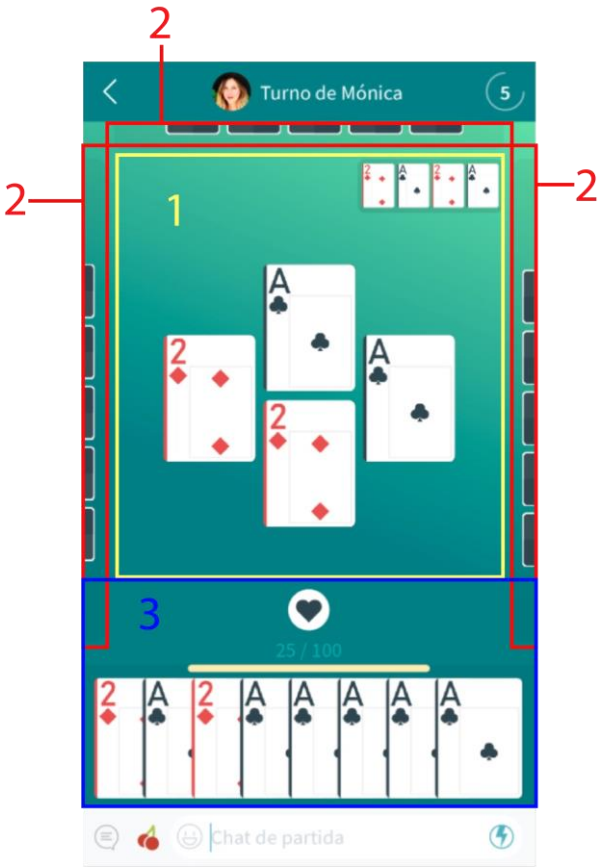

Figura 13. Diseño de la fase de juego de la belote con sus elementos principales marcados

Empezando por la mesa central, esta tiene dos estados principales según la fase donde se encuentre la partida, como se observa en la Figura 14. Durante la fase inicial de selección de triunfo (1), se mostrará la carta de selección en la posición de arriba de la mesa (1.a), con un dialogo de confirmación debajo de esta (1.b). Este dialogo a su vez tiene dos variaciones, dependiendo si la partida se encuentra en el primer (1.b1) o segundo (1.b2) turno de selección de triunfo. Durante el primer turno el dialogo se permitirá aceptar la carta y palo de triunfo o no, en el segundo turno mostrará un botón por cada palo, permitiendo el palo a gusto.

Durante la fase de juego (2), habrá una mesa central similar a la mostrada en la Figura 13. Las cartas se posicionarán separadas entre sí y en la posición del jugador que la lanzó (2.a) a medida que se van lanzando. Una vez lanzadas las 4 cartas, una animación

indicando la carta victoriosa se ejecutará, limpiando la mesa hacia el jugador que ha ganado la baza. En la esquina superior derecha se expondrán las cartas jugadas en la baza anterior (2.b), dando información muy útil al jugador.

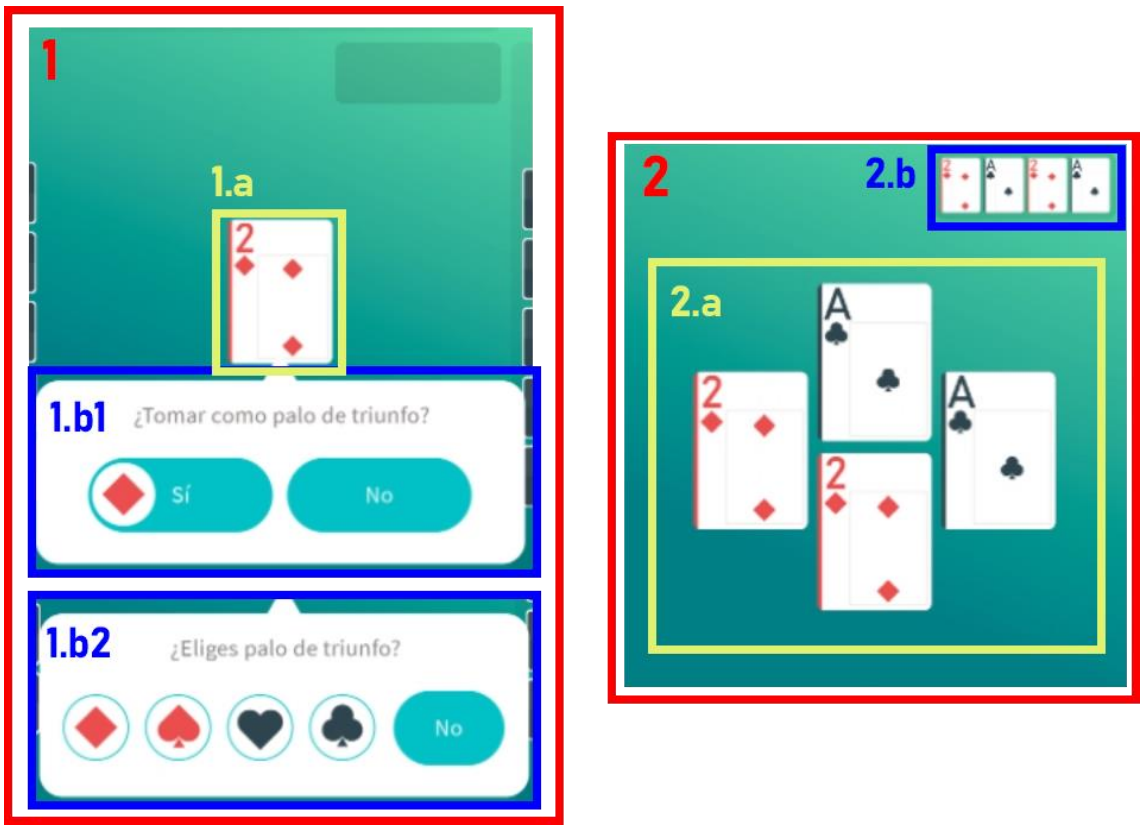

Figura 14. Fases y elementos de la mesa central

Pasando ahora al análisis de la mano del jugador inferior (Figura 15), se observa primero el palo de triunfo (1) una vez elegido con un icono grande sobre un círculo blanco que resalte respecto al fondo. Segundo, se mostrarán los puntos actuales de la pareja respecto del total justo debajo del palo de triunfo. Y justo debajo de estos está la barra indicativa del tiempo de turno restante. Por último, están las cartas que el jugador posee, las cuales se superponen entre ellas, utilizando la letra e icono de palo para indicar su valor a pesar de no poder ver la carta al completo.

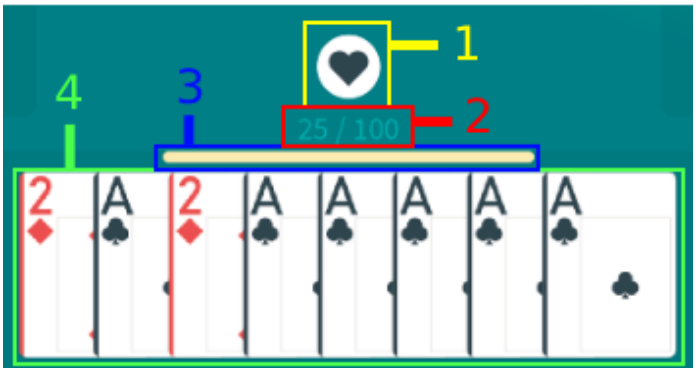

Figura 15. Mano del jugador y sus elementos principales

Pasando ahora a los gráficos de los anuncios (Figura 16), se muestran tres posibles casos. Primero está el caso del anuncio del belote (1), el cual es un anuncio especial, el cual se pensó implementar en un principio como una animación que ocupase la pantalla entera. Luego, en los otros dos casos, se muestra un anuncio ejemplo, con cartas y textos falsos. Pero se consigue mostrar la idea de tener un cartel con la información real del anuncio apareciendo del lado de la pantalla respectivo al jugador que lo realiza.

Figura 16. Diseño inicial de los anuncios

Al final de cada ronda, y como se ha comentado en los análisis del resto de implementaciones, se ha optado por un dialogo completo, el cual mostrará todos los diferentes tipos de puntos a tener en cuenta en el conteo (Figura 17). Estos son los puntos de las bazas, anuncios, premios y el total de puntos de la ronda, después, se indican los puntos marcados, dependiendo del contrato, y al final de todos los puntos totales de los que se disponen. Además, en este gráfico se indicará la pareja ganadora de la ronda mediante una animación que coloreará de amarillo su desglose de puntos.

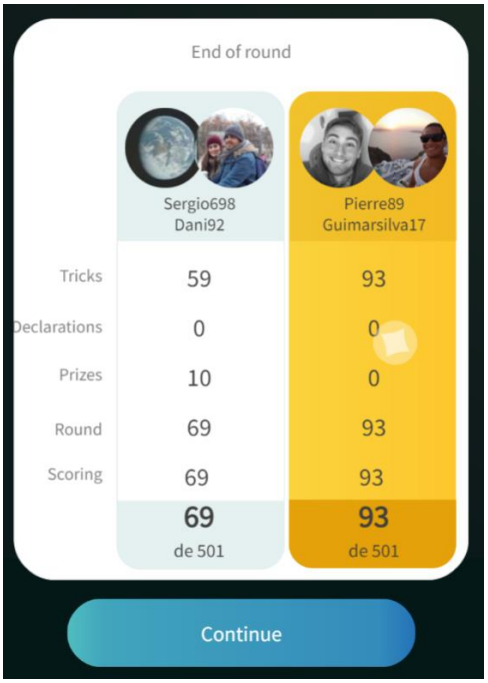

Figura 17. Dialogo de puntos en cada final de ronda

Reconexión y posesión por parte de los bots

Cuando un jugador se desconecta de una partida, es necesario asegurar una experiencia realista para el resto de los jugadores en partida y permitir al jugador desconectado volver a la partida de forma fluida. Se ha unificado la solución de estos dos requisitos como una única propuesta ya que tanto requisitos como soluciones van unidas de la mano. Se utilizará un diagrama de flujo de ejecución (Figura 18) para explicar la solución propuesta.

Empezando por el momento en el que un jugador decide salir de la aplicación, perdiendo momentáneamente la conexión con la partida. Entonces, el servidor recibirá desde el cliente un mensaje de desconexión, comenzando una cuenta atrás para la desconexión definitiva del jugador y creando un bot con el estado actual del jugador desconectado que jugará a partir de este momento. En este momento, las elecciones las realiza la inteligencia artificial, y ya que esta posesión se efectúa en el momento el jugador abandona la partida, el resto de los jugadores no notarán ningún cambio en sus partidas.

A partir de aquí existen dos caminos posibles, el jugador no vuelve antes de acabar la cuenta atrás, por lo que el bot pasará a jugar el resto de la partida por él; o el jugador vuelve a conectarse, retomando la partida con el estado actual después de haber pasado cierto tiempo. Analizando este caso, una vez el jugador vuelve, este realizará al servidor una petición del estado actual de la partida, recibiendo un mensaje de estado que le permitirá iniciar los gráficos correctamente, pudiendo continuar con la partida sin ningún inconveniente.

Todo este proceso puede ser llevado a cabo por cualquier jugador de la partida y en cualquier momento, y resulta crucial para el éxito del juego y la satisfacción de los jugadores.

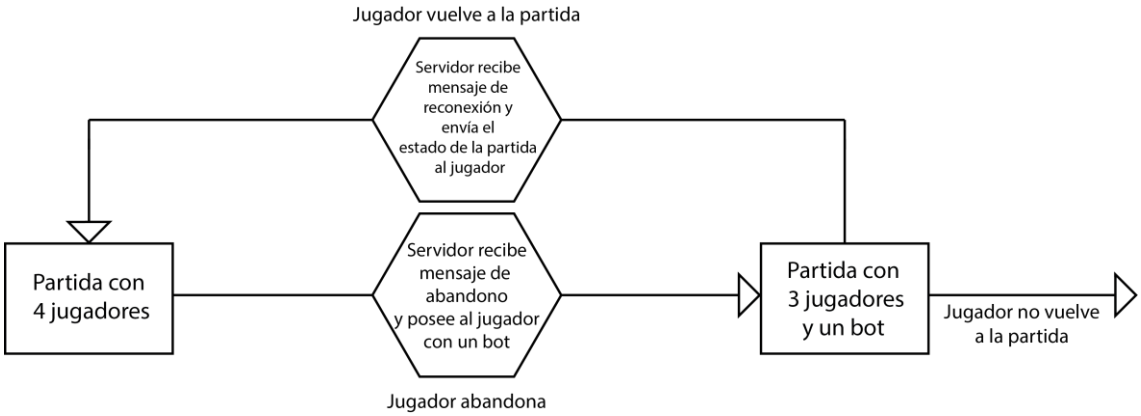

Figura 18. Diagrama de posesión y reconexión de jugadores

Inteligencia artificial competente

Como se indica en el requisito, la belote requiere de una IA competente al tratarse de un juego por parejas, donde las acciones de uno de los jugadores tienen un elevado impacto en el desarrollo de la partida. Por este motivo, se tiene como requisito crear una inteligencia artificial que pueda no solo ofrecer un reto entretenido a los jugadores contra los que jugará, si no también ser de la mayor ayuda posible a sus compañeros. Todo esto se tendrá que realizar de la forma más equilibrada posible, sin favorecer unos casos sobre los otros.

Para resolver este requisito, se propuso desarrollar una IA siguiendo una estrategia diferente a las presentes en el resto de los juegos de la aplicación. Esta nueva dirección hará uso de heurísticas para llegar a la decisión que tomar, las cuales a pesar de no ser completamente perfectas ofrecerán una acción lo suficientemente aceptable como para ofrecer un reto al igual que una ayuda en el caso necesario. Estas heurísticas son algoritmos mediante los cuales los bots de las partidas valorarán cada una de las acciones disponibles en dado momento, eligiendo entonces la que creen más favorable.

Para cada elección, los bots tendrán unos parámetros clave para la elección de la acción a realizar. Estos se refieren a las estadísticas para tener en cuenta por un jugador a la hora de razonar su jugada, p.e. ayudar a tu compañero, seguridad con la que una carta especifica ganará la ronda, entre otros. A su vez, los parámetros tendrán unos pesos asignados, indicando la importancia que dar a dicha estadística. Este método permitirá diseñar el funcionamiento de la IA a partir de una estrategia descrita en lenguaje natural.

Este acercamiento ofrecerá al equipo de desarrollo y mantenimiento de la belote una forma sencilla de modificar la forma de actuar de la IA en un futuro.

3.4 Fases de desarrollo propuestas

Poder tener una idea inicial de cuándo el juego podrá ser finalizado y planear su lanzamiento resultaba una tarea complicada de llevar a cabo sin unas fases de desarrollo bien definidas. Siendo uno de los objetivos principales del proyecto, la empresa propuso implementar un flujo de trabajo que pasaría a ser común para todos los juegos desarrollados en un futuro.

Esta mejora nació de la necesidad de mejorar la comunicación con todas las partes del equipo, permitiendo mejorar la situación actual pudiendo realizar mejores estrategias de negocio y una mayor transparencia para el equipo. Además, esta propuesta obligaría a un mínimo de tres a cuatro sesiones de prueba con diferentes partes del equipo, asegurando la calidad de cada una de las partes desarrolladas en cada fase.

3.4.1 Desarrollo ágil dentro de la empresa

Actualmente la gran mayoría de las empresas de desarrollo software se centran en una organización de trabajo y equipo cada vez más ágiles, hecho que se ve acentuado en las compañías más jóvenes, que ya nacieron con tales metodologías en mente. Nuestro caso no es excepción, dentro de la empresa existe una organización de equipo como "organismo" (13), en el cual los trabajadores se organizan por equipos de trabajo y no hay una jerarquía piramidal definida. En estas organizaciones los diferentes equipos dentro de la empresa trabajaran entre ellos, coordinados, pero no liderados por lo que tradicionalmente sería visto como una figura de jefe.

Dentro de *Playjoy* hay definidos cuatro equipos: marketing y atención al cliente, control de datos, aplicación y videojuegos. Entre estos equipos, hay trabajadores compartidos, esto se debe al reducido número de trabajadores y la gran ambición del proyecto. Este documento se centrará en el equipo de los videojuegos, puesto que la belote forma parte de este.

El equipo de videojuegos está compuesto por seis personas, tres desarrolladores, una artista, un diseñador y un tester. A la hora de desarrollar un nuevo juego, toman acción todas las partes del equipo, y a grandes rasgos existe ya un flujo de trabajo definido y funcional. Primero, el diseñador decide como ha de ser el juego, que funcionalidades poseerá y como será gráficamente, después el artista se dedicará a crear los gráficos decididos por el diseñador, poniéndolos a disposición de los desarrolladores. Es entonces cuando comienza la segunda fase, desarrollando e implementando el juego tanto en la aplicación como en la web. Por último, el tester trabajará con los desarrolladores probando el juego, indicando los fallos hasta que el juego cuenta con el visto bueno del diseñador.

3.4.2 Nueva organización del desarrollo de los juegos

Es en la fase de desarrollo y testeo donde la empresa buscaba una mejora, ya que anteriormente no quedaba claro el estado actual del desarrollo, impidiendo al equipo de marketing saber cuándo realizar las campañas sobre dicho juego y organizarse en un

futuro. Para encontrar solución a este problema, la empresa decidió reunirse al comienzo del desarrollo para probar una nueva organización en el desarrollo de los juegos, dividiendo el desarrollo en tres fases, cada una abarcando unos objetivos bien definidos y unos tiempos de desarrollo.

Además, cada fase contará con una ronda de prueba realizada por diferentes partes del equipo, asegurando una versión estable y limpia de fallos cuando se pasa al desarrollo de la siguiente. Estas fases son: pre-alfa, alfa, beta, versión 1.0 o "Gold" (14) y una cuarta fase opcional, la versión 1.1, la cual incorporaría los elementos no estrictamente necesarios para la salida del juego.

Estas versiones componen los MVP de este proyecto, dirigidos a diferentes partes del equipo, marcando diferentes hitos a completar definidos según la parte del equipo. Un MVP (siglas de *Minimum Viable Product* en inglés) consiste en partes del desarrollo, sin necesidad de solo ser programación, que validan el correcto funcionamiento de esta y sirven para focalizar el esfuerzo en un objetivo válido y de mínimo esfuerzo (15).

Se pasa ahora a definir el alcance de cada versión del desarrollo, al igual que su parte del equipo objetivo.

Pre-Alfa

Esta primera fase está centrada en el desarrollo de la lógica del juego, con el objetivo final de poder realizar simulaciones de partidas completas con bots que jueguen de forma aleatoria. Como se explica en el siguiente punto, los juegos están organizados en dos módulos principalmente, servidor y cliente, siendo el primero quien ejecuta la lógica de la partida y donde los bots toman sus decisiones.

Por lo tanto, esta fase se enfoca en el desarrollo y funcionamiento correcto de esta lógica, sentando las bases del funcionamiento del juego. Para comprobar el correcto funcionamiento se dispone de tests unitarios como de tests de estrés, pudiendo simular miles de partidas, comprobando que estas se completan correctamente y no surgen errores fatales.

Una vez llegados al punto donde las partidas se pueden simular sin problemas fatales, el código es presentado al equipo objetivo. Estos son los técnicos los cuales comprobarán que el código cumple con los estándares necesarios y realizarán los tests unitarios y simulaciones que vean necesarios.

Alfa

Siendo la primera versión donde se dispondrán de ambas partes principales del juego (cliente y servidor), esta versión se denominará alfa. Buscando tener la primera iteración de las funcionalidades más básicas del juego completo, sin ser su forma final y a falta del resto de funcionalidades. Se desarrollará una versión jugable, con falta de funcionalidades no vitales y todavía con un gran número de errores por comprobar.

En el caso de la belote, las funcionalidades básicas se definieron como la capacidad de lanzar cartas en el turno del jugador, tener una fase de selección de triunfo funcional, que los anuncios se realicen correctamente sin gráficos y se puedan acabar partidas jugando el número necesario de rondas. Ya que estas funcionalidades han sido desarrolladas y testeadas en profundidad en la lógica, se centrarán todos los recursos en el desarrollo del cliente y los gráficos del juego.

Una vez ya implementadas funcionalidades, se realizarán sesiones de testeo, donde el equipo técnico y artístico podrán participar. Esta unión se debe a la naturaleza mezclada del cliente, el cual necesita reaccionar y mantener un estado propio de la partida a la vez que mostrar los gráficos correctamente.

Beta

El objetivo de esta tercera fase es disponer de un juego con todas sus funcionalidades, las cuales se encuentran en su versión final, pero sin refinar, faltando algunos errores menores por solucionar y ligeros cambios de diseño. Las principales funcionalidades que añade esta versión son la reconexión al juego y la primera versión de unos bots con una inteligencia artificial competente.

La primera funcionalidad es vital para el lanzamiento del juego, ya que la gran mayoría de jugadores en algún momento de sus partidas cambiarán de aplicación o sufrirán una pérdida de conexión. Es aquí donde entra la reconexión, funcionalidad que permite volver a la partida una vez abandonada temporalmente, pudiendo continuar sin problemas.

La segunda de ellas también resulta vital, sobre todo al poco después de lanzar el juego, ya que muchas partidas necesitarán de estos bots competentes para rellenar sus jugadores. Por lo tanto, ya que muchas partidas contarán con bots y jugadores reales, la IA ofrecerá una experiencia satisfactoria a estos.

Además, en esta versión se pulirán el resto de las funcionalidades ya existentes, añadiendo y corrigiendo gráficos principalmente. Para probar esta versión beta, se realizarán de nuevo sesiones de testeo con el equipo técnico y artístico, igual que en la versión anterior.

Versión 1.0 o "Gold"

Esta versión marca la primera iteración viable para lanzar en la aplicación, su objetivo es tener el juego completo sin ningún error fatal, además de la gran mayoría de errores de lógica y gráficos arreglados. Se busca ofrecer un juego jugable y sin bugs conocidos. Por este motivo, la principal tarea a realizar en esta fase serán testeos con el equipo de pruebas y calidad, el cual se compone de técnicos, artistas y soporte de usuario, estresando el juego lo máximo posible antes de s lanzamiento.

A medida que los fallos vayan apareciendo, el desarrollador deberá ir arreglándolos, es por esto por lo que se las sesiones de testeo durante esta fase serán semanales y comprobarán la corrección de los bugs conocidos al igual que buscar nuevos errores. Todo error, por mínimo que sea, será registrado y después el diseñador del juego junto al desarrollador priorizarán estos errores registrados, pudiendo dejar errores para un futuro en caso de no poder ceñirse a los tiempos establecidos.

Una vez finalizada esta versión, el juego podrá ser implementado en la aplicación. Es aquí donde dependiendo de los errores todavía por arreglar, se decidirá implementar el juego directamente con su economía activada o se esperará a tener la siguiente versión finalizada para incorporar esta funcionalidad a nivel de aplicación.

Versión 1.1

En esta cuarta y última fase del desarrollo, se trabaja ya con una versión lanzada, y el objetivo pasa a ser añadir funcionalidades adicionales, las cuales poseen el resto de los juegos, pero no son vitales para el lanzamiento de estos. Se componen de dos funcionalidades, el tutorial del juego y los retos de este.

La gran mayoría de juegos en la aplicación cuentan con un tutorial, el cual se juega la primera vez que un usuario inicia el juego en específico. Mientras que las situaciones de los tutoriales son forzadas en el código, se busca ofrecer una experiencia lo más realista posible, mostrando todas las funcionalidades del juego, preparando a los jugadores primerizos a ser capaces de jugar una partida normal al acabar.

Los retos, por otra parte, añaden valor a la economía de la aplicación, incentivan a los jugadores a probar nuevos juegos y aportan experiencia para el avance de niveles de los usuarios. Estos retos ofrecen acciones a realizar dentro de los juegos a cambio de monedas y experiencia, y son añadidos a una lista propia de cada usuario. Los retos se organizan por juegos, ofreciendo un total de ocho retos diarios, con un máximo de dos retos por juego.

Una vez finalizada esta versión, el juego se encuentra en la fase de mantenimiento del juego, donde se controlará semanalmente el progreso de los jugadores, y se atenderán los bugs que van a apareciendo.

3.4.3 Diagrama Gantt con las versiones propuestas

El diagrama Gantt nació a principios del siglo 20, con el objetivo de visualizar los problemas de organización del trabajo y recursos dentro de una empresa. Actualmente, se utiliza principalmente para organizar tareas dentro de un equipo, pudiendo indicar los recursos a utilizar en cada fase y su duración. Además, esta solución resulta sencilla de desarrollar y su vista general ayuda a la comprensión de la a veces alta complejidad estructural de un proyecto (16).

En la Figura 19 se presenta el diagrama Gantt creado con los tiempos estimados para cada fase del desarrollo de la belote. Este diagrama resulta muy simple ya que el juego será desarrollado de forma lineal por un solo miembro del equipo, resultando en un diagrama sin solapamientos y sin necesidad de marcar los recursos necesarios para cada fase.

Siguiendo el diagrama, el desarrollo de la belote está previsto a comenzar en enero de 2021, tomando este primer mes para desarrollar la fase alfa. La duración de un mes viene marcada por la relativa sencillez del desarrollo de la lógica, pues esta es similar a otros juegos ya en la aplicación como el tute o la brisca. Seguido de esta, se encuentra la versión alfa, la cual muestra una estimación temporal de otro roadmap mensual. De nuevo, esta estimación viene definida por la reutilización de los gráficos comunes en ortos juegos, los cuales permiten la incorporación de una versión base funcional en un corto periodo de tiempo.

Seguida de la alfa, se encuentra la versión beta, la cual resulta ser la fase más compleja, ocupando un total de tres roadmaps. Esta larga duración se debe a la incorporación de todos los gráficos específicos del juego, necesitando la aprobación de cada implementación por parte del equipo de diseño del juego. Además, esta versión también busca incorporar las dos tareas más complejas y delicadas de implementar, la reconexión y una primera versión de la inteligencia artificial.

Por último, están las versiones Gold y 1.1 (mostrada en la Figura 19 como V1.0 por un error), estas dos versiones vuelven a ocupar un mes de planificación. En el caso de la versión Gold, esta busca solucionar todos los fallos relevantes previos al lanzamiento del juego, por lo que se tratará de un roadmap únicamente de solución de errores. Para la versión 1.1, la cual incorpora retos y tutorial, los cuales al igual que los primeros casos, su desarrollo es facilitado por los elementos creados en otros juegos de la aplicación.

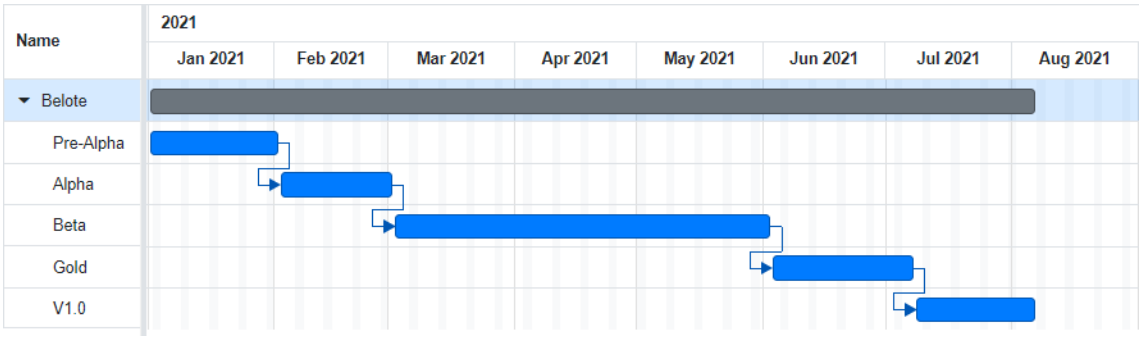

Figura 19. Diagrama de Gantt mostrando la duración esperada de cada fase

4. Diseño de la implementación

4.1 Estado de la tecnología en la actualidad y comparativa

Pese a que el proyecto ha sido desarrollado dentro de una empresa ya con una infraestructura bien definida que justifica finalmente las tecnologías usadas, vale la pena realizar un pequeño estudio de mercado. En este capítulo se analizarán las diferentes tecnologías disponibles actualmente para el desarrollo de videojuegos que cumplirían con los requisitos del proyecto.

El estudio se centrará en las herramientas para el desarrollo del *front-end* de los juegos, ya que es la parte que funciona independientemente del resto de la aplicación. La parte del *back-end* del proyecto se comparte con videojuegos y aplicación, lo cual requeriría de otro estudio completamente diferente, tratando temas no relevantes para el desarrollo de los juegos.

Debido a la necesidad de desarrollar sobre diferentes plataformas (iOS, Android y web), solo se tuvieron en cuenta las herramientas que permiten este tipo de desarrollo. Tras un breve estudio, las herramientas más utilizadas son (17):

- **Unity**: uno de los motores gráficos con mayores posibilidades y comunidades de la actualidad. Ofrece un medio de desarrollo basado en C# y una de las mayores piscinas de plataformas sobre las que publicar los juegos.
- **Unreal**: con sus últimas versiones, este motor gráfico gratuito se ha establecido como uno de los más potentes en el entorno gráfico. Utiliza C++ como lenguaje de desarrollo y, aunque menor que la lista ofrecida por Unity, presenta un amplio catálogo de plataformas sobre las que publicar.
- **Flutter**: futuro framework oficial para el desarrollo de software para Android, Flutter permite desarrollar aplicaciones multiplataforma, para web, iOS y Android.
- **Xamarin**: framework de desarrollo software multiplataforma propiedad de Microsoft, que utiliza C# y .NET como bases de su código.
- **HTML5/CreateJS**: framework base para todo desarrollo web, conjunto a JavaScript y CSS. CreateJS es una librería para desarrollo HTML5 que permite trabajar los gráficos a través del código JavaScript presente en la web.

	Unity	**Unreal**	**Flutter**	**Xamarin**	**HTML5/CreateJS**
Lenguaje	C#	C++	Dart	C#	JavaScript
Comunidad	Grande	Grande	Media	Grande	Grande

Curva de aprendizaje	Mínima	Elevada	Elevada	Media	Media
Coste multiplataforma	Medio	Elevado	Mínimo	Mínimo	Mínimo
Simpleza estructural	Media	Media	Media	Media	Media

Tabla 2. Comparativa de las tecnologías multiplataforma actuales

Viendo la tabla, se observa que tanto Xamarin como HTML5 ofrecen el mejor balance entre todas las características. Entre estas dos, la empresa decidió decantarse por HTML5 debido a la previa experiencia de los desarrolladores junto con el gran abanico de posibilidades que abre debido a su enorme comunidad y su historia. La empresa contaba con mucha experiencia desarrollando los juegos sobre la plataforma Flash, por lo que para este nuevo proyecto buscaron alternativas cuyo flujo de trabajo fuese lo más similar posible. Fue dentro del entorno HTML5 junto al grupo de librerías de JavaScript llamadas CreateJS.

Pasando al desarrollo del *back-end*, donde se soporta la lógica de los juegos y se obtiene el estado de sus partidas utilizando Amazon Web Services (AWS). El servidor utiliza el servicio AWS Elastic Beanstalk para la implementación de su aplicación Java, ejecutándose en los computadores en la nube ofrecidos por el servicio Amazon EC2.

4.2 Tecnologías utilizadas

Ahora se profundizará en las tecnologías utilizadas, explicando en qué consisten cada una y cuál es su uso dentro del proyecto. Al tratarse del desarrollo de un videojuego, se han utilizado tecnologías que abordan los diferentes aspectos dentro de su desarrollo. Empezando con el lenguaje utilizado para programar todo el proyecto, su entorno de desarrollo y herramienta de automatización, y acabando con el apartado gráfico, exponiendo el conjunto de librerías gráficas usadas y la herramienta para diseñar los gráficos.

4.2.1 Kotlin

Se trata de un lenguaje multiplataforma, con tipado estático, de uso general y con inferencia de tipos. Nació de las manos del líder de JetBrains[15] Dmitry Jemerov, con el ambicioso objetivo de ser la evolución y futura sustitución de Java.

Con tal de poder cumplir su objetivo y facilitar el cambio a este nuevo lenguaje para empresas ya establecidas, el código es compilado para ser interpretado con la JVM (*Java Virtual Machine*) desde su nacimiento en 2011. Más tarde, en 2017, la potencia del lenguaje fue ampliada añadiendo la capacidad de compilar su código directamente a JavaScript, pudiendo utilizar este código compilado. Esto permite poder utilizar este

[15] https://www.jetbrains.com/

código JavaScript dentro de una plantilla HTML, pudiendo así conseguir un código JavaScript seguro de fallos de tipos, muy comunes en aplicaciones web.

El aspecto multiplataforma, fue el principal motivo por el que fue elegido lenguaje único para el proyecto entero de *Playjoy*, permitiendo a todos los aspectos de la aplicación utilizar un lenguaje común. Como ha sido mostrado en el estudio de las tecnologías disponibles para el desarrollo de los videojuegos, estos utilizan HTML5 junto a CreateJS para la implementación de los gráficos, y el servicio Elastic Beanstalk para la implementación del servidor. Ambas plataformas utilizan diferentes lenguajes, JavaScript para los gráficos y Java para el servidor, lenguajes a los que Kotlin permite ser compilado.

4.2.2 CreateJS

CreateJS es un Framework para la visualización de gráficos y animaciones dentro de páginas web. En conjunto con Adobe [16], la empresa GSkinner[17] creó esta agrupación de librerías para poder trabajar con los gráficos creados en la plataforma Adobe Animate. Se compone de un total de 4 librerías, EaselJS, TweenJS, PreloadJS y SoundJS, cada una con unos objetivos en específico. Para los juegos dentro de la aplicación, la empresa utiliza EaselJS, TweenJS y SoundJS, siendo las dos primeras las más utilizadas.

EaselJS se utiliza para cargar los gráficos exportados de Adobe Animate mediante el código JavaScript de la aplicación, pudiendo incorporar estos a la escena del juego. Además, esta permite utilizar estilos CSS en caso de que se busque modificar el posicionamiento o tamaño de los gráficos. Una vez los gráficos están en la escena, se utiliza la librería TweenJS para animarlos, habiendo creado una librería propia de animaciones comunes entre los diferentes juegos. Para reproducir los diferentes sonidos del juego se utiliza la librería SoundJS.

4.2.3 AWS Elastic Beanstalk y Amazon EC2

Desde 2002, Amazon ofrece los llamados *Amazon Web Services*[18] (AWS), poniendo a disposición de los clientes diversas herramientas para sus páginas web. Pero es en 2011 cuando comenzaron a ofrecer los servicios *Elastic Beanstalk*[19], un conjunto de servicios más pequeños que presentan un entorno de despliegue rápido y sencillo de utilizar. Los servicios ofrecidos son *Amazon Elastic Compute Cloud* (EC2), *Amazon Elastic Container Service* (ECS), *AWS Auto Scaling* y *Elastic Load Balancing* (ELB). Gracias a estos, *Elastic Beanstalk* controla de forma automática características como suministro de recursos, balance de carga, escalado automático o monitorización de la aplicación.

[16] https://www.adobe.com/
[17] https://gskinner.com/
[18] https://aws.amazon.com/
[19] https://aws.amazon.com/elasticbeanstalk/details/

Para desarrollar las aplicaciones web que el servicio puede desplegar se ofrecen una amplia lista de lenguajes de programación, entre los que se encuentra .NET, Python, PHP o Java. Este último es el utilizado para desarrollar el *back-end* de la aplicación, donde se encuentra la lógica de gestión de partidas al igual que la lógica de estado de cada partida.

4.2.4 Gradle

A la hora de trabajar en proyectos utilizando diferentes tecnologías, existen herramientas de gran ayuda que permiten realizar las diferentes compilaciones necesarias en una misma tarea. Gradle[20] se trata de estas herramientas, basado en los conceptos de Apache Ant y Apache Maven, con la diferencia de incluir un lenguaje específico para la programación de las tareas a realizar. Esta herramienta soporta el trabajo con Java (a la vez que otros lenguajes compilados para JVM como Kotlin), C/C++ y JavaScript.

El motivo por el cual fue elegida en vez de otras opciones fue el soporte oficial para Kotlin con el que cuenta a modo de plugin. Este facilita la automatización de la compilación en código para la JVM y JavaScript, permitiendo elegir que archivos se compilan en cada lenguaje. De esta manera se puede trabajar de forma uniforme con las dos partes dentro de un mismo proyecto.

[20] https://gradle.org/

4.2.5 IntellIJ IDEA

Los IDE (*Integrated Development Environment*) se tratan de programas de desarrollo que presentan cierto número de facilidades para el desarrollador. Estas consisten al menos de un editor de texto, herramientas de automatización y un debugger, aunque muchos de estos programas también incorporan los compiladores y/o interpretes necesarios para la ejecución de los programas.

IntellIJ (Figura 20), IDE desarrollado por JetBrains, soporta un gran número de lenguajes, entre los cuales se encuentran Java, Kotlin, HTML, CSS, JavaScript, pudiendo ser aumentada esta lista a través de plugins. Es por este soporte tan amplio de lenguajes que fue elegido herramienta para el desarrollo del videojuego propuesto. Este soporte consiste principalmente en ayudas a la hora de escribir el código, recomendando nombres de variables, métodos y/o variables disponibles en el momento o indicando errores en tiempo de programación y recomendaciones de buenas prácticas.

Otro aspecto importante esta herramienta es la habilidad para ejecutar test unitarios del código Kotlin escrito, pudiendo reunirlos todos en un único archivo. Esta habilidad junto al uso de puntos de parada permite focalizar las pruebas en métodos específicos para comprobar su correcto funcionamiento.

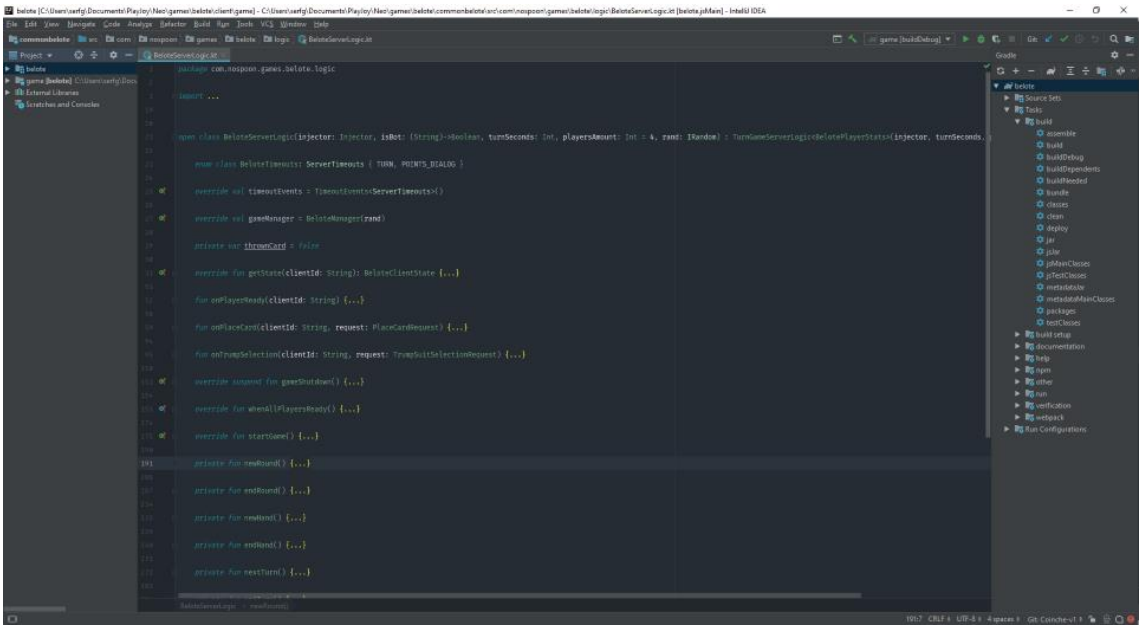

Figura 20. Captura del entorno de trabajo IntellIJ con la clase "BeloteManager" abierta.

4.2.6 Adobe Animate

Dentro del paquete de aplicaciones software que ofrece Adobe se encuentra Adobe Animate (Figura 21). Originalmente llamado Adobe Flash Professional, este producto se utiliza para el diseño de vectores y realizar animaciones con ellos. Por este motivo, ha sido ampliamente utilizado para realizar series animadas, publicidad o videojuegos, como se trata en este caso.

La elección de esta herramienta recae principalmente en el previo conocimiento de Adobe Flash Professional por parte de los artistas del trabajo. Además, ofrece un entorno de trabajo familiar gracias a pertenecer a la familia Adobe, donde los programadores también pueden realizar cambios de referencias a los gráficos al igual que ligeras modificaciones a estos.

Gracias a la interoperabilidad de Adobe Animate con las librerías de CreateJS, los gráficos creados en esta herramienta son exportados en archivos JavaScript. Estos son después referenciados desde el código del cliente de los juegos para cargar los gráficos necesarios y ubicarlos en su posición. Los programadores utilizan esta herramienta en numerosas ocasiones para cambiar las referencias a los gráficos al igual que retocar mínimamente ciertos gráficos.

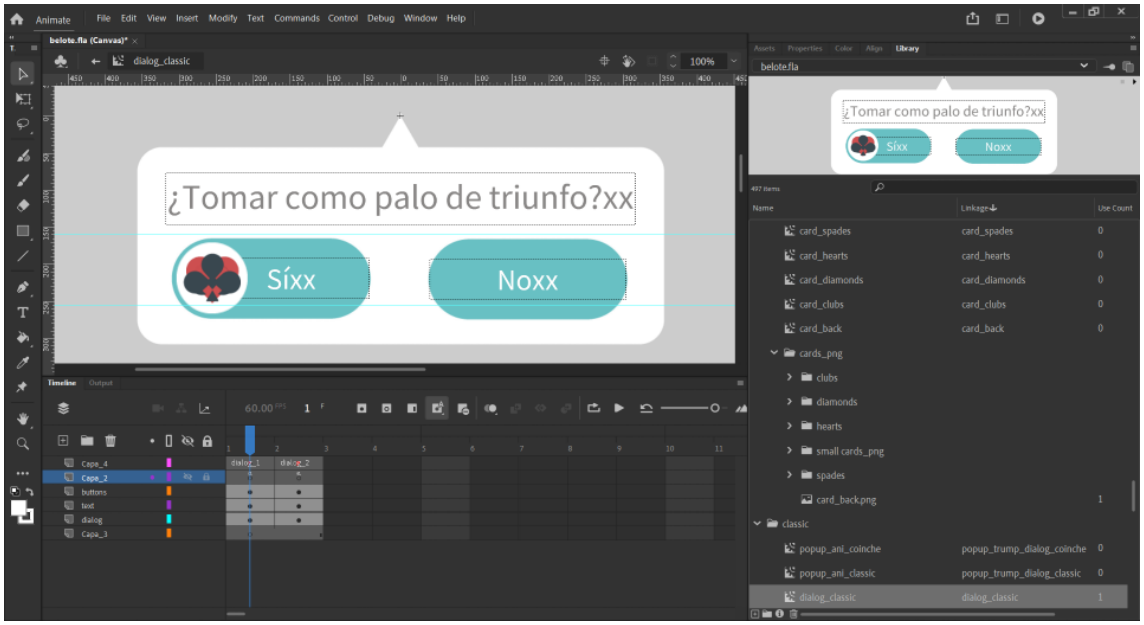

Figura 21. Adobe Animate con el gráfico "dialog_classic" abierto.

4.3 Arquitectura de la aplicación *Playjoy*

La aplicación *Playjoy* se encuentra organizada en tres grandes bloques como se observa en la Figura 22: servidor o *back-end*, aplicación (web y móvil) y juegos, formando estos dos últimos en conjunto la parte del cliente o *front-end*. Además, se dispone de un módulo web administrador ofreciendo una gestión visual para elementos de la aplicación controlados únicamente por el servidor, como retos, la economía o el bloqueo manual de usuarios maliciosos. Entre los diferentes clientes y el servidor se utilizan diferentes lenguajes, lo cual gracias a las capacidades de desarrollo multilenguaje de Kotlin no resulta un problema a la hora del desarrollo.

Con estos diferentes módulos se ha creado un ecosistema para la aplicación, donde estos pueden comunicarse e interoperar entre ellos. Este ecosistema está compuesto por las diferentes versiones de la aplicación móvil, la versión web de la aplicación (la cual difiere en gran medida de la versión móvil), la aplicación web donde se ejecutan las partidas de los juegos y los servicios proveídos por Beanstalk, alrededor de los cuales giran el resto de los componentes. Más alejado se encuentra el módulo administrador, el cual ayuda a visualizar cierta información y realizar ciertas acciones directamente con Beanstalk, sin interferir directamente con el resto de los componentes.

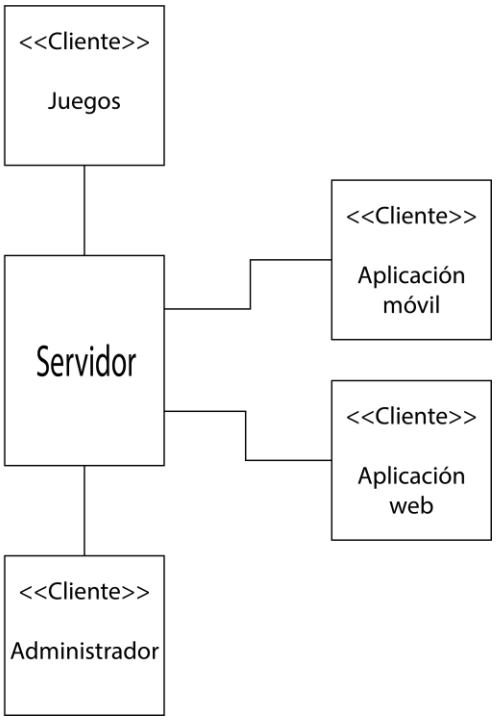

Figura 22. Estructura general de los diferentes componentes del proyecto

Cabe anotar que la aplicación móvil de iOS posee los juegos embebidos dentro de la aplicación debido a la política de seguridad de Apple, expuesto en el punto 3.1.3(a) de su guía de revisión[21]. De este modo, el cliente de juegos existe dentro del cliente de la aplicación en la versión de la tienda de Apple.

[21] https://developer.apple.com/app-store/review/guidelines/#safety

Se utilizará un ejemplo de uso para explicar cómo los componentes operan entre ellos. Poniéndose en la piel de un jugador que desea entrar en una partida de la belote, primero de todo debería entrar en uno de los clientes, ya sea una de las diferentes versiones móviles o la versión web ofrecida. Una vez dentro del cliente, el jugador entrará al *lobby* del juego, donde podrá empezar el proceso de emparejado con otros jugadores con quienes jugar la partida, este proceso genera dentro de servidor ejecutándose en las máquinas virtuales de EC2 una sala donde se llevará a cabo la partida. A esta se irán uniendo el resto de los jugadores hasta rellenar el resto de los huecos restantes, lo que iniciará la partida de la belote. Es entonces cuando el servidor comienza la ejecución de la lógica que controla el juego. En el siguiente punto se explicará la ejecución de este punto en adelante.

4.3.1 Arquitectura de los juegos dentro de la aplicación *Playjoy*

Focalizando en los juegos, se pasará a explicar con más detalle los diferentes componentes utilizados para la correcta ejecución de los juegos y como estos trabajan internamente.

Arquitectura de los clientes de juegos

Profundizando en la estructura de la parte front-end de los juegos, se muestra en la Figura 23 las clases principales de este módulo. Como controlador principal, está la clase "BeloteGameScene", cuyas funciones principales son la comunicación con el servidor y orquestar el resto de las clases inferiores para representar correctamente el estado del juego. Esta clase posee una conexión con el servidor mediante la cual recibe y envía mensajes, y a medida que recibe estos, llama a los métodos del resto de clases para modificar el estado de la partida.

Después, se encuentra la clase "BeloteTableManager", la cual posee métodos para modificar directamente los gráficos relacionados con las manos y anuncios de los otros jugadores y utilizando la clase "BeloteTableElements" para controlar las cartas en la mesa y elementos como las cartas de la baza anterior o el gráfico de ayuda con el orden de las cartas. Esta última clase contiene la lógica necesaria para modificar estos elementos.

Por último, para controlar los gráficos de la mano del jugador inferior, se cuenta con la clase "BeloteHand", la cual posee métodos muy similares a los encontrados en la clase "BeloteTableManager" para controlar las manos de los oponentes. La principal diferencia entre estas dos clases reside en la adición del sistema de *inputs* del jugador, permitiendo interactuar con las cartas de su mano y con los diálogos del juego, enviando a través de la clase principal los mensajes al servidor de las acciones. El estado de este jugador, al igual que se hará con los bots, se guardará en la clase "BelotePlayer", guardando una lista con las cartas, puntos y pareja del jugador principalmente.

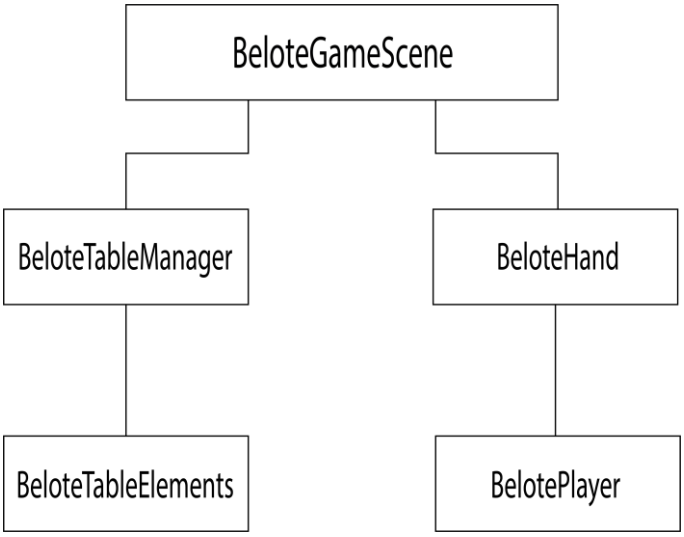

Figura 23. Estructura del front-end de los juegos

Arquitectura del servidor

Pasando ahora a la estructura del back-end de los juegos (Figura 24), ejecutada en el servidor, se encuentra una estructura más simple que la de los clientes. Primero de todo, similar al caso de los clientes, la clase "BeloteGameServerLogic" es la encargada de comunicarse con el cliente, recibiendo sus mensajes y enviándole los mensajes necesarios para comunicar el avance de las partidas.

La lógica donde se calcula el avance de la partida reside en la clase "BeloteManager", la cual lleva el progreso de las rondas y fases de juego, además de decidir los ganadores de las bazas y rondas. Esta clase además guarda el estado de la partida en todo momento, permitiendo adquirirlo cuando sea necesario desde el cliente, el estado de los cuatro jugadores es guardado en las instancias de "BelotePlayer".

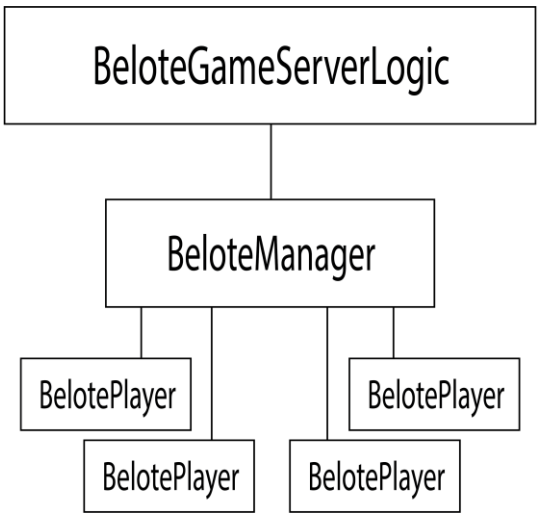

Figura 24. Estructura del back-end de los juegos

Arquitectura de los bots

Al igual que en el caso de la aplicación y los propios juegos, los bots presentan una división propia de clases, aunque mucho más simple que en el resto de los aspectos del proyecto. Como resumen, los bots emulan clientes del juego ejecutándose en el servidor intercambiando mensajes con él y reaccionando a ellos, al igual que hacen los clientes móviles. También tendrán una instancia de la clase "BelotePlayer" asociada, teniendo acceso a las mismas utilidades de jugador que cualquier otro cliente del juego.

La organización de clases de los bots es mucho más simple que el resto, constituyéndose de dos clases principales: "BeloteClient" y "BelotePlayer". La primera clase es la más importante de las dos, siendo la clase que recibe los mensajes por parte del servidor, realiza la toma de decisiones por parte de la inteligencia artificial y envía los mensajes de acciones al servidor. La segunda clase, únicamente ofrecerá las utilidades necesarias para poder simular el estado de un jugador real, como las cartas en posesión o la pareja del equipo.

4.4 Comunicación cliente-servidor

Como se ha visto, los juegos están separados en dos elementos principales, un cliente o front-end en el cual se ejecuta la parte gráfica de estos, y un servidor o back-end encargado de llevar a cabo toda la lógica de la partida, llevar su progreso y mantener un estado de esta. Estos dos elementos se ejecutan en diferentes dispositivos, requiriendo de un sistema de comunicación entre ellos, donde el servidor pueda comunicar a los clientes el progreso de la partida y enviar su estado, y los clientes puedan comunicar las acciones que han llevado a cabo y el servidor pueda avanzar la partida como se requiera. Para solucionar este problema, la empresa desarrolló un sistema de comunicación mediante mensajes, donde se crearán un número definido de mensajes específicos, a los cuales podrán reaccionar ambas partes del sistema.

Estos mensajes se definen en dos clases distintas (Figura 25) según su origen, estas son "MessagesFromServer" (1) y "MessagesFromClient" (2) conteniendo los mensajes procedentes del servidor y cliente respectivamente. Cada mensaje dentro de la lista de los mensajes definidos constituye una llamada *data class*[22] de Kotlin, las cuales se encargan únicamente de almacenar información. Esta información se trata de la necesaria por la otra parte para reaccionar según sea debido, ya sea información sobre la acción tomada por el cliente recibida por el servidor, o la información de la reacción a dicha acción recibida por el cliente.

[22] https://kotlinlang.org/docs/data-classes.html

```
@Serializable
data class MyTrumpTurn(val firstRound: Boolean, val tutorial: Boolean = false)
@Serializable
data class MyNormalTurn(val currentWinner: HandWinner?, val call: BeloteCall? = null)
@Serializable
data class OthersTurn(val clientId: String, val call: BeloteCall? = null)
```

1

```
@Serializable
data class PlaceCardRequest(val card: BeloteCard)

@Serializable
data class TrumpSuitSelectionRequest(val playsTrump: Boolean, val trumpSuit: FrenchSuit? = null)
```

2

Figura 25. Muestra de mensajes de comunicación cliente-servidor

Para enviar la información en estos mensajes es necesario formatearlos en formato JSON[23], ya que este documento se puede enviar como parámetro en las llamadas API utilizadas para comunicar cliente con servidor. Este formateo se realiza mediante el sistema de serialización de clases de datos genérico de Kotlin[24]. Una vez formateada la información a enviar junto al mensaje, la parte que desea enviar este realizará una llamada API con el tipo de mensaje y su contenido como parámetros. Una vez recibida la información, la parte receptora ejecutará el método asociado a dicho mensaje, (en la Figura 26 se muestra un ejemplo de uno de estos métodos). Estos ejecutarán el código necesario para poder reaccionar correctamente a dicha información, llevando a cabo las tareas que sean necesarias.

En el método ofrecido como ejemplo, se observa el método que realizará las acciones necesarias cuando el cliente recibe el mensaje indicando su turno normal. Durante este turno, se ejecutará un sonido de turno, comenzará la cuenta atrás del turno del jugador y las cartas de este se desbloquearán para poder lanzar la que desee.

[23] https://www.json.org/json-en.html
[24] https://kotlinlang.org/docs/serialization.html

```
@onMessage("MyNormalTurn")
private fun onMyNormalTurn(msg: MyNormalTurn) {
    currentWinner = if(msg.currentWinner != null) msg.currentWinner.player else ""
    activePlayer = myPlayer.id
    hand.isMyTurn = true
    BeloteSounds.BELOTE_TURN_SOUND.play()
    launch {
        if(msg.call != null) {
            wait(0.5.seconds)
            callsManager.showCallText(myPlayer.id, msg.call)
        }
        hand.blockCards(true)
        hand.filterValidCards(tableManager.getTableCards(), currentWinner)
        updateTurnInfo(myPlayer.id)
        turnTimeout.start()
        switchTurn()
    }

    if(AUTO_MODE) turnTimeoutAction()
}
```

Figura 26. Método "onMyNormalTurn", que lleva a cabo las acciones al comenzar el turno del jugador

5. Desarrollo del juego

En este capítulo se explorarán los elementos que han resultado más complejos a la hora de desarrollar e implementar la belote, además de los efectos que estos tuvieron en las fechas para cada versión marcadas en un principio, resultando en atrasos inesperados. También se expondrán en parte las pruebas unitarias llevadas a cabo para conseguir una lógica de juego limpia y fiable.

5.1 Implementación de los gráficos propuestos

Comenzando por la implementación de los gráficos propuestos, estos resultaron ser una de las partes más complejas a llevar cabo durante el desarrollo. En parte esta complejidad se debe al desconocimiento de las tecnologías utilizadas y a la necesidad de nuevos sistemas y gráficos específicos para la belote. Se recorrerán los motivos por los que ciertos gráficos necesitaron cambios respecto a los gráficos propuestos y sus modificaciones específicas. Por último, se mostrará el resultado final de estos cambios con ejemplos ejecutados ya en la aplicación a tiempo real.

5.1.1 Dificultades al seguir los diseños iniciales

El flujo para la creación de gráficos a la hora de desarrollar un nuevo juego pasa por un par herramientas según su propósito hasta ser implementados finalmente en el juego. Primero de todo, el equipo de diseñadores se encarga de realizar los prototipos del juego específico en Adobe Illustrator[25] (mostrados anteriormente en la página 35), estos ayudan al resto del equipo a hacerse una idea de cómo se verá el juego. Luego, el mismo equipo de diseñadores comienza a implementarlos en Adobe Animate, a cuyo proyecto tienen acceso tanto diseñadores como programadores. Esto se debe a que es el último paso antes de exportar los gráficos para ser instanciados en el juego a través de la librería EaselJS.

Es debido a este uso de diferentes herramientas que a la hora de seguir los diseños iniciales pueden aparecer complejidades a la hora de adaptarlos o incluso surgen modificaciones una vez vistos en las pantallas móviles o en la web. El caso de la belote no fue excepción, donde se necesitaron hacer varios a justes a tamaños al igual que cambios a ciertos gráficos y mejoras. Los cambios residen en la manera de mostrar los anuncios, mientras que las mejoras residen sobre todo retoques a los tamaños de los gráficos y ciertas medidas para hacer el juego más claro para el jugador.

Estos cambios y mejoras no solo nacen de las mentes de los programadores a ver las posibles dificultades la hora de implementar los gráficos propuestos, sino también de las

[25] https://www.adobe.com/es/products/illustrator.html

sesiones de pruebas de validación, donde los clientes juegan partidas y ofrecen opiniones respecto a los gráficos ya implementados.

5.1.2 Modificaciones realizadas a los diseños iniciales

En este apartado se explorarán los cambios y mejoras realizadas respecto a los gráficos originales. Para los cambios, se mostrará la versión implementada comparada con la versión original, marcando los cambios realizados y los motivos detrás de estos. Se comenzará con los cambios a los gráficos de los anuncios, continuando con las mejoras a los gráficos originales y finalizando con las adiciones llevadas a cabo.

Modificaciones a los gráficos de los anuncios

Los anuncios durante las partidas de la belote son muy relevantes, pudiendo decantar la ronda o incluso la partida entera para uno de los dos equipos. Es por este motivo que se decidieron añadir unos gráficos que resaltaran estos, asegurando que todos los jugadores conocieran los puntos de anuncio en juego. En la Figura 27 se observan comparados los gráficos de los anuncios del jugador inferior. A la izquierda están los gráficos originales (1) y a la derecha la implementación final (2). Estos gráficos son utilizados durante la fase donde se resuelven los ganadores de los anuncios, mostrando únicamente las cartas de estos.

Como se puede observar, el cambio principal se observa en la falta del título y la puntuación del anuncio, teniendo cada uno su propio motivo para ser eliminado. Primero, el titulo se decidió innecesario ya que este se muestra a todos los jugadores a la hora de declarar en anuncio, apareciendo un mensaje flotante con el texto desde la mano del jugador que declara su combinación. Luego, la puntuación se dejó fuera del diseño final ya que en un principio daba a entender que la pareja que resultaba ser ganadora de los anuncios sumaba los puntos al momento, cuando no es así. Los puntos de los anuncios solo se sumarán si dicha pareja resulta también ganadora de la ronda.

Con estos cambios, el resultado final son unos gráficos más simples, eliminando la información redundante y evitando desinformar a los jugadores.

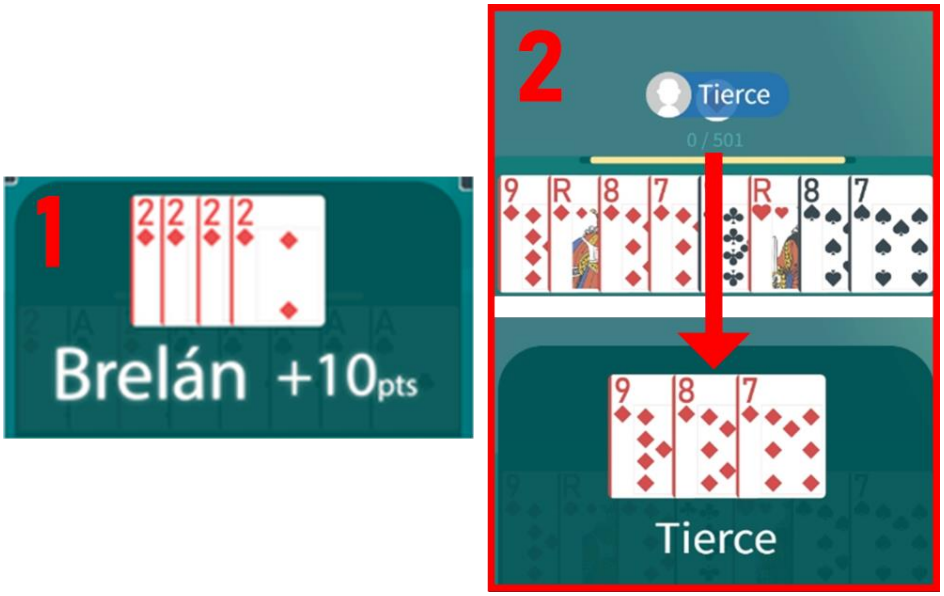

Figura 27. Comparativa entre las versiones de los anuncios

El anuncio belote también recibió un cambio, esta vez más notable que el caso anterior de los anuncios generales. Es cierto que esta combinación de rey y reina de triunfo es muy importante durante la partida, pues son los únicos puntos inamovibles que una pareja puede marcar durante la partida. Es por este motivo que se incorporó en el prototipo con una animación que ocuparía la pantalla entera marcando la resolución de este anuncio (Figura 28, imagen 1). A pesar de esta desmedida animación, la combinación no ofrece una gran cantidad de puntos, por lo que se decidió reducir a dos simples mensajes flotantes indicando el estado del anuncio (2).

Este cambio ofreció una mayor fluidez en las partidas, ya que estas no se verían frecuentemente interrumpidas por una gran animación, además de ofrecer una visualización más acorde a la cantidad de puntos que asegura este anuncio.

Figura 28. Comparativa entre las versiones del anuncio belote

Mejoras a los gráficos originales

Pasando a las mejoras generales respecto a los gráficos originales, estas se componen de cambios menores, los cuales consiguen un efecto mayor al verlos todos unidos dentro de la implementación final. Estos ajustes afectan a casi todos los gráficos del juego, pero en este apartado solo se explorarán los más relevantes y que cuentan con un mayor peso. Se explorarán tres cambios de facetas diferentes del juego, todos con el objetivo de incrementar la comprensión de los gráficos y hacerlos más entendibles.

Primero, durante la fase de selección de triunfo, a la hora de hacer sesiones de pruebas, los jugadores menos relacionados con el juego y sus reglas dudaron del objetivo de la carta posicionada en el centro de la pantalla. Esta carta resulta ser la carta que indica el palo de triunfo a elegir al igual que una de las cartas que serán entregadas si es aceptada, por lo que su malentendido debía ser evitado en todo momento. Para su solución, se añadió a esta carta un halo que la rodease, marcando que esta carta era especial y requería de la atención del jugador. Se puede observar el cambio en la Figura 29, viendo en la imagen 1 la versión original y en la imagen 2 la versión implementada.

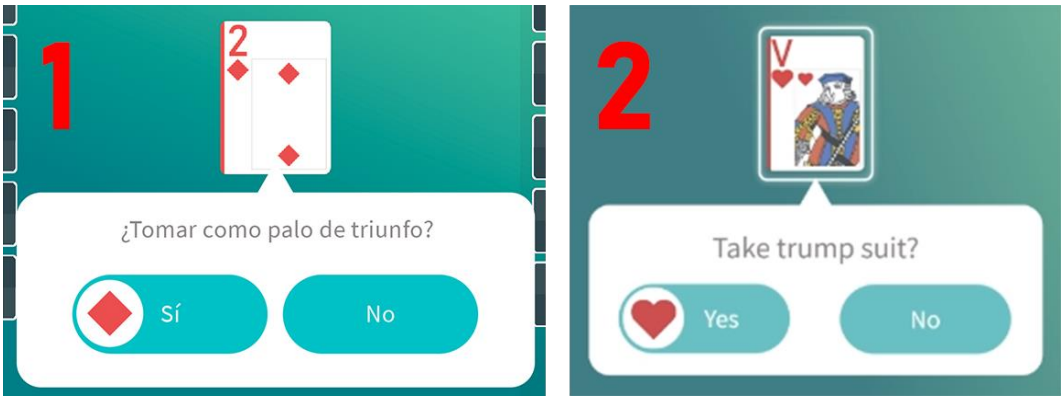

Figura 29. Comparativa entre las versiones del dialogo de selección de triunfo

Continuando con estos cambios, el siguientes surgieron a la hora de implementarlos por primera vez y ver el juego desde su configuración móvil, se trata de los gráficos encargados demostrar las cartas jugadas en la baza anterior. Al comienzo, estas utilizaban las mismas texturas utilizadas en las tarjetas de la mano, las cuales al reducirse tanto en tamaño y mostrarse en una pantalla de teléfono, resultaba muy complicado de entender. Entonces, se crearon nuevas texturas para estas cartas las cuales simplificaban la información, mostrando únicamente el número de la carta y el palo de esta. Se puede observar en la Figura 30, teniendo el diseño antiguo en la imagen 1 y el nuevo diseño en la imagen 2, ambos vistos en una pantalla de juego completa.

Figura 30. Comparativa de los gráficos de las cartas de la baza anterior

Finalmente, de nuevo, a la hora realizar sesiones de pruebas con otros jugadores no relacionados con el desarrollo del juego, todos ellos confundieron en algún momento los gráficos del palo de tréboles con el palo de picas (Figura 31, imagen 1). Por este motivo se buscó refinar el diseño del palo de tréboles (2), dotándole de una textura más lejana al diseño del palo de picas. En la figura se puede observar ambos palos comparados, picas a la izquierda y tréboles a la derecha (en el diseño inicial, se cometió el error de colorear las picas de rojo, modificándose durante la creación de los gráficos en Adobe Animate).

Figura 31. Comparativa de los palos de picas y tréboles

Gráficos añadidos no tenidos en cuenta

Finalmente, durante las sesiones de validación, los jugadores no familiarizados con la belote y sus reglas se encontraron perdidos a la hora de lanzar una carta, ya que los valores de las cartas y su orden difiere de los juegos tradicionales españoles. Mientras que en la mayoría de los juegos el orden de las cartas sería As-Rey-Reina-Jinete-10-9-8-

7, la belote cuenta con dos órdenes diferentes, As-10-Rey-Reina-Valet-9-8-7 si no son de triunfo y Valet-9-As-10-Rey-Reina-8-7 si son de triunfo. Para solucionar este problema, el equipo se fijó en otras implementaciones, donde se implementa un gráfico de ayuda mostrando el orden de las cartas. En este caso, se ha incorporado un gráfico de ayuda mostrando el orden de las cartas sin y con triunfo, permitiendo además a los jugadores más experimentados cerrar la ayuda en caso de que no la vean necesaria. En la Figura 32 se muestra el gráfico abierto (1) y cerrado (2). Este gráfico aparece en la esquina superior izquierda del juego.

Figura 32. Gráfico de ayuda añadido al juego

5.2 Reconexión a la partida

Como una de las funcionalidades a las que más que más importancia se le ha dado en esta memoria como punto vital, la reconexión de la partida resultó además ser una de las tareas más complejas durante el desarrollo del juego. Esta complejidad reside en el mensaje "BeloteState" el cual como su nombre indica envía desde el servidor el estado actual de la partida a los jugadores que han abandonado y están volviendo a entrar a esta.

Este estado necesita contener todas las variables que resultan vitales para poder pasar de un estado limpio de la partida al estado completo actual de esta. Esto quiere decir que el mensaje ha de contener desde cosas básicas como cuál es tu pareja en la partida, o el tiempo de turno del juego, hasta cosas mucho más específicas como las cartas jugadas en la ronda anterior. Además de esta dificultad, esta transmisión se ha de realizar idealmente con el menor número de variables posibles, simplificando la recopilación de datos necesarios, reduciendo de esta manera la posibilidad de generar errores a la hora de la ejecución.

5.2.1 Dificultad para conseguir y enviar un estado completo

Como se ha indiciado antes, una de las mayores dificultades de esta tarea resultó ser conseguir el estado completo del juego, sin echar en falta ninguna variable que impidiese mostrar el estado actual al completo de la partida al jugador recién conectado. Este

requerimiento de ofrecer un estado desde cero genera una necesidad de guardar con numerosas variables, resultando en un código de la parte de servidor más complejo.

Luego, la mayoría de estas variables han de ser almacenadas en el mensaje a enviar, lo cual resulta en un mensaje mucho más grande que el resto, con numerosos parámetros entro de este. De esta forma el mensaje resulta muy complicado de comprender para el equipo de programadores, siendo solo entendido completamente por el programador inicial. Este tamaño y complejidad hace que a la hora de solucionar errores sea complicado señalar al culpable y además hace recaer en el programador inicial todo pensamiento necesario para solucionar dicho error.

A pesar de estas dificultades, la solución una vez implementada ofrece una experiencia hacia el usuario fluida y veloz, donde automáticamente, una vez el jugador vuelve a la partida ya entrará con los gráficos mostrando el estado guardado en el servidor. De esta forma se consigue una experiencia de juego móvil ideal, donde los jugadores pueden salir de la aplicación sin preocuparse de la partida, volviendo a ella y reentrando en la partida en curso en su estado actual. Además, junto a la posesión de los bots, el resto de los jugadores no detectarán el abandono al igual que el jugador que salió de la partida puede estar seguro de que sus cartas están en buenas manos.

5.3 Dificultades al seguir las fechas propuestas

En el capítulo tres, se presentaron unas nuevas fases de desarrollo del juego a seguir junto a unas fechas de finalización ideales. El objetivo de esto es poder programar mejor la estrategia a seguir y llevar un mejor control sobre el desarrollo y sus posibles retrasos. En el caso del desarrollo de la belote se planeó un comienzo del desarrollo en enero de 2021, con una finalización del proyecto completo a principios de agosto 2021. Esta línea temporal supone una aproximación temporal mayor a la media de otros juegos, siendo el motivo de esto la jornada reducida de trabajo y los desconocimientos de la tecnología. A pesar de este margen dado, el desarrollo se vio retrasado en gran medida, resultando en un lanzamiento del juego más tardío de lo esperado.

Se pasará ahora a revisar las líneas temporales de cada versión al igual que ofrecer un diagrama de Gantt con las fechas finales.

Pre-Alfa

Esta primera versión únicamente contiene el desarrollo de la lógica del juego dentro del servidor, parte más sencilla de llevar a cabo ya que el no resultó tan desconocida a la hora de desarrollar. Además, resulta ser la fase más sencilla sobre la cual realizar pruebas, ya que admite tests unitarios de cada uno de sus métodos, pudiendo forzar casos poco comunes y estresar así la lógica.

A pesar de esto, sufrió un retraso igualmente, siendo finalizada una semana después de lo planeado, el nueve de febrero en vez del día uno.

Alfa

Pasando a la versión alfa, esta contiene la primera implementación de los gráficos, contando únicamente con los necesarios y vitales para poder realizar pruebas sobre la comunicación cliente-servidor y comprobar que los gráficos iniciales resultan efectivos.

Sobre esta versión se realizaron pruebas entre los equipos de programación para detectar posibles bugs de lógica y el equipo de diseño para comprobar el correcto funcionamiento de gráficos. Donde surgió la necesidad de modificar algunos de los gráficos mencionados anteriormente, específicamente los cambios a los gráficos de las cartas jugadas en la baza anterior y el cambio realizado para diferenciar mejor los palos de picas y tréboles.

Puesto que durante la sesión de pruebas realizada una semana antes de la fecha de vencimiento no salieron bugs mayores, este MVP pudo cerrarse cumpliendo con la línea temporal marcada, finalizando en uno de marzo.

Beta

Esta fase se programó como las más duradera y posiblemente la que más contenido contendría, pues se buscaba tener una versión ya con todos los gráficos incorporados en su versión final y lo más importante, entregar una versión que contase con un sistema de reconexión funcional y una primera versión de los bots. Ya se sabe que estas dos últimas han resultado dos de los mayores problemas del desarrollo.

Durante el desarrollo de esta versión se realizaron sesiones de pruebas semanales, con diferentes partes del equipo, ofreciendo una visión más amplia y cercana a los jugadores que en un futuro jugarán a la belote. El objetivo de estas sesiones además era confirmar una reconexión fiable y una inteligencia artificial que cumpliese con los estándares del resto de juegos y los objetivos marcados para la belote. Además, de estas sesiones más amplias y cercanas a la realidad nació la necesidad de llevar a cabo el resto de los cambios a lo gráficos antes mencionados, al igual que la adición del gráfico de ayuda.

Debido a estas dificultades y cambios a gráficos, esta versión a pesar de su ampliada predicción sufrió un retraso de dos meses, finalizando la tarea el cuatro de agosto en vez de la fecha prevista del uno de junio.

Versión 1.0 o "Gold"

Con los retrasos generados por la fase anterior, esta versión 1.0 comenzó dos meses más tarde de lo planeado, siendo esta versión la primera versión que lanzada en tienda. Por este motivo, esta fase requisó la solución de los errores más relevantes surgidos durante las pruebas anteriores. Además, para esta versión se aumentaron el número de partidas simuladas con diferentes versiones de la IA, con el objetivo observar patrones incorrectos o poco óptimos que pudiesen afectar a la eficacia de esta. El funcionamiento de estas simulaciones será visto con mayor profundidad en la página 81.

Una vez se finalizó de refinar los elementos necesarios para la incorporación del juego en el entorno de producción, el equipo de desarrolladores encargados de la aplicación se encargó de implementar el juego dentro de esta. Esta primera versión del juego, a pesar de ser testeada internamente en profundidad y su IA fue revisada hasta conseguir un comportamiento aceptable, el juego se lanzó en el llamado modo beta dentro de la aplicación. De esta manera, el juego no aceptaría apuestas de ningún tipo, siendo un juego gratis de jugar, ya que sería la primera vez que se explotaba de cara al público y por tanto todavía podía dar lugar a errores no conocidos.

Continuando con el atraso de la versión anterior, esta versión 1.0 se finalizó el 13 de septiembre, resultando en una demora de otros dos meses respecto a la fecha inicial de la segunda semana de junio.

Versión 1.1

Por último, ya con la belote en explotación, se continuó el desarrollo de los dos últimos elementos del juego, siendo el tutorial y los retos. Estos contribuyeron a cumplir con el estándar del resto de juegos dentro de la aplicación, permitiendo a la belote a salir de su modo beta y poder comenzar a participar en la economía interna de la aplicación.

El desarrollo de estos dos elementos no supuso ningún atraso a la entrega de esta versión, puesto que se realizaron basándose en el resto de los juegos, utilizando estructuras y compartiendo parte del código para su realización. A pesar de esto, al lanzar el juego en explotación surgieron ciertos problemas que impedían jugar partidas a ciertos usuarios. Estos problemas por suerte no fueron numerosos, sin superar la cinquena, y estando todos ellos relacionados con la reconexión del juego.

Una vez solucionados estos errores y probados el tutorial y los restos, se procedió a actualizar las versiones de los clientes, añadiendo la habilidad de apostar dentro del juego, marcando así la salida de su modo beta. Esta fase contó con un retraso de dos semanas respecto al tiempo de desarrollo previsto, lo cual junto a los dos meses de retraso acarreados resultó en una salida de la versión 1.1 el ocho de noviembre, con un total de tres meses más tarde de lo esperado.

5.3.1 Diagrama de Gantt real

En la Figura 33 se representa ahora un nuevo diagrama de Gantt mostrando los retrasos explicados en los puntos anteriores, pudiéndose comparar con la Figura 19. El principal impacto que presentó este retraso se encuentra en el aplazamiento del desarrollo futuro de la coinché hasta noviembre, donde se dio por finalizada la fase de desarrollo de la belote.

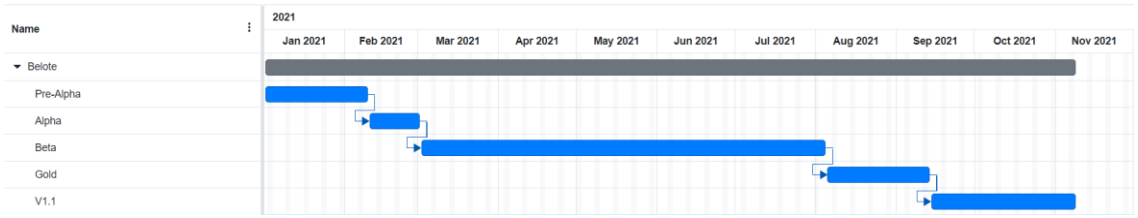

Figura 33. Diagrama de Gantt mostrando los tiempos reales de desarrollo

5.4 Pruebas unitarias y resultados de las sesiones de pruebas

En este punto se explicarán las pruebas unitarias llevabas a cabo para la depuración de los métodos más relevantes encontrados en la lógica. Puesto que en el caso del proyecto en concreto las pruebas unitarias solo están disponibles para la lógica, a la hora de probar el front-end del juego se realizaron diferentes pruebas entre el equipo de la empresa. En estas pruebas se iniciaban diferentes partidas entre nosotros y a solas para probar el correcto funcionamiento de elementos relevantes como la reconexión o la inteligencia artificial.

5.4.1 Pruebas unitarias llevadas a cabo

Las pruebas unitarias son pruebas específicas realizadas para probar partes específicas de un programa. Luego, el resultado de esta ejecución se compara con el resultado correcto esperado, justificando el correcto funcionamiento del programa probado en caso de coincidir (18). Aplicado el uso al proyecto en cuestión, estas pruebas unitarias se utilizaron para comprobar el correcto funcionamiento de los métodos más relevantes y los métodos pilares de la lógica llevada por el servidor. Estas pruebas, por lo tanto, se ejecutan desde el back-end, en una clase especifica llamada "BeloteTests", donde se alojan estas pruebas unitarias. Se repasará un método en específico para explicar el funcionamiento general de estas pruebas.

Se presenta la prueba unitaria llamada "testSpecificCardMelds" (Figura 34), la cual se encarga de probar el código especifico relacionado con el cálculo de las combinaciones en las manos de los jugadores. Recorriendo brevemente el código, este comienza declarando la variable "testMelds" la cual guardará el resultado del cálculo de los anuncios existentes. Luego se declara e inicializa la lista "testCards", la cual contiene una lista de cartas simulando una mano de un jugador, esta variable permite representar cualquier caso de cartas en una mano que se quiera probar. Una vez iniciadas las dos variables, se añade a la lista "testMelds" los anuncios devueltos por el método "checkForCalls" realizado sobre la lista "testCards". Por último, se escribe en pantalla los anuncios conseguidos y en la línea se encuentra la comprobación de los anuncios conseguidos con el resultado esperado. Si este coincide con el conseguido de la ejecución la prueba será satisfactoria.

Como notas al código de la Figura 34, sobre la declaración del método se encuentra la línea "@Test", la cual sirve para indicar al IDE que dicho método se trata de un test

unitario y debe poder ejecutarse individualmente. Esto nos permite ejecutarlo sin tener que compilar el código entero, ya que se busca únicamente probar el código que aparece dentro del método. Posteriormente, la inicialización de la variable "testCards" se hace de manera implícita, creando la lista a la vez que se instancia la variable. Esta lista está compuesta de objetos llamados "BeloteCard", los cuales como su nombre indica contienen los datos de las cartas de la belote. Estos objetos se están instanciando mediante el método infijo "of", este ejecuta el código relacionado con el constructor de "BeloteCard", permitiendo realizar instanciaciones mucho más legibles. Por último, el método "checkForCalls" posee un parámetro de entrada, el cual ha de ser el palo del juego que marque el triunfo. Este se utilizará en el método para diferenciar anuncios en caso de repetirse, priorizando los de triunfo, en el caso de la prueba se utilizó el palo de diamantes como ejemplo ya que no influye en el resultado final.

```
@Test
fun testSpecificCardMelds() {
    val testMelds: MutableList<BeloteCallData> = mutableListOf()
    val testCards = mutableListOf(
            TEN of HEARTS, QUEEN of HEARTS, ACE of HEARTS,
            TEN of PIKES, JACK of PIKES, QUEEN of PIKES,
            KING of PIKES, ACE of PIKES)

    testMelds.add(testCards.checkForCalls(DIAMONDS))

    println(testMelds)

    assertEquals(BeloteCall.QUINT, testMelds.first().call)
}
```

Figura 34. Test unitario "testSpecificCardMelds"

Este tipo de pruebas se utilizó principalmente durante la fase pre-alfa, ya que únicamente permiten probar los métodos ejecutados en los servidores de la aplicación. Durante este tiempo fueron cruciales para poder obtener la lógica sólida y probada que se buscaba, permitiendo continuar con el desarrollo del juego sin problemas mayores en esta. Esto no quiere decir que no fuesen de gran utilidad durante el resto de las sesiones de prueba ya que las pruebas unitarias, como se ha visto en el caso ejemplo estudiado, permiten forzar cualquier caso específico que se quiera depurar.

5.4.2 Sesiones de pruebas con el equipo llevadas a cabo

Puesto que las pruebas antes mencionadas no son de ayuda a la hora de probar los gráficos del juego, se decidió organizar el máximo número de sesiones de prueba posibles con diferentes partes del equipo. Estas sesiones funcionarían como sesiones de validación del producto, estando presentes las diferentes partes relevantes en el desarrollo y explotación del producto. Estas partes eran simuladas por los diferentes

miembros del equipo, ya que las diferentes facetas encontradas en él lo permiten. Los objetivos se estas pruebas de validación varían según la fase en la que se encontrase el desarrollo. Estos objetivos ya han sido revisados en la página 41 junto con el público objetivo.

6. IA desarrollada

6.1 Uso de *bots* dentro de los juegos de *Playjoy*

En este punto se explicará, analizará y desarrollará una inteligencia artificial para los bots que son utilizados dentro de los juegos. El término bot es una referencia al término robot, en este caso los robots son jugadores dentro de las partidas controlados por una inteligencia artificial ejecutada en el servidor de la aplicación. Estos bots son utilizados principalmente en dos casos dentro de los juegos de la aplicación: poblar partidas con menos de 4 jugadores y poseer a los jugadores que abandonan la partida.

El primer caso surgió de la necesidad de reducir los tiempos de búsqueda de partidas en línea de los juegos. Ya que las partidas requieren de 4 jugadores, la búsqueda de partidas se puede alargar hasta durar varios minutos, lo cual es muy perjudicial para la experiencia de usuario. Puesto que uno de los principales objetivos de la aplicación es ofrecer una experiencia de juego rápida, una vez la búsqueda de partidas sobrepasa cierto tiempo, el servidor asignará *bots* a los asientos que faltan por rellenar dentro de las partidas. Estos jugadores cuentan con perfiles creados por la empresa, para no dar a conocer su naturaleza como bots y ofrecer una experiencia de juego real en línea.

El segundo caso es innato a la experiencia multijugador en línea, y es acentuada por la naturaleza móvil de la aplicación. Cuando uno de los jugadores conectados abandona la partida, ya sea decisión propia o por fallo de su dispositivo, un *bot* poseerá a su jugador dentro de la partida y pasará a jugar por esta persona mientras está ausente. En estos casos la IA adquirirá el estado de la partida en el que el jugador abandonó y pasará a tomar decisiones según este. Desde este momento pueden ocurrir dos cosas, la persona que abandonó la partida puede reconectarse, caso en el que el bot dejará de poseerle y el jugador pasará a jugar de nuevo; o pasado cierto tiempo, el servidor asumirá el abandono de la persona y esta no podrá volver a reconectarse, pasando a ser el *bot* quien controle al jugador hasta el final de la partida.

Cabe destacar que en el caso de coexistir varios *bots* dentro de la partida, todos estos utilizarán la misma IA con diferentes estados. Esto quiere decir que el proceso de toma de decisiones es el mismo, y ayuda a simplificar el proceso de desarrollo y refinamiento de estas.

6.2 Estado actual de las inteligencias artificiales de *Playjoy*

Todos los juegos cuentan con una inteligencia artificial, la cual es desarrollada junto al juego principal y refinada una vez se obtienen datos del funcionamiento de esta. Dentro del catálogo de 9 juegos existentes en la aplicación, se agrupan sus inteligencias artificiales en dos grupos, inteligencias básicas e inteligencias reactivas.

Las primeras inteligencias son las más simples, las cuales no cuentan con ningún proceso de razonamiento según el estado actual de las partidas. Estas actuarán con la primera acción disponible en su mano, sin pensar en cual sería la mejor acción para ganar la partida. Este tipo de inteligencia es únicamente utilizada por el dominó actualmente, ya que se trata del juego con menos toma de decisiones relevantes en la lista.

El segundo tipo de inteligencias es utilizado por el resto de los juegos desarrollados. Este sí que tiene en cuenta el estado actual de la partida, pero no realiza una valoración real de la situación, si no que actúa especificando los posibles estados y cómo debe actuar frente a ellos.

6.2.1 Crítica al estado actual de las inteligencias artificiales

Empezando por las inteligencias básicas, estas resultan claramente inferiores en cuanto a la toma de decisiones frente a sus compañeras las inteligencias reactivas. El hecho de no tener ningún tipo de razonamiento del estado actual de las partidas las hace inviables para la gran mayoría de juegos. Ya que normalmente se dispone de más de una acción disponible, ese razonamiento es necesario a la hora de tomar la mejor decisión posible frente a las acciones disponibles y favorecer a la victoria de la partida. Esta necesidad para la toma de decisiones correcta se ve agravada por la posibilidad de que esta inteligencia esté jugando con un jugador real, impactando sus acciones a la partida de este jugador.

En cuanto a las inteligencias reactivas, a pesar de ser muy eficaces frente a la gran mayoría de situaciones, resultan ser mucho más complejas en su implementación. La necesidad de especificar el mayor número de posibles situaciones frente a las que reaccionar hace que estas requieran frecuentes cambios a lógica una vez finalizado su desarrollo. Además, generan un código más complejo de entender por gente ajena a su desarrollo, haciendo necesario más comentarios para cada estado especificado.

6.2.2 Solución propuesta

Habiendo visto los puntos negativos de las inteligencias artificiales actuales, se propuso una nueva estrategia intentando solucionar los fallos que vio más relevantes. Basándose en inteligencias artificiales A*, se propone el desarrollo de una IA que haga uso de funciones heurísticas para evaluar el estado de la partida en el que se encuentra y tomar una decisión según este. Para reducir el peso de la ejecución de esta IA en el servidor, no realizará ninguna búsqueda de estados futuros, solo tendrá en cuenta el estado actual de la partida.

La heurística utilizada contará con un cierto número de parámetros definidos tras un estudio de las características más relevantes del estado de la partida, al igual que cada uno de ellos será ponderado mediante una tabla de pesos por parámetro. Por lo tanto, cada vez que la IA tenga que realizar una elección, aplicará un valor a cada parámetro según el estado de la partida y después lo ponderará según los pesos especificados. Los

valores de parámetros los mantendrá en un rango de [0, 1] para mantener consistencia entre ellos.

Debido a la naturalidad competitiva por parejas de la belote se ha buscado una iteración más avanzada sobre las inteligencias artificiales actuales. La cual ofrezca una mejor comprensión por parte del equipo de programadores y un mejor mantenimiento en caso de mejoras a realizar en un futuro. Además, con ligeros cambios, ofrecen la posibilidad de ser entrenadas mediante simulaciones de partidas donde los pesos de los parámetros se autoajustan para llegar a los más eficaces.

Esta solución, a pesar de utilizar términos como pesos u operaciones heurísticas para su descripción, comunes en las redes neuronales o inteligencias dinámicas con retrospectiva o valoración de casos futuros, no resulta ser ninguno de estos casos en un principio. Esta inteligencia propuesta es una solución estática y sin entrenamiento automático. La decisión de estos dos factores para la solución viene dada por el alto coste computacional y de desarrollo que convendría, ya que se busca encontrar la solución más ligera de ejecutar a gran escala sin bloquear los servidores. Además, el desarrollo viene atado a unas fechas y costes previstos, los cuales se verían altamente afectados en el caso de desarrollar la infraestructura necesaria para contar con estos elementos.

6.3 Análisis de las estrategias

Ahora que se conocen los tipos de inteligencias ya existentes y la propuesta para la nueva inteligencia a desarrollar, se hará un análisis y comparativa de estas. Para la comparativa se tendrá en cuenta la sencillez del diseño e implementación de la inteligencia, la capacidad de mejora del código, su legibilidad y la capacidad de adaptarse al estado actual de la partida.

	Básica	**Reactiva**	**Heurística**
Sencillez de diseño	Elevada	Baja	Baja
Sencillez de implementación	Baja	Media	Media
Capacidad de mejora	Elevada	Media	Elevada
Legibilidad del código	Elevada	Media	Elevada
Capacidad de adaptación	Baja	Media	Elevada

Tabla 3. Tabla comparativa de las diferentes estrategias de inteligencia artificial

Como se puede observar en la tabla superior, todas las inteligencias presentan puntos a favor al igual que en contra, por el uso actual de los dos tipos originales queda justificado dependiendo del juego. En el caso de la belote, las inteligencias básicas quedan descartadas, ya que la inteligencia ha de ser suficientemente competente como para poder ofrecer un desafío a los jugadores más avanzados y ayudar a ganar a sus jugadores

compañeros. Esto deja con dos posibles opciones, las inteligencias reactivas y heurísticas, y como se puede ver, la segunda siendo una evolución de la primera resulta más atractiva.

Tanto las inteligencias reactivas como las heurísticas comparten el mismo y (único) problema, y este es su complejidad de diseño. Son motivos diferentes los motivos por los que resultan ser complejas de diseñar, mientras que en el primer caso reside en el hecho de necesitar tener en cuenta el mayor número de situaciones y estrategias posibles, en las heurísticas reside en la complejidad del diseño de una estrategia competente e implementable con el menor número de parámetros posibles.

En el resto de los parámetros se ve que la inteligencia propuesta consigue mejorar a las inteligencias más comunes actuales. Gracias a su diseño con parámetros y pesos, una vez conseguida una estrategia aceptable solo hará falta retocar los pesos de los parámetros para mejorarla. Además, puesto que estos parámetros tienen un identificador marcado, el código final resulta altamente comprensible, pudiendo observar que parámetros son más importantes para los bots a la hora de tomar sus decisiones. Finalmente, si fuese necesario cambiar la estrategia por alguna diferente, se dispondrá de más capas de abstracción a través de los pesos y la asignación de los valores de parámetros mediante las cuales mejorar o adaptar antiguas estrategias a nuevas más eficaces.

6.4 Diseño de la nueva inteligencia artificial

Como se ha mencionado antes, el diseño de esta inteligencia artificial utilizando una heurística puede resultar ser la parte más complicada de todo su desarrollo. Esto se debe a la necesidad de conocer las reglas de la belote en profundidad y además poder decidir en una estrategia eficaz a la hora de ganar partidas.

6.4.1 Estrategia elegida para la inteligencia artificial

Existen numerosas formas de jugar a la belote, como ocurre en otros muchos juegos de cartas, algunas de ellas siendo más arriesgadas y otras siendo más seguras. Para el diseño de la estrategia a seguir en este caso se opta directamente por la opción más segura, ya que, en su mayoría de casos, los bots serán emparejados junto a jugadores reales y se debe favorecer al máximo las posibilidades de victoria. A su vez, con esta estrategia los jugadores podrán aprender como los bots juegan, dando mayores posibilidades de victoria contra ellos a los jugadores más experimentados. Esta estrategia resulta muy simple una vez examinada, para su correcto entendimiento es necesario conocer en profundidad las reglas del juego explicadas en el anexo 3.

La estrategia por seguir comienza durante la fase de selección de triunfo (Figura 35), donde se tendrán en cuenta las seis cartas disponibles (las cinco cartas iniciales y la carta que marca el palo de triunfo). Se decidirá si aceptar o no el triunfo según el número de cartas de dicho triunfo que tengamos, el número de anuncios del que se dispone con esas seis cartas y su nivel, si se dispone de un anuncio belote con ese palo y por último con cuantos puntos asegurados se cuenta. En caso de llegar a la segunda fase de selección de

triunfo, donde se puede elegir el palo libremente, se realizará esa misma valoración para cada uno de los cuatro palos. Durante ambas fases se valorará también los valores en el caso de pasar de turno y quedar con las 5 cartas originales y teniendo la posibilidad de jugar con un palo de triunfo diferente.

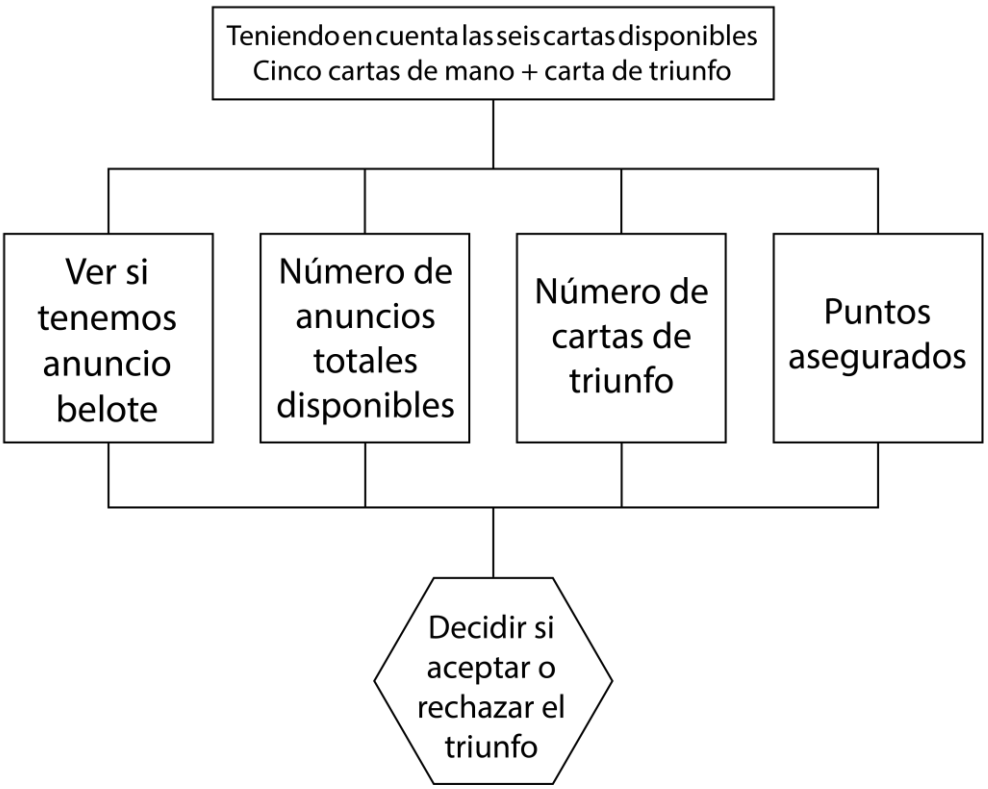

Figura 35. Esquema de la estrategia de los bots durante la fase selección de triunfo

Una vez el palo de triunfo ha sido elegido y se comienzan a jugar las cartas de la baza, se puede separar la estrategia en dos situaciones (Figura 36). Primero, si eres el primer jugador de la baza, es recomendable jugar las cartas más altas que no son del palo de triunfo, asegurándote los puntos de dichas cartas para sumar en las bazas finales. Esto significa jugar primero los ases y dieces de cada palo en ese orden, evitando jugar las cartas del palo de triunfo hasta el final. Una vez solo se dispongan de cartas de triunfo, se dará prioridad máxima a los valets de triunfo, ya que superan en valor a cualquier otra carta, guardándolos para la baza final y asegurar así las diez de últimas.

La siguiente situación por contemplar es cuando no eres el primer jugador de la baza y tienes que responder a las cartas ya tiradas. Estas elecciones están muy restringidas por las reglas del juego, ya que obligan a jugar una carta del palo inicial de la baza, y en caso de no disponer de dicho palo, del palo de triunfo. Por este motivo las elecciones tomadas tendrán el objetivo de minimizar los puntos ganados por los oponentes en caso de liderar ellos la baza (tener la carta más alta de las cartas en la mesa), o asegurar cartas de puntos menores (cartas excepto as y diez) en caso de que nuestro compañero lidere la baza. Si estas obligado a lanzar carta de triunfo, se priorizará tirar las cartas de menor valor, asegurando como siempre que las cartas de mayor valor (valets y nueves) estarán disponibles a final de la ronda.

El siguiente paso del diseño de la inteligencia artificial será la selección de los parámetros más relevantes y el peso de cada uno de ellos para seguir la estrategia descrita.

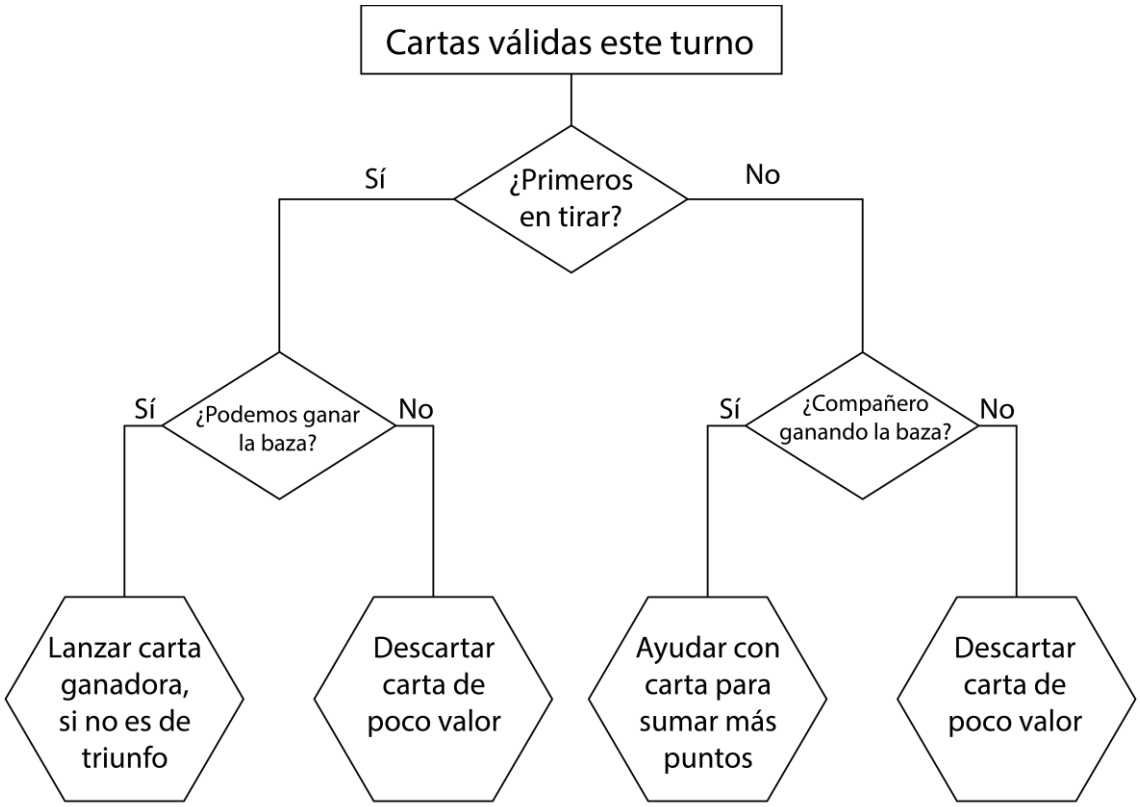

Figura 36. Esquema de la estrategia de los bots durante la fase de juego de bazas

6.4.2 Parámetros y pesos elegidos para la inteligencia artificial

Ahora, se realizará un desglose de la estrategia, repasando los puntos a seguir en cada fase de la ronda y cuáles serán los parámetros utilizados para cumplir cada punto.

Parámetros y pesos durante la selección de triunfo

Durante la fase de selección de triunfo, los puntos a seguir quedan muy marcados:

- Número de cartas del palo de triunfo: muy autodescriptivo, este punto se representará con el parámetro *'NUM_TRUMP_CARDS'*. Este se calculará directamente contando el número de cartas de triunfo en nuestra mano.
- Número de anuncios disponibles: de nuevo muy descriptivo, se representará mediante el parámetro *'CALLS'*. Se calcularán los anuncios que existen con las seis cartas de nuestra mano y su nivel.
- Anuncio belote: igual de simple que el primer punto, se representará directamente mediante el parámetro *'BELOTE'*. Se verá si se tiene el anuncio de belote con el palo de triunfo.

- Puntos asegurados: este punto de nuevo será representado por un único parámetro *'ASSURED_POINTS'*, pero resulta ser el más complicado de calcular ya que requiere de un cálculo de dichos puntos, los cuales hay que valorar con cierta incertidumbre y no es un cálculo seguro.

Una vez descritos los parámetros, es necesario definir los pesos para cada uno de ellos, hay que recordar que todos estos pesos han de sumar un total de 1 entre todos:

- *NUM_TRUMP_CARDS*: se le asignará un peso elevado ya que estas cartas de triunfo aseguran en parte las bazas donde las usemos, pero sigue teniendo mucha incertidumbre ya que no se conocen las cartas de los oponentes. Le será asignado un 0.2 del total.
- *CALLS*: este parámetro puede asegurar muchos puntos en la ronda, en caso de disponer de un anuncio elevado. Sigue habiendo cierto grado de incertidumbre ya que los oponentes podrían tener anuncios mayores y cancelar nuestros puntos. Le será asignado un 0.2 del total.
- *BELOTE*: a pesar de ser el parámetro que asegura puntos a final de ronda en caso de cumplirse, son muy pocos puntos y el rey y reina de triunfo no son cartas de puntos elevados, por lo que no aseguran bazas ganadas. Le será asignado un 0.1 del total.
- *ASSURED_POINTS*: este último parámetro resulta ser el más prioritario ya que gracias al cálculo realizado y a pesar de la incertidumbre, muestra una idea clara de cuántos puntos se pueden conseguir por nuestra cuenta a final de la ronda, y cuál es nuestra posibilidad más realista de ganar o no la ronda. Le será asignado un 0.5 del total.

Parámetros y pesos durante la fase de juego de bazas

Luego, durante la fase donde se juegan las manos ambas situaciones principales de la estrategia tendrán los mismos puntos para simplificar la implementación, estos serán:

- Probabilidad de ganar la mano: este punto indica, como dice el nombre, la probabilidad de la carta que está siendo valorada de ganar la mano a jugar. Para realizar este cálculo se tienen en cuenta las cartas desconocidas mayores que la carta y las propias cartas poseídas (p.e. si se tienen el as y diez de un mismo palo no triunfo, ambas tendrán el total del punto). Viene definido por el parámetro *WIN*.
- Importancia de guardar para el final: como se explica en la estrategia, es conveniente guardar las cartas de triunfo para el final, elevando este valor a medida que aumenta el valor de la carta. Es definido con el parámetro *RESERVE*.
- Importancia de descarte de la carta: este punto trabaja junto al punto anterior, ya que las cartas de no triunfo bajas tienen que ser descartadas ya que tienen muy pocas posibilidades de ganar la ronda. Se define mediante el parámetro *DISCARD*.

- Minimizar el número de puntos ganados por los oponentes: cuando un oponente va ganando la mano, se recomienda tirar una carta del menor valor posible para que se ganen los mínimos puntos posibles. Se define con el parámetro *HURT_OPPONENT*S
- Maximizar el número de punto ganados por el equipo: al contrario que el punto anterior, cuando el compañero va ganando la baza, es recomendable tirar una carta del mayor valor posible. Se define con el parámetro *HELP_PARTNER*.

Se pasa ahora a definir los pesos para los parámetros utilizados a la hora de elegir la carta a tirar:

- *WIN*: ya que este valor es una aproximación en muchos casos al no conocer la mayoría de las cartas disponibles por los oponentes o compañero, tendrá un peso algo conservador, no definiendo únicamente si esa carta se juega o no. Tiene un peso de 0.25.
- *RESERVE*: este valor será el más relevante, ya que es de vital importancia guardarse las cartas más elevadas de triunfo para asegurar el bonus de 10 puntos extra además de los puntos de la mano. Se le asignará un valor de 0.4.
- *DISCARD*: es cierto que este parámetro indica la importancia para descartar una carta de bajo valor, pero hay que asegurarse que no solo se descarta la carta, si no que se hace de una forma eficaz. Se le asignará un valor de 0.1.
- *HURT_OPONENTS*: este valor tampoco será definitorio de la carta a elegir, pero si puede ayudar a la hora de desempatar junto a otra carta, ya que aumentará a menor sea el valor de la carta en caso de estar ganando un oponente. Tendrá un peso de 0.15. Es mayor a los pesos de descarte y ayuda al compañero porque es muy favorable reducir los puntos ganados por el equipo contrario.
- *HELP_PARTNER*: ya que muchas veces no es seguro que nuestro compañero gane la baza a pesar de ir ganando momentáneamente, se priorizará dañar a los oponentes a la hora de tomar la decisión. Este tendrá un peso de 0.1.

6.4.3 Proceso de valoración de acciones a tomar

Una vez explorada la estrategia a seguir junto a los parámetros y sus pesos para cada fase de la ronda, se explicará la línea de ejecución de cada caso mediante pseudocódigo. Se comenzará con la fase de selección de triunfo y se continuará con la fase de juego.

Pseudocódigo de la fase la selección de triunfo

1. **Inicio**
2. **Inicializar** diccionario de parámetros junto a su peso
3. **Inicializar** diccionario de parámetros con valor nulo para la acción de aceptar
4. **Inicializar** diccionario de parámetros con valor nulo para la acción de rechazar
5. **Valorar** el parámetro *CALLS* para cada opción
6. **Inicializar** diccionario con los valores de aceptar y rechazar para cada palo

7. **Para** cada palo:
    a. **Valorar** el parámetro *BELOTE*
    b. **Valorar** el parámetro *ASSURED_POINTS*
    c. **Valorar** el parámetro *NUM_TRUMP_CARDS*
    d. **Multiplicar** cada valor de parámetro por su peso
8. **Si** estado del juego == primera ronda de triunfo:
    a. **Devolver** acción a tomar del palo de triunfo
9. **Si no**:
    a. **Devolver** acción con mayor valor del palo con mayor suma de valores
10. **Fin**

Primero de todo, se inician los diccionarios que contienen los parámetros y sus pesos y los parámetros y sus valores. Se creará una instancia de este último par para cada acción disponible en el caso de la fase de triunfo, aceptar o rechazar el triunfo. Entonces se valorará el parámetro *CALLS*, común a ambas e indiferente al palo, ya que los anuncios no dependen del triunfo. Esta valoración tendrá en cuenta las seis cartas disponibles en caso de aceptar y solo las cinco de la mano en el caso de rechazar.

A continuación, se inicializa un cuarto diccionario, esta vez teniendo como parejas los palos junto a sus valores para ser aceptados o rechazados. Seguido, se recorrerán los palos en este último diccionario, valorando el resto de los parámetros sí dependientes del palo. Una vez valorados los parámetros según la acción, su valor se multiplicará por su peso, para posteriormente ser sumados y ofrecer así el valor de la heurística para la acción.

Una vez se posee el resultado de la heurística para acción según el palo, se separa la ejecución en si la partida está en la primera ronda de selección de triunfo o en la segunda. Durante la primera, solo se tendrán en cuenta los valores de las acciones del palo de triunfo único a elegir, mientras que en la segunda se tendrán en cuenta los valores de las acciones del palo que obtenga la suma total mayor de los valores. Por último, se devolverá la acción con mayor valor total.

Pseudocódigo de la fase de juego de bazas

1. **Inicio**
2. **Adquirir** cartas válidas de la mano
3. **Si** cartas válidas == 1:
    a. **Devolver** única carta válida para jugar
4. **Si no**:

a. **Inicializar** diccionario de parámetros junto a su peso
b. **Inicializar** diccionario de parámetros con valor nulo
c. **Inicializar** diccionario de cartas con sus propios parámetros
d. **Para** cada carta válida:
    i. **Si** la mesa está vacía:
        1. **Valorar** el parámetro *WIN*
        2. **Valorar** el parámetro *RESERVE*
        3. **Valorar** el parámetro *HURT_OPONENTS*
    ii. **Si no**:
        1. **Valorar** el parámetro *WIN*
        2. **Valorar** el parámetro *RESERVE*
        3. **Si** la pareja va ganando la ronda:
            a. **Valorar** el parámetro *HELP_PARTNER*
    iii. **Valorar** el parámetro DISCARD
    iv. **Multiplicar** cada valor de parámetro por su peso
e. **Devolver** carta con mayor suma total de valores

5. **Fin**

En este caso, se comienza la ejecución observando las cartas validas a jugar en el turno y si solo existe una única carta a lanzar, se cortará la ejecución del algoritmo devolviendo directamente esta. En caso contrario, se comienza un proceso similar a la fase de selección de triunfo, comenzando por la inicialización de los diccionarios con los pesos y valores de los parámetros. A diferencia de la fase anterior, se creará un diccionario asociando las cartas válidas para jugar con sus propios parámetros junto a sus valores.

Entonces, se recorrerán estas cartas una a una y valorarán los diferentes parámetros. Primero, se separa la ejecución si el bot es el primer jugador de la baza, en cuyo caso valora el parámetro *WIN* respecto a la posibilidad de dicha carta para ganar a cualquier otra carta; seguido se valorarán el resto de los parámetros. En caso de no ser el primer jugador de la baza, el parámetro *WIN* se valorará respecto a si la carta a valorar gana o no a las cartas ya en la mesa; además, en este caso, si el jugador que está ganando la baza actualmente es el compañero, se valorará también el parámetro *HELP_PARTNER*. Por último, se valora el parámetro *DISCARD*, común en ambos casos.

Al igual que en la fase previa, se procede a multiplicar los valores de los parámetros por sus presos para obtener el valor final de su heurística. En este caso, la decisión final es más simple, eligiendo la carta cuya función ofrezca un valor mayor.

6.4.4 Posesión de jugadores que desconectan

Cuando un jugador se desconecta durante el transcurso de una partida de forma permanente o temporal, es vital asegurar que la partida continúe sin ningún problema sin interrumpir la experiencia del resto de jugadores. Es por este motivo que los bots serán capaces de poseer a los jugadores reales desconectados y tomar decisiones por ellos mientras están ausentes, dando así una sensación de juego continúa para el resto.

El deseado flujo de funcionamiento de esta característica ha de ser el siguiente: cuando un jugador se desconecta de la partida, es poseído por un bot y comenzará un contador interno. A partir de este momento, si el jugador vuelve el bot dejará de poseerle, si no vuelve el bot pasará a poseerle hasta el final de la partida. Para visualizar mejor este proceso, referirse a la Figura 18, donde se explica el proceso de reconexión y posesión por parte de la IA.

6.5 Desarrollo de la inteligencia artificial

Ya definida y diseñada la estrategia a seguir, es hora de pasar al desarrollo de esta inteligencia artificial. Se tratarán el desarrollo de los métodos y operaciones clave para seguir la estrategia y el funcionamiento de estos una vez implementados. También se comentarán las mayores dificultades a la hora del desarrollo, tratando sus implicaciones y como se sobrellevaron en el momento.

6.5.1 Métodos y operaciones clave

Conteo de puntos asegurados

A la hora de decidir si aceptar el palo de triunfo o no, el parámetro a evaluar con más peso resulta ser la aproximación a los puntos que aseguran las cartas en nuestra mano. Por este motivo, el cálculo de estos puntos resulta ser clave para el éxito de la inteligencia artificial.

La función es un método de extensión de listas de cartas de la belote, el cual requiere el palo de triunfo a evaluar como único parámetro, presentando la siguiente cabecera:

```
private fun List<BeloteCard>.getSecuredPoints(trumpSuit: FrenchSuit? = null): Int
```

Figura 37. Cabecera del método 'getSecuredPoints'

Se puede observar que devuelve un valor de tipo *Int*, el cual resultan ser los puntos aseguras con las cartas actuales según el palo indicado. El cálculo se realiza mediante una búsqueda ordenada de mayor a menor de las cartas que aseguran una victoria, según sean de triunfo o no. Esto es, se acumulan los puntos que asegura cada as que poseamos, seguidos del 10 del mismo palo si se tiene, en el caso del palo de triunfo, se comienza con el valet. Una vez encontrada una carta que con corta el orden descendente, se pasa al

siguiente triunfo. De este modo, se recorren todos los palos normales, para los cuales solo se tienen en cuenta los ases y dieces, ya que son las más elevadas; en el caso del palo de triunfo indicado, se realiza la búsqueda con todas las cartas ya que, a pesar de no tener gran valor, tienen prioridad en la victoria frente al resto de palos. Una vez recorridos todos los palos, se devuelve el valor de los puntos acumulados.

Confirmación que los oponentes poseen una carta de triunfo mayor

En la fase de juego, siempre se busca ganar la última baza ya que asegura 10 puntos adicionales, es por este motivo que se busca lo antes posibles obligar a los oponentes a descartar sus cartas de triunfo. Además, cuando se valora tirar una carta de triunfo, hay que tener especial cuidado y tener en cuenta las cartas conocidas de los oponentes en dicho momento. Es por estos motivos que se necesita un método a la hora de evaluar las cartas de triunfo que asegure si existen cartas mayores de triunfo.

Este simple método utiliza el estado de la partida para asegurar que no existen triunfos mayores al que se quiere mirar, o si se puede obligar a un oponente a deshacerse de un triunfo mayor. Su cabecera es la siguiente:

```
fun oponentsHaveHigherTrump(): Boolean {
```

Figura 38. Cabecera del método 'oponentsHaveHigherTrump'

El método devuelve un booleano indicando si existe un oponente con una carta de triunfo mayor en estos momentos. Para calcular el estado actual de la partida, se poseen una serie de variables las cuales indican las cartas conocidas de los oponentes y el compañero, al igual que las cartas desconocidas en general. La única forma de conocer las cartas de otros jugadores es durante la fase de presentación de los anuncios, donde los ganadores de estos muestran las combinaciones al resto de jugadores. A partir de este momento, la inteligencia los tendrá en cuenta para sus cálculos. El cálculo se realiza con estos conocimientos, evaluando si la mayor carta de triunfo conocida de los oponentes es mayor a la mayor carta de triunfo de nuestro equipo.

6.5.2 Desafíos encontrados durante el desarrollo

Durante el desarrollo de la inteligencia artificial, existió un reto principal, el cual era necesario superar para cumplir los objetivos marcados. Este está condicionado por la ambición del estudiante, la cual chocaba con una falta de infraestructura en el proyecto actual.

Pesos correctos para cumplir la estrategia marcada

Para que la inteligencia artificial desarrollada cumpliera con la estrategia diseñada, se debían tener unos valores correctos para cada uno de los parámetros, que asegurasen una toma de decisiones de acuerdo con lo marcado.

La forma ideal para obtener estos valores hubiese sido entrenando a la IA, ofreciéndole unos valores iniciales y una cierta aleatoriedad con la que explorase diferentes pesos. Entonces, en los casos donde ganase las bazas o rondas, mover los valores originales hacia esos nuevos valores explorados. Este entrenamiento acabaría convergiendo en unos valores óptimos, los cuales serían utilizados como valores finales.

Desafortunadamente, no se contaba con la infraestructura ni con el tiempo necesario para desarrollarlo, por lo que se realizó una aproximación a esta estrategia de forma manual. Este acercamiento manual requería de un mínimo desarrollo a la estructura actual utilizada para probar los bots del resto de juegos. En resumen, este método consiste en la simulación de partidas de bots donde los pesos serán elegidos arbitrariamente, siendo los más exitosos los elegidos para la versión a implementar de la IA.

Previo a la prueba de diferentes pesos, es necesario un sistema de simulaciones de partidas, donde se puede elegir el número de partidas a simular y se puedan ver diferentes resultados una vez finalizadas las partidas. A la hora del desarrollo, el proyecto ya contaba con un sistema de simulación de partidas, pero estas no mostraban ningún tipo de dato al final según los resultados, ya que solo eran utilizadas para comprobar el correcto funcionamiento de los bots y la lógica del juego. Es por este motivo que se tuvo que realizar unos cambios a la implementación, haciendo los cálculos para los resultados buscados.

Por suerte solo se necesitarán los resultados porcentuales de partidas ganadas y perdidas por cada equipo, ya que se enfrentarán a diferentes versiones de IA con diferentes pesos para ver cual resulta el más victorioso. Se llamará a todo este proceso “entrenamiento manual”.

Las simulaciones necesarias se realizarán en la clase llamada “BeloteSimulation”, la cual inicia un servidor local donde simular las partidas. Estas simulaciones junto con el *Logger* disponible por parte del servidor emulado, con el cual se escribirá en la consola, permitiendo llevar un conteo de las partidas que gana cada equipo, y pudiendo realizar el cálculo de las partidas ganadas por cada equipo al final de la simulación.

Durante las pruebas de la IA, se simularon miles de partidas modificando los valores sus parámetros, llegando a unos los cuales conseguían ganar al mayor número de IA contra las que se enfrentaban. Una vez implementado el juego en la versión de producción los valores se irán regulando según se vea su eficacia frente a los jugadores humanos.

7. Conclusiones y trabajo futuro

En este último capítulo se concluirá la memoria de este trabajo. Se comenzará repasando la aplicación de los conocimientos adquiridos durante los estudios de la carrera, continuando con la valoración del cumplimiento de los objetivos marcados al inicio del documento y se finalizará con una breve visión al futuro marcando los trabajos futuros a realizar.

7.1 Valoración de los conocimientos aplicados

El objetivo principal del trabajo es poder demostrar los conocimientos adquiridos durante los años universitarios, los cuales permitieron el desarrollo satisfactorio del proyecto. Se repasarán los aspectos más relevantes del proyecto donde los conocimientos adquiridos fueron de mayor ayuda, siendo estos los relacionados con la ingeniería del software y diseño y programación de inteligencias artificiales. También se hablará de la competencia transversal relacionada con la rápida adaptación a nuevas tecnologías, la cual resultó de gran ayuda durante el todo el proceso de desarrollo.

7.1.1 Puesta en práctica de la ingeniería del software

Se comenzará por la puesta en práctica de los conocimientos adquiridos en la rama de ingeniería del software en un ámbito de trabajo real y con un flujo de trabajo adaptado a una aplicación ya en el mercado. Durante los años cursando las asignaturas de la especialización en el grado se realizaron numerosos proyectos donde poner en práctica los conocimientos teóricos adquiridos, pero esta oportunidad presentó un nuevo nivel en el que tener que utilizar estos.

Principalmente, esta aplicación se aprecia en la utilización de metodologías ágiles a la hora de la propuesta de las nuevas fases de desarrollo para los juegos dentro de la empresa. Esta propuesta orbita alrededor de la idea del desarrollo enfocado a MVP, siendo validados por diferentes clientes según su objetivo.

Además, los conocimientos adquiridos en la carrera ayudaron en la adaptación al nuevo ambiente de trabajo, permitiendo entender términos como sprint, tablero Kanban o backlog. Esta ayuda resultó crucial a la hora de poder organizar las tareas dentro de las épicas que formaban los MVP, y poder ayudar a mejorar el desarrollo ágil dentro de la empresa.

7.1.2 Conocimientos detrás del diseño y desarrollo de la nueva IA

Continuando con el diseño y desarrollo de la nueva inteligencia artificial, se pudo definir una IA totalmente innovadora dentro de la empresa y que conseguía mejorar en muchos de los aspectos en los que otras se quedaban atrás. Estos conocimientos resultan ser una

mezcla de los adquiridos durante el curso del grado y los aprendidos durante la experiencia de intercambio realizada, demostrando así la eficacia de estos programas y su posible aplicación en el mundo laboral.

Estos conocimientos no solo ayudaron al diseño de una inteligencia artificial más refinada y compleja en este aspecto, sino que también permitieron marcar unos límites que esta necesitaba cumplir para poder aplicarse de forma realista. Hay que tener en cuenta que se debían seguir unas fechas marcadas para el desarrollo, dejando margen para una posibilidad de mejora en un futuro. Es aquí donde también entran los conocimientos a la hora de decidir qué tipo de IA a llevar a cabo, resultando en un diseño estático, con heurísticas y con capacidad de ser entrado.

7.1.3 Rápida adaptación a las nuevas tecnologías

Finalmente, también se explorará el impacto que tuvo la competencia transversal de adaptación a nuevas tecnologías, ya que, a pesar de no estar relacionada directamente con una asignatura o especialización, sí que habla de los conocimientos transversales que ofrece el curso de un grado como el de ingeniería informática.

Dentro de la empresa se utilizan unas herramientas con las cuales no se poseía ninguna relación ni conocimiento previo, pero aun así resultó sorprendente la rapidez con la cual se consiguió adaptar a estas y comenzar a ser eficiente y eficaz durante el tiempo de trabajo. Haciendo una acción de retrospectiva, se observa que esta rápida adaptación no fue casualidad, y se consiguió gracias al elevado y variado número de tecnologías utilizadas y aprendidas durante la asistencia en la universidad.

Resulta sorprendente cuán importante este aspecto resulta ser, puesto que, aplicado en un futuro, abre muchas puertas a nuevos conocimientos y poder estar siempre actualizado en nuevas tecnologías, aspecto crucial en el mundo informático. Esta habilidad no hubiese sido posible si durante el curso de la universidad no se hubiesen explorado tantos lenguajes tan diferentes entre ellos o se hubiese trabajado con diferentes IDE.

7.2 Impacto de la explotación de la belote

Como uno de los objetivos principales del desarrollo de la belote se encuentra aumentar el número de jugadores franceses dentro de la aplicación. En este punto se comprarán diferentes datos que confirman este crecimiento y por lo tanto la eficacia de la introducción de este nuevo juego al catálogo de *Playjoy*. Se presentarán diferentes tipos de variables utilizadas en la empresa para medir el impacto de la aplicación dentro de un mercado especifico. Estos datos se podrán dividir en estadísticas directas, las cuales representan el número de jugadores actuales y el número de instalaciones y su evolución, y luego están los datos de participación, los cuales muestran el involucramiento de estos jugadores en la aplicación utilizando el DAU (Usuarios Activos Diariamente en inglés) y MAU (Usuarios Activos Mensualmente en inglés) de la aplicación.

DAU, MAU y en especial su ratio DAU/MAU son datos que sirven para conocer la lealtad y participación de los jugadores con una aplicación. Además, se han utilizado desde hace más de una década en juegos en línea en los cuales esta lealtad marca el valor del producto a corto y largo plazo (19). Dentro de *Playjoy* estos datos ayudan a medir la probabilidad existente de los jugadores quedándose en el ecosistema sin saltar entre diferentes aplicaciones similares.

Con el belote incorporado al catálogo el día 15 de septiembre de 2021, los datos mostrados tomarán principios de septiembre como fecha de inicio y mediados de junio como fecha final.

7.2.1 Estadísticas de la belote

La empresa no cuenta con cálculos estadísticos específicos para cada juego, por lo que resulta imposible conseguir un número fiable de jugadores actuales de la belote. A pesar de esto, sí se pueden realizar llamadas a la base de datos con el objetivo de conseguir el número total de partidas jugadas hasta la fecha, resultando ser un total de 21.349 a fecha de redacción. Mientras que este número no representa los datos del juego actual antes mencionados, permite observar que el juego es funcional y nos permite calcular una media de 2.135 partidas jugadas por mes desde su salida en septiembre. Se puede observar entonces que el juego ha sido jugado de manera constante desde su lanzamiento, implantándose de esta manera como un juego siendo explotado dentro de la aplicación.

7.2.2 Impacto en los jugadores generales de la aplicación

En este punto se repasará el impacto que la belote ha tenido en el ámbito general de la aplicación. Para estudiar este impacto se expondrá el DAU, MAU y el proporción DAU/MAU.

Comenzando por el DAU, MAU y su ratio DAU/MAU (Figura 39), se puede observar como la aplicación cuenta con una tendencia positiva en los usuarios activos diariamente (1), lo cual es un claro signo de crecimiento en la base de jugadores activa. Este incremento se puede apreciar en los usuarios activos mensualmente (2), cuya base de jugadores activos mantenidos del mes anterior aumenta cada mes, indicando que parte de los jugadores se mantienen en la aplicación una vez creada la cuenta. A pesar de estos datos positivos, se presenta una proporción DAU/MAU estable, lo cual a pesar de no ser ideal marca un crecimiento estable y una lealtad del usuario fiable.

A pesar de estos datos indicar un crecimiento positivo general de la aplicación, mediante ellos no se puede crear una imagen clara del impacto en la aplicación que el belote ha presentado. Esto se debe a que el juego está muy enfocado al mercado francés y no apunta a ser el juego principal del groso de jugadores de la aplicación.

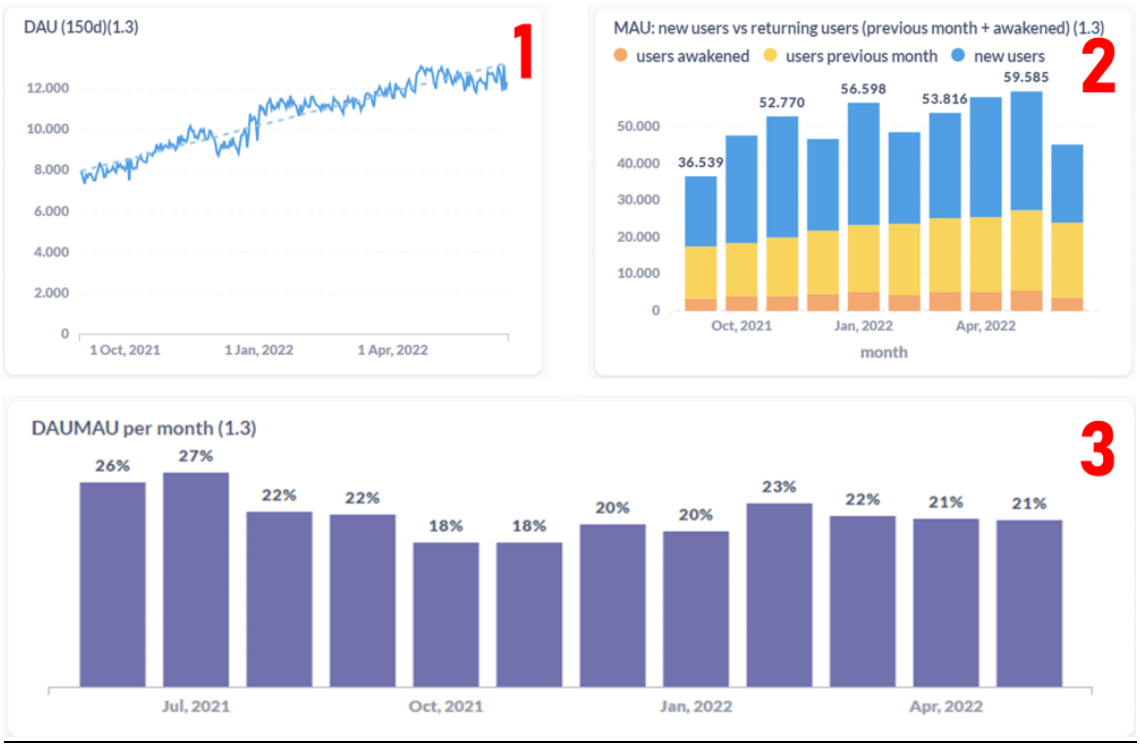

Figura 39. DAU, MAU y ratio DAU/MAU de Playjoy

7.2.3 Impacto en el mercado francés

Ahora se estudiará el impacto que la belote ha tenido en el mercado francés, comunidad donde este es un juego tradicional y es el objetivo principal de esta incorporación. Este estudio será más detallado que el anterior, mostrando no solo los datos anteriores si no incluyendo también los inicios de sesión por primera vez de los jugadores franceses y una comparación del número total de jugadores procedentes de Francia antes de incluir la belote, y el número actual.

De nuevo, se comenzará por el DAU, MAU y su ratio (Figura 40), donde se observa un incremento mucho más marcado. En cuanto al DAU (1), este obtuvo un gran crecimiento desde un total prácticamente nulo hasta un total de unos cien usuarios diarios activos. Este incremento ocurre a mediados de septiembre al incorporar la belote, presentando un crecimiento constante desde entonces hasta un total de 600 usuarios franceses activos diariamente. Pasando al MAU (2), este resulta ligeramente diferente al general, puesto que posee una distribución de jugadores diferentes, donde en el caso general se encuentra cercano al 50% de jugadores nuevos y mantenidos del mes anterior, en el caso de Francia existe un número de jugadores nuevo mucho mayor que al número de jugadores mantenido. Este se debe al hecho de ser un mercado nuevo, donde se comienza de una base de jugadores casi nula, y la gran mayoría son jugadores nuevos interesados por primera vez en la aplicación. Esta disparidad queda también reflejada en el bajo porcentaje DAU/MAU, el cual apoya un número de jugadores nuevos mayor, los cuales no muestran tanta lealtad hacia la aplicación como el resto de los mercados más establecidos.

Figura 40

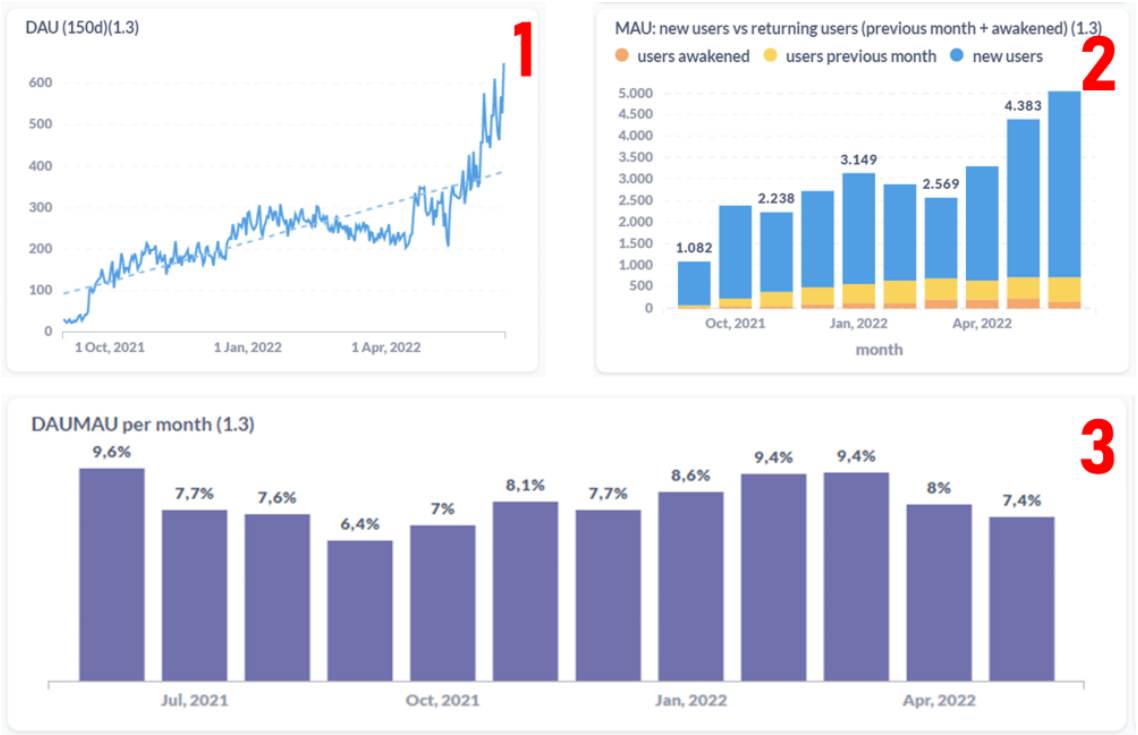

Figura 40. DAU, MAU y su ratio DAU/MAU del mercado francés

Pasando ahora a los inicios de sesión por primera vez (Figura 41), se observa una gráfica similar a las anteriores, con un crecimiento constante a partir de mediados de septiembre. Este número soporta a las gráficas anteriores puesto que estos nuevos inicios de sesión constantes suponen la llegada de nuevos jugadores franceses, los cuales aumentan el número de jugadores diarios, pero al no ser habituales en la aplicación, disminuyen el valor del DAU/MAU.

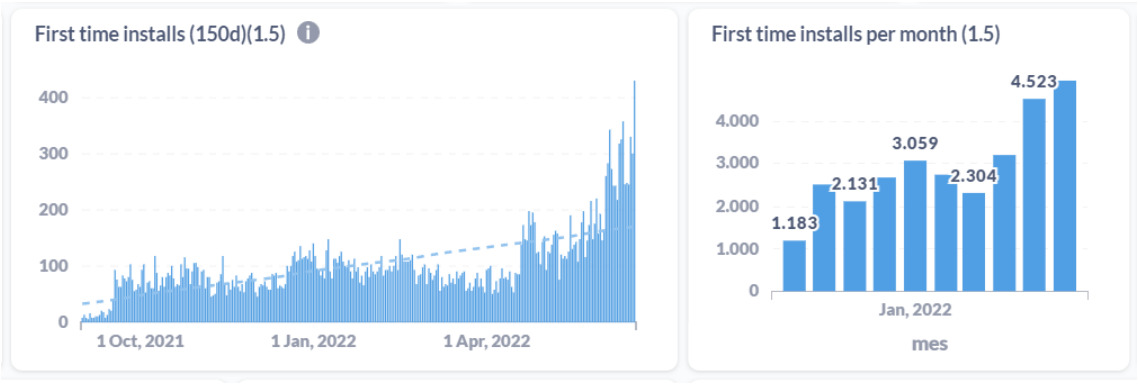

Figura 41. Primeros inicios de sesión en Francia

Para concluir este punto, se compararán el número total de jugadores franceses en las fechas anteriores a la implementación de la belote con el número actual (Figura 42), demostrando la eficacia que la incorporación de este juego a presentado en el objetivo de introducir la aplicación al mercado francés. Se observa un incremento de 1.886 jugadores (1) hasta 27.076 (2). Este dato hay que tomarlo con un grano de sal, puesto que como se

puede observar en el resto de los países, viene acompañado de un aumento general de la base de jugadores en la aplicación.

1

group	users
OTHERS	186.812
ESPAÑA	84.732
LATAM	67.848
RUSIA	36.089
BRPT	13.703
EN	5.452
FRANCIA	1.886

2

group	users
OTHERS	228.329
ESPAÑA	169.379
LATAM	157.155
BRPT	37.761
RUSIA	37.140
FRANCIA	27.076
EN	15.333

Figura 42. Comparativa de jugadores previos a la belote y posteriores

7.3 Valoración de las nuevas fases de desarrollo

Para el desarrollo de este juego se propuso el diseño y seguimiento de nuevas fases comunes para el desarrollo de juegos futuros. De esta forma, se conseguiría una programación del videojuego separada en MVP con objetivos comunes, permitiendo a todo el equipo tener una idea del progreso con solamente mirar la fase en la que se encuentra el juego en específico. Además, con esta separación se espera poder organizar las campañas de marketing de forma más eficaz, ya que una vez alcanzados ciertos hitos de desarrollo el equipo de marketing podrá hacerse una idea más clara del tiempo restante para su finalización.

Estas fases están unidas a unos tiempos de desarrollo específicos, como ha sido visto en la página 41. Estas fechas fueron marcadas por el equipo de desarrollo utilizando los conocimientos previos obtenidos con la creación de los otros juegos, y suponían el escenario ideal de desarrollo. A pesar de este conocimiento previo, durante el desarrollo del juego todavía no se tenía una maestría completa de las herramientas, por lo que se sufrieron notables retrasos frente a estas fechas, los motivos específicos quedan reflejados en la página 64.

A pesar de estos retrasos, la nueva organización mostró muchos positivos y cumplió con sus objetivos principales de mostrar un progreso más transparente al resto del equipo y ayudar con las campañas de marketing y cuando llevarlas a cabo. Además, gracias al avance marcado por hitos y con un objetivo final por fase, se consiguió un producto final mucho más refinado, ya que no se avanzaría hasta la siguiente fase hasta cumplir dichos objetivos.

Por lo tanto, teniendo en cuenta que lo retrasos se dieron por falta de familiarización con las herramientas de desarrollo, el equipo valoró esta nueva organización como algo positivo y necesario para futuros juegos.

7.4 Trabajo futuro

Ya con la belote implementada en la aplicación y una comunidad francesa decente en tamaño, es hora de tratar el trabajo futuro tanto para la belote como para este mercado francés. Se hablará del siguiente juego con objetivo de aumentar y afianzar el mercado francés al igual que el trabajo actual en la belote y el futuro de la inteligencia artificial diseñada e implementada.

7.4.1 Desarrollo de la coinché

La belote, al tratarse de un juego tradicional, cuenta una infinidad de variantes según el lugar específico donde se juegue, pero solo unas pocas de ellas marcan diferencias suficientes como para ser tratadas de juegos nuevos. En la empresa, se diferencian tres versiones principales gracias al conocimiento adquirido en proyectos previos: versión clásica, clásica con anuncios y la coinché. De estas, la última resulta ser la más diferente, pues cuenta con una lógica de juego totalmente diferente. Es por este motivo que esta versión será el siguiente juego por desarrollar con objetivo de aumentar el mercado francés y afianzarse dentro de este. Este desarrollo de espera que sea mucho más corto que el anterior, pues el juego comparte las bases de la lógica de la belote.

Además, también servirá para observar el progreso en la adquisición de conocimientos de las herramientas y refinamiento de las fases de desarrollo nuevas vistas en esta memoria.

7.4.2 Fase de mantenimiento de la belote

En cuanto a la belote, a pesar de haber finalizado su desarrollo, se continúa trabajando en ella mediante el seguimiento de fallos y especial atención al cliente. Este seguimiento será continuo a partir de este momento, solucionando los errores que surjan al igual que monitorizando las opiniones de los usuarios para posibles mejoras.

7.4.3 Mejoras a la IA y su proceso de entrenamiento

Finalizando, la idea inicial de la inteligencia artificial a desarrollar se vio acortada por las limitaciones técnicas del momento. Con el objetivo de implementar una inteligencia artificial similar en el resto de los juegos, también se busca crear la infraestructura necesaria para poder exprimir al máximo las capacidades de esta implementación. Esta infraestructura permitirá el entrenamiento automático de la IA, permitiendo conseguir de este modo unos valores óptimos de manera más eficiente y eficaz.

8. Referencias

1. **AEVI.** *La industria del videojuego en España Anuario 2019.* s.l. : AEVI, 2019.

2. **Newzoo.** *Global Games Market Report 2021.* s.l. : Newzoo, 2021.

3. —. *Global Games Market Report 2020.* s.l. : Newzoo, 2020.

4. —. *Global Games Market Report 2019.* s.l. : Newzoo, 2019.

5. **AEVI.** *La industria del videojuego en España Anuario 2020.* s.l. : AEVI, 2020.

6. **Monnen, Devin and Goldberg, Martin.** Space Odyssey: The long journey of Spacewar! from MIT to computer labs around the world. *Kinephanos: Journal of Media Studies and Popular Culture.* June 2015, 2015, Special Issue.

7. *Anecdotes: Magnavox and Intel: an Odyssey and the early days of the Arpanet.* **Mazor, Stanley, Salmon, Peter y T. Kirsten, Peter.** 3, s.l. : IEEE, 2009, Vol. 31. 1934-1547.

8. **Ernkvist, Mirko.** *Down many times, but still playing the game: creative destruction and industry crashes in the early video game industry 1971-1986.* 2008.

9. **Rutter, Jason and Bryce, Jo.** *Understanding digital games.* London : SAGE Publications Ltd, 2006. 1-4129-0033-6.

10. **Negron, Sage.** *Will we ever see another generation of handheld consoles?* s.l. : CBR, 2020.

11. **plinq.** *Plinq Mobile Games 2019.* s.l. : Hubspot, 2019.

12. *El análisis DAFO y los objetivos estratégicos.* **Díaz Olivera, Armando Pablo y Matamoros Hernández, Idalberto Benjamín.** Marzo, s.l. : Contribuciones a la economía, 2011.

13. **MkKinsey Agile Tribe.** *Los 5 rasgos distintivosde las organizaciones ágiles.* 2018.

14. **Somerville, Ian.** *SOFTWARE ENGINEERING (9TH EDITION).* 78-0-13-703515-1.

15. **Melegati, Jorge, et al.** *MVP and experimentation in software startups: a qualitative survey.* s.l. : IEEE, 2020.

16. *Gantt charts revisited: A critical analysis of its roots and implications to the management of projects today.* **Geraldi, Joana and Lechter, Thomas.** 4, s.l. : International Journal of Managing Projects in Business, 2012, Vol. 5. 1753-8378.

17. **Mozolevska, Victoria.** *Cross-platform game development: when versatility matters.* s.l. : Kevuru Games, 2021.

18. **Hunt, Andrew and Thomas, David.** *Pragmatic unit testing in Java with JUnit.* s.l. : The Pragmatic Bookshelf, 2003. 013211917X.

19. **Lee, Katherine.** *Non-Financial Metrics, DAU, MAU and DAU/MAU: An Integrative Framework and Analysis.* s.l. : University of Pensylvania, 2022.

Anexo 1. NoSpoon

NoSpoon Tech Lab es una empresa emergente, la cual ofrece equipos de trabajo y material para llevar a cabo ideas de diferentes ámbitos novedosas y potencialmente exitosas. La empresa nació en 2016 a manos de Carles Pons y Daniel G. Blázquez, los cuales se adentraban en este nuevo proyecto tras su éxito con la empresa *Akamon* fundada en 2011 o el estudio de videojuegos *Exelweiss* en 2000.

Los proyectos llevados a cabo dentro de la empresa pueden surgir desde el interior de esta o pueden ser propuestas por agentes externos. Estas ideas pasan por un proceso de selección, si esta se ve con potencial el equipo procederá a crear un prototipo interno y este será validado para su futuro desarrollo. Si la propuesta sigue adelante, será asignada con un plan de ejecución y un equipo dedicado a su desarrollo. Más adelante, con el proyecto en marcha, se irá buscando la total independencia del equipo, formando una empresa autosuficiente funcionando dentro del grupo de empresas nacidas de NoSpoon.

Actualmente la empresa cuenta con numerosos proyectos en el mercado, desde aplicaciones que ofrecen servicios de tragaperras digitales a casinos (*Triple Cherry*[26]), hasta aplicaciones para ayudar al estudio de oposiciones como (*GoKoan*[27]), pasando por aplicaciones de aprendizaje para niños (*Kokoro Kids*[28]), juegos de realidad aumentada (*Play&Go*[29]) o aplicaciones sociales recopilatorias de juegos tradicionales (*Playjoy*).

[26] https://www.3cherry.com/
[27] https://www.gokoan.com/
[28] https://kokorokids.app/en/main-home/
[29] https://playgoxp.com/

Anexo 2. Aplicación *Playjoy*

En este anexo se explorará la aplicación Playjoy mediante capturas de pantalla correspondientes a la última versión publicada en la tienda, mostrando el contexto donde se implementará el juego desarrollado. Se mostrarán los elementos principales de la aplicación, comenzando por sus cuatro pestañas, su menú lateral donde acceder al perfil de usuario, la tienda incorporada en la aplicación junto a sus elementos y por último se mostrará como buscar partidas poniendo la Belote como ejemplo.

Pestañas de la aplicación

Los contenidos principales de la aplicación están organizados en cuatro pestañas como se puede observar en la Figura 43, al igual que otras aplicaciones como *Instagram*[30] o *Twitter*[31], entre las cuales se navega utilizando el menú horizontal en la parte inferior. Este menú y la barra de arriba están presentes en todas las pestañas, ofreciendo esta última el nombre de la pestaña actual y la imagen de perfil con la cual acceder al menú lateral. Estas cuatro pestañas son hogar (1), retos (2), amigos (3) y chat (4).

Hogar (1) es la ventana que se abrirá al entrar en la aplicación, mostrando en la parte superior una pancarta o *banner* mostrando diferentes ofertas al igual los juegos introducidos recientemente. Debajo de este están la lista de juegos disponibles, donde se encuentra la belote desarrollada. Además, mientras está abierta esta pestaña la barra superior mostrará las monedas disponibles para jugar.

Segundo, está retos (2) donde se muestran los retos semanales agrupados en la parte superior y los retos diarios en la parte inferior. Los retos semanales son comunes a todos los jugadores, mientras que en los diarios se asignan cuatro parejas de retos según los juegos más jugados del usuario. Estos retos una vez aportados dan recompensas como experiencia, monedas, cartones para jugar al bingo o tiradas gratis para las tragaperras virtuales o *slots*.

Después se encuentran amigos (3) y chat (4) las cuales están muy conectadas entre ellas. En la primera, se podrán ver los amigos actuales que tenemos y las peticiones de amistad pendientes, al igual que añadir a nuevos amigos introduciendo su nombre de perfil. Por último, en chat, se mostrarán los jugadores con los que recientemente ha jugado el usuario. Y en la parte inferior se podrán ver los chats con los jugadores recientes o amigos junto a los grupos de chat globales.

[30] https://www.instagram.com/
[31] https://twitter.com/

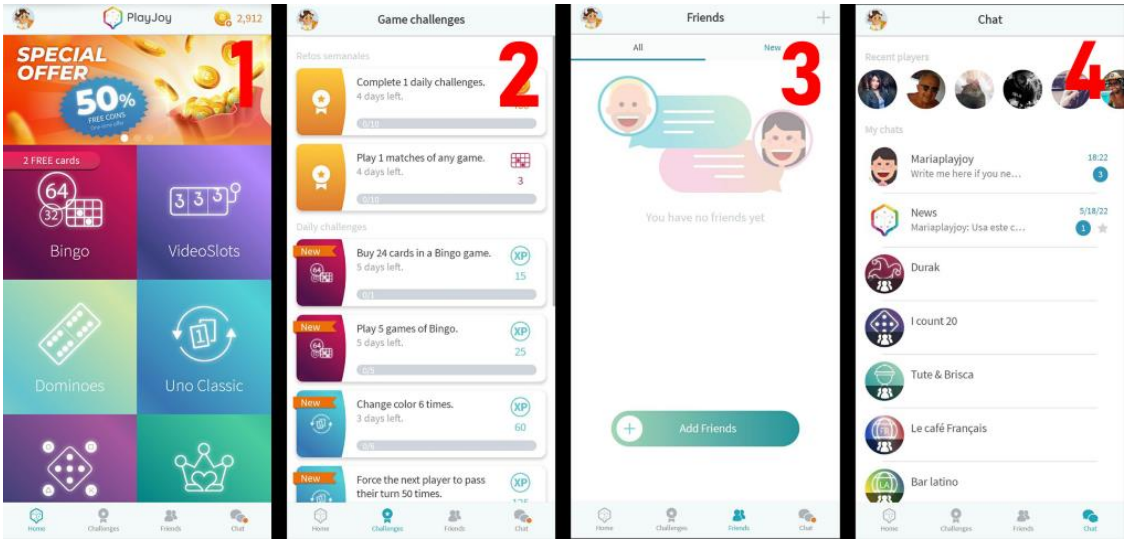

Figura 43. Capturas de pantalla mostrando las pestañas de Playjoy

Menú lateral y perfil de usuario

En la Figura 44 se muestra el menú lateral (1) y la ventana de perfil (2). Como se ha comentado antes, si se pulsa en la foto de perfil se abrirá un menú lateral (1) el cual ofrecerá al usuario acceso al perfil, tienda, subscripción, invitación a amigos, canjear códigos de ofertas, la ayuda y el botón de ajustes.

Profundizando en la pestaña de perfil (2), esta es muy similar a *Twitter*, mostrando la foto de perfil de usuario, su nivel, monedas y sus juegos más jugados junto a sus puntos. Además, ofrece la funcionalidad de compartir la aplicación con otras personas, llevando a estas a la pantalla de descarga y otro acceso más a los ajustes. La pestaña mostrada en la captura corresponde al perfil del usuario que inició sesión en la aplicación, pero esta misma estructura a excepción del botón de compartir es utilizada para mostrar los perfiles del resto de usuarios.

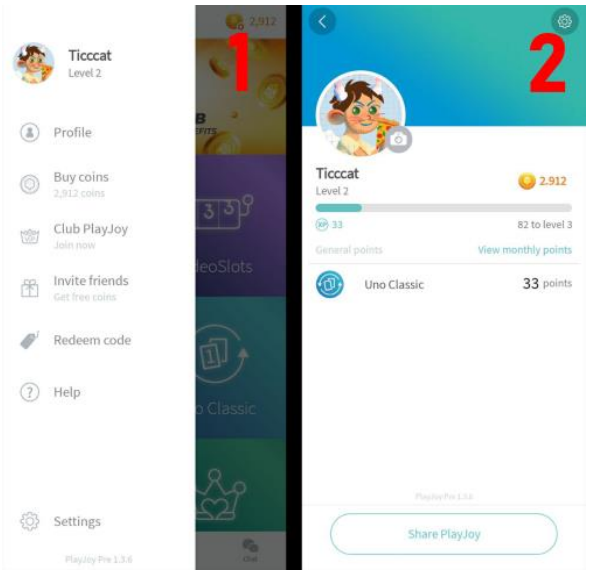

Figura 44. Capturas del menú lateral y perfil de usuario en Playjoy

Tienda y elementos ofertados en la aplicación

Pasando a la tienda de la aplicación, mostrada en la Figura 45, se pueden observar los tres principales menús de esta, por los que se puede navegar utilizando los botones de la barra superior. Primero está el menú de monedas (1), utilizado para comprar los packs de monedas, con las que apostar para jugar las partidas, comprar cartones del bingo o jugar a los *slots*.

Los otros dos menús club (2) y VIP (3) corresponden al sistema de subscripciones mensuales de la aplicación, siendo cada uno un diferente nivel con diferentes ventajas. La subscripción de nivel club, ofrece 3.000 monedas al mes, la funcionalidad de abrir chats privados ilimitados, aumentar el máximo de amigos disponibles y por último añadirá una corona al perfil al igual que una nueva imagen de fondo. El nivel VIP ofrece las mismas ventajas que el nivel inferior, cambiando las 3.000 monedas por 10.000, una lista de amigos mayor y un bono del 20% en todas las compras de monedas.

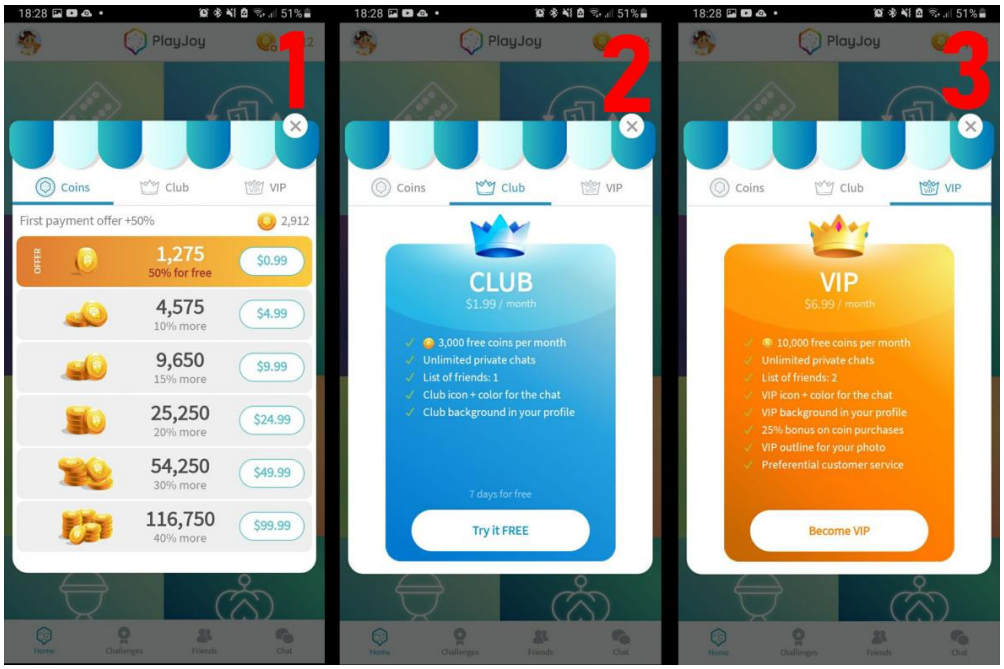

Figura 45. Capturas de la tienda de Playjoy

Proceso para buscar partidas en la aplicación

Por último, en la Figura 46 se muestra la pantalla de búsqueda de partida (2), a la cual se accede pulsando en el icono del juego al que se quiera jugar. Se ha utilizado la belote como juego de ejemplo, mostrando así su apariencia dentro de la lista junto al resto de juegos.

Esta ventana es el paso previo para entrar en una partida con otros jugadores. Permite cambiar la apuesta hecha para la partida, recargar las monedas, los juegos que se pueden completar jugando a este juego, dando acceso a los rankings y reglas del juego. En la parte superior se muestra la mesa de la partida, mostrando los perfiles de los

jugadores a medida que van entrando a la partida. En estos perfiles se podrá ver la foto, nivel y subscripción.

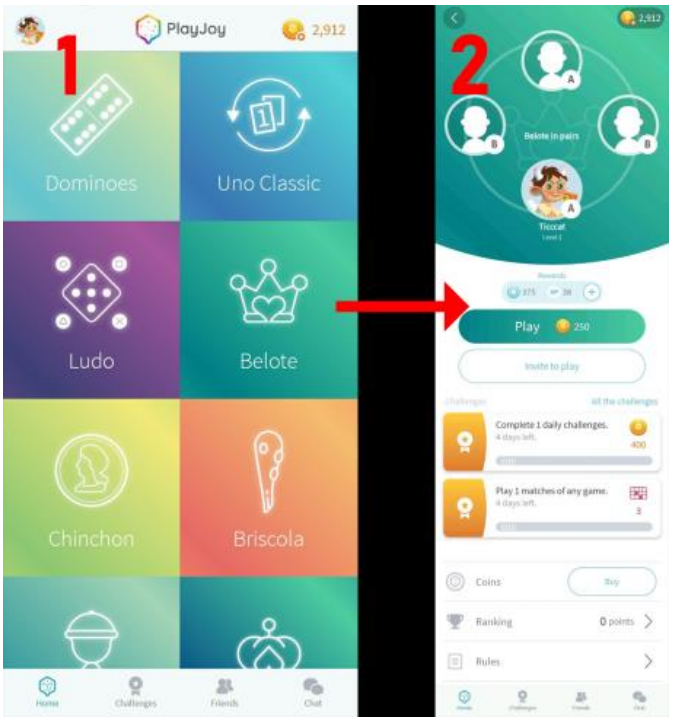

Figura 46. Capturas con el proceso para buscar partida en Playjoy

Anexo 3. Belote: Documento de diseño

Resumen

La belote se trata de un juego de cartas francés, famoso en varios países europeos. Existen diferentes modalidades del juego, siendo la principal la jugada por 4 jugadores en parejas utilizando una baraja de póker de 32 cartas (solo se utilizan las cartas A-R-D-J-10-9-8-7 de cada palo). Parecido a otros juegos como la brisca o el tute, el juego se desarrolla jugando rondas compuestas de ocho "bazas" contra la pareja contraria. Durante estas rondas los jugadores irán marcando puntos mediante dichas bazas, anuncios y otros tipos puntos especiales. La pareja que primero alcancé los 1000 puntos será la ganadora.

Jugabilidad

Las reglas del juego varían según el país o región donde se juegue. Para este anexo, se utilizarán las reglas descritas por la Federación Francesa de la belote[32].

Selección del palo de triunfo

Al principio de cada ronda, se reparten cinco cartas a cada jugador y se coloca una carta en el centro de la mesa, boca arriba. Los jugadores entonces pasarán a elegir el "palo de triunfo" (triunfo para simplificar) de la ronda. Este palo de triunfo denominará las cartas ganadoras de la ronda, junto a las cartas de más valor.

Se podrán realizar un máximo de dos vueltas para elegir el triunfo. En la primera, los jugadores tendrán la opción de "coger" el palo de triunfo marcado por la carta dejada boca arriba, teniendo que elegir a conveniencia de las primeras cinco cartas que les han sido entregadas. Si en cambio todos los jugadores deciden pasar su turno sin coger el triunfo marcado, se comenzará la segunda vuelta, donde los jugadores podrán elegir arbitrariamente el palo de triunfo según la conveniencia.

Una vez un jugador coja el palo de triunfo, la carta dejada boca arriba será entregada a este jugador junto con otras dos cartas. Al resto de jugadores les serán entregadas tres cartas, teniendo entonces un total de 8 cartas en la mano. A partir de este momento se empiezan a jugar las bazas con el triunfo como palo ganador.

Si durante ninguna de las dos rondas se coje el triunfo, se reiniciará la ronda, volviendo a repartir cinco cartas diferentes a los jugadores y comenzando con la ronda de selección de triunfo de nuevo.

[32] official_rulhttp://funwhist.ca/static/pdf/official_rules_belote.pdfes_belote.pdf(funwhist.ca)

Juego de las rondas

Una vez elegido el palo de triunfo, se empieza la fase de juego de las rondas. Esta está compuesta de ocho bazas donde los jugadores jugarán sus cartas con el objetivo de acumular puntos. La primera baza la empieza el jugador que empezó la ronda de selección de triunfo, tirando una carta a su elección, las bazas siguientes las comenzarán el ganador de la baza anterior. Las bazas las ganarán los jugadores que lancen la carta de mayor valor, priorizando las cartas del palo de triunfo.

A excepción de la primera carta tirada en cada baza, que será de libre elección, el resto de los jugadores deberán seguir unas reglas a la hora de jugar sus cartas en un orden específico. Si la primera carta no es del palo de triunfo estas son:

1. Se deberá jugar una carta del palo marcado por la primera carta lanzada en la baza, este se llama palo pedido.
2. Si el jugador no dispone de ninguna carta del palo pedido
    a. Deberá jugar una carta del palo de triunfo, teniendo que ganar a cualquier otra carta de triunfo de la mesa, a no ser que sea el compañero quien está en control de la baza
    b. Si no se dispone de una carta de triunfo mayor a la que se encuentra en la mesa, se tendrá que descartar una carta cualquiera de triunfo.
    c. Si el jugador no tiene ninguna carta de triunfo, se tendrá que descartar una carta de cualquier otro palo.

Si en cambio la primera carta lanzada es una carta de triunfo, solo se aplicarán las reglas descritas en los puntos 2.a, 2.b y 2.c.

Orden y valor de las cartas

Según si las cartas son del palo de triunfo o no, presentan dos órdenes y valores diferentes. Siendo estos:

No triunfo

Carta	Valor
A	11
10	10
R	4
D	3
V	2
9	0
8	0
7	0

Tabla 4. Orden y valor de las cartas de no triunfo

Triunfo

Carta	Valor

V	20
9	14
A	11
10	10
R	4
D	3
8	0
7	0

Tabla 5. Orden y valor de las cartas de triunfo

Belote y rebelote

Cuando un jugador posee el rey y reina del palo de triunfo, se dice que tiene el "belote", lo cual sumará 20 puntos a su pareja al final de la ronda. Esta combinación será anunciada por el jugador que la posea, cuando lance cada una de las cartas, diciendo "belote" al tirar la primera y "rebelote" al tirar la segunda. No se ha de hacer en ningún orden especifico.

Anuncios

Los anuncios son las diferentes combinaciones de cartas que los jugadores pueden tener tras el reparto. Estas combinaciones son llamadas anuncios y se declaran durante la primera baza, cada jugador tendrá que declarar en su turno si posee algún anuncio o no. Al final de la primera baza se decide quien ha sido la pareja ganadora de los anuncios, esto es cual posee el anuncio más potente. Las combinaciones posibles, de mayor a menor potencia y puntuación son:

Anuncios

Combinación	Valor
Brelán de valet	200
Brelán de 9	150
Brelán	100
Tercia más quinta	120
Quinta	100
Cuarta	50
Tercia	20

Tabla 6. Orden y valor de los anuncios.

De mayor a menor, Brelán se refiere a la combinación realizada al poseer las cuatro cartas de mismo valor, pero de diferente palo, sin tener en cuenta estas combinaciones realizadas al poseer todos los sietes u ochos. Después, como su nombre indica, la tercia,

cuarta y quinta se refieren a las combinaciones realizadas al poseer una escalera de tres, cuatro o cinco cartas del mismo palo respectivamente. En caso de tener dos combinaciones diferentes, solo se acumularán en el caso de tener tercia más quinta, de otro modo solo se elegirá la más potente.

En caso de ambos equipos poseer el mismo anuncio, se elegirá ganador el anuncio cuyas cartas sean del palo de triunfo y en caso de no ser, tengan mayor orden. Por ejemplo, una tercia compuesta de A-R-D será mayor que una tercia D-V-10. En caso de ningún anuncio ser de triunfo y ambas combinaciones tengan cartas del mismo orden, los anuncios se empatarán y ningún equipo sumará sus puntos.

Diez de ultimas y capote

La pareja que gane la última baza de cada ronda obtendrá 10 puntos extras, aumentando a 162 el número de puntos obtenidos como mínimo en cada ronda.

Si una pareja gana todas las bazas de una ronda, además de los diez de últimas, obtendrán 90 puntos extra, sumando a un total de 100 puntos extra al final de la ronda.

Conteo de puntos

Al final de las rondas, se resuelve el contrato y se suman los puntos obtenidos por cada pareja. Estos están compuestos por los puntos de bazas, anuncios, belote/rebelote y los puntos extras. Existen tres casos según la resolución del contrato:

1. Contrato completado: si la pareja de elige el palo de triunfo gana la ronda (consigue más puntos contando bazas y belote que la pareja contraria), los puntos conseguidos por las parejas serán sumados normalmente.
2. Contrato fallado: si la pareja que elige el contrato pierde la ronda, todos los puntos menos los del belote serán sumados a la pareja no electora.
3. Litigio: si ambas parejas empatan a puntos (obtienen un total de 81 puntos cada una), la pareja electora no sumará puntos esta ronda y estos se guardarán para la pareja ganadora de la siguiente ronda.

La partida la ganará la primera pareja que llegué a 1001 puntos (este número ha sido reducido a 501 puntos para la versión implementada).

Anexo 4. Elementos gráficos comunes en todos los juegos

En este anexo se repasarán los gráficos comunes en todos los juegos, los cuales se implementan de forma automática ya que se implementan a través de las clases base de los juegos. Estos gráficos comunes se tratan de los perfiles de los usuarios dentro de la partida y los gráficos relacionados con la funcionalidad del chat dentro de la partida y todas sus acciones.

Como aclaración, muchas de las plantillas del juego, como las cartas de la mano, o los puestos donde colocar las cartas en la mesa son iguales en todos los juegos, pero no comunes entre ellos. Por este motivo no han sido incluidos en este apartado, ya que requieren cambios entre ellos relacionados con las texturas de las cartas a utilizar, las cuales en el caso de la belote son únicas.

Perfiles de usuario dentro de la partida

Como se muestra en el anexo 2, los usuarios de la aplicación cuentan con su propia pantalla de perfil, donde se muestra su nombre, foto de perfil, nivel, entre otra información propia. Esta información también se quiso incorporar dentro de las partidas, permitiendo conocer al resto de jugadores y fomentando la interacción entre ellos. Esta implementación de los perfiles dentro de los juegos se muestra en la Figura 47, donde se presentan los perfiles de los otros tres jugadores en este orden: izquierda (1), arriba (2) y derecha (3). La única diferencia entre ellos es su posición relativa con su mano, compartiendo el resto de los elementos, por este motivo se analizarán estos de forma genérica.

Primero de todo, se muestra la foto de perfil del usuario junto a su nombre (desenfocados para proteger sus datos) siendo la información personal compartida, y debajo de estos está su nivel. Debajo de este, encontramos un botón de chat privado, el cual abre una pestaña de chat privado como se puede ver en la imagen 2 de la Figura 47, de esta forma se aumenta de gran manera la interacción entre los jugadores dentro de las partidas y favoreciendo también esta interacción fuera de las partidas. Por último, encima de todos los datos propios del usuario y el botón de chat, se muestran los puntos actuales de la partida que el jugador tiene.

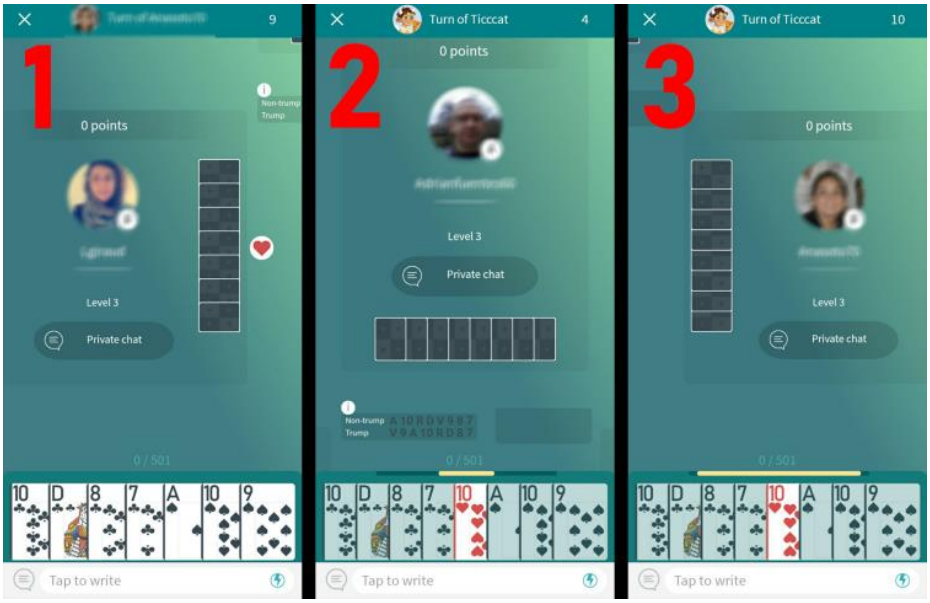

Figura 47. Capturas mostrando los perfiles de los jugadores dentro de la partida

Elementos gráficos y funcionales del chat dentro de las partidas

A continuación, se mostrarán los elementos gráficos comunes relacionados con el chat dentro de las partidas, mostrados en la Figura 48. En la primera imagen (1) se encuentra la pantalla de juego durante una partida, resaltando la parte inferior de esta, donde se encuentra la barra con la que se accederá a todos los elementos del chat dentro de partida. Este elemento cuenta con tres componentes principales, a la izquierda está el botón que abre el componente completo de chat (2); en el centro está la zona de entrada de texto, la cual permite chatear con los mensajes flotantes dentro de la partid a (3); y por último, a la derecha, está el botón de mensajes y emojis rápidos, el cual permite enviar mensajes rápidos predefinidos como mensajes flotantes (4).

Comenzando por el primer elemento, esta instancia sobre la pantalla de la partida un componente que ofrece una experiencia de chat completo, permitiendo al jugador a acceder a la mayoría de las funcionalidades del chat, al igual que cuando se encuentra fuera de las partidas. A pesar de no tener acceso a todas las acciones disponibles desde fuera de los juegos, el jugador tendrá acceso a las más relevantes durante la partida, las cuales son el chat grupal de la partida, sus propios chats privados y el chat con la atención al cliente, pudiendo reportar errores de la misma partida en la que se encuentra. Además, este componente ofrece una experiencia completa de mensajería instantánea, sin diferencia de lo que el usuario podría experimentar en cualquier otra aplicación.

Luego, el elemento central (3) permite enviar los llamados mensajes flotantes, los cuales aparecerán en la esquina superior izquierda, siendo visibles para todos los jugadores durante un corto periodo de tiempo. Además, estos mensajes se enviarán simultáneamente en el chat grupal de la partida, pudiendo acceder a ellos mediante el botón antes descrito. Finalmente, el elemento de la derecha (4) abre un dialogo con diferentes opciones de mensajes simples y rápidos de enviar, al igual que emojis

preestablecidos. Al igual que los mensajes escritos mediante el elemento central, estos se mostrarán en la partida como mensajes flotantes y serán enviados al chat grupal, pudiendo ser accedidos desde el chat completo.

Figura 48. Capturas de pantalla mostrando las funcionalidades del chat dentro de la partida

Anexo 5. Objetivos de Desarrollo Sostenible

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	**Alto**	**Medio**	**Bajo**	**No Procede**
ODS 1. **Fin de la pobreza.**				**x**
ODS 2. **Hambre cero.**				**x**
ODS 3. **Salud y bienestar.**				**x**
ODS 4. **Educación de calidad.**				**x**
ODS 5. **Igualdad de género.**			**x**	
ODS 6. **Agua limpia y saneamiento.**				**x**
ODS 7. **Energía asequible y no contaminante.**				**x**
ODS 8. **Trabajo decente y crecimiento económico.**		**x**		
ODS 9. **Industria, innovación e infraestructuras.**		**x**		
ODS 10. **Reducción de las desigualdades**.				**x**
ODS 11. **Ciudades y comunidades sostenibles**.				**x**
ODS 12. **Producción y consumo responsables.**				**x**
ODS 13. **Acción por el clima.**				**x**
ODS 14. **Vida submarina.**				**x**
ODS 15. **Vida de ecosistemas terrestres.**				**x**
ODS 16. **Paz, justicia e instituciones sólidas.**				**x**
ODS 17. **Alianzas para lograr objetivos.**				**x**

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

Como quedan marcados en la tabla, el proyecto puede relacionarse con varios de los ODS, siendo estos la igualdad de género en menor medida, trabajo decente e industria, innovación e infraestructuras.
Comenzando con el primer objetivo, la igualdad de género se encuentra presente no tanto en el proyecto específico desarrollado sino en la aplicación donde se encuentra. Dentro de esta se lucha por erradicar todos actos de misoginia intencionado, castigando a los jugadores que la llevan a cabo de las formas más duras posible.
Luego, en el caso del segundo objetivo, el trabajo decente viene proporcionado por la elevada captación de nuevos trabajadores becarios, ofreciendo una gran oportunidad de

crecimiento no solo financiero si no también personal a estos nuevos trabajadores en el mundo laboral.

Finalmente, dentro de la empresa se tiene mucha importancia a la posible innovación en las tecnologías utilizadas y posibles nuevas tecnologías a desarrollar. Este aspecto está muy fomentado en el ambiente de trabajo, ya que todos los proyectos financiados por NoSpoon han nacido de su interior, propuestos por trabajadores.