

Erica Lisbeth Poaquiza Cango

PROYECTO FINAL DESARROLLO DE SOFTWARE: KILLERGAME



**Francesc
de Borja Moll**
Centre Integrat de
Formació Professional

Tutor:

Juan Miguel Ramon Tur

CIFP FRANCESC DE BORJA MOLL

Desarrollo Multiplataforma

2022/23

Erica Lisbeth Poaquiza Cango

PROYECTO FINAL DESARROLLO DE SOFTWARE: KILLERGAME

Tutor:

Juan Miguel Ramon Tur

CIFP FRANCESC DE BORJA MOLL

Desarrollo Multiplataforma

2022/23

ÍNDICE GENERAL

ÍNDICE GENERAL	IV
1 - INTRODUCCIÓN	13
1.1 - DESCRIPCIÓN DEL PROYECTO GLOBAL	13
1.2 - PROPUESTA	13
1.3 - ESTRUCTURA DEL DOCUMENTO	14
2 - DESCRIPCIÓN DE LA ARQUITECTURA GLOBAL DEL PROYECTO	16
2.1 METODOLOGÍA	16
2.2 FASES DEL DESARROLLO DEL SOFTWARE	17
2.3 METODOLOGÍA SCRUM	18
2.4 - PLANIFICACIÓN DEL PROYECTO	20
2.5 - ANÁLISIS	20
2.5.1 - PARTES INTERESADAS	21
2.5.2 - REQUISITOS DE LA APLICACIÓN	21
2.5.2.1 - REQUISITOS FUNCIONALES	21
2.5.2.2 - REQUISITOS NO FUNCIONALES	22
2.5.2.3 - REQUISITOS DE SISTEMA	22
2.5.3 - REGLAS DEL JUEGO	22
2.5.4 - PRUEBAS DE ACEPTACIÓN DE REQUISITOS	23
2.5.4.1 - OBJETIVOS DE LAS PRUEBAS DE ACEPTACIÓN DE REQUISITOS	23
2.5.4.2 - DESCRIPCIÓN DE LAS PRUEBAS	23
2.5.5 - DISEÑO	24
2.5.5.1 - MODELO CONCEPTUAL DEL PROYECTO	24
2.5.5.2 - DIAGRAMA DE CLASES	25
2.5.5.2.1 - DIAGRAMA DE CLASES GENERAL APLICACIÓN PRINCIPAL	26
2.5.5.2.2 - DIAGRAMA DE CLASES GENERAL APLICACIÓN MÓVIL	27
2.5.5.2.3- DIAGRAMA DE CLASES GENERAL APLICACIÓN SONIDO	27
3 - PLANIFICACIÓN Y DESCRIPCIÓN DE SUBPROYECTOS	28
3.1 SUBPROYECTOS	28
3.1.1 Coordinación	28
3.1.2 Main Controller / Integrador	28
3.1.3 Visual	29
3.1.4 Comunicaciones	29
3.1.5 Aplicación móvil	29
3.1.6 Sonido	29
3.1.7 Cluster	29
3.2 Asignación de tareas / roles por subtareas y por colaborador	30
4 - DESCRIPCIÓN DETALLADA DEL SUBPROYECTO INDIVIDUAL	32

4.2 - ROL ASIGNADO	32
4.2.1 - COORDINACIÓN	32
4.2.2 - RESPONSABILIDADES ADICIONALES	33
5 - PLANIFICACIÓN DE LAS TAREAS DEL SUBPROYECTO INDIVIDUAL	34
5.1 - INTRODUCCIÓN	34
5.2 - DIAGRAMA DE GANTT	34
6 - DETALLE DE LAS DEDICACIONES Y TAREAS REALIZADAS	35
6.1 - PREPARACIÓN REPOSITORIO GITHUB	35
6.2 - DISEÑO DE LAS PANTALLAS	35
6.5 - IMPLEMENTACIÓN DE LA LÓGICA DE COMUNICACIÓN	36
6.6 - PRUEBAS Y CORRECCIONES	36
7 - DESCRIPCIÓN DEL ENTORNO TECNOLÓGICO Y HERRAMIENTAS USADAS	37
7.1 - VISUAL STUDIO CODE v1.67.1	37
7.2 - ECLIPSE IDE v2022-23 (4.23.0)	37
7.3 - ANDROID STUDIO 2021.3.1 PATCH 1	37
7.4 - INTELLIJ IDEA 2022.3.1	38
7.5 - ATLASSIAN JIRA CLOUD	38
7.6 - GITHUB	38
7.7 GIT	39
7.7.1 - ESTRUCTURA DE REPOSITORIOS	39
7.7.2 - ESTRUCTURA DE RAMAS	39
7.7.3 - ESTRUCTURA DE PERMISOS	41
8 - DESCRIPCIÓN DE LA ARQUITECTURA DEL SUBPROYECTO	42
9 - MANUALES DE USUARIO E INSTALACIÓN	45
9.1 - APLICACIÓN PRINCIPAL	45
9.2 - APLICACIÓN MÓVIL	45
9.2.1 - INSTALACIÓN	46
10 - CONCLUSIONES SOBRE EL PROYECTO GLOBAL Y SUBPROYECTO	50
10.1 - INTRODUCCIÓN	50
10.1.1 - CONCLUSIONES DEL PROYECTO GLOBAL	50
10.1.1.1 - REQUISITOS LOGRADOS	51
10.1.1.2 - EXPECTATIVAS Y MEJORAS	52
10.1.1.3 - LECCIONES APRENDIDAS	52
10.1.2 - CONCLUSIONES DEL SUBPROYECTO PROYECTO INDIVIDUAL	53
10.1.2.1 - EXPECTATIVAS Y MEJORAS	53
10.1.2.1.1 - MEJORAS EN LA PLANIFICACIÓN Y GESTIÓN	53

1 - INTRODUCCIÓN

1.1 - DESCRIPCIÓN DEL PROYECTO GLOBAL

El proyecto final para el ciclo formativo de desarrollo de aplicaciones multiplataforma se basa en desarrollar un juego que a gran escala partiendo de la base del video juego “asteroids”.

El desarrollo en lugar de centrarse en un aplicativo monousuario y mono equipo, se centra en replicar la idea del videojuego “asteroides” a gran escala, donde una aplicación se conecta con diferentes ordenadores, manejando así la pantalla de cada uno de estos equipos. Estas pantallas forman una matriz tipo “videowall” de 3 x 4 pantallas, formando así en su conjunto una pantalla gigante de juego, donde cada pantalla representa una porción del mapa y del juego en sí.

Mediante comunicaciones todos los equipos o pantallas, quedan interconectados entre sí permitiendo a los jugadores poder desplazarse por la totalidad de las pantallas e interactuar con objetos representados tanto en la misma pantalla como en cualquier otra pantalla que conforme el video wall.

Por otro lado, a fin de interactuar con la parte visual del videojuego. Se desarrolla paralelamente una aplicación móvil que sirve como mando a distancia del juego, en esta aplicación se representa por una parte, un menú del juego, donde poder realizar las configuraciones pertinentes, permitir la conexión con la matriz de pantallas y por último, una pantalla con los controles del juego para poder manejar la nave y realizar las diferentes acciones configuradas.

1.2 - PROPUESTA

A fin de llevar a cabo el proyecto, se propone lo siguiente:

El desarrollo de una aplicación que instalada en cada equipo que conforma la matriz de pantallas se encarga de las comunicaciones entre pantallas y móviles, la gestión de la lógica del juego (físicas, reglas del juego, etc...) y de visualizar la porción de juego que le corresponde según su ubicación en la matriz de pantallas.

El desarrollo de un aplicativo móvil que servirá de controlador del juego que permite la interacción del usuario con las pantallas y los controles de juego.

Por último, el desarrollo de una aplicación paralela a la aplicación principal encargada de gestionar los sonidos del juego.

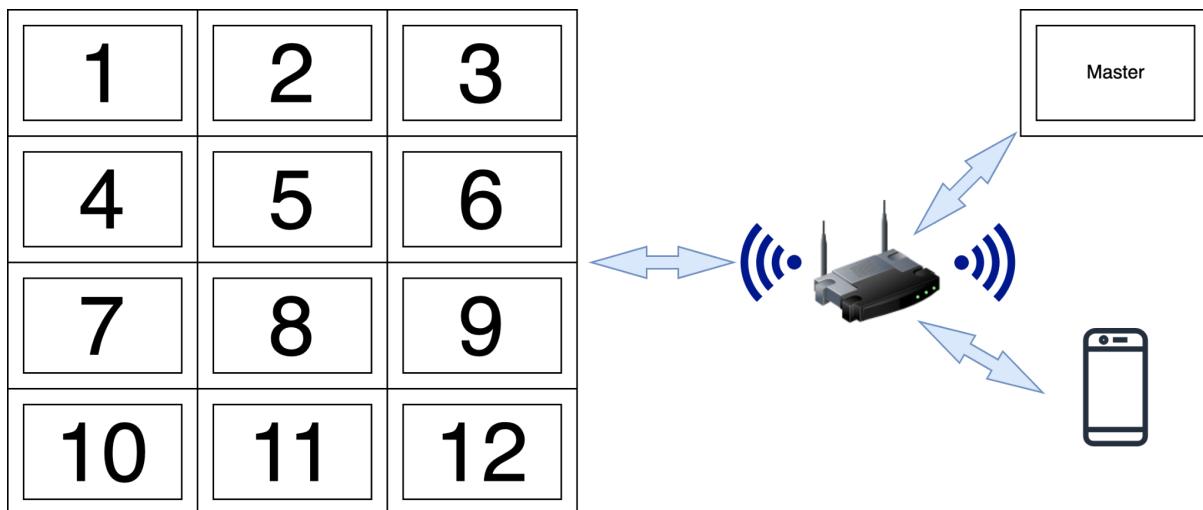


Figura 1.1 : Diagrama básico del proyecto

1.3 - ESTRUCTURA DEL DOCUMENTO

La estructura del resto de documento es la siguiente:

- Capítulo 2: se describe la metodología usada y la arquitectura global del proyecto.
- Capítulo 3: se realiza una planificación y descripción de todos los subproyectos involucrados.
- Capítulo 4: se realiza una descripción detallada del subproyecto individual.
- Capítulo 5: se describe la planificación de las tareas del subproyecto individual.
- Capítulo 6: se detallan las dedicaciones y tareas realizadas.
- Capítulo 7: se describe el entorno tecnológico usado y las herramientas usadas para el desarrollo.
- Capítulo 8: se describe la arquitectura de todos los subproyectos realizados.

- Capítulo 9: se describen todos los módulos desarrollados.
- Capítulo 10: se indican los manuales de usuario y guías de instalación.
- Capítulo 11: conclusiones finales sobre el proyecto global y subproyectos.

2 - DESCRIPCIÓN DE LA ARQUITECTURA GLOBAL DEL PROYECTO

2.1 METODOLOGÍA

La metodología de desarrollo de software se define como un marco de trabajo en el cual se planifica, se estructura y se controla todo el proceso de desarrollo de software.

El uso de una metodología de desarrollo de software resulta imprescindible para poder tener una correcta organización, poder controlar todas las fases del desarrollo y al tener una guía procedural podemos gestionar mejor los recursos disponibles, así como por consecuencia tener una mayor eficiencia del tiempo invertido, convirtiéndose en un ahorro de tiempo y costes.

Además, al seguir todo un proceso se obtiene un resultado final de mejor calidad.

Existen diferentes modalidades y cada una de ellas se organiza y funciona de manera distinta, todo ello para disminuir la tasa de fallos y conseguir un resultado final adecuado. No obstante, cada una de ellas está más indicada según el tipo de desarrollo de software que se lleve a cabo. Por eso es importante elegir la metodología adecuada según el desarrollo de cara a tener un trabajo más fluido y un resultado final acorde.

Las metodologías se dividen en dos grandes grupos:

-Metodologías tradicionales: son aquellas que tienen una estructura bien marcada, se planifica la totalidad del trabajo y una vez está completamente detallado se comienza el desarrollo.

-Metodología ágil: son aquellas que trabajan de manera más flexible y con menos documentación que las metodologías tradicionales. La metodología ágil busca proporcionar en poco tiempo pequeñas piezas de software en funcionamiento para mejorar la satisfacción del cliente. Por otro lado, al ser más flexible acepta cambios constantes que puedan surgir durante las diferentes etapas por tal de adaptarse a los cambios de manera efectiva.

2.2 FASES DEL DESARROLLO DEL SOFTWARE

1 - ESTUDIO DE VIABILIDAD

En esta primera fase del desarrollo de software se analiza el conjunto de necesidades del software, los criterios que se tienen en cuenta son tácticos, relacionados con aspectos económicos, técnicos, legales y operativos.

En esta primera fase, los resultados del estudio de viabilidad serán la base para tomar la decisión de seguir adelante o abandonar el proyecto.

2 - ANÁLISIS

En esta fase se determina cuáles serán las necesidades y los objetivos que tiene que cumplir el proyecto. Reunir todos estos requisitos que se deben cumplir para el desarrollo del software para llevar a cabo todo el proceso y cumplir con los objetivos finales.

De esta última parte se obtiene una memoria de especificación de requisitos, que contiene una especificación completa de todo lo que debe de hacer el programa.

En esta etapa es muy importante consensuar todo lo que se requiere que haga el software y será aquello que se seguirá en todas las etapas. Ya sea para un cliente final o para un desarrollo propio o interno, ya que no una vez iniciado el proceso no se podrán requerir nuevos resultados.

3 - DISEÑO

En esta fase se define la organización de la estructura y de todos los elementos necesarios para el desarrollo de software en base a las exigencias definidas en la fase anterior.

Se diseña la arquitectura del programa, así como un diseño detallado del mismo, con todos los componentes concretos e interfaces necesarias y cómo se relacionan cada uno de ellos entre sí para que funcionen de manera correcta.

Seguidamente, en esta fase se realizan los algoritmos necesarios para el cumplimiento de requisitos y también se realizan los análisis necesarios para saber que herramientas usar en la etapa de codificación.

4 - IMPLEMENTACIÓN

Todo el diseño y arquitectura provenientes de la fase anterior se tiene que codificar en el lenguaje de programación elegido.

Los componentes del software se desarrollan por separado, se buscan los errores y se realizan pruebas unitarias. Después se ensamblan para componer el programa final, este se encontraría en su versión "alfa". La primera versión del programa final para poder acceder a la siguiente etapa.

5 - VERIFICACIÓN Y ACEPTACIÓN

Partiendo de la primera versión alfa del software, en esta etapa se debe probar y ejecutar el código final para verificar que todo funciona correctamente y comparar los resultados finales con los requisitos iniciales para comprobar que cumple con todos ellos.

Una vez se ha comprobado que todo funciona correctamente y cumple con todos los requisitos el software estará listo para su entrega al cliente final y su lanzamiento.

6 - MANTENIMIENTO

Para finalizar, en esta última etapa se realiza el mantenimiento y mejora continua del software.

En caso de que se implemente alguna mejora es probable que se tenga que volver a la fase diseño para comprobar que se adapta a los cambios solicitados.

Es esencial mantener el programa constantemente actualizado para que siga siendo relevante.

2.3 METODOLOGÍA SCRUM

Durante la realización de este proyecto se hace uso de la metodología ágil scrum, que consiste en dividir el trabajo en períodos cortos llamados "sprints", generalmente de 2 a 4 semanas. El equipo selecciona las tareas que se compromete a completar durante cada sprint.

Durante el sprint, el equipo tiene reuniones diarias rápidas para mantenerse al tanto del progreso y resolver cualquier problema. Al final del sprint, se muestra el trabajo realizado al cliente y se recibe su retroalimentación.

Después de cada sprint, el equipo se reúne para analizar cómo fue el proceso y buscar formas de mejorarlo. Scrum se enfoca en la colaboración y la adaptación continua.

El Product Owner es responsable de definir y priorizar el trabajo, y el Scrum Master ayuda al equipo a seguir el proceso y a superar los obstáculos.

En resumen, Scrum es un enfoque ágil que divide el trabajo en sprints, tiene reuniones diarias y busca la mejora continua del equipo. Se centra en la colaboración, la adaptación y la entrega de valor al cliente.

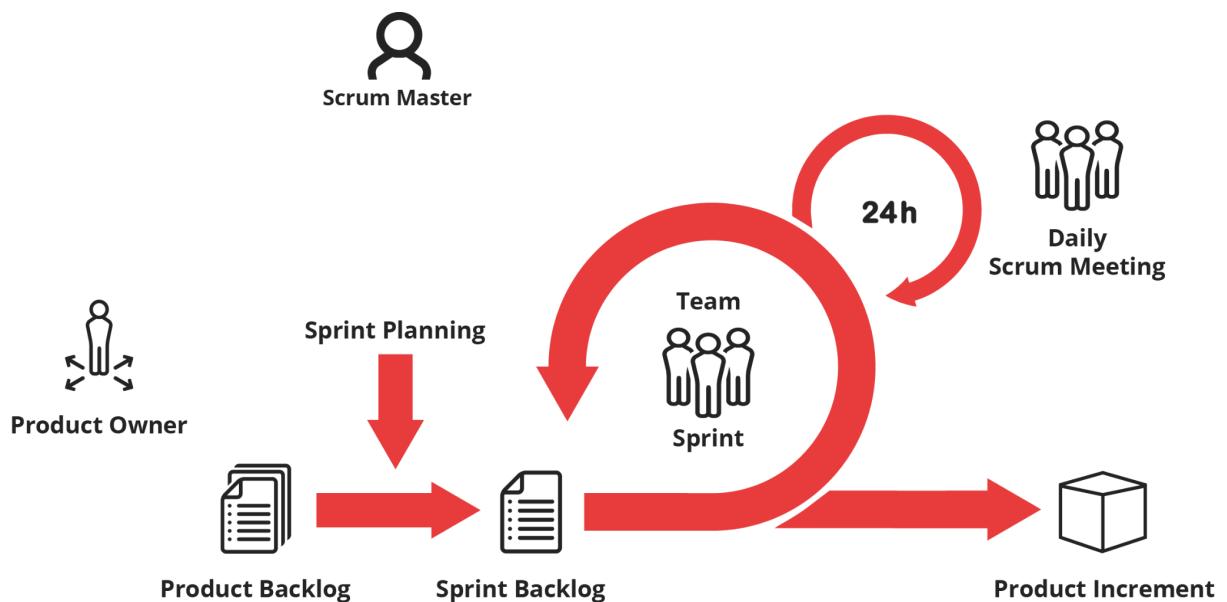


Figura 2.1: Ilustración desarrollo metodología scrum

2.4 - PLANIFICACIÓN DEL PROYECTO

Con el fin de llevar un control exhaustivo de las fases del proyecto, hitos y fechas límite, se realiza un diagrama de gantt que servirá de guía inicial sobre las fechas en las que se llevará a cabo cada fase del proyecto .

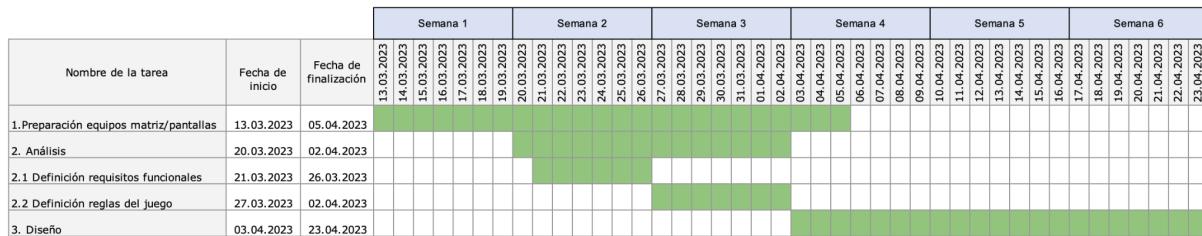


Figura 2.2: Diagrama de Gantt. Fases 1, 2 y 3.

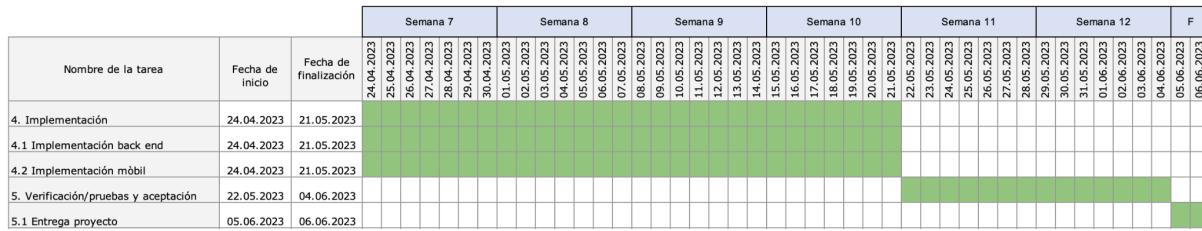


Figura 2.3: Diagrama de Gantt. Fases 4 y 5.

2.5 - ANÁLISIS

En este capítulo, se identifican las partes interesadas de la aplicación, que se corresponden a los actores que utilizarán la aplicación, además se especifica cuál es su función dentro de esta. Se definen los requisitos de la aplicación, que hacen referencia a las funcionalidades que debe tener la misma. Seguidamente se llevan a cabo las pruebas de aceptación de requisitos donde se comprueba que el sistema cumple los requisitos definidos inicialmente.

2.5.1 - PARTES INTERESADAS

Los actores que harán uso de la aplicación son :

- Cliente / Jugador : interactúa con la aplicación para ejecutar el juego.

2.5.2 - REQUISITOS DE LA APLICACIÓN

Para determinar que deben poder hacer los usuarios de la aplicación y las características de esta, hay que definir los requisitos funcionales, no funcionales y de sistema.

2.5.2.1 - REQUISITOS FUNCIONALES

Las funcionalidades que deben tener las partes interesadas son :

- CLIENTE

RF_Cliente_01: Conectarse con la matriz de pantallas

RF_Cliente_02: Editar las propiedades del juego

RF_Cliente_03: Elegir diferentes escenarios de juego

RF_Cliente_04: Elegir tipo de nave

RF_Cliente_05: Visualizar estadísticas del juego

RF_Cliente_06: Jugar al video juego

RF_Cliente_07: Debe tener modo de juego - Duelo por equipos

RF_Cliente_08: Debe tener máximo 2 equipos

RF_Cliente_09: Máximo 8 jugadores en total

RF_Cliente_10: Asignación aleatoria de equipos

RF_Cliente_11: Posibilidad de disparar

RF_Cliente_12: Límite tiempo de partida

RF_Cliente_13: Objetos estáticos interactivos

2.5.2.2 - REQUISITOS NO FUNCIONALES

Los requisitos no funcionales se corresponden a :

- RnF_01 : El sistema debe ser confiable y seguro
- RnF_02 : La aplicación debe admitir varios usuarios
- RnF_03 : La aplicación debe ofrecer transparencia del proceso a los diferentes tipos de usuarios
- RnF_04 : La aplicación tiene que emitir sonidos para interactuar con el jugador.
- RnF_05 : El desarrollo de la aplicación debe llevarse a cabo mediante el uso de Java.

2.5.2.3 - REQUISITOS DE SISTEMA

Los requisitos de sistema de la aplicación son los siguientes :

- RS_01 : La aplicación debe ser adaptable a cualquier dispositivo / plataforma

2.5.3 - REGLAS DEL JUEGO

1. Modo de juego: Duelo por equipos.
2. 2 equipos.
3. Máximo 8 jugadores (4 en cada equipo).
4. Asignación aleatoria de equipo (si los jugadores son pares se realiza asignación aleatoria de equipo, si los jugadores son impares se asigna al equipo que menos jugadores tiene).
5. No hay fuego amigo.
6. Se establece un límite de tiempo de partida, gana el equipo con más bajas y en caso de las mismas bajas se declara empate.
7. 1 arma por nave (con posibilidad de ampliación a varias armas).

8. Movimiento de las naves tipo “tanque” (rotación 360° sobre el mismo eje y avance, no hay botón de frenado).
9. Se declara un máximo de vida por nave por el valor de “100”.
10. Posibilidad de configurar el nivel de daño del arma.
11. Las naves pueden chocar pero no restan vida.
12. Existen objetos estáticos contra los que las naves pueden chocar (asteroides, planetas, etc...).
13. 3 escenarios diferentes de juego.

2.5.4 - PRUEBAS DE ACEPTACIÓN DE REQUISITOS

Las pruebas de aceptación pertenecen a una de las últimas etapas previas a la implementación del software, es muy importante realizar las pruebas correspondientes en cada etapa de desarrollo ya que un error no detectado al inicio del desarrollo del proyecto puede necesitar cincuenta veces más esfuerzos para ser solucionado que si es detectado a tiempo.

2.5.4.1 - OBJETIVOS DE LAS PRUEBAS DE ACEPTACIÓN DE REQUISITOS

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario / cliente de dicho sistema que determine su aceptación, desde el punto de vista de la funcionalidad y rendimiento.

2.5.4.2 - DESCRIPCIÓN DE LAS PRUEBAS

Las pruebas de aceptación son definidas por el usuario / cliente y preparadas por el equipo de desarrollo.

Estas pruebas van dirigidas a comprobar que el sistema cumple los requisitos de funcionamiento, recogidos en el catálogo de requisitos. Para así conseguir la aceptación final del sistema por parte del usuario / cliente

Para ello, en primer lugar se realiza una reunión inicial con el cliente para tener una primera aportación sobre los requisitos iniciales del software. Una vez obtenidos los

requisitos iniciales, estos son contrastados por el equipo de desarrollo y mediante reuniones posteriores con el cliente se hacen aproximaciones para verificar los requisitos obtenidos.

Durante la fase de pruebas de aceptación se realizan tres reuniones de aproximación en la que se han definido en cada reunión los requisitos a cumplir.

Finalmente, se realiza una reunión final con el cliente, con ya una aproximación final a los objetivos que debe realizar la aplicación en la que el cliente acepta los requisitos funcionales, no funcionales y de sistema la aplicación a desarrollar.

Este proceso de reunión-aproximación queda plasmado en el siguiente gráfico:

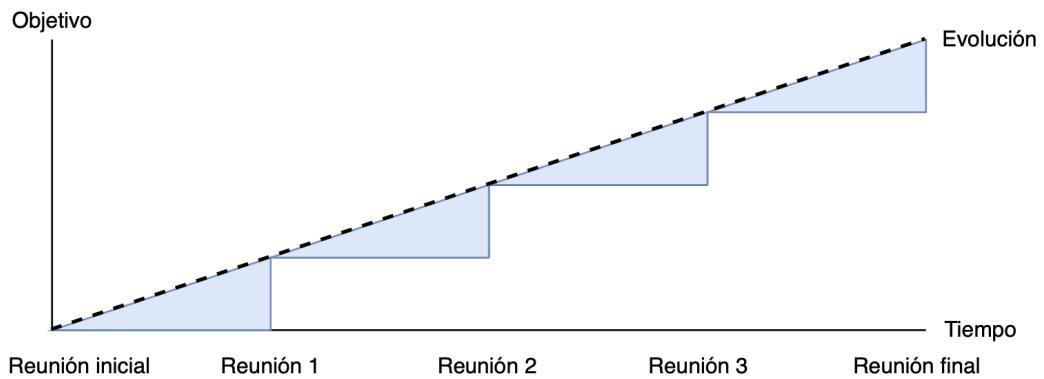


Figura 2.4 : Gráfico proceso de reunión-aproximación

2.5.5 - DISEÑO

En este apartado se procede a diseñar, de forma conceptual, la aplicación que permite jugar al juego killer game. En el capítulo apartado se han especificado los requisitos de la aplicación y tras este análisis se procede a diseñar la aplicación.

2.5.5.1 - MODELO CONCEPTUAL DEL PROYECTO

El diseño de la aplicación sigue la estructura marcada por los requisitos funcionales y no funcionales definidos en el capítulo anterior, se opta por hacer uso de 4 elementos principales que darán soporte completo a la aplicación, un conjunto de punto de acceso wifi + switch ethernet para asegurar las comunicaciones entre

equipos y móviles, un equipo master que lanza un script que al ser ejecutado en el equipo master ejecuta en todos los equipos que conforman el videowall la aplicación principal del juego y un programa java que controla el sonido del juego, una aplicación que se ejecuta en cada equipo del videowall que se encarga de la lógica del juego, de la visualización del juego y de las comunicaciones y por último, una aplicación móvil que permite interactuar con los equipos del videowall, realizar configuraciones y servir de mando de juego.

Se entiende que con estos 4 elementos se da soporte completo y son suficientes para realizar una aplicación funcional que permite cumplir con todos los requisitos funcionales y no funcionales

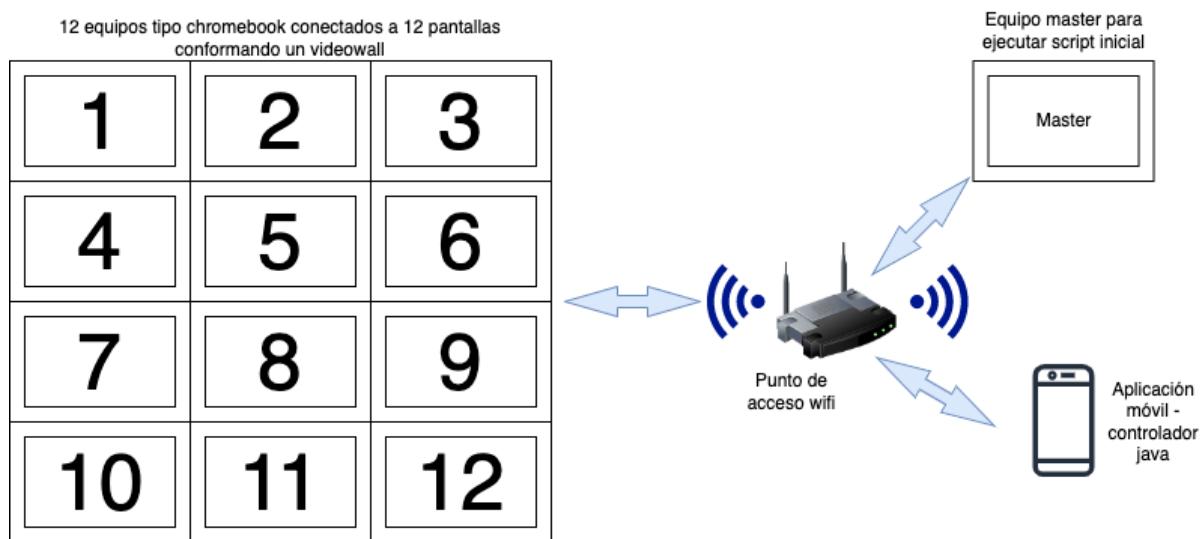


Figura 2.5 : Modelo conceptual del proyecto

2.5.5.2 - DIAGRAMA DE CLASES

En este punto se muestran los diagramas de clases de los subproyectos derivados del proyecto principal de manera esquemática.

2.5.5.2.1 - DIAGRAMA DE CLASES GENERAL APLICACIÓN PRINCIPAL

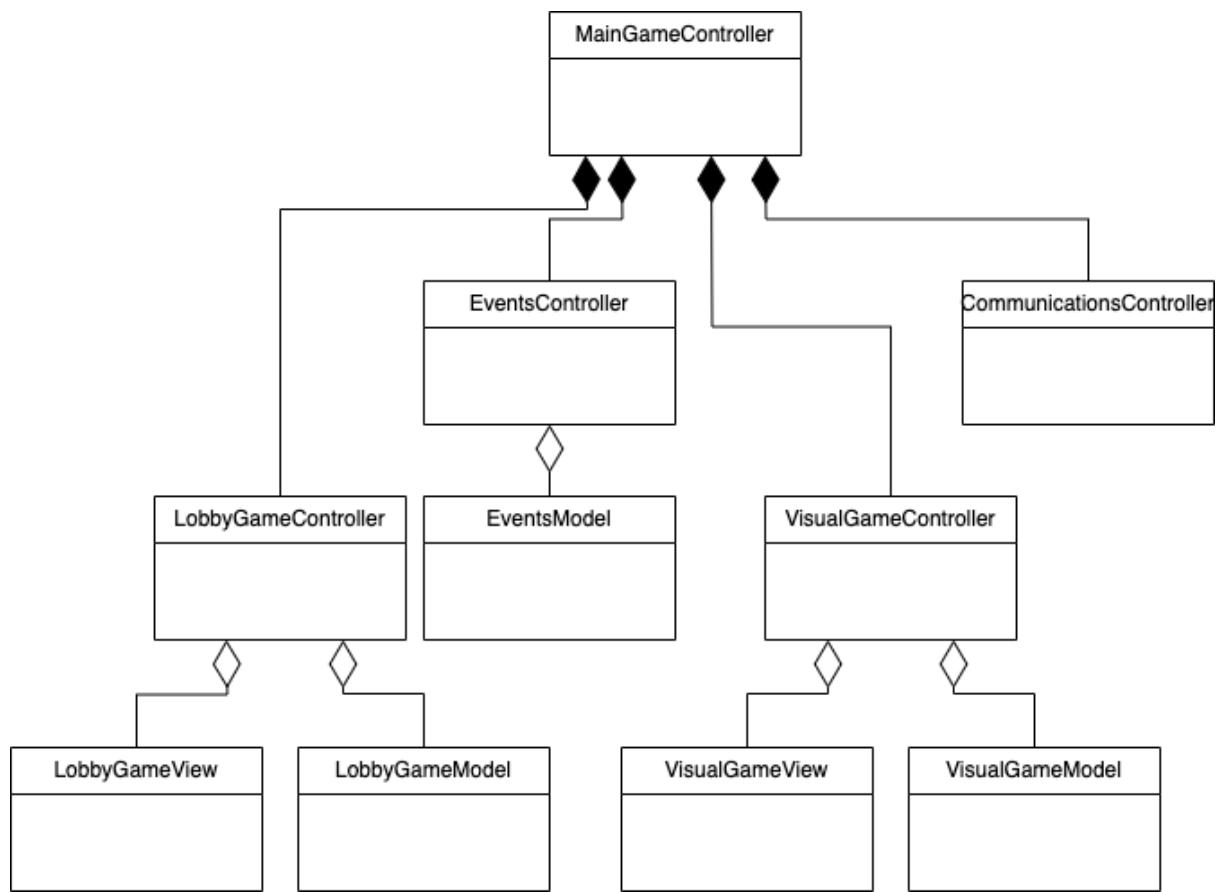


Figura 2.6 : Diagrama de clases general aplicación principal

2.5.5.2.2 - DIAGRAMA DE CLASES GENERAL APLICACIÓN MÓVIL

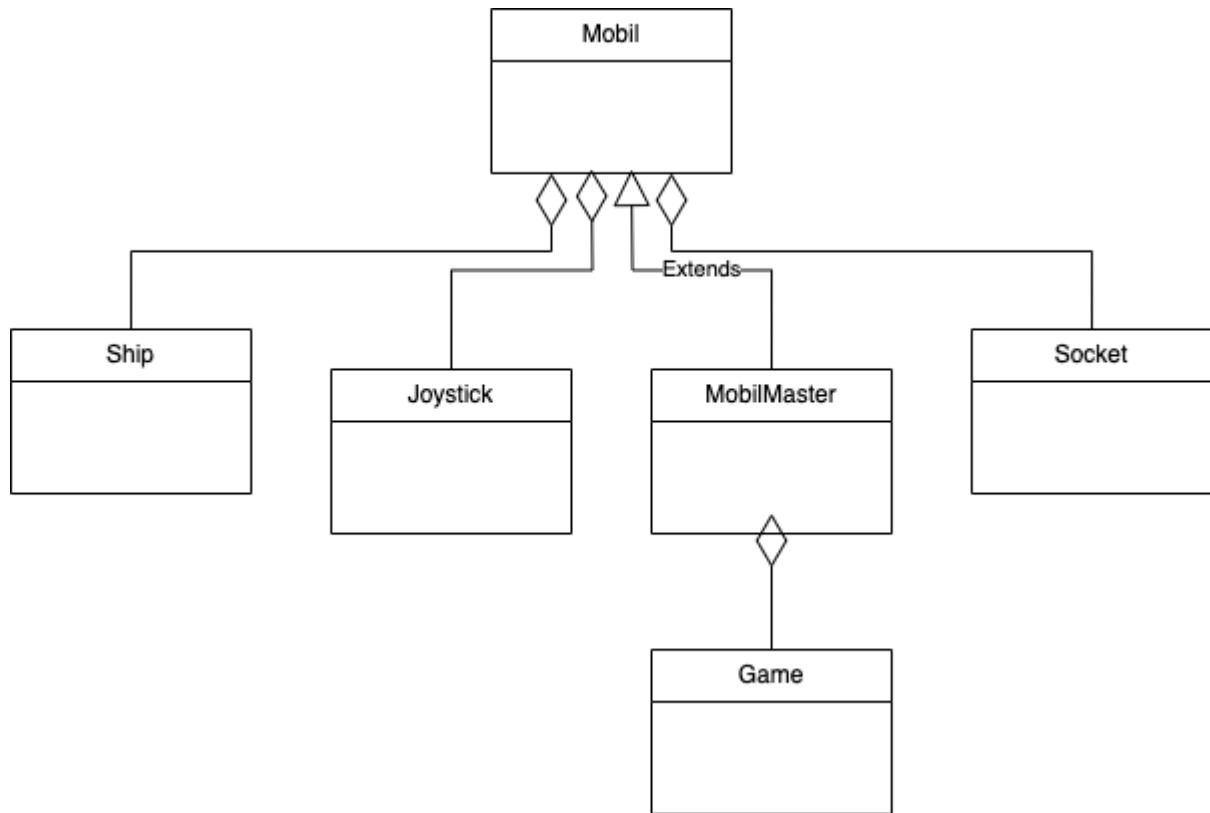


Figura 2.7 : Diagrama de clases general aplicación móvil

2.5.5.2.3- DIAGRAMA DE CLASES GENERAL APLICACIÓN SONIDO

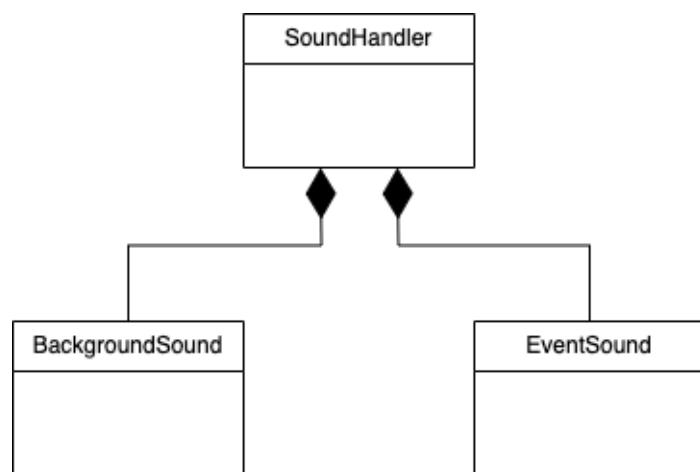


Figura 2.8 : Diagrama de clases general aplicación sonido

3 - PLANIFICACIÓN Y DESCRIPCIÓN DE SUBPROYECTOS

En este capítulo se describe la planificación y descripción de todos los subproyectos relacionados con el desarrollo global del proyecto. También por otra parte, se da forma a la estructura jerárquica a seguir a fin de definir correctamente los roles involucrados en el desarrollo así como su función en cada punto de este.

3.1 SUBPROYECTOS

A fin de cumplir con todos los objetivos y requerimientos funcionales del proyecto se han establecido 7 subproyectos.

3.1.1 Coordinación

En este proyecto se realiza la coordinación general del proyecto global, asegurando una correcta comunicación entre todos los integrantes del equipo, resolución de dudas, seguimiento de fechas límites, calidad del proyecto y la consecución del objetivo final.

Este subproyecto vendría a estar englobado en la figura de “Scrum Master” dentro del marco de la metodología ágil scrum.

Adicionalmente, cada subproyecto cuenta con un coordinador que se encarga de coordinar su subproyecto asignado y tener una correcta interlocución vertical con el “Scrum Master”.

3.1.2 Main Controller / Integrador

Este subproyecto consiste en elaborar un controlador principal que sirve de nexo de comunicación entre las distintas partes del proyecto principal, la configuración del juego, el controlador de eventos y la lógica de juego. A su vez, también se encarga de la correcta integración de las distintas partes del proyecto.

3.1.3 Visual

Este subproyecto consiste en representar de manera visual todos los elementos del juego, actúa de interfaz gráfica y gestiona la capacidad de mostrar en cada monitor el campo de juego, estadísticas, etc...

Además se encarga de la incorporación y gestión de las físicas del juego para asegurar una experiencia de juego satisfactoria.

3.1.4 Comunicaciones

Este subproyecto consiste en implementar todos los métodos necesarios para asegurar las comunicaciones tanto entre los diferentes equipos que conforman el cluster de juego principal, así como las comunicaciones entrantes y salientes de los dispositivos móviles que actuarán de mando de juego.

3.1.5 Aplicación móvil

Este subproyecto desarrolla una aplicación móvil completa para servir de interfaz entre el jugador y el cluster principal. Esta aplicación cuenta con la posibilidad de personalizar aspectos del juego así como de proporcionar al jugador un mando de juego completo.

3.1.6 Sonido

Este subproyecto gestiona mediante una aplicación externa todo lo referente a los sonidos del juego, actúa para reproducir tanto sonido ambiental del juego así como de sonidos específicos para cada evento del juego, como pueden ser disparos o explosiones.

3.1.7 Cluster

Este subproyecto se encarga del montaje físico del videowall y cluster de equipos, así como toda la arquitectura de hardware necesaria para el correcto funcionamiento del entorno de juego.

3.2 Asignación de tareas / roles por subtareas y por colaborador

En este punto se especifica mediante una tabla los diferentes subproyectos que componen el proyecto global así como los responsables asignados a cada subproyecto y tarea dentro de este subproyecto.

Coordinación / Scrum Master	Cosme Torandell
Móvil	
Responsabilidad	Nombre
Coordinador	Erica Lisbeth
Parte partida	Àngel Barceló
Parte configuración	Erica Lisbeth
Visual	
Responsabilidad	Nombre
Coordinador	Carlos Blanco
Parte Visual Chromebooks / Testing - Rendimiento	Marc Barceló
Parte Visual Chromebooks / Fondo y coordinación background	Carlos Rubio
Parte Visual Chromebooks	Robert
Parte Visual Chromebooks / Físicas	Melissa
Parte Físicas	Carlos Blanco
Parte Visual Chromebooks / Representación objetos estáticos	Marcos Nazco
Main Controller / Integrador	

Responsabilidad	Nombre
Coordinador	Marcos Lopez
Controller config game	Zhiyun
Events controller	Juan Sanchez
Controller config game	Antoni X Bascuñana

Sonido	
Responsabilidad	Nombre
Coordinador	Marcos Lopez
Sonido	Sergio Torres

Cluster	
Responsabilidad	Nombre
Coordinador	Josep Faios Suau

Comunicaciones	
Responsabilidad	Nombre
Coordinador	Miquel Andreu
Comunicaciones Chromebook	Miquel Andreu
Comunicaciones móvil	Karina

Cuadro 3.1 : Tabla de asignación de subproyectos / responsabilidad

4 - DESCRIPCIÓN DETALLADA DEL SUBPROYECTO INDIVIDUAL

En esta sección se describe en detalle el subproyecto de la aplicación móvil, la cual actúa como un mando de juego para el proyecto KillerGame. Se detallarán sus funciones y características, así como las responsabilidades del equipo encargado de su desarrollo.

4.2 - ROL ASIGNADO

En este caso, se ha asignado como proyecto individual la tarea de coordinador del proyecto móvil.

El coordinador del proyecto móvil es el responsable de liderar y supervisar el desarrollo de la aplicación móvil para el juego KillerGame. Su función incluye coordinar las tareas asignadas a cada miembro del equipo, asegurarse de que se cumplan los plazos establecidos y garantizar que la aplicación cumpla con todos los requisitos establecidos. Además, debe mantener una comunicación fluida con el resto de los coordinadores de subproyectos y el Scrum Master para asegurar la integración de la aplicación móvil en el proyecto global. En resumen, el coordinador del proyecto móvil es el responsable de garantizar que la aplicación móvil sea desarrollada de manera efectiva y cumpla con los requisitos establecidos para el éxito del proyecto KillerGame.

4.2.1 - COORDINACIÓN

Las tareas que debe realizar el coordinador del proyecto móvil son:

- Liderar y supervisar el desarrollo de la aplicación móvil para el proyecto KillerGame.
- Coordinar las tareas asignadas a cada miembro del equipo.
- Asegurarse de que se cumplan los plazos establecidos.
- Garantizar que la aplicación cumpla con todos los requisitos establecidos.

- Mantener una comunicación fluida con el resto de los coordinadores de subproyectos y el Scrum Master.
- Asegurar la integración de la aplicación móvil en el proyecto global.
- Garantizar que la aplicación móvil sea desarrollada de manera efectiva.
- Asegurarse de que la aplicación móvil cumpla con los requisitos establecidos para el éxito del proyecto KillerGame.

En resumen, el coordinador del proyecto móvil es responsable de liderar y coordinar el desarrollo efectivo de la aplicación móvil, asegurándose de que se cumplan los requisitos del proyecto y se integre adecuadamente en el proyecto global.

4.2.2 - RESPONSABILIDADES ADICIONALES

Además de las tareas de coordinación, el equipo encargado del proyecto móvil también es responsable de desarrollar la lógica de comunicación con el cluster y de diseñar las pantallas de conexión entre la aplicación móvil y los equipos del cluster.

La lógica de comunicación debe permitir que la aplicación móvil se comunique de manera efectiva con los equipos del cluster y la aplicación principal del juego, garantizando una experiencia de juego fluida y sin problemas técnicos.

Por otro lado, las pantallas de conexión deben ser diseñadas de manera intuitiva y fácil de usar, para que los jugadores puedan conectarse fácilmente a la aplicación principal del juego desde sus dispositivos móviles.

En resumen, el equipo encargado del proyecto móvil también es responsable de desarrollar la lógica de comunicación con el cluster y de diseñar las pantallas de conexión entre la aplicación móvil y los equipos del cluster, para garantizar una experiencia de juego satisfactoria para los usuarios.

5 - PLANIFICACIÓN DE LAS TAREAS DEL SUBPROYECTO INDIVIDUAL

5.1 - INTRODUCCIÓN

En este apartado se va a ilustrar en un diagrama de Gantt la planificación de las tareas del subproyecto individual, que en este caso es el proyecto móvil para el juego KillerGame. El objetivo de esta planificación es garantizar que el equipo encargado del desarrollo de la aplicación móvil cumpla con los plazos establecidos y garantice la integración de la aplicación en el proyecto global.

5.2 - DIAGRAMA DE GANTT

Tarea	Duración	Fecha de inicio	Fecha de finalización
Análisis de requisitos y preparación del proyecto de GitHub	13 días	03/03/2023	15/03/2023
Diseño de la aplicación	20 días	15/03/2022	3/04/2022
Diseño de las pantallas de conexión	27 días	04/04/2023	30/04/2023
Implementación de la lógica de comunicación	31 días	01/05/2023	31/05/2023
Pruebas y correcciones	8 días	01/06/2023	08/06/2023
Entrega final	-	12/06/2023	-

Figura 5.1 : Diagrama de Gantt subproyecto móvil.

6 - DETALLE DE LAS DEDICACIONES Y TAREAS REALIZADAS

En este apartado se detallan las dedicaciones y tareas realizadas en el subproyecto individual encargado del desarrollo de la aplicación móvil para el juego KillerGame. Se describirán en detalle las tareas realizadas por cada miembro del equipo, así como el tiempo dedicado a cada tarea y los resultados obtenidos. Este análisis permitirá evaluar el rendimiento del equipo y garantizar que se cumplan los plazos establecidos para el éxito del proyecto global.

6.1 - PREPARACIÓN REPOSITORIO GITHUB

Una de las primeras tareas que se realizó en el subproyecto de la aplicación móvil fue la preparación del repositorio de GitHub. El repositorio fue nombrado "killergamemobile" y se creó con el fin de facilitar la colaboración y el control de versiones entre los miembros del equipo encargado del desarrollo de la aplicación móvil.

Se dedicaron **13 días** para esta tarea, comenzando el **03/03/2023** y finalizando el **15/03/2023**. Durante este tiempo, se llevó a cabo el análisis de los requisitos para la preparación del repositorio y se trabajó en la creación de la estructura base del repositorio. El resultado obtenido fue un repositorio completamente funcional y preparado para el desarrollo de la aplicación móvil..

6.2 - DISEÑO DE LAS PANTALLAS

El diseño de las pantallas de la aplicación móvil fue una tarea importante en el subproyecto de la aplicación móvil. Se dedicaron **20 días** para esta tarea, comenzando el **15/03/2023** y finalizando el **03/04/2023**. Durante este tiempo, se trabajó en la creación de un diseño intuitivo y fácil de usar para los usuarios. Con la colaboración de An

Se crearon pantallas para la configuración del juego, la conexión con los equipos del cluster, la elección de personajes y la visualización de estadísticas de juego. El resultado obtenido fue un diseño de pantallas completo y coherente con la imagen general del proyecto.

6.5 - IMPLEMENTACIÓN DE LA LÓGICA DE COMUNICACIÓN

La implementación de la lógica de comunicación fue una tarea crítica en el subproyecto de la aplicación móvil. Se dedicaron **31 días** para esta tarea, comenzando el **01/05/2023** y finalizando el **31/05/2023**. Durante este tiempo, se trabajó en la creación de una lógica de comunicación efectiva entre los equipos del cluster y la aplicación principal del juego.

El resultado obtenido fue una lógica de comunicación efectiva que garantiza una experiencia de juego fluida y sin problemas técnicos.

La implementación de la librería de comunicaciones fue de Karina.

6.6 - PRUEBAS Y CORRECCIONES

La realización de pruebas y correcciones fue una tarea importante en el subproyecto de la aplicación móvil. Se dedicaron **8 días** para esta tarea, comenzando el **01/06/2023** y finalizando el **08/06/2023**. Durante este tiempo, se realizaron pruebas exhaustivas para asegurar que la aplicación móvil cumple con todos los requisitos establecidos y se realizaron correcciones necesarias para garantizar su correcto funcionamiento.

El resultado obtenido fue una aplicación móvil completamente probada y sin errores.

7 - DESCRIPCIÓN DEL ENTORNO TECNOLÓGICO Y HERRAMIENTAS USADAS

En este apartado se describen todas las herramientas y tecnologías utilizadas en la fase de implementación.

7.1 - VISUAL STUDIO CODE v1.67.1

Visual Studio Code es un editor de código fuente desarrollado por Microsoft, compatible con Windows, Linux, macOS.

Incluye soporte para depuración, resaltado de sintaxis, finalización inteligente de código (texto predictivo) y refactorización de código.

La gran ventaja de este editor de código es que es personalizable por los usuarios y compatible con gran cantidad de extensiones, esto permite que sea extremadamente versátil. Gracias a las extensiones es compatible con un gran número de lenguajes.

7.2 - ECLIPSE IDE v2022-23 (4.23.0)

Eclipse es un entorno de desarrollo de software multi-plataforma y el más usado para el desarrollo en Java.

Es de código abierto y gratuito.

Eclipse permite extender sus funciones mediante el desarrollo de plugins además de proporcionar herramientas para la gestión de espacios de trabajo, escribir, desplegar, ejecutar y depurar aplicaciones.

7.3 - ANDROID STUDIO 2021.3.1 PATCH 1

Android Studio es un entorno de desarrollo integrado (IDE) utilizado para crear aplicaciones móviles en la plataforma Android. Es la herramienta principal para los desarrolladores de Android, ya que proporciona una interfaz completa y eficiente para escribir, depurar y probar aplicaciones.

Android Studio está basado en el popular IDE IntelliJ IDEA de JetBrains y está respaldado por Google. Proporciona una amplia gama de características y herramientas que facilitan el desarrollo de aplicaciones para dispositivos Android.

7.4 - INTELLIJ IDEA 2022.3.1

IntelliJ IDEA es un entorno de desarrollo integrado (IDE) creado por JetBrains. Es una potente herramienta diseñada específicamente para el desarrollo de software en diferentes lenguajes de programación, como Java, Kotlin, Groovy, Scala y más.

IntelliJ IDEA ofrece una amplia gama de características y funcionalidades que aumentan la productividad de los desarrolladores.

7.5 - ATLASSIAN JIRA CLOUD

Atlassian Jira es una plataforma de gestión de proyectos y seguimiento de problemas diseñada para equipos de desarrollo de software. Proporciona a los equipos una forma flexible de planificar, rastrear y colaborar en el desarrollo de software, así como en la gestión de proyectos en general.

Jira se centra en la metodología ágil y utiliza tableros Kanban y Scrum para facilitar la organización y el seguimiento de tareas. Los equipos pueden crear proyectos, definir tareas y asignarlas a miembros del equipo, establecer fechas límite y prioridades, y hacer un seguimiento del progreso en tiempo real.

7.6 - GITHUB

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git.

Se utiliza principalmente para la creación de código fuente de programas informáticos.

La plataforma está creada para que los desarrolladores suban el código de sus aplicaciones y herramientas, y que como usuario puedas descargarla la aplicación y

también entrar en el perfil para leer sobre dicha aplicación o colaborar en su desarrollo.

Mediante un sistema de gestión de versiones los desarrolladores pueden administrar su proyecto, ordenando el código de cada una de las nuevas versiones que sacan de las aplicaciones para evitar confusiones. Así, al tener copias de cada una de las versiones de su aplicación, no se perderán los estados anteriores cuando se vaya a actualizar.

7.7 GIT

A fin de obtener una estructura lógica de colaboración mediante GitHub se ha creado una organización llamada “killergameorg” en la que dentro de ella, se han creado 3 repositorios independientes.

7.7.1 - ESTRUCTURA DE REPOSITORIOS

Cada uno de estos repositorios atiende a un subproyecto del proyecto global y contiene el código fuente de las aplicaciones desarrolladas.

La estructura es la siguiente:

- KillerGame: En el siguiente [enlace](#) se puede acceder a este repositorio de GitHub donde está alojado el subproyecto que se encarga de la lógica del juego, de la visualización y de las comunicaciones del proyecto.
- KillerGameMobile: En el siguiente [enlace](#) se puede acceder a este repositorio de GitHub donde está alojado el subproyecto que se encarga del aplicativo móvil android que sirve de mando a distancia para interactuar con el juego.
- KillerGameSound: En el siguiente [enlace](#) se puede acceder a este repositorio de GitHub donde está alojado el subproyecto que se encarga del aplicativo que controla los sonidos del juego.

7.7.2 - ESTRUCTURA DE RAMAS

Adicionalmente, en cada uno de los repositorios se ha creado un esquema de ramas para permitir un desarrollo colaborativo ordenado y seguro.

La estructura de ramas es la siguiente:

- Main: Esta es la rama principal de la aplicación, esta rama contiene el código funcional y final de la aplicación.
- Rama subproyecto: Estas ramas son las dedicadas al desarrollo de cada subproyecto antes de ser finalmente subidas y juntadas con la rama principal.

En este caso, existe una rama para cada subproyecto, por lo tanto la estructura de ramas de subproyecto es la siguiente:

- Visual
 - Communications
 - Lobby
 - Events
 - Main-controller
- Dev: Dentro de cada subrama de cada subproyecto se crea esta rama a fin de que cada colaborador pueda realizar su desarrollo sin interferir con la subrama de cada subproyecto. Una vez el colaborador ha finalizado una parte del código libre de errores y funcional se subirá el código a la rama principal de su subproyecto.

Mediante el uso de conventional commits y versionado semántico se han registrado cada una de las partes y actualizaciones de las aplicaciones hasta llegar al desarrollo completo de la aplicación. En la siguiente ilustración se refleja la estructura de ramas del subproyecto principal con la rama visual.

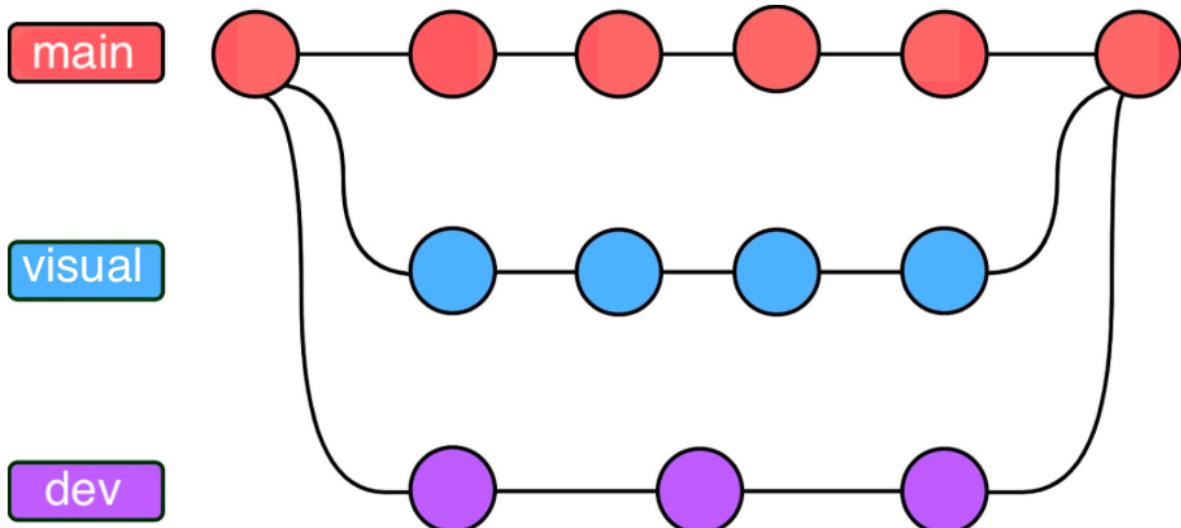


Figura 7.1 : Diagrama de ramas visual

7.7.3 - ESTRUCTURA DE PERMISOS

Con el fin de obtener una seguridad sobre los commits realizados en los repositorios se ha definido una estructura de permisos adecuada.

Para ello, se han creado 3 grupos de colaboradores clave con diferentes roles, cada uno de ellos tiene los permisos asignados para poder realizar un tipo de acciones predefinidas.

- Directive: Es el rol jerárquico más elevado, tiene los permisos totales de control sobre el repositorio y es el único que puede autorizar subidas de código a la rama principal. Este rol se le otorga al coordinador del equipo o scrum master.
- Controllers: Este rol tiene poder dentro de su rama de desarrollo de subproyecto, es la persona que autoriza a los colaboradores con rol “Devs” a subir código a la rama principal de su subproyecto. Este rol se le otorga a los coordinadores de cada subproyecto.
- Devs: Este rol tiene el poder usar la rama “dev” y poder subir el código en ella. No tiene permisos para subir código a ninguna otra rama y por tanto, necesita de aprobación de su coordinador asignado. Este rol se le otorga a los desarrolladores de cada subproyecto.

8 - DESCRIPCIÓN DE LA ARQUITECTURA DEL SUBPROYECTO

En este apartado se describe la arquitectura del subproyecto de KillerGameMobile. Se explicará la arquitectura general de la aplicación móvil así como sus módulos que la componen. La arquitectura está dividida en paquetes en los que se dividen las diferentes secciones del proyecto:

- Controller: Aquí se implementará el controlador de sonido que se encargará de administrar el sonido de la aplicación con su propia lógica..
- Services: Aquí se implementará el servicio de sonido que se encargará la implementación de sonido, este contará con una clase que extenderá de Service para que pueda funcional. Y después declarar el servicio en Manifest.xml.
- DTO: Aquí se almacenan los singletons que se encargan de almacenar los datos de la aplicación. Tales como la información de la partida o el estado en el que está la aplicación.
- Communications: En este paquete se almacena toda la librería de comunicaciones implementada por Karina.
- Clients: En esta carpeta se almacena la implementación de la librería de comunicaciones, con la lógica del juego KillerGame, con la gestión de paquetes, el proceso de identificación de dispositivos, etc.
- KillerGameJoystick: Aquí se almacenan las activitys de la aplicación Android.

Este es el diagrama de clases general del proyecto KillerGameMobile:

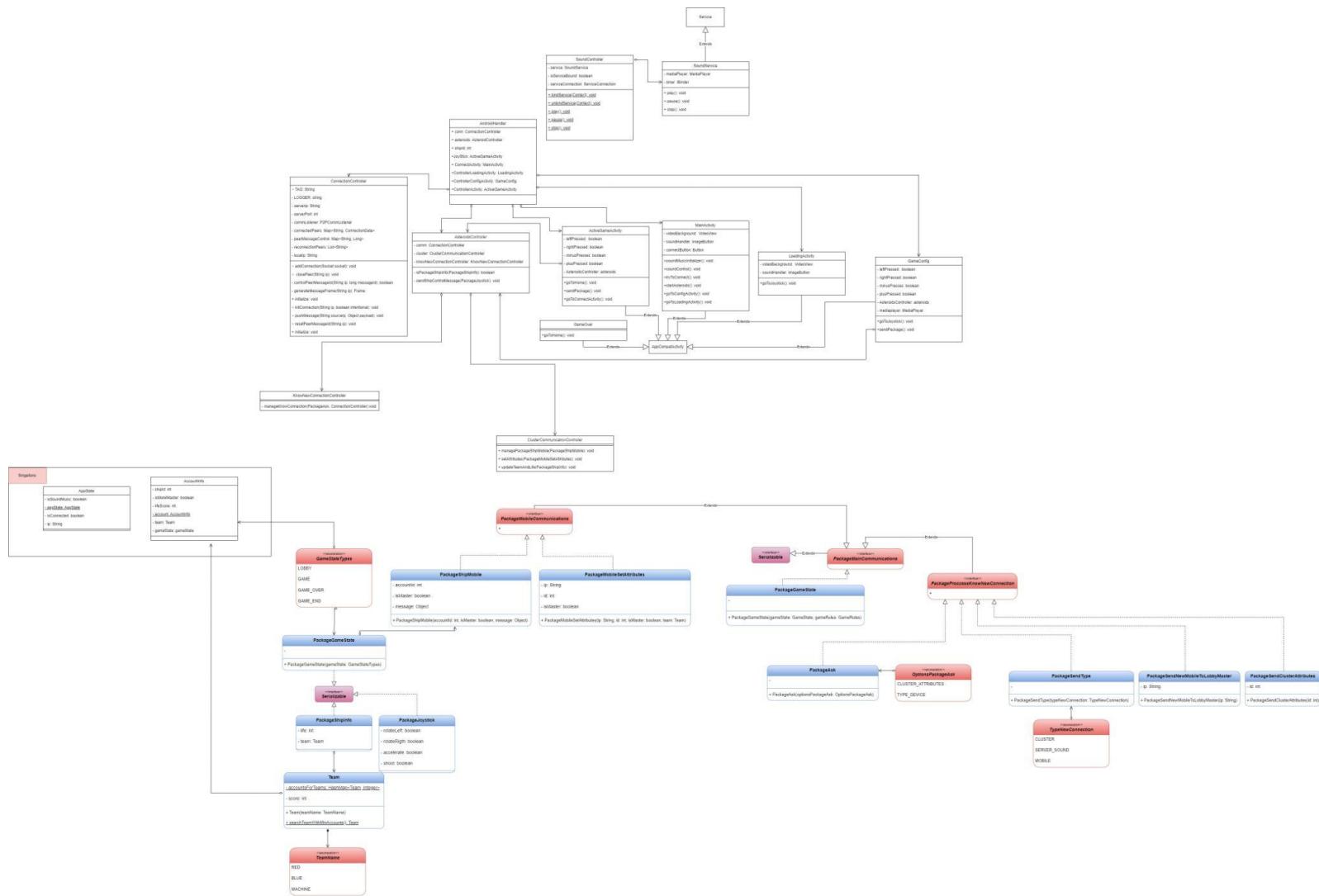


Figura 8.1 : Diagrama de clases ([link](#))

El objetivo que se ha buscado con esta estructura de directorios, es que la aplicación sea lo más modular posible, para que llegado a un punto pueda ser escalable fácilmente con muy pocos costes. También se ha tenido en cuenta la organización de paquetes que se enviarán al cluster que, utilizando la programación orientada a objetos, se ha logrado una jerarquía común para ambas aplicaciones.

9 - MANUALES DE USUARIO E INSTALACIÓN

En este módulo se describe como el usuario final puede ejecutar la aplicación final, tanto la aplicación principal que se ejecuta en el cluster de pantallas como la aplicación móvil.

9.1 - APLICACIÓN PRINCIPAL

Para ejecutar la aplicación principal se tienen que seguir los siguientes pasos:

1. Tal como se indica en el diagrama básico del proyecto, el cluster se compone de 12 equipos conectados a 12 pantallas y un equipo master. En este equipo se carga el archivo P2P.jar del juego en la siguiente carpeta “/home/dam/mountedFolder/P2P.jar”
2. Seguidamente, se ha elaborado un script (script.sh), que una vez ejecutado entra en comunicación con los 12 equipos del cluster y ejecuta simultáneamente la aplicación, simplificando así el procedimiento de ejecución.

```
#!/usr/bin/env sh

# Montar carpeta compartida
sudo mount -t cifs -o username=anonymous,password= //192.168.1.50/public /home/dam/mountedFolder/

# Seleccionar monitor
export DISPLAY=:0

# Configurar resolucion de monitor
echo "--DELMODE"
xrandr --delmode VGA1 1024x768
echo "--NEWMODE"
xrandr --newmode "1024x768" 85.25 1368 1440 1576 1784 768 771 781 798 -sync
echo "--ADDMODE"
xrandr --addmode VGA1 1024x768
echo "--OUTPUT"
xrandr --output VGA1 --mode 1024x768

# Ejecutar JAR
cd /home/dam/java/
java -jar /home/dam/mountedFolder/P2P.jar
```

Figura 10.1 : Código script.sh

9.2 - APLICACIÓN MÓVIL

Para ejecutar la aplicación móvil se tienen que seguir los siguientes pasos:

9.2.1 - INSTALACIÓN

En primer lugar, hay que preparar el dispositivo móvil android para permitir que se pueda conectar con Android Studio e instalar la aplicación directamente desde ahí.

Para ello, tenemos que habilitar el modo depuración USB de las opciones de desarrollador del móvil android. Los pasos a seguir son los siguientes:

1. En el menú de ajustes e información del terminal, buscar la opción “Número de compilación” y hacer click en esa opción 7 veces hasta que aparezca el mensaje “¡Ahora están activadas las opciones para desarrolladores!”

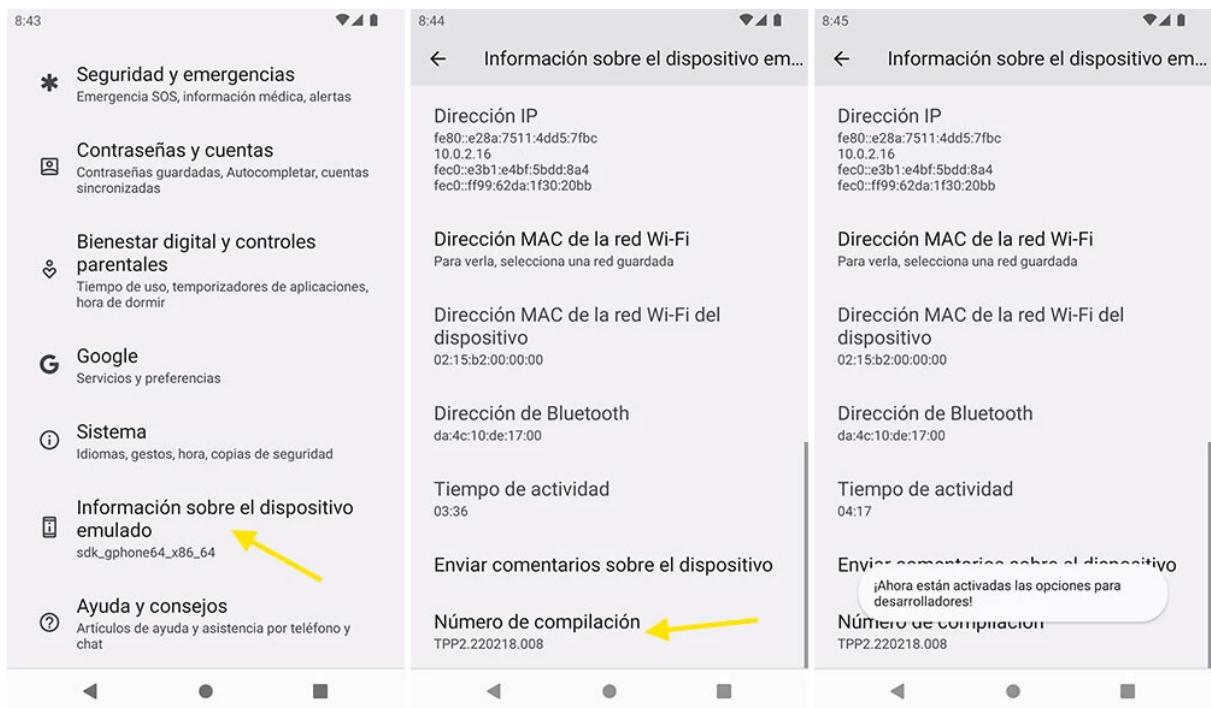


Figura 10.2 - Activación opciones desarrolladores

2. Una vez activado el menú de desarrollador, activar la opción de “Depuración por USB”.

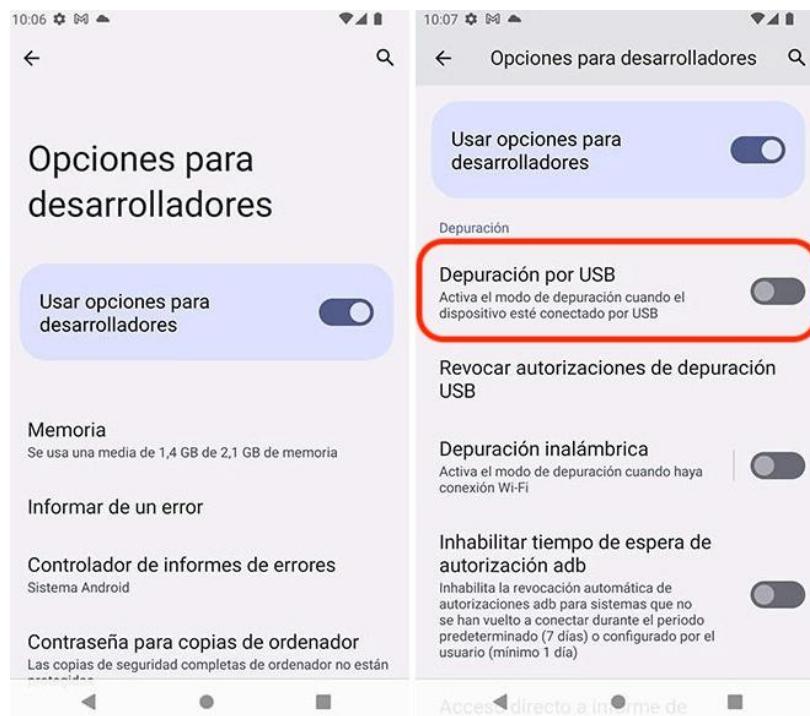


Figura 10.3 - Activación depuración por USB

Estos dos pasos anteriores tienen que realizarse la primera vez que conectamos un dispositivo móvil android a Android Studio, una vez realizados estos dos pasos ya no es necesario volver a realizarlos.

3. Seguidamente, descargamos la aplicación móvil desde el repositorio de GitHub desde el siguiente [enlace](#).
4. Importamos el proyecto en Android Studio, para ello nos dirigimos a la opción “File” -> “New” -> “Import Project”. Seguidamente seleccionamos el proyecto descargado.

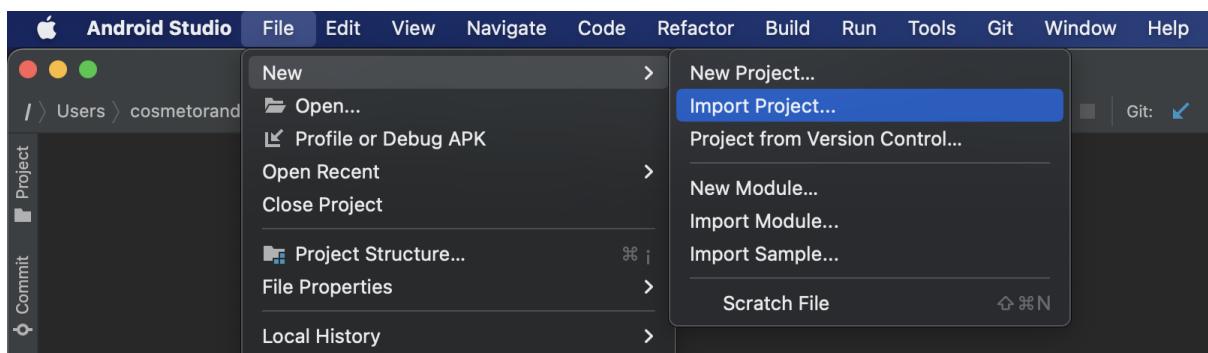


Figura 10.4 - Importar proyecto android 1

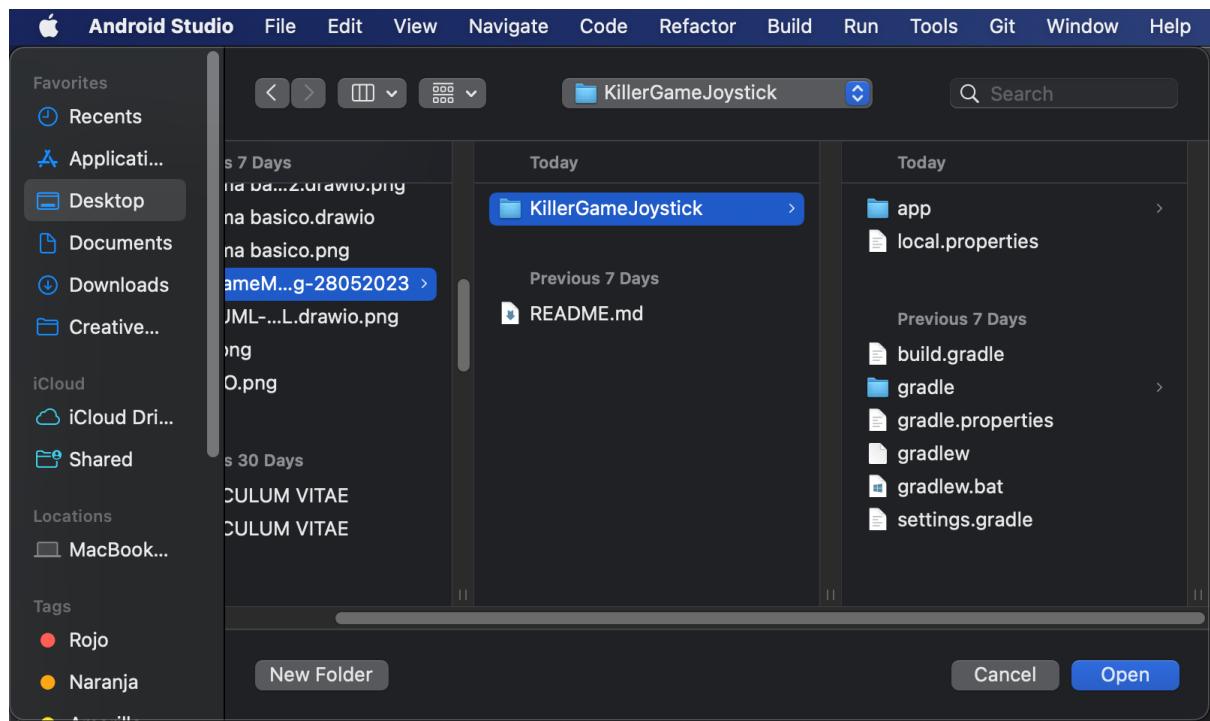


Figura 10.5 - Importar proyecto android 2

5. Una vez Importado el proyecto, conectamos el dispositivo android al ordenador, nos aseguramos de que el equipo reconoce el dispositivo correctamente y hacemos click sobre el botón “Play”.

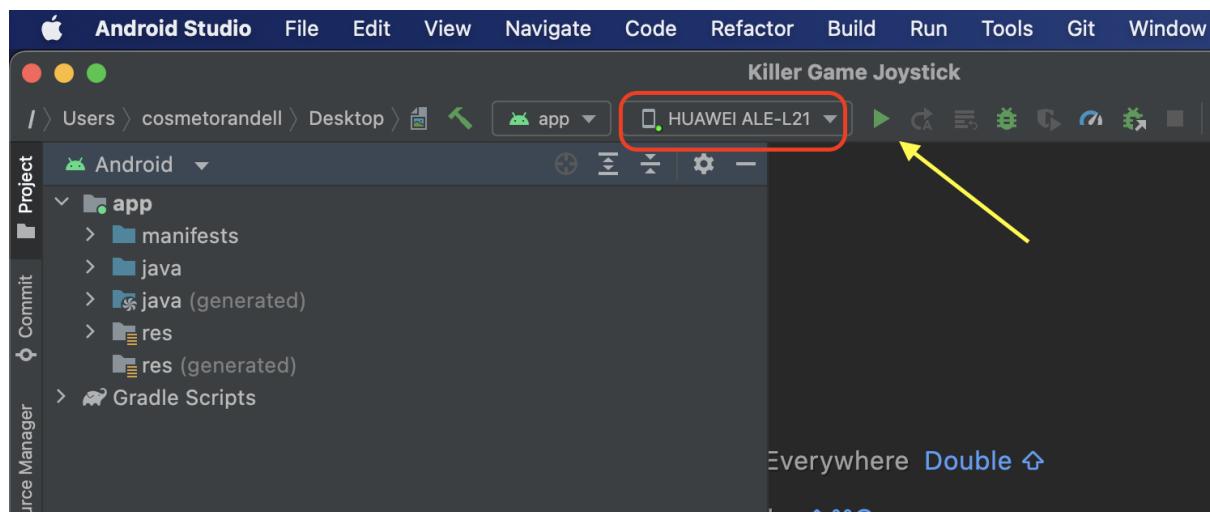


Figura 10.6 - Instalación aplicación en dispositivo

6. Iniciamos la aplicación presionando el icono de la aplicación.

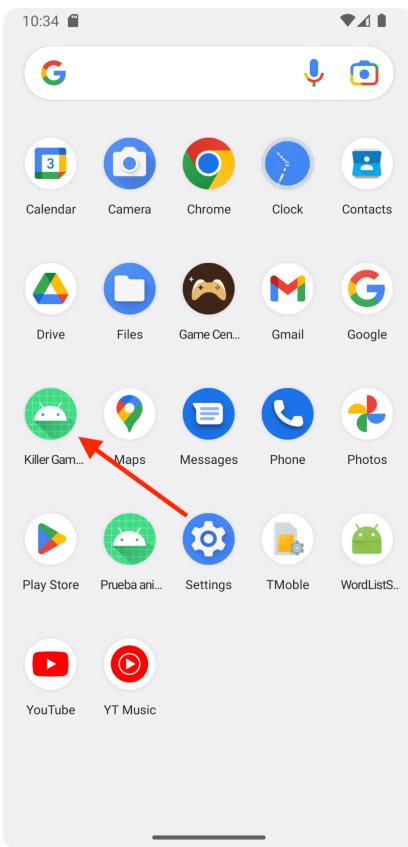


Figura 10.7 - Inicio de la aplicación móvil

7. Una vez ejecutada la aplicación, presionamos el botón “Press to connect”

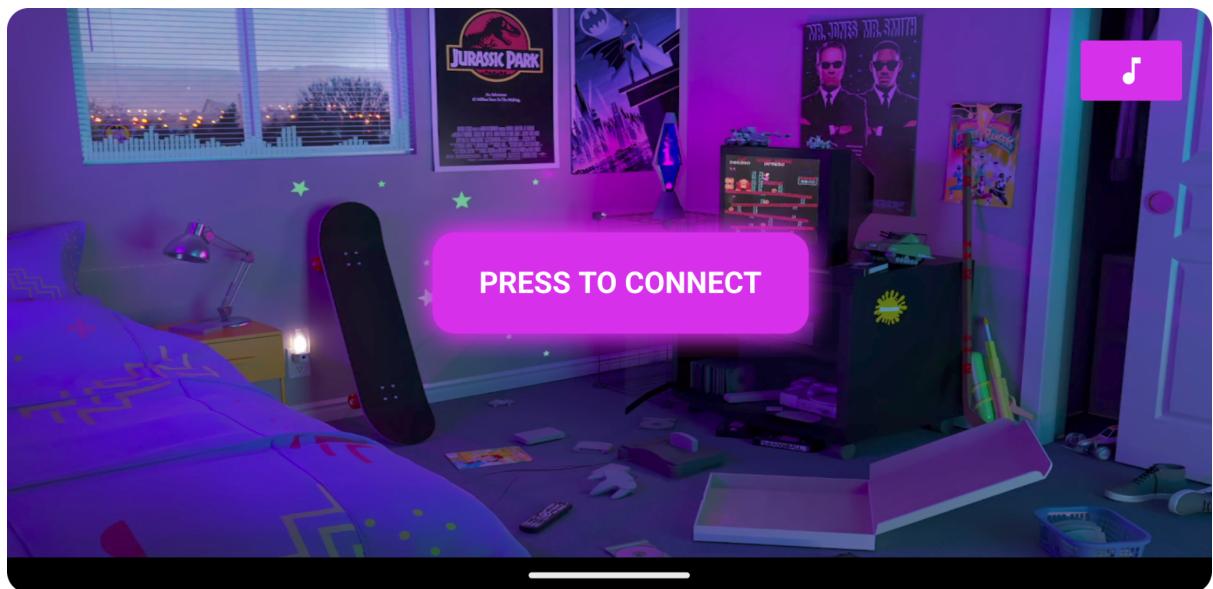


Figura 10.8 - Pantalla inicial del mando de juego

10 - CONCLUSIONES SOBRE EL PROYECTO GLOBAL Y SUBPROYECTO

10.1 - INTRODUCCIÓN

En este documento se ha presentado el proyecto KillerGame, que incluye tanto la aplicación principal para el juego como la aplicación móvil complementaria. Se ha explicado detalladamente la arquitectura y los módulos del subproyecto KillerGameMobil. Además, se ha proporcionado información sobre los manuales de usuario e instalación para ambas aplicaciones.

El desarrollo de la aplicación, con una vista en global, se ha llevado con percances mínimos. Con la ayuda en el desarrollo de Ángel Barceló y Karina resolviendo todas las dudas sobre la librería de comunicaciones el tiempo de desarrollo se ha hecho bastante ameno.

10.1.1 - CONCLUSIONES DEL PROYECTO GLOBAL

En este punto, se aborda desde una perspectiva global las conclusiones de la totalidad del proyecto y se analizan los objetivos logrados contrastados con las expectativas iniciales. Es importante destacar que, a pesar de los desafíos técnicos y organizativos, se lograron muchos resultados positivos en el proyecto. Por ejemplo, se mejoró significativamente la eficiencia en el uso de recursos, se redujeron los tiempos de producción y se mejoró la calidad del producto final. Además, se experimentó con nuevas técnicas y herramientas que pueden ser útiles en futuros proyectos. Sin embargo, también es importante reconocer que hubo dificultades y desafíos en el proceso. Estos incluyen problemas de comunicación, falta de recursos y obstáculos imprevistos en el camino. En general, el proyecto fue una experiencia valiosa que permitió aprender y mejorar en muchos aspectos.

10.1.1.1 - REQUISITOS LOGRADOS

ID	Requisito	Estado
RF01	Conectarse con la matriz de pantallas	X
RF02	Editar las propiedades del juego	X
RF03	Elegir diferentes escenarios de juego	
RF04	Elegir tipo de nave	
RF05	Visualizar estadísticas del juego	
RF06	Jugar al video juego	X
RF07	Debe tener modo de juego - Duelo por equipos	
RF08	Debe tener máximo 2 equipos	
RF09	Máximo 8 jugadores en total	
RF10	Asignación aleatoria de equipos	
RF11	Posibilidad de disparar	
RF12	Límite tiempo de partida	
RF13	Objetos estáticos interactuables	
RNF01	El sistema debe ser confiable y seguro	X
RNF02	La aplicación debe admitir varios usuarios	X
RNF03	La aplicación debe ofrecer transparencia del proceso a los diferentes tipos de usuarios	X
RNF04	La aplicación tiene que emitir sonidos para interactuar con el jugador	X

RNF05	El desarrollo de la aplicación debe llevarse a cabo mediante el uso de Java	X
RS01	La aplicación debe ser adaptable a cualquier dispositivo / plataforma	X

Cuadro 11.1: Matriz requisitos logrados

10.1.1.2 - EXPECTATIVAS Y MEJORAS

En términos generales, el proyecto no ha cumplido con las expectativas del cliente, dado que durante la fase de análisis se especificaron 13 requisitos funcionales, 5 no funcionales y 1 requisito de sistema resultando en un total de 19 requisitos, de los cuales el 47,3 % fueron ejecutados.

Los requisitos que no se han alcanzado han sido debido a problemas en la definición inicial que han impedido cumplir con la deadline. El principal punto de mejora sería la implementación de los requisitos faltantes, a fin de conseguir un jueglio funcional y disfrutable.

10.1.1.3 - LECCIONES APRENDIDAS

Este proyecto ha sido una gran oportunidad para aprender, no tanto en cuanto a habilidades de programación, más bien a organizarse e integrarse en un grupo de trabajo. Yo siendo la coordinadora del proyecto móvil he tenido que mejorar mis habilidades blandas como la comunicación, organización, etc.

Con los percances que ha habido, como miembros del grupo que no cumplían los requisitos para ciertas tareas, falta de conocimiento en el uso de herramientas o mis propios percances, he aprendido que una buena organización y marcarse plazos establecidos ayuda a llegar a un objetivo.

En una visión global, puedo decir que este proyecto al necesitar una buena comunicación, me ha enseñado a expresar mis ideas de una manera más entendible e integrarse en un grupo de trabajo donde toda opinión tenga que ser escuchada.

10.1.2 - CONCLUSIONES DEL SUBPROYECTO PROYECTO INDIVIDUAL

En este punto se detallan las conclusiones del subproyecto de la aplicación móvil. A continuación se describen las enseñanzas y reflexiones que se han obtenido durante el desarrollo del proyecto.

10.1.2.1 - EXPECTATIVAS Y MEJORAS

La aplicación móvil cumple con los requisitos mínimos funcionales. Utiliza una librería de comunicaciones y se comunica con las pantallas para enviar y recibir información, lo que permite la interacción con el usuario. Sin embargo, hay varias cosas que se pueden mejorar en cuanto a la experiencia del usuario, como incluir señales gráficas para hacer la experiencia de juego más agradable.

En cuanto a la implementación del código, se podría refactorizar módulos como los controladores de la lógica de comunicación para obtener un código más limpio y fácil de entender.

10.1.2.1.1 - MEJORAS EN LA PLANIFICACIÓN Y GESTIÓN

Es evidente que la planificación y gestión del tiempo del proyecto no ha sido adecuada, lo que ha llevado a una distribución desigual de tareas y una falta de cumplimiento de los objetivos finales.

Para futuros proyectos, se recomienda que la planificación y gestión del tiempo se realice de manera más exhaustiva y equitativa, para asegurar que todas las tareas se completen de manera efectiva y en el tiempo establecido.

Además, se sugiere una comunicación más efectiva entre los miembros del equipo para asegurar que todas las tareas se completen en el tiempo establecido, y para detectar posibles problemas en una etapa temprana y poder resolverlos de manera oportuna.

Otro aspecto importante a mejorar es la distribución de tareas entre los miembros del equipo, asegurándose de que cada uno tenga una carga de trabajo equitativa y adecuada a sus habilidades y capacidades.

En resumen, para asegurar una planificación y gestión del tiempo efectiva en futuros proyectos, se recomienda una planificación más exhaustiva, una comunicación efectiva y una distribución equitativa de tareas.