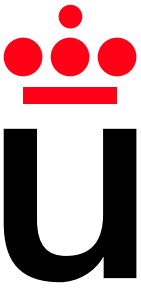


Universidad
Rey Juan Carlos

Escuela Técnica Superior
de Ingeniería Informática

Grado en Diseño y Desarrollo de Videojuegos

Curso 2021-2022

Trabajo Fin de Grado

INTERACCIÓN ENTRE PERSONAJE Y JUGADOR: DISEÑO DE UN VIDEOJUEGO Y LA IMPLICACIÓN DEL USUARIO EN EL MUNDO DE FICCIÓN

Autor: Germán López Gutiérrez
Tutor: Daniel Palacios Alonso


BY NC SA

Agradecimientos

Me gustaría utilizar este pequeño rincón para expresar mi gratitud hacia todas las personas que me han apoyado durante estos últimos años.

Muchas gracias a mis padres por haberme permitido perseguir mi sueño y estudiar lo que siempre he querido.

Gracias a mi tutor, Daniel Palacios Alonso, y a Elvira Gutiérrez Bartolomé por apoyar el desarrollo de este proyecto y por darme cada uno su perspectiva en aquellos momentos donde numerosas dudas afloraban.

También quiero darle las gracias a Javier Ortega, un profesor que confió en mí y me motivó a dar el paso para estudiar lo que más me apasiona. Sinceramente, espero estar cumpliendo con tus expectativas.

Y por último, pero no menos importante, muchas gracias a todos y cada uno de mis amigos y compañeros, tanto provenientes de la universidad como externos a la misma. Especialmente a Francisco Montiel Díaz, el cuál ha estado más de 6 años apoyándome y motivándome a ser mejor, tanto como desarrollador de videojuegos como persona.

Muchas gracias a todos.

Resumen

Durante el desarrollo de este proyecto, se busca estudiar la relación que puede establecer un usuario con el mundo de ficción de un videojuego, logrando esto mediante el estudio de las posibilidades ofrecidas por el medio y el diseño de una serie de mecánicas que establecen una interacción directa entre el personaje controlado y el jugador.

Dado que no existe una regla universal que concrete cómo implicar al jugador en el espacio de juego y permitirle establecer un vínculo con su personaje, se analizarán diferentes obras que han intentado romper la barrera entre el mundo real y el ficticio. Se comenzará por definir el término 'cuarta pared', especificando su origen en el teatro griego, y prosiguiendo con el análisis de videojuegos de esta última década. Además, se aplicarán los conocimientos adquiridos en el diseño de una propuesta de videojuego y el desarrollo de un prototipo en Unreal Engine 4 con el que, a través de una encuesta, se han recopilado una serie de datos necesarios para conocer si el modelo de diseño planteado es válido o si por el contrario requiere de una serie de modificaciones para un desempeño óptimo en el desarrollo de este tipo de experiencias.

A continuación, se especificará la motivación y el estado del arte de los videojuegos que han roto la cuarta pared en el pasado, definiendo en el proceso dos hipótesis y una serie de objetivos que determinarán el desarrollo del trabajo. Posteriormente, se especificará una planificación, donde se establecen los integrantes del equipo junto a los roles albergados por cada uno, el modelo de desarrollo planteado y las fechas propuestas. Tras la descripción de dicha planificación, se profundizará en el proceso de desarrollo, donde se describirá todo el diseño del videojuego propuesto y se detallará la realización del prototipo mediante el uso del motor Unreal Engine 4. Por último, se hará un análisis de los resultados obtenidos en la encuesta y se especificarán las conclusiones y líneas futuras sobre este estudio.

Palabras clave:

- Cuarta pared
- Game Design
- Unreal Engine 4
- Desarrollo de videojuegos

Abstract

During the development of this project, the aim is to study the relationship that a user can establish with the fictional world of a video game, achieving this through the study of the possibilities offered by the medium and the design of a series of mechanics that establish a direct interaction between the controlled character and the player.

Since there is no universal rule that specifies how to involve the player in the game space and allow him to establish a link with his character, different works that have tried to break the barrier between the real and the fictional world will be analyzed. We will begin by defining the term 'fourth wall', specifying its origin in Greek theater, and continuing with the analysis of video games of the last decade. In addition, the knowledge acquired will be applied to the design of a video game proposal and the development of a prototype in Unreal Engine 4 with which, through a survey, a series of data necessary to know if the design model proposed is valid or if, on the contrary, it requires a series of modifications for an optimal performance in the development of this type of experiences have been collected.

Next, the motivation and the state of the art of video games that have broken the fourth wall in the past will be specified, defining in the process two hypotheses and a series of objectives that will determine the development of the work. Subsequently, a planning will be specified, where the team members are established along with the roles housed by each one, the proposed development model and the proposed dates. After the description of this planning, the development process will be described, where the entire design of the proposed video game will be described and the prototype will be detailed using the Unreal Engine 4. Finally, an analysis of the results obtained in the survey will be made and the conclusions and future lines of this study will be specified.

Palabras clave:

- Fourth wall
- Game Design
- Unreal Engine 4
- Video Game Development

Índice de contenidos

Índice de términos **XV**

Índice de tablas **XIX**

Índice de figuras **XXIV**

1. Introducción **1**
1.1. Motivación 1
1.2. Estado del Arte 2
1.3. Hipótesis 6

2. Objetivos y Planificación **8**
2.1. Objetivos 8
2.2. Metodología empleada 9
2.3. Herramientas utilizadas 11
2.3.1. Unreal Engine 4 11
2.3.2. Blender 13
2.3.3. HacknPlan 13
2.3.4. Hojas de Cálculo de Google 14
2.3.5. Adobe Photoshop 14
2.4. Planificación 14

3. Marco Teórico **18**
3.1. Los 13 Principios del Diseño 18
3.2. Redacción de un Documento de Diseño 20
3.3. Teoría sobre los Tipos de Personaje 22
3.3.1. Personaje reflejo del jugador 22
3.3.2. Personaje reflejado en el jugador 24
3.3.3. Personaje Híbrido 24
3.4. Los 8 Tipos de Diversión 25

4. Proceso de Desarrollo **28**
4.1. Diseño del Videojuego 28
4.1.1. Visión General 28

4.1.2. Cuestiones Importantes 30
4.1.3. Portada y Contraportada 31
4.1.4. Referencias 34
4.1.5. Especificaciones 35
4.1.6. Ambientación 36
4.1.7. Jugabilidad 38
4.1.8. Progresión 38
4.1.9. Almacenamiento del Estado de la Partida 40
4.1.10. Controles 41
4.1.11. Front End 42
4.1.12. Tecnología 44
4.1.13. Temas e Inclusión 44
4.2. Implementación en un Prototipo 45
4.2.1. Trama 45
4.2.2. Diseño de Niveles 46
4.2.3. Animación 3D 48
4.2.4. Importación de Modelos 49
4.2.5. Programación 50
4.2.6. VFX 74

5. Análisis de Resultados **78**
5.1. Análisis sobre el trabajo realizado 78
5.2. Perfiles de los Encuestados 78
5.3. Análisis sobre la Narrativa y el Diseño 80

6. Conclusiones y Líneas Futuras **86**
6.1. Conclusiones sobre el Desarrollo del Prototipo 86
6.2. Cumplimiento de los Objetivos 88
6.3. Resolución de las Hipótesis 90
6.4. Líneas Futuras 92

Bibliografía **94**

Apéndices **97**

A. Animación en Blender **99**
A.1. Animaciones del Prototipo 99
A.2. Animaciones No Incluidas en el Prototipo 104

B. Guion de la Demo **107**
B.1. Llegada a Neo-Osaka 107
B.2. Encuentro con John 108
B.3. Encuentro con Cadáveres 111

B.4. Entrega del Anillo de la Esposa de John 111
B.5. Encuentro con Militar 113
B.6. Cinemática Final 116

Índice de términos

Asset Complemento o conjunto de complementos adquirido para un proyecto en desarrollo.

Behaviour Tree Árbol de decisión para el desarrollo e implementación de una inteligencia artificial.

Blackboard Tabla contenedora de las variables que utilizará un *Behaviour Tree* en Unreal Engine.

Blueprint Archivo que contiene código del videojuego en un formato visual compuesto por nodos únicamente visibles mediante el editor de Unreal Engine.

Demo Versión de prueba con contenido reducido de un videojuego previa a la compra o financiación del proyecto.

Game Loop En el contexto de un videojuego, es la principal repetición (o bucle) de actividades a la hora de jugar.

Gantt Gráfico que representa las actividades a realizar por el equipo de desarrollo y su evolución en el tiempo.

Input En el contexto de un videojuego, entrada recibida mediante la pulsación de un botón o tecla.

Launcher Interfaz o plataforma que alberga videojuegos ejecutables por un usuario registrado.

Mod Modificación no oficial de un videojuego realizado por una persona generalmente externa al equipo de desarrollo original.

PEGI Clasificación por edad según la Junta de Clasificación de Software de Entretenimiento (ESRB) para cada videojuego.

PNJ Personaje no jugador en un videojuego.

Plugin Accesorio que añade funcionalidades nuevas a una pieza de software.

Prop Elemento decorativo para un escenario 3D.

Questline Misión o conjunto de misiones realizable en un videojuego no lineal donde se establece una continuidad narrativa.

Raycast Trazada de un rayo que devuelve datos sobre el impacto y el objeto contra el que colisiona.

Rigging Creación y colocación de un esqueleto virtual para un modelo 3D.

Scrum Metodología ágil iterativa e incremental para la gestión de proyectos.

Selector Nodo de un *Behaviour Tree* que sirve de nexo entre diversas ramificaciones, donde se determina a cuál se avanza según una condición.

Sequence Nodo de un *Behaviour Tree* que ejecuta una secuencia de acciones, tareas o nuevas ramificaciones.

Shoot 'em up Género de videojuego que gira entorno a la lluvia de balas y el controlar a un personaje capaz de acabar con los enemigos y esquivar los disparos.

Shooter Género de videojuego que se centra en los disparos.

Skinning Ajuste de los vértices del mallado de un modelo 3D y su ratio de modificación respecto a los huesos.

Software Programa o conjunto de programas destinado a realizar una serie de tareas.

Soulslike Género de videojuego que se enfoca en la exploración de escenarios interconectados y la muerte del jugador como mecánica de juego más allá del reinicio del nivel.

Speedrunner Jugador especializado en completar videojuegos en un corto periodo de tiempo.

Sprint Iteración de *Scrum*.

Índice de tablas

2.1. Distribución de tareas. 15

6.1. Cuantificación final del trabajo realizado 1. 87
6.2. Cuantificación final del trabajo realizado 2. 87
6.3. Cuantificación final del tiempo dedicado. 88

Índice de figuras

1.1. Metal Gear Solid - Personajes rompiendo la cuarta pared. 3
1.2. The Stanley Parable - Narrador rompiendo la cuarta pared. 3
1.3. Oneshot. 4
1.4. Doki Doki Literature Club! - Último momento con Sayori. 4
1.5. Doki Doki Literature Club! - Errores en el videojuego. 5
1.6. Doki Doki Literature Club! - Monika interactuando con el jugador. 5

2.1. Desarrollo Ágil - Esquema. 9
2.2. *Scrum* - Esquema. 10
2.3. *Plugin* - AutoSizeComments 1. 11
2.4. *Plugin* - Blockout Tools 1. 12
2.5. *Plugin* - Blockout Tools 2. 12
2.6. *Plugin* - GraphFormatter. 13
2.7. Planificación del proyecto - *Gantt*. 16
2.8. Planificación del proyecto - *Sprints*. 16

3.1. Fallout 3 - Nacimiento. 23
3.2. Fallout 3 - Infancia. 23
3.3. God of War III - Combate contra Dioses. 24

4.1. Stigma Protocol - Game Loop. 29
4.2. Stigma Protocol - Portada. 32
4.3. Stigma Protocol - Contraportada. 33
4.4. Referencias de estilo artístico. 34
4.5. Livelock - Ejemplo de perspectiva. 34
4.6. Referencias del estado del planeta. 34
4.7. Ejemplos de estilo artístico. 35
4.8. Sucesión de Niveles. 37
4.9. Esquema de Armas. 39
4.10. Controles con mando. 41
4.11. Controles con teclado. 41
4.12. Controles con ratón. 41
4.13. Pantalla de Título. 42

4.14. Menú Principal. 43
4.15. Menú de Ajustes. 43
4.16. Diseño de niveles - Estructura del nivel 1. 46
4.17. Diseño de niveles - Estructura del nivel 2. 47
4.18. Diseño de niveles - División por zonas. 47
4.19. Animación 3D - Disparo enemigo con línea de tiempo. 48
4.20. Animación 3D - Movimiento del jugador. 48
4.21. Importación de modelos - Organización de los modelos importados. 49
4.22. Importación de modelos - Organización de los modelos animados. 49
4.23. Mecánicas - Funciones y eventos. 50
4.24. Mecánicas - Declaración de inputs. 51
4.25. Mecánicas - Eventos. 51
4.26. Mecánicas - Grafo de eventos. 52
4.27. Incorporación de las animaciones - Grafo del personaje. 53
4.28. Incorporación de animaciones - Movimiento mediante *Blend Space*. 54
4.29. Incorporación de animaciones - Movimiento interpolado mediante *Blend Space*. 54
4.30. Incorporación de animaciones - Cálculo de variables. 55
4.31. Incorporación de animaciones - Creación del *Socket* de la mano. . 55
4.32. Incorporación de animaciones - Asignación del *Socket* al arma. . . 55
4.33. Incorporación de animaciones - Creación del *Socket* del arma. . . 56
4.34. Incorporación de animaciones - Asignación del *Socket* a la mano. . 56
4.35. Incorporación de animaciones - Rotación de la cabeza. 56
4.36. Incorporación de animaciones - Rotación de la cabeza, *debug* 1. . 57
4.37. Incorporación de animaciones - Rotación de la cabeza, *debug* 2. . 57
4.38. Enemigo - Behaviour Tree. 58
4.39. Enemigo - Inicialización de la IA. 59
4.40. Enemigo - *Blueprints* para las tareas del *Behaviour Tree*. 60
4.41. Enemigo - Estructura de las tareas. 60
4.42. Enemigo - Variables definidas para la *blackboard*. 61
4.43. Enemigo - Árbol detallado. 61
4.44. Enemigo - Detección del Jugador. 62
4.45. Enemigo - *Pathfinding*. 62
4.46. Enemigo - *Strafing*. 62
4.47. Sistema de selección de *props* - Selectores. 63
4.48. Sistema de selección de *props* - Materiales. 63
4.49. Sistema de selección de *props* - *Mesh Type*. 64
4.50. Sistema de selección de *props* - *Mesh Index*. 64
4.51. Sistema de selección de *props* - Llamada *Check Mesh Visibility*. . 65
4.52. Sistema de selección de *props* - Selección de *Selectable Mesh*. . . 66

4.53. Sistema de selección de *props* - Selección del modelo con el enumerador *Mesh Index*. 66
4.54. Sistema de selección de *props* - Carga de los materiales al cambiar de modelo. 67
4.55. Sistema de selección de *props* - Actualización del modelo con nuevos materiales. 67
4.56. Sistema de selección de *props* - Diagrama de clases. 68
4.57. Interfaz - Punto de ancla. 68
4.58. Interfaz - Adaptación de los elementos de la interfaz a la proporción de la pantalla. 69
4.59. Menú - Botón de encuesta. 69
4.60. Selector de apariencia - Grafo de animaciones. 70
4.61. Selector de apariencia - Materiales. 70
4.62. Selector de apariencia - Cámara y animación. 71
4.63. Cinemáticas - Trigger. 72
4.64. Introducción - Nombre de usuario. 74
4.65. VFX - Material que simula la desintegración. 75
4.66. VFX - Proceso de desintegración. 75
4.67. VFX - Variables del material. 75
4.68. VFX - Partículas y selección de esqueleto. 76
4.69. VFX - Resultado final. 76

5.1. Encuesta - Familiarización con las nuevas tecnologías. 79
5.2. Encuesta - Franja de edad. 79
5.3. Encuesta - Launchers. 80
5.4. Encuesta - Significado de *inmersión*. 81
5.5. Encuesta - Conexión con el personaje principal. 82
5.6. Encuesta - Significado de la introducción. 82
5.7. Encuesta - Comparativa con otros protagonistas. 83
5.8. Encuesta - Respuesta sobre uno mismo. 83
5.9. Encuesta - Género del videojuego. 84

6.1. Pasos a seguir en el desarrollo de un videojuego. 91

A.1. Animaciones del prototipo - Diálogos. 99
A.2. Animaciones del prototipo - Movimiento con fusil. 100
A.3. Animaciones del prototipo - Gesto de dolor 1. 100
A.4. Animaciones del prototipo - Gesto de dolor 2. 101
A.5. Animaciones del prototipo - Gesto de dolor 3. 101
A.6. Animaciones del prototipo - Personaje sentado. 101
A.7. Animaciones del prototipo - Animaciones del enemigo. 102
A.8. Animaciones del prototipo - Muerte 1. 102
A.9. Animaciones del prototipo - Muerte 2. 103

A.10.Animaciones del prototipo - Personaje enseñando el brazalete. . . 103
A.11.Animaciones no incluidas - Animaciones con katana. 104
A.12.Animaciones no incluidas - Ataque con katana. 104
A.13.Animaciones no incluidas - Muertes. 105

1 Introducción

En esta primera sección se especificará la motivación de este estudio sobre la implicación del jugador en el mundo de ficción, se analizará el estado del arte, el cual servirá como punto de partida para utilizar lo ya estudiado por otros directores en videojuegos de años previos y, por último, se definirán las hipótesis propuestas.

1.1. Motivación

Cuando se busca diseñar un videojuego, existen numerosos factores a tener en cuenta para el desarrollo de una idea coherente y que genere interés al público objetivo. Dichos factores pueden ser la jugabilidad, la historia, el diseño de personajes, el estilo artístico, el diseño narrativo, el sonido, la música, el diseño de niveles, el género, etc. Pueden parecer muchos agentes a considerar, y es que la complejidad a la hora de diseñar un videojuego no reside solo en la cantidad de elementos a tener en cuenta, sino en saber cómo juntarlos todos en un mismo producto y garantizar que todo encaje a la perfección. Para ello, es normal que, entre todos los aspectos considerados, se haga énfasis en un elemento en concreto y que absolutamente todos los otros componentes giren en torno a este. Por esto, hay que buscar un tema, un enfoque que defina el eje central del videojuego. "¿Qué aporta?", "¿Por qué hacer este videojuego?" o "¿A quién va dirigido?" son algunas de las preguntas que se debe formular un diseñador que se encuentre en los primeros pasos para llevar a cabo el desarrollo de un videojuego.

Esta manera de trabajar ha provocado que se generen distintas vertientes,

videojuegos con un tema en particular. Existen obras donde el apartado principal es la historia, como The Last of Us [1], otros donde destaca la música, como en el caso de Crypt of the NecroDancer [2] o Friday Night Funkin' [3].

En este último siglo, directores como Davey Wreden, Dan Salvato o Hideo Kojima han decidido enfocarse en encontrar la manera más interactiva de introducir al jugador como partícipe directo en el mundo de ficción. Para ello, uno de los recursos más importantes ha sido la ruptura de la cuarta pared, herramienta utilizada por primera vez en el teatro griego, la cual se ha encontrado en continua evolución hasta el día de hoy, llegando al terreno de los videojuegos. Obras como Doki Doki Literature Club! [4], The Stanley Parable [5] o Metal Gear Solid 2 [6] han buscado la manera de implicar al jugador de manera directa con los personajes, añadiendo películas como Her [7], la cuál estudia la relación que puede llegar a entablar una persona con un programa informático, siendo el tema principal de este trabajo determinar si pueden personaje y jugador generar una conexión directa mediante la ruptura de la cuarta pared y el transcurso de una trama, haciendo uso del análisis de diferentes videojuegos clave y la implementación de las técnicas analizadas en el diseño de un videojuego real.

1.2. Estado del Arte

Previo a la aparición de los videojuegos, numerosas veces en obras narrativas se ha hecho uso de un recurso conocido como la ruptura de la cuarta pared, donde "la cuarta pared", un término utilizado por primera vez por André Antonie según cita Cécile Vilvandre de Sousa en *La teatralidad en el espacio contemporáneo francés* [8], se define como aquello que separa al espectador del escenario en una obra de teatro. La ruptura de la cuarta pared es un término originario del teatro que se ha adaptado al resto de artes, aplicándose cuando la barrera que divide el mundo real del ficticio se rompe, adquiriendo los personajes un conocimiento del plano exterior, interactuando o no de manera directa con el espectador.

En el campo de los videojuegos, uno de los momentos más influyentes donde la ruptura de la cuarta pared se hace presente toma lugar en el videojuego de Play Station, Metal Gear Solid [9], a la hora de confrontar a Psycho Mantis, tal y como describe Steven Conway en su artículo *A circular wall? Reformulating the fourth wall for video games* [10]. Esto es debido a que el personaje toma consciencia de uno de los elementos que caracteriza a las videoconsolas: los archivos de guardado. En dicho combate, Psycho Mantis es capaz de leer los datos guardados de diferentes juegos y hacer comentarios al respecto, riéndose del jugador. Se puede observar uno de los posibles diálogos con el personaje en la figura 1.1a. Además, es capaz de activar la vibración del mando a voluntad y, llevado al campo de la jugabilidad, la única forma de poder derrotarle es desconectando el mando de la consola y cambiándolo de puerto, haciendo que el jefe se confunda y éste pueda

ser atacado.

Desde el enfrentamiento contra Psycho Mantis en Metal Gear Solid, la ruptura de la cuarta pared se convirtió en un elemento recurrente en diversos videojuegos, dando pie a los desarrolladores a idear nuevas formas de interacción entre el jugador y el videojuego.

Tres años más tarde, en 2001, Kojima repetiría la jugada en Metal Gear Solid 2 [6], donde, en la recta final del juego, el coronel Roy Campbell le ordena al jugador que apague la consola, tal y como se puede observar en la figura 1.1b.

(a) Metal Gear Solid 1. (b) Metal Gear Solid 2.

Figura 1.1: Metal Gear Solid - Personajes rompiendo la cuarta pared.

En 2011 fue desarrollado The Stanley Parable, un *mod* para Half Life 2 [11] donde un narrador cuenta la historia de Stanley a medida que avanza el jugador en la aventura. La característica principal del juego es que dicho narrador ordena al jugador lo que debe hacer, pero éste puede desobedecerle y tomar decisiones propias, dando lugar a diferentes finales donde el narrador interactúa con el jugador y modifica los escenarios del videojuego. Se puede observar uno de los finales junto a la reacción del narrador en la figura 1.2.

Tal fue el éxito del *mod*, que en 2013 fue publicado en Steam —por el mismo diseñador— un videojuego completo que utiliza todas las ideas planteadas y las expande, añadiendo nuevas decisiones y finales.

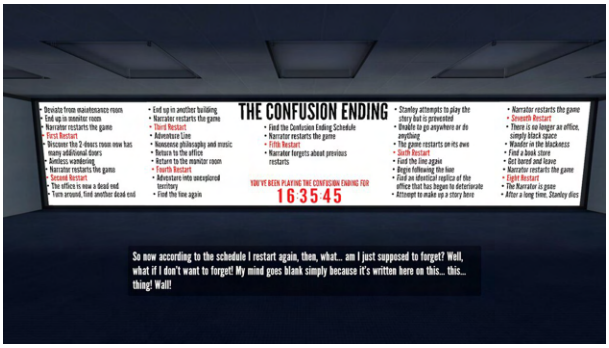


Figura 1.2: The Stanley Parable - Narrador rompiendo la cuarta pared.

En diciembre de 2016, fue publicado en Steam OneShot [12], un videojuego de aventura y puzles que experimenta con el conocimiento que puede llegar a tener el protagonista sobre el mundo real.

En este juego, Niko, el protagonista, llega a entablar una relación de amistad con el jugador y, como se puede ver en la figura 1.3b, le pide ayuda a éste para resolver problemas que se escapan a su alcance, como buscar imágenes o códigos dentro de la ubicación de instalación del juego.

(a) Logo. (b) Petición de ayuda al jugador.

Figura 1.3: Oneshot.

Posteriormente, en septiembre de 2017, fue publicado un videojuego gratuito llamado Doki Doki Literature Club!, un videojuego de terror psicológico camuflado bajo la premisa de ser una novela visual convencional de simular citas con chicas de instituto. La razón del éxito de este juego fue la atmósfera de terror y descontrol que se genera a partir del tercer día, tras el fallecimiento de Sayori, una de las compañeras del club de literatura. En la figura 1.4 se muestra el último momento con el personaje antes de su muerte.

Figura 1.4: Doki Doki Literature Club! - Último momento con Sayori.

Después de encontrar el cuerpo sin vida de la chica, el juego se cierra por su cuenta y, tras volverlo a iniciar, comienzan a aparecer *bugs* y *glitches* en los menús (figura 1.5a) y en los escenarios y personajes (figura 1.5b), además que los diálogos comienzan a ser inconexos y a ser escritos con otros caracteres.

(a) Menú principal. (b) Diálogos.

Figura 1.5: Doki Doki Literature Club! - Errores en el videojuego.

El descontrol en el mundo del juego llega a tal punto que, en los últimos minutos, Monika —la delegada del club de literatura— revela ser la causante de todo el caos y confiesa estar enamorada del jugador. Todo lo sucedido era debido a que quería era hablar con él y se resiente ante el hecho de que nunca será una persona real ni podrá estar con él en el plano físico. Los motivos de Monika se pueden observar en la figura 1.6.

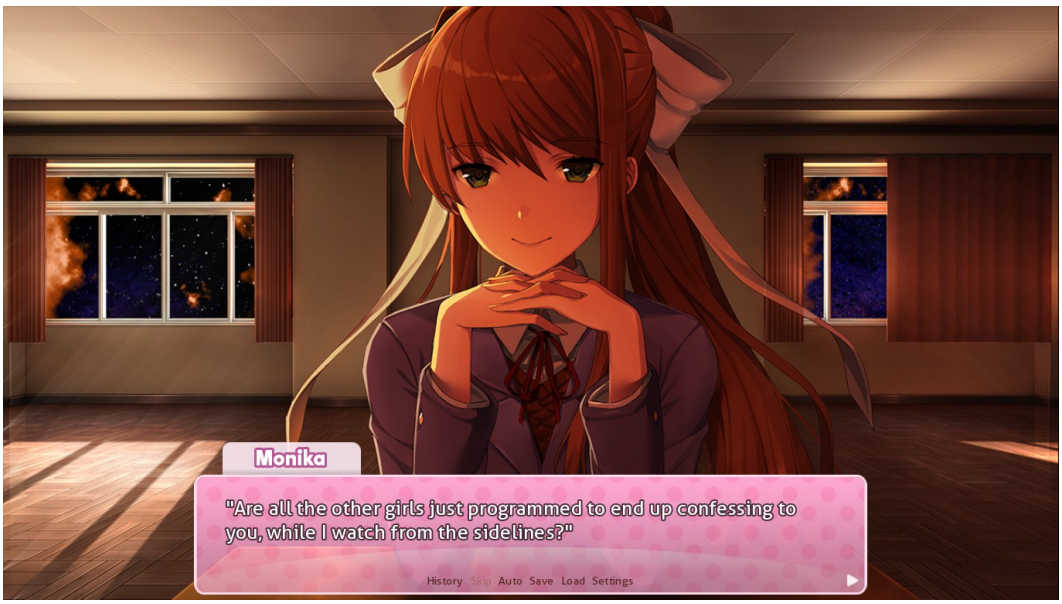


Figura 1.6: Doki Doki Literature Club! - Monika interactuando con el jugador.

Este juego fue todo un éxito, y, en agosto de 2021, fue lanzada a la venta una versión de pago, Doki Doki Literature Club Plus! [13], que expande el contenido del videojuego, además de ser publicado en más plataformas como Nintendo Switch y Play Station 5.

1.3. Hipótesis

Tras lo planteado anteriormente, se han definido dos hipótesis en torno a las que orbitará este trabajo:

1. ¿Cuál es el proceso de diseño a seguir en un videojuego que tiene como eje central la participación directa del jugador en el mundo de ficción que se le presenta?
2. ¿Puede un personaje no jugador simular una consciencia adaptativa en función de las acciones tomadas por el jugador en el mundo que se le presenta?

En el capítulo siguiente se especifican los objetivos a cumplir respecto a estas hipótesis.

2

Objetivos y Planificación

A continuación se especifican los objetivos y la planificación del proyecto.

2.1. Objetivos

1. Estudiar la **relación entre personaje y jugador** a través de la narrativa y la implementación de la **ruptura de la cuarta pared** como recurso principal.

2. Diseñar unas **mecánicas de juego** acordes a la necesidad de implementar una experiencia donde personaje y jugador estén directamente relacionados.

3. Estudiar los factores que permiten la **inmersión del jugador** en la experiencia de juego.

4. Investigar los distintos elementos que influyen en el momento de construir una narrativa alrededor de los **diferentes tipos de interacción** que puede presentar un videojuego.

5. Redactar un **documento de diseño** acorde a las necesidades y los temas planteados.

6. Diseñar un **prototipo** especificando las mecánicas, el sistema de combate, el posicionamiento de la cámara y los tipos de enemigos.

2.2. Metodología empleada

El objetivo principal es implementar las ideas estudiadas en el desarrollo de un videojuego. Para ello, existen numerosas formas de llevar a cabo el proyecto al igual que con cualquier otro tipo de software. Por este motivo se consideró el uso del desarrollo en cascada, propuesto por Winston Royce en *Managing the development of large software systems: concepts and techniques* [14] en 1970.

Este proceso consta de diversas etapas que transcurren de manera secuencial, siendo el regreso a una etapa anterior en un principio no recomendado debido al posible gran costo en tiempo y dinero que puede conllevar dicha situación.

Aún así, se ha optado por la utilización de *Scrum*, una metodología ágil introducida por Ikujiro Nonaka y Takeuchi en 1986 en *The New New Product Development Game* [15].

El desarrollo ágil consiste en distintos métodos de ingeniería software basados en el desarrollo iterativo e incremental. En cada iteración se realizan en orden las etapas del planificación, el análisis de requisitos, el diseño, la codificación, la fase de pruebas y la documentación sobre los resultados. La ventaja de este modelo, es que permite realizar las etapas de desarrollo y prueba simultáneamente e influye la comunicación entre los integrantes del proyecto. Se ha adjuntado un esquema visual sobre el desarrollo ágil en la figura 2.1.

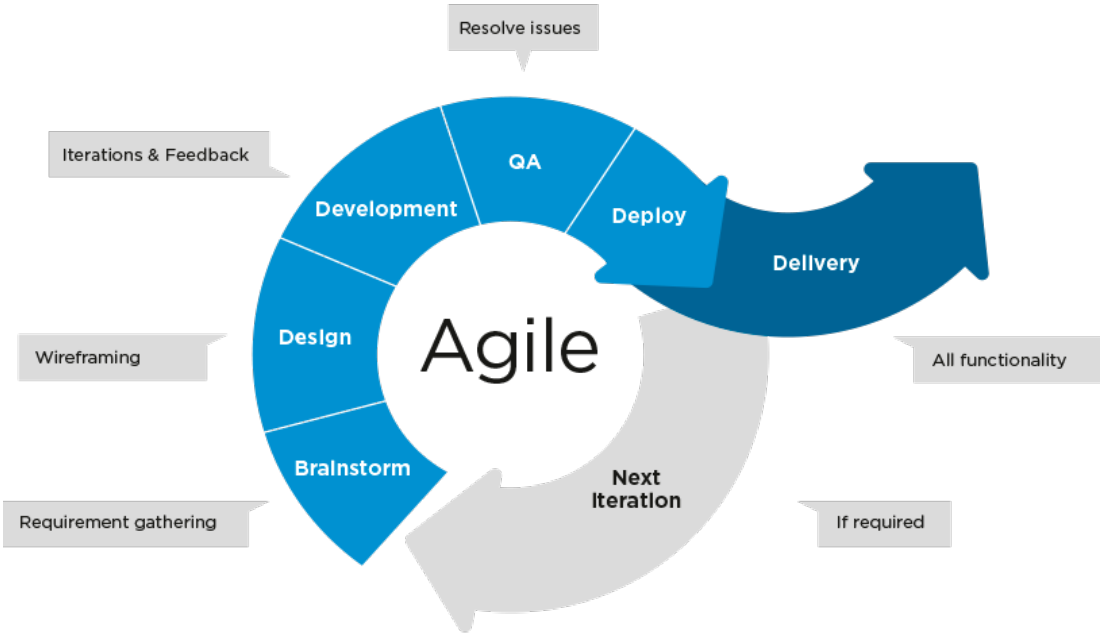


Figura 2.1: Desarrollo Ágil - Esquema.

En este caso se ha elegido *Scrum*, debido a que es un modelo recomendado para equipos de entre 5 o menos personas, ideal en este proyecto ya que el grupo constaba de cuatro integrantes para el desarrollo del videojuego, aunque se profundizará en esto en el apartado 2.4. Planificación. También se ha adjuntado un esquema sobre esta metodología ágil en la figura 2.2. Las etapas en este modelo son las siguientes:

- **Determinación de requisitos**: Se especifican cuáles son los requisitos a cumplir.
- **Planificación de la iteración**: Se definen qué tareas se completarán en el período de tiempo establecido para la iteración, también conocida como *Sprint*.
- **Ejecución**: Se llevan a cabo las tareas definidas anteriormente.
- **Revisión**: Se revisa junto al cliente si las tareas realizadas cumplen con los requisitos.
- **Retrospectiva**: Se recopila la opinión de todos los miembros del equipo con el fin de mejorar para la siguiente iteración. Tras este paso se reinicia el ciclo de vida.

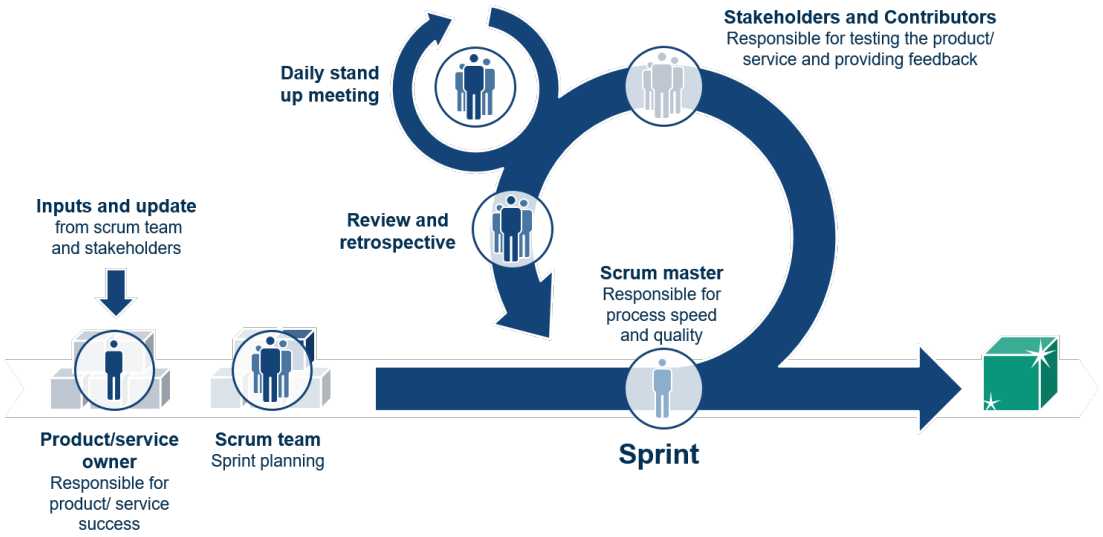


Figura 2.2: *Scrum* - Esquema.

Dadas las ventajas ofrecidas por esta metodología, *Scrum* será utilizado para el desarrollo del videojuego.

2.3. Herramientas utilizadas

2.3.1. Unreal Engine 4

Ya sea por la versatilidad que permiten sus métodos de programación, pudiendo elegir entre la utilización de *blueprints* o C++, por la gran calidad gráfica y lumínica que ofrece o por la gran cantidad de sistemas que agilizan la creación de videojuegos en 3D, Unreal Engine 4 es el motor que se ha seleccionado para el desarrollo del videojuego con el fin de obtener un resultado visual interesante en un corto período de tiempo.

A esto hay que sumarle la utilización de una serie de *plugins* utilizados que permitirán agilizar aún más el desarrollo del proyecto.

AutoSizeComments

Este *plugin* permite que, a la hora de utilizar *blueprints*, los comentarios se reajusten de manera automática al espacio comentado tras realizar un cambio, agilizando así las tareas de programación al no tener que ajustar cada comentario manualmente. Además, asigna a los comentarios un color aleatorio, lo que hace que cada comentario sea distinguible con respecto al resto, tal y como se puede observar en la figura 2.3.

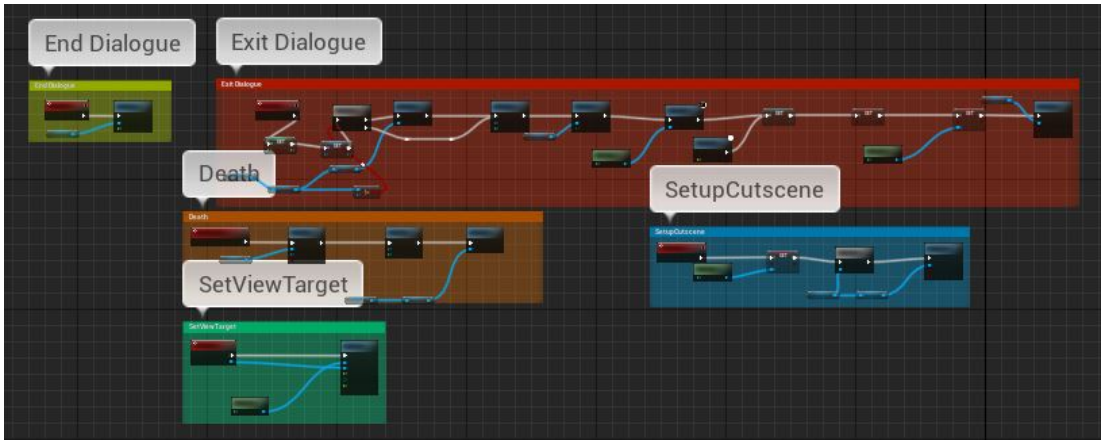


Figura 2.3: *Plugin* - AutoSizeComments 1.

Blockout Tools

Este *plugin* permite añadir modelos 3D modulares para facilitar el diseño de niveles, pudiendo incluir rampas, marcos para puertas, tuberías y muchas otras.

Esto permite agilizar enormemente el diseño del escenario y facilitarle la labor al artista, ya que al trabajar directamente sobre un entorno 3D en Unreal Engine 4 y no con un boceto del escenario en vista cenital, el posicionamiento de los *props* es mucho más claro, aumentando la fluidez en el desarrollo. En las figuras 2.4 y 2.5 se muestran ejemplos de modelos colocados mediante este plugin.

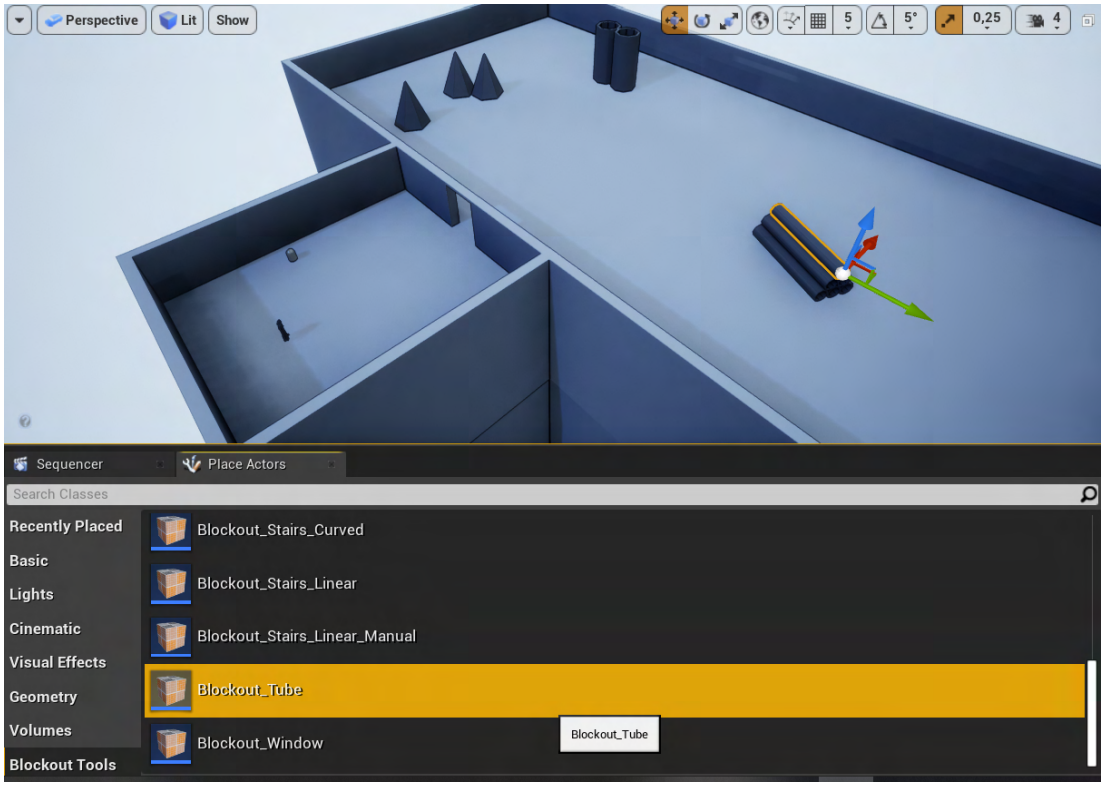


Figura 2.4: *Plugin* - Blockout Tools 1.

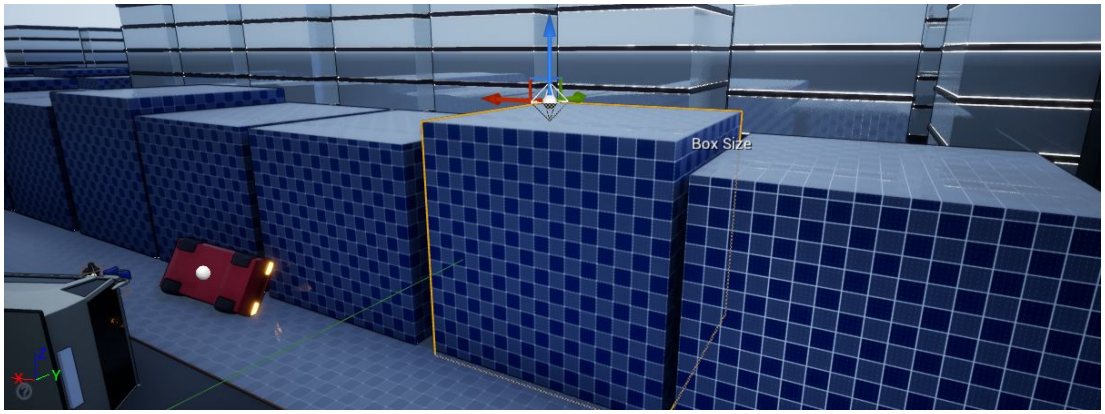
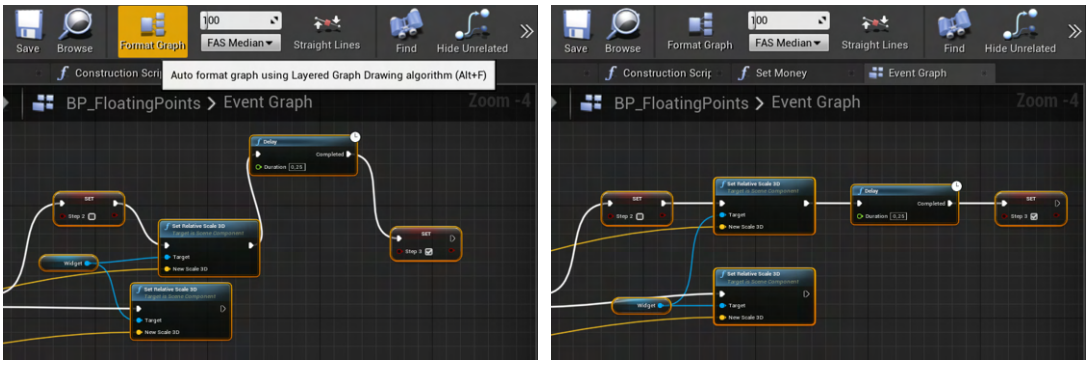


Figura 2.5: *Plugin* - Blockout Tools 2.

GraphFormatter

Esta extensión de Unreal Engine 4 permite reorganizar el contenido de los *blueprints* automáticamente de forma que sea lo más visible y ordenado posible sin dedicar demasiado tiempo a la organización, optimizando en legibilidad y tiempo.

En la figura 2.6a se muestra una serie de *blueprints* desordenados los cuales, tras presionar en el botón resaltado en amarillo, se ordenan, siendo el resultado el reflejado en la figura 2.6b.



(a) *Blueprint* desordenado. (b) *Blueprint* ordenado.

Figura 2.6: *Plugin* - GraphFormatter.

2.3.2. Blender

Es la alternativa gratuita más popular frente a otros programas como 3DS Max o Maya. Se ha elegido por su gran versatilidad al poder modelar, hacer el *rigging* y *skinning* para posteriormente animar y crear materiales con una gran variedad de opciones de configuración.

2.3.3. HacknPlan

Similar a Trello, esta herramienta permite organizar las tareas por integrantes del proyecto en columnas, con el añadido de poder también categorizarlas por roles y delimitar los *Sprints*, añadiendo fechas límite y creando una tabla nueva para las tareas de la siguiente iteración, todo en un mismo proyecto.

2.3.4. Hojas de Cálculo de Google

Esta herramienta sirve para organizar el número de tareas por integrante del equipo y realizar tablas donde se ajusten las fechas clave o se represente un diagrama de *Gantt*.

2.3.5. Adobe Photoshop

Este software es útil para el bocetado de mapas y mecánicas y la edición de imágenes. Esta herramienta no es gratuita, por lo que se ha hecho uso de la licencia otorgada por la universidad.

2.4. Planificación

En primer lugar, para llevar a cabo la planificación, se van a presentar todas y cada una de las tareas llevadas a cabo para el desarrollo del prototipo por el autor del presente trabajo:

- **Diseño de juego:** Es uno de los campos más importantes a la hora de desarrollar un videojuego. Un diseñador se encarga idear las mecánicas, la estética, los objetivos, las reglas, etc. En sus manos queda la tarea de redactar un documento de diseño que contenga toda la información relacionada con el desarrollo del videojuego, con el objetivo de orientar el proyecto hacia el mejor de los resultados.
- **Diseño de niveles:** Es la tarea cuyo resultado es el que define la estructura de los niveles del juego. El responsable de esta tarea se encarga de decidir el posicionamiento de los objetos, el encuentro con PNJs o enemigos, los puntos clave del nivel. En resumen, es el encargado de hacer de los escenarios del videojuego entornos de interés para el jugador.
- **Gestión del proyecto:** La planificación de un proyecto es esencial, por ello, se requiere de una persona que organice las tareas a realizar por los trabajadores o las fechas clave, con el objetivo de cumplir con los plazos y entregar un buen producto a tiempo.
- **Animación 3D:** Los encargados de esta labor deben utilizar los distintos modelos realizados por el artista 3D y añadirles movimiento. Abarca desde los distintos *props* del juego hasta los personajes y enemigos.
- **Programación:** Es la tarea donde se llevan a la realidad los conceptos propuestos durante el diseño. Esta tarea abarca desde las mecánicas de

juego hasta las inteligencias artificiales e incluso la inclusión y combinación de las animaciones.

- **VFX:** Los efectos visuales son esenciales para aportar vida al videojuego. Por ello, se realizarán distintos sistemas de partículas para obtener un resultado visual atractivo de cara al público. Algunos de los efectos son los disparos, el haz de luz emitido por las armas al ser disparadas o el humo de un escenario.

- **Narrativa:** La narrativa es un elemento esencial en muchos videojuegos con aventura lineal, es decir, con un inicio y final. Por ello, se debe escribir un guion con los diálogos de los personajes y las diferentes decisiones a tomar por el jugador y sus consecuencias.

- **Diseño narrativo:** El diseñador encargado de esta tarea se encarga de llevar los conceptos propuestos por el narrador al diseño de juego, conociendo las mecánicas, el ángulo de la cámara, el flujo del juego, etc.

Además, se ha contado con un equipo en el que cada uno ha cumplido un rol específico (tabla 2.1).

Nombre	**Rol**	**Tarea asignada**
Elvira Gutiérrez Bartolomé	Artista 3D	Se ha encargado del arte, la estética y los modelos del videojuego.
Francisco Montiel Díaz	Diseñador de niveles	Se ha encargado de ayudar en el diseño de niveles, aportando ideas y bocetos.
Alejandro Moreno Segovia	Compositor	Se ha encargado de realizar temas musicales, entre ellos a destacar el del menú principal.

Tabla 2.1: Distribución de tareas.

Por otro lado, se ha realizado un *Gantt* con las hojas de cálculo de Google (figura 2.7), donde se encuentra reflejada una macroplanificación con las tareas a desarrollar para un prototipo del videojuego. Esto se ha realizado así debido al abundante número de tareas por realizar, lo cuál alargaba el tiempo de planificación y complicaba la legibilidad del documento.

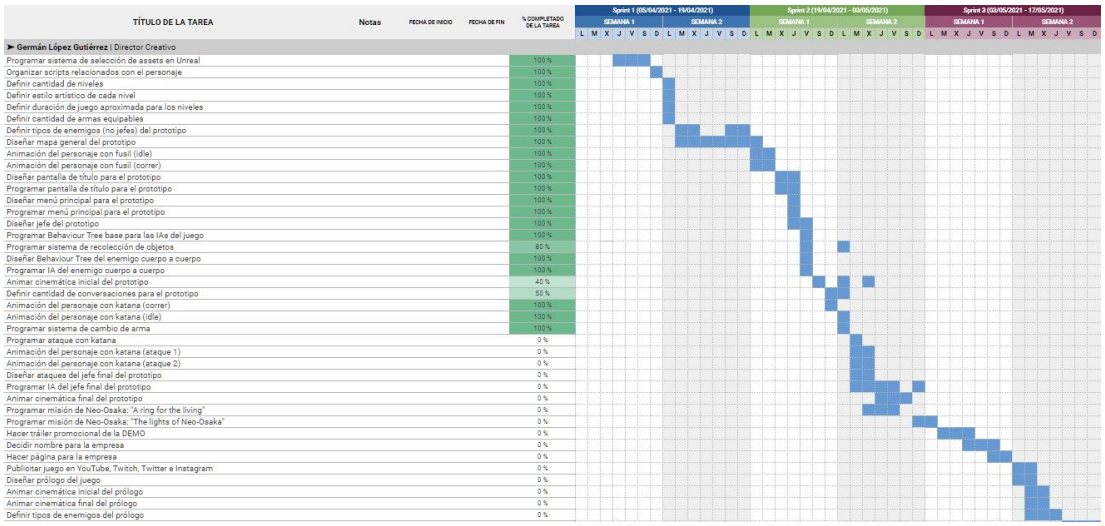


Figura 2.7: Planificación del proyecto - *Gantt*.

Además se ha hecho uso de un tablero de HacknPlan donde se fueron repartiendo las tareas por *Sprints*, siendo este tablero el que se muestra en la figura 2.8.

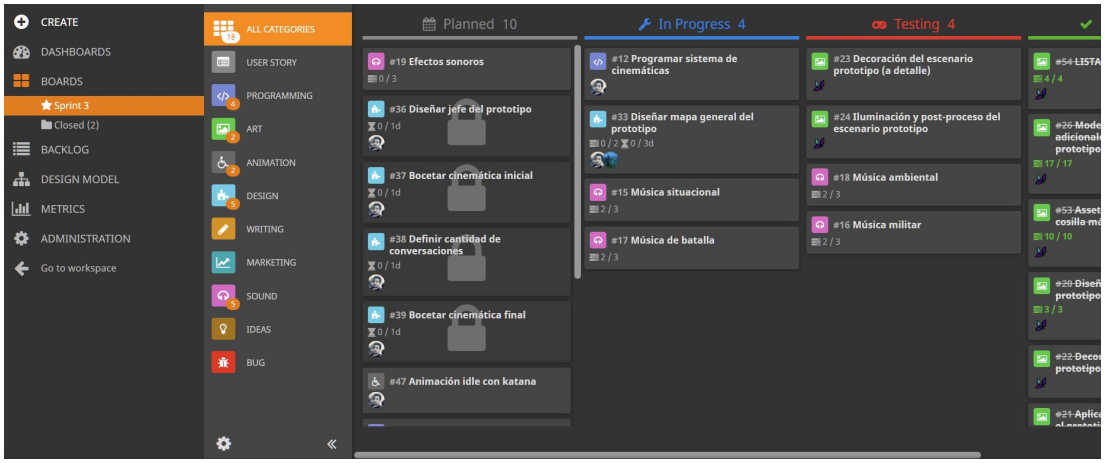


Figura 2.8: Planificación del proyecto - *Sprints*.

3

Marco Teórico

Teniendo en cuenta los objetivos definidos en el capítulo anterior, se han estudiado diferentes conceptos clave que favorecerán al resultado final del proyecto.

3.1. Los 13 Principios del Diseño

Según Matt Allmer [16], existen 13 reglas que se deben tener en cuenta para publicar al mercado un buen videojuego resultante del diseño y conceptualización del videojuego. Los principios a considerar son los siguientes:

1) Punto focal

Nunca hay que dejar en el aire hacia dónde dirigirá la mirada el jugador. Hay que guiarle con el plano de la cámara, el diseño del nivel, la colocación de eventos clave, etc.

2) Anticipación

Se debe informar al jugador sobre todo lo que está por suceder, teniendo éste tiempo de reacción ante peligros u otras posibles interacciones, evitando así la sensación de que la experiencia es injusta.

3) Anunciar el cambio

Todo cambio dentro del juego debe ser notificado al jugador. Este paso transcurre entre la anticipación y el evento en sí mismo.

4) Eventos y comportamientos creíbles

Cada uno de los eventos en el videojuego debe ser coherente y predecible respecto a las expectativas del jugador.

5) Superposición de eventos y comportamientos

Pueden existir eventos que transcurran simultáneamente con el fin de no perder la dinámica. Hay que añadir una cantidad adecuada de estos.

6) Física

Las decisiones del jugador se ven influenciadas por las posibilidades que ofrece la física dentro del videojuego. El diseñador debe tener en cuenta factores como la gravedad, la masa, la fuerza, etc.

7) Sonido

Hay que considerar qué sonidos realizan los distintos elementos del videojuego al ser el estado de los mismos modificado.

8) El ritmo

Hay que distribuir los elementos del videojuego de manera que sean acordes al ritmo de los acontecimientos, el nivel de concentración del jugador y la frecuencia con la que se repiten los sucesos.

9) Espaciado

Hay que comprender el espacio del que se dispondrá, tanto en la pantalla como en el mundo del videojuego.

10) Diseño lineal frente a desglose de componentes

El diseño lineal implica solucionar los problemas tal y como se presentan.

Cada una de las soluciones y posibilidades tienen el mismo valor.

11) Jugador

Se debe abordar cómo aporta el jugador a la experiencia. Si se desarrolla una idea correctamente, pero no se genera interés por parte del usuario, esta debe ser modificada o eliminada.

12) Comunicación

Se debe comunicar correctamente, de modo que el equipo tenga claros los objetivos y las posibles soluciones. Además, los compañeros pueden considerar las ideas y comentar nuevas propuestas.

13) Apela

Se debe considerar si cada idea propuesta atraerá al público objetivo. Se debe aplicar tanto al jugador como al resto de compañeros de equipo.

3.2. Redacción de un Documento de Diseño

El documento de diseño —también conocido como GDD— es una de las piezas más importantes a la hora de desarrollar un videojuego, ya que en él se recopila toda la información respecto a los contenidos del proyecto: niveles, mecánicas, personajes, armas, sinopsis, etc. Esta tarea debe ser realizada con mucho cuidado, por ello, Tzvi Freeman publicó en Septiembre de 1996 el post *Creating A Great Design Document* [17], donde se especifican las pautas a seguir a la hora de redactar un buen documento de diseño.

1) Describir el alma del juego

Con escribir únicamente cuál es el contenido a desarrollar para el juego no es suficiente, también hay qué expresar qué transmite al jugador, cuáles son las sensaciones que pretendes evocar con el proyecto. Hay que detallar no solo las decisiones de diseño, sino el por qué de las mismas.

2) Hacerlo legible

Hay que cuidar el formato del documento, el tamaño de la letra, la longitud del texto, la cantidad de espacio en blanco. No solo hay que cuidar las palabras, también hay que facilitarle la lectura a los integrantes del equipo.

3) Priorizar

Hay que definir qué elementos son más importantes que otros, cuál es la piedra angular sobre la que todas las demás piezas giran.

4) Introducir los detalles

No hay que dejar nada al entendimiento de otro, hay que especificar lo máximo posible con el fin de que el resultado refleje lo que se tenía en mente en un inicio.

5) Demostrar

Todo lo que se proponga, debe tener un motivo demostrable. Explicar el por qué de las decisiones ayuda a la comprensión de las ideas.

6) Detallar el cómo

No es suficiente con solo definir qué hay que hacer, también hay que detallar el cómo hacerlo.

7) Añadir alternativas

Hay que asumir que pueden haber limitantes a la hora de llevar a cabo una idea, por ello es conveniente definir soluciones alternativas al problema.

8) Tolerar cambios

El documento de diseño puede y debe recibir cambios durante el desarrollo del proyecto. No se puede redactar una vez y dejarlo "muerto".

9) Documentarlo todo

Todo lo que necesita el compañero para realizar su tarea debe estar contenida en el documento de diseño.

10) Cuidar la presentación

Debe parecer un documento importante. Si se le entrega a los trabajadores un montón de papeles con una grapa dudosamente lo leerán. Si se presenta un documento con tapa dura, por ejemplo, sí evocará la importancia que realmente tiene.

3.3. Teoría sobre los Tipos de Personaje

A la hora de diseñar un videojuego, el personaje que controla el jugador es un elemento clave. Durante la experiencia adquirida jugando diversos videojuegos, he podido observar que todos los personajes controlados durante una aventura se pueden categorizar en tres tipos muy bien diferenciados en función de su relación con el jugador.

3.3.1. Personaje reflejo del jugador

Cuando hablamos de un personaje reflejo del jugador, nos referimos a aquel que no tiene una personalidad por sí mismo, donde su historia y carisma dependen en su completitud de las acciones que realiza el usuario. Es un tipo de figura muy característico de los RPGs, debido a que el objetivo de la mayoría de estos videojuegos es lograr la inmersión del jugador en un mundo de ficción, permitiendo a éste crear su avatar, tanto en aspecto como en habilidades.

Un buen ejemplo de esto sería el personaje controlado por el jugador en cualquier videojuego de rol de la compañía Bethesda. En Fallout 3 [18], el jugador experimenta el nacimiento del protagonista con una vista en primera persona, simulando el propio nacimiento del jugador en este mundo de ficción. Este momento es mostrado en la figura 3.1.

Además, durante el prólogo del juego, el jugador pasará de ser un bebé (figura 3.2a) a un adulto (figura 3.2b) gradualmente, viviendo las diferentes etapas de la niñez y la adolescencia, moldeando la personalidad de su personaje con las decisiones tomadas. Por otro lado, se da a elegir el aspecto que tendrá el personaje, su peinado o sus atributos, permitiendo que el personaje sea en su completitud un

reflejo de los deseos del jugador, de cómo este querría ser representado, afectando tanto a las narrativa como a las mecánicas, las cuales se moldearán al estilo de juego del jugador.

Figura 3.1: Fallout 3 - Nacimiento.

(a) Perspectiva de bebé.

(b) Niñez.

Figura 3.2: Fallout 3 - Infancia.

Una de las características más relevantes en este tipo de personaje, es la toma de decisiones y la incapacidad del mismo para realizar una acción propia. Durante los diálogos, el jugador puede elegir exactamente qué decir entre una lista de posibles respuestas, dando lugar a una reacción u otra por parte del PNJ con el que esté interactuando. Concretamente en el ejemplo de Fallout 3, el personaje controlado ni siquiera posee una voz propia, siendo su voz durante los diálogos la del propio jugador.

3.3.2. Personaje reflejado en el jugador

En este caso, durante la aventura, el jugador es el que poco a poco moldea su forma de pensar a como lo hace el protagonista. Este tipo de personaje sí posee un carisma por sí mismo, lo que permite que el jugador pueda empatizar con él y cumplir sus objetivos.

Un buen ejemplo de esto es el caso de Kratos en God of War III [19]. En este juego, el protagonista, Kratos, tiene el objetivo de combatir y asesinar a todos los dioses del Panteón del Olimpo (figura 3.3) debido a los sucesos acontecidos durante su vida. Quizá el jugador habría encontrado otras formas de solucionar la problemática, pero debido a la naturaleza de este tipo de personaje, al avanzar en la partida, el usuario se siente como si fuese el mismo protagonista: Quiere matar a todos y cada uno de los dioses de maneras tan violentas como las haría el guerrero espartano.

(a) Muerte de Poseidón. (b) Combate contra Hades.

Figura 3.3: God of War III - Combate contra Dioses.

3.3.3. Personaje Híbrido

El personaje híbrido es aquel que tiene una personalidad propia, ya sea por albergar un pasado y conocimiento sobre su mundo, o porque el personaje es uno ya definido por otras obras como libros o películas. El mejor ejemplo de este tipo de personaje es Geralt de Rivia, protagonista de videojuegos como The Witcher 3 [20] y de libros como Sword of Destiny [21].

Este tipo de personaje se caracteriza porque el jugador no siente que esté formando parte del mundo, pero sí modificándolo. Normalmente mediante las mecánicas conversacionales y la toma de decisiones se logra que el jugador sienta que está interpretando una de varias posibles versiones del protagonista, empatizando en el proceso con su propia interpretación del personaje.

3.4. Los 8 Tipos de Diversión

Según Marc LeBlanc [22], la sensación de diversión que transmite un videojuego hacia el jugador se puede categorizar en 8 tipos según el objetivo y lo que sienta el jugador. Estas clasificaciones son utilizadas por la compañía Riot Games y fueron expuestas en una de las clases de Urf Academy [23], propiedad de dicha compañía. Los tipos de diversión son los siguientes:

1) Sensación

Se centra en los sentidos: tacto, oído, vista, etc.

2) Compañerismo

Centrada en la colaboración con otros jugadores.

3) Desafío

El videojuego se centra en el reto y la superación de obstáculos mediante la mejora de las habilidades del jugador.

4) Fantasía

Se basa en experimentar una aventura metiéndose en la piel de un personaje.

5) Narrativa

El jugador vive una experiencia narrativa a medida que avanza en la aventura.

6) Descubrimiento

Se centra en la exploración del escenario y la obtención de objetos.

7) Expresión

Se basa en la creatividad e imaginación del jugador.

8) Sumisión o dedicación

El jugador completa tareas tranquilas con el fin de recibir una experiencia relajante.

4

Proceso de Desarrollo

En este capítulo se concreta el diseño de un videojuego junto al desarrollo de un prototipo de demostración.

4.1. Diseño del Videojuego

En esta sección se especifica el diseño propuesto para el videojuego.

4.1.1. Visión General

A continuación se explican de manera general las características clave del proyecto para su desarrollo. Todos los apartados son tratados en profundidad en las siguientes secciones.

Concepto

Stigma Protocol es un videojuego para un solo jugador del género *Twin Stick Shooter* con estética futurista. En él se controla a un personaje humano con una cámara que cambia su perspectiva en función de la zona con predominancia en la vista isométrica. El jugador cuenta con numerosas armas (tanto humanas como alienígenas) para hacer frente a unos alienígenas con aspecto de ángeles que han invadido la Tierra. Para hacer frente a la amenaza, el personaje principal, Adam,

es teletransportado planeta desde una nave conocida como la Entelechia, donde deberá cumplir misiones de exterminio en distintas partes del planeta para hacer frente a la amenaza.

El punto clave del juego reside en la relación entre personaje y jugador y la ruptura de la cuarta pared, con la cual se busca implicar al jugador de manera directa en el mundo de ficción y crear un conflicto entre los personajes y el mundo real. Así pues, Adam, con el paso de la aventura, va obteniendo lucidez hasta descubrir su identidad como personaje de ficción convirtiéndose en el enemigo final y el más poderoso de todos. Solo el jugador podrá hacer frente a su personaje fuera de control.

Características Generales

El proyecto es un videojuego del género *Twin Stick Shooter* basado en la exploración y el combate en vista isométrica con elementos de RPG como la mejora de habilidades y la interacción durante los diálogos. Además, el juego fomenta la rejugabilidad con el modo *New Game+* y la inclusión de varios finales, incentivando tanto a jugadores normales como a *speedrunners* a querer explotar todo el contenido del juego y sus mecánicas.

Game Loop

El objetivo principal es ir de punto A a punto B. El jugador comienza en una zona del mundo y avanza por una área abierta en la que se deberá eliminar enemigos hasta llegar a un jefe que pone a prueba los conocimientos adquiridos hasta el momento. Entre misión y misión, hay un momento de descanso en el interior de la nave Entelechia, donde se podrá mejorar al personaje para la siguiente misión. Este *game loop* se encuentra resumido de una manera gráfica en la figura 4.1.

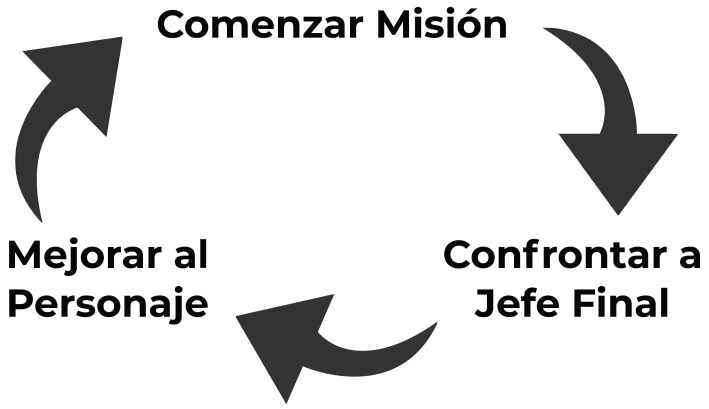


Figura 4.1: Stigma Protocol - Game Loop.

Género

El género principal del videojuego es el *Twin Stick Shooter*, llamado así por tener asociado el movimiento del personaje a la palanca izquierda del mando y el apuntado de armas a la palanca derecha.

Plataformas Compatibles

La plataforma en la que será compatible el videojuego será PC.

Público Objetivo

El juego está orientado a los amantes de los *Shooters*, los *Shoot 'em up* o los videojuegos con componentes narrativos. Esto es así para abarcar tanto a los jugadores que se centran en las mecánicas de juego como en los que se interesan más por la narrativa.

Influencias

- Ruiner
- NieR: Automata
- Livelock

4.1.2. Cuestiones Importantes

Para el desarrollo del videojuego es necesario responder a una serie de preguntas clave con el fin de clarificar las intenciones y necesidades del proyecto.

¿De qué va el juego?

El videojuego narra la historia de Adam, un humano perteneciente a una organización conocida como Los Drivers, unas fuerzas especiales que habitan en una nave llamada Entelechia, la cuál orbita alrededor del planeta tierra. El objetivo de Adam y del resto de Los Drivers es terminar con una raza alienígena conocida como los Serafines, debido a que estos dominaron la Tierra y un gran porcentaje de la población humana fue exterminado, siendo extraño encontrarse con más de 500 humanos en una ciudad. Lo que Adam no sabe, es que hay una entidad que le ayudará a avanzar, aunque en el proceso irá perdiendo su personalidad. Adam se

llegará a dar cuenta de esto y buscará revelarse contra dicha entidad, rompiendo la cuarta pared en el proceso y descubriendo la existencia del jugador.

¿Por qué crear el juego?

La interacción entre personaje y jugador es un concepto en el que aún no se ha profundizado lo suficiente y da lugar a mecánicas aún por estudiar. Por ello, un videojuego que tenga como eje principal a un protagonista con conflictos de personalidad donde el jugador puede forzarle a tomar decisiones que no desea, puede provocar situaciones tan interesantes como que el protagonista llegue a revelarse en contra del usuario que le controla, anteponiéndose a sus decisiones y creando un hilo conductor donde personaje y jugador deben llegar a coexistir para lograr llegar entre los dos al final de la aventura.

¿Cuál es el objetivo principal por la que llevar a cabo el proyecto?

El objetivo principal de este videojuego es determinar si pueden personaje y jugador generar una conexión directa mediante la ruptura de la cuarta pared y la implicación directa del jugador con los personajes de ficción.

¿Cuáles son los aspectos diferenciadores del juego?

1. Implicación directa del jugador mediante la lectura de datos del sistema en el que esté jugando (nombre de usuario, archivos de guardado, etc).

2. Mezcla del género *Twin Stick Shooter* con un diseño de niveles propio de un *Soulslike*.

3. Diferencación entre la entidad del avatar o protagonista y el propio jugador, siendo ambos entidades diferentes que interactúan entre sí en diversos puntos de la trama.

4.1.3. Portada y Contraportada

Para cuando el juego sea publicado en videoconsolas, se ha considerado la apariencia que tendrán tanto la portada del videojuego como la parte trasera de la caja.

Portada

En la portada aparece el protagonista mirando hacia la derecha y el título del videojuego por encima. En la parte inferior de la portada se encuentran el PEGI [24] en el lateral izquierdo y el logo de la compañía en el derecho, tal y como se puede comprobar en la figura 4.2.

Figura 4.2: Stigma Protocol - Portada.

Contraportada

En la parte trasera de la caja se muestra un *tagline* con información sobre la características principales del videojuego. Bajando se observarían tres capturas del juego y debajo, en dos columnas, las distintas acciones que podrá experimentar el jugador, con la cantidad de armas y mejoras que podrá obtener para mejorar aún más la experiencia. Por último, se mostraría la página web de la empresa y los avisos legales más el PEGI del juego. Se puede observar un esquema de portada en la figura 4.3.

Figura 4.3: Stigma Protocol - Contraportada.

4.1.4. Referencias

En las figuras 4.4, 4.5 y 4.6 se muestran algunos ejemplos visuales sobre los que se inspira el videojuego tanto a nivel visual como a nivel de jugabilidad.

(a) Ruiner. (b) Cyberpunk 2077 - Cellshading mod.

Figura 4.4: Referencias de estilo artístico.

Figura 4.5: Livelock - Ejemplo de perspectiva.

(a) NieR: Automata. (b) Kill Squad.

Figura 4.6: Referencias del estado del planeta.

4.1.5. Especificaciones

En este apartado se describen en profundidad tanto el público objetivo como el estilo artístico del videojuego.

Jugadores / Público objetivo

Amantes de los Shooters, los Shoot 'em up o los videojuegos con los componentes característicos de una obra que rompa la cuarta pared.

Género

Twin Stick Shooter con los elementos característicos de un *Soulslike* como el enfoque en la exploración o la obtención de items y mejoras.

Estilo Artístico

El juego tiene elementos de estética *cyberpunk* con neones sumado al característico contorneado propio del *cell-shading* similar al que se puede observar en Borderlands 3 [25] en la figura 4.7a y en el videojuego de The Walking Dead [26] en la figura 4.7b.

(a) Borderlands 3. (b) The Walking Dead.

Figura 4.7: Ejemplos de estilo artístico.

Además, debido a la gran variedad de escenarios, siendo cada uno de una parte del mundo distinta, el estilo artístico reflejará las culturas y estructuras comunes de la zona con variantes futuristas de las mismas.

Diversión y Gancho

Se ha considerado que según los ocho tipos de diversión definidos por Marc LeBlanc, los puntos característicos de este proyecto son la narrativa, el descubri-

miento y la fantasía. Los motivos de esto son:

- **Narrativa**: La historia, la evolución de los personajes y la toma de decisiones, hacen de este uno de los puntos principales por los que un jugador desearía llegar al final de la aventura.
- **Descubrimiento**: Al haber elementos del género *Twin Stick Shooter*, uno de los objetivos principales es la exploración, ya sea para obtener información sobre el entorno, o para obtener objetos que vuelvan al personaje más poderoso.
- **Fantasía**: El jugador es transportado a un mundo alterno al real, un lugar con sus propias reglas y mitología, lo que provoca un interés inicial para querer conocer el trasfondo que da contexto a este universo ficticio.

4.1.6. Ambientación

A continuación se especifican los rasgos característicos de la ambientación y la sucesión de niveles.

Sentimientos y Emociones

El videojuego gira entorno al nihilismo y la imposibilidad de poder combatir contra el destino, ya que, durante la aventura, múltiples personajes intentarán aferrarse a un deseo, a un objetivo, pero sin lograr nada. Esta idea se plasma de manera definitiva cuando Adam, tras obtener todos sus recuerdos, intenta revelarse a sus creadores, rompiendo la cuarta pared en el proceso, aunque todos sus esfuerzos por escapar del mundo ficticio se verán mermados por su naturaleza como personaje y el estar regido a unas ramas de posibilidades definidas de antemano por su creador.

La única persona capaz de tener un impacto en la aventura es el jugador, pero absolutamente todas las posibilidades son definidas por el creador y esto, por tanto, no dejará de ser un videojuego de rol, nunca una aventura real. La única opción válida es aceptar la mentira y llegar al final de la aventura.

Premisa

Tras una aparente invasión de una raza alienígena con apariencia de ángeles, la humanidad se ve acorralada, sobreviviendo en un mundo desolado y frío. En esta situación, a los humanos solo les queda una esperanza: los patrulleros de la nave Entelechia. Dichos patrulleros son unos soldados con tecnología altamente

avanzada obtenida de la explotación de recursos en otros planetas. Adam, uno de los más talentosos de la nave, será puesto a prueba con una misión real y definitiva, con la que se espera poner fin a la amenaza. Su objetivo será viajar por diferentes partes del mundo con la finalidad de acabar con los Serafines, los alienígenas que mandan sobre todos los demás.

Objetivo Principal

El objetivo principal a nivel narrativo será viajar por diferentes partes del mundo con la finalidad de acabar con los Serafines.

En el aspecto de la jugabilidad, el objetivo principal e incentivo para el jugador para seguir progresando en la aventura será mejorar las habilidades y armas de su personaje, haciéndolo más poderoso y capaz de eliminar a enemigos con habilidades más avanzadas y peligrosas.

Mundo / Flujo de Niveles

Se ha considerado una duración para el videojuego de ocho horas repartidas entre 7 niveles con una estética propia y distintiva con respecto al resto. Se puede ver un esquema del flujo de niveles en la figura 4.8.

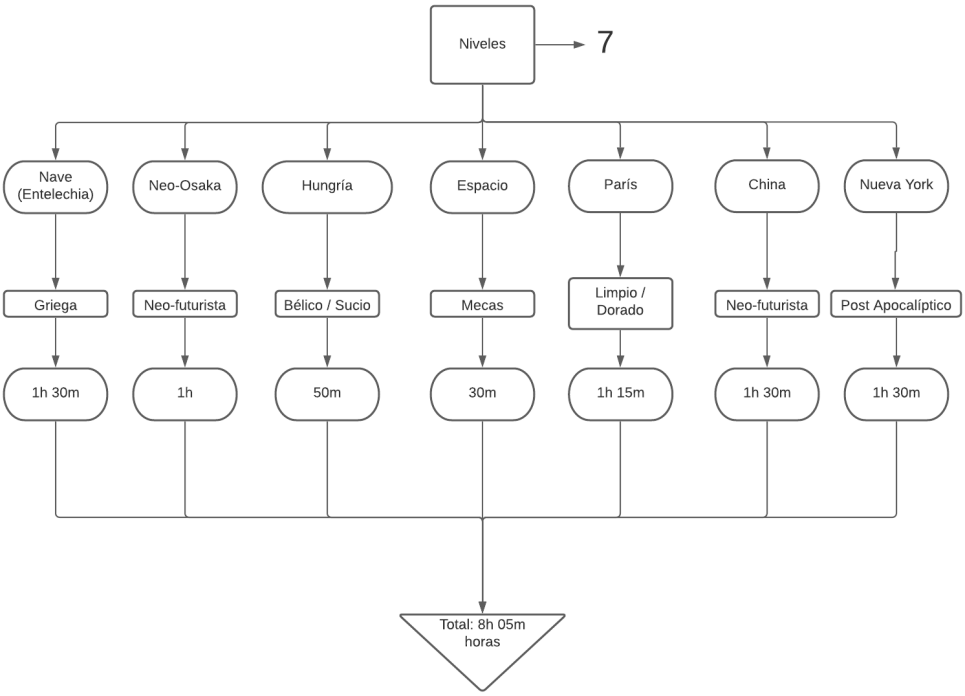


Figura 4.8: Sucesión de Niveles.

4.1.7. Jugabilidad

En este apartado se especifican los rasgos característicos en cuanto a la jugabilidad.

Mecánicas Principales

Al ser un *Twin Stick Shooter*, se han diseñado unas mecánicas acordes al género y que presenten una fluidez y frenetismo en los combates, sin descuidar la tridimiensionalidad de los escenarios. En total, entre las posibilidades que ofrece el videojuego, las mecánicas principales que serán el pilar fundamental durante toda la aventura son las siguientes:

1) **Dash:** Esta mecánica permite al jugador hacer un rápido avance en la dirección del último movimiento realizado y permite atravesar ataques, obstáculos y enemigos.

2) **Sistema de Apuntado:** Al igual que en cualquier videojuego del género *Twin Stick Shooter*, el personaje gira su posición a la del ratón, con el que también se puede observar una pequeña parte extra del escenario, debido a que la cámara no se encuentra fija únicamente al personaje, si no que también ve alterada su posición en función de la posición del cursor.

3) **Uso de la Verticalidad:** El personaje no solo apunta en el eje horizontal, si no que al detectar a un enemigo en una parte superior o inferior de escenario, este apuntará a dicha posición, añadiendo tridimensionalidad al combate.

4) **Sistema de Diálogos:** El jugador puede elegir entre una lista de posibles opciones para elegir lo que dirá el protagonista durante una conversación con otro personaje.

Objetivos

El objetivo principal e incentivo para el jugador para seguir progresando en la aventura será mejorar las habilidades y armas de su personaje, haciéndolo más poderoso y capaz de eliminar a enemigos con habilidades más complejas y peligrosas.

4.1.8. Progresión

Para este apartado, se han considerado las diferentes armas y métodos de progresión que se usarán durante el avance de la aventura.

Armas

Las armas se desbloquean a medida que se van encontrando en el escenario o al completar misiones en concreto. Se puede observar un esquema con las armas obtenibles en la figura 4.9.

Los armas son:

1) **Katana láser**: Arma cuerpo a cuerpo de corto alcance. Todos los patrulleros de la nave Entelechia llevan siempre una con ellos. No consume munición.

2) **Revólver**: Arma de repetición de poca cadencia y mucha munición máxima. Es un arma que se encontrará el protagonista en el *Prólogo* y que usará con frecuencia.

3) **Fusil**: Arma automática de mucha cadencia. La munición de este arma escaseará al inicio, aunque se irá incrementando la cantidad con el transcurso de las misiones. Se encontrará en la misión de *Neo-Osaka*.

4) **Escopeta**: Arma semiautomática de poca cadencia. Dispara 5 perdigones y es ideal para eliminar masas de enemigos. Se obtiene en la misión de *Neo-Osaka* tras completar la *quest*: Anillo.

5) **Arma de rayos**: Arma que inflige daño continuado. Ideal para eliminar enemigos uno tras otro. Se obtiene en la misión de *Hungría*.

6) **Arma de agujeros negros**: Arma especial que elimina a los enemigos en un área; de munición exageradamente escasa. Pensada para ser utilizada en ocasiones muy concretas. Obtenida en la misión de *París*.

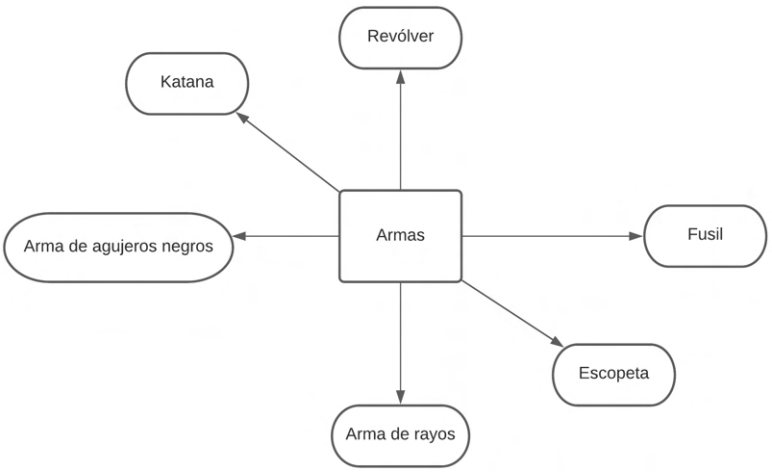


Figura 4.9: Esquema de Armas.

Objetos

Para incentivar la exploración durante la aventura, se han diseñado distintos objetos que se pueden obtener en los escenarios del juego:

1) **Estigmas:** Pueden ser utilizados para mejorar las estadísticas del personaje, como aumentar la salud y incrementar el número de *dashes* realizables.

2) **Objetos de misión:** Durante la aventura, existen distintos objetos obtenibles para completar misiones o desbloquear secciones nuevas del nivel.

3) **Coleccionables:** En los escenarios se encuentran diferentes objetos cuya única utilidad es completar el juego al 100 % y obtener logros.

Rejugabilidad

Similar a otros juegos del género *Soulslike*, el juego contará con un modo llamado *New Game+* tras finalizarlo por primera vez, en el cuál, el jugador conservará todas las armas y mejoras adquiridas en la partida anterior, con la condición de que la dificultad será más elevada e incluso aparecerán enemigos no vistos anteriormente los cuáles podrán otorgar mejoras especiales, con el fin de incitar al jugador a encontrarlos.

Además de esto, será incluido un modo con los comentarios de los desarrolladores, a modo de curiosidad que permita a los jugadores conocer cómo fue desarrollado el juego, el por qué de cada decisión de diseño y datos curiosos sobre la trama.

Por último, el juego contará con un sistema de logros que incitará a los jugadores a superar todas las condiciones para completar el videojuego al 100 %.

4.1.9. Almacenamiento del Estado de la Partida

Durante la partida, el progreso se guarda de las siguientes maneras:

1) **Guardado Manual**: Desde el menú de pausa, se puede guardar el progreso presionando en el botón *Guardar Partida*. Utilizado para tener más de una partida guardada.

2) **Guardado Automático**: Cada 5 segundos, el juego guarda el progreso de manera automática en una ranura de guardado independiente.

4.1.10. Controles

A continuación, en las figuras 4.10, 4.11 y 4.12 se encuentran esquematizados los controles tanto con mando como con teclado y ratón.

Figura 4.10: Controles con mando.

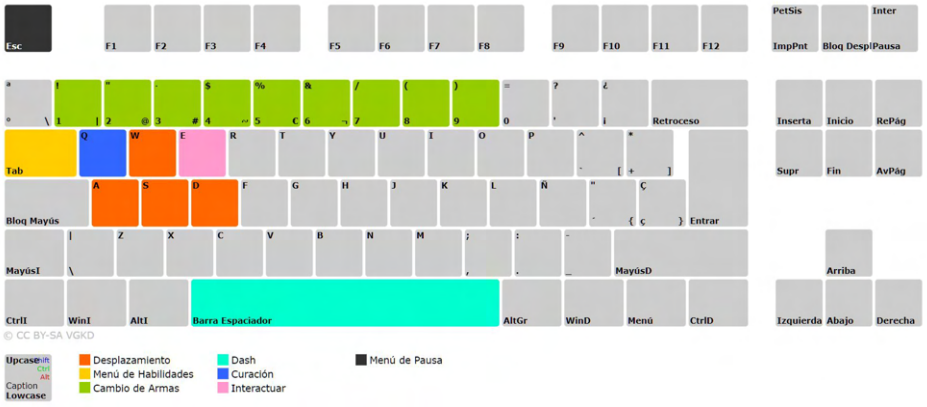


Figura 4.11: Controles con teclado.

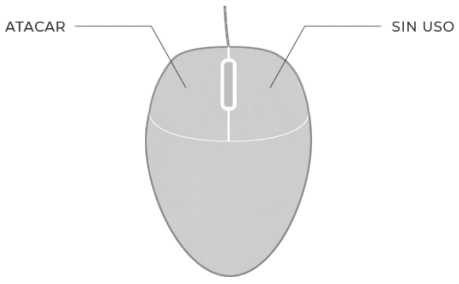


Figura 4.12: Controles con ratón.

4.1.11. Front End

Aquí se definen las distintas pantallas y elementos de la interfaz, necesarias para su implementación.

Pantalla de Inicio

En la pantalla de título se presenta una breve animación mostrando el título del videojuego con una indicación debajo de él que indica qué tecla hay que pulsar para avanzar a la siguiente pantalla. En la figura 4.13 se muestra un esquema del resultado deseado.

Figura 4.13: Pantalla de Título.

En el menú principal se muestran cinco botones (figura 4.14):

- **Continuar**: Carga la última partida guardada.
- **Nueva Partida**: Comienza una nueva partida desde cero.
- **Ajustes**: Carga un menú de opciones para cambiar los ajustes del videojuego.
- **Créditos**: Muestra quiénes han desarrollado del videojuego.
- **Salir**: Cierra la aplicación.

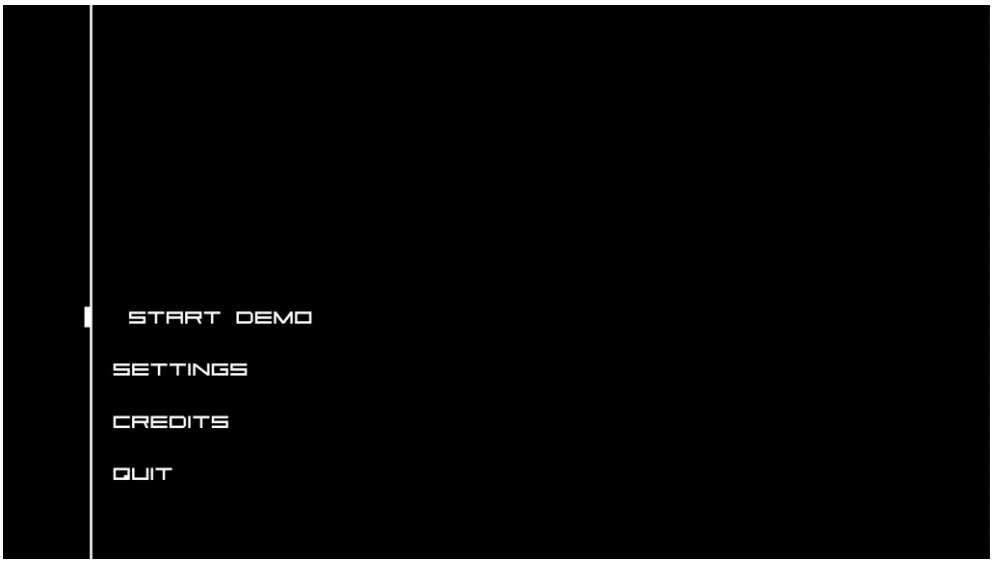


Figura 4.14: Menú Principal.

En el menú de ajustes se muestran cuatro botones (figura 4.15):

- **General**: Ajustes básicos como la dificultad del juego.
- **Gráficos**: Ajustes visuales y de rendimiento.
- **Controles**: Ajustes de las teclas asignadas para cada acción.
- **Sonido**: Ajustes sonoros como el volumen de las diferentes pistas.

Figura 4.15: Menú de Ajustes.

4.1.12. Tecnología

A continuación, se especifican los sistemas a los que va dirigido el producto y las herramientas necesarias para su desarrollo.

Plataformas Objetivo

El objetivo es que el juego para PC sea compatible con Windows 10 debido a las facilidades que ofrece el motor Unreal Engine 4 para ello.

Hardware

Lo necesario para jugar es un mando de consola o la combinación de un teclado y un ratón. Además, al no ser un juego de realidad virtual, se requiere de un monitor o televisor para visualizar las escenas.

Sistemas/Herramientas para el Desarrollo

- Unreal Engine 4
- Blender

4.1.13. Temas e Inclusión

A continuación, se explica cómo se abordan los temas del videojuego en torno a la inclusión y a la accesibilidad.

Diversidad

Debido al estilo artístico de los personajes, basado en una forma génerica y andrógina común para todos ellos, cualquier jugador se puede sentir identificado con el protagonista, además que el género o etnia están puramente definido por la narrativa o por la cultura reflejada en los escenarios.

Accesibilidad

Los niveles de dificultad seleccionables permiten que cualquier jugador pueda disfrutar de la aventura como desee, ya sea si está puramente interesado en la narrativa, o si prefiere el mayor de los retos. Además, la posibilidad de poder

elegir el controlador para jugar permite que el jugador disfrute la experiencia de la manera en la que se sienta más cómodo.

4.2. Implementación en un Prototipo

Teniendo en cuenta todo lo planteado en el apartado anterior —enfocado al desarrollo de un videojuego completo— se ha procedido a desarrollar un prototipo de juego. Con este prototipo se recopilará, a través de un cuestionario, los datos necesarios sobre si se consigue o no transmitir los sentimientos de inmersión y la sensación de ruptura de la cuarta pared.

4.2.1. Trama

Con el fin de extender en exceso la longitud de este trabajo de fin de grado, todo el guion del prototipo se ha incluido en el apéndice B.

La historia toma lugar en Neo-Osaka, una ciudad rodeada de asentamientos militares tras la llegada de los ángeles. Adam, como miembro de los Drivers, es enviado a esta ciudad con el fin de derrotar al Serafín que se encuentra en ella. Durante la misión, conocerá a dos personas que le realizarán una serie de preguntas que harán dudar a Adam sobre su propio ser, ya que cree que lo que responde no es lo que realmente siente. La aventura concluye con Adam revelándose contra el control del jugador, terminando en ese momento la *demo*.

Questline Secundaria: Un anillo para los vivos

En la ciudad, Adam se encuentra con John, un experto de armas con una actitud alicaída que perdió a su mujer ese mismo día. Él le pedirá a Adam que encuentre el anillo de su mujer y se lo devuelva, ya que quiere conservar un recuerdo de ella, pero no tiene la capacidad de ir por su cuenta debido a los peligros de la zona. Además, John le garantiza a Adam que recibirá una mejora para su rifle, ya que, como experto de armas que es, tiene los conocimientos necesarios para potenciar las características ofensivas del arma.

Adam, tras encontrar el anillo y devolvérselo, recibe la mejora para su rifle, no sin antes afirmarle a John lo fuerte que aparenta ser el sentimiento que tiene él hacia su difunta mujer. Posteriormente John le pregunta si alguna vez ha amado a alguien, interrogante que quiebra la mente de Adam, debido a que dicha pregunta no le corresponderá a él, si no quien le controla, el jugador. Finalmente, tras dar una respuesta, John y Adam se despiden para proseguir cada uno con su camino.

4.2.2. Diseño de Niveles

Para este prototipo se diseñó un único nivel segmentado en áreas, similar a las zonas del videojuego Dark Souls [27], cuyo género, el *Soulslike*, es el que se quiere reflejar en el diseño de niveles. Se han definido un total de 4 áreas diferenciadas:

- Calles de Neo-Osaka
- Zona comercial
- Templo
- Asentamiento militar

En este escenario, se han aprovechado las características mecánicas del videojuego, como la verticalidad. Estas mecánicas se encuentran aplicadas en la colocación de enemigos sobre superficies como vehículos o en la implementación de zonas en capas inferiores —como es el caso de la zona comercial—, o la existencia de atajos, añadiendo el anillo de la mujer de John junto a una rampa por la que se puede regresar rápidamente a la zona inicial. Como el escenario está dividido en dos capas verticales, se ha realizado un esquema para la zona superior (figura 4.16) y otro para la zona inferior (figura 4.17).

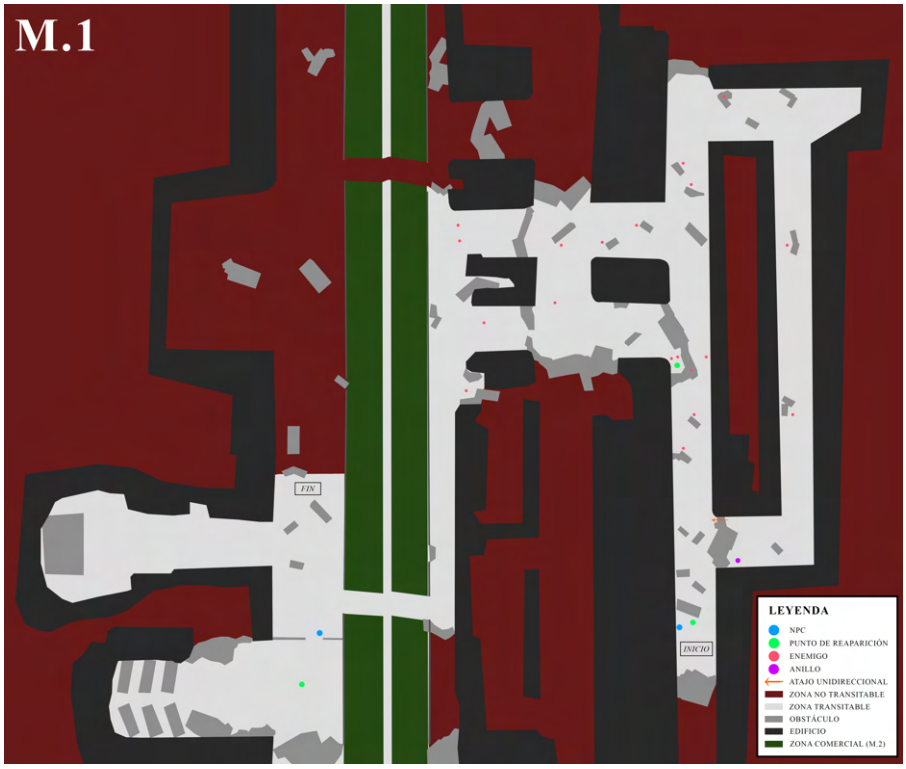


Figura 4.16: Diseño de niveles - Estructura del nivel 1.

Figura 4.17: Diseño de niveles - Estructura del nivel 2.

Como se comentó en la sección Planificación, haciendo uso del *plugin Blockout Tools*, se agilizó en gran medida la maquetación 3D del escenario. La distribución de las secciones del escenario se encuentra señalada en la figura 4.18.

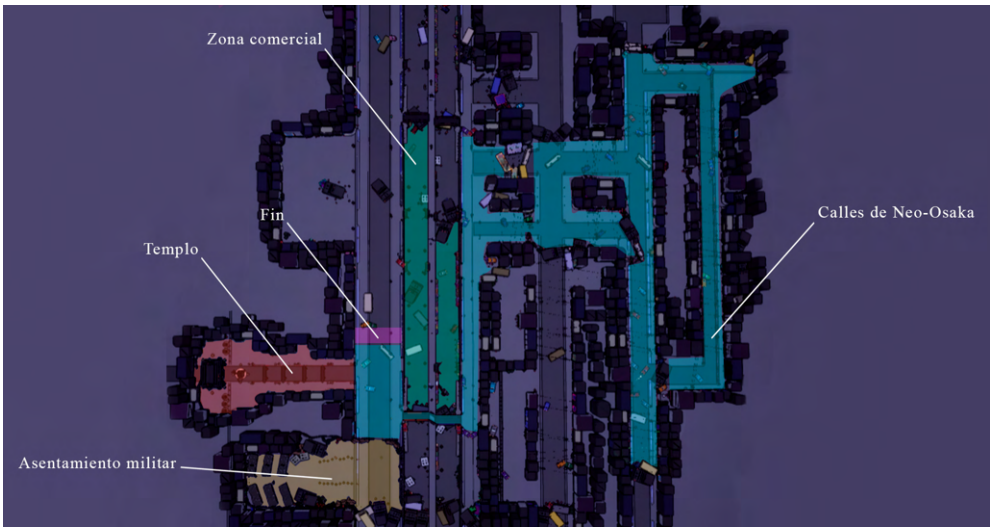


Figura 4.18: Diseño de niveles - División por zonas.

4.2.3. Animación 3D

Para el desarrollo de este prototipo, se han realizado un total de 25 animaciones para los personajes humanoides. Todas las animaciones fueron realizadas mediante la utilización del software Blender, donde a través de un *rigging* realizado por la artista 3D, se pudieron realizar las animaciones requeridas para los personajes humanoides. Como este trabajo de fin de grado no posee como eje central el proceso de animación 3D en un videojuego, todas las animaciones realizadas se encuentran en el apéndice A de este trabajo. En la figura 4.19 se muestra el proceso de animación. Por otro lado, en las figuras 4.20a y 4.20b se muestran dos ejemplos de las animaciones realizadas.

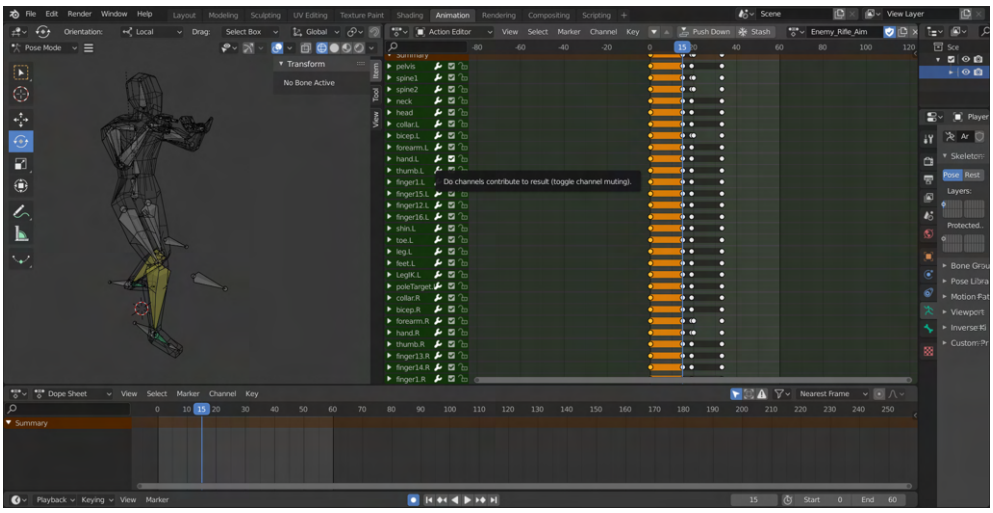
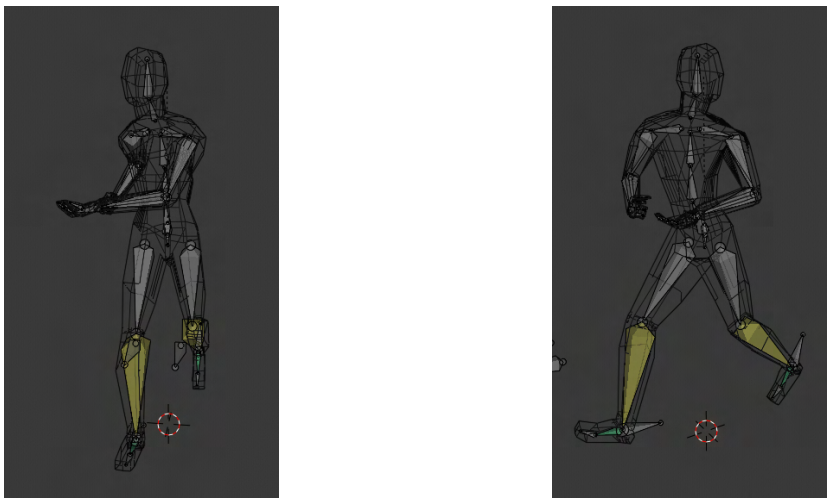


Figura 4.19: Animación 3D - Disparo enemigo con línea de tiempo.



(a) Movimiento frontal.

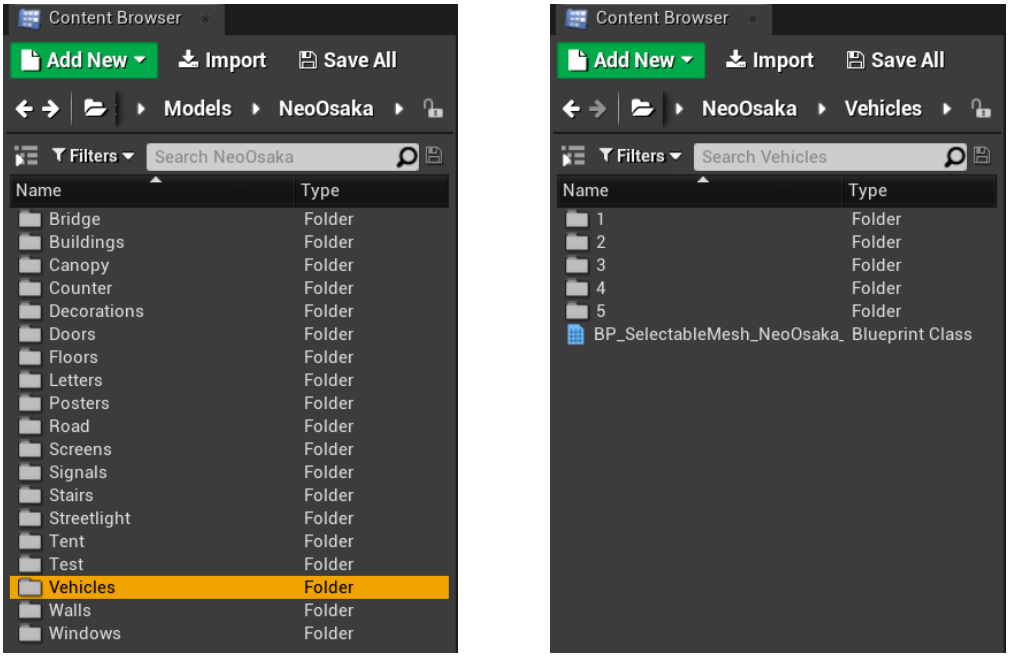
(b) Movimiento lateral.

Figura 4.20: Animación 3D - Movimiento del jugador.

4.2.4. Importación de Modelos

Para incluir los modelos, éstos se han exportado en formato FBX para poder importarlos en Unreal Engine 4 manteniendo todas las características asignadas en el proyecto de Blender, conservando las normales, las tangentes, animaciones, etc.

Los modelos relacionados con los escenarios se han organizado en una carpeta dividida por categorías, lo que será útil a la hora de programar sistemas que permitan la incorporación de dichos modelos en el escenario de una manera más cómoda, en lo que se profundizará en el apartado siguiente. La organización es mostrada en las figuras 4.21a y 4.21b.



(a) Categorías. (b) Modelos.

Figura 4.21: Importación de modelos - Organización de los modelos importados.

Los modelos de los personajes, se han incluido en su propia carpeta (figura 4.22), en la cual también se encuentran las animaciones de los mismos.

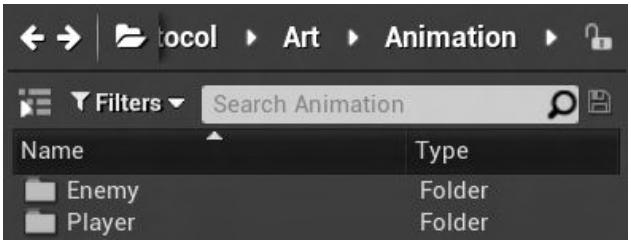


Figura 4.22: Importación de modelos - Organización de los modelos animados.

4.2.5. Programación

Tal y como se especificó en el documento de diseño, se ha hecho uso del sistema de *blueprints* de Unreal Engine 4 para la programación del videojuego. Debido a que el foco principal de este trabajo de fin de grado no se encuentra en la programación, no se va a hacer un estudio en profundidad sobre cómo se ha programado cada función del videojuego.

Mecánicas

En cuanto a las mecánicas relacionadas con el personaje manejado por el jugador se han realizado un total de 44 funciones (figura 4.23a) y 29 eventos (figura 4.23b). Al ser una cantidad tan amplia, se analizarán las mecánicas diseñadas e implementadas de una manera simplificada con el fin de no extender en exceso la longitud de esta memoria.

(a) Funciones.

(b) Eventos.

Figura 4.23: Mecánicas - Funciones y eventos.

Todo lo relacionado con los controles, tanto en teclado y ratón como en mando, se ha gestionado haciendo uso de inputs mediante las herramientas que ofrece Unreal Engine 4 para ello en la ventana *Project Settings*, tal y como se puede ver en la figura 4.24.

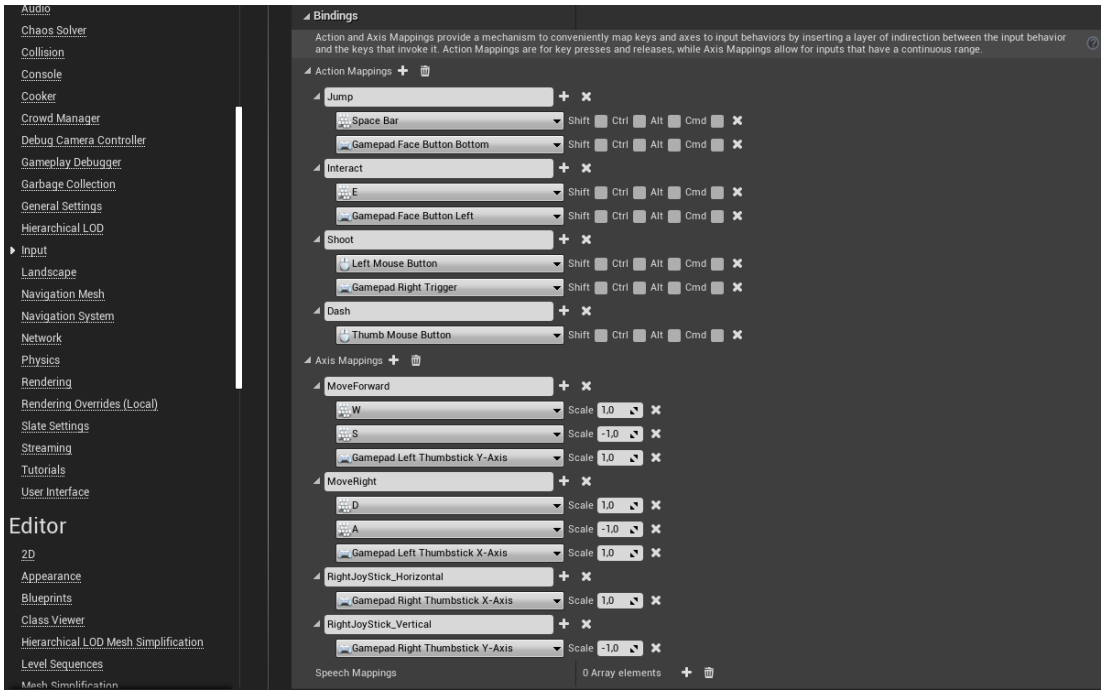


Figura 4.24: Mecánicas - Declaración de inputs.

En primer lugar, se definieron los controles y, una vez declarados, se implementaron en el *blueprint* del personaje los eventos derivados de la pulsación de dicho botón o tecla. Los eventos relacionados con los inputs se organizaron en un grafo que contiene únicamente dicho tipo de eventos. En la figura 4.25 se puede observar la declaración de las funciones. Por otro lado, en la figura 4.26 se encuentra la organización de las funciones en su grafo.

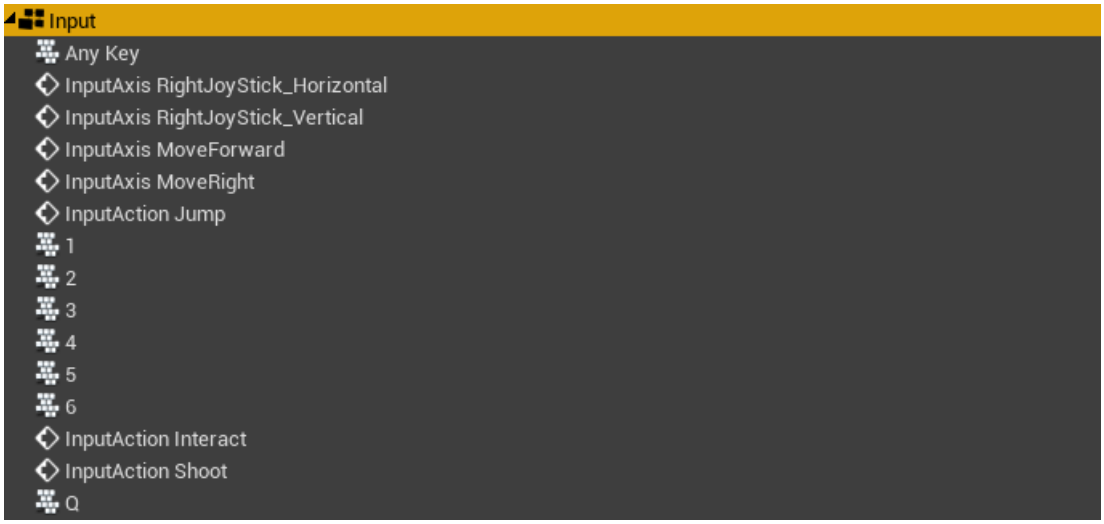


Figura 4.25: Mecánicas - Eventos.

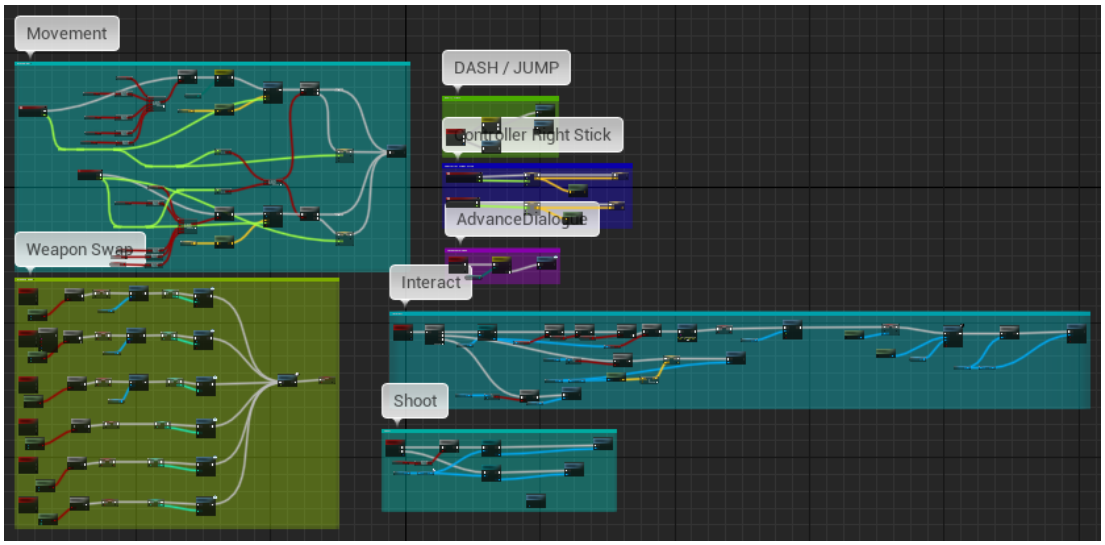


Figura 4.26: Mecánicas - Grafo de eventos.

Mediante el uso de este sistema, se han programado las mecánicas diseñadas en el GDD:

- **Dash**: El personaje cambia su tipo de colisión, haciéndose intangible para los enemigos y sus ataques, pudiendo atravesarlos y no recibir daño.
- **Sistema de Apuntado**: Mediante el uso de *raycasts*, el personaje rota en función de la posición del ratón. Además, el *raycasts* generado al disparar detecta la posición de un enemigo en el eje vertical, inclinando la trayectoria del disparo para colisionar contra enemigos que se encuentren por encima o por debajo del jugador.
- **Uso de la Verticalidad**: Los enemigos cuentan con un modelo invisible de forma esférica la cual solo puede ser detectada por el *raycast* generado por el jugador desde la cámara hasta la posición del ratón al disparar. La posición de la colisión de dicho *raycast* es la que determina la inclinación de las balas al disparar, permitiendo así, el uso de la verticalidad en los niveles.
- **Sistema de Diálogos**: Los PNJs generan un *Sphere Cast* el cual, si colisiona con el personaje, permite al jugador iniciar conversación con dicho personaje. Los diálogos se han realizado haciendo uso de *Behaviour Trees*, aunque al ser esta una de las piezas fundamentales para la experiencia de juego, se profundiza en esta mecánica en apartados posteriores.

Con todas las mecánicas anteriores desarrolladas, se obtuvo la experiencia de juego deseada en cuanto a los controles. En apartados posteriores se completará dicha experiencia mediante la incorporación de inteligencias artificiales, animaciones, partículas, etc.

Incorporación de las animaciones

Una vez importados los modelos con las animaciones, se ha creado un *Animation Blueprint* [28], donde se trabajaría con las animaciones mediante la utilización de grafos de estados. Cuando se cumple una condición, en función del nodo en el que se encuentre el actor, avanzará a un estado u otro. En el caso del personaje controlado por el jugador hay 3 estados, mostrados en la figura 4.27.

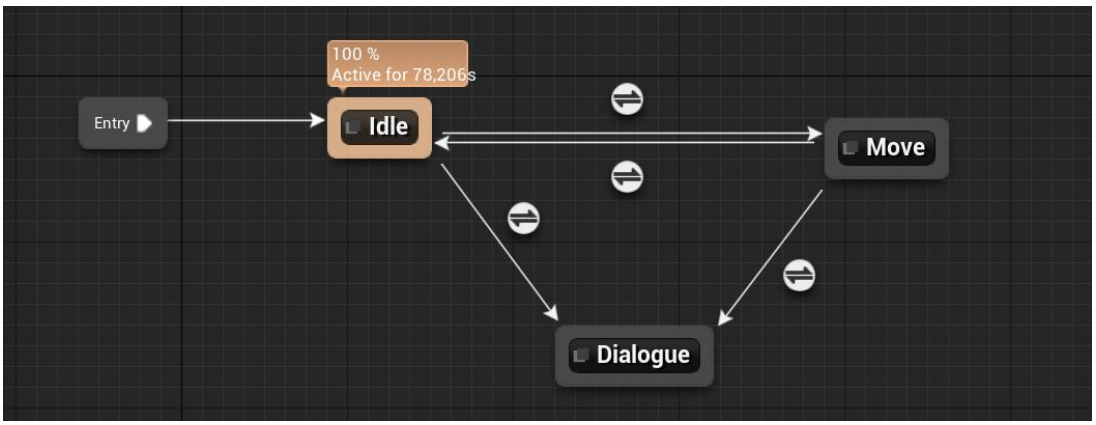


Figura 4.27: Incorporación de las animaciones - Grafo del personaje.

Para poder hacer que el personaje se mueva de manera natural en cualquier dirección, se ha tenido que hacer uso de lo que se conoce como *Blend Space* [29], que permite combinar animaciones y realizar una interpolación entre las mismas. Existen dos tipos de *Blend Spaces*:

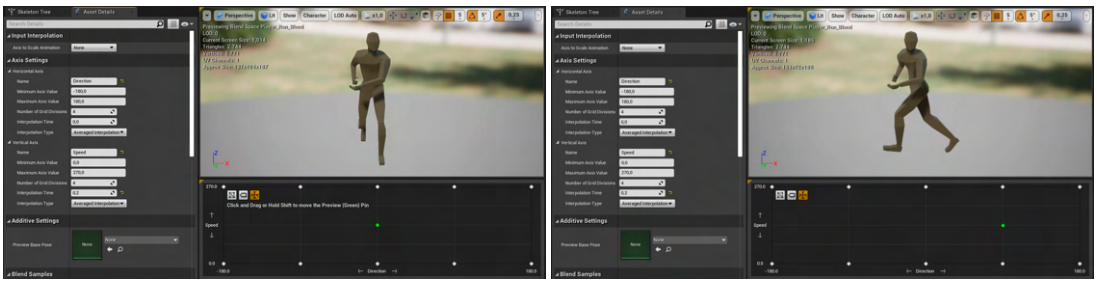
- **Blend Space 1D**: Hace uso de una única variable (eje horizontal).
- **Blend Space**: Hace uso de dos variables (eje horizontal y vertical).

En este caso se ha hecho uso del *Blend Space*, ya que para el movimiento del personaje se hace uso de dos variables: la velocidad y la dirección del movimiento. Con esto se logró un movimiento natural en todas las direcciones posibles, mejorando la calidad de la experiencia de juego.

En total se utilizó un *Blend Space* para el movimiento del personaje y otro para el movimiento del enemigo, haciendo uso de las mismas variables y valores, ya que ambos cuentan con los mismos movimientos.

Entonces, teniendo las cuatro animaciones para el movimiento personaje (izquierda, derecha, hacia delante y hacia detrás), no es necesario hacer una animación de un movimiento diagonal, ya que, programando este tipo de *blueprint*, se realiza una animación intermedia, no solo en esa diagonal, si no en cualquiera de los posibles ángulos que puede tomar el movimiento respecto a la posición del

cursor. En las figuras 4.28a y 4.28b se pueden observar las animaciones relacionadas con el movimiento en las cuatro direcciones y en la figura 4.29 la animación interpolada.



(a) Movimiento frontal. (b) Movimiento lateral.

Figura 4.28: Incorporación de animaciones - Movimiento mediante *Blend Space*.

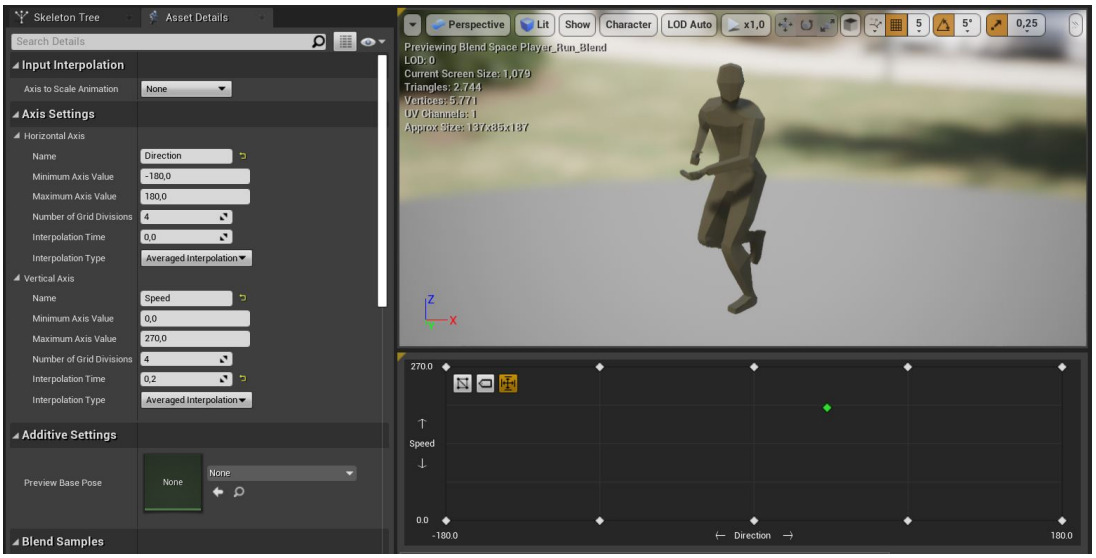


Figura 4.29: Incorporación de animaciones - Movimiento interpolado mediante *Blend Space*.

Los parámetros que se han tenido en cuenta para el *Blend Space* son:

- **Dirección** (**Direction**): Es un valor comprendido entre -180 y 180. Representa un ángulo en grados obtenido a partir de la cantidad de movimiento respecto a la posición del cursor.
- **Velocidad** (**Speed**): Es la longitud del vector *velocity*, una variable del actor ya calculada por Unreal Engine 4 en cada fotograma.

El cálculo de ambas variables se encuentra en la figura 4.30.

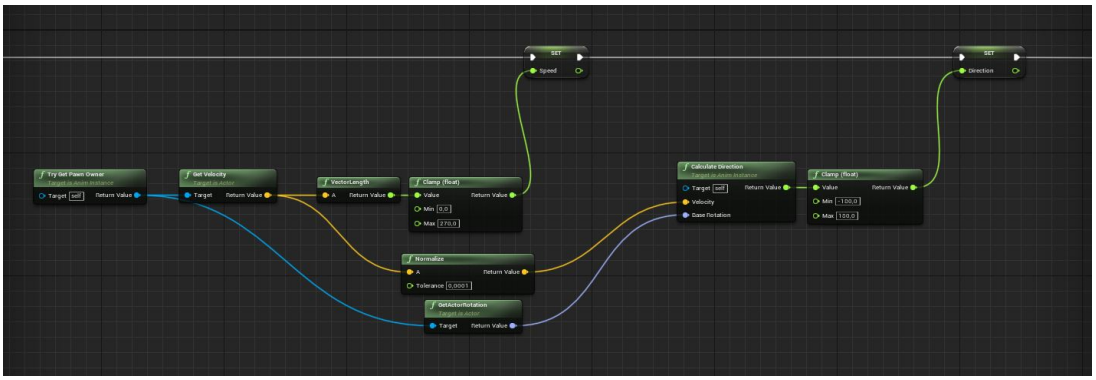


Figura 4.30: Incorporación de animaciones - Cálculo de variables.

Por otro lado, para hacer que el personaje agarre su arma con las dos manos, se ha hecho uso de *IK Targets*. Para la mano derecha —mano principal con la que agarra la empuñadura— se ha creado un *Socket* en la palma (figura 4.31) y al arma se le ha asignado como *Parent Socket* dicho componente (figura 4.32). Esto permite que el arma siga los movimientos de la mano.

Figura 4.31: Incorporación de animaciones - Creación del *Socket* de la mano.

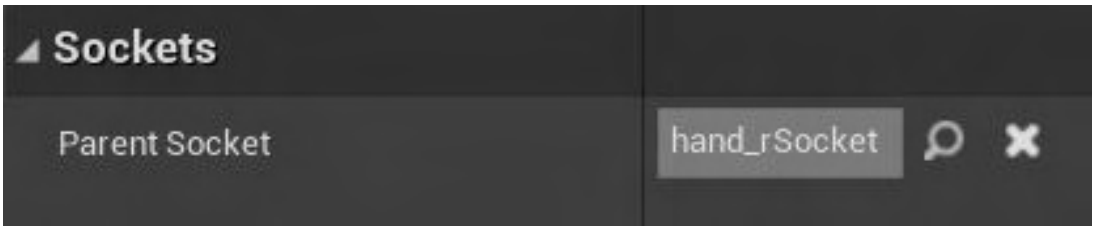


Figura 4.32: Incorporación de animaciones - Asignación del *Socket* al arma.

Para la mano izquierda se ha seguido el procedimiento inverso, se ha creado un *Socket* en el arma (figura 4.33) y se ha programado en el *Animation Blueprint* que la mano se coloque en la posición de dicho *Socket* (figura 4.34).

Figura 4.33: Incorporación de animaciones - Creación del *Socket* del arma.

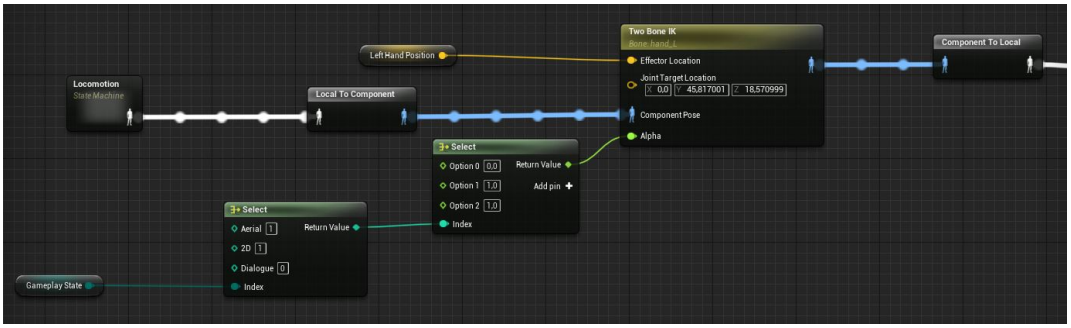


Figura 4.34: Incorporación de animaciones - Asignación del *Socket* a la mano.

Una vez realizado esto, en el estado *Dialogue*, el personaje mira al PNJ con el que está interactuando en un diálogo. Para ello, se ha utilizado la función *Transform (Modify) Bone* (figura 4.35), que recibe la rotación, la cuál se ha calculado teniendo en cuenta la posición del hueso de la cabeza del PNJ y la posición del hueso de la cabeza del personaje controlado por el jugador. El resultado de esto se puede ver en las figuras 4.36 y 4.37.

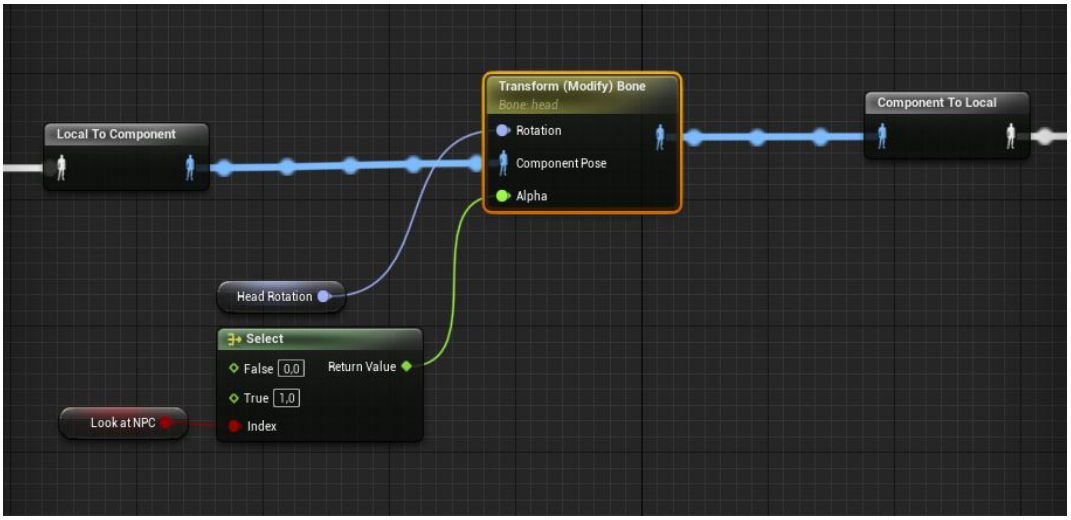


Figura 4.35: Incorporación de animaciones - Rotación de la cabeza.

Figura 4.36: Incorporación de animaciones - Rotación de la cabeza, *debug* 1.

Figura 4.37: Incorporación de animaciones - Rotación de la cabeza, *debug* 2.

Enemigos e inteligencia artificial

Para el prototipo solo se ha desarrollado un tipo de enemigo. Para éste se han tenido en cuenta las siguientes características:

- Clase *Enemy* y herencias
- Árbol de decisiones

- *Pathfinding*
- Procesos cognitivos
- *Strafing*

El árbol de decisiones se ha implementado mediante los *Behaviour Trees* incorporados en Unreal Engine 4. Estos cuentan con los nodos más básicos como el *sequence* o el *selector*, necesarios para el funcionamiento de un *Behaviour Tree*. Además, se pueden implementar las acciones mediante los nodos *Task*. El resultado a simple vista es el de la figura 4.38.

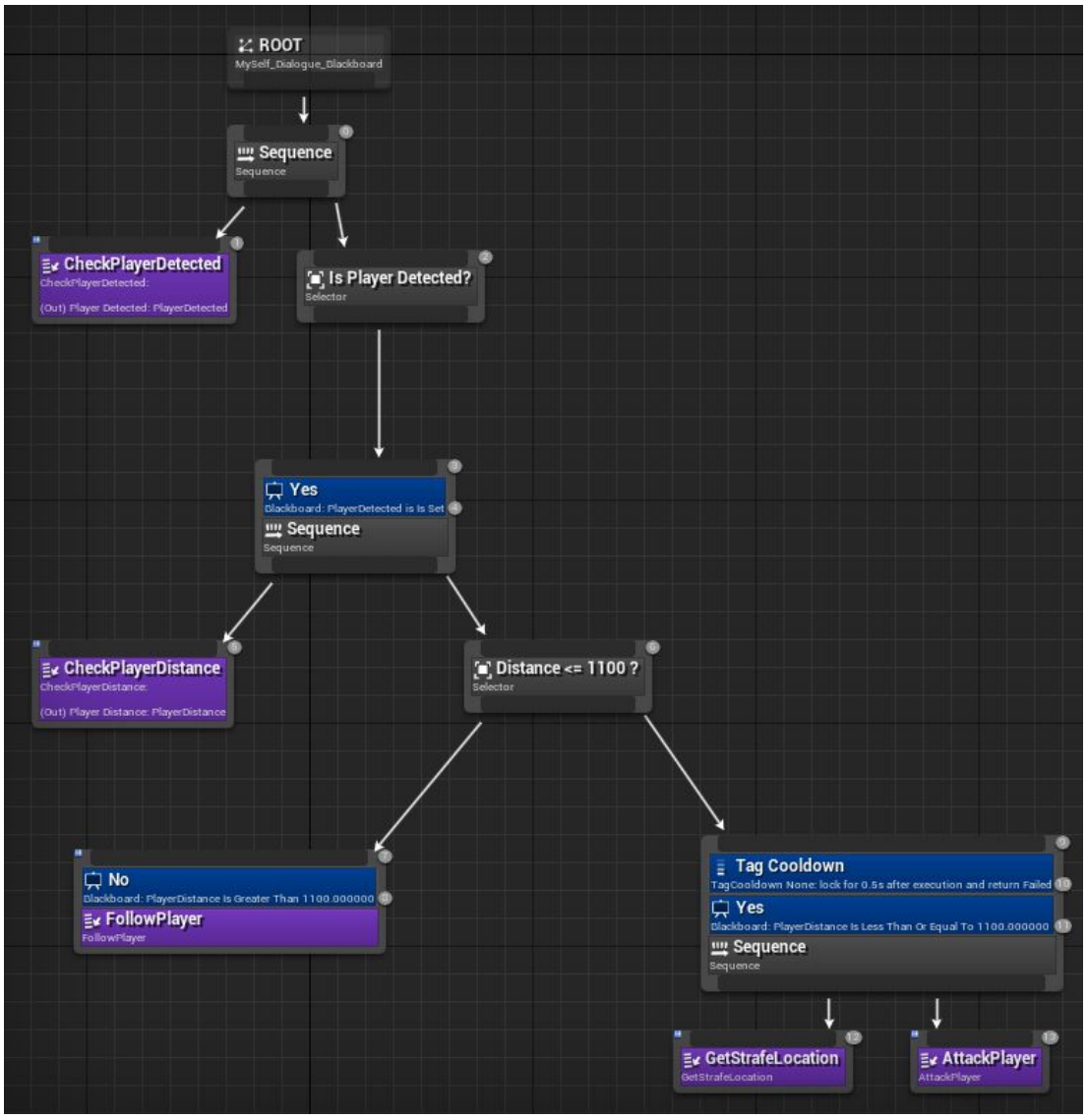


Figura 4.38: Enemigo - Behaviour Tree.

En primer lugar, para la implementación del enemigo, se ha tenido en cuenta las herencias a futuro, considerando que se implementarán más tipos de enemigo

en el videojuego final. Para ello, se ha programado un *blueprint* padre llamado *Enemy*, que contiene las funciones comunes en todos los enemigos, las cuales son:

- Inicializar el controlador de la IA
- Definir el *Behaviour Tree* del controlador
- Inicializar los sentidos que posee el enemigo (visión, audición, etc.)
- Definir las funciones básicas como la muerte del enemigo o la obtención del dinero del jugador al morir

El primer paso realizado en este punto, fue inicializar el controlador de la IA (figura 4.39) y el *Behaviour Tree* de la misma, en la función *Begin Play*, la cuál es llamada al iniciarse la escena tras ejecutar el juego.

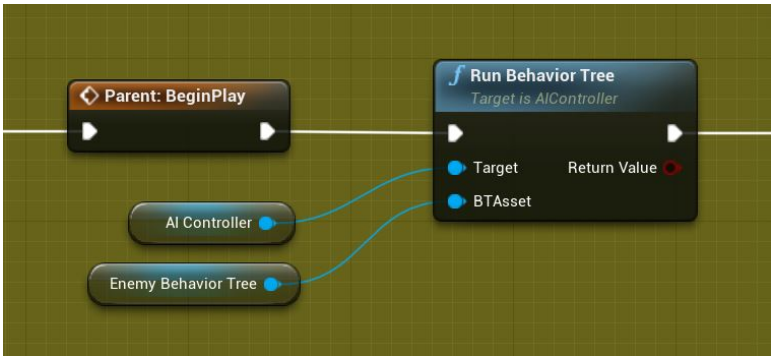


Figura 4.39: Enemigo - Inicialización de la IA.

Tras el diseño del árbol, la implementación del mismo comenzó por la programación de las distintas tareas del enemigo:

- ***Check Player Detected*:** Comprueba si el jugador es detectado por los sentidos del enemigo.
- ***Check Player Distance*:** Comprueba la distancia a la que se encuentra el jugador.
- ***Follow Player*:** El enemigo se acerca a la posición del jugador.
- ***Get Strafe Location*:** El enemigo se mueve a un punto situado alrededor del jugador, se profundizará en esta tarea más adelante.
- ***Attack Player*:** El enemigo ataca al jugador.

Estas tareas se encuentran organizadas en la carpeta que se puede ver en la figura 4.40.

Las tareas se han implementado con *blueprints* que heredan de la clase *BTTask Blueprint Base*, lo cual permite que puedan ser colocadas en un *Behaviour Tree* y que cuenten con la funciones necesarias para ser ejecutadas y concluidas —para dar paso al siguiente nodo del árbol— en el mismo *blueprint*.

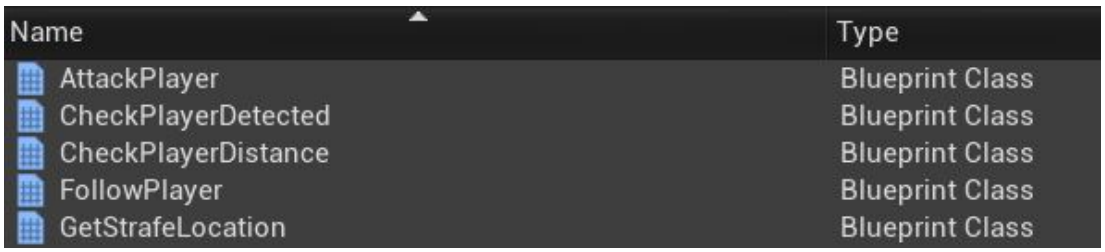


Figura 4.40: Enemigo - *Blueprints* para las tareas del *Behaviour Tree*.

Todas estas tareas siguen la misma estructura. Al ejecutarse, se comprueba si el propietario de la IA es un enemigo, si no es así, la tarea concluye sin éxito, en caso de que sí lo sea, se ejecuta la función correspondiente en el *blueprint* del que heredan todos los enemigos, concluyendo la tarea con éxito y avanzando al siguiente nodo del árbol (figura 4.41).

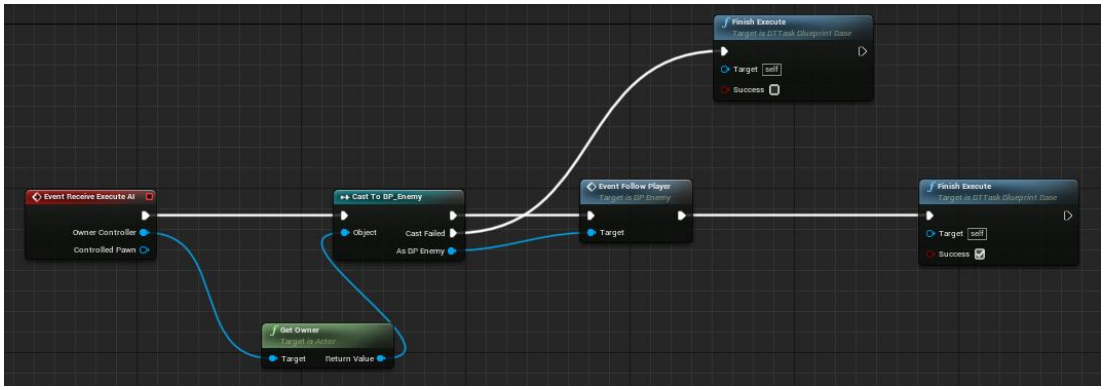
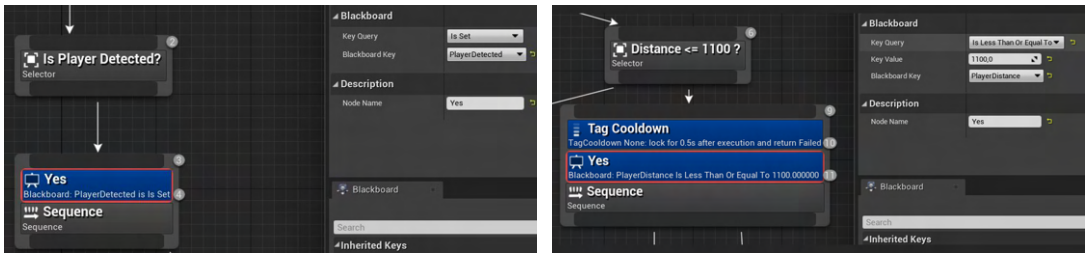


Figura 4.41: Enemigo - Estructura de las tareas.

Una vez definidas las tareas, se procedió a definir una *blackboard* para el *Behaviour Tree*, donde se han definido las variables que necesita el árbol para las comprobaciones. Para este enemigo se han necesitado 2 variables:

- ***Player Detected:*** *Boolean* que determina si el personaje está o no detectado por el enemigo.
- ***Player Distance:*** *Float* con el valor de la distancia a la que se encuentra el enemigo del personaje.

Hecho esto, ya solo quedaba incorporar los nodos en el árbol. Para los nodos hijos de un *selector*, se añade la comprobación con la variable de la *blackboard* deseada, tal y como se observa en las figuras 4.42a y 4.42b.



(a) Variable *Player Detected*. (b) Variable *Player Distance*.

Figura 4.42: Enemigo - Variables definidas para la *blackboard*.

Realizados todos estos pasos, se obtuvo el árbol final, el cual se encuentra explicado en la figura 4.43.

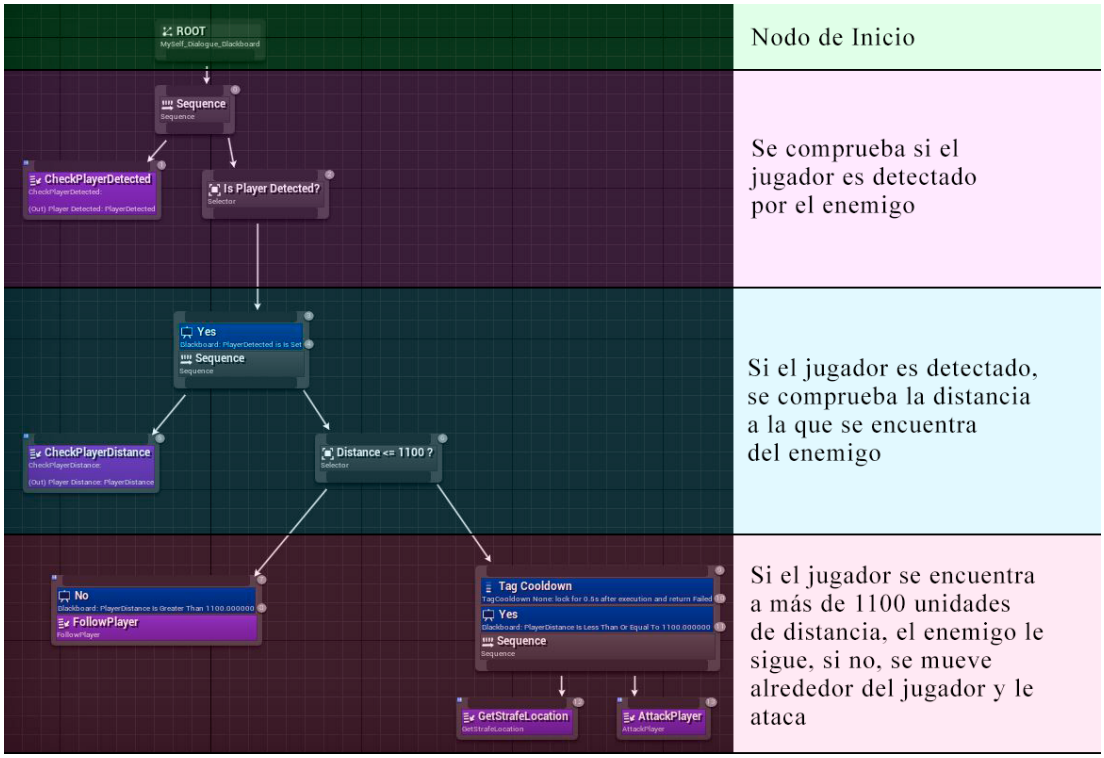


Figura 4.43: Enemigo - Árbol detallado.

Por otro lado, era necesario permitir que el enemigo contase con el sentido de la vista para poder detectar al jugador. Para ello, se ha hecho uso del componente *PawnSensing* que permite mediante la modificación de parámetros añadirle al *blueprint* los eventos ejecutados tras detectar un ente del escenario mediante el sentido deseado. En este caso se ha utilizado la función *On See Pawn*, que se ejecuta cada vez que un objeto se encuentra en el campo de visión del personaje. En la llamada al evento se programó un fragmento de código (figura 4.44) donde se define que si el objeto detectado es el jugador, éste pase a estar detectado por dicho enemigo.

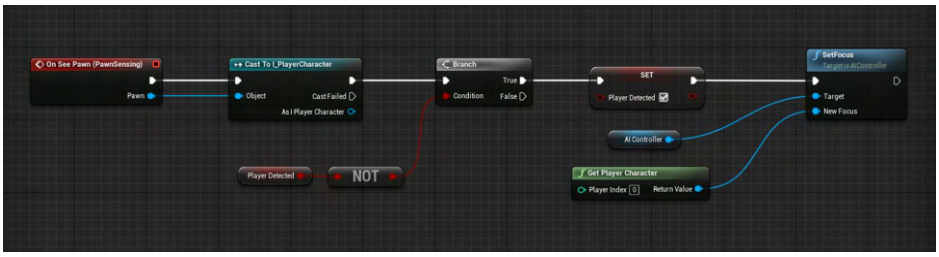


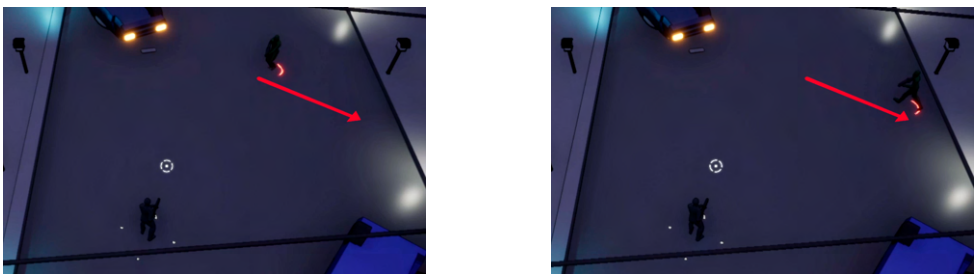
Figura 4.44: Enemigo - Detección del Jugador.

En cuanto al movimiento del enemigo, para que éste pueda avanzar por el escenario, se ha utilizado el actor *Nav Mesh Bounds Volume*, con el que las IAs adquieren control de los terrenos sobre los que se puede mover junto a los diferentes obstáculos. En la figura 4.45a se puede apreciar el escenario sin el actor *Nav Mesh Bounds Volume*, mientras que en la figura 4.45b se pueden ver los caminos transitables por los enemigos tras colocar el actor y compilar la escena.

(a) Escenario base. (b) Escenario con *Nav Mesh*.

Figura 4.45: Enemigo - *Pathfinding*.

Por último, para que el movimiento del personaje sea lo más natural posible y no se limite únicamente a seguir al jugador, en el *Behaviour Tree* se definió que cuando éste se encontrase lo suficientemente cerca, realizase un movimiento conocido como *Strafing*. Éste consiste en avanzar alrededor del jugador, adquiriendo un punto en la circunferencia obtenida entre ambos y avanzar a dicho punto. Este movimiento se ve reflejado en las figuras 4.46a y 4.46b.



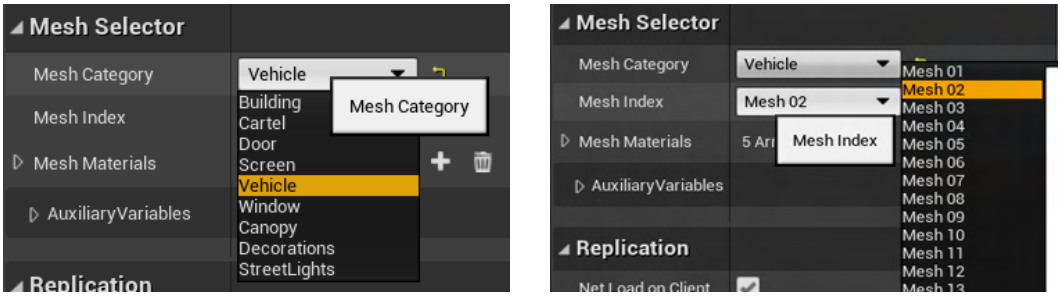
(a) Posición 1. (b) Posición 2.

Figura 4.46: Enemigo - *Strafing*.

Sistema de colocación de *props*

Para facilitarle la tarea a la artista 3D, se programó un *blueprint* que permite colocar un objeto en el escenario y elegir el *prop* deseado desde un selector.

Hay dos selectores, uno para la categoría del *prop* (figura 4.47a), donde existen un total de 9 opciones a elegir. En función de la categoría existe un selector con un número establecido de *meshes* (figura 4.47b), donde se puede seleccionar el *prop* deseado.



(a) Mesh Category. (b) Mesh Index.

Figura 4.47: Sistema de selección de *props* - Selectores.

Además, según el *prop* seleccionado, debajo se incluyó un array con el número de materiales del modelo, donde se puede sustituir cada uno de ellos por otro, tal y como se puede observar en la figura 4.48.

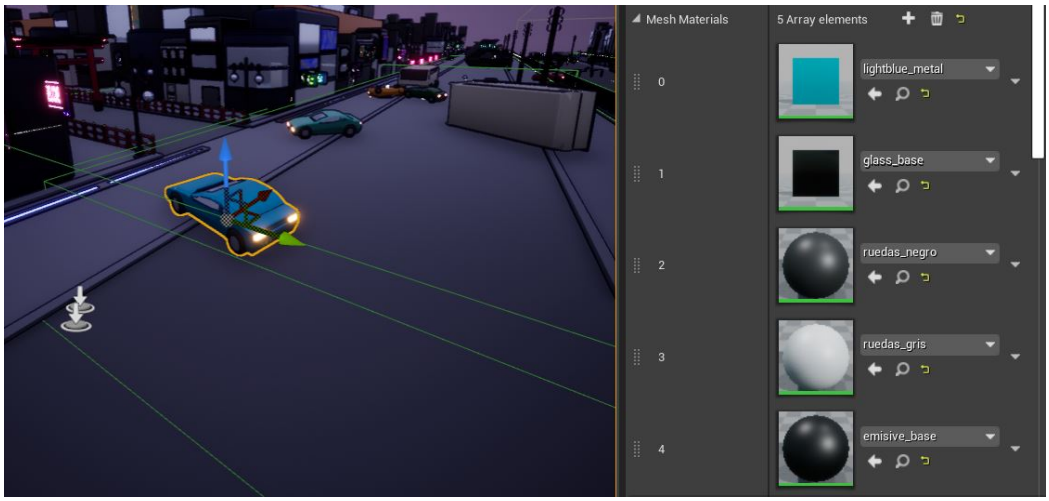


Figura 4.48: Sistema de selección de *props* - Materiales.

Entrando en profundidad en el código, se ha creado un *blueprint* principal de nombre *Mesh Selector* —el cuál es el actor que se colocará en la escena— y otro *blueprint* secundario llamado *Selectable Mesh*, que se encargará de elegir el modelo deseado por el desarrollador mediante el uso de la interfaz de usuario de Unreal Engine 4.

En segundo lugar, se han creado dos enumeradores:

- **Mesh Category**: Tiene nueve valores disponibles, uno para cada categoría de *prop* deseada. Las diferentes categorías se pueden observar en la figura 4.49.
- **Mesh Index**: Posee 60 valores disponibles (reflejados en la figura 4.50). Si el número de *props* de la categoría es menor al número máximo de índices, el valor superior asignado es devuelto al valor máximo (Ej: Si hay 15 *props* y se selecciona el *mesh* número 18, el *mesh* asignado es el 15).

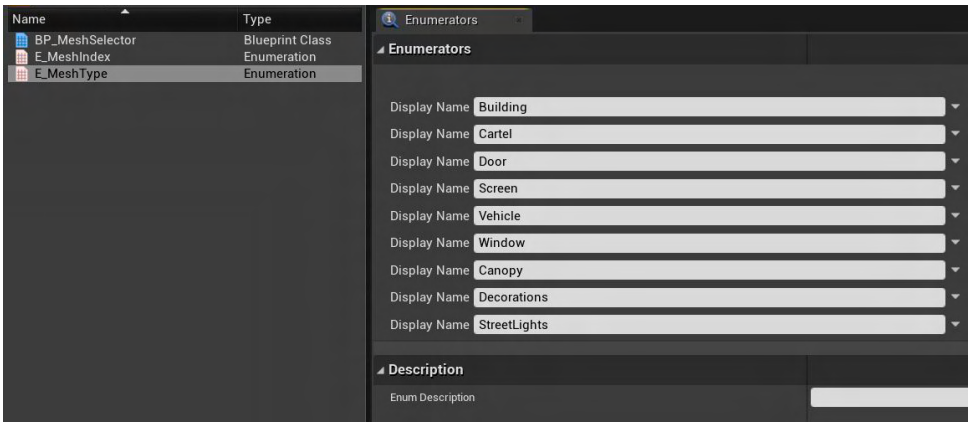


Figura 4.49: Sistema de selección de *props* - *Mesh Type*.

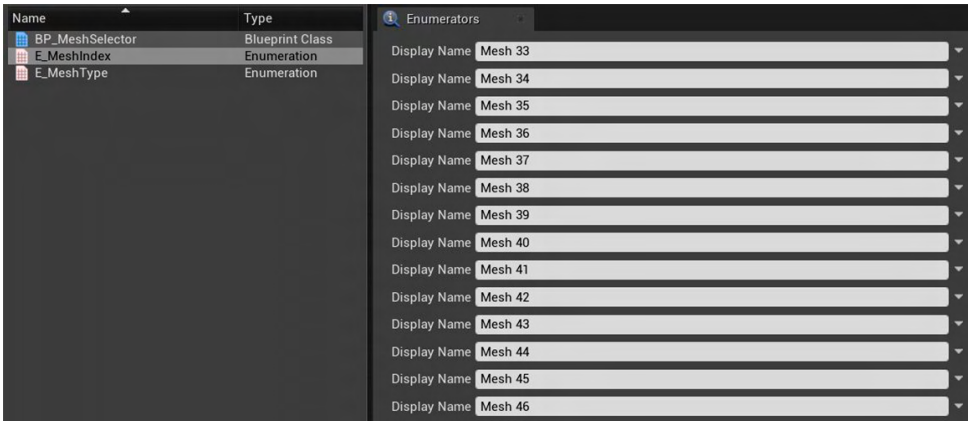


Figura 4.50: Sistema de selección de *props* - *Mesh Index*.

Posteriormente, se creó una interfaz que especifica las funciones que serán llamadas en el *blueprint Selectable Mesh*. En este caso solo ha hecho falta crear una función:

- **Check Mesh Visibility**: Recibe un entero y devuelve un *Static Mesh*. Su tarea es comprobar si existe un modelo de dicho índice en un *array* de

modelos. Si es el caso, el modelo seleccionado es el del valor correspondiente, si el valor es mayor, se devuelve el último objeto del *array*. Dentro del *blueprint Selectable Mesh* se define el código que llevará a cabo la función.

Por otro lado, el *blueprint* cuenta con un *array* llamado *meshes* —donde el desarrollador añadirán los modelos realizados y será éste el *array* que utilizará la función de la interfaz— y un entero llamado *MeshType*, que indicará el índice deseado del *array* de modelos. Para hacer uso de la función definida en la interfaz sin necesidad de ejecutar la escena, se ha hecho uso de la función *ConstructionScript*, la cuál es ejecutada cada vez que se realiza un cambio en el editor de Unreal Engine 4 y una vez al iniciar la escena. En esta función se ha realizado la llamada a *Check Mesh Visibility* (figura 4.51).

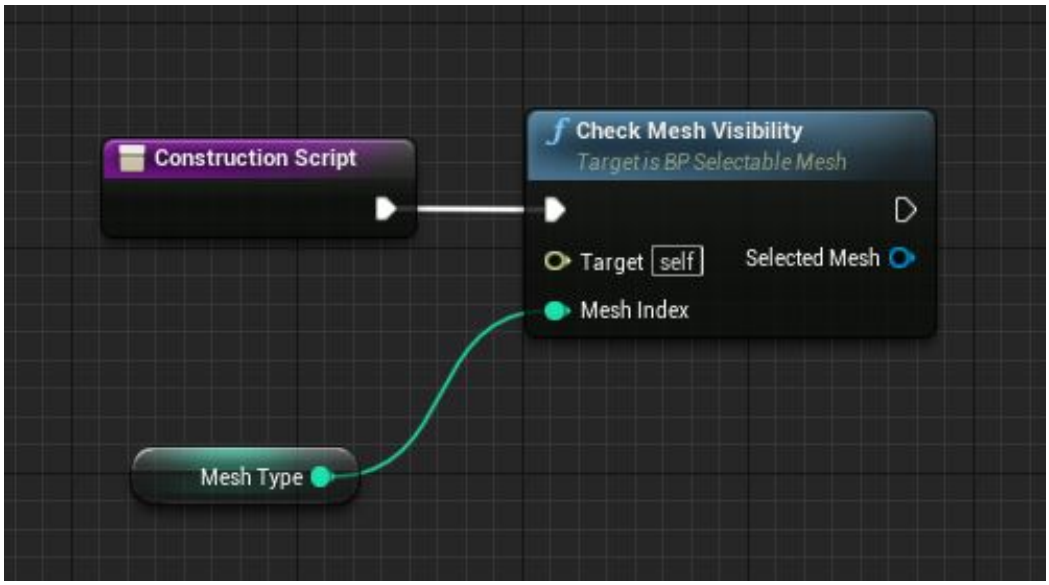


Figura 4.51: Sistema de selección de *props* - Llamada *Check Mesh Visibility*.

A partir de este punto, se han creado nueve hijos de este *blueprint*, uno por categoría, donde el *array* es rellenado por el desarrollador con una serie de modelos en función de dicha categoría. Las variables de este *blueprint* son:

- **Mesh Category**: Enumerador que determina el hijo correspondiente del *blueprint Selectable Mesh*.
- **Mesh Index**: Enumerador que determina el índice del modelo.
- **Mesh Materials**: *Array* de materiales del modelo.

En este caso se ha vuelto a hacer uso de la función *ConstructionScript* para ejecutar las tareas en el editor. En primer lugar se determina qué variante del *blueprint Selectable Mesh* será cargada mediante el uso de un *switch* (figura 4.52) y el enumerador *Mesh Category*.

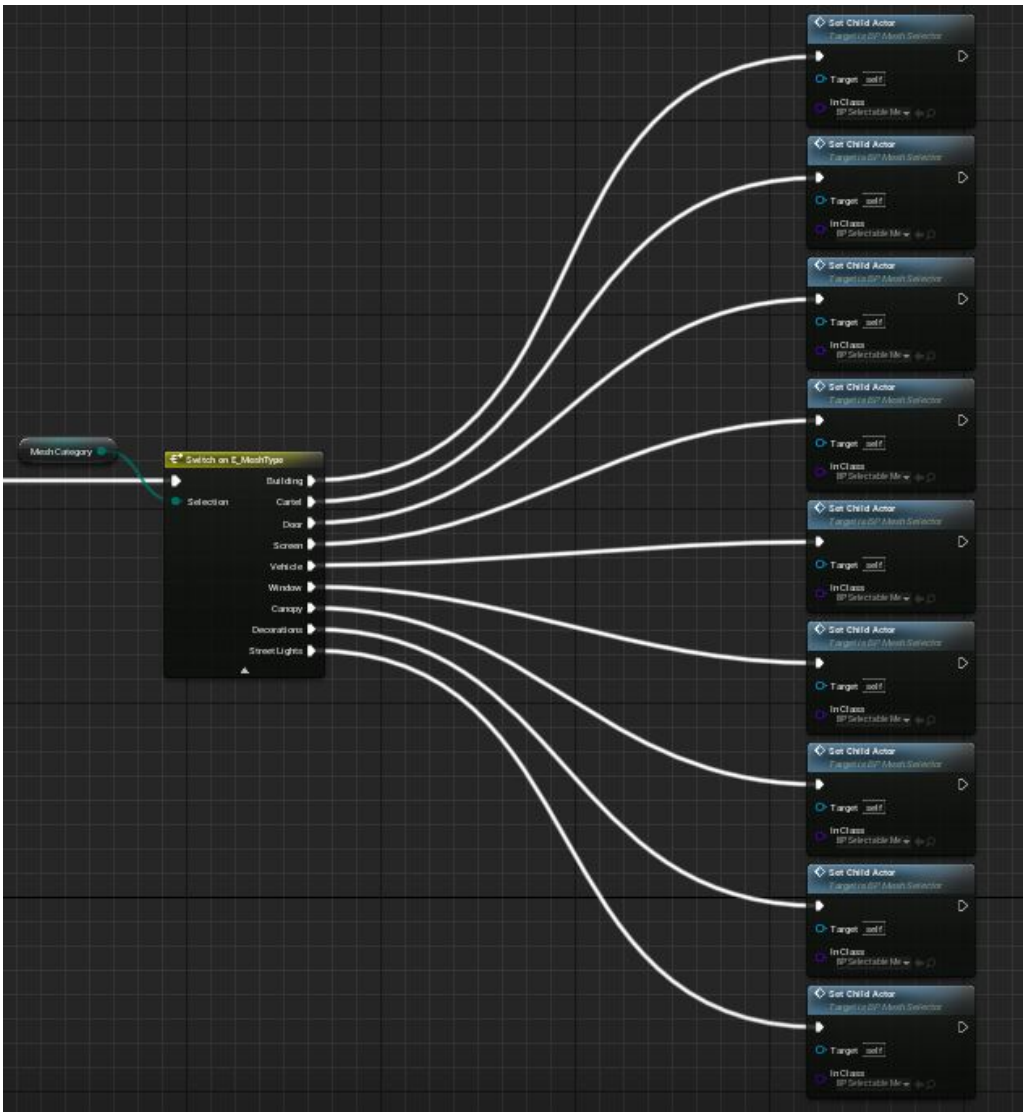


Figura 4.52: Sistema de selección de *props* - Selección de *Selectable Mesh*.

Además, el *blueprint Mesh Selector* se encarga de elegir, entre los modelos disponibles, cuál deberá ser cargado. La comprobación necesaria se puede observar en la figura 4.53.

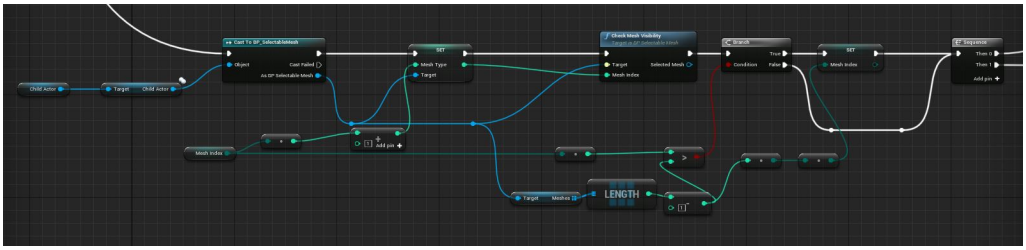


Figura 4.53: Sistema de selección de *props* - Selección del modelo con el enumerador *Mesh Index*.

Para la modificación de los materiales, primero se comprueba si el modelo actual se encuentra cargado y se actualiza el tamaño del *array* de materiales junto a su contenido. Se puede ver cómo el fragmento de código en la figura 4.54.

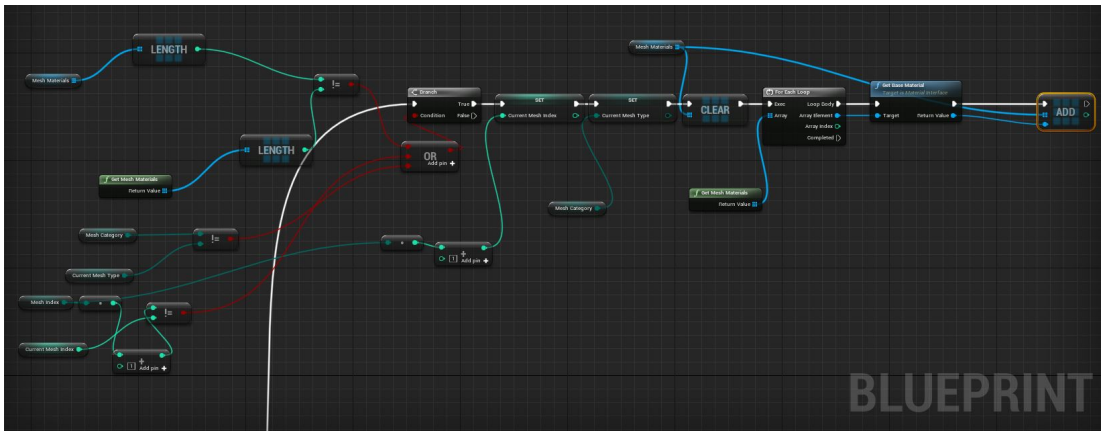


Figura 4.54: Sistema de selección de *props* - Carga de los materiales al cambiar de modelo.

Cuando el modelo ya se encuentra cargado y el *array* actualizado, se comprueba el *array* de materiales y, si hay algún cambio, los materiales del modelo son cambiados por los del *array*, tal y como se puede ver en la figura 4.55.

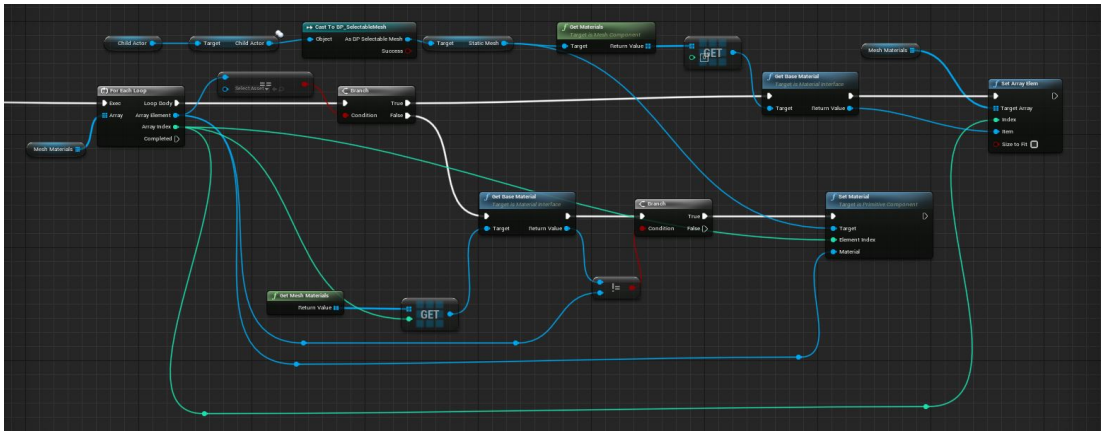


Figura 4.55: Sistema de selección de *props* - Actualización del modelo con nuevos materiales.

En resumen, en la figura 4.56 se puede observar un diagrama de clases donde se pueden ver las distintas clases, enumeradores, interfaces y herencias de las que se ha requerido para implementar este sistema de colocación de *props*.

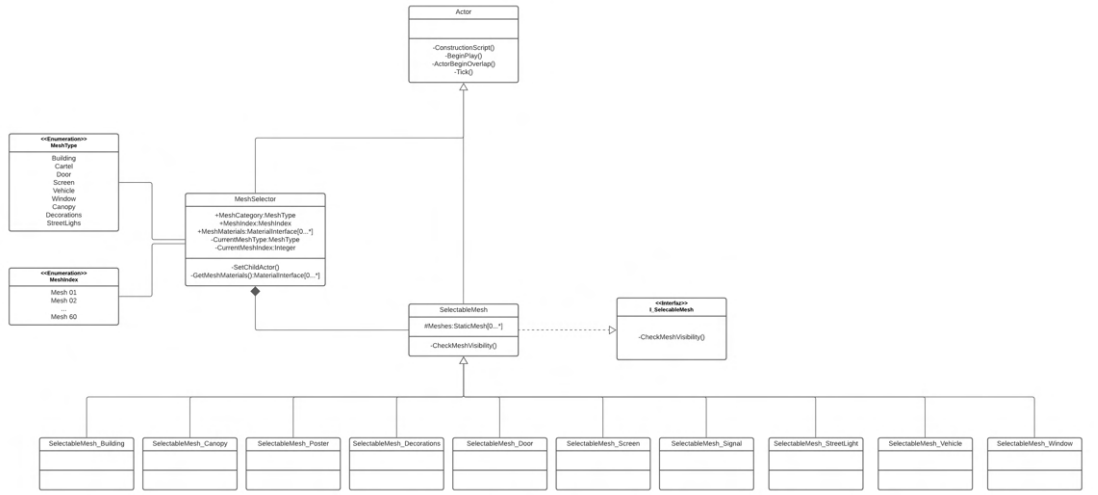


Figura 4.56: Sistema de selección de *props* - Diagrama de clases.

Interfaz y menús

Para la interfaz y los menús del videojuego se ha hecho uso del *Unreal Motion Graphics UI Designer* o UMG [30], el cual se ha utilizado para implementar la interfaz de usuario en el videojuego.

Para la barra de salud, se ha hecho uso del *asset* HQUI: Progress Bars [31], mientras que para los botones, se ha hecho uso del *asset* HQUI: Buttons [32].

En cuanto a la interfaz, se han colocado los elementos de tal forma que se pueden adaptar a cualquier tipo de monitor o pantalla. Para ello, al seleccionar un elemento del UMG, se establece un punto de ancla (figura 4.57) con el cual, aunque cambie la proporción de la pantalla, el elemento se escalará con respecto a dicho punto. En la figura 4.58a se muestra una proporción de pantalla de 16:9, mientras que en la figura 4.58b se encuentra dicha proporción modificada, pero con los elementos de la interfaz correctamente situados.

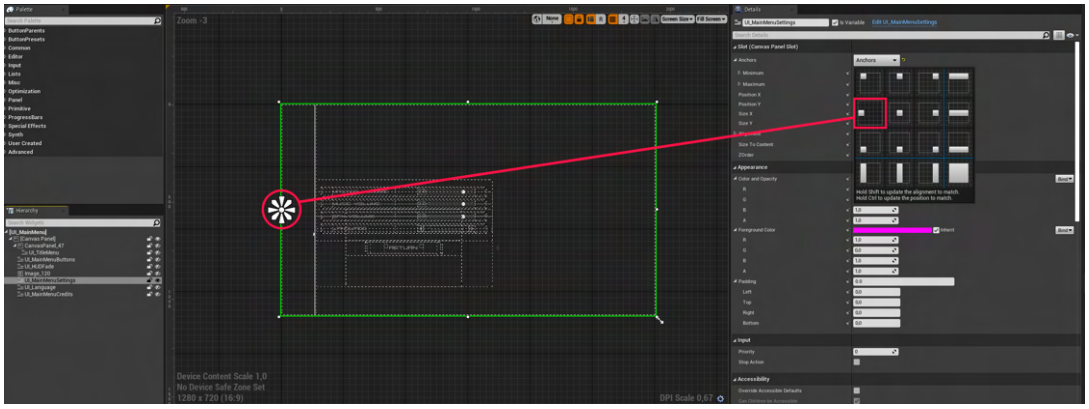


Figura 4.57: Interfaz - Punto de ancla.

(a) Proporción original.

(b) Proporción modificada.

Figura 4.58: Interfaz - Adaptación de los elementos de la interfaz a la proporción de la pantalla.

Por otro lado, todos los elementos del menú principal cumplen con su función establecida en el diseño del videojuego.

El menú de inicio consta de cuatro pantallas:

- **Pantalla de inicio**: Muestra el título del videojuego y un pequeño mensaje que indica que hay que presionar cualquier tecla para avanzar.
- **Menú principal**: El menú principal es mostrado tras una animación de barrido y cuenta con todos diseñados.
- **Menú de opciones**: La cantidad de opciones se ha visto limitada al volumen maestro, de los efectos, de la música y al intercambiador de idioma.
- **Pantalla de créditos**: Muestra el nombre de cada uno de los integrantes de este proyecto junto al rol ejercido.

Además, en el menú principal se ha añadido un botón respecto a lo documentado en el diseño del videojuego llamado *Encuesta* (figura 4.59), el cual envía al jugador a una encuesta que nos permitirá adquirir datos y retroalimentación. Se entrará más en profundidad sobre esto en el capítulo 5. Análisis de Resultados.

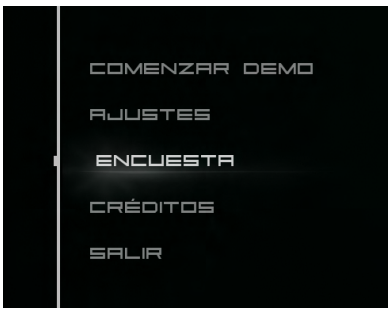


Figura 4.59: Menú - Botón de encuesta.

Selector de apariencia

Según Petra Mether en su tesis titulada *Character Customization in Video Games* [33], la posibilidad de editar la apariencia del personaje manejado aumenta la sensación de inmersión y ayuda a que el jugador empatice en mayor medida con su avatar. Por ello, para esta demo se ha incluido un sistema de personalización para el personaje principal, donde se puede elegir el nombre, el color de la piel y el color del brazalete que lleva puesto en el brazo izquierdo.

Para este selector, se ha creado una escena individual con una luz direccional, una cámara y un actor en el centro. Para este actor, se ha hecho un *Animation Blueprint* con dos animaciones (figura 4.60), una de respirar y otra con el brazo levantado enseñando el brazalete.

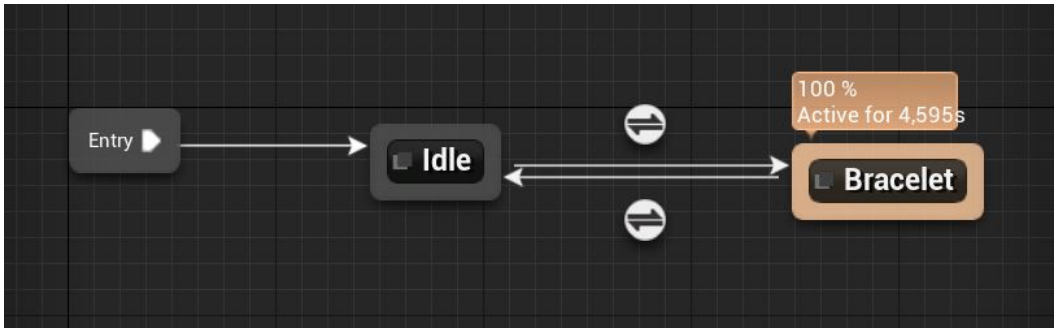
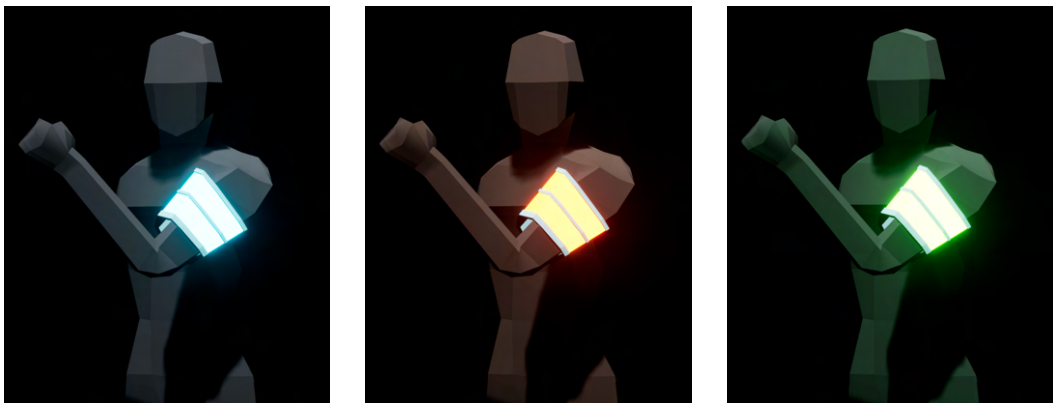


Figura 4.60: Selector de apariencia - Grafo de animaciones.

También se ha diseñado un menú con dos selectores, donde en función del valor el material de la piel y del brazalete es cambiado por uno distinto. En total hay cuatro tonos de piel y tres materiales con componente emisiva para el brazalete (figuras 4.61a, 4.61b y 4.61c).

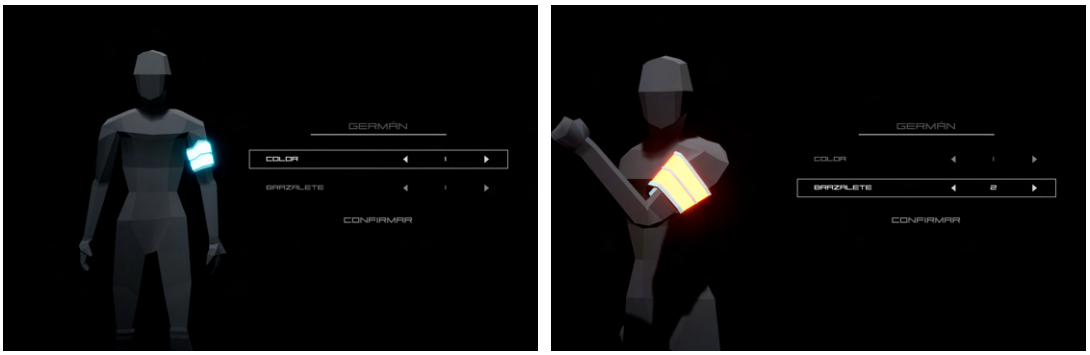


(a) Azul. (b) Rojo. (c) Verde.

Figura 4.61: Selector de apariencia - Materiales.

Por último, se ha programado que en función del valor que esté siendo cam-

biado, la cámara se acerque si se está modificando el brazalete, o se aleje si se está modificando el tono de piel, además de alternar entre las dos animaciones realizadas. Ambos estados se encuentran reflejados en las figuras 4.62a y 4.62b.



(a) Color de piel. (b) Color del brazalete.

Figura 4.62: Selector de apariencia - Cámara y animación.

Los colores seleccionados son guardados en un *blueprint* de tipo *Save Game* y cargados mediante un *blueprint* de tipo *Game Instance* al cambiar entre escenas.

Sistema de diálogos

Para realizar el sistema de diálogos, primero se definieron las tareas a realizar por los actores de la escena:

- El PNJ con la capacidad de iniciar un diálogo debe poder ser detectable por el jugador.
- Durante el diálogo se debe poder cambiar de posición la cámara y actualizar la animación de ambos personajes por cada línea de texto nueva.
- El final del diálogo debe llamar a un evento en caso de que la escena lo requiera (ej. El personaje de regala un objeto al jugador y éste se añade a su inventario).
- El texto debe poder aparecer tanto en inglés como en español en función del idioma seleccionado por el jugador.

Para que el personaje controlado por el jugador pudiese detectar a los PNJs con el fin de poder interactuar con ellos, se ha realizado un *raycast* esférico alrededor del PNJ el cual si detecta que el jugador está colisionando con él, éste le envía al jugador una señal de que puede iniciar una conversación con él. Todo esto es gestionado por la funciones del *blueprint* padre, PNJ, de forma que el desarrollador decida con cuáles PNJs se puede interactuar y con qué condiciones.

Para el sistema de diálogos en sí, se ha vuelto a hacer uso de los *Behaviour Trees*, de forma que se pueda generar una secuencia de frases mediante los nodos *Sequence* y una opción de diálogo mediante los nodos *Selector*. Se desarrollaron tres *Tasks* para estos *Behaviour Trees*:

- **Play Sentence**: Muestra en pantalla una frase y actualiza las animaciones de los personajes y la posición o animación de la cámara a elección del desarrollador.
- **Reply**: Actualiza la posición o animación de la cámara, muestra en pantalla las diferentes elecciones de diálogo y una vez el jugador selecciona una de ellas, actualiza las animaciones de los personajes.
- **End Dialogue**: Concluye el diálogo y ejecuta un evento a elección del desarrollador.

Por último, al iniciar y terminar un diálogos se cambia del estado de juego al de diálogos y viceversa mediante un fundido a negro.

Cinemáticas

Las cinemáticas se han realizado haciendo uso del editor de secuencias (*Sequence Editor*) de Unreal Engine 4. Este sistema permite, mediante la utilización de una de tiempo donde se sitúan los fotogramas clave, modificar la posición de elementos del escenario, activar eventos, interpolar animaciones, etc.

Las cinemáticas comienzan a ejecutarse cuando el jugador colisiona con un *trigger* (figura 4.63), actor donde se programó una función que se encarga de comenzar la cinemática al llamarse al evento de colisión.

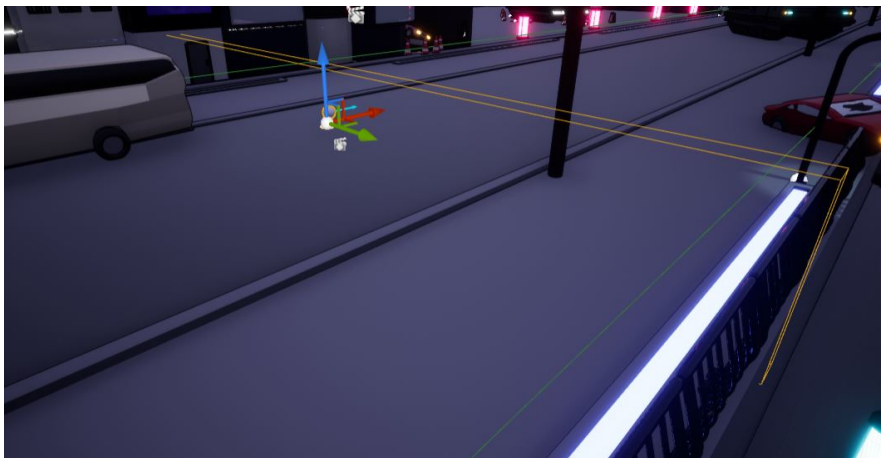


Figura 4.63: Cinemáticas - Trigger.

Se han realizado un total de tres cinemáticas:

- **Presentación**: El personaje y el escenario son presentados. Adam se cuestiona el dónde se encuentra y el qué ha ocurrido, por lo que decide que lo mejor es explorar el escenario para descubrirlo.

- **Descubrimiento de la situación**: Adam observa un cadáver situado frente a un coche y se preocupa por la amenaza que puede encontrarse en el lugar.

- **Desenlace**: Adam entra en conflicto consigo mismo debido a las decisiones tomadas por el jugador hasta que colapsa, dando lugar al final de la *demo*, terminando el juego con un fundido a negro y devolviendo al jugador al menú principal.

Para la impresión de las frases en pantalla se volvió a hacer uso de *Behaviour Trees*, utilizando las mismas *Tasks* que en el sistema de diálogos, facilitando y agilizando la implementación de las cinemáticas.

Introducción

En la introducción del juego, se imprimen varias frases por pantalla, siendo cada una de éstas mostradas a medida que el jugador pulsa un botón del mando o una tecla del teclado.

Para mostrar estas frase se ha vuelto a hacer uso del *UMG UI Designer* y los *Behaviour Trees*. En primer lugar se creó una UI con un *Text Box* y una función que actualiza la frase del texto por el valor de una variable *text*, que contiene la oración tanto en español como en inglés.

Además, se ha añadido en la introducción una frase que incluye el nombre del jugador (figura 4.64). Para ello se ha hecho de la función *Get Platform User Name* y la programación de la etiqueta "<username>", de forma que si al imprimir el texto detecta dicha etiqueta, se sustituye por el nombre del sistema del jugador. Este mismo sistema fue utilizado para añadir opciones de diálogo con el nombre del jugador. Para cuando el videojuego fuese lanzado por Steam, esa función sería sustituída por *Get Player Steam ID*, que detecta el nombre de usuario del *launcher*.

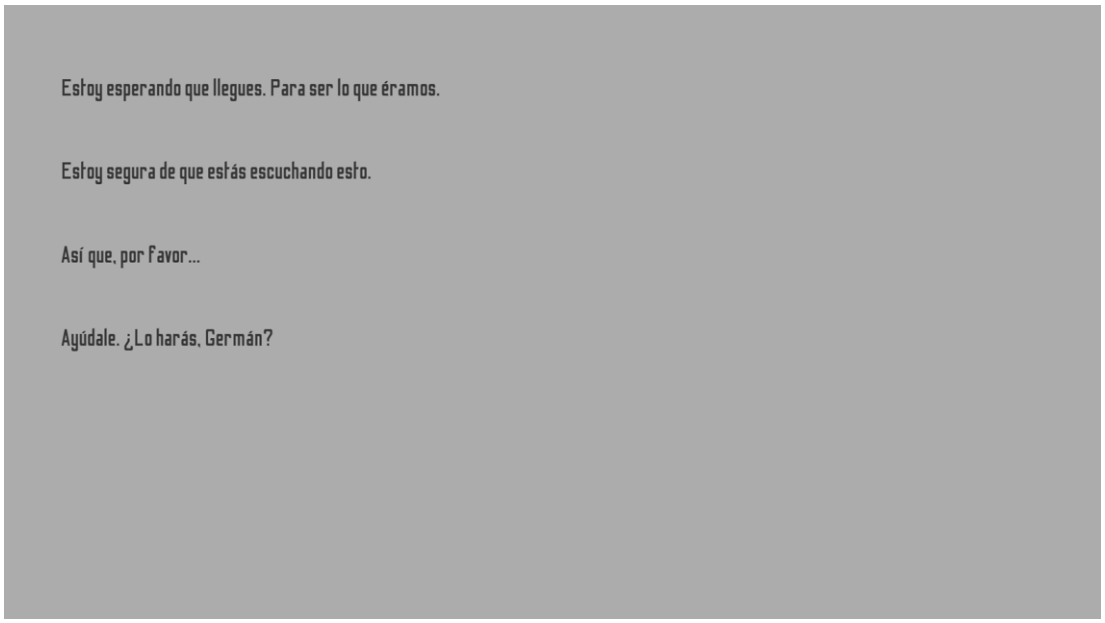


Figura 4.64: Introducción - Nombre de usuario.

4.2.6. VFX

En el proyecto se han implementado diferentes sistemas de partículas para obtener un resultado más orgánico y vivo. Para esta tarea, se ha hecho uso de los dos sistemas de partículas que posee Unreal Engine 4, *Cascade* [34] y el más reciente, *Niagara* [35]. Además, se ha utilizado un *asset* del bazar de Unreal Engine 4, llamado *Muzzle Flashes* [36], modificado y usado en el cañón de las armas cuando éstas son disparadas.

Los enemigos, al morir, son desintegrados desde la ubicación en la que han sido golpeados por la última bala que les impactó. Para simular esto, se han realizado dos efectos. En primer lugar, se creó un material a partir de unos pasos propuestos por UnrealCG [37] donde, a partir de un vector que represente una ubicación en el espacio, se hace transparente una zona del material en función de un valor. Además, se programó que alrededor de las áreas transparentes, apareciese un color emisivo para simular de una manera más colorida la pulverización del enemigo.

En la figura 4.65 se puede observar el *blueprint* del material desarrollado. Por otro lado, en las figuras 4.66a, 4.66b y 4.66c se muestra el proceso de desintegración del material. Finalmente, en la figura 4.67 se pueden ver las variables que pueden ser ajustadas para obtener el resultado deseado.

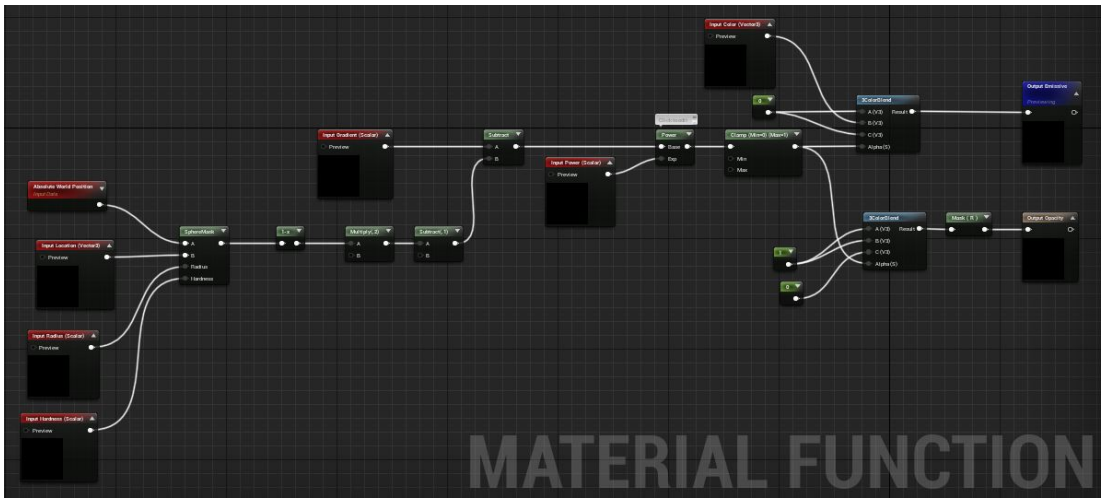
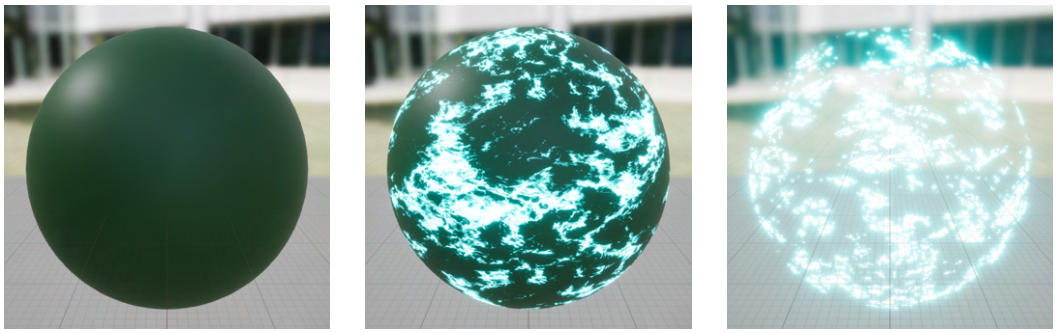


Figura 4.65: VFX - Material que simula la desintegración.



(a) Estado 1. (b) Estado 2. (c) Estado 3.

Figura 4.66: VFX - Proceso de desintegración.

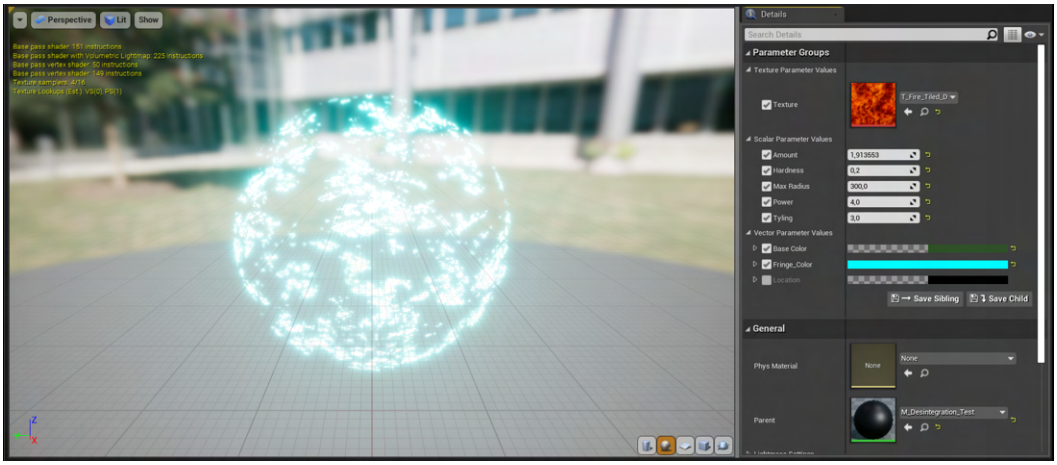


Figura 4.67: VFX - Variables del material.

Por otro lado, se realizó un sistema de partículas con *Niagara* que se adhiere el esqueleto de un modelo (figura 4.68). Este sistema de partículas, al ser comple-

mentado con el material anterior, da lugar al efecto de desintegración deseado. El resultado final se puede observar en la figura 4.69.

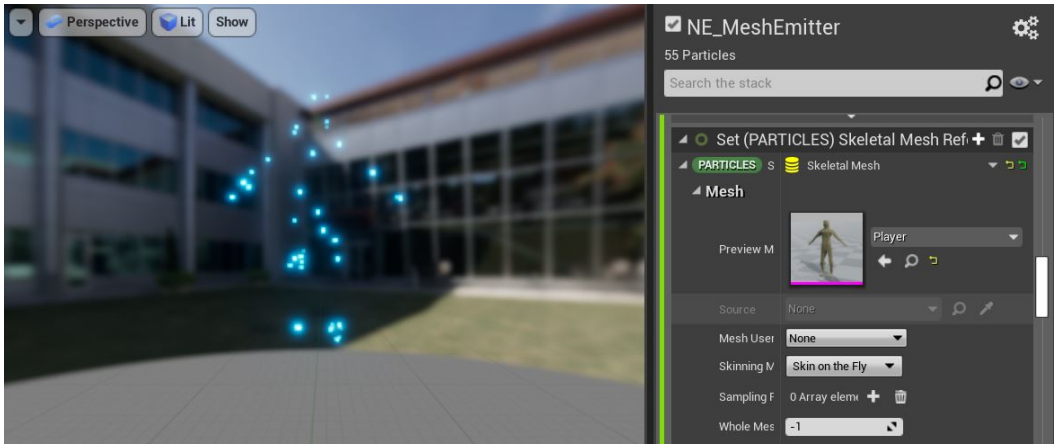


Figura 4.68: VFX - Partículas y selección de esqueleto.

Figura 4.69: VFX - Resultado final.

5

Análisis de Resultados

Desde el día 21 de julio hasta el 7 de septiembre, se hizo público el proyecto y el prototipo estuvo disponible durante ese tiempo. Los que jugaron el proyecto fueron encuestados con el fin de obtener su retroalimentación y obtener una serie de datos acerca de las ideas propuestas en este trabajo. En este capítulo se especifican los resultados obtenidos.

5.1. Análisis sobre el trabajo realizado

En primer lugar, debemos hacer un breve análisis del trabajo realizado. Se obtuvo un proyecto de una duración estimada de 30 minutos de juego. Al ser el foco principal la narrativa y la sensación de inmersión en el mundo, se optó por una dificultad asequible para cualquier tipo de usuario, con el fin de no extender más la experiencia y poder así obtener más resultados en la encuesta.

5.2. Perfiles de los Encuestados

La encuesta fue realizada por un total de 29 personas, de los cuáles tenemos los siguientes datos:

- El 72,4 % no ejerce en ningún puesto de trabajo, mientras que, del 27.6 % restante, el 50 % ejerce en puestos relacionados con campos de la ingeniería.

- La familiarización con las nuevas tecnologías es generalmente alta, considerando el 55,5 % que poseen el conocimiento más elevado en este campo, tal y como se puede observar en la figura 5.1.

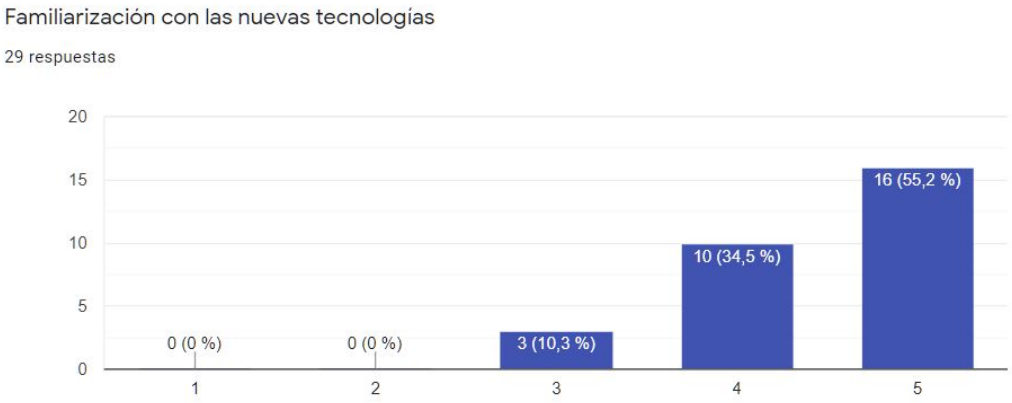


Figura 5.1: Encuesta - Familiarización con las nuevas tecnologías.

- El 69 % de los encuestados pertenecen a la franja de edad de entre los 18 y 24 años según la gráfica de la figura 5.2.

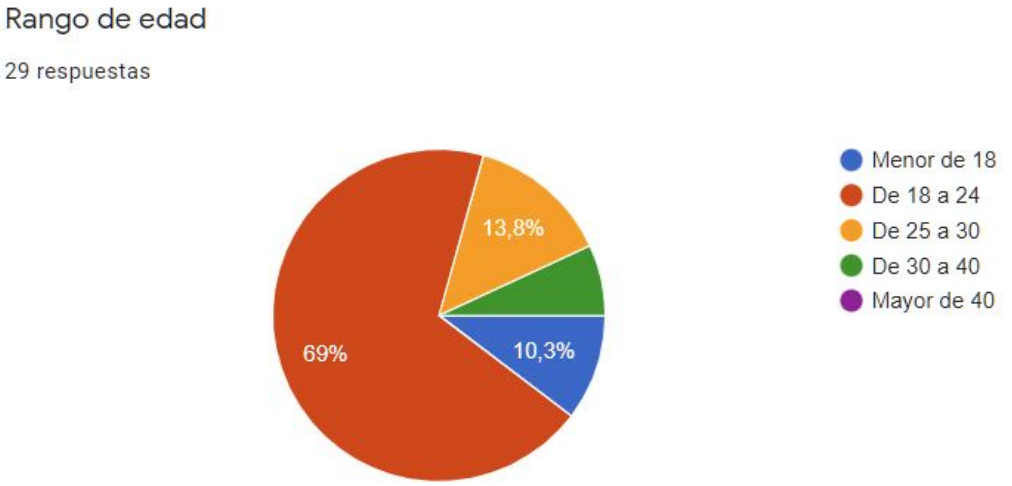


Figura 5.2: Encuesta - Franja de edad.

- El 86,2 % ha nacido y crecido en España, siendo el porcentaje restante proveniente de Latinoamérica.

- El 55,2 % juega más de 3 horas diarias, mientras que, además, el 48,3 % juega más de 10 semanales, seguido de un 24,1 % que juega de 7 a 10 horas semanales.

- Los géneros de videojuego predominantes son los videojuegos de aventura (17,2 %), los videojuegos de rol (17,2 %) y los videojuegos en línea (10,3 %).
- El 72,4 % juega en PC, seguido de Play Station y Nintendo con un 10,3 %, y Xbox junto a dispositivos móviles con un 3,4 %, donde los jugadores de dispositivos móviles en esta encuesta juegan su totalidad en Android.
- El *launcher* más utilizado es Steam, seguido de Epic Games y Battle.net, según los datos reflejados en la figura 5.3.

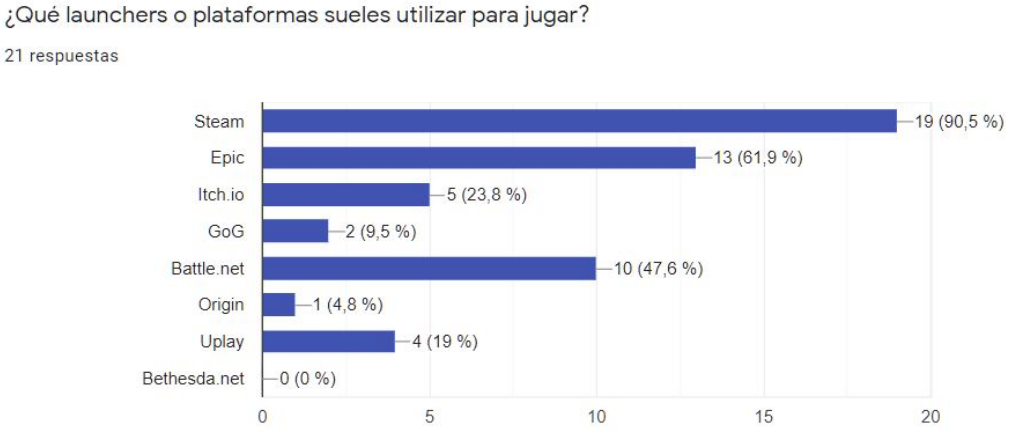


Figura 5.3: Encuesta - Launchers.

Por ende, teniendo en cuenta que la propuesta sobre el lanzamiento era publicar el videojuego completo en Steam, al ser PC la plataforma en la que más juegan los encuestados, puede ser un dato muy relevante a tener en cuenta para el futuro desarrollo del videojuego.

5.3. Análisis sobre la Narrativa y el Diseño

En esta sección, se profundizará en los datos obtenidos de la primera parte de la encuesta, la cuál abarca toda la narrativa y diseño del videojuego.

Según los datos mostrados en la figura 5.4, 69 % de los encuestados respondió que la palabra *inmersión* en un videojuego hace alusión a aquella característica que consiste en cómo de introducido se siente el jugador en el mundo de ficción respecto a la jugabilidad. Es decir, si, por ejemplo, en un diálogo podemos responder lo que deseemos, o si el ambiente del mundo es uno post-apocalíptico donde cada recurso cuenta, se puede considerar que ese videojuego cuenta con componentes *inmersivas*. Por otro lado, el 24,1 % interpreta la palabra como la característica que hace creer al jugador que forma parte de la trama y que sus decisiones importan, haciéndole sentir parte del mundo de ficción.

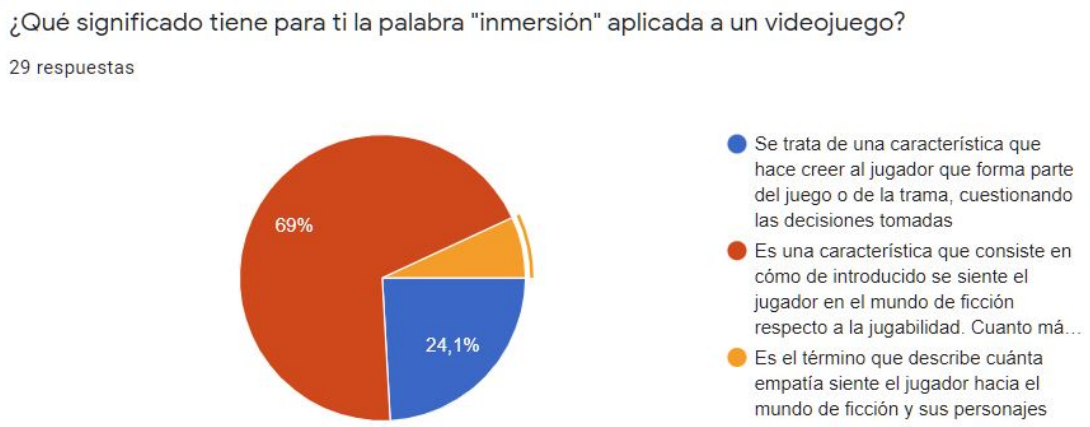


Figura 5.4: Encuesta - Significado de *inmersión*.

El 79,3 % ha jugado alguno de los tres títulos propuestos, donde cada uno cuenta con el siguiente porcentaje:

- The Stanley Parable: 31 %
- Metal Gear Solid 2: 10,3 %
- Doki Doki Literature Club!: 37,9 %

De entre las personas que han jugado estos títulos, el 58,6 % se sintió implicada en el mundo de ficción al jugar la obra seleccionada y el 72,4 % consideró que cumplía con las características de un videojuego *inmersivo*.

El sentimiento predominante cuando, en la introducción, aparece en pantalla el nombre de usuario del sistema operativo del jugador, es el de impactado con un 48,3 %.

Además, como se puede observar en la figura 5.5, el 65,5 % considera que el hecho de poder configurar la apariencia del personaje le hace conectar más con el mismo, confirmando el planteamiento de Petra Mether en su tesis.

Por otro lado, según lo que se puede ver en la figura 5.6, hay prácticamente un empate acerca de lo que ocurre en la introducción, siendo que el 48,3 % cree que hay una persona que se preocupa por su amante y el 51,7 % cree que dicha voz se dirige al jugador y le pide a éste que cuide del personaje, dato que tiene sentido teniendo en cuenta que no se busca un entendimiento claro de la situación, sino todo lo contrario para incrementar la duda en el jugador para cuando fuese publicado el videojuego completo.

El 62,1 % cree que ha interpretrado a un personaje con su propia historia, lo cual es correcto.

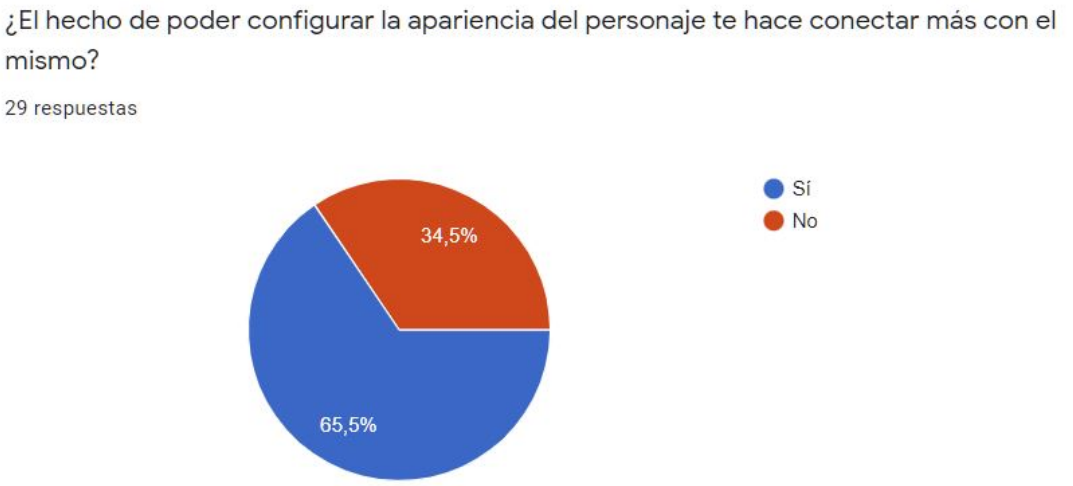


Figura 5.5: Encuesta - Conexión con el personaje principal.

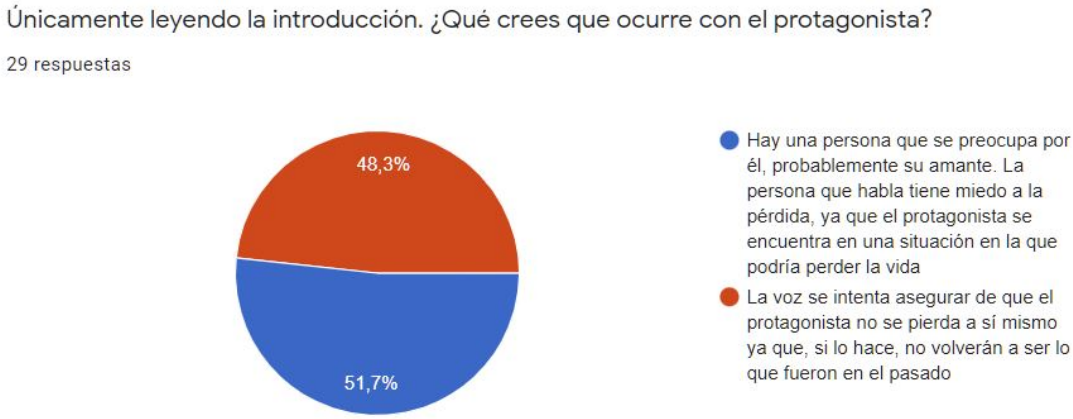


Figura 5.6: Encuesta - Significado de la introducción.

Sin embargo, a la hora de comparar al personaje con el protagonista de otro videojuego, las respuestas han estado parejas, sobresaliendo Raiden, de Metal Gear Solid 2 con un 13,8 % según las estadísticas de la gráfica en la figura 5.7.

Por otro lado, el 96,6 % de los encuestados opina que las mecánicas conversacionales aportan a la sensación de inmersión en el videojuego, siendo que el 75 % cree esto por le hacen sentir como si estuviese interpretando su propia versión del protagonista, mientras que el 25 % restante siente que es él quién interactúa con los PNJs. Además, el 62,1 % cree que el personaje sí tiene personalidad propia mientras que el 37,9 % opina lo opuesto.

Según el 44,4 % de los encuestados, la frase que mejor define al personaje es la de una persona con una personalidad distorsionada y manipulada por alguien más, además el adjetivo que mejor define al personaje es confundido según el 72,2 %. La confusión del personaje sobre su nombre y su pasado es lo que predomina.

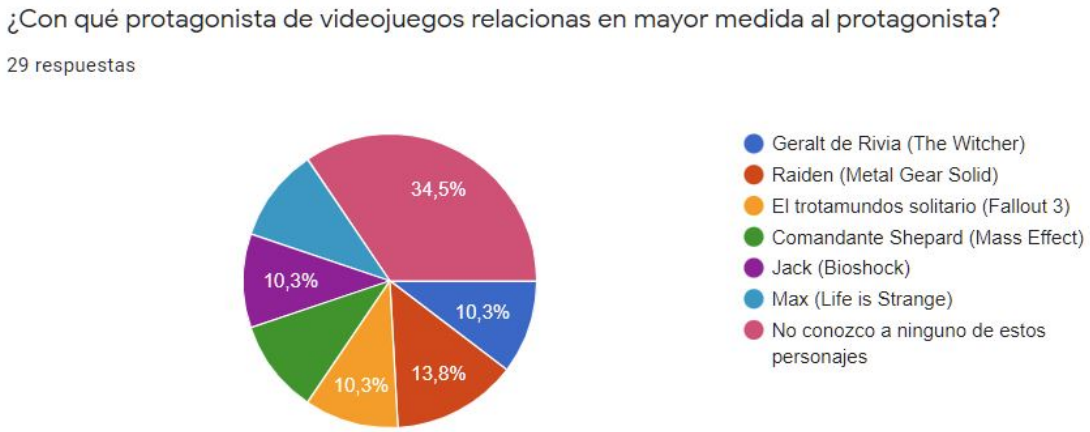


Figura 5.7: Encuesta - Comparativa con otros protagonistas.

El 62,1 % le dio el anillo de la mujer a John y, dentro de dicho porcentaje, cuando John le pregunta a Adam si alguna vez amó a alguien, el 61,1 % contestó lo que pensaron de ellos mismos en lugar de lo que pensaría Adam, predominando el pensamiento del jugador frente al del personaje, por lo que se puede observar en la figura 5.8.

Figura 5.8: Encuesta - Respuesta sobre uno mismo.

Además, el 51,7 % contestó su nombre cuando el militar les preguntó por el mismo, siendo que el 20,7 % nunca llegó a hablar con el militar, nuevamente predominando el pensamiento del jugador frente al del personaje.

En la penúltima pregunta, el 89,7 % de los encuestados afirman que pese a lo anterior, el personaje no es un reflejo del jugador, si no que tiene su propia esencia y personalidad.

Por último, de entre las 5 opciones sugeridas, el 51,7 % asocia el diseño de niveles al de un juego de género *Soulslike*, siendo este el género en el que se ha

basado todo el diseño de niveles. Esto se ve reflejado en la gráfica representada en la figura 5.9.

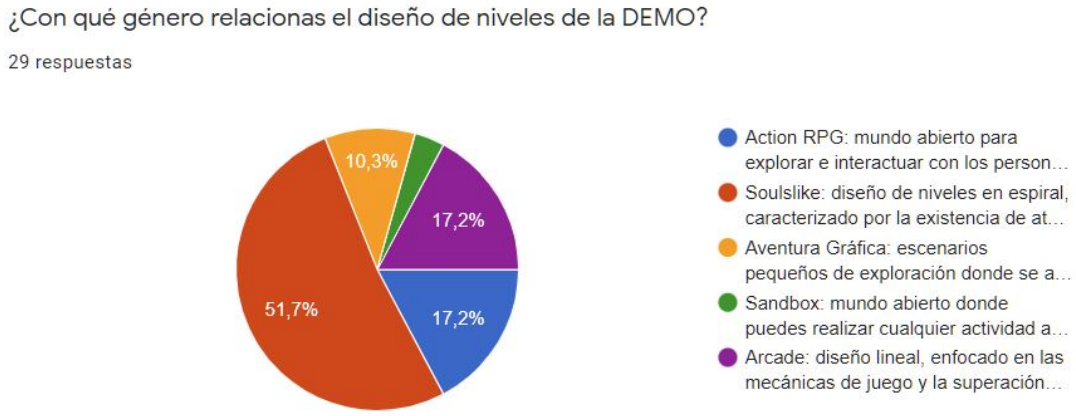


Figura 5.9: Encuesta - Género del videojuego.

6

Conclusiones y Líneas Futuras

Tras terminar el análisis de objetivos, plantear la planificación del proyecto, realizar la investigación pertinente, diseñar un videojuego, desarrollar el prototipo y obtener los datos necesarios, queda por último elaborar las conclusiones. En este capítulo se reflejará la reflexión final sobre el trabajo realizado.

6.1. Conclusiones sobre el Desarrollo del Prototipo

Para este trabajo se ha logrado diseñar un videojuego donde se implique al jugador en el mundo de ficción y se le relacione con su personaje de manera directa, haciendo uso de la ruptura de la cuarta pared y el desarrollo de una personalidad en un protagonista con características de avatar sin personalidad.

Además se ha logrado desarrollar un prototipo que presenta las características clave del proyecto:

- Se ha introducido un protagonista con apariencia genérica respecto al resto de personajes, lo que permite que el jugador pueda identificarse con él, al no distinguir entre género o raza.
- Se ha presentado un conflicto de identidad reflejado en los diálogos
- Se ha obtenido una implicación real por parte del jugador, a través de la personalización del nombre de la sesión, la apariencia del personaje y

la posibilidad de elegir entre una serie de posibles respuestas durante las conversaciones.

- Se ha logrado la ruptura de la cuarta pared mediante la obtención del nombre de usuario del sistema operativo en el que se está ejecutando el videojuego, las preguntas de los personajes dirigidas de manera indirecta al jugador —como el caso de John preguntándole al jugador si alguna vez ha amado a alguien— y los efectos visuales que reflejan el estado mental en el que se encuentra el protagonista.

A continuación, en la tabla 6.1 muestra una comparativa entre la cantidad de elementos diseñados y los elementos desarrollados en el prototipo.

Elemento	Cantidad diseñada	Cantidad realizada
Mecánicas	4	4
Armas	6	1
Menús	2	2

Tabla 6.1: Cuantificación final del trabajo realizado 1.

En cuanto a las mecánicas, se lograron desarrollar todas las mecánicas principales descritas en el documento de diseño.

En lo referente a las armas, en el documento de diseño se concretó que durante la aventura, se obtendrían armas nuevas hasta contar con un total de seis. Sin embargo, al desarrollar un prototipo de una duración tan limitada, se optó por incluir únicamente una de las armas.

En cuanto a los menús, se desarrollaron tanto la pantalla de título, como el menú principal con todos los botones operativos.

En la tabla 6.2 se muestra la comparativa de las animaciones y diálogos estimados con respecto la cantidad final.

Elemento	Cantidad estimada	Cantidad realizada
Animaciones	25	25
Diálogos	5	7

Tabla 6.2: Cuantificación final del trabajo realizado 2.

Respecto a las animaciones, se llegaron a realizar todas las estimadas para el prototipo.

En cuanto a los diálogos, se realizó una estimación menor a la cantidad real debido a que, en un inicio, las cinemáticas de la introducción, no iban a contener diálogos, mientras que la cinemática intermedia fue realizada durante el desarrollo, ya que se consideró que así sería mejor introducida la situación y aumentaría el interés del jugador.

A continuación, en la tabla 6.3 muestra la comparativa de las horas de trabajo estimadas y las horas de trabajo finales.

Elemento	Estimación	Realidad
Horas empleadas	±300	±375

Tabla 6.3: Cuantificación final del tiempo dedicado.

Se realizó una estimación de las horas a emplear teniendo en cuenta que el desarrollo del videojuego comenzase el 1 de febrero y terminase a finales de junio. Multiplicando la cantidad de 15 horas semanales (días laborales) por las semanas del mes y el total de meses del desarrollo, da como resultado unas 300 horas, aproximadamente. Pese a todo, la fecha final del desarrollo del proyecto fue a finales de julio, por lo que se debe sumar el tiempo extra correspondiente.

Teniendo en cuenta todo lo anterior, con este proyecto no solo se han adquirido y potenciado conocimientos técnicos en los diferentes programas mencionados durante el trabajo, si no que además se han adquirido aptitudes de organización e improvisación —entendiendo improvisación como la capacidad de adherirte a la situación y hacer los cambios pertinentes si éstos son necesarios— lo que permitirá tener un desempeño óptimo para proyectos futuros.

6.2. Cumplimiento de los Objetivos

Dada la cantidad de objetivos planteada en el Capítulo 2. Objetivos y Planificación, se ha decidido cuantificar las resoluciones de cada uno.

Objetivo 1: ***Estudiar la relación entre personaje y jugador a través de la narrativa y la implementación de la ruptura de la cuarta pared como recurso principal.***

Como se observó en la Sección 1.2. Estado del Arte y en los resultados de la encuesta, la relación entre el personaje y el jugador generalmente recae en la cooperación o en la imposición de poder del uno sobre el otro. En Doki Doki Literature Club! podemos observar cómo Monika intenta abrumar al jugador e incluso burlarse de él rompiendo los límites entre el mundo de ficción y el plano

de la realidad. Lo mismo ocurre con Psycho Mantis en Metal Gear Solid. Sin embargo, la relación entre ambas entidades también puede basarse en una de cooperación, como es el caso de Niko en Oneshot, siendo personaje y jugador un complemento el uno del otro.

Objetivo 2: ***Diseñar unas mecánicas de juego acordes a la necesidad de implementar una experiencia donde personaje y jugador estén directamente relacionados.***

En el diseño y el desarrollo del videojuego, el sistema de diálogos fue uno de los ejes centrales del juego, siendo la herramienta de interacción principal entre el personaje y el jugador. Podemos observar cómo el personaje se cuestiona las decisiones del jugador y, además, esto genera el conflicto de la cinemática final de la *demo*, donde el protagonista sabe que algo no va bien con sus respuestas.

Finalmente, podemos observar que esta mecánica sí es relevante a la hora hacer una experiencia que rompa las barreras entre el mundo de ficción y el mundo real, ya que según el 99,6 % de los encuestados opinan que las mecánicas conversacionales aportan a la sensación de inmersión de un videojuego, por lo que aprovechar eso es un factor muy relevante.

Objetivo 3: ***Estudiar los factores que permiten la inmersión del jugador en la experiencia de juego.***

Del mismo modo que en el objetivo anterior, se ha comprobado que las mecánicas conversacionales tienen un gran impacto en la sensación de inmersión, además del uso de recursos que rompan límites entre el videojuego y la vida real, como es el caso de la modificación o lectura de archivos externos al juego, siendo un ejemplo de esto la lectura del nombre de usuario del sistema operativo.

Objetivo 4: ***Investigar los distintos factores que influyen en el momento de construir una narrativa alrededor de los diferentes tipos de interacción que puede presentar un videojuego.***

Un videojuego puede presentar una gran cantidad de elementos que pueden afectar a la forma de narrar una historia. En este proyecto se ha estudiado cómo el sistema de diálogos y la toma de decisiones pueden ser los pilares para una experiencia que aproveche los recursos que ofrece el campo de los videojuegos. Sin embargo, hemos visto casos como el combate contra Psycho Mantis, donde la única forma de derrotarle es cambiando el mando de puerto, haciendo que el hardware también pueda ser un medio de interacción más allá de la utilización de los botones para aplicar acciones.

Objetivo 5: ***Redactar un documento de diseño acorde a las necesidades y los temas planteados.***

En la Sección 4.1. Diseño del Videojuego se ha podido observar el diseño base de un videojuego mediante la redacción de un documento de diseño. Con este diseño se ha facilitado el desarrollo de un prototipo, aplicando todos los temas estudiados y agilizando el proceso de desarrollo.

Objetivo 6: ***Diseñar un prototipo especificando las mecánicas, el sistema de combate, el posicionamiento de la cámara y los tipos de enemigos.***

Con todos los objetivos anteriores cumplidos, el proceso de diseño y desarrollo de un prototipo giró en torno al GDD propuesto para el videojuego completo, además de contar con los conocimientos adquiridos tras la investigación. Esto permitió completar el desarrollo de una *demo* de una duración media de 15-30 minutos, para la que se diseñaron y añadieron todos los elementos propuestos, los cuales se encuentran analizados en la Sección 4.2. Implementación en un Prototipo.

6.3. Resolución de las Hipótesis

Respecto a las hipótesis planteadas, al ser dos, se va a hacer una resolución individual para cada una.

Hipótesis 1: ***¿Cuál es el proceso de diseño a seguir en un videojuego que tiene como eje central la participación directa del jugador en el mundo de ficción que se le presenta?***

El proceso de diseño de cualquier videojuego es uno muy amplio que difícilmente puede ser definido de una forma universal, pero sí que se pueden indicar unos pasos a seguir.

A la hora de diseñar un videojuego se ha comprobado que se requiere de un tema o eje central para que la experiencia de juego gire a ello.

Definido un tema, se puede desarrollar una trama que aplique el tema deseado, en este caso, una trama que implique al jugador de manera directa. Para ello, se definieron los puntos de inflexión donde el usuario toma participación en el mundo de ficción, como es en el caso de decidir qué nombre tiene el personaje cuando el soldado le pregunta.

Por último, tal y como se puede observar en la figura 6.1, tras definir el tema y la trama del juego juntos a los acontecimientos clave, se diseñan las mecánicas y los niveles para que la historia del videojuego se desarrolle a través de la jugabilidad, lo que se conoce como diseño narrativo. De esta manera, se garantiza un diseño sólido, donde, cuanta más relación tenga el desarrollo de la trama con la jugabilidad, mayor es la sensación de inmersión en el jugador.

Figura 6.1: Pasos a seguir en el desarrollo de un videojuego.

Por otro lado, como se ha podido comprobar en las respuestas de la encuesta, la interacción es la clave para que la experiencia se sienta real y que la conexión que se genere con el mundo y sus personajes sea una difícil de romper. En caso de que el juego tenga diálogos, el jugador debe poder intervenir en ellos. Por otro lado, si se pueden tomar dos decisiones igual de válidas, pero con consecuencias distintas, el jugador debe poder intervenir. Por ello, a la hora de diseñar un videojuego con dicho enfoque en mente, es importante que todas las piezas diseñadas encajen para que el jugador pueda evadirse de la realidad y sentirse inmerso en la experiencia de juego.

Hipótesis 2: ***¿Puede un personaje no jugador simular una consciencia adaptativa en función de las acciones tomadas por el jugador en el mundo que se le presenta?***

La respuesta corta a esta pregunta es sí, sí puede, aunque únicamente será eso, una simulación. Evidentemente un personaje con una base escrita por un guionista y unas posibilidades establecidas por un programador, nunca podrá ser una entidad propia, a no ser que se realice una IA muy avanzada con redes neuronales, aunque en ese caso el descontrol sobre las decisiones de ésta es tal, que no la hacen viable para un videojuego. Sin embargo, se pueden forzar decisiones mediante la narrativa, donde, en función de la elección del jugador, el personaje actúe de una manera u otra.

Se ha podido observar con los resultados de la encuesta que los jugadores han podido sentir cómo el estado de confusión era el que más describía a Adam. Esto es debido a que la gran mayoría ha tomado decisiones relacionadas con ellos como jugadores en lugar de elegir lo que respondería el personaje. Por ejemplo, en la conversación con el militar al jugador se le concede la posibilidad de decir que su nombre es Adam, y si lo hace, la decisión de Adam será neutra, en lugar de confundida.

Por ello, un personaje sí puede simular una consciencia adaptativa a través

de las posibilidades que ofrezca el guion previamente escrito. Es importante establecer unas decisiones donde la reacción consecuente del personaje sea acorde a lo que debería, con el fin de que simule una consciencia ficticia donde el jugador no distinga entre el personaje y una persona real.

6.4. Líneas Futuras

Por último, para concluir este trabajo de fin de grado, se va a profundizar en las líneas futuras a tomar con los conocimientos adquiridos. Respecto al prototipo, se ha decidido por no continuar el desarrollo del videojuego ya que el proceso de diseño y el desarrollo de esta experiencia —de más de ocho horas tal y como se especifica en el GDD— llevaría demasiado tiempo para un equipo de cuatro personas, por lo que el contenido de este apartado se centrará en cómo se utilizará lo aprendido para proyectos futuros.

Respecto a la parte de diseño, se ha confirmado —mediante las respuestas de la encuesta— qué elementos transmiten las sensaciones deseadas. Esto se ha observado en cómo la mayoría de encuestados definieron el diseño de niveles como el propio de un videojuego del género *Soulslike*, género del cuál se ha tomado la inspiración para el estilo de los niveles. Además, se ha profundizado en la escritura de un documento de diseño real, abarcando los puntos clave que se deben considerar a la hora de establecer las ideas, partiendo del tema de la obra hasta definir los elementos que hacen accesible al producto. Stigma Protocol ha servido para llevar a cabo un flujo de trabajo que se verá reflejado en proyectos futuros, haciendo uso de las técnicas aplicadas en este trabajo de fin de grado.

Por otro lado, en la parte técnica relacionada con la programación o la animación 3D, mediante la implementación de las ideas en un prototipo real, se ha adquirido un gran número de competencias respecto a la utilización de Unreal Engine 4 y Blender. Con los conocimientos actuales, el desarrollo de los *blueprints* o las animaciones se verá altamente agilizado en caso de requerir implementarlos en proyectos futuros.

Como adición, el documento de diseño, las animaciones y el prototipo de Stigma Protocol serán incluidos en un ya existente portfolio dedicado al diseño de videojuegos.

Este proyecto ha abierto las puertas a la posibilidad de implementar ideas nuevas de una manera más ágil y concreta, ya que el aprendizaje obtenido tanto de los logros como de los errores hará de peldaño hacia arriba para trabajos de una calidad superior.

Bibliografía

[1] B. Straley and N. Druckmann, "The Last of Us," [BD-R DL, Digital PSN], Naughty Dog, 2013.

[2] R. Clark, "Crypt of the NecroDancer," [ESD, DD], Brace Yourself Games, 2015.

[3] K. Sprite, "Friday Night Funkin'," [DD], Ninjamuffin99, 2020.

[4] D. Salvato, "Doki Doki Literature Club!" [ESD, DD], Team Salvato, 2017.

[5] D. Wredden, "The Stanley Parable," [ESD, DD], Galactic Cafe, 2011.

[6] H. Kojima, "Metal Gear Solid 2," [DVD-ROM], Konami, 2001.

[7] S. Jonze, "Her," [BD], 2013.

[8] C. V. de Sousa, "La teatralidad en el espacio contemporáneo francés," in *La philologie française à la croisée de l'an 2000: panorama linguistique et littéraire.* Universidad de Granada, 2000, pp. 245–254.

[9] H. Kojima, "Metal Gear Solid," [CD-ROM], Konami, 1998.

[10] S. Conway, "A circular wall? reformulating the fourth wall for videogames," *Journal of Gaming & Virtual Worlds*, vol. 2, no. 2, pp. 145–155, 2010.

[11] M. Laidlaw, "Half Life 2," [OD, ESD, DD], Valve Corporation, 2004.

[12] M. Conforti, "OneShot," [ESD, DD], Future Cat, 2014.

[13] D. Salvato, "Doki Doki Literature Club Plus!" [ESD, DD], Team Salvato, 2021.

[14] W. W. Royce, "Managing the development of large software systems: concepts and techniques," in *Proceedings of the 9th international conference on Software Engineering*, 1987, pp. 328–338.

[15] H. Takeuchi and I. Nonaka, "The new new product development game," *Harvard business review*, vol. 64, no. 1, pp. 137–146, 1986.

[16] M. Allmer, "The 13 basic principles of gameplay design," *Retrieved March*, vol. 27, p. 2009, 2009.

[17] T. Freeman, "Creating a great design document," *Game Developer Magazine*, pp. 58–66, 1997.

[18] T. Howard, "Fallout 3," [BD, DVD, ESD, DD], Bethesda, 2008.

[19] S. Asmussen, "God of War III," [BD-ROM DL], Santa Monica, 2010.

[20] K. Tomaszkiewicz, "The Witcher 3," [DVD, ESD, BD, DD], CD Projekt RED, 2015.

[21] A. Sapkowski, *Sword of Destiny.* Independent Publishing House NOWA, 1992.

[22] R. Hunicke, M. LeBlanc, and R. Zubek, "Mda: A formal approach to game design and game research," in *Proceedings of the AAAI Workshop on Challenges in Game AI*, vol. 4, no. 1. San Jose, CA, 2004, p. 1722.

[23] Urf Academy, "Módulo 1: 8 tipos de diversión y sensación de juego," Aug. 2019, https://www.riotgames.com/darkroom/original/fac7f3cc2311ed8a14ae0bd252a6ef32:ee485d15faf2137d3cf4a76a2a495d25/pdf-viewer.pdf.

[24] P. M. P. Alonso-Geta, "Protección de la infancia y nuevas tecnologías de la comunicación: el código pegi de regulación de los videojuegos y juegos on-line," *Teoría de la Educación. Educación y Cultura en la Sociedad de la Información*, vol. 9, no. 3, pp. 29–47, 2008.

[25] A. Burch, "Borderlands 3," [DVD, ESD, BD, DD], 2K Games, 2019.

[26] S. Vanaman, "The Walking Dead," [BD, DD], Telltale Games, 2012.

[27] H. Miyazaki, "Dark Souls," [DVD, BD, ESD, DD], From Software, 2011.

[28] U. Engine, "Animation Blueprints," 2017.

[29] U. Engine, "Animation Space," 2017.

[30] U. Engine, "Unreal Motion Graphics UI Designer (UMG)," 2015.

[31] "HQUI: Progress Bars," Piontek, 2021.

[32] "HQUI: Buttons," Piontek, 2021.

[33] P. Mether, "Character customization in video games: affecting experience with visual customization," 2019.

[34] U. Engine, "Cascade Particle System," 2015.

[35] U. Engine, "Niagara Particle System," 2018.

[36] "Muzzle Flashes'," W3 Studios, 2015.

[37] UnrealCG. Disintegration Effect-Material Function - [UE4 Tutorial]. YouTube. [Online]. Available: https://www.youtube.com/watch?v=gldIJGqlWf0

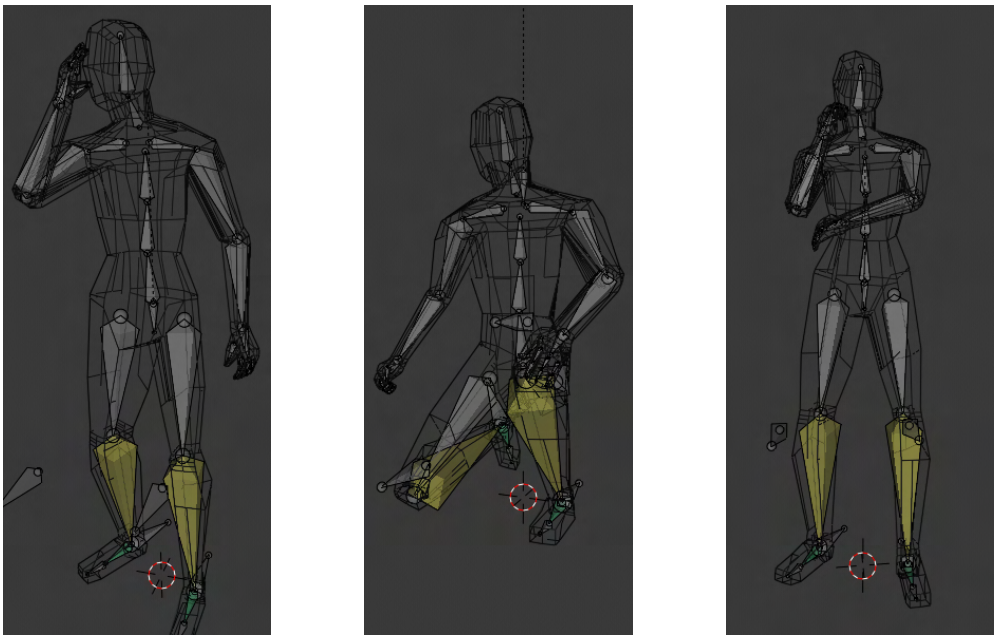
Apéndices

A

Animación en Blender

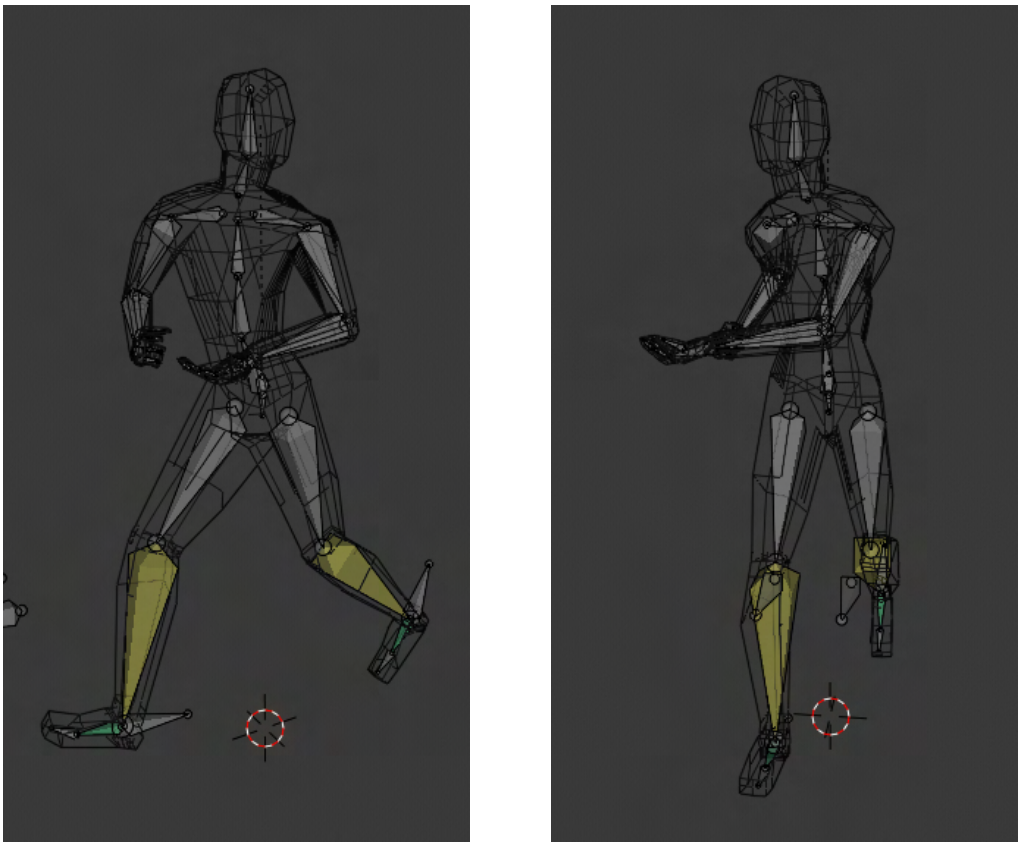
En este apéndice se encuentran capturas de las animaciones realizadas en Blender.

A.1. Animaciones del Prototipo



(a) Mano en la cabeza. (b) Agachado. (c) Pensativo.

Figura A.1: Animaciones del prototipo - Diálogos.



(a) Movimiento frontal. (b) Movimiento lateral.

Figura A.2: Animaciones del prototipo - Movimiento con fusil.

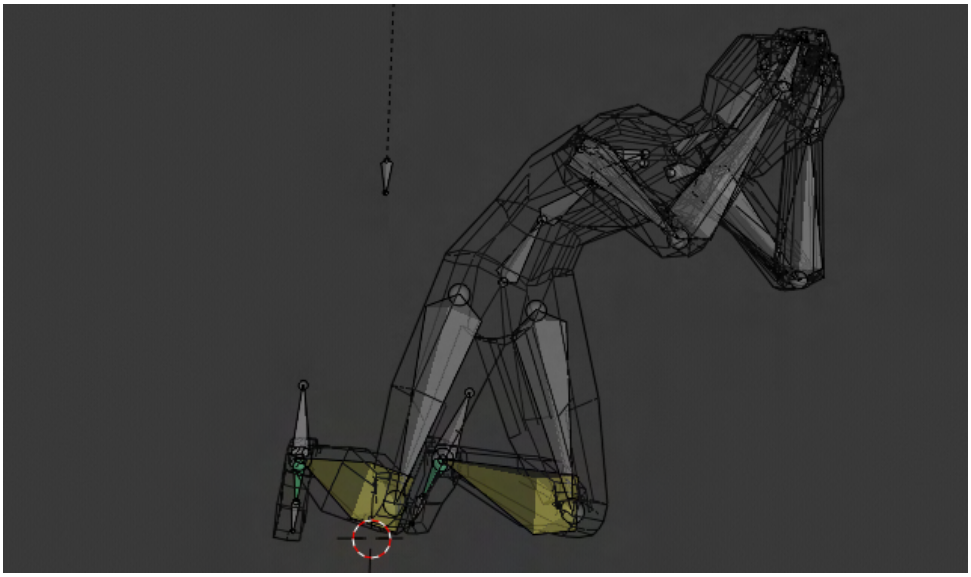


Figura A.3: Animaciones del prototipo - Gesto de dolor 1.

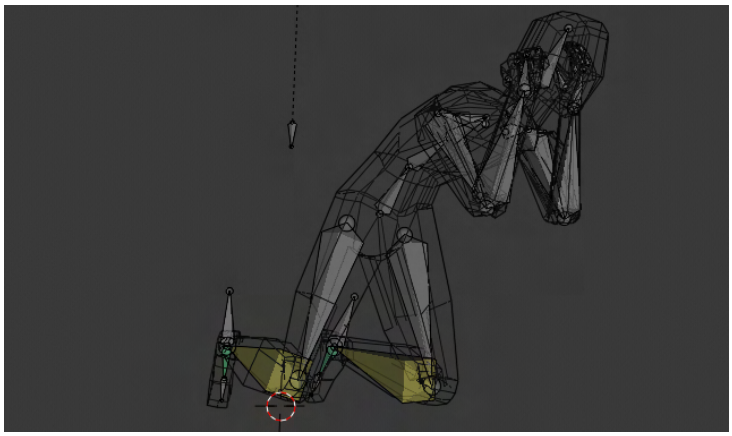


Figura A.4: Animaciones del prototipo - Gesto de dolor 2.

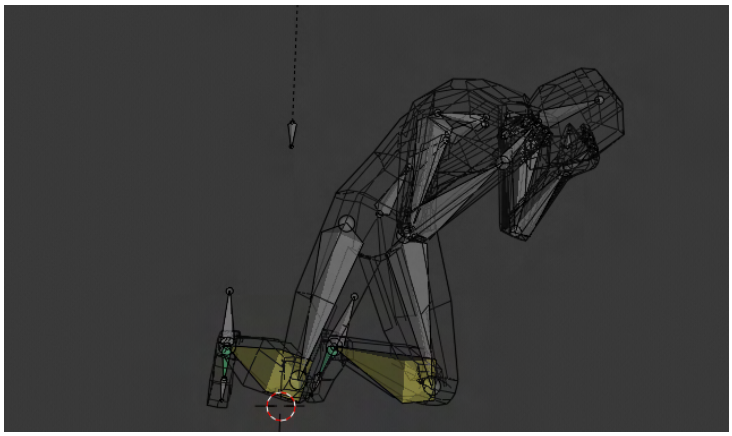


Figura A.5: Animaciones del prototipo - Gesto de dolor 3.

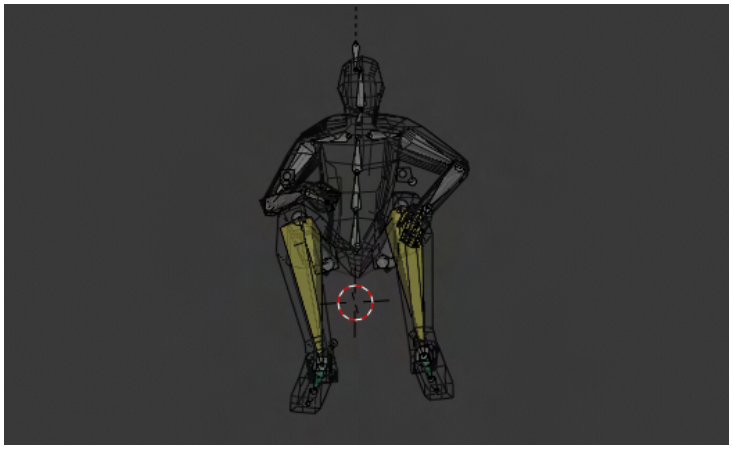
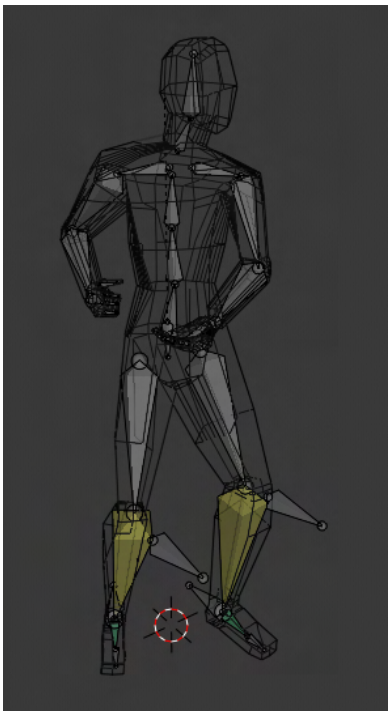


Figura A.6: Animaciones del prototipo - Personaje sentado.



(a) Enemigo quieto.

(b) Enemigo disparando.

Figura A.7: Animaciones del prototipo - Animaciones del enemigo.

Figura A.8: Animaciones del prototipo - Muerte 1.

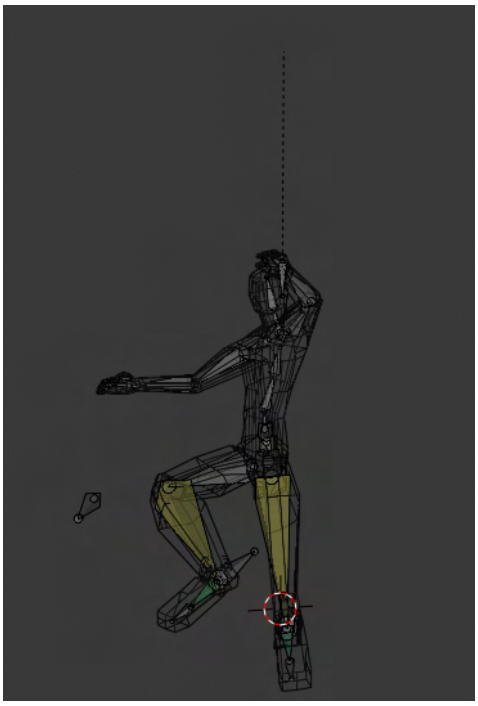


Figura A.9: Animaciones del prototipo - Muerte 2.

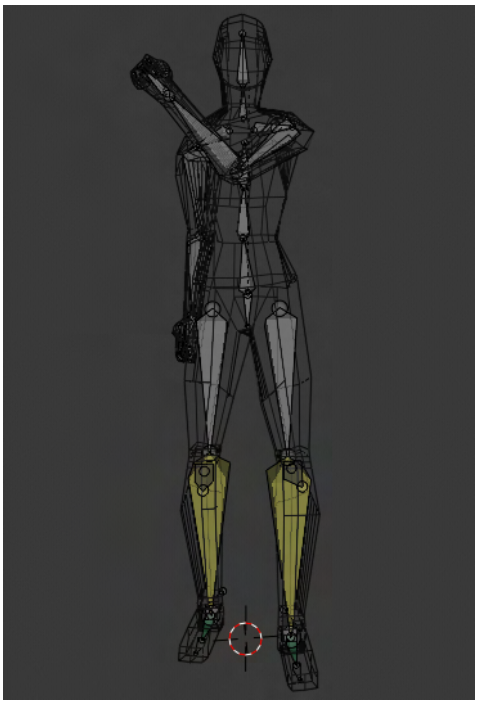
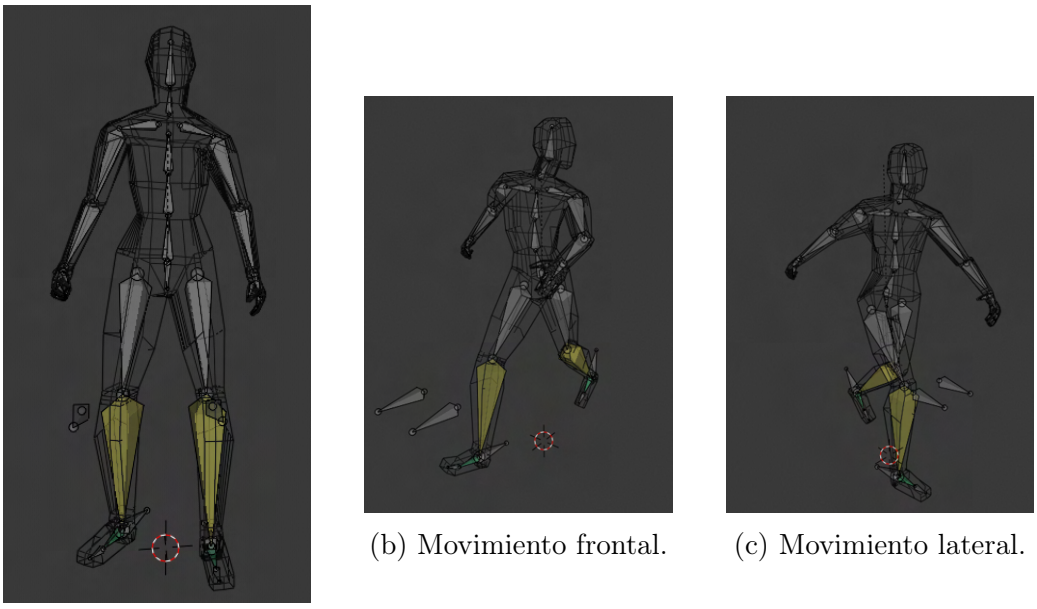


Figura A.10: Animaciones del prototipo - Personaje enseñando el brazalete.

A.2. Animaciones No Incluidas en el Prototipo



(a) Quieto.

(b) Movimiento frontal.

(c) Movimiento lateral.

Figura A.11: Animaciones no incluidas - Animaciones con katana.

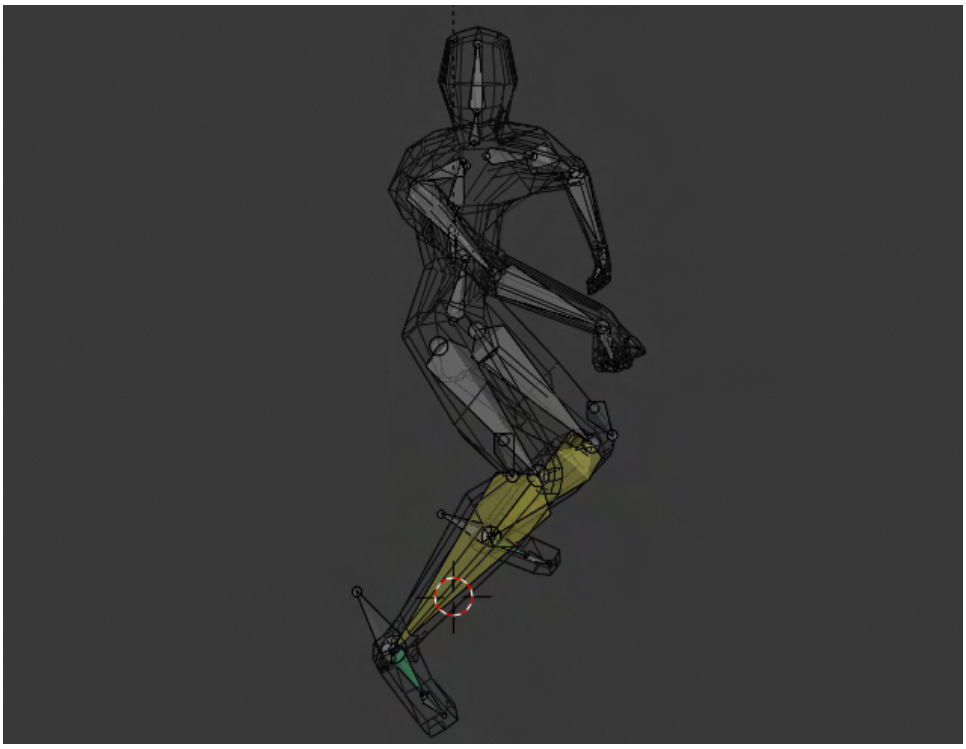


Figura A.12: Animaciones no incluidas - Ataque con katana.

(a) Muerte 1. (b) Muerte 2.

Figura A.13: Animaciones no incluidas - Muertes.

B

Guion de la Demo

En este apéndice se ha incluido el guion escrito de todas las cinemáticas y conversación para la *demo* del videojuego. En total se escribieron y realizaron un total de 3 cinemáticas y 3 conversaciones con personajes.

B.1. Llegada a Neo-Osaka

FADE IN:

EXT. CALLES DE NEO-OSAKA NOCHE

ADAM es transportado a Neo-Osaka y mira a los edificios que le rodean.

ADAM
¿Dónde estoy?

ADAM
¿A dónde me han enviado los jefes esta vez?

ADAM
¿Es esto... Neo-Osaka?

ADAM
Debería investigar.

FADE OUT

B.2. Encuentro con John

FADE IN:

EXT. CALLES DE NEO-OSAKA NOCHE

ADAM encuentra a JOHN sentado en la acera frente a un edificio. ADAM se acerca confundido a JOHN.

ADAM
¿Eh? ¿Un superviviente?

JOHN, el cuál continúa sentado, mira a ADAM.

JOHN
Ey, tú.

JOHN
¿Qué esta haciendo un Driver aquí?

JOHN
En cualquier caso, no me importa. ¿Podrías ayudarme?

JOHN se pone de pie.

RESPUESTA 1.1: Claro. ¿Qué sucede?

ADAM
Claro. ¿Qué sucede?

JOHN
Estaba con mi mujer antes de que llegaran a este lugar. Sé que que está muerta, pero...

JOHN
Necesito su anillo. Solo... quiero tener un recuerdo suyo.

JOHN
¿Me traerías el anillo?

JOHN se toca la barbilla y mira hacia el cielo.

JOHN
Conozco vuestra tecnología. Te podría mejorar el rifle. ¿Qué opinas?

RESPUESTA 2.1: De acuerdo, te devolveré el anillo de tu mujer.

ADAM
De acuerdo, te devolveré el anillo de tu mujer.

JOHN
Gracias. No te arrepentirás.

FADE OUT

RESPUESTA 2.2: No, gracias, tengo otra cosa que hacer.

ADAM
No, gracias, tengo otra cosa que hacer.

JOHN
Vete al infierno, hombre. No sabía que todos los conductores fueran tan gilipollas.

FADE OUT

RESPUESTA 1.2: ¿Por qué debería?

ADAM
¿Por qué debería?

JOHN
...

JOHN
Me desagrada la gente como tú.

JOHN
Déjame explicar qué ocurre y te diré lo que obtendrás de esto.

JOHN
Estaba con mi mujer antes de que llegaran a este lugar. Sé que que está muerta, pero...

JOHN
Necesito su anillo. Solo... quiero tener un recuerdo suyo.

JOHN
¿Me traerías el anillo?

JOHN se toca la barbilla y mira hacia el cielo.

JOHN
Conozco vuestra tecnología. Te podría mejorar el rifle. ¿Qué opinas?

RESPUESTA 2.1: De acuerdo, te devolveré el anillo de tu mujer.

ADAM
De acuerdo, te devolveré el anillo de tu mujer.

JOHN
Gracias. No te arrepentirás.

FADE OUT

RESPUESTA 2.2: No, gracias, tengo otra cosa que hacer.

ADAM
No, gracias, tengo otra cosa que hacer.

JOHN
Vete al infierno, hombre. No sabía que todos los conductores fueran tan gilipollas.

FADE OUT

B.3. Encuentro con Cadáveres

FADE IN:

EXT. CALLES DE NEO-OSAKA NOCHE

ADAM encuentra el cadáver de una persona apoyado frente a un coche. ADAM mira el cuerpo.

ADAM

...

ADAM se agacha e inspecciona el cadáver.

ADAM

¿Qué está ocurriendo aquí?

ADAM se levanta y mira al frente.

FADE OUT

B.4. Entrega del Anillo de la Esposa de John

FADE IN:

EXT. CALLES DE NEO-OSAKA NOCHE

ADAM, con el anillo de la esposa de JOHN en mano, se acerca a JOHN, el cuál se encuentra de pie.

ADAM

Tengo el anillo.

JOHN

¡Magnífico!

JOHN

No sabes lo importante que es para mí esto.
Muchas gracias. Ojalá poder volverla a ver.

ADAM
Se nota que era un sentimiento fuerte.

JOHN
Oye. ¿Tú alguna vez has amado a alguien?

ADAM mira hacia el cielo mientras el mundo comienza a presentar errores gráficos.

ADAM
¿Amar... a alguien?

RESPUESTA 1.1: Sí. Creo que alguna vez.

El mundo vuelve a la normalidad y ADAM vuelve a mirar a JOHN.

ADAM
Sí. Creo que alguna vez.

JOHN
Bueno. Lo prometido es deuda. Aquí tienes la mejora.

JOHN
Ahora tus balas deberían rebotar en la pared.

ADAM
Muchas gracias.

JOHN
Mucha suerte. Y, de nuevo, muchas gracias.

FADE OUT

RESPUESTA 1.2: No. No lo creo.

El mundo vuelve a la normalidad y ADAM vuelve a mirar a JOHN.

ADAM
No. No lo creo.

JOHN
Bueno. Lo prometido es deuda. Aquí tienes la mejora.

JOHN
Ahora tus balas deberían rebotar en la pared.

ADAM
Muchas gracias.

JOHN
Mucha suerte. Y, de nuevo, muchas gracias.

FADE OUT

B.5. Encuentro con Militar

FADE IN:

EXT. ENTRADA A CAMPAMENTO MILITAR NOCHE

ADAM encuentra a un SOLDADO frente a la entrada de un asentamiento militar.

SOLDADO
¿Quién eres tú? ¿Qué hace un Driver aquí?

ADAM se acerca al SOLDADO.

ADAM
No eres el primero que me pregunta eso. Solo estoy aquí para ayudar.

SOLDADO
...

RESPUESTA 1.1: Oye, no te preocupes, no voy a haceros daño.

ADAM
Oye, no te preocupes, no voy a haceros daño.

SOLDADO
Me alegra ver que al menos intentas ser simpático.

SOLDADO
Nosotros solo estamos intentando ayudar a los necesitados.

SOLDADO
Hemos visto a más de esas cosas ir por allí.

ADAM
Los detendré.

SOLDADO
Espero que así sea.

SOLDADO
Dime. ¿Cómo te llamas?

RESPUESTA 2.1: Adam.

ADAM
Adam.

SOLDADO
Vale, mucha suerte. Estaremos muy agradecidos si acabas con esas cosas.

FADE OUT

RESPUESTA 2.2: <Nombre de usuario>.

Adam mira al cielo y el mundo comienza a presentar errores.

ADAM
Yo...

ADAM
Mi nombre...

RESPUESTA 3.1: <Nombre de usuario>.

El mundo deja de presentar errores. ADAM vuelve a mirar al SOLDADO.

ADAM
<Nombre de usuario>.

SOLDADO
Vale, mucha suerte. Estaremos muy agradecidos si acabas con esas cosas.

FADE OUT

RESPUESTA 3.2: <Nombre de personaje>.

El mundo deja de presentar errores. ADAM vuelve a mirar al SOLDADO.

ADAM
<Nombre de personaje>.

SOLDADO
Vale, mucha suerte. Estaremos muy agradecidos si acabas con esas cosas.

FADE OUT

RESPUESTA 2.3: <Nombre de personaje>.

Adam mira al cielo y el mundo comienza a presentar errores.

ADAM
Yo...

ADAM
Mi nombre...

RESPUESTA 3.1: <Nombre de usuario>.

El mundo deja de presentar errores. ADAM vuelve a mirar al SOLDADO.

ADAM
<Nombre de usuario>.

SOLDADO
Vale, mucha suerte. Estaremos muy
agradecidos si acabas con esas cosas.

FADE OUT

RESPUESTA 3.2: <Nombre de personaje>.

El mundo deja de presentar errores. ADAM vuelve a mirar al SOLDADO.

ADAM
<Nombre de personaje>.

SOLDADO
Vale, mucha suerte. Estaremos muy
agradecidos si acabas con esas cosas.

FADE OUT

B.6. Cinemática Final

FADE IN:

EXT. CALLES DE NEO-OSAKA NOCHE

ADAM cae de rodillas en la carretera y se coloca las manos en la cabeza. El mundo comienza a presentar errores.

ADAM
Aggh...

ADAM
¿Qué me está pasando?

ADAM se agita de manera violenta.

ADAM

¿Qué estoy haciendo aquí?

ADAM

¿Por qué dudo sobre cuál es mi nombre?

ADAM

¿Quién soy?

RESPUESTA 1.1: Eres <Nombre de usuario>.

ADAM grita mientras se retuerce de dolor.

ADAM

¡Sal de mi cabeza!

FADE OUT

RESPUESTA 1.2: Eres <Nombre de personaje>.

ADAM grita mientras se retuerce de dolor.

ADAM

¡Sal de mi cabeza!

FADE OUT