UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN DESARROLLO DE
VIDEOJUEGOS Y REALIDAD VIRTUAL

Implementación de mecánica speak to action para medición de nivel de inmersión dentro de un videojuego First Person Shooter

DIEGO MORALES VELIZ.

Profesor Guía: CLAUDIA DE LA FUENTE.

Memoria para optar al título de
Ingeniero en Desarrollo de Videojuegos y Realidad Virtual

Talca – Chile
27 de Julio, 2026

UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN DESARROLLO DE
VIDEOJUEGOS Y REALIDAD VIRTUAL

Implementación de mecánica speak to action para medición de nivel de inmersión dentro de un videojuego First Person Shooter

DIEGO MORALES VELIZ.

Profesor Guía: CLAUDIA DE LA FUENTE.

Memoria para optar al título de
Ingeniero en Desarrollo de Videojuegos y Realidad Virtual

El presente documento fue calificado con nota: _____

Talca – Chile

27 de Julio, 2026

"Mientras estemos bajo el mismo cielo, no deberíamos estar tristes cuando nos separemos."
Wolfgang Grimmer

"Dudar es fracasar."
Isshin Ashina

AGRADECIMIENTOS

Agradecimientos a Mis Padres y Hermana por apoyarme en todo lo que siempre necesite y a Dios por darme la oportunidad de estudiar, haberme levantado cuando me quería dar por vencido y cuando tenía que enfrentar las peores situaciones.

TABLA DE CONTENIDOS

página

Dedicatoria i

Agradecimientos ii

Tabla de Contenidos iii

Índice de Figuras vi

Índice de Tablas xi

Resumen xii

1 Capítulo 1: Introducción 1

1.1 Objetivo General 2
1.2 Objetivos Específicos 2
1.3 Pregunta de investigación 2
1.4 Hipótesis 3
1.5 Esquema del documento 3

2 Capítulo 2: Estado del Arte 5

2.1 Juegos que usan la mecánica de interacción por voz 5
2.2 Estudios que investigan el uso de la mecánica de interacción por voz . 6

3 Capítulo 3: Marco Teorico 9

3.1 ¿Qué es la Inmersión? 9
3.2 Motores de reconocimiento de voz ASR 10

4 Capítulo 4: Marco Metodológico 13

4.1 Metodología Scrum 13
4.2 Aplicación en el proyecto 13
4.3 Inmersive Experiencie Questionare(IEQ) 14
4.4 Etapas del Proyecto 15
4.4.1 Etapa 1: Planificación 15

4.4.2 Etapa 2: Desarrollo e Implementación 15
4.4.3 Etapa 3: Implementación y Ajuste del Motor ASR 16
4.4.4 Etapa 4: Experimentación 16
4.4.5 Etapa 5: Análisis y Resultados 17

5 Capítulo 5: Diseño **19**
5.1 Etapa 1: Planificación 19
5.1.1 Diseño Experimental 21
5.1.2 Condiciones del experimento 21
5.1.3 Condición 1: Control Tradicional: Mouse y Teclado 21
5.1.4 Condición 2: Speak to Action: Controles mediante voz 22
5.1.5 Participantes 22
5.1.6 Criterios de evaluación 23
5.1.7 Criterios de Exclusión 23
5.1.8 Instrumentos del experimento 23
5.1.9 Uso del IEQ dentro del estudio 23
5.1.10 Procedimiento 23
5.1.11 Secuencia del procedimiento 24
5.1.12 Métricas a evaluar 24
5.2 Etapa 2: Desarrollo e Implementación 25
5.2.1 Product Backlog 25
5.2.2 Sprint Backlog 26

6 Capítulo 6: Implementación **30**

7 Capítulo 7: Resultados **35**
7.1 Análisis descriptivo 35
7.2 Análisis inferencial: Prueba t de Student 36

8 Capítulo 8: Conclusión **39**

Bibliografía **41**

A: Anexos **46**
A.1 Carta Gantt 46
A.2 Game Design Document(GDD) 46

A.3 Descripción General 49
A.3.1 Concepto del Juego 49
A.3.2 Forma de Interacción 49
A.4 Jugabilidad y Mecánicas 50
A.4.1 Jugabilidad 50
A.4.2 Personajes 53
A.5 Historia y Narrativa 55
A.5.1 Diseño Narrativo 55
A.6 Niveles y Progresión 57
A.6.1 Sinopsis del nivel 57
A.6.2 Camino Crítico 57
A.7 Interfaces 60
A.7.1 Hud y Pantallas 60
A.7.2 Controles 61
A.7.3 Propuesta de Audio 62
A.7.4 Propuesta Visual 62
A.8 Implementación 65
A.8.1 Gameplay de la Version Final del prototipo 65
A.8.2 Codigos Principales del Sistema de Interacción por Voz 65
A.9 Preguntas del cuestionario IEQ 85
A.10 Resultados Obtenidos 89
A.10.1 Tabla General de Datos Obtenidos. 89
A.10.2 Respuestas Versión por Voz 90
A.10.3 Respuestas Versión Teclado y Ratón 105

ÍNDICE DE FIGURAS

página

6.1 Primeros commits del prototipo. 31
6.2 Versión funcional del prototipo. 31
6.3 Implementación inicial del UI y tutorial. 32
6.4 Commits de trabajo del nivel prototipo. 32
6.5 Optimización de Vosk. 33
6.6 Versión final del prototipo, con ambas versiones funcionando, teclado ratón y voz. 34
6.7 Commits finales. 34

7.1 Media por subdimensión del cuestionario IEQ, comparando la condición de Teclado/Ratón y Voz (N=20). 36
7.2 Resultados de la prueba t de Student para muestras relacionadas, muestra completa (N=20). 37
7.3 Resultados de la prueba t de Student para muestras relacionadas, subconjunto contrabalanceado (N=8). 38

A.1 Carta Gantt. 46
A.2 Superhot. 63
A.3 Davigo. 63
A.4 Kour.io. 64
A.5 CloudPunk. 64
A.6 Tales of the Neonsea. 65
A.7 Imagen de Unity 3D del manager principal Vosk y sus componentes . 66
A.8 Diagrama de Flujo del Funcionamiento de Vosk 67
A.9 Diagrama de Clases del Funcionamiento de Vosk 68
A.10 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 69
A.11 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 70
A.12 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 70

A.13 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 71
A.14 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 71
A.15 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 72
A.16 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 72
A.17 Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono. 73
A.18 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 73
A.19 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 74
A.20 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 74
A.21 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 75
A.22 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 75
A.23 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 76
A.24 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 76
A.25 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 77
A.26 Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk. 78
A.27 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 78
A.28 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 79
A.29 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 79

A.30 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 80
A.31 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 81
A.32 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 82
A.33 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.. 82
A.34 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 83
A.35 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 84
A.36 Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador. 85
A.37 Respuestas Versión por Voz – Pregunta 1. 90
A.38 Respuestas Versión por Voz – Pregunta 2. 91
A.39 Respuestas Versión por Voz – Pregunta 3. 91
A.40 Respuestas Versión por Voz – Pregunta 4. 92
A.41 Respuestas Versión por Voz – Pregunta 5. 92
A.42 Respuestas Versión por Voz – Pregunta 6. 93
A.43 Respuestas Versión por Voz – Pregunta 7. 93
A.44 Respuestas Versión por Voz – Pregunta 8. 94
A.45 Respuestas Versión por Voz – Pregunta 9. 94
A.46 Respuestas Versión por Voz – Pregunta 10. 95
A.47 Respuestas Versión por Voz – Pregunta 11. 95
A.48 Respuestas Versión por Voz – Pregunta 12. 96
A.49 Respuestas Versión por Voz – Pregunta 13. 96
A.50 Respuestas Versión por Voz – Pregunta 14. 97
A.51 Respuestas Versión por Voz – Pregunta 15 97
A.52 Respuestas Versión por Voz – Pregunta 16 98
A.53 Respuestas Versión por Voz – Pregunta 17. 98
A.54 Respuestas Versión por Voz – Pregunta 18. 99
A.55 Respuestas Versión por Voz – Pregunta 19. 99
A.56 Respuestas Versión por Voz – Pregunta 20. 100

A.57 Respuestas Versión por Voz – Pregunta 21. 100
A.58 Respuestas Versión por Voz – Pregunta 22. 101
A.59 Respuestas Versión por Voz – Pregunta 23. 101
A.60 Respuestas Versión por Voz – Pregunta 24. 102
A.61 Respuestas Versión por Voz – Pregunta 25. 102
A.62 Respuestas Versión por Voz – Pregunta 26. 103
A.63 Respuestas Versión por Voz – Pregunta 27. 103
A.64 Respuestas Versión por Voz – Pregunta 28. 104
A.65 Respuestas Versión por Voz – Pregunta 29. 104
A.66 Respuestas Teclado y Ratón – Pregunta 1. 105
A.67 Respuestas Teclado y Ratón – Pregunta 2. 106
A.68 Respuestas Teclado y Ratón – Pregunta 3. 106
A.69 Respuestas Teclado y Ratón – Pregunta 4. 107
A.70 Respuestas Teclado y Ratón – Pregunta 5. 107
A.71 Respuestas Teclado y Ratón – Pregunta 6. 108
A.72 Respuestas Teclado y Ratón – Pregunta 7. 108
A.73 Respuestas Teclado y Ratón – Pregunta 8. 109
A.74 Respuestas Teclado y Ratón – Pregunta 9. 109
A.75 Respuestas Teclado y Ratón – Pregunta 10. 110
A.76 Respuestas Teclado y Ratón – Pregunta 11. 111
A.77 Respuestas Teclado y Ratón – Pregunta 12. 111
A.78 Respuestas Teclado y Ratón – Pregunta 13. 112
A.79 Respuestas Teclado y Ratón – Pregunta 14. 112
A.80 Respuestas Teclado y Ratón – Pregunta 15. 113
A.81 Respuestas Teclado y Ratón – Pregunta 16. 113
A.82 Respuestas Teclado y Ratón – Pregunta 17. 114
A.83 Respuestas Teclado y Ratón – Pregunta 18. 114
A.84 Respuestas Teclado y Ratón – Pregunta 19. 115
A.85 Respuestas Teclado y Ratón – Pregunta 20. 115
A.86 Respuestas Teclado y Ratón – Pregunta 21. 116
A.87 Respuestas Teclado y Ratón – Pregunta 22. 116
A.88 Respuestas Teclado y Ratón – Pregunta 23. 117
A.89 Respuestas Teclado y Ratón – Pregunta 24. 117
A.90 Respuestas Teclado y Ratón – Pregunta 25. 118

A.91 Respuestas Teclado y Ratón – Pregunta 26. 119
A.92 Respuestas Teclado y Ratón – Pregunta 27. 119
A.93 Respuestas Teclado y Ratón – Pregunta 28. 120
A.94 Respuestas Teclado y Ratón – Pregunta 29. 120
A.95 Respuestas Teclado y Ratón – Pregunta 30. 121
A.96 Respuestas Teclado y Ratón – Pregunta 31. 122

ÍNDICE DE TABLAS

página

3.1 Comparación de motores ASR considerados para el prototipo. 11

5.1 Product Backlog 25
5.2 Detalles de cada uno de los Sprints. 26
5.3 Sprint backlog 29

A.1 Armas del prototipo. 53
A.2 Habilidades del prototipo. 54
A.3 Resumen general de resultados: puntaje total (muestra completa y contrabalanceada) y análisis por subdimensión del IEQ. 89

RESUMEN

La industria del videojuego está en constante movimiento, integrando nuevas modalidades de interacción destinadas a mejorar la inmersión del jugador. Desde los controles de guitarra pasando por controles de movimiento hasta los nuevos dispositivos de Realidad Virtual.

Una modalidad poco estudiada pero con gran potencial se basa en la utilización de la voz como método de control. Aunque ya disponemos de juegos y estudios que han utilizado la voz mediante comandos, la evidencia sugiere que la voz, como complemento de las entradas convencionales puede ofrecer un aumento en la inmersión y también favorece la accesibilidad para personas con discapacidades motoras.

Diversos trabajos reportan aumentos en la inmersión al introducir comandos de voz, mientras que otros se centran únicamente en aspectos técnicos de su implementación en videojuegos FPS. Sin embargo, ninguno de estos trabajos avanza hacia una evaluación empírica de los efectos de la voz sobre la inmersión del jugador específicamente en este género, lo que constituye el vacío de investigación que motiva este trabajo: la ausencia de evidencia empírica sobre el impacto inmersivo de la interacción por voz en videojuegos FPS.

Ante la ausencia de evidencia empírica sobre el impacto inmersivo de la interacción por voz en videojuegos FPS, se propuso el desarrollo de un prototipo de videojuego FPS con dos sistemas de control: uno basado en la mecánica de interacción por voz Speak to Action y otro convencional (teclado y mouse). El desarrollo del prototipo se guió mediante la metodología ágil Scrum, mientras que la inmersión percibida por los jugadores se midió mediante el cuestionario IEQ (Immersive Experience Questionnaire).

Los resultados obtenidos mostraron un aumento estadísticamente significativo en la inmersión reportada al utilizar la interacción por voz en comparación con el control convencional, con un tamaño de efecto grande, tendencia que se mantuvo consistente incluso al controlar el orden de aplicación de las condiciones. Estos hallazgos aportan evidencia empírica novedosa sobre el potencial de Speak to Action como mecánica capaz de aumentar significativamente la inmersión del jugador dentro del género FPS.

1. Capítulo 1: Introducción

La industria del videojuego continúa creciendo incluyendo nuevas modalidades de interacción que buscan maximizar la inmersión del jugador. Desde periféricos o dispositivos como guitarras o controles de movimiento, hasta tecnologías de realidad virtual, cada innovación ha otorgado una capa adicional de experiencia interactiva.

Sin embargo, existe una forma de interacción verbal (también denominada Speak to Action), que permite al jugador realizar acciones mediante órdenes verbales, es una posible forma de interacción, todavía poco estudiada y que permitiría incrementar la inmersión así como la accesibilidad. Esto puede verse ya en videojuegos como Lifeline[1] o Tom Clancy's Endwar[2], pero parece que, hasta el momento, esta interacción se ha visto confinada a géneros específicos como el survival horror o la estrategia en tiempo real, quedando relegados otros géneros como el de First Person Shooter.

Estudios previos como el de Mattey et al.[3] el de N.Zargrahm et al.[4] y Setiawan et al.[5] ha demostrado que los comandos de voz pueden aumentar la participación del jugador, pero la mayoría se ha centrado en aspectos técnicos y no en la medición empírica de la inmersión, especialmente en FPS, un género donde la respuesta rápida y la concentración son cruciales.

Por lo tanto, la presente investigación se originó en la propuesta de implementar el mecanismo de interacción por voz Speak to Action en un videojuego FPS, para así estudiar su efectividad en el sentido de la inmersión del jugador. Se optará por implementar una metodología ágil tipo Scrum, para llevar a cabo un desarrollo iterativo del prototipo y donde se irán incorporando los feedbacks que vayan surgiendo. Para estudiar la inmersión se aplicará el Immersive Experience Questionnaire (IEQ), test que ha sido validado durante otros trabajos de investigación anteriores. Se evaluarán

comparativamente diferentes motores de reconocimiento de voz (ASR), siendo Vosk uno de ellos, para encontrar el motor más robusto y eficiente a la hora del uso de un videojuego.

Esta investigación aportará evidencia y conocimiento empírico de un campo no estudiado como es el fenómeno Speak to Action en videojuegos FPS y sus implicaciones en diseño inmersivo.

1.1. Objetivo General

Implementar la mecánica de interacción por voz Speak to Action en un videojuego FPS para analizar su impacto en la inmersión del jugador.

1.2. Objetivos Específicos

- Evaluar comparativamente distintos motores de reconocimiento de voz (ASR) a nivel teórico para seleccionar el motor más viable a implementar en Unity.
- Diseñar un prototipo de videojuego FPS que integre la mecánica Speak to Action, considerando comandos clave de control por voz y su equivalente en control convencional.
- Realizar pruebas experimentales con usuarios para comparar la versión convencional y la versión con interacción por voz, utilizando instrumentos de medición de inmersión(IEQ).
- Analizar los resultados obtenidos con el fin de evaluar diferencias en la inmersión percibida, entre ambas condiciones y así comprobar y verificar la hipótesis.

1.3. Pregunta de investigación

¿La implementación de la mecánica Speak to Action en un videojuego FPS aumenta el nivel de inmersión del jugador en comparación con el control convencional basado en teclado y mouse?.

1.4. Hipótesis

Los jugadores que utilicen la versión con la mecánica Speak to Action reportarán un mayor nivel de inmersión subjetiva que en la versión con control convencional.

1.5. Esquema del documento

En este documento se verán los capítulos:

- Capítulo 2: Estado del Arte, donde se mencionan juegos que utilizan la mecánica de interacción por voz y estudios que investigan el uso de la mecánica de interacción por voz.
- Capítulo 3: Marco Teórico, donde se describe la definición de inmersión y cómo se aborda esta en el contexto de los videojuegos y también se revisan diferentes motores de reconocimiento de voz ASR que se comparan de forma teórica para decidir cual se implementará dentro del prototipo de la investigación.
- Capítulo 4: Metodología, donde se describen las etapas de la metodología scrum que se usarán para el desarrollo de la investigación, el Immersive Experience Questionnaire(IEQ), que será el cuestionario que se usará para la experimentación dentro de la investigación, además de las diferentes etapas que se seguirán para el desarrollo, etapas como: Planificación, Desarrollo e implementación, Implementación y Ajuste del Motor ASR, Experimentación, y Análisis y Resultados.
- Capítulo 5: Diseño, donde se desarrolla la etapa 1 de planificación, mencionando el proceso de investigación, comparación de motores ASR, carta gantt y creación del GDD y la etapa 2: desarrollo e implementación, donde se desarrolla la estructura de trabajo con sprints que se seguirá para el desarrollo del prototipo.
- Capítulo 6: Implementación, donde se explica cómo fue el proceso del desarrollo del prototipo, sus funcionalidades y sistemas principales hasta llegar a una versión completa y su posterior implementación del motor ASR seleccionado, explicando como fue el proceso de funcionamiento y optimización de dicho

motor hasta llegar a las dos versiones del prototipo, la de voz junto a la de teclado y ratón para los experimentos.

- Capítulo 7: Resultados, donde se muestran los resultados obtenidos de las encuestas y los datos obtenidos de ambas muestras, posteriormente se describe su análisis y si este es relevante para el cumplimiento de los objetivos y la hipótesis, además de análisis estadísticos basados directamente en la muestra.
- Capítulo 8: Conclusión, donde se recuerda lo que se hizo, válida cumplimiento de cada objetivo, y se comenta trabajo futuro que servirá para más investigaciones.

2. Capítulo 2: Estado del Arte

En este capítulo se presentarán diferentes juegos que utilizan la mecánica de interacción por voz(Speak to Action) y diferentes estudios que investigan varias áreas de aplicación de esta mecánica, además de sus efectos en la inmersión, accesibilidad y eficacia técnica.

2.1. Juegos que usan la mecánica de interacción por voz

Con el paso del tiempo, varios videojuegos han incluido en su jugabilidad la interacción por voz como eje principal. Lifeline (2003), un videojuego de aventura y terror de Sony Computer Entertainment para PlayStation 2, fue uno de los primeros, ya que, tras poderse jugar con el micrófono de la consola, permitía ejecutar acciones y jugar al personaje principal mediante reconocimiento de voz[1].Más tarde, Tom Clancy's EndWar (2008) de Ubisoft, introdujo una mecánica parecida en un juego de estrategia y acción, ya que se podía controlar unidades militares únicamente a través de comandos por voz[2]. Después, One Hand Clapping (2020) de Bad Dream Games abarcó la interacción vocal creativa; donde los jugadores tienen que resolver acertijos cantando o tarareando con el micrófono, siempre integrando voz y ritmo como elemento central de la experiencia[6].En años recientes, Mage Arena (2025) incorpora comandos de voz en combates mágicos dentro de un entorno competitivo multijugador, donde la pronunciación y claridad de los hechizos determinan la efectividad del ataque[7]. Por último, el proyecto Shout & Shoot (aún en desarrollo) propone un enfoque innovador dentro del género FPS multijugador, en el que las acciones del jugador dependen del tono y volumen de la voz, siendo el único shooter actual que utiliza completamente la mecánica Speak to Action[8].

Al profundizar en el funcionamiento técnico de estas implementaciones, se identifican dos enfoques de interacción por voz. El primero, basado en *reconocimiento de comandos mediante palabras clave*, es empleado por Lifeline, Tom Clancy's EndWar y Mage Arena. Lifeline [1] utiliza un diccionario cerrado de más de 5.000 palabras organizadas por contexto de juego, bajo un esquema de *push-to-talk*. EndWar [2] licenció el middleware comercial *VoiceIn* de Fonix Speech [9], empleando una gramática de sintaxis fija (`<unidad><orden><información extra>`) en lugar de habla libre. Mage Arena [7], más reciente, solo escucha activamente la palabra clave del hechizo seleccionado en el libro de hechizos del jugador, aceptando coincidencias parciales para comandos de dos términos [10].El segundo enfoque, empleado por One Hand Clapping y Shout & Shoot, prescinde por completo del reconocimiento de comandos hablados y se basa en el análisis de señal acústica cruda del micrófono. One Hand Clapping [6] analiza la frecuencia fundamental (tono) de la voz del jugador en tiempo real dentro de Unity, previa una etapa de calibración del rango vocal individual [11], mientras que Shout & Shoot [12] utiliza únicamente el volumen o amplitud del micrófono para determinar la visibilidad del jugador ante sus oponentes y la activación de sus armas.

Esta revisión técnica orientó decisiones concretas del sistema de voz desarrollado: se optó por el enfoque de reconocimiento de comandos por palabras clave, más adecuado para acciones discretas de un FPS que el análisis de señal acústica cruda, se filtra el vocabulario activo según el contexto del jugador (por ejemplo, solo se procesan comandos de recarga si el arma correspondiente está equipada, como en Lifeline y Mage Arena), y se utiliza coincidencia por subcadena sobre los resultados parciales del motor Vosk, siguiendo la lógica de reconocimiento parcial identificada en Mage Arena.

2.2. Estudios que investigan el uso de la mecánica de interacción por voz

Han existido diversas investigaciones han explorado la interacción por voz en videojuegos, evaluando su impacto en la inmersión, la accesibilidad y la eficacia técnica de los sistemas de reconocimiento de voz (ASR).
Primero se debe mencionar que como se demostró en el estudio de Fraser Allison et

al[13].este medio de interacción por voz lleva existiendo desde hace más de 40 años, incluida su experimentación y a pesar de eso, aún hoy en dia esta modalidad sigue siendo muy marginal y poco estudiada.

Estudios como los de Susumu Harada et al[14]. han demostrado que el uso de vocalizaciones no verbales(sonidos que se emiten sin usar palabras) pueden ofrecer tiempos de respuesta significativamente menores que comandos de voz tradicionales, acercándose al rendimiento del teclado en videojuegos de acción.

Estudios como el de Nilsson y Ohman[15] que se enfocaron en la implementación de un sistema de comandos en lenguaje natural(texto) para controlar acciones dentro de un juego demostró que interfaces de lenguaje natural pueden alcanzar altos niveles de precisión en videojuegos cuando el dominio de interacción es acotado y bien definido.

Mattey et al.[3] demostraron que los controles por voz incrementan significativamente la inmersión y participación del jugador frente a los controles tradicionales, al permitir una conexión más natural con el entorno del juego.

Milan WevelSiep et al[16]. investigaron la integración de sistemas de diálogo controlados por voz en entornos narrativos dentro de videojuegos, abordando las limitaciones de los enfoques existentes.

En el estudio de Andrei Copaceanu et al[17]. se investigó el cual metodo implementación de entrada por voz es más eficaz, mecánica independiente o alternativa tradicional a métodos de entrada, aplicando esta mecánica en un forma de un videojuego narrativo.

Graham y Wright et al.[18] evaluaron el motor Vosk, mostrando que sus modelos más optimizados lograron una precisión del 90 % y una latencia de 1,2 s, confirmando su viabilidad para juegos en tiempo real, aunque requiere gran optimización técnica para mantener su rendimiento estable.

Por su parte, Zargham et al.[4] investigaron la interacción verbal con NPCs, identificando su alto potencial inmersivo en géneros narrativos como los RPG, aunque los FPS fueron considerados menos aptos por su enfoque en la acción. Aun así, los autores destacaron el valor de la voz para aumentar la conexión emocional y la accesibilidad.

Investigaciones como las de Abdoulqadir y Fernando Loizides[19] destacan que los juegos digitales con reconocimiento de voz pueden aumentar la motivación y autonomía del usuario, aunque persisten desafíos técnicos relacionados con la precisión

del ASR y ruido ambiental.
Un trabajo como el de Strzałko[20] analiza los enfoques y desafíos de implementar el reconocimiento de voz en videojuegos, destacando su potencial en medios como VR, accesibilidad y control de interfaces, así como limitaciones técnicas asociadas a la precisión.
Setiawan et al.[5] implementaron un prototipo FPS controlado por voz en Unity, comprobando la eficacia técnica de los comandos vocales para disparo y movimiento, aunque con limitaciones por ruido y calidad del micrófono, y sin evaluar directamente el impacto en la inmersión.
Más recientemente, el estudio ACVI (AI Companion with Voice Interaction)[21] introdujo la comunicación por voz en tiempo real con compañeros de IA en el videojuego Arena Breakout: Infinite, logrando un 87,2 % de comprensión de comandos y mejorando la fluidez y naturalidad de la interacción.

En términos generales, estas investigaciones evidencian que la interacción por voz puede aumentar la inmersión, accesibilidad y realismo, aunque la mayoría de estudios se centran en su eficiencia técnica más que en su impacto emocional o de experiencia. Esto revela una oportunidad de investigación en explorar la mecánica de interacción por voz Speak to Action dentro del género FPS, enfocándose en medir su impacto en la inmersión del jugador.

3. Capítulo 3: Marco Teorico

En este apartado se presentarán tanto las características del género del tipo de videojuego que se implementará, la definición del concepto de inmersión en el contexto de los videojuegos y una revisión y evaluación comparativa de los diferentes motores ASR que existen[22].

3.1. ¿Qué es la Inmersión?

La inmersión en videojuegos es un estado psicológico donde el jugador se siente totalmente absorbido por el entorno digital, olvidando el mundo físico y experimentando una sensación de "estar allí". Se logra mediante la estimulación sensorial, una narrativa atrapante y mecánicas intuitivas que generan una profunda conexión emocional y concentración.

De acuerdo con Brown y Cairns[23], la inmersión no es un estado único sino un fenómeno gradual que se despliega en tres niveles progresivos. El primer nivel, denominado engagement(compromiso), requiere que el jugador invierta tiempo, esfuerzo y atención en aprender a jugar y familiarizarse con los controles. Desde ahí, el jugador puede alcanzar el segundo nivel, engrossment(absorción), en el cual las características del juego logran afectar directamente las emociones del jugador y los controles se vuelven "invisibles", es decir, el jugador deja de ser consciente de estar utilizándolos. Finalmente, el nivel más profundo es la total immersion (inmersión total),donde el jugador experimenta una sensación de presencia y desconexión de la realidad tan intensa que el juego se convierte en lo único que importa.

Los conocidos FPS(First Person Shooter) son videojuegos donde el jugador lleva a cabo la acción desde el punto de vista del personaje, mientras divisa a través de sus ojos y cuyo núcleo jugable reside en el combate usando armas de fuego o con proyectiles. Tiene una perspectiva en primera persona(subjetiva),Las mecánicas de base son apuntar y disparar,cuerpo a cuerpo, movimientos más fluidos, con elevado dinamismo inmediato y elementos de HUD muy bajos en pantalla. Los FPS buscan una inmersión sensorial y emocional elevada gracias a su perspectiva subjetiva, además de que estudios como el de Lennart Nacke y Craig A. Lindley[24] que han demostrado que los juegos FPS son particularmente adecuados para analizar fenómenos como la inmersión y flow, debido a sus mecánicas intensas en combate y su perspectiva en primera persona, buscando también la presencia del jugador dentro del juego, siendo un ámbito importante en los estudios de Tamborini et al.[25] y McCall et al.[26],además según estudios de Kiran Ijaz et al[27]. han demostrado la importancia que tiene la inmersión y la autonomía dentro del disfrute del jugador, siendo la inmersión un predictor importante de la motivación y la intención de uso futuro en experiencias inmersivas, lo que los convierte en un campo ideal para evaluar la influencia de la voz como medio de interacción y más teniendo en cuenta que ha sido un medio de interacción que ha llevado existiendo por más de 42 años[13] y no ha existido una implementación reconocida de esta mecánica en algun FPS.

3.2. Motores de reconocimiento de voz ASR

En este apartado se revisarán las características de diferentes motores de interacción por voz (ASR), como motor, soporte, plataformas, latencia/rendimiento, ventajas y desventajas.

Dichos motores presentan diferentes ventajas y desventajas entre sí, las cuales se resumen comparativamente en la Tabla 3.1. A continuación se detallan las características de cada uno.

Motor	Tipo	Licencia	Plataformas	Ventajas	Desventajas
Vosk	Offline	Gratis / Código abierto	Windows, Linux, Android, iOS	Soporta múltiples idiomas; funciona sin conexión; alta personalización; comunidad activa.	Precisión variable según modelo; requiere optimización para latencia; sensible al ruido.
Google Cloud Speech to Text	Online	Pago (API)	Android, iOS, Windows, macOS, Linux, VR/AR	Alta precisión; soporte para 120 idiomas; robusto; excelente documentación.	Requiere Internet; latencia dependiente de red; costos por uso; privacidad.
Undertone (Whisper Offline)	Offline	Pago	Multiplataforma en Unity	Alta precisión; 99 idiomas; respuesta rápida; traducción; tamaño reducido.	Requiere licencia; depende del proveedor.
Recognissimo	Offline	Pago (con versión gratuita)	Windows, macOS, Linux, Android, Quest, iOS, visionOS, WebGL	Ligero y rápido; configuración simple; 21 idiomas; expresiones regulares.	Licencia de pago; modelos pesados; comunidad pequeña.

Tabla 3.1: Comparación de motores ASR considerados para el prototipo.

Como primer motor ASR se tiene a Vosk[28]. Vosk es un reconocido motor ASR (Automatic Speech Recognition) de código abierto que habilita el reconocimiento de voz offline y es compatible con varios lenguajes y plataformas, como Windows, Linux, Android e iOS. Su principal ventaja es su bajo consumo de recursos, lo que habilita su uso incluso en dispositivos móviles, así como una gran comunidad, así como una activa documentación y uso del motor. Pese a lo anterior, su precisión podría variar en función del modelo de lenguaje de entrada y de la calidad del audio, habiéndo observado latencia y errores de reconocimiento en ambientes ruidosos y con vocabularios no entrenados. En investigaciones recientes[18], Vosk demostró buen rendimiento tras ciertos ajustes de optimización, lo que le otorga potencialidad para aplicaciones del ámbito de videojuegos interactivos.

Otra opción a revisar es Speech Recognition using Google Cloud[29, 30], el cual es un motor ASR basado en la nube que requiere conexión a Internet para funcionar. Es multiplataforma, compatible con Android, iOS, Windows, macOS, Linux y entornos VR/AR, y ofrece soporte para más de 120 idiomas[29, 30]. Su principal fortaleza radica en su alta precisión, modelos robustos y soporte comercial, lo que lo hace ideal para aplicaciones con necesidades profesionales. Sin embargo, presenta dependencia total de red, con posibles problemas de latencia en conexiones inestables, además de

ser un servicio de pago sujeto a licencias y costos por uso de API. También plantea riesgos de privacidad al procesar datos en servidores externos.

Otra alternativa a revisar es Undertone Offline Whisper AI Voice Recognition[31], el cual es un motor ASR offline diseñado específicamente para Unity, basado en la tecnología Whisper AI. Soporta 99 idiomas, ofrece traducción integrada, alto rendimiento y compatibilidad multiplataforma, lo que lo hace adecuado para experiencias de juego inmersivas. Entre sus ventajas destacan su funcionamiento sin conexión a internet, precisión elevada, bajo peso(77 mb) y respuesta rápida[31]. No obstante, su principal desventaja es el costo de licencia, ya que no es una herramienta gratuita.

Por último se tiene a Recognissimo: Offline Speech Recognition[32, 33] es otra opción a revisar. Este es un plugin ASR offline para Unity, compatible con Windows, macOS, Linux, Android, iOS, Meta Quest, visionOS y WebGL. Soporta más de 21 idiomas y está desarrollado en C++, ofreciendo baja latencia y alto rendimiento[32, 33]. Entre sus ventajas destacan su modo sin conexión, instalación sencilla sin necesidad de código, gestión automática de almacenamiento y permisos, y detección de voz integrada. Es una opción ligera, eficiente y estable para proyectos interactivos. Sin embargo, su principal desventaja es que requiere licencia comercial, lo que implica un costo monetario para su uso.

4. Capítulo 4: Marco Metodológico

4.1. Metodología Scrum

La metodología a revisar para esta investigacion es la metodología Scrum[34, 35, 36, 37, 38, 39], la cual es un enfoque ágil de desarrollo de software basado en el trabajo iterativo y colaborativo. Divide el proyecto en ciclos cortos llamados sprints, que suelen durar entre dos y cuatro semanas. El equipo realiza una funcionalidad por cada Sprint, revisa la forma de haberla implementado y elige las tareas a realizar en el siguiente Sprint, de forma que se produce de manera continua una adaptación del desarrollo de acuerdo a los resultados y el feedback recibido. Scrum se basa en tres roles fundamentales: el Product Owner que prioriza y define lo que se debe hacer, el Scrum Master (SM) que acompaña y apoya el uso de la metodología en el equipo y el equipo de desarrollo que destaca por su autonomía y auto-organización para conseguir el objetivo de cada Sprint. Entre sus ventajas más importantes se encuentran: la flexibilidad para adaptarse a los cambios, la mejora continua del producto, la interacción continua entre el propio equipo y la continua entrega de versiones funcionales que a su vez se adaptan al feedback del cliente.

4.2. Aplicación en el proyecto

Para el desarrollo del prototipo se utilizará la metodología Scrum adaptada, definida en[34, 35, 36, 37, 38, 39] como un framework ligero orientado a resolver problemas complejos mediante soluciones adaptativas[34, 35, 36, 37, 38, 39]. Aunque

Scrum está diseñado para equipos multidisciplinarios, en esta investigación el desarrollo será realizado por una sola persona. Debido a ello, los roles del Scrum Team y Srum Master se condensan en el propio investigador. El proyecto se organizará en Sprints, ciclos de trabajo de duración fija donde se llevan a cabo todas las actividades necesarias para avanzar en el producto. Dado que esta investigación requiere construir un prototipo funcional de un shooter FPS para evaluar la inmersión mediante controles por voz, se optará por Sprints semanales en lugar de mensuales. Esta división más breve permite avanzar de manera incremental, obtener retroalimentación temprana y ajustar rápidamente el desarrollo, especialmente durante la experimentación con diferentes motores de reconocimiento de voz (ASR). La planificación del Sprint definirá el trabajo a realizar cada semana. El conjunto formado por el objetivo del Sprint, los elementos seleccionados y el plan para desarrollarlos constituye el Sprint Backlog [34, 35, 36, 37, 38, 39]. Dentro de la investigación, los Sprints pueden ser descompuestos en sub-tareas aún más pequeñas para determinar un avance controlado y medido. Al finalizar el Sprint se llevará a cabo la revisión del Sprint completado para averiguar, en las circunstancias cuando los resultados son contrastables con lo que se ha planificado si hay que realizar un ajuste o no para los próximos Sprints. Este proceso permitirá mantener una mejora continua en el desarrollo del prototipo [34, 35, 36, 37, 38, 39].

4.3. Inmersive Experiencie Questionare(IEQ)

El Immersive Experience Questionnaire (IEQ) es un instrumento que se utiliza como una escala con la que se puede medir el nivel de inmersión subjetiva que vive un jugador a lo largo de una sesión de juego. Su elaboración realizada por Jennett et al. (2008)[40], presenta una respuesta a una necesidad de cuantificar aquellos factores psicológicos que inciden en la inmersión, uniendo conceptos de flujo, absorción cognitiva, y presencia descritos en la sección 3.1. El cuestionario evalúa seis dimensiones: atención, disociación temporal, transporte, desafío, implicación emocional y disfrute. Cada escala se centra en un aspecto distinto de la inmersión que el jugador establece tanto mental como emocionalmente con el mismo videojuego. Su uso es una escala tipo Likert de 31 ítems en donde los sujetos valoran unas afirmaciones de sus experiencias de juego (por ejemplo, "Perdí el sentido del tiempo mientras jugaba.º "Sentí como si estuviera dentro del propio mundo del videojuego"). En el contexto

de esta investigación, el IEQ se utilizará como instrumento principal para comparar el nivel de inmersión entre la versión del prototipo con control convencional (mouse y teclado) y la versión con interacción por voz (Speak to Action). El cuestionario será aplicado a cada participante al finalizar cada una de las dos sesiones de juego (una por condición), siguiendo el diseño intra-sujetos contrabalanceado descrito en la Etapa 4 de este marco metodológico (sección 4.4.4).

4.4. Etapas del Proyecto

4.4.1. Etapa 1: Planificación

La primera etapa del proyecto consiste en la planificación inicial, cuyo propósito es definir con claridad los objetivos, alcances y requerimientos fundamentales del problema y cómo resolverlo. Esta fase establece las bases conceptuales y técnicas necesarias para orientar el desarrollo posterior. Esta etapa corresponde a la recopilación y organización de todas las funcionalidades requeridas del prototipo y del proceso experimental. Esto incluye tareas como: revisión de literatura, análisis comparativo de motores ASR, especificación del entorno de pruebas y elaboración del Immersive Experience Questionnaire (IEQ) como instrumento de medición de inmersión. También se dejó definida la planificación del prototipo mediante el Game Document Design(GDD), junto a la planificación de la estructura de trabajo que se va a seguir basada en la metodología scrum adaptada y definida mediante el uso de la carta gantt.

4.4.2. Etapa 2: Desarrollo e Implementación

Durante esta etapa se establece la forma en la cual se abordara el problema planteado, mediante la definición del propio prototipo, sus mecánicas base y el cómo se abordará la implementación de las mecánicas alternativas de Speak to Action que se buscan poner a prueba, todo esto mediante el Game Document Design(GDD), donde se aborda en mayor detalle el diseño del prototipo.

Además durante esta etapa se establece la definición de la estructura general de los Sprints semanales, definición de objetivos preliminares para cada uno, los cri-

terios de avance y los resultados esperados de cada sprint. Con ello se garantiza una ruta de trabajo clara y adaptable, permitiendo ajustar el desarrollo según los resultados experimentales, las pruebas técnicas de ASR y el progreso real del prototipo.

De esta forma en esta etapa se tiene el desarrollo del prototipo, basado en la metodología scrum adaptada para su uso individual, junto al desarrollo del prototipo y siguiendo el desarrollo gracias a los sprints.

4.4.3. Etapa 3: Implementación y Ajuste del Motor ASR

En esta etapa se implementó dentro de Unity 3D el motor Vosk, seleccionado previamente en base al análisis comparativo realizado en la etapa de Planificación (sección 4.4.1), evaluando su rendimiento técnico dentro del prototipo y realizando los ajustes necesarios para su correcta integración y funcionamiento, dependiendo no solo de su rendimiento técnico sino también de su accesibilidad y modularidad para adaptarlo a las necesidades específicas del prototipo. El periodo de experimentación contempla que esta etapa se desarrollará después de terminado el desarrollo del prototipo, dejando espacio para centrarse por completo en esta etapa sin necesidad de frenar o interrumpir el desarrollo del prototipo base.

4.4.4. Etapa 4: Experimentación

Esta etapa corresponde al proceso de evaluación empírica con usuarios, cuyo propósito es medir el nivel de inmersión generado por las dos versiones del prototipo: la versión tradicional de mouse y teclado y la versión con la mecánica de control por voz Speak to Action. Para ello, se utilizará el Immersive Experience Questionnaire (IEQ), instrumento previamente seleccionado como método de medición subjetiva de inmersión.

Los participantes serán evaluados bajo condiciones experimentales controladas, siguiendo el diseño intra-sujetos definido en la metodología. Cada sujeto probará ambas modalidades del prototipo en un orden contrabalanceado para evitar sesgos derivados del efecto de orden o familiarización. Tras completar cada sesión de juego, los participantes responderán el cuestionario IEQ, permitiendo obtener un registro detallado de su percepción de inmersión en cada condición.

El objetivo central de esta fase es recopilar datos comparativos entre las dos versiones del prototipo. La información obtenida permitirá identificar diferencias en la experiencia del jugador, especialmente en términos de inmersión subjetiva, fluidez de interacción y sensaciones asociadas al control mediante comandos de voz. Estos resultados servirán como base para el análisis cuantitativo posterior, orientado a determinar si la mecánica Speak to Action tiene un impacto significativo sobre la inmersión en juegos del género FPS.

4.4.5. Etapa 5: Análisis y Resultados

En esta etapa se realizará el análisis de todos los datos obtenidos durante la fase de experimentación. Una vez recopilados los cuestionarios IEQ completados por los participantes en las condiciones del control tradicional y control por voz, se procederá a organizar y procesar la información de acuerdo con las dimensiones evaluadas por el instrumento: atención, disociación temporal, transporte, desafío, implicación emocional y disfrute.

Los datos serán limpiados, verificados y posteriormente tabulados para cada participante, separando los puntajes obtenidos en la condición de control tradicional y la condición con la mecánica Speak to Action. A partir de esta organización, se calcularán los puntajes totales del IEQ, así como los promedios, medianas y desviaciones estándar para cada dimensión, permitiendo caracterizar el comportamiento general de ambas modalidades.

Posteriormente, se aplicará la prueba t de Student para muestras relacionadas, adecuada para el diseño intra-sujetos utilizado en esta investigación. Esto permitirá determinar si existen diferencias significativas en los niveles de inmersión entre ambas versiones del prototipo. Además, se analizarán patrones relevantes en las subdimensiones para identificar qué aspectos específicos de la experiencia se ven más influenciados por el uso de comandos de voz.

Finalmente, se procederá a interpretar los resultados que se hayan conseguido, tanto con carácter cuantitativo como cualitativo, enlazando con la literatura revisada

y los objetivos iniciales formulados en la investigación. Esta sección finaliza con la redacción de las conclusiones preliminares y la discusión de los resultados, así como la enumeración de las limitaciones del estudio y las oportunidades de trabajos futuros.

5. Capítulo 5: Diseño

5.1. Etapa 1: Planificación

Durante esta etapa se investigó artículos científicos e información relevante relacionada al uso de la mecánica de interacción por voz como medio de inmersión en juegos[3, 4, 19], en dichas investigaciones también se mencionaba su potencial como control alternativo y su forma de ser un control más accesible para algunos usuarios, sumado a eso se encontraron también investigaciones que resaltaban aspectos técnicos en cuanto a implementación y uso en diferentes ámbitos[5, 21, 14, 15, 16, 17, 20], también se implementó en juegos del género FPS, pero no se exploró su apartado de inmersión, por ende para cumplir con la necesidad de medir el nivel de inmersión de esta mecánica de interacción por voz en juegos del género First Person Shooter, se definió la idea de hacer un prototipo del genero Shooter FPS en unity 3D.

Como parte de esta planificación, se evaluaron comparativamente distintos motores de reconocimiento de voz (ASR) candidatos para su implementación dentro de Unity, análisis detallado en la sección 3.2 del Marco Teórico. En base a dicho análisis, se decidirá implementar y probar dentro de Unity 3D como opción principal al motor Vosk debido a sus compatibilidades con los requerimientos del prototipo, como su carácter gratuito, de código abierto, por ende adaptable y optimizable al proyecto y su compatibilidad nativa con Unity, dejando en segunda opción a Recognissimo, también nativo, solo si es que el tiempo y los recursos lo permitían. Las otras 2 opciones, Undertone y Google Cloud Speech to Text, se dejarán de lado debido a su carácter de pago y a sus enfoques diferentes a los requerimientos del prototipo, pues Undertone se enfoca en la transcripción en tiempo real de idiomas mediante la

voz dentro de Unity, un enfoque más comunicativo cuya adaptabilidad de código a los requerimientos del prototipo no estaba garantizada. En cuanto a Google Cloud Speech to Text, exige licencia de API de paga, necesita de internet para funcionar y demanda de muchos pasos de integración dentro de Unity, sin certeza sobre la posibilidad de modificar su código.

Otro aspecto que se planificó fue la estructura de trabajo mediante el uso de la metodología scrum adaptada, de la cual se usará la planificación de trabajo mediante Sprints.

Las tareas a realizar de esta investigación se definirán mediante el uso de una carta Gantt para la planificación de trabajo:
Anexo a carta gantt: A.1.

Esta Carta Gantt considera tareas ya realizadas, como fue el proceso de investigación de lecturas relacionadas al tema, los motores ASR y la planificación y creación del GDD, así como también se consideraron los meses donde se trabajaran las etapas del sprint definidas de forma muy general, pero sujetas a división de tareas y subtareas durante su periodo de trabajo.

Con respecto al prototipo, se planificó un Shooter del género FPS para ser desarrollado en el motor de videojuegos de Unity 3D, para la idea base se tuvo un videojuego ambientado en el futuro donde un robot debe defender un edificio de un ataque de robots liderados por un sujeto extraño, el juego de base está planeado para tener 2 versiones, una tradicional con teclado y ratón y otra mediante los controles por voz, estos últimos implementados mediante comandos para acciones específicas.

Estos comandos no incluyen ni el moverse ni el apuntar, pues dentro de la jugabilidad el moverse y apuntar son aspectos que se estarán ejecutando constantemente, mientras que disparar, cubrirse y demás acciones son opcionales dependiendo de la situación, bajo este ámbito las acciones de disparar, usar diferentes habilidades, cambiar de arma y recargar si se consideran para su uso mediante el control de voz, pues son mecánicas fundamentales del género, pero dependientes del contexto, además de que no hay una forma orgánica(al menos que exista o se haya investigado aún)

de moverse, apuntar y disparar usando la voz ejecutándose en simultáneo, pues son acciones que se hacen casi al mismo tiempo y la implementación del control de voz dentro del movimiento y apuntado, entorpecería innecesariamente una acción que debería ser simple de ejecutar, a la vez de interrumpir el flujo orgánico secuencial de ejecutar una acción detrás de otra o ambas al mismo tiempo(moverse, apuntar y disparar).

Teniendo ese diseño de los comandos de voz en cuenta se creó el GDD donde se abarca más en detalle la idea y diseño general del juego. El Game Design Document (GDD) completo puede consultarse en el anexo: A.2.

5.1.1. Diseño Experimental

Se harán experimentos usando el IEQ Cuestionnare, con sujetos de manera presencial, Intra-sujetos (within subjects/contrabalanceado), lo que quiere decir que se usarán a sujetos de prueba que probaran las dos versiones del juego, la versión FPS tradicional de mouse y teclado y la versión FPS con la mecánica de interacción por voz Speak To Action ya implementada, se espera que con la prueba de ambas versiones por parte de cada sujeto se obtenga una comparación directa entre ambas.

5.1.2. Condiciones del experimento

Para la evaluación del impacto de la mecánica Speak to Action sobre la inmersión en juegos del género First Person Shooter (FPS), el experimento se dividirá en dos condiciones bien diferenciadas, cada una representando un modo alternativo de interacción. Estas condiciones permiten una comparación controlada acerca de la variación del jugador cuando este utiliza controles tradicionales frente a comandos de voz.

5.1.3. Condición 1: Control Tradicional: Mouse y Teclado

En esta condición, los participantes jugarán una versión del prototipo utilizando únicamente los controles estándar de los juegos FPS. Esto permitirá establecer una

línea base sobre la cual se comparará el efecto de la interacción por comandos de voz.

Esta condición se medirá mediante las siguientes características: Movimiento mediante las teclas WASD, mira y precisión dependientes del mouse, las acciones principales de recarga, disparo y demas se realizan mediante teclas especificadas, no existe componente de control por voz y en términos generales es un esquema de juego tradicional en el género de los FPS. De esta forma se medirá el nivel de inmersión que produce esta versión del prototipo, estableciendo un punto de referencia para su contraste con la versión con la mecánica Speak to Action ya implementada.

5.1.4. Condición 2: Speak to Action: Controles mediante voz

En esta condición, el jugador utilizará comandos de voz para realizar las acciones principales dentro del prototipo. Esta versión integrará un motor ASR, el cual interpretará órdenes vocales y las traduce en acciones dentro del juego.

Esta condición se medirá mediante las siguientes características: el movimiento básico puede mantenerse con el teclado y así evitar un movimiento tosco e innecesariamente rebuscado, sin embargo acciones clave como recargar, cambiar arma, entre varias más, se ejecutarán mediante comandos de voz, el sistema ASR reconocerá palabras clave o frases previamente definidas, la interacción depende de la precisión del reconocimiento de voz como de la claridad del jugador. Esto buscará evaluar la inmersión generada por un modo de control más expresivo, dinámico y natural, en contraste con las entradas físicas más estándares y así determinar si la mecánica Speak to Action incrementa la inmersión subjetiva, la sensación de agencia del jugador y la conexión con le juego, en contraste con los controles tradicionales.

5.1.5. Participantes

El experimento se llevará a cabo con una muestra estimada de 10 a 20 participantes, un número más acorde para estudios exploratorios de interacción en videojuegos y suficiente para obtener variabilidad estadística en los puntajes del IEQ.

5.1.6. Criterios de evaluación

Personas entre 18 y 30 años, Experiencia básica o intermedia con videojuegos del género FPS, capacidad de comunicación verbal con claridad en el idioma utilizado dentro del prototipo, familiaridad mínima con el uso de mouse y teclado.

5.1.7. Criterios de Exclusión

Dificultades severas del habla, volumen o articulación que impidan emitir comandos básicos, discapacidad auditiva o problemas que afecten directamente la percepción del sonido del juego, participantes con epilepsia fotosensible o condiciones incompatibles con videojuegos en primera persona.

5.1.8. Instrumentos del experimento

El instrumento principal para la medición de la inmersión será el Immersive Experience Questionnaire (IEQ). Si bien su definición completa se presenta en el Marco Teórico, en este apartado se especifica su uso dentro del experimento.

5.1.9. Uso del IEQ dentro del estudio

Tras completar cada una de las dos condiciones(control tradicional y por voz), el participante deberá responder el cuestionario IEQ. El cuestionario consiste en 31 ítems con una escala de likert que evalúa seis dimensiones: atención, disociación temporal, transporte, desafío, implicación emocional y disfrute. Los puntajes obtenidos en cada condición serán comparados estadísticamente, el IEQ permitirá cuantificar el nivel de inmersión subjetiva y analizar diferencias entre las dos modalidades de control.

5.1.10. Procedimiento

El experimento se realizará de manera individual, siguiendo un diseño intrasujetos donde cada participante experimentará ambas condiciones del prototipo. Para evitar efectos de orden, se utilizará contrabalanceo.

5.1.11. Secuencia del procedimiento

1. Bienvenida y Consentimiento informado.
Se explicará el propósito general del estudio.

2. Asignación aleatoria del orden experimental.
Grupo 1: Condición 1 - Condición 2.
Grupo 2: Condición 2 - Condición 1.

3. Primera fase de juego.
El participante juega la primera condición durante 8–14 minutos, tiempo suficiente para familiarizarse y realizar las acciones clave.

4. Aplicación del IEQ(Fase 1).
El participante completa el cuestionario evaluando su inmersión en la primera condición.

5. Pausa breve de 2-3 minutos.
Esto reduce levemente el arrastre emocional y mejora la comparabilidad.

6. Segunda fase del juego.
El participante juega la segunda condición bajo las mismas reglas y duración que la primera fase.

7. Aplicación del IEQ(Fase 2).
Se completa el cuestionario nuevamente, evaluando la segunda modalidad.

8. Comentarios finales.

9. Cierre del Experimento.

5.1.12. Métricas a evaluar

El estudio se centrará en métricas subjetivas obtenidas a partir del IEQ, las cuales permiten cuantificar el grado de inmersión experimentado por cada participante

en ambas condiciones.

Métricas Principales: Puntajes total del IEQ.

Metricas por Subdimensiones: Se analizaran también los valores individuales de cada dimensión del IEQ, como la atención, disociación temporal, transporte, desafío, implicación emocional, disfrute.

Estas métricas permiten identificar qué aspectos de la inmersión mejoran o no cuando el jugador utiliza comandos de voz para la interacción con el juego.

5.2. Etapa 2: Desarrollo e Implementación

En esta etapa se presenta el cómo se abordará la estructura de trabajo de sprints para el desarrollo del prototipo funcional del género FPS que implementa el sistema tradicional de juego mediante teclado y ratón Cabe destacar que se consideran solo los sprints referentes al prototipo base para esta sección.

5.2.1. Product Backlog

Comenzando con la definición del Product Backlog, que definirá las tareas base que después se abordarán en más detalle.

Product Backlog
Mecánicas Base del jugador.
Sistema de Armas.
Sistema de Combate.
Sistema de Enemigos.
Audio.
Sistema de Habilidades.
Jefe Final.
HUD.
Nivel Prototipo.

Tabla 5.1: Product Backlog

Lo siguiente es la definición de la duración de cada elemento del product backlog y de su definición como sprint.

Sprint	**Épicas**	**Duración**
1	Mecánicas Base del jugador.	2 Semanas.
2	Sistema de Armas.	2 Semanas.
3	Sistema de Combate.	2 Semanas.
4	Sistema de Enemigos.	2 Semanas.
5	Audio.	1 Semana.
6	Sistema de Habilidades.	2 Semanas.
7	Jefe Final.	1 Semana.
8	HUD.	1 Semana.
9	Nivel Prototipo.	2 Semanas.

Tabla 5.2: Detalles de cada uno de los Sprints.

5.2.2. Sprint Backlog

A continuación se detalla cada sprint con sus respectivas tareas a realizar para el desarrollo del prototipo.

Categoría	**Épicas**
Mecánicas Base del jugador.	▪ Implementación del control del Movimiento. ▪ Implementación de salto. ▪ Implementación de agacharse. ▪ Implementación de correr.
Sistema de Armas.	▪ Implementación de mecánica de disparo base. ▪ Implementación de sistema de armas.

Categoría	**Épicas**
Nivel Prototipo.	▪ Diseño general de las áreas del nivel. ▪ Modelado general de las áreas del nivel. ▪ Implementación del nivel dentro del motor gráfico.
Sistema de Combate.	▪ Implementación de Sistema de Vida y Habilidades para el jugador. ▪ Implementación de Sistema de Ataque de los enemigos (ataques cuerpo a cuerpo y a distancia). ▪ Implementación de Sistema de Vida para los enemigos. ▪ Implementación del Sistema de Daño para el jugador y enemigos. ▪ Implementación del Sistema de Muerte para el jugador y enemigos. ▪ Implementación de Comportamiento de Patrulla para los enemigos. ▪ Implementación de Sistema de detección del jugador.
Sistema de Habilidades.	▪ Implementación de la Habilidad Slowy (ralentización del tiempo). ▪ Implementación de la Habilidad Counter (Parry). ▪ Implementación de la Habilidad Royal (Escudo).

Categoría	**Épicas**
Sistema de Enemigos.	▪ Implementación de enemigo de disparo (Robot de Asalto normal). ▪ Implementación de enemigo de disparo especial (Robot de asalto azul que otorga una habilidad). ▪ Implementación de enemigo cuerpo a cuerpo (Robot Reprogramado). ▪ Implementación de enemigo cuerpo a cuerpo especial (Robot Reprogramado Verde que otorga una habilidad).
Jefe Final.	▪ Implementación de movimiento de transporte lateral. ▪ Implementación de movimiento de Esquivar balas del jugador. ▪ Implementación de Ataques Cuerpo a Cuerpo. ▪ Implementación de Ataque a distancia con pistola.
HUD.	▪ Implementación de interfaz HUD de vida, armadura, habilidad, arma y cantidad de munición restante.

Categoría	**Épicas**
Audio.	▪ Implementación de Audio para el movimiento del jugador (moverse, saltar, correr). ▪ Implementación de Audio para cada una de las armas. ▪ Implementación de Audio (tema de fondo) para el nivel. ▪ Implementación de Audio (tema de fondo) para el combate. ▪ Implementación de audio para enemigos. ▪ Implementación de Tema de fondo para la batalla contra el jefe Final. ▪ Implementación de Audios para el jefe. ▪ Implementación de Audios para las Habilidades (activación y desactivación).

Tabla 5.3: Sprint backlog

6. Capítulo 6: Implementación

Se comenzó con una versión base del prototipo, con las principales mecánicas base implementadas, como mecánicas base, sistema de armas, sistema de combate, audio principalmente.

Posterior se empezó de inmediato la implementación del software de voz ASR "Vosk", el cual es gratuito y compatible con unity, Su implementación consto de descargar una carpeta con los archivos necesarios, colocarlos dentro de unity y a partir de los códigos que hacían funcionar el motor hacer un código central enfocado a ejecutar algunas de las acciones del juego con la voz, este código se colocó dentro de un objeto vacío en la escena, cumpliendo el rol de un objeto de tipo "manager" y sirvió como prueba de su funcionamiento. Logrando hacerlo funcionar pero a una velocidad para nada optima, en ese entonces tomaba solo un comando "bang" para el disparo, pero su velocidad de disparo era con un cooldown de 1 segundo, como se muestra en el commit de la Figura 6.1.

Con el Motor ASR funcionando solo quedaba darle forma al prototipo, implementando sistema de combate, sistema de enemigos, variedad de armas, sistema de habilidades, Audio. Contando con una versión funcional del prototipo, pero sin tener un mapa o un objetivo claro, tal como se observa en la Figura 6.2.

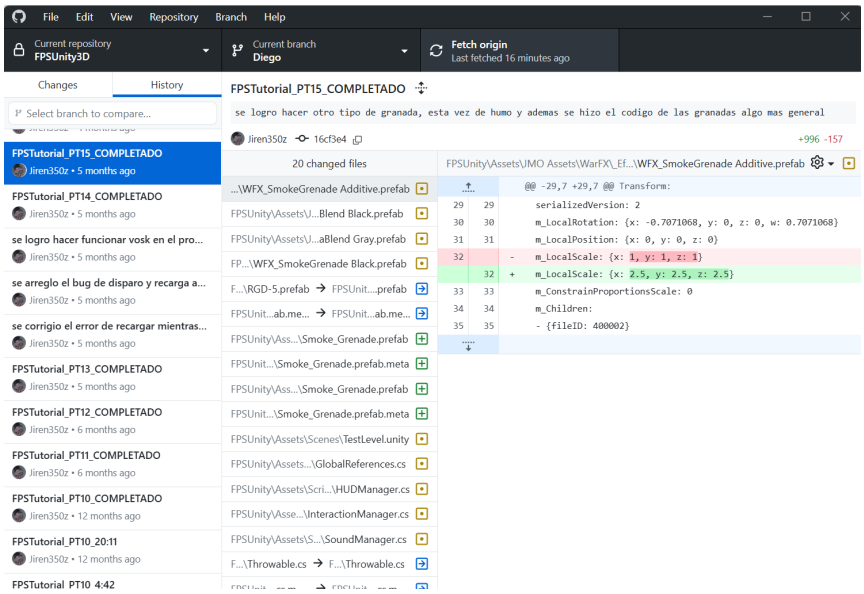


Figura 6.1: Primeros commits del prototipo.

Figura 6.2: Versión funcional del prototipo.

Posteriormente se trabajó en el nivel prototipo, planeado como si fuera un prólogo de corta duración, tomando bastante tiempo pero logrando completarlo, además de agregar assets faltantes y trabajar el HUD dejando listo el nivel prototipo y su respectivo UI, posterior a eso se agregó un sistema simple de autoguardado para reiniciar desde el punto de control y un tutorial básico para iniciar, como se muestra

en las Figuras 6.3–6.4.

Figura 6.3: Implementación inicial del UI y tutorial.

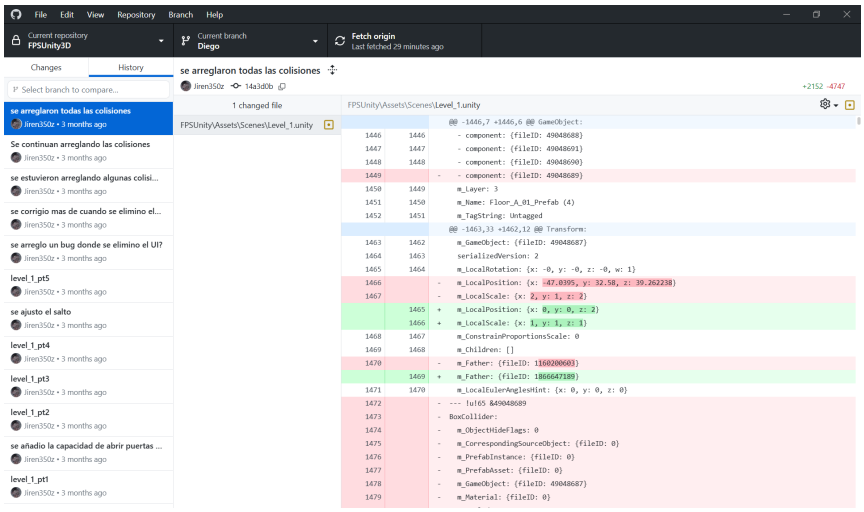


Figura 6.4: Commits de trabajo del nivel prototipo.

Se trabajó en optimizar Vosk, aspecto que fue relativamente asequible en poco tiempo, en comparación al trabajo en el resto del prototipo, logrando bajar el cooldown de 1 segundo a 0.03 segundos, siendo un tiempo óptimo y estable para las pruebas de usuario. Aunque cabe mencionar que al principio se fue probando de a poco pues primero fueron 0.5, luego 0.1, pero al final se optó por dejar 0.03 por temas

de estabilidad, pues para ese punto se tiene un código más extenso y modular que servía para ejecutar varias acciones del gameplay mediante la voz, por ende solo bastaba con bajarle el valor a la variable çooldown seconds"que se encarga del tiempo de espera entre un comando y otro, sin embargo bajar el cooldown también implicaba modificar el cooldown de la metralleta la cual funcionaba con 2 modos, uno para un solo disparo("guts") y otro para el modo rafaga("attack"), el cual necesitaba pasos extra como mantener la palabra para funcionar mejor, lo que hacía que ese comando y ese modo de disparo tuviera su propio código ligado al cooldown seconds, por lo cual si se bajaba el cooldown sin cuidado podría afectar negativamente la estabilidad de esa arma, como se observa en la Figura 6.5.

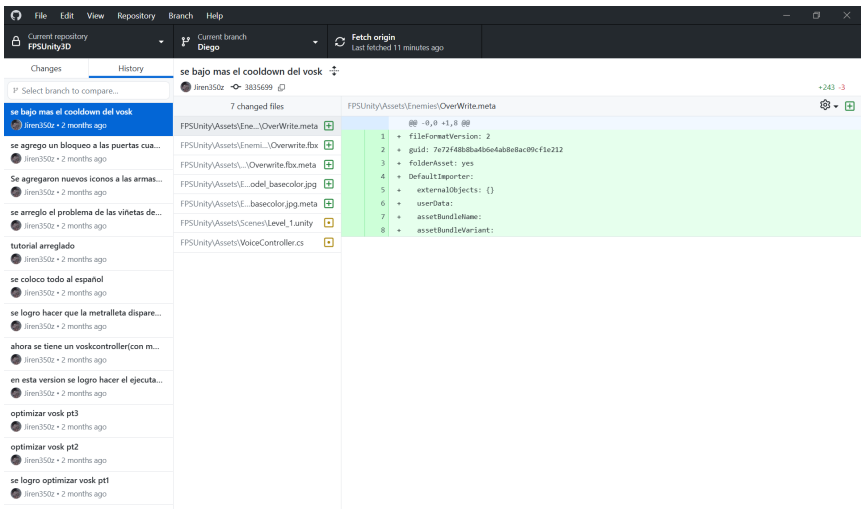


Figura 6.5: Optimización de Vosk.

Finalmente a partir de la versión final se hicieron 2 versiones, una exclusivamente con comandos de voz para acciones principales y una versión teclado y ratón, para así probar ambas versiones contrabalanceadas en las pruebas de usuario, tal como se observa en los commits de las Figuras 6.6–6.7.

Figura 6.6: Versión final del prototipo, con ambas versiones funcionando, teclado ratón y voz.

Para ver la versión final del prototipo en funcionamiento, se puede acceder al Anexo A.8.1,"Gameplay de la Versión Final del Prototipo".

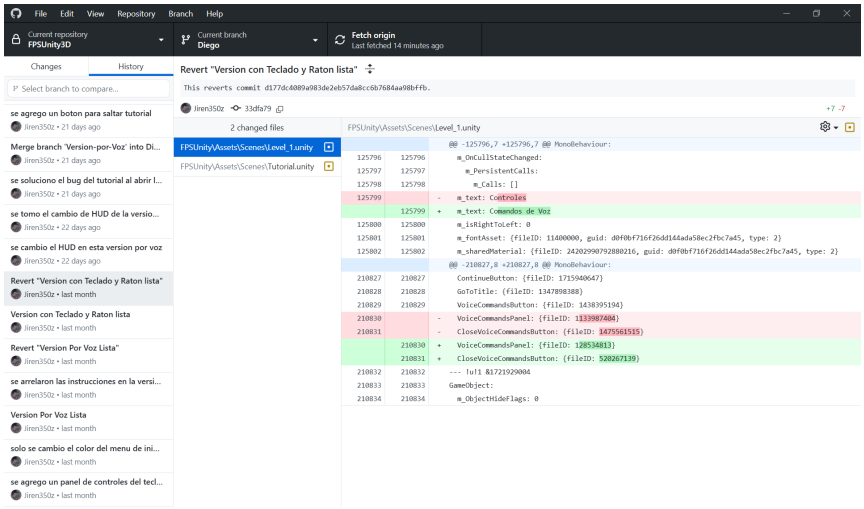


Figura 6.7: Commits finales.

Para revisar en detalle la implementación de los códigos principales que componen el sistema de interacción por voz, se puede consultar el Anexo A.8.2, "Códigos Principales del Sistema de Interacción por Voz".

7. Capítulo 7: Resultados

7.1. Análisis descriptivo

Antes de aplicar la prueba de inferencia estadística, se presenta un análisis descriptivo de los resultados del cuestionario IEQ siguiendo el enfoque original de Jennett et al.[40], quienes agrupan los 31 ítems del instrumento en 6 dimensiones: atención, disociación temporal, transporte, desafío, implicación emocional y disfrute. Debido a problemas en la recolección de datos, 2 de los 31 ítems originales no pudieron registrarse correctamente en la versión aplicada a la condición de voz, por lo que se decidió excluir ambos ítems del análisis en las dos condiciones, con el fin de asegurar la equivalencia matemática entre versiones y no sesgar la comparación. En consecuencia, el análisis se realiza sobre los 29 ítems comunes entre ambas versiones, cuyas distribuciones completas de respuesta por pregunta (frecuencias 1-5) se incluyen en el Anexo A.10.

La Figura 7.1 resume la media obtenida en cada subdimensión del IEQ para ambas condiciones. Cinco de las seis subdimensiones mostraron una diferencia estadísticamente significativa a favor de la condición de voz: Atención ($M_{KB} = 3{,}59$, $M_{Voz} = 4{,}31$; $p < 0{,}001$), Desafío ($M_{KB} = 3{,}59$, $M_{Voz} = 3{,}89$; $p = 0{,}002$), Implicación Emocional ($M_{KB} = 3{,}33$, $M_{Voz} = 3{,}68$; $p = 0{,}007$), Disociación Temporal ($M_{KB} = 2{,}97$, $M_{Voz} = 3{,}44$; $p = 0{,}045$) y Transporte ($M_{KB} = 3{,}62$, $M_{Voz} = 3{,}96$; $p = 0{,}046$). La mayor diferencia se observó en Atención, con un tamaño de efecto grande ($d = 1{,}13$), sugiriendo que la necesidad de emitir comandos de voz demanda un involucramiento activo y sostenido por parte del jugador, en línea con la dimensión de control y autonomía descrita por Brown y Cairns[23] como componente central

de la inmersión.

En contraste, la subdimensión Disfrute no mostró diferencias significativas entre condiciones ($M_{KB} = 4{,}15$, $M_{Voz} = 4{,}18$; $p = 0{,}789$), siendo además la de mayor puntaje en ambas versiones. Esto sugiere que el prototipo resultó igualmente disfrutable independientemente del modo de control utilizado, y que la ventaja de Speak to Action no radica en generar mayor agrado, sino en profundizar mecanismos específicos de involucramiento cognitivo y emocional durante la experiencia de juego.

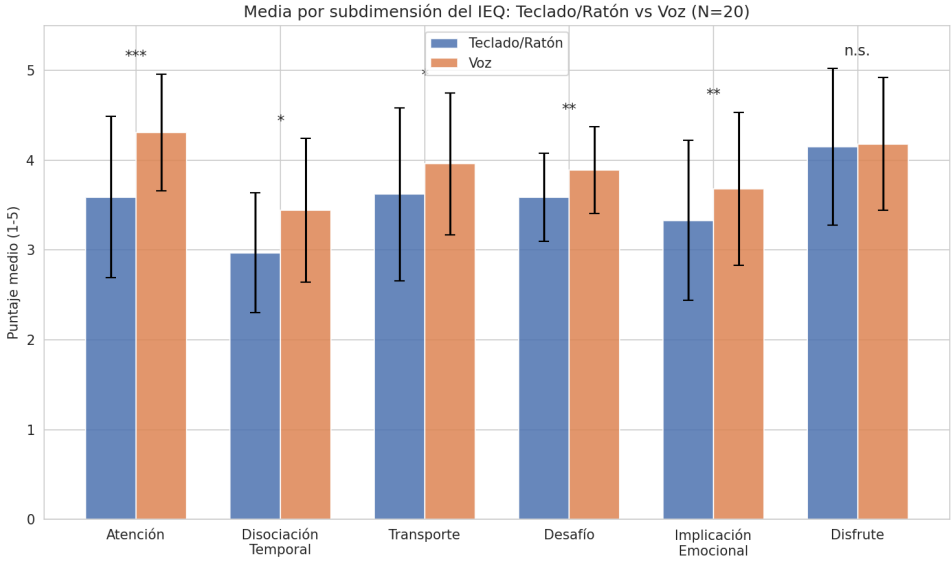
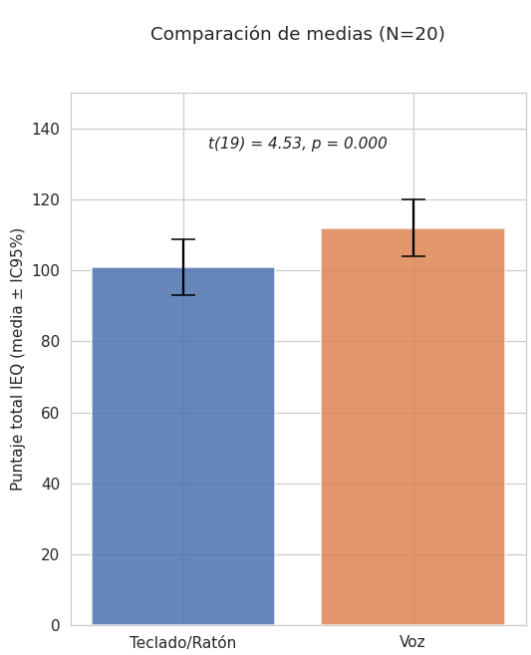


Figura 7.1: Media por subdimensión del cuestionario IEQ, comparando la condición de Teclado/Ratón y Voz (N=20).

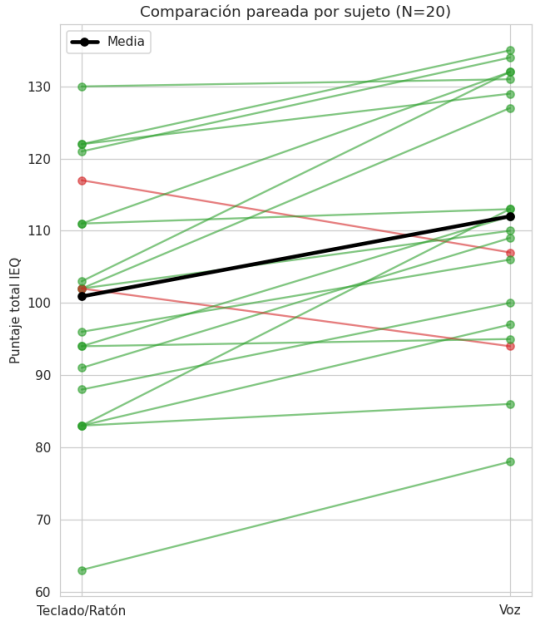
7.2. Análisis inferencial: Prueba t de Student

El análisis inferencial se presenta en dos etapas complementarias. En primer lugar, se analiza la muestra completa disponible (N=20), que corresponde a la totalidad de los participantes que completaron ambas versiones del cuestionario, excluyendo únicamente a dos sujetos con datos incompletos. Esta muestra ofrece la mayor potencia estadística posible dentro del estudio. Sin embargo, dado que el orden de aplicación de las condiciones (voz primero vs. teclado primero) no fue idéntico para todos los participantes, se realiza un segundo análisis sobre un subconjunto contrabalanceado de N=8 (4 sujetos por cada orden de aplicación). Este análisis permite verificar que

la diferencia observada en la muestra completa no se deba a un efecto de orden o aprendizaje es decir, que los resultados no estén sesgados por la posición en que cada participante experimentó cada condición, fortaleciendo así la validez interna de los hallazgos obtenidos en la muestra completa.



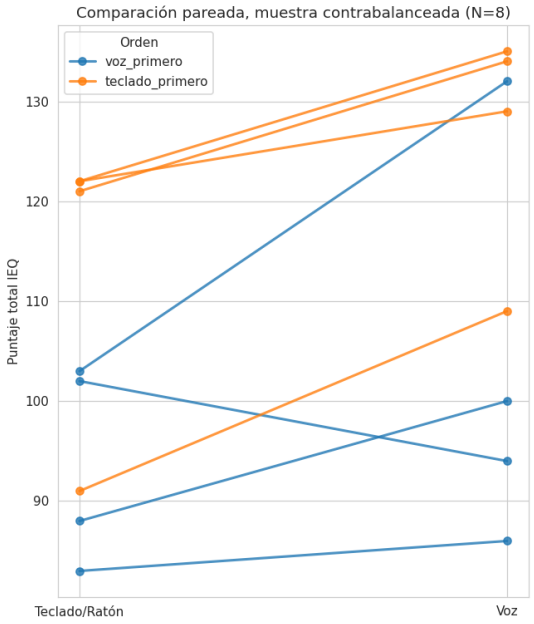
(a) Comparación de medias (N=20)



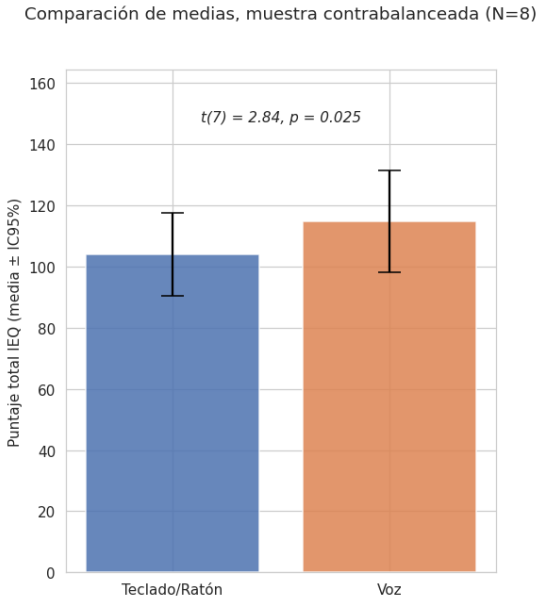
(b) Comparación pareada por sujeto (N=20)

Figura 7.2: Resultados de la prueba t de Student para muestras relacionadas, muestra completa (N=20).

La puntuación total media del IEQ, calculada según el esquema de recodificación de Jennett et al.[40], fue de 100,90 (DE=16,85) en la condición teclado/ratón y de 112,00 (DE=17,13) en la condición de voz (Figura 7.2a). La prueba t de Student para muestras relacionadas mostró una diferencia estadísticamente significativa entre ambas condiciones, t(19)=4,53t(19)=4,53 t(19)=4,53, p¡0,001p¡0,001 p¡0,001, con un tamaño de efecto grande (d=1,01d=1,01 d=1,01). El gráfico de líneas pareadas (Figura 7.2b) confirma que esta tendencia es consistente a nivel individual: 17 de los 20 participantes (85 %) reportaron mayor inmersión con la versión de voz, frente a solo 3 que reportaron lo contrario, con una diferencia media de 11,10 puntos.



(a) Comparación pareada, muestra contrabalanceada (N=8)



(b) Comparación de medias, muestra contrabalanceada (N=8)

Figura 7.3: Resultados de la prueba t de Student para muestras relacionadas, subconjunto contrabalanceado (N=8).

En el subconjunto contrabalanceado (Figura 7.3, n=8, 4 sujetos por orden de aplicación), la media en la condición teclado/ratón fue de 104,00 (DE=16,07) y en voz de 114,88 (DE=19,96), con una diferencia media de 10,88 puntos, prácticamente idéntica en magnitud a la observada en la muestra completa. La prueba t confirmó una diferencia estadísticamente significativa, t(7)=2,84t(7)=2,84 t(7)=2,84, p=0,025p=0,025 p=0,025, con un tamaño de efecto igualmente grande (d=1,00d=1,00 d=1,00). El gráfico de líneas pareadas por orden de aplicación (Figura 7.3a) no evidencia un patrón sistemático donde un orden particular favorezca artificialmente los resultados: tanto los participantes que probaron voz primero como los que probaron teclado primero mostraron incrementos en su puntaje de inmersión al usar Speak to Action. Esta consistencia entre ambos análisis —muestra completa y subconjunto controlado por orden— refuerza la validez del hallazgo principal: la mecánica Speak to Action produjo un aumento significativo y de magnitud considerable en la inmersión subjetiva reportada por los jugadores. Para mas informacion sobre los datos revisar la tablaA.3

8. Capítulo 8: Conclusión

El objetivo general de esta investigación fue implementar la mecánica de interacción por voz Speak to Action en un videojuego FPS para analizar su impacto en la inmersión del jugador. A partir del desarrollo del prototipo y del análisis experimental realizado, se considera que este objetivo fue cumplido satisfactoriamente. Respecto a los objetivos específicos, se evaluó comparativamente el motor de reconocimiento de voz Vosk, seleccionado por su viabilidad de integración en Unity. se diseñó e implementó un prototipo funcional que integra la mecánica Speak to Action junto a su equivalente en control convencional, permitiendo una comparación directa entre ambas modalidades. Asimismo, Se realizaron pruebas experimentales con usuarios bajo un diseño intra-sujetos contrabalanceado, aplicando el cuestionario IEQ para medir la inmersión subjetiva en ambas condiciones. Finalmente, se analizaron los resultados obtenidos mediante la prueba t de Student para muestras relacionadas, lo que permitió comprobar y verificar la hipótesis planteada.

En cuanto a la hipótesis de investigación que los jugadores que utilicen la versión con la mecánica Speak to Action reportarán un mayor nivel de inmersión subjetiva que en la versión con control convencional los resultados obtenidos permiten aceptarla. Tanto en la muestra completa (N=20) como en el subconjunto contrabalanceado (N=8), la puntuación total de inmersión fue significativamente mayor en la condición de voz que en la condición de teclado/ratón: $t(19) = 4{,}53$, $p < 0{,}001$, $d = 1{,}01$ para la muestra completa, y $t(7) = 2{,}84$, $p = 0{,}025$, $d = 1{,}00$ para el subconjunto contrabalanceado. La consistencia de estos resultados entre una muestra completa y un subconjunto controlado específicamente por orden de aplicación descarta que la diferencia observada se deba a un efecto de orden o de aprendizaje, fortaleciendo la

validez del hallazgo. El tamaño de efecto grande obtenido en ambos análisis ($d \approx 1{,}0$) indica además que la magnitud de esta diferencia no es meramente marginal, sino sustancial.

Estos resultados responden directamente a la pregunta de investigación planteada: la implementación de la mecánica Speak to Action en un videojuego FPS sí aumenta el nivel de inmersión del jugador en comparación con el control convencional basado en teclado y mouse, dentro del contexto y las condiciones evaluadas en este estudio. Pese a lo anterior, se observó cierta variabilidad interindividual en la magnitud del efecto: si bien la mayoría de los participantes (17 de 20) reportó mayor inmersión con la voz, un grupo menor mostró la tendencia inversa. Esto sugiere que la percepción de inmersión frente a Speak to Action podría estar moderada por diferencias individuales entre jugadores (por ejemplo, familiaridad previa con comandos de voz o con el género FPS), un aspecto que estudios futuros con muestras de mayor tamaño podrían explorar con mayor profundidad. Una limitación de esta investigación fue el no implementar y probar empíricamente las otras tres alternativas evaluadas: Recognissimo, Undertone y Google Cloud Speech to Text, principalmente debido a su carácter de pago y a las restricciones de presupuesto y tiempo del proyecto. Si bien la comparación teórica realizada permitió fundamentar la elección de Vosk como motor más adecuado para los requerimientos del prototipo, no fue posible contrastar de forma práctica su rendimiento real(precisión de reconocimiento, latencia, robustez frente al ruido) frente a estas alternativas dentro de un contexto de juego FPS. Se recomienda que investigaciones futuras evalúen empíricamente estos motores en condiciones similares, con el fin de determinar si alguno de ellos ofrece un desempeño superior a Vosk para este tipo de aplicaciones.

Como observación adicional derivada de la experimentación, varios participantes presentaron dificultades de pronunciación con los comandos en inglés utilizados en el prototipo, lo que en algunos casos dificultó su desempeño durante las pruebas. Como trabajo futuro evaluar una versión del prototipo con comandos en español, tanto para reducir esta barrera lingüística como para explorar, si un prototipo de este tipo podría emplearse como herramienta de apoyo para el desarrollo de la pronunciación en inglés. Además también se podría incluir pruebas cualitativas para captar impresiones generales de la mecánica.

Bibliografía

[1] Lifeline (video game). https://en.wikipedia.org/wiki/Lifeline_(video_game).

[2] Ubisoft. Tom Clancy's EndWar. https://store.ubisoft.com/us/tom-clancys-endwar/575ffdbda3be1633568b4f20.html.

[3] C. Mattey. Overvoice: Voice controlled games engagement study. Technical report, University of San Francisco, 2018.

[4] N. Zargham, M. A. Friehs, L. Tonini, D. Alexandrovsky, E. G. Ruthven, L. E. Nacke, and R. Malaka. Let's talk games: An expert exploration of speech interaction with npcs. *International Journal of Human–Computer Interaction*, 41(5):3592–3612, 2024.

[5] W. P. A. Setiawan, F. D. C. Putra, A. B. Gumelar, R. A. Romadhonny, M. Rizqi, and R. D. Nugroho. Development of first-person shooting games using human voice command and its potential use for serious game engines. In *Proc. 2019 Int. Seminar on Application for Technology of Information and Communication (iSemantic)*, pages 110–115, 2019.

[6] Bad Dream Games. One Hand Clapping. https://baddreamgames.com/one-hand-clapping/.

[7] Mage arena. https://gam3s.gg/mage-arena/.

[8] Shout & shoot. https://store.steampowered.com/app/3324800/Shout__Shoot/.

[9] Game Developer. Fonix Supplies Voice Command For Tom Clancy's EndWar. https://www.gamedeveloper.com/game-platforms/fonix-supplies-voice-command-for-i-tom-clancy-s-endwar-i-.

[10] Mage Arena Fandom Wiki. Spells. `https://mage-arena.fandom.com/wiki/Spells`.

[11] Press Play Media. Developer Interview: One Hand Clapping. `https://pressplaynews.net/2021/09/22/developer-interview-one-hand-clapping/`, 2021.

[12] 80 Level. Shout & Shoot by Shout Games Is a FPS Where You Use Voices Weapon. `https://80.lv/articles/a-multiplayer-fps-where-you-use-voices-to-see-shoot-enemies`, 2024.

[13] Fraser Allison, Marcus Carter, and Martin Gibbs. A history of voice interaction in games. In *Proceedings of the 1st International Joint Conference of DiGRA and FDG*, pages 1–10. Digital Games Research Association, 2016.

[14] Susumu Harada, Jacob O. Wobbrock, and James A. Landay. Voice games: Investigation into the use of non-speech voice input for making computer games more accessible. In *Proceedings of the IFIP TC13 Conference on Human-Computer Interaction (INTERACT 2011)*, volume 6946 of *Lecture Notes in Computer Science*, pages 11–29. Springer, 2011.

[15] Pontus Nilsson and Wilhelm Öhman. Natural language interfaces in computer games: A study of nli accuracy in risk, May 2016.

[16] Milan Wevelsiep, Nicholas Thomas Walker, Nicolas Wagner, and Stefan Ultes. A voice-controlled dialogue system for npc interaction using large language models. In *Proceedings of the 15th International Workshop on Spoken Dialogue Systems Technology (IWSDS)*, pages 29–38, Bamberg, Germany, 2025. Association for Computational Linguistics.

[17] Andrei Copaceanu, Jonathan Weinel, and Stuart Cunningham. Using voice input to control and interact with a narrative video game. pages 345–347, 2024.

[18] O. Graham and D. Wright. Investigating the use of vosk for speech to text in interactive gaming applications. *Preprints.org*, 2025.

[19] Chra Abdoulqadir and Fernando Loizides. Interaction, artificial intelligence, and motivation in children's speech learning and rehabilitation through digital games: A systematic literature review. *Information*, 16(7):599, 2025.

[20] Dominik Strzałko. Voice controlled games: The approach and challenges of implementing speech recognition and voice control in games. In *Position and Communication Papers of the 16th Conference on Computer Science and Intelligence Systems (FedCSIS)*, volume 26 of *Annals of Computer Science and Information Systems*, page 230. IEEE, 2021.

[21] Revolutionizing ai companion in fps games. `https://openreview.net/pdf/6ad918b24b6e55c04039bc5605161334472ed9d5.pdf`. Preprint under review at ICLR 2025.

[22] Laura Ermi and Frans Mäyrä. Fundamental components of the gameplay experience: Analysing immersion. In Suzanne de Castell and Jennifer Jenson, editors, *Changing Views: Worlds in Play – Selected Papers of the 2005 Digital Games Research Association's Second International Conference*, pages 15–27, Vancouver, 2005. Digital Games Research Association & Simon Fraser University.

[23] Emily Brown and Paul Cairns. A grounded investigation of game immersion. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems*, pages 1297–1300, Vienna, Austria, 2004. ACM.

[24] Lennart E. Nacke and Craig A. Lindley. Flow and immersion in first-person shooters: Measuring the player's gameplay experience. In *Proceedings of the International Academic Conference on the Future of Game Design and Technology (FuturePlay)*, pages 81–88. ACM, 2008.

[25] Ron Tamborini and Paul Skalski. The role of presence in the experience of electronic games. In Peter Vorderer and Jennings Bryant, editors, *Playing Video Games: Motives, Responses, and Consequences*, pages 225–240. Lawrence Erlbaum Associates, 2006.

[26] Rod McCall, Shaleph O'Neill, and Fiona Carroll. Measuring presence in virtual environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–8, Vienna, Austria, 2004. ACM.

[27] Kiran Ijaz, Naseem Ahmadpour, Yifan Wang, and Rafael A. Calvo. Player experience of need satisfaction (pens) in an immersive virtual reality exercise platform describes motivation and enjoyment. *International Journal of Human–Computer Interaction*, 36(13):1195–1204, 2020.

[28] Alpha Cephei Inc. Vosk Speech Recognition Toolkit. https://alphacephei.com/vosk/https://github.com/Yeti47/Vosk4Unity.

[29] Game Content Deals. Speech Recognition using Google Cloud PRO 2025. https://gamecontentdeals.com/assets/tools/speech-recognition-using-google-cloud-pro-2025/.

[30] Unity Technologies. Streaming Speech Recognition using Google Cloud (VR/AR/Mobile). https://assetstore.unity.com/packages/tools/ai-ml-integration/streaming-speech-recognition-using-google-cloud-vr-ar-mobile-des-115172.

[31] Undertone AI. Undertone: Offline Whisper AI Voice Recognition. https://marketplace.unity.com/packages/tools/ai-ml-integration/undertone-offline-whisper-ai-voice-recognition-246877.

[32] Unity Asset Collection. Recognissimo Offline Speech Recognition (Free Download). https://unityassetcollection.com/recognissimo-offline-speech-recognition-free-download.

[33] Recognissimo. Recognissimo Offline Speech Recognition (Unity Asset Store). https://assetstore.unity.com/packages/tools/audio/recognissimo-offline-speech-recognition-203101.

[34] Bartosz Pałka. Scrum in video games development – literature review. *Business and Public Administration Studies*, 17(1):33–55, 2023.

[35] Ken Schwaber and Jeff Sutherland. The scrum guide. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf, 2020.

[36] Zainab Masood, Rashina Hoda, and Kelly Blincoe. Real world scrum: A grounded theory of variations in practice. *IEEE Transactions on Software Engineering*, 48(5):1579–1591, 2022.

[37] Adam Alami and Oliver Krancher. How scrum adds value to achieving software quality? *Empirical Software Engineering*, 27(7):165, 2022.

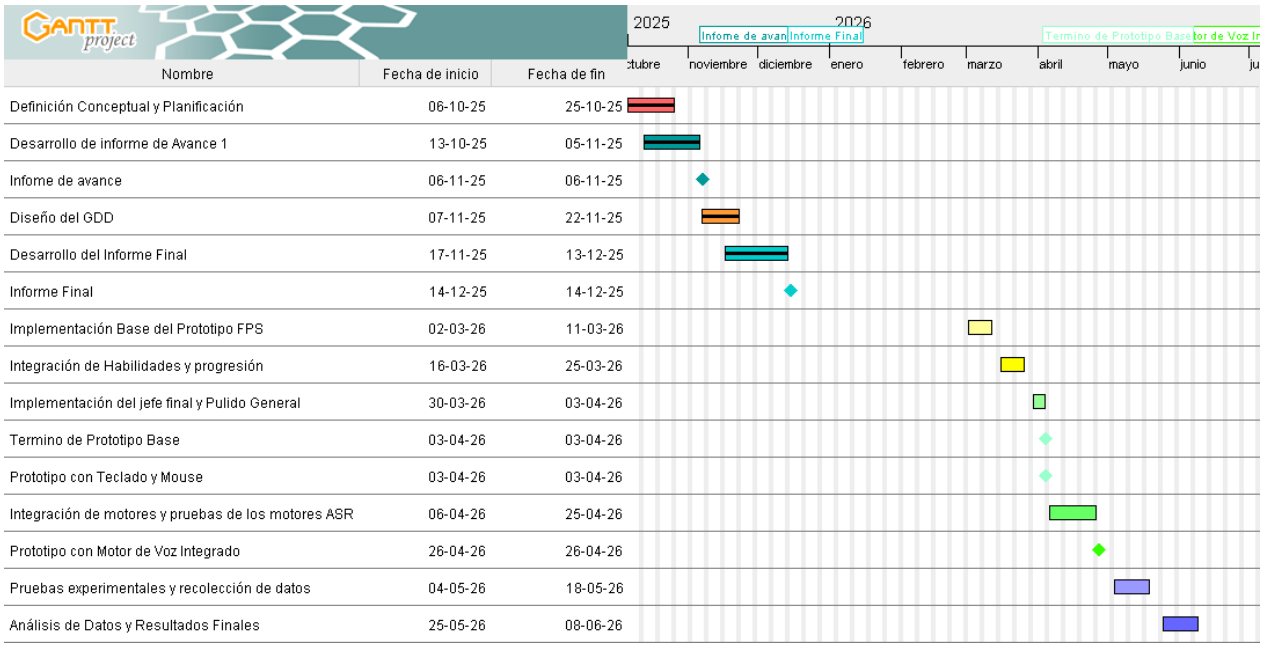
[38] Adrielle Cristina Sassa, Isabela Alves de Almeida, Tábata Nakagomi Fernandes Pereira, and Milena Silva de Oliveira. Scrum: A systematic literature review. *International Journal of Advanced Computer Science and Applications*, 14(4):173–181, 2023.

[39] Bernardo Vasconcelos de Carvalho and Carlos Henrique Pereira Mello. Scrum agile product development method literature review, analysis and classification. *Product: Management & Development*, 9(1):39–46, 2011.

[40] Charlene Jennett, Anna L. Cox, Paul Cairns, Samira Dhoparee, Andrew Epps, Tim Tijs, and Alison Walton. Measuring and defining the experience of immersion in games. *International Journal of Human-Computer Studies*, 66(9):641–661, 2008.

A. Anexos

A.1. Carta Gantt



Nombre	Fecha de inicio	Fecha de fin
Definición Conceptual y Planificación	06-10-25	25-10-25
Desarrollo de informe de Avance 1	13-10-25	05-11-25
Informe de avance	06-11-25	06-11-25
Diseño del GDD	07-11-25	22-11-25
Desarrollo del Informe Final	17-11-25	13-12-25
Informe Final	14-12-25	14-12-25
Implementación Base del Prototipo FPS	02-03-26	11-03-26
Integración de Habilidades y progresión	16-03-26	25-03-26
Implementación del jefe final y Pulido General	30-03-26	03-04-26
Termino de Prototipo Base	03-04-26	03-04-26
Prototipo con Teclado y Mouse	03-04-26	03-04-26
Integración de motores y pruebas de los motores ASR	06-04-26	25-04-26
Prototipo con Motor de Voz Integrado	26-04-26	26-04-26
Pruebas experimentales y recolección de datos	04-05-26	18-05-26
Análisis de Datos y Resultados Finales	25-05-26	08-06-26

Figura A.1: Carta Gantt.

A.2. Game Design Document(GDD)

VOX CORE

Game Design Document

Por Diego Morales

Índice

A.3 Descripción General 49
A.3.1 Concepto del Juego 49
A.3.2 Forma de Interacción 49
A.4 Jugabilidad y Mecánicas 50
A.4.1 Jugabilidad 50
A.4.2 Personajes 53
A.5 Historia y Narrativa 55
A.5.1 Diseño Narrativo 55
A.6 Niveles y Progresión 57
A.6.1 Sinopsis del nivel 57
A.6.2 Camino Crítico 57
A.7 Interfaces 60
A.7.1 Hud y Pantallas 60
A.7.2 Controles 61
A.7.3 Propuesta de Audio 62
A.7.4 Propuesta Visual 62
A.8 Implementación 65
A.8.1 Gameplay de la Version Final del prototipo 65
A.8.2 Codigos Principales del Sistema de Interacción por Voz 65
A.9 Preguntas del cuestionario IEQ 85
A.10 Resultados Obtenidos 89
A.10.1 Tabla General de Datos Obtenidos. 89
A.10.2 Respuestas Versión por Voz 90
A.10.3 Respuestas Versión Teclado y Ratón 105

A.3. Descripción General

A.3.1. Concepto del Juego

VOX CORE es un juego de disparos en primera persona, ambientado en un futuro, en una sociedad de robots y humanos, donde el protagonista es un guardia robot que tiene la capacidad de absorber información sobre cualquier cosa, este es un tipo de robot modificado con esa característica. En una ocasión en que el protagonista estaba en una fiesta en él edifico tecnologico de robotica OpenMind donde trabaja, edificio que se encarga de crear tecnología relacionada a la robótica, cuando de repente aparece un sujeto con apariencia de robot a hacer un ataque terrorista en la fiesta, acabando con el alcalde, el cual era un robot, este sujeto vino acompañado de robots armados y de repente este sujeto empieza a acabar con la vida de los humanos de la fiesta, un dato importante de este sujeto es que tiene la habilidad de reprogramar robots, con solo tocarlos o directamente disparándoles un dardo que los reprograma, el protagonista durante el ataque no puede impedir que el alcalde fuese eliminado, pero intenta luchar contra los robots, mientras ve como estos atacan a los humanos de la fiesta y robots que antes eran amigables eran reprogramados para atacar a cualquier humano, sin embargo cuando finalmente el protagonista llega con el sujeto, este sufre de una emboscada por parte de este sujeto y sus robots, teniendo que sacarse su propio chip y pasarlo a un cuerpo de un robot que estaba por las cercanías, el cuerpo del protagonista quedo en manos del sujeto, y el protagonista en un nuevo cuerpo, pero con sus habilidades de absorción intactas deberá volver a hacerse más fuerte para poder detener a este sujeto misterioso y descubrir que trama.

A.3.2. Forma de Interacción

El Jugador controla a un personaje principal en un ambiente 3D de vista en primera persona. El juego sigue la movilidad tradicional de un Shooter FPS, con un personaje capaz de moverse en suelo, saltar y usar todo tipo de armas que irá obteniendo durante el transcurso del juego. La forma de administración de armas se elegirá mediante el teclado, como un shooter tradicional. Sin embargo también está la opción de control de acciones como recargar, cambio de armas, disparar, entre otras mediante controles de voz. Contando también con habilidades con las cuales

ya comenzará el juego, cuya interacción también será mediante teclas y mediante comandos de voz.

El ambiente del juego se divide en pisos, pues al estar en un edificio el jugador tendrá que ir ascendiendo desde lo más bajo hasta lo más alto para llegar al jefe final, obteniendo mejoras en habilidades durante el transcurso de la partida. Aunque para fines del prototipo solo se tendrá un nivel jugable de principio a fin.

El juego busca generar las sensaciones de accion frenetica durante los enfrentamientos a tiros en los juegos FPS, los controles buscarán ser lo más accesibles y menos toscos posibles, junto a unos controles por voz que buscan ser lo más precisos posibles, con animaciones y acciones rápidas, para dar aún más la sensación de rapidez y frenetismo durante el gameplay, buscando así mantener la atención del jugador en el la acción del juego. La idea es tener momentos de acción, intercalados con momentos de exploración, búsqueda de munición y recursos para después pasar a la acción.

A.4. Jugabilidad y Mecánicas

A.4.1. Jugabilidad

El loop de juego sigue el flujo tradicional de un juego de disparos en primera persona, de alternativa y principal se busca que la jugabilidad también pueda ser disfrutada mediante el uso de controles de voz que sirvan para interactuar con algunas de las acciones de un FPS, como disparar, recargar, cambiar de arma. Un aspecto a mencionar es la habilidad de absorción de información del protagonista, representado dentro del juego mediante las diferentes armas que el jugador obtendrá, usando el teclado para selección y uso, pero con los controles por voz bastará con decir un comando para cambiar de arma, además también el jugador podrá tener ya de base habilidades como ralentizar tiempo y escudo extra, estas habilidades se representarán por una tecla en la cual el jugador la ocupa con una tecla o dice un comando de voz para su uso. Al morir el jugador podrá reaparecer desde el último punto de guardado del jugador. Además de curarse ya sea con el entorno, por ejemplo objetos de recarga de energía como con objetos de curación(baterías).

Reglas del juego

Reglas Explícitas a.Obtienes armas permanentes a lo largo del prototipo.

b.Matar a los enemigos premia al jugador al desbloquear la puerta requerida para pasar.

c.Puedes usar diferentes armas y habilidades, usando controles por teclado y raton como comandos de voz.

d.La munición se puede obtener mediante el entorno.

e.La forma de obtención de armas es o buscándolas por el entorno.

f.Al apagarse(morir) el jugador, reaparecerá desde el último punto de guardado.

g.Para curarse hay baterías que el jugador puede recoger, además de tener escudos de protección los cuales también se pueden recargar con su propio consumible.

h.El jugador tiene vida representada por la energía y una barra de protección, que sirve como vida extra que al agotarse quedará expuesta la vida principal.

i.Si el jugador se queda sin energía, se apaga, lo que significa que perdió.

Reglas Implícitas a.Algunas habilidades te otorgan ventajas a la hora de enfrentar a algunos enemigos.

b.Algunas habilidades son importantes para avanzar en el prototipo.

c.El nivel tiene un diseño lineal, con un camino fijo a seguir.

d.Habrán ocasiones de acción intercaladas de otras de exploración.

e.Algunos enemigos son más resistentes, rápidos, agresivos, además de que algunos

son más vulnerables a un tipo de arma.

Consecuencias de las acciones en el juego

Al eliminar enemigos estos permiten que las puertas cerradas se desbloqueen.

Al explorar se obtienen armas, municiones o recargas de energía y protección.

Al explorar se desbloquea parte del camino, pues este a veces no es del todo lineal.

Al apagarse apareces desde el último punto de guardado. Hay puntos de control automáticos.

Comportamientos del jugador que se promueven o desalientan

Se promueve el acabar con los enemigos que el jugador se cruce, pues así este mismo puede avanzar en el nivel.

Se promueve la exploración para obtención de recursos, recargas de energía, municiones, escudos, etc.

Se promueve el ser agresivo, pero precavido, contra los enemigos.

Se promueve el usar diferentes tipos de armas dependiendo el enemigo.

Se promueve el tener precaución cuando se tiene poca energía y hay muchos enemigos.

Se promueve el aprovechar el momento oportuno para usar una habilidad.

Se desalienta el quedarse quieto durante un enfrentamiento o tiroteo.

Se desalienta el ir de frente ante muchos enemigos.

Se desalienta el no explorar lo suficiente.

Se desalienta el no recargar energía.

Se desalienta el no recoger munición.

A.4.2. Personajes

C0DEX - R3PL1K - 0103(Personaje del jugador)

El jugador juega con un robot modelo de la unidad especial de seguridad privada (U.E.S.P), el cual queda envuelto en un ataque terrorista a gran escala en el edificio tecnológico de Robótica OpenMind, cuya misión será parar este ataque.

El jugador puede moverse, correr, caminar saltar, agacharse, disparar armas y esta equipado con distintas habilidades que tendrá formas y funciones dependiendo del caso, además de sus respectivos nombres para ser usadas, mismo caso para las armas, las cuales tendrán su propio nombre para ser equipadas, dentro del prototipo jugable se tendrán habilidades como:

Arma	**Descripción**
"SteelCowBoy"	Pistola normal.
"Doomy"	Escopeta de 4 cañones.
"Dexy"	Rifle de asalto.

Tabla A.1: Armas del prototipo.

Habilidad	**Función**
"freeze" = se activa/desactiva la habilidad.	Ralentiza el tiempo por 5 segundos.
"shield" = se activa/desactiva el escudo. Esta habilidad ya estará de base con el jugador para este prototipo.	Escudo que al ser usado absorberá el daño, tendrá una barra de vida; si la barra de vida del escudo baja al mínimo, el escudo se romperá y se tendrá recuperar tomando un tipo de bateria especifica para el escudo para que este se regenere y vuelva a estar operativo.

Tabla A.2: Habilidades del prototipo.

Enemigos y Personajes no jugables

Robot de asalto(De color negro y ojos rojos)

Robot que venía con el sujeto misterioso, Tendrá un comportamiento común de un enemigo de disparo, patrullará por una ruta fija, si ve al jugador irá hasta él y disparara.

Robot Reprogramado(De color blanco y ojo rojos)

Robots que estabas en la fiesta, esta vez reprogramados por el sujeto misterioso para ser totalmente hostiles, atacarán al jugador con ataques cuerpo a cuerpo.

Robot Reprogramado de combate avanzado(De color morado y ojos rojos)

Robot que es del mismo modelo que los robots reprogramados pero más fuerte y resistente.

OV3RWR1T3 – R3C0D3(el sujeto misterioso)

Este jefe no será enfrentable y solo aparecerá en una cinemática tipo viñeta de cómic donde acorralaron al protagonista con la ayuda de sus robots programados.

Peligros

A.5. Historia y Narrativa

A.5.1. Diseño Narrativo

El mundo del juego es un futuro distante donde la tecnología es altamente avanzada y la sociedad es compartida por humanos y robots, pero no todo siempre fue paz, pues hubo un tiempo donde la raza humana oprimia con mucha violencia a los robots, hasta que los robots respondieron en el evento de la "revolucion blanca", evento donde miles de robots de color blanco de manera organizada y con violencia moderada se tomaron el control del pais en un dia, al final de esa revolucion los robots solo exigieron derechos propios y mas beneficios que los pongan a la par de la raza humana, a lo cual el pais accedio, la revolucion blanca demostro el poder de las maquinas y su autocontrol y logica avanzada, pues no buscaron imponer un regimen totalitario militarista que oprima a los humanos, pues estas sabian del pasado de la humanidad y no querian cometer los mismos errores humanos, ningun humano perdio la vida ese dia, las maquinas solo los sometieron y contuvieron de forma controlada. Este evento generó que posteriormente hubiera una época de paz, pero aún así quedaron vestigios del odio hacia las máquinas en forma de humanos que no pueden dejar el pasado atrás.

El jugador toma el control del robot modelo de la unidad especial de seguridad privada (U.E.S.P), un robot de seguridad del edificio OpenMind, un edificio de tecnología robótica que es atacado por un terrorista misterioso con apariencia de robot.

El jugador se encuentra en el sector de Innovación A-7, desde el área de recepción y seguridad y tendrá que ir atravesando aquel piso del edificio acabando con enemigos y explorando para llegar hasta los laboratorios del ala de investigación aquel sujeto misterioso.

La meta final de C0DEX – R3PL1K es detener al sujeto misterioso, persiguiendolo por todo el piso del sector de innovación.

El prototipo termina con el sujeto misterioso levantando a una horda de robots

reprogramados los cuales terminan rodeando al protagonista listos para atacar, mientras el sujeto misterioso está parado desde lejos sabiendo que el protagonista no le podrá hacer nada, esto mediante una cinemática tipo viñetas de cómic.

Hitos Importantes de la Historia

En el inicio el jugador se enfrentará a un camino semilineal que se expresa de la siguiente manera:

1.Introducción del conflicto: Primero se hará un contexto introductorio del juego, mediante texto o viñetas tipo cómic, donde se contextualiza que una gran feria de la innovación tecnológica se celebra en el edificio OpenMind, hasta que de la nada un sujeto misterioso apareció con robots de asalto bajo su mando y empezó a matar y tocar a los robots del lugar volviendolos agresivos contra otras máquinas, el jugador escucha las alarmas y al salir de su sala de entrenamiento se topa con todo el caos exterior.

2.Terminada la introducción el jugador parte desde el sector de seguridad y recepción, durante el trayecto, este se topará con enemigos que derrotar y un montón de caos a su alrededor, teniendo que avanzar para seguir al sujeto.

3.Encuentro con el sujeto misterioso: al llegar a la entrada a los laboratorios del ala de investigación el jugador se topará con el sujeto misterioso, el cual levantara las manos para resucitar a los robots que se encuentran dormidos a su alrededor, terminando con el protagonista rodeado de robots.

Personajes Importantes

OV3RWR1T3 – R3C0D3(antagonista principal)

Aparecerá solo al final, siendo el antagonista principal, sus motivos son un misterio para el protagonista y el jugador, su único rol dentro del prototipo es servir de cliffhanger (o final en suspenso) para el final del prototipo.

A.6. Niveles y Progresión

El prototipo al ser corto no contará con más niveles que solo 1, su exploración como ya se mencionó antes, será lineal, con un camino fijo a seguir.

Se contará con solo 1 nivel, el cual será el sector de Innovación A-7.

A.6.1. Sinopsis del nivel

Sector de Innovacion A-7

El prototipo se desarrolla íntegramente dentro del Edificio de Innovación Robótica A-7, una instalación de alta seguridad dedicada al desarrollo de IA, prótesis cyber mecánicas y robots autónomos.

Tras el ataque del antagonista OV3RWR1T3, todo el edificio ha sido tomado por robots reprogramados. El objetivo del jugador es avanzar por el edificio hasta llegar al Laboratorio Central, donde se encuentra el jefe final del prototipo.

A.6.2. Camino Crítico

1. Zona Inicial: Área de Recepción y Seguridad

Función: Introducción, pequeño tutorial, zona segura.

Ambientación: sillas y objetos desordenados, robots civiles dañados en el suelo y una alarma constante sonando.

Objetivos en esta zona:

Presentar movimiento, salto, agacharse, mirar, recoger arma inicial, indicar las cualidades del cuerpo robótico del protagonista, mostrar los usos de la mecánica Speak to Action, con acciones principales del juego.

2. Batalla 1: Pasillo de Vigilancia

Función: La primera zona de combate.

Enemigos:

2–4 Robots de Asalto

Objetivo del combate:

Enseñar una acción clave por voz(Solo versión Speak to Action), por ejemplo: “reload”, “cambiar arma(diciendo el nombre)” “Shield” (habilidad de escudo). El jugador aprenderá que pronunciar bien afecta que la acción se ejecute.

Diseño del espacio:

Pasillo estrecho con coberturas, luces de emergencia y chispas.

3. Exploración 1: Taller de Mantenimiento

Zona más amplia y tranquila.

Actividad: Revisar mesas de herramientas, encontrar la escopeta, descubrir munición, curas.

Objetivo de la zona:

Dar un respiro corto del combate y enseñar al jugador que debe prestar atención a su entorno, pues es en este donde encontrará armas y recursos útiles.

4. Batalla 2: Depósito de Servobots

Enemigos:

2-4 Robots reprogramados

Objetivo del combate:

Avanzar a la siguiente Área.

5. Batalla 3: Zona de producción de Servobots.

Enemigos:

2-4 Robots reprogramados

2 Robots de Asalto

Objetivo del combate:

Avanzar a la siguiente Área.

Actividad: Revisar un rincon, donde se encontrara el rifle de asalto o metralleta.

6. Batalla Final: Zona de almacenamiento de Robots de combate:

Zona llena de tanques con robots dentro, todos estan descativados pero hay 2 capsulas que estan abiertas.

Enemigos:
2 Robots Reprogramado de combate avanzado.
2 Robots Reprogramados.
Objetivo del combate:

Avanzar a la siguiente Área.

7. Zona de Transicion: Sala de base de datos

Zona más amplia y tranquila, llena de estantes de bases de datos.

Actividad: Solo es una zona para tranquilizar al jugador al final, sirviendo más como una zona de transición para el final del prototipo.

Objetivo de la zona:

Dar un respiro del combate y darle la sensación al jugador de que ya se terminó el prototipo.

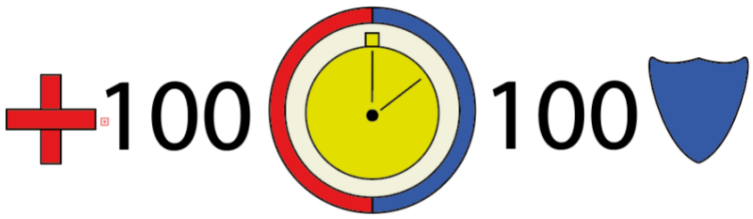
Final del prototipo: Entrada al Laboratorio Central.

El jugador ve en una cinematica tipo vinetas de comic al antagonista terminando el prototipo con un cliffhanger en el cual el protagonista y el antagonista cruzan miradas, mientras el protagonista es rodeado de robots reprogramados.

A.7. Interfaces

A.7.1. Hud y Pantallas

Para la información general sobre vida, armadura y habilidades se usará un diseño circular donde se indicará la vida y armadura estilo half life, pero con un centro donde se indicarán la habilidad que se estará usando.



En la parte izquierda irá la vida, en la derecha, la armadura y al centro un icono con la habilidad.

Para la información relacionada con la cantidad de balas, se usará el clásico estilo similar a half life 2 con el indicador de balas actuales y cantidad en cargador, además de indicar el tipo de munición.

Para la selección de armas se usara el clasico método por numero de teclado.

A.7.2. Controles

Para el movimiento se usarán las teclas WASD, para agacharse el control izquierdo, saltar con la barra de espacio, interactuar con la tecla F, correr con la tecla SHIFT IZQUIERDO, ya sea si se mantiene o se pulsa solo una vez, mirar y apuntar con el mouse, cambiar de armas y habilidades con los las teclas numéricas, usar una tecla como la letra Q para un cambio rápido de arma, la tecla E para un cambio rápido entre habilidades, para la selección por ruleta, se tiene a la tecla TABULADOR para selección de armas y la tecla T para selección de habilidades.

Para disparar se tiene el CLICK IZQUIERDO. del mouse, para apuntar con más precisión el CLICK DERECHO. del mouse, con la rueda de ratón se podrán cambiar

las armas, para usar una habilidad se usará la tecla ALT IZQUIERDA, para recargar se usará la tecla R y el menú se abrirá y cerrará con la tecla ESCAPE.

Para los controles de voz, se considerarán las acciones de disparar, cambiar de armas y habilidades, recargar. El jugador deberá decir un comando asociado a estas acciones para activarlas, en el caso del disparo será "Shoot", si se dice completo se tendrá un bonus de daño, para recargar bastará con "Reload", para las habilidades, dependiendo del comando, se seleccionan y activan de inmediato, en el caso de las armas, dependiendo de su nombre, solamente se equipan, en el entorno habrá objetos como puertas o demás que serán interactivos con la tecla de interacción, pero con el uso de comandos de voz estas interacciones tendrán un cambio sutil.

A.7.3. Propuesta de Audio

Para el audio se tendrán sonidos ambientales, de combate y de jefe, además de una canción para el menú de inicio.

El estilo de música es tipo tecno, synthware o electrónica, siendo frenética en las situaciones de acción y calmada en las situaciones de exploración.

A.7.4. Propuesta Visual

Para la propuesta visual se tiene un estilo low poly, similar a juegos como super hot, davigo, kour.io, entre otros.

Figura A.2: Superhot.

Figura A.3: Davigo.

Figura A.4: Kour.io.

Para la paleta de colores, se buscan tonos neones estilo futurista con colores intensos.

Figura A.5: CloudPunk.

Figura A.6: Tales of the Neonsea.

A.8. Implementación

A.8.1. Gameplay de la Version Final del prototipo

A continuación se presenta un video demostrativo del gameplay de la versión final del prototipo, mostrando el funcionamiento de ambas modalidades de control: teclado y ratón, y voz mediante la mecánica Speak to Action.

- YouTube: `https://www.youtube.com/watch?v=5OjFCC5WjAc`
- Google Drive: `https://drive.google.com/file/d/1Wa1KzIPbbvk8PabsIjAK930wlDV6tKFZ/view`

A.8.2. Codigos Principales del Sistema de Interacción por Voz

El sistema de interacción por voz del prototipo está compuesto por tres scripts principales que trabajan de forma coordinada. VoiceProcessor es el componente encargado de capturar el audio del micrófono en tiempo real, detectar la presencia de voz mediante un umbral de volumen configurable, y entregar el audio capturado en fragmentos (frames) de tamaño fijo mediante eventos. VoskSpeechToText recibe estos fragmentos de audio a través de dicho evento, los procesa utilizando el motor de reconocimiento de voz Vosk en un hilo separado para no bloquear el hilo principal

de Unity, y genera como resultado transcripciones parciales y finales del habla detectada, las cuales expone mediante su propio evento de transcripción. Finalmente, VoiceController actúa como el componente central del sistema: se suscribe al evento de transcripción de VoskSpeechToText, interpreta el texto reconocido en busca de palabras clave predefinidas (comandos de disparo, recarga, cambio de arma y activación de habilidades), y ejecuta las acciones correspondientes sobre los sistemas de armas, escudo y ralentización del tiempo del jugador. A continuación se presentan los fragmentos de código más relevantes de cada uno de estos tres componentes.

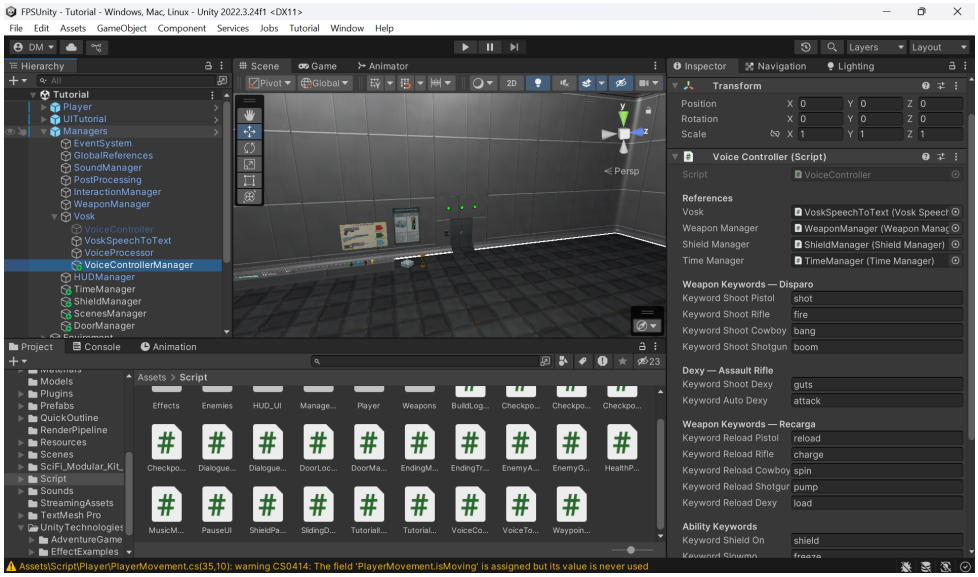


Figura A.7: Imagen de Unity 3D del manager principal Vosk y sus componentes

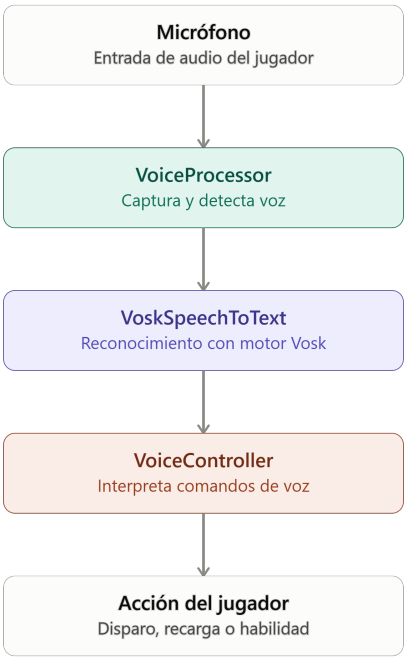


Figura A.8: Diagrama de Flujo del Funcionamiento de Vosk

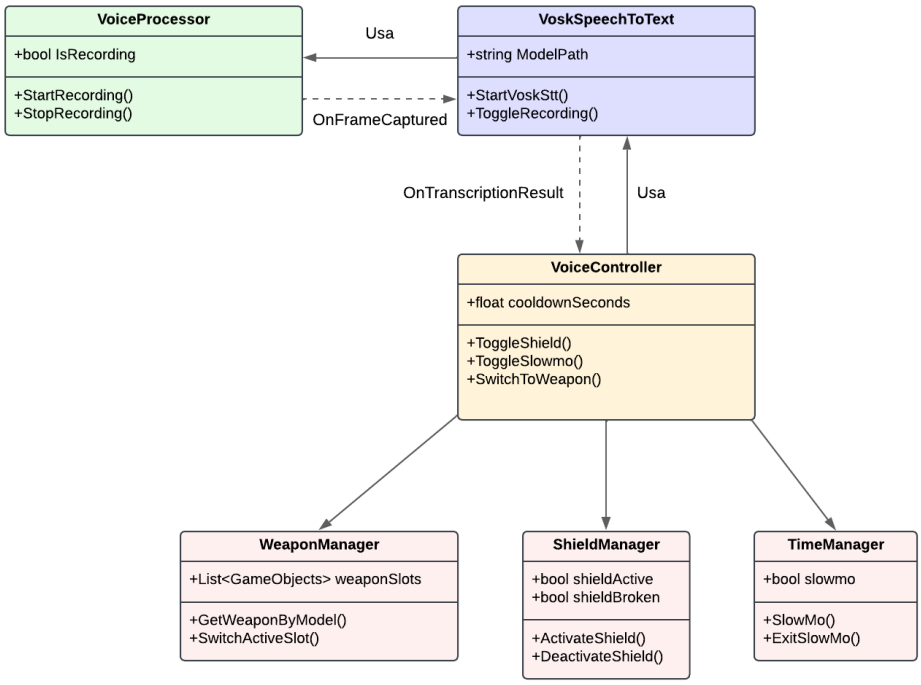


Figura A.9: Diagrama de Clases del Funcionamiento de Vosk

VoiceProcessor

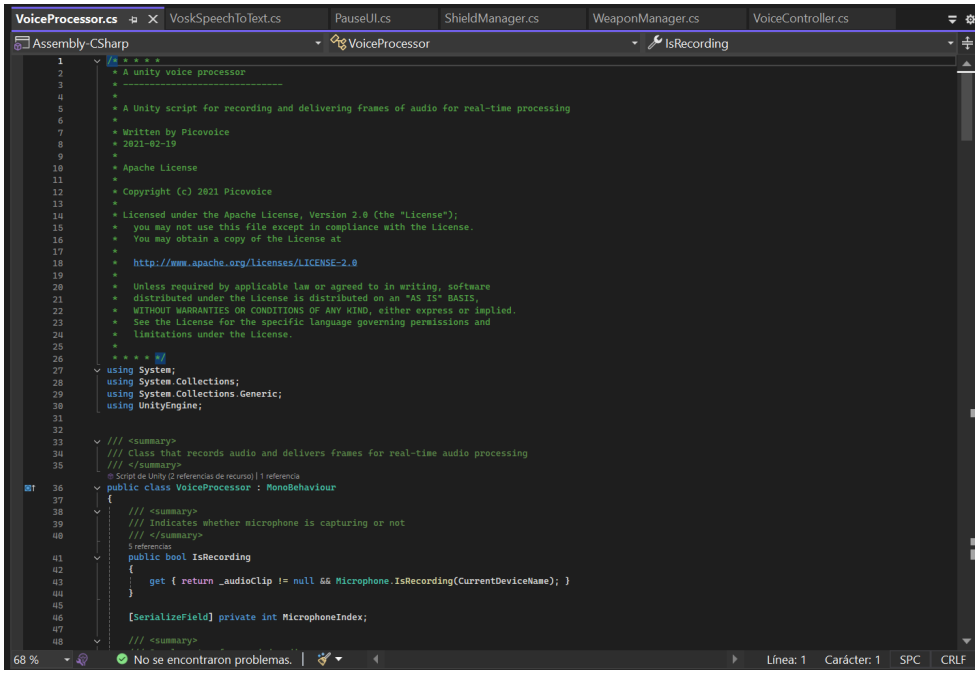


Figura A.10: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

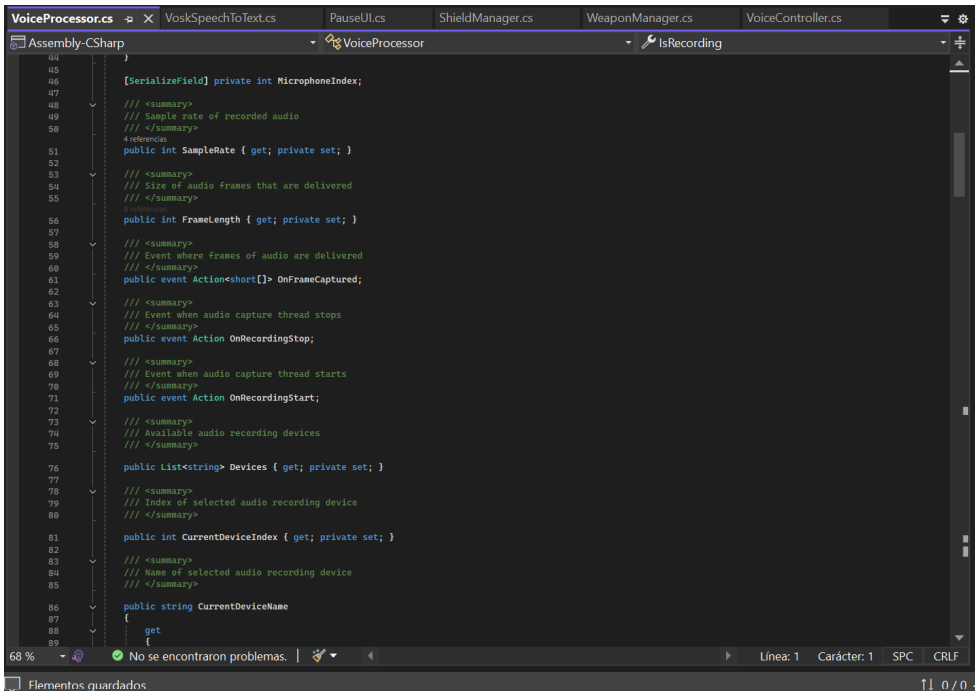


Figura A.11: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

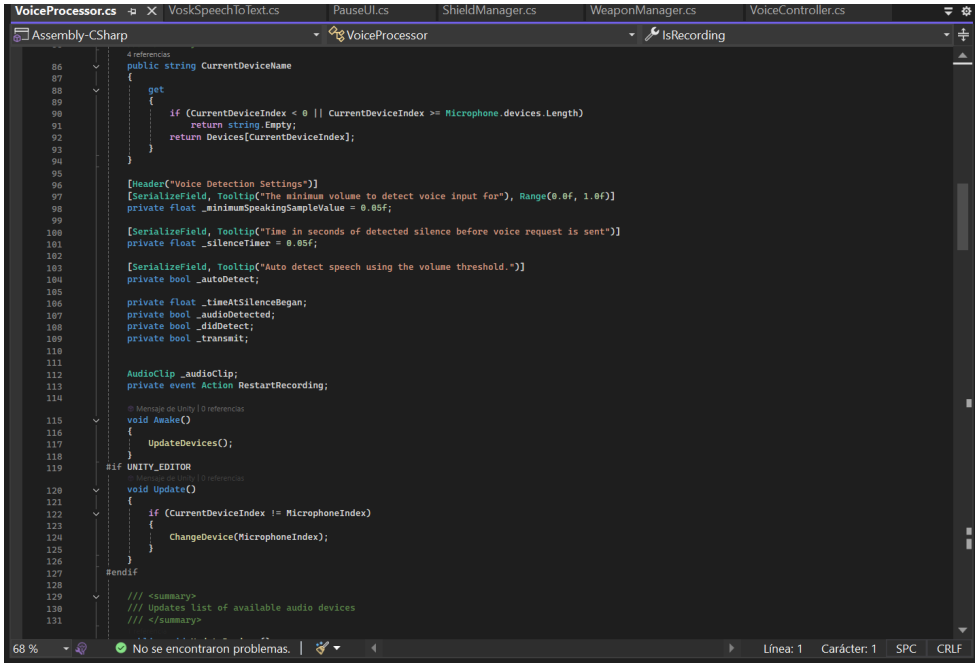


Figura A.12: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

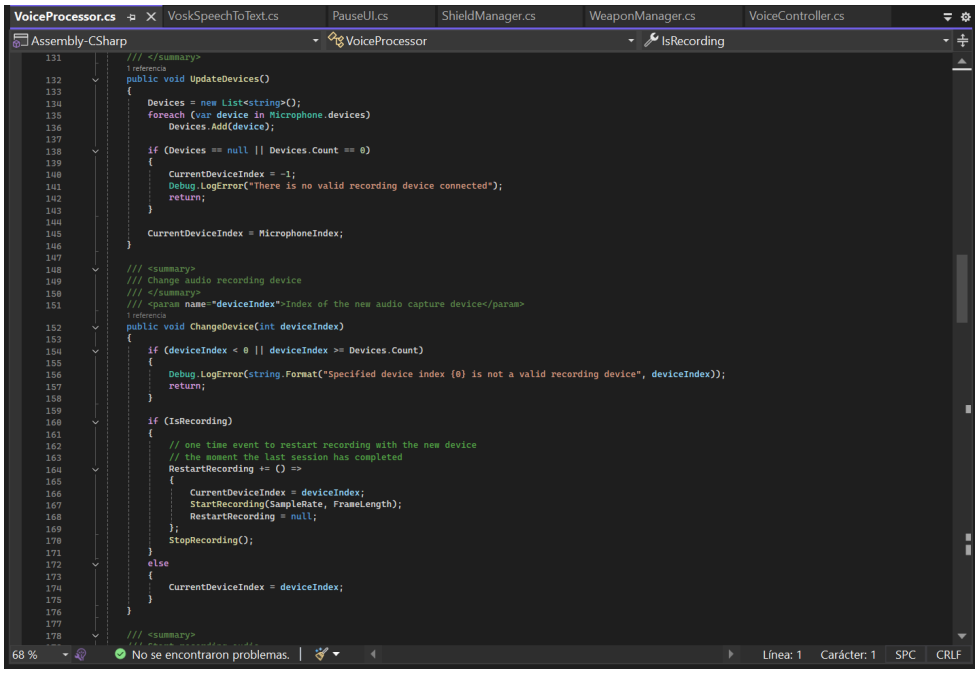


Figura A.13: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

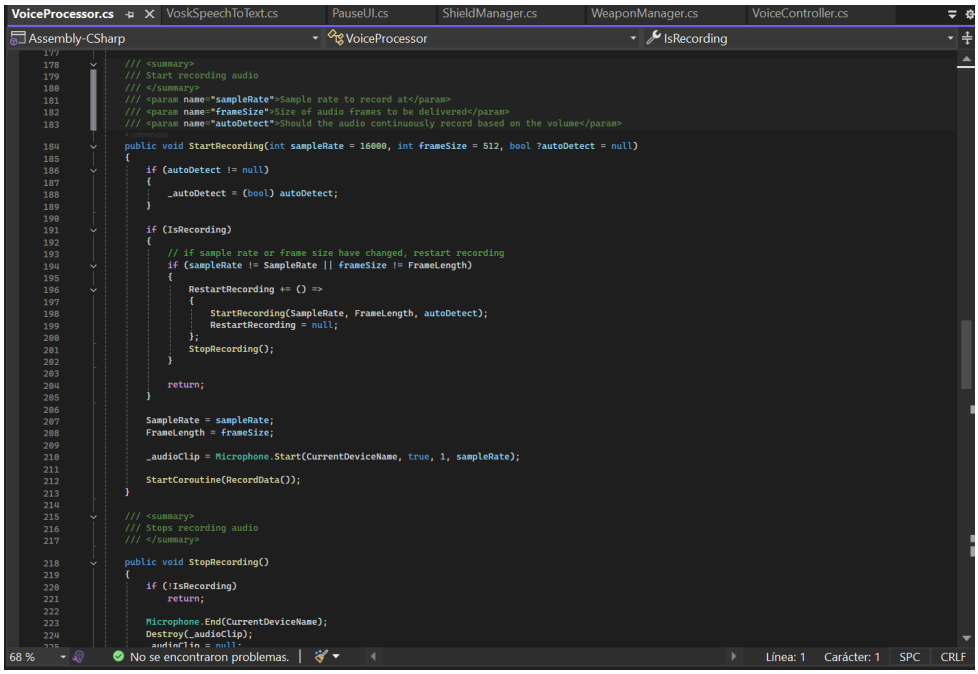


Figura A.14: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

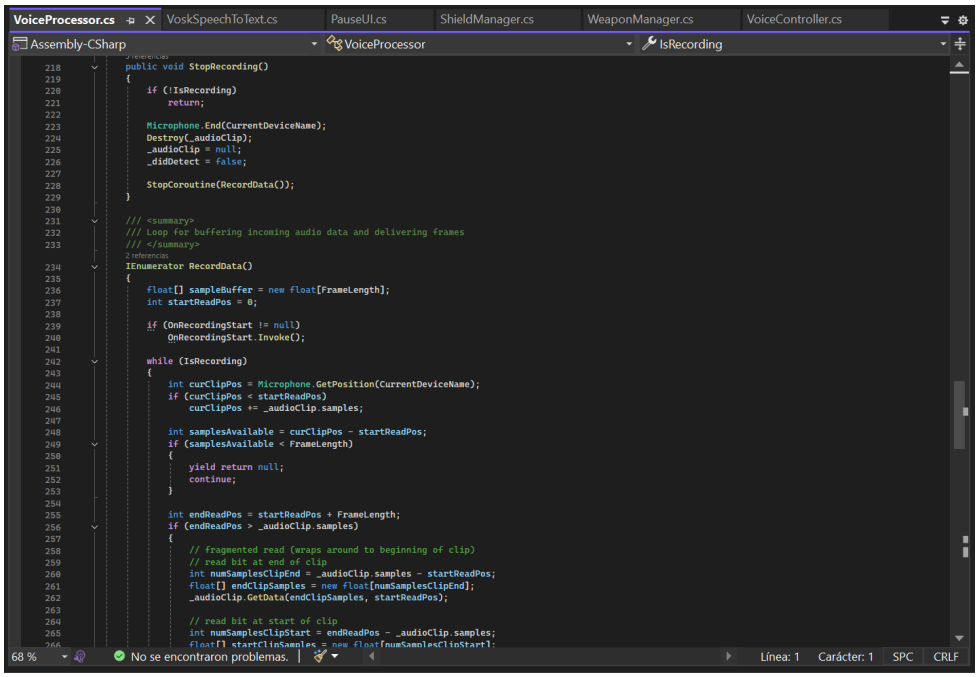


Figura A.15: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

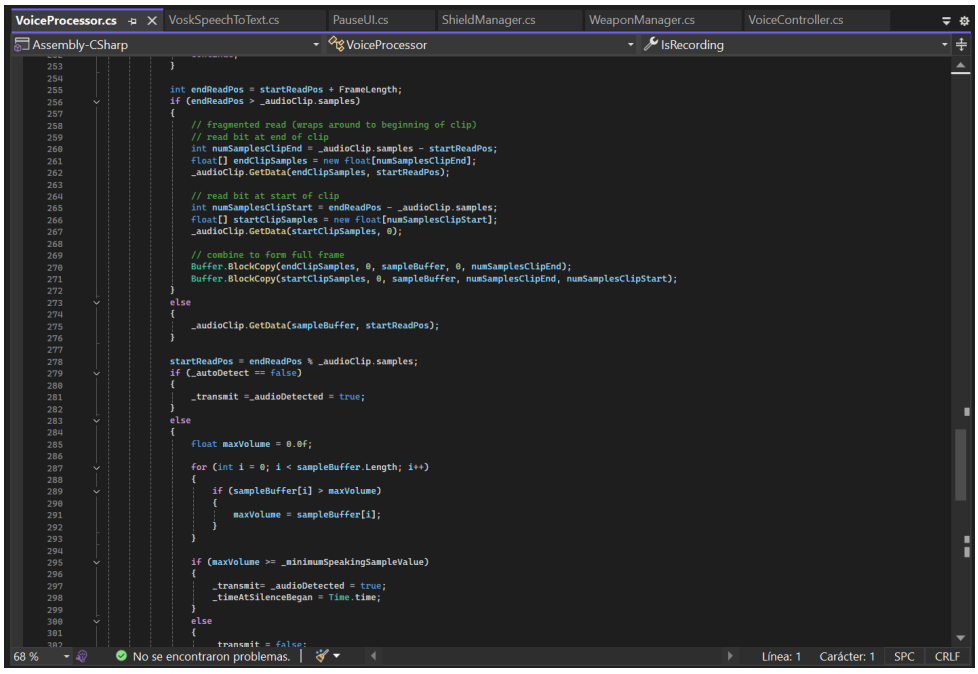


Figura A.16: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

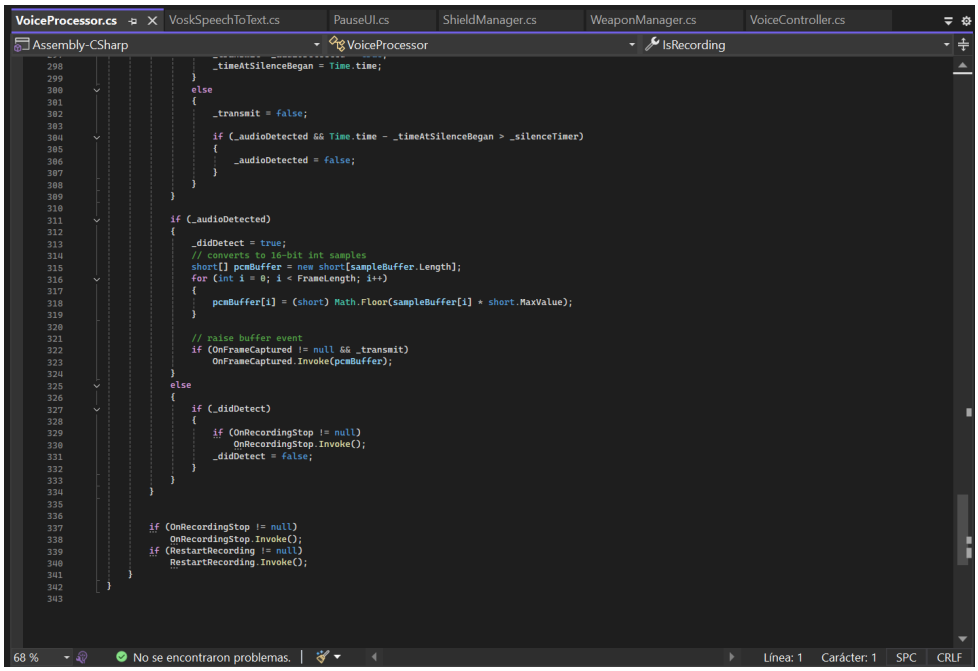


Figura A.17: Fragmento de código de VoiceProcessor: captura y detección de audio del micrófono.

VoskSpeechToText

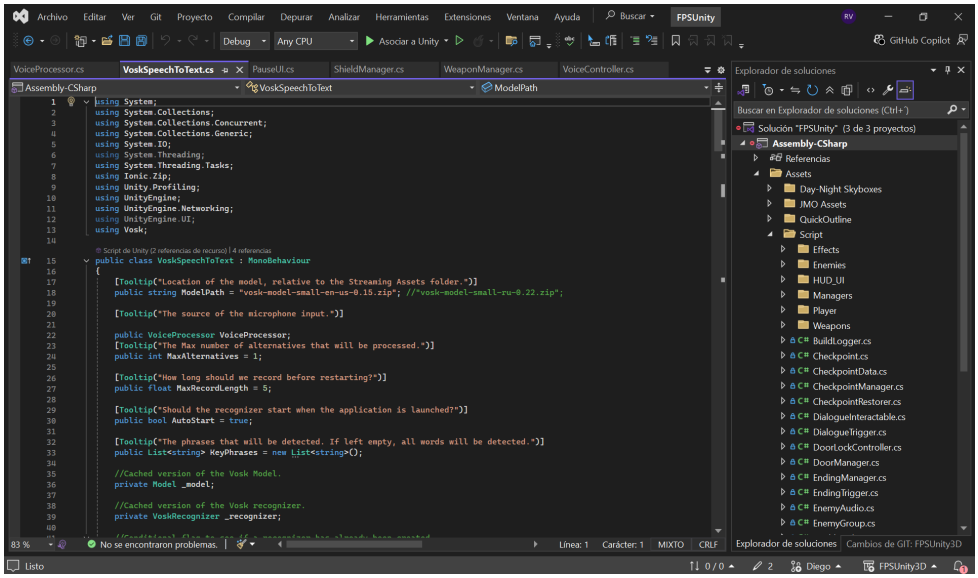


Figura A.18: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

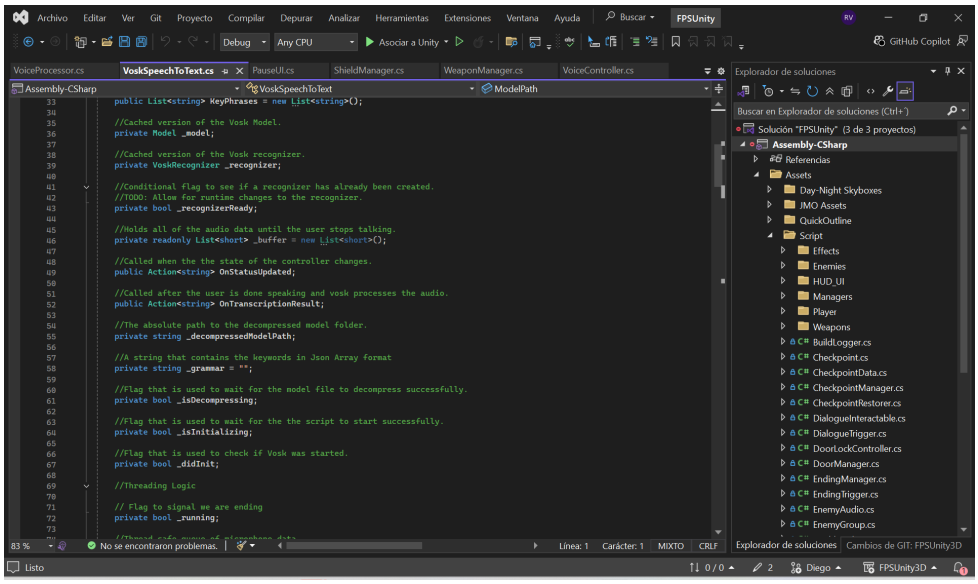


Figura A.19: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

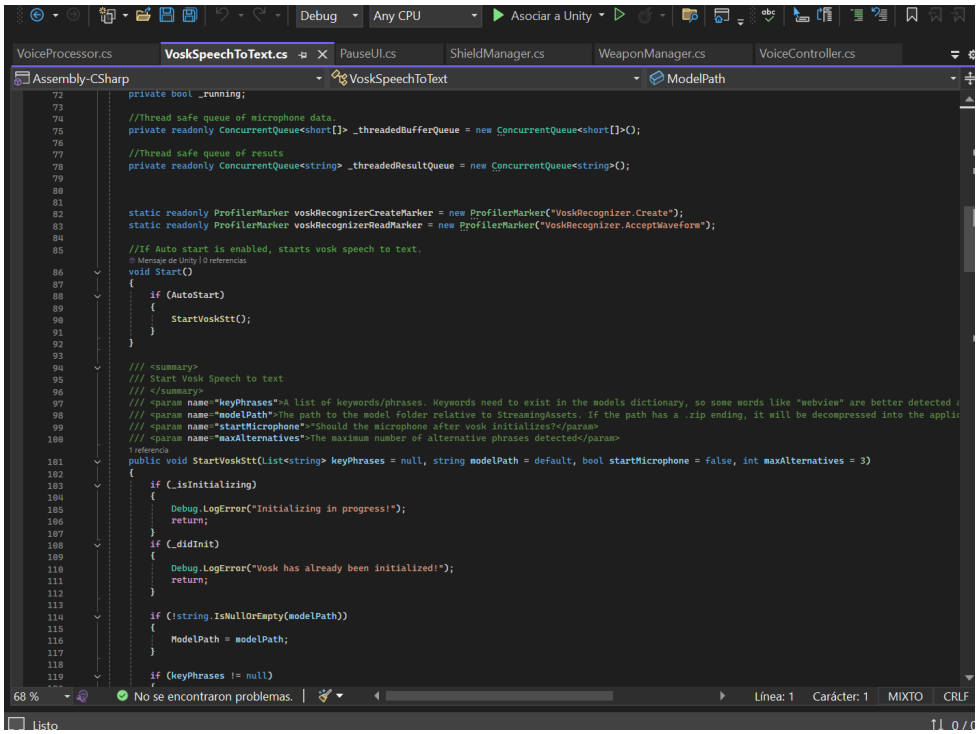
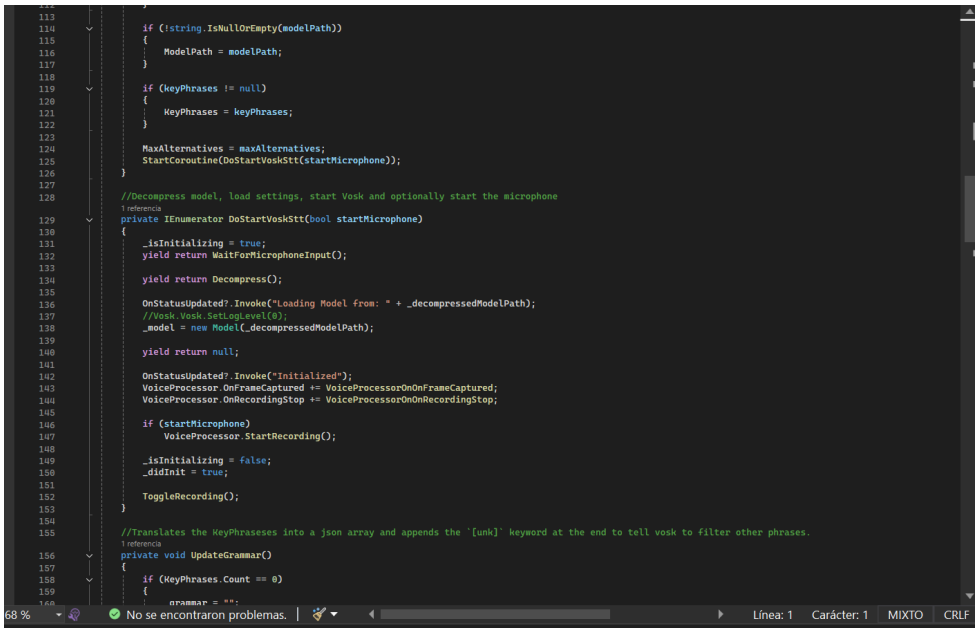


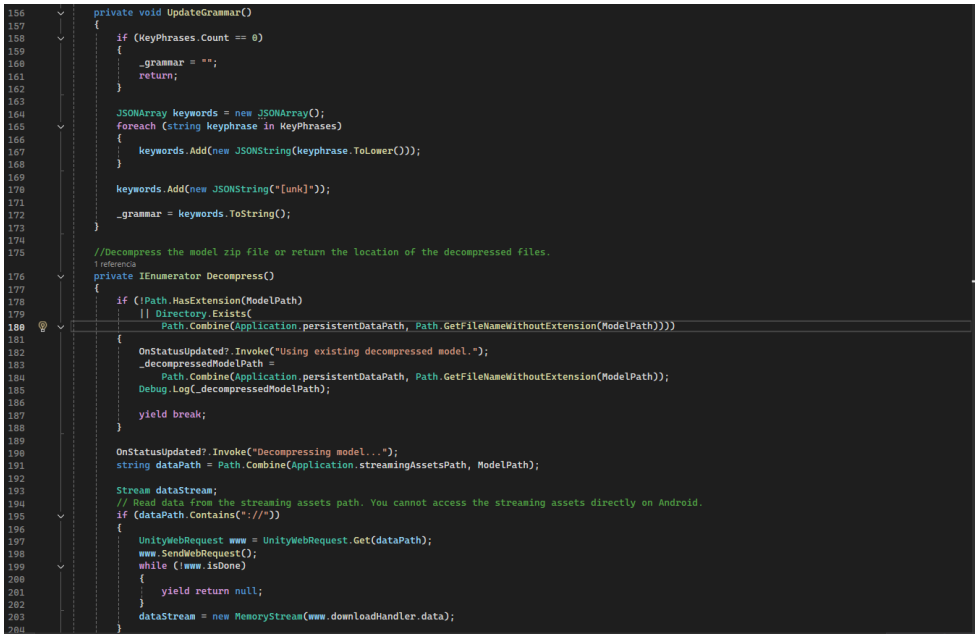
Figura A.20: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.



```
        if (!string.IsNullOrEmpty(modelPath))
        {
            ModelPath = modelPath;
        }

        if (keyPhrases != null)
        {
            KeyPhrases = keyPhrases;
        }

        MaxAlternatives = maxAlternatives;
        StartCoroutine(DoStartVoskStt(startMicrophone));
    }

    //Decompress model, load settings, start Vosk and optionally start the microphone
    1 referencia
    private IEnumerator DoStartVoskStt(bool startMicrophone)
    {
        _isInitializing = true;
        yield return WaitForMicrophoneInput();

        yield return Decompress();

        OnStatusUpdated?.Invoke("Loading Model from: " + _decompressedModelPath);
        //Vosk.Vosk.SetLogLevel(0);
        _model = new Model(_decompressedModelPath);

        yield return null;

        OnStatusUpdated?.Invoke("Initialized");
        VoiceProcessor.OnFrameCaptured += VoiceProcessorOnOnFrameCaptured;
        VoiceProcessor.OnRecordingStop += VoiceProcessorOnOnRecordingStop;

        if (startMicrophone)
            VoiceProcessor.StartRecording();

        _isInitializing = false;
        _didInit = true;

        ToggleRecording();
    }

    //Translates the KeyPhraseses into a json array and appends the `[unk]` keyword at the end to tell vosk to filter other phrases.
    1 referencia
    private void UpdateGrammar()
    {
        if (KeyPhrases.Count == 0)
        {
            grammar = "";
```

68 % No se encontraron problemas. Línea: 1 Carácter: 1 MIXTO CRLF

Figura A.21: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.



```
private void UpdateGrammar()
{
    if (KeyPhrases.Count == 0)
    {
        _grammar = "";
        return;
    }

    JSONArray keywords = new JSONArray();
    foreach (string keyphrase in KeyPhrases)
    {
        keywords.Add(new JSONString(keyphrase.ToLower()));
    }

    keywords.Add(new JSONString("[unk]"));

    _grammar = keywords.ToString();
}

//Decompress the model zip file or return the location of the decompressed files.
1 referencia
private IEnumerator Decompress()
{
    if (!Path.HasExtension(ModelPath)
        || Directory.Exists(
            Path.Combine(Application.persistentDataPath, Path.GetFileNameWithoutExtension(ModelPath))))
    {
        OnStatusUpdated?.Invoke("Using existing decompressed model.");
        _decompressedModelPath =
            Path.Combine(Application.persistentDataPath, Path.GetFileNameWithoutExtension(ModelPath));
        Debug.Log(_decompressedModelPath);

        yield break;
    }

    OnStatusUpdated?.Invoke("Decompressing model...");
    string dataPath = Path.Combine(Application.streamingAssetsPath, ModelPath);

    Stream dataStream;
    // Read data from the streaming assets path. You cannot access the streaming assets directly on Android.
    if (dataPath.Contains("://"))
    {
        UnityWebRequest www = UnityWebRequest.Get(dataPath);
        www.SendWebRequest();
        while (!www.isDone)
        {
            yield return null;
        }
        dataStream = new MemoryStream(www.downloadHandler.data);
    }
```

Figura A.22: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

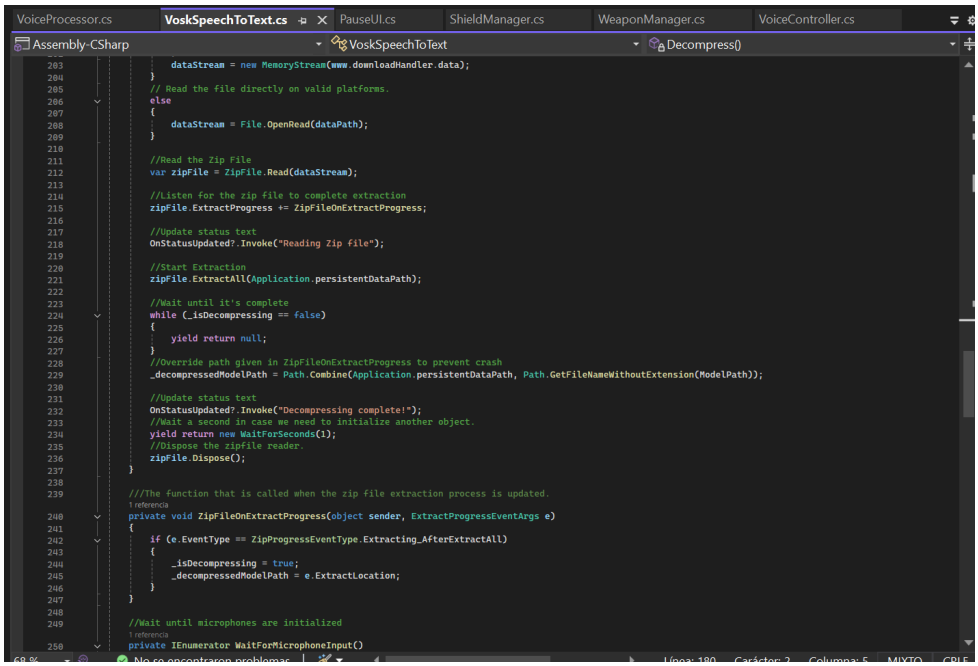


Figura A.23: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

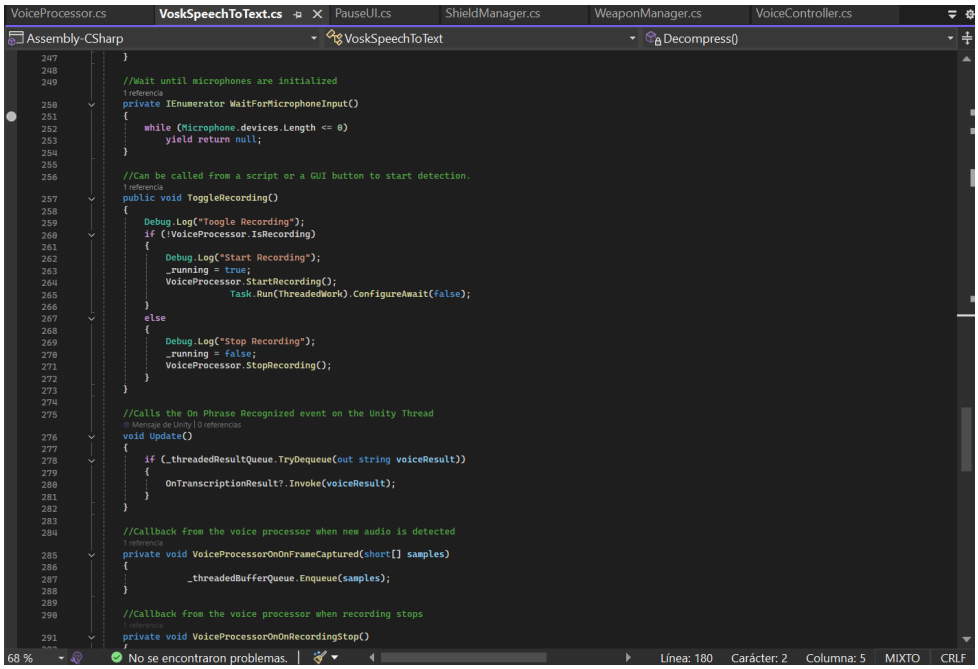


Figura A.24: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

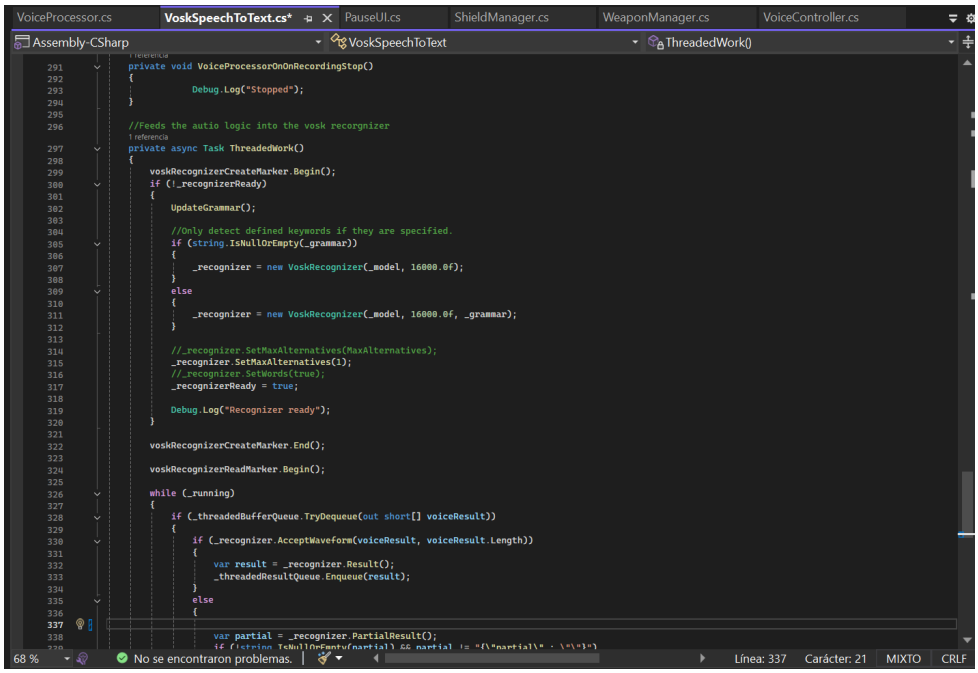


Figura A.25: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

```
        voskRecognizerCreateMarker.End();

        voskRecognizerReadMarker.Begin();

        while (_running)
        {
            if (_threadedBufferQueue.TryDequeue(out short[] voiceResult))
            {
                if (_recognizer.AcceptWaveform(voiceResult, voiceResult.Length))
                {
                    var result = _recognizer.Result();
                    _threadedResultQueue.Enqueue(result);
                }
                else
                {

                    var partial = _recognizer.PartialResult();
                    if (!string.IsNullOrEmpty(partial) && partial != "{\"partial\" : \"\"}")
                        _threadedResultQueue.Enqueue(partial);
                }
            }
            else
            {
                // Wait for some data
                //await Task.Delay(100);
                await Task.Delay(10);
            }
        }

        voskRecognizerReadMarker.End();
    }


}
```

Figura A.26: Fragmento de código de VoskSpeechToText: procesamiento del audio mediante el motor Vosk.

VoiceController

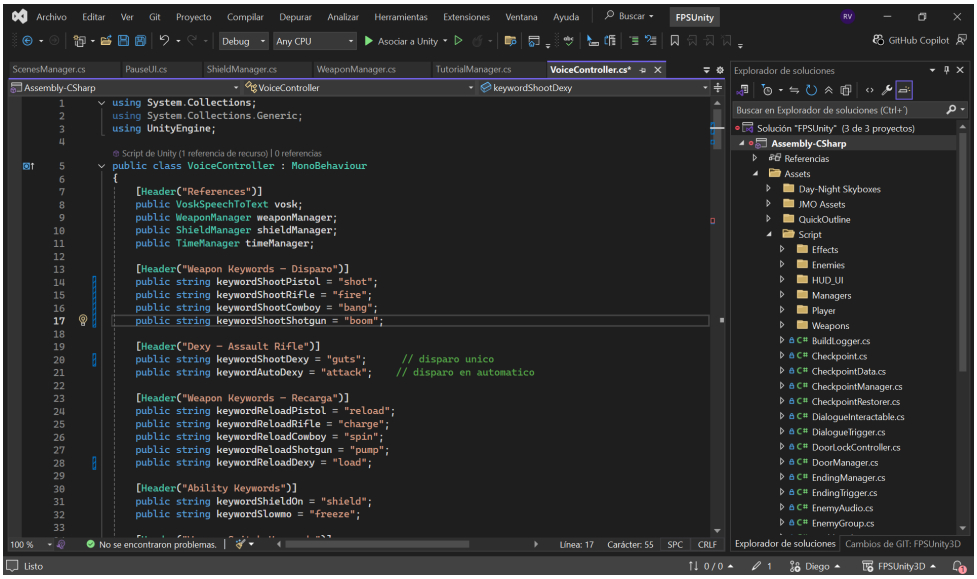


Figura A.27: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

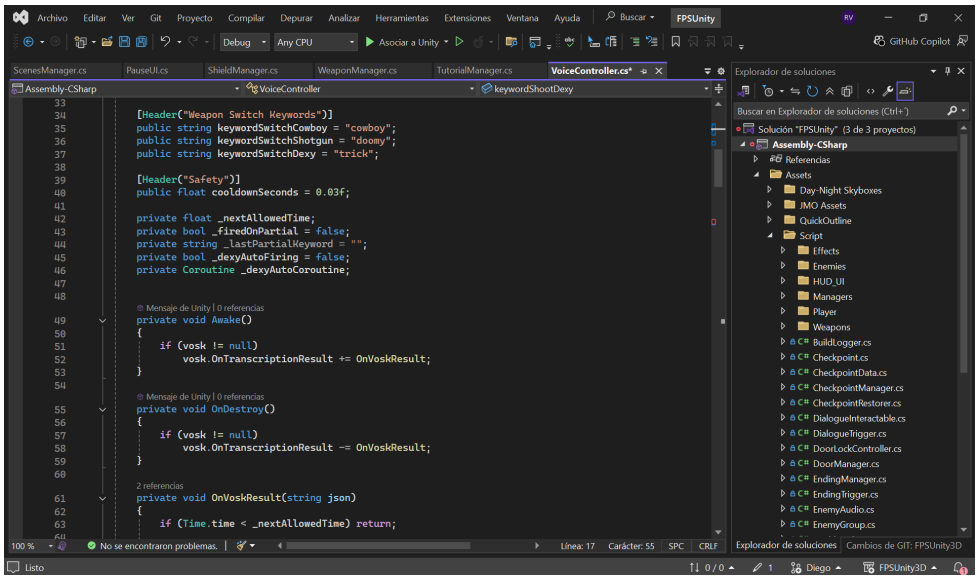


Figura A.28: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

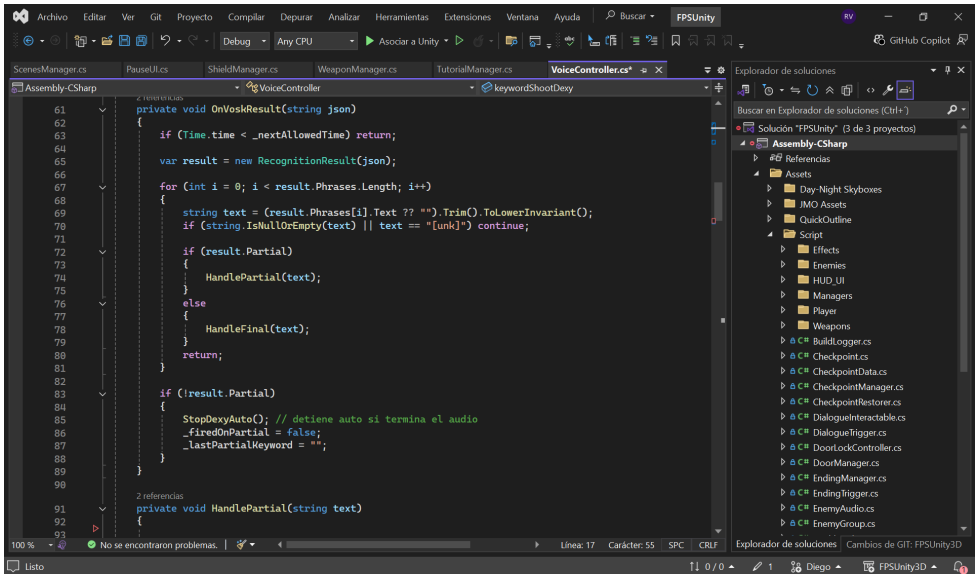


Figura A.29: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

```
2 referencias
private void HandlePartial(string text)
{

    // si el partial tiene contenido pero no tiene la palabra "attack", detiene el auto
    if (_dexyAutoFiring && !text.Contains(keywordAutoDexy))
    {
        StopDexyAuto();
        return; // sale inmediatamente
    }

    if (_firedOnPartial) return;

    // --- Disparos --- //
    if (TryShootWeapon(text, keywordShootPistol, Weapon.WeaponModel.C1911)) return;
    if (TryShootWeapon(text, keywordShootRifle, Weapon.WeaponModel.AK47)) return;
    if (TryShootWeapon(text, keywordShootCowboy, Weapon.WeaponModel.SCowBoy)) return;
    if (TryShootWeapon(text, keywordShootShotgun, Weapon.WeaponModel.DoomyShotgun)) return;

    if (TryShootWeapon(text, keywordShootDexy, Weapon.WeaponModel.DexyAssaultRifle)) return;

    // Auto Dexy
    if (text.Contains(keywordAutoDexy))
    {
        Weapon dexy = weaponManager.GetWeaponByModel(Weapon.WeaponModel.DexyAssaultRifle);
        if (dexy != null && dexy.isActiveWeapon && !_dexyAutoFiring)
        {
            _dexyAutoFiring = true;
            _dexyAutoCoroutine = StartCoroutine(AutoFireDexy(dexy));
        }
        return;
    }
```

Figura A.30: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

```
// --- Recargas --- //
if (TryReloadWeapon(text, keywordReloadPistol, Weapon.WeaponModel.C1911)) return;
if (TryReloadWeapon(text, keywordReloadRifle, Weapon.WeaponModel.AK47)) return;
if (TryReloadWeapon(text, keywordReloadCowboy, Weapon.WeaponModel.SCowBoy)) return;
if (TryReloadWeapon(text, keywordReloadShotgun, Weapon.WeaponModel.DoomyShotgun)) return;
if (TryReloadWeapon(text, keywordReloadDexy, Weapon.WeaponModel.DexyAssaultRifle)) return;

// --- Habilidades --- //
if (text.Contains(keywordShieldOn))
{
    _firedOnPartial = true;
    _lastPartialKeyword = keywordShieldOn;
    _nextAllowedTime = Time.time + cooldownSeconds;
    ToggleShield();
    return;
}

if (text.Contains(keywordSlowmo))
{
    _firedOnPartial = true;
    _lastPartialKeyword = keywordSlowmo;
    _nextAllowedTime = Time.time + cooldownSeconds;
    ToggleSlowmo();
    return;
}
// --- Cambio de armas --- //
if (text.Contains(keywordSwitchCowboy))
{
    _firedOnPartial = true;
    _nextAllowedTime = Time.time + cooldownSeconds;
    SwitchToWeapon(Weapon.WeaponModel.SCowBoy);
    return;
}
```

Figura A.31: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

```
        // --- Cambio de armas --- //
        if (text.Contains(keywordSwitchCowboy))
        {
            _firedOnPartial = true;
            _nextAllowedTime = Time.time + cooldownSeconds;
            SwitchToWeapon(Weapon.WeaponModel.SCowBoy);
            return;
        }
        if (text.Contains(keywordSwitchShotgun))
        {
            _firedOnPartial = true;
            _nextAllowedTime = Time.time + cooldownSeconds;
            SwitchToWeapon(Weapon.WeaponModel.DoomyShotgun);
            return;
        }
        if (text.Contains(keywordSwitchDexy))
        {
            _firedOnPartial = true;
            _nextAllowedTime = Time.time + cooldownSeconds;
            SwitchToWeapon(Weapon.WeaponModel.DexyAssaultRifle);
            return;
        }

    }
```

Figura A.32: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

```
3 referencias
private void SwitchToWeapon(Weapon.WeaponModel model)
{
    Weapon[] allWeapons = FindObjectsOfType<Weapon>();
    for (int i = 0; i < WeaponManager.Instance.weaponSlots.Count; i++)
    {
        if (WeaponManager.Instance.weaponSlots[i].transform.childCount > 0)
        {
            Weapon w = WeaponManager.Instance.weaponSlots[i].transform.GetChild(0).GetComponent<Weapon>();
            if (w != null && w.thisWeaponModel == model)
            {
                WeaponManager.Instance.SwitchActiveSlot(i);
                return;
            }
        }
    }
}
```

Figura A.33: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador..

```
1 referencia
private void HandleFinal(string text)
{
    // Detiene el auto fire si el resultado final no contiene la keyword
    if (!text.Contains(keywordAutoDexy))
    {
        StopDexyAuto();
    }

    if (!_firedOnPartial)
    {
        // Si no disparó en parcial, intenta actuar ahora
        HandlePartial(text);
    }
    // Resetea para el próximo comando
    _firedOnPartial = false;
    _lastPartialKeyword = "";
}

// --- Helpers --- //

5 referencias
private bool TryShootWeapon(string text, string keyword, Weapon.WeaponModel model)
{
    if (!text.Contains(keyword)) return false;

    Weapon weapon = weaponManager.GetWeaponByModel(model);
    if (weapon != null && weapon.isActiveWeapon)
    {
        _firedOnPartial = true;
        _lastPartialKeyword = keyword;
        _nextAllowedTime = Time.time + cooldownSeconds;
        weapon.TryFireFromVoice();
    }
    return true;
}
```

Figura A.34: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

```
5 referencias
private bool TryReloadWeapon(string text, string keyword, Weapon.WeaponModel model)
{
    if (!text.Contains(keyword)) return false;

    Weapon weapon = weaponManager.GetWeaponByModel(model);
    if (weapon != null && weapon.isActiveWeapon)
    {
        _firedOnPartial = true;
        _lastPartialKeyword = keyword;
        _nextAllowedTime = Time.time + cooldownSeconds;
        weapon.TryReloadFromVoice();
    }
    return true;
}

1 referencia
private void ToggleShield()
{
    if (shieldManager == null) return;
    if (shieldManager.shieldBroken) return;
    if (shieldManager.shieldActive)
        shieldManager.DeactivateShield();
    else
        shieldManager.ActivateShield();
}

1 referencia
private void ToggleSlowmo()
{
    if (timeManager == null) return;
    if (timeManager.slowmo)
        timeManager.ExitSlowMo();
    else
        timeManager.SlowMo();
}
```

Figura A.35: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

```
1 referencia
private IEnumerator AutoFireDexy(Weapon dexy)
{
    float maxAutoTime = 3f; // maximo 3 segundos de auto aunque no llegue el resultado final
    float elapsed = 0f;

    while (_dexyAutoFiring)
    {
        elapsed += 0.03f;
        if (elapsed >= maxAutoTime)
        {
            StopDexyAuto();
            yield break;
        }
        if (dexy == null || !dexy.isActiveWeapon || dexy.bulletsLeft <= 0)
        {
            StopDexyAuto();
            yield break;
        }
        dexy.TryFireFromVoice();
        yield return new WaitForSeconds(0.03f);
    }
}


5 referencias
private void StopDexyAuto()
{
    _dexyAutoFiring = false;
    if (_dexyAutoCoroutine != null)
    {
        StopCoroutine(_dexyAutoCoroutine);
        _dexyAutoCoroutine = null;
    }
}
```

Figura A.36: Fragmento de código de VoiceController: interpretación de comandos de voz y ejecución de acciones del jugador.

A.9. Preguntas del cuestionario IEQ

A continuación se mostrarán las preguntas del cuestionario IEQ mencionado en el marco teorico:

En base a tu experiencia en el juego Responda las siguientes preguntas marcando con un círculo el número correspondiente. En particular, recuerde que estas preguntas le preguntan cómo se sintió al final del juego.

¿Hasta qué punto te llamó la atención el juego?
Nada. 1 2 3 4 5 Mucho.

¿Hasta qué punto te sentiste concentrado en el juego?
Nada. 2 3 4 5 Mucho.

¿Cuánto esfuerzo dedicaste al juego?
Muy poco 1 2 3 4 5 Mucho.

¿Sentiste que te esforzaste al máximo?
Para nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto perdiste la noción del tiempo?
En absoluto 1 2 3 4 5 Mucho.

¿Hasta qué punto te sentiste consciente de estar en el mundo real mientras jugabas?
Nada. 1 2 3 4 5 Muchísimo.

¿Hasta qué punto te olvidaste de tus preocupaciones cotidianas?
Para nada. 1 2 3 4 5 Mucho.

¿Hasta qué punto eras consciente de ti mismo en tu entorno?
Nada. 1 2 3 4 5 Muy consciente.

¿Hasta qué punto notó los eventos que ocurrían a su alrededor?
Nada 1 2 3 4 5 Mucho.

¿Sentiste en algún momento la necesidad de dejar de jugar y ver qué estaba pasando a tu alrededor?
De nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto sentiste que interactuabas con el entorno del juego?
Nada. 1 2 3 4 5 Mucho.

¿En qué medida te sentiste separado de tu entorno del mundo real?
De nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto sentiste que el juego era algo que estabas experimentando, en lugar de algo que simplemente estabas haciendo?

De ningún modo 1 2 3 4 5 Muchísimo.

¿En qué medida tu sensación de estar en el entorno del juego fue más fuerte que tu sensación de estar en el mundo real?
De ningún modo 1 2 3 4 5 Muchísimo.

¿En algún momento te sentiste tan involucrado que ni siquiera te diste cuenta de que estabas usando los controles?
Para nada 1 2 3 4 5 Muchísimo.

¿En qué medida sentiste que ibas avanzando en el juego según tu propia voluntad?
De nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto te pareció desafiante el juego?
Nada 1 2 3 4 5 Muy difícil.

¿Hubo algún momento durante el juego en el que simplemente quisiste rendirte?
Para nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto te sentiste motivado mientras jugabas?
Nada 1 2 3 4 5 Mucho.

¿Hasta qué punto te resultó fácil el juego?
Nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto sentiste que progresabas hacia el final del juego?
Nada 1 2 3 4 5 Mucho.

¿Qué tan bien crees que te desempeñaste en el juego?
Muy malo 1 2 3 4 5 Muy bien.

¿Hasta qué punto te sentiste emocionalmente conectado con el juego?
Nada 1 2 3 4 5 Muchísimo.

¿Hasta qué punto te interesaba ver cómo progresarían los eventos del juego?
Nada 1 2 3 4 5 Mucho.

¿Cuánto querías "ganar.[el] juego?
Para nada 1 2 3 4 5 Muchísimo.

¿Tenías dudas sobre si ganarías o perderías el juego?
Para nada 1 2 3 4 5 Muchísimo.

¿En algún momento te sentiste tan involucrado que quisiste hablar directamente sobre el juego?
De nada 1 2 3 4 5 Muchísimo.

¿Cuánto disfrutaste de los gráficos y las imágenes?
Nada 1 2 3 4 5 Mucho.

¿Cuánto dirías que disfrutaste jugando?
Nada. 1 2 3 4 5 Mucho.

Cuando te interrumpieron, ¿te decepcionó que el juego terminará?
Para nada 1 2 3 4 5 Muchísimo.

¿Te gustaría volver a jugar?
¡Claro que no! 1 2 3 4 5 Definitivamente sí.

A.10. Resultados Obtenidos

A.10.1. Tabla General de Datos Obtenidos.

Análisis	**N**	**Media Teclado/Ratón**	**Media Voz**	**Diferencia**	**Prueba t**	**Significancia**
Puntaje total	20	100,90	112,00	+11,10	t(19)=4,53, $p < 0{,}001$	*** (d=1,01)
Puntaje total (contrabalanceado)	8	104,00	114,88	+10,88	t(7)=2,84, p=0,025	* (d=1,00)
Atención	20	3,59	4,31	+0,72	$p < 0{,}001$	***
Desafío	20	3,59	3,89	+0,30	p=0,002	**
Implicación Emocional	20	3,33	3,68	+0,35	p=0,007	**
Transporte	20	3,62	3,96	+0,34	p=0,046	*
Disociación Temporal	20	2,97	3,44	+0,47	p=0,045	*
Disfrute	20	4,15	4,18	+0,03	p=0,789	n.s.

Tabla A.3: Resumen general de resultados: puntaje total (muestra completa y contrabalanceada) y análisis por subdimensión del IEQ.

La Tabla A.3 consolida los resultados de la prueba t de Student para muestras relacionadas aplicada tanto al puntaje total de inmersión (en la muestra completa de N=20 y en el subconjunto contrabalanceado de N=8) como a cada una de las seis subdimensiones del IEQ evaluadas sobre la muestra completa.

En ambos análisis de puntaje total, la condición de voz obtuvo una media significativamente mayor que la condición de teclado/ratón, con un tamaño de efecto grande en ambos casos (d es aproximadamente 1,0), lo que confirma la consistencia del hallazgo principal independientemente del subconjunto de datos analizado.

A nivel de subdimensión, cinco de las seis mostraron diferencias significativas a favor de la condición de voz, siendo Atención la de mayor magnitud (d=1,13). Únicamente la subdimensión Disfrute no mostró diferencia estadísticamente significativa entre condiciones, siendo además la subdimensión con mayor puntaje en ambas versiones del prototipo. Los símbolos de significancia se interpretan de la siguiente forma: *** p¡0,001, ** p¡0,01, * p¡0,05, n.s. = no significativo.

A.10.2. Respuestas Versión por Voz

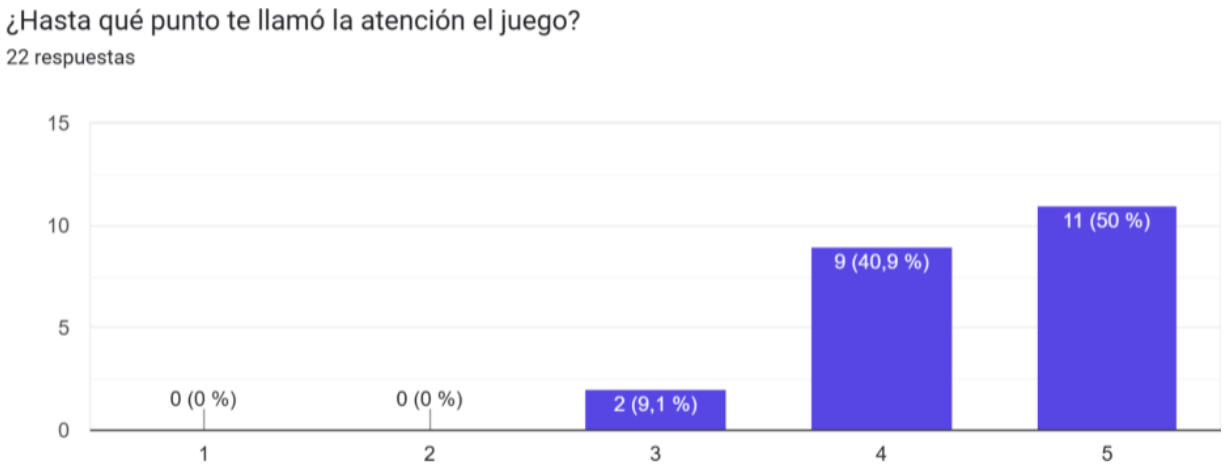


Figura A.37: Respuestas Versión por Voz – Pregunta 1.

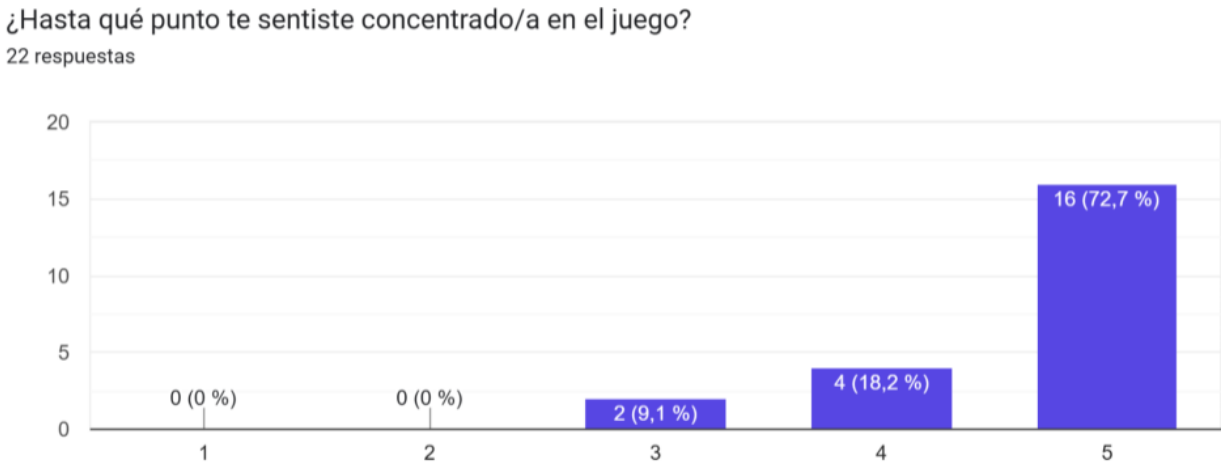


Figura A.38: Respuestas Versión por Voz – Pregunta 2.

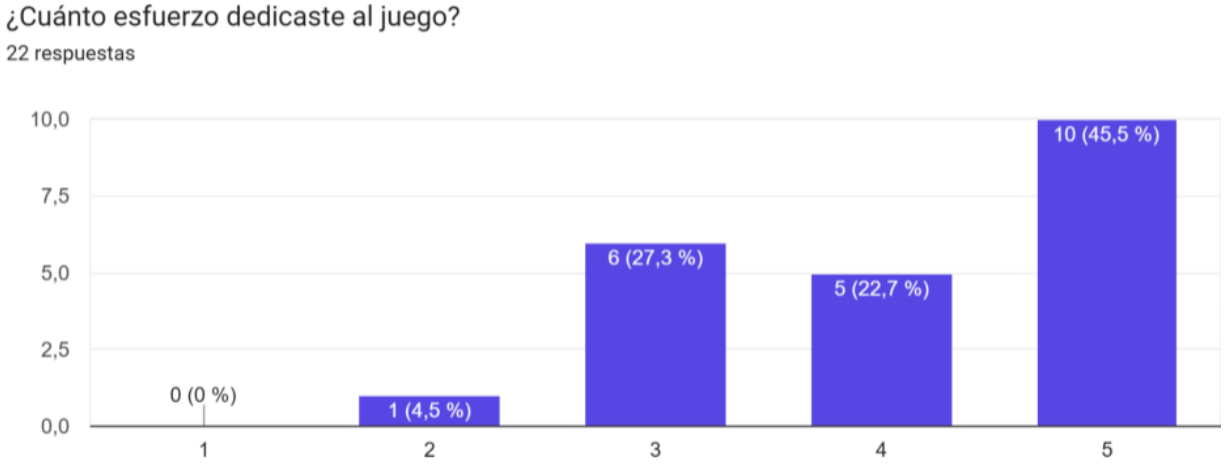


Figura A.39: Respuestas Versión por Voz – Pregunta 3.

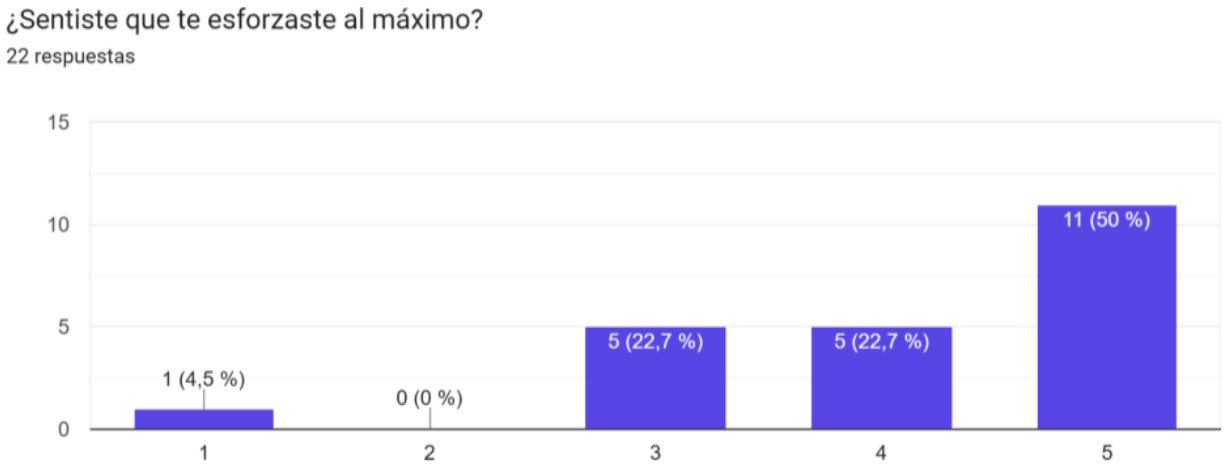


Figura A.40: Respuestas Versión por Voz – Pregunta 4.

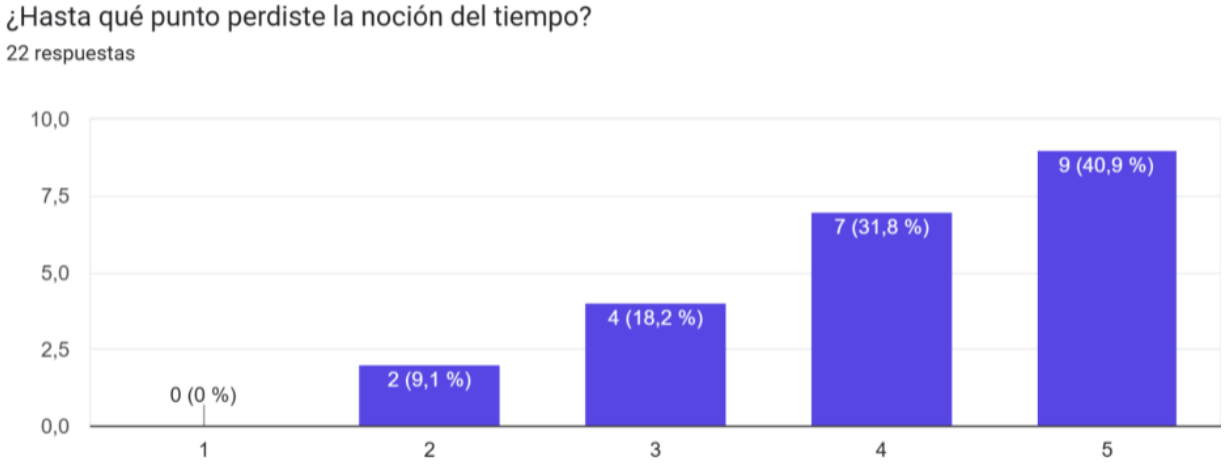


Figura A.41: Respuestas Versión por Voz – Pregunta 5.

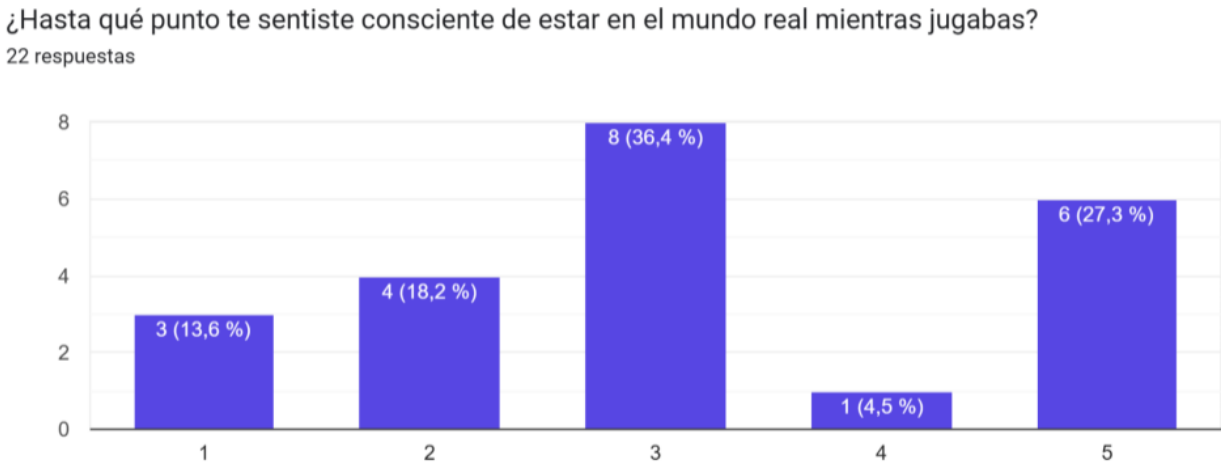


Figura A.42: Respuestas Versión por Voz – Pregunta 6.

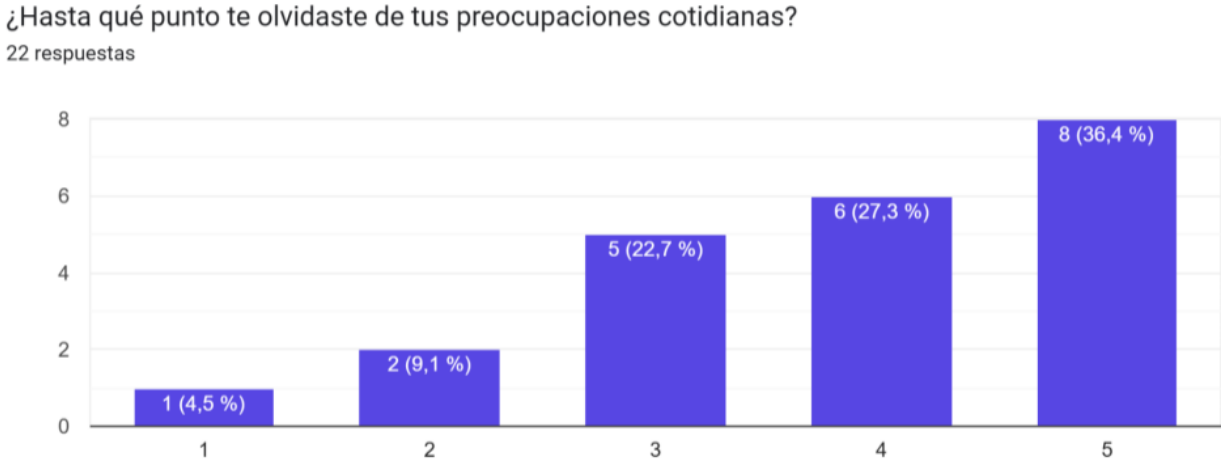


Figura A.43: Respuestas Versión por Voz – Pregunta 7.

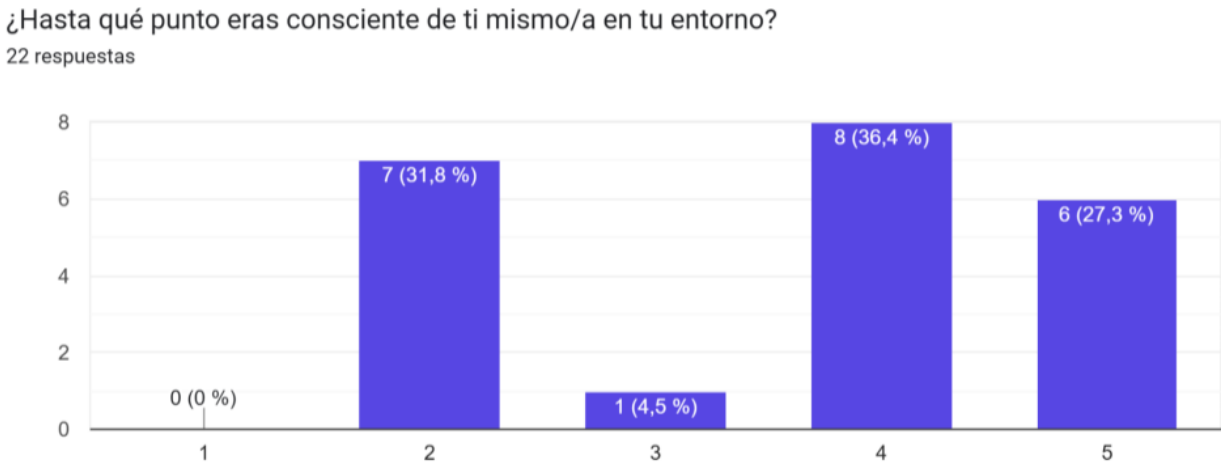


Figura A.44: Respuestas Versión por Voz – Pregunta 8.

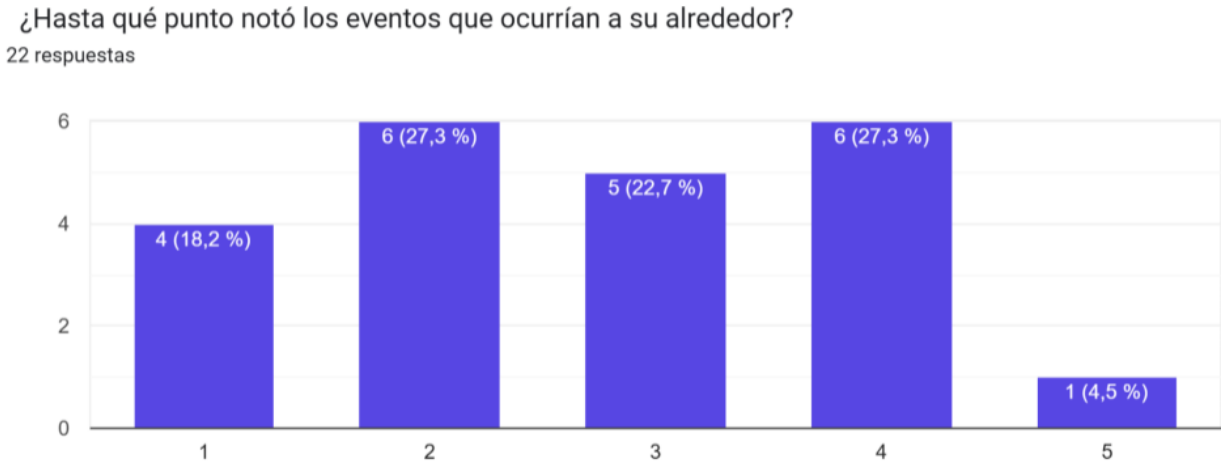


Figura A.45: Respuestas Versión por Voz – Pregunta 9.

Figura A.46: Respuestas Versión por Voz – Pregunta 10.

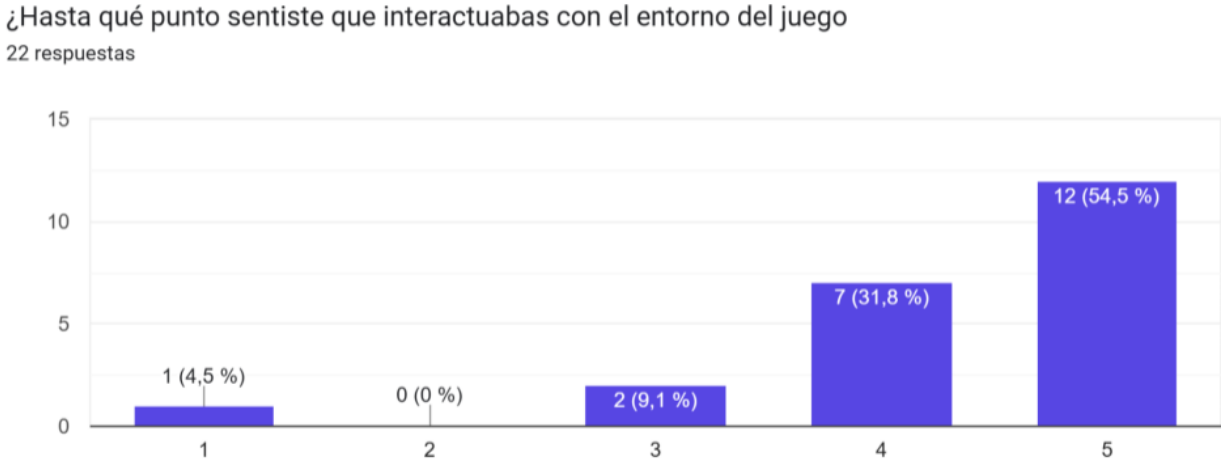


Figura A.47: Respuestas Versión por Voz – Pregunta 11.

Figura A.48: Respuestas Versión por Voz – Pregunta 12.

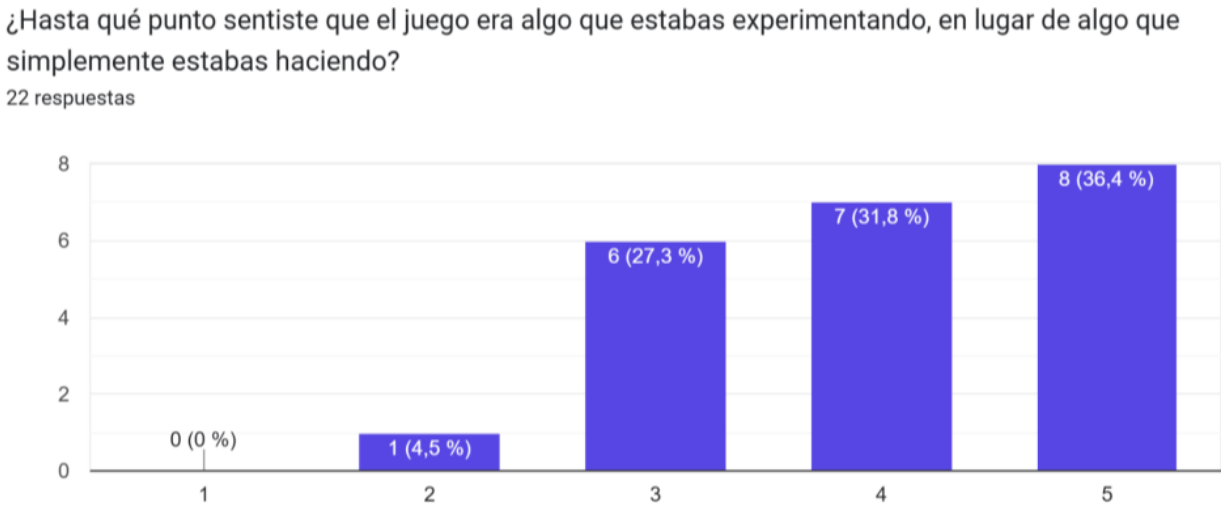


Figura A.49: Respuestas Versión por Voz – Pregunta 13.

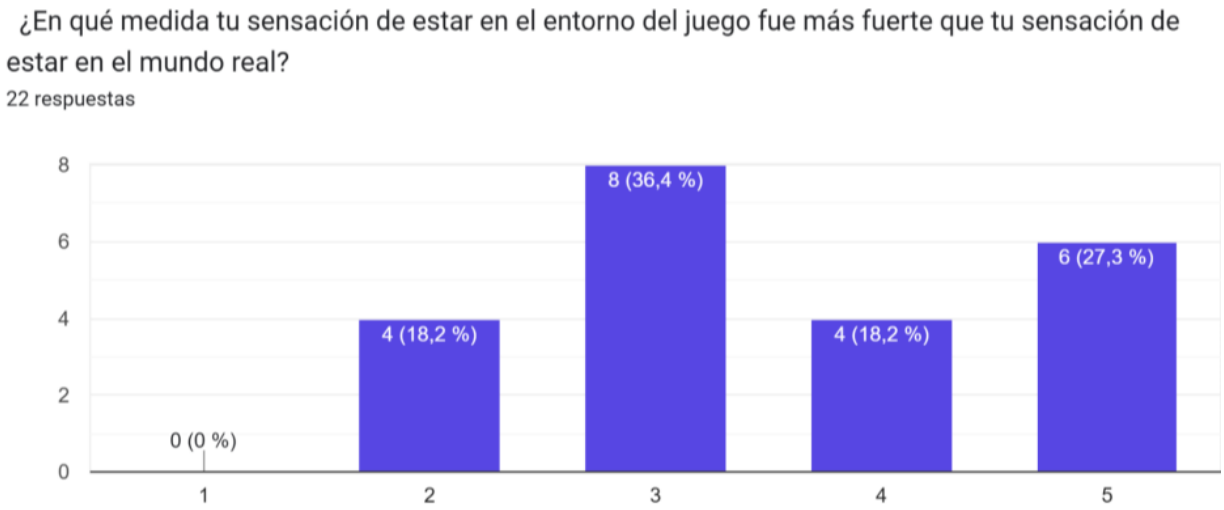


Figura A.50: Respuestas Versión por Voz – Pregunta 14.

Figura A.51: Respuestas Versión por Voz – Pregunta 15

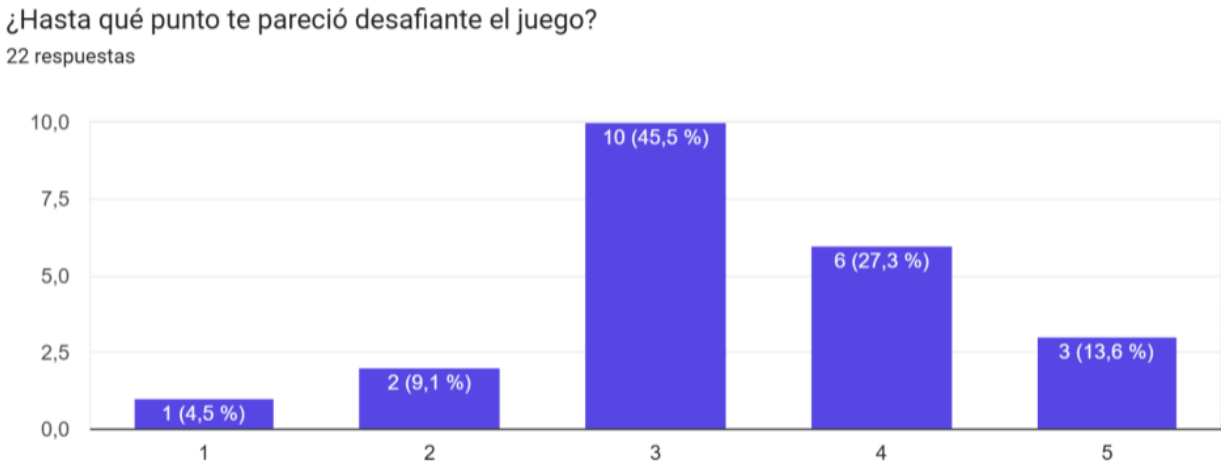


Figura A.52: Respuestas Versión por Voz – Pregunta 16

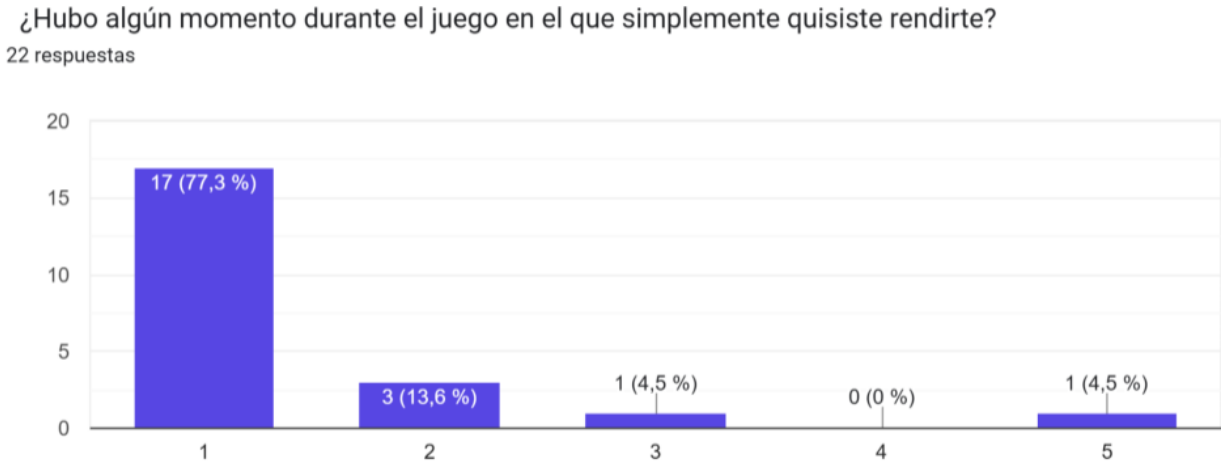


Figura A.53: Respuestas Versión por Voz – Pregunta 17.

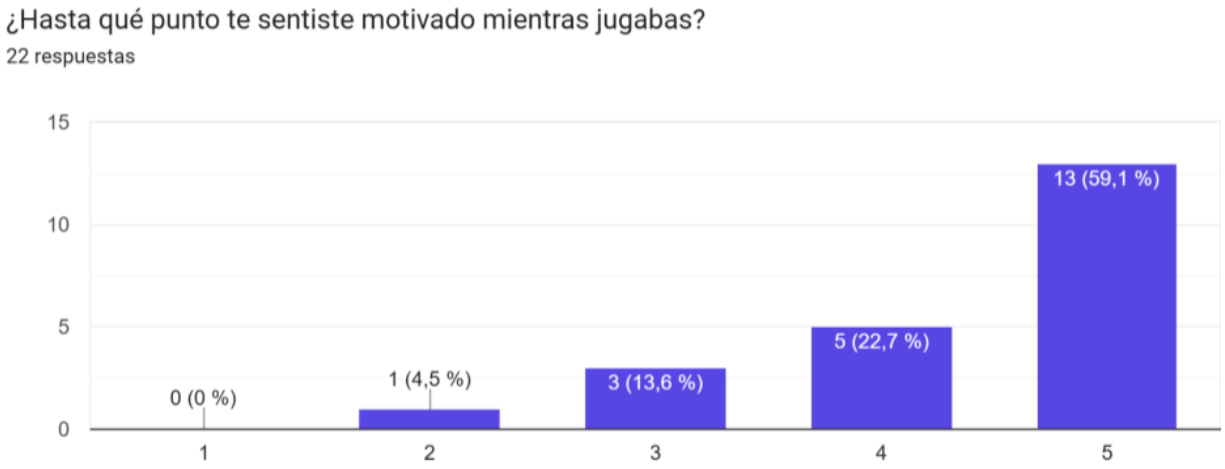


Figura A.54: Respuestas Versión por Voz – Pregunta 18.

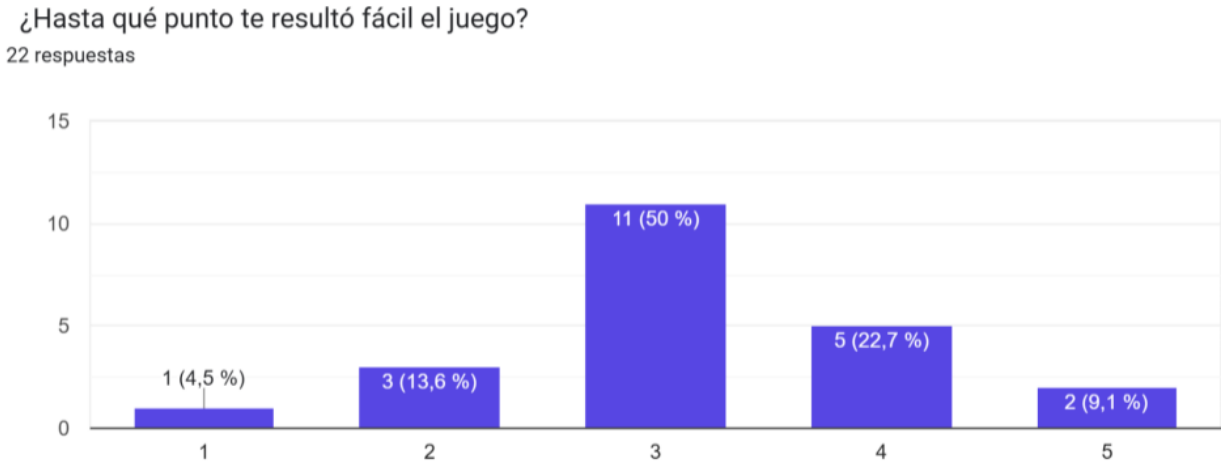


Figura A.55: Respuestas Versión por Voz – Pregunta 19.

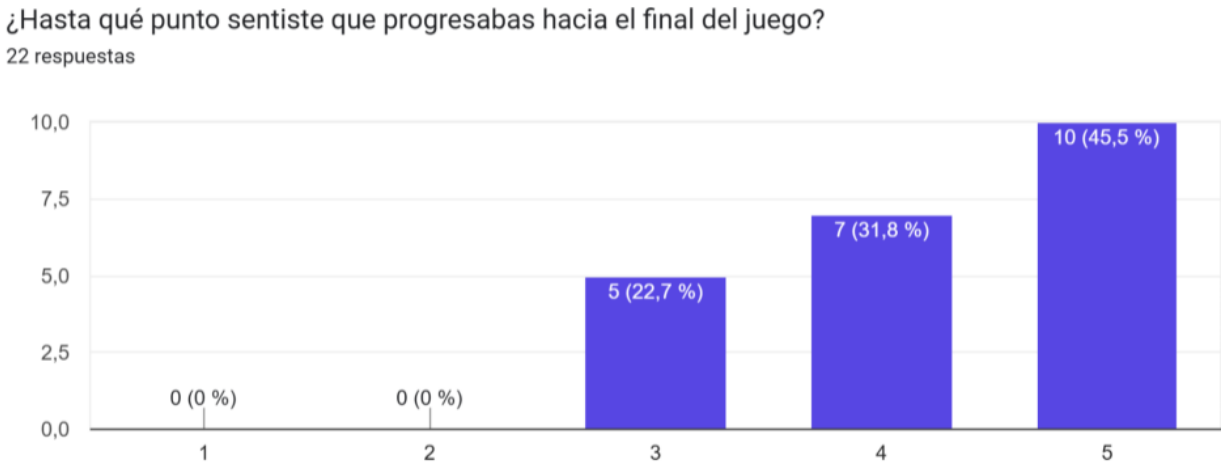


Figura A.56: Respuestas Versión por Voz – Pregunta 20.

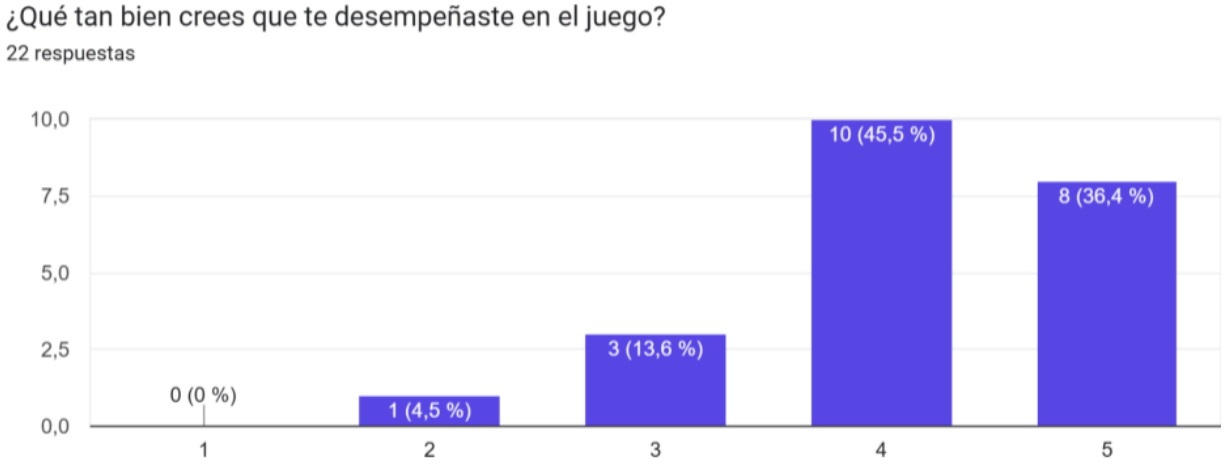


Figura A.57: Respuestas Versión por Voz – Pregunta 21.

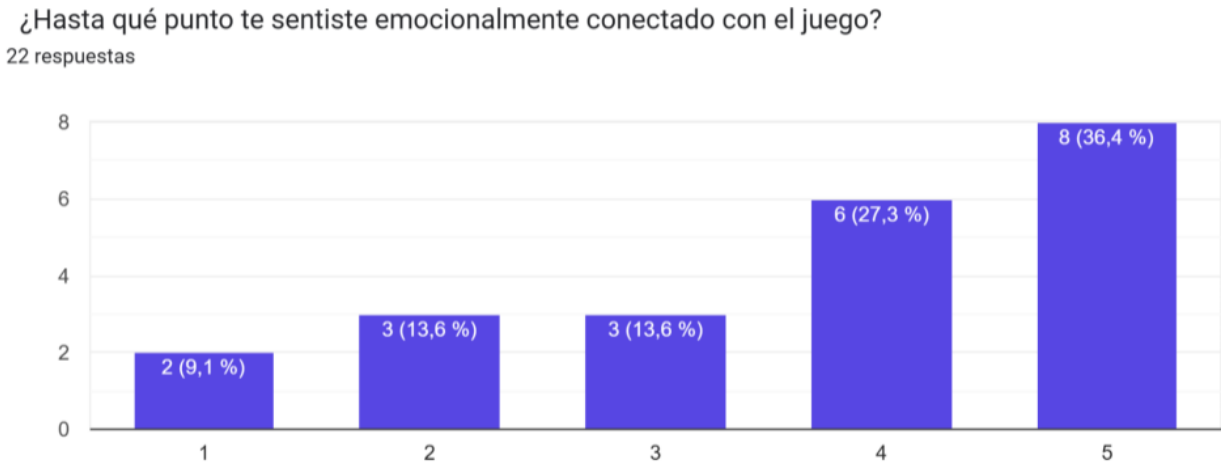


Figura A.58: Respuestas Versión por Voz – Pregunta 22.

Figura A.59: Respuestas Versión por Voz – Pregunta 23.

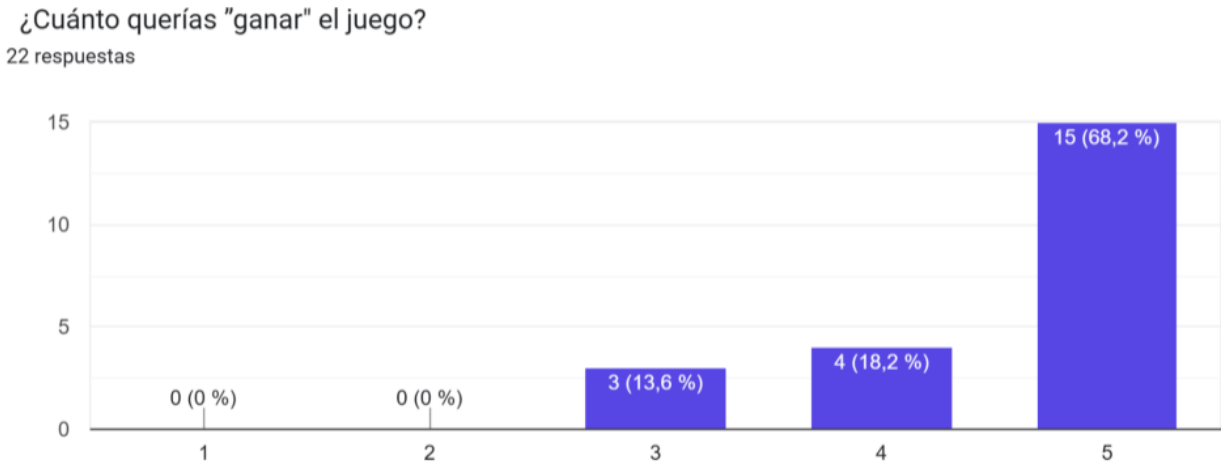


Figura A.60: Respuestas Versión por Voz – Pregunta 24.

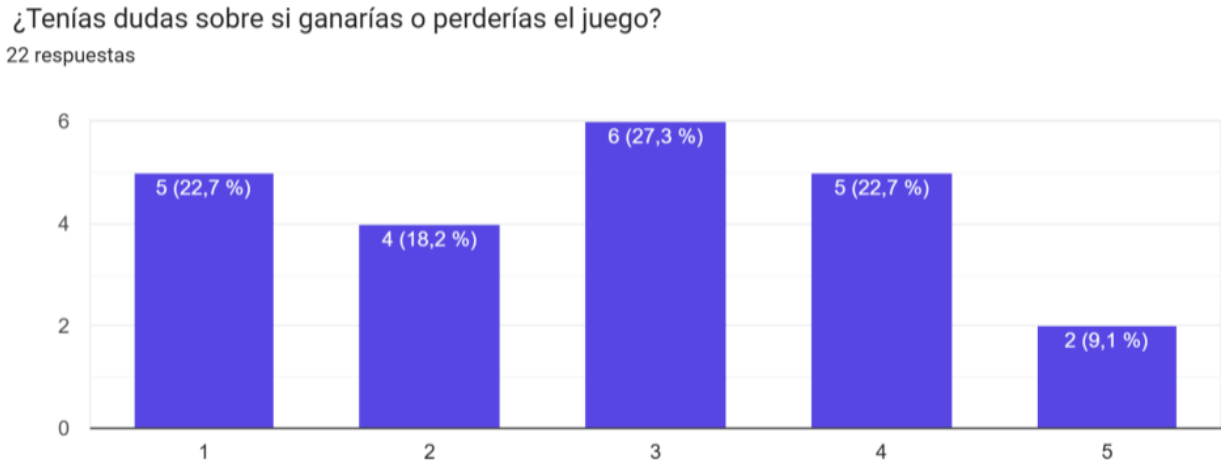


Figura A.61: Respuestas Versión por Voz – Pregunta 25.

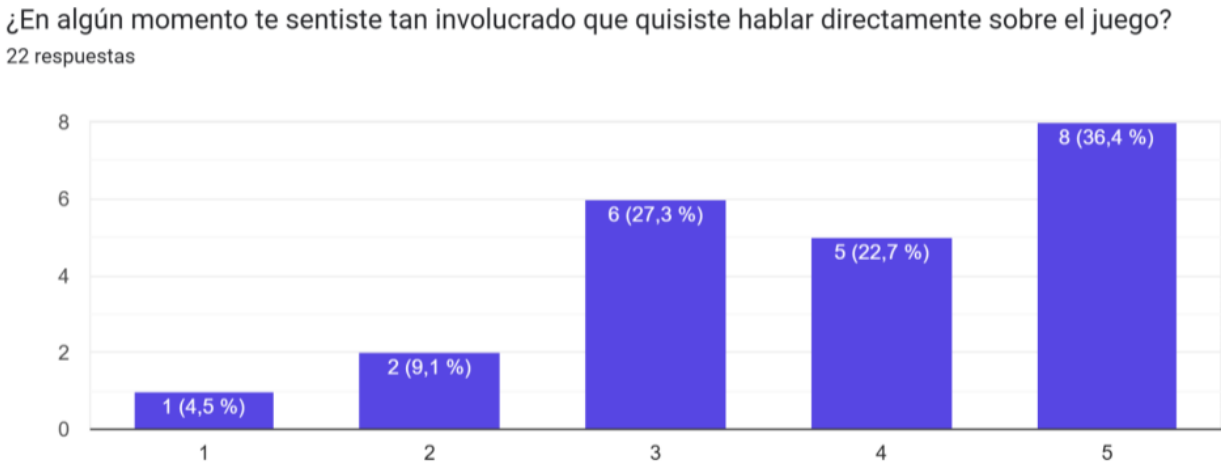


Figura A.62: Respuestas Versión por Voz – Pregunta 26.

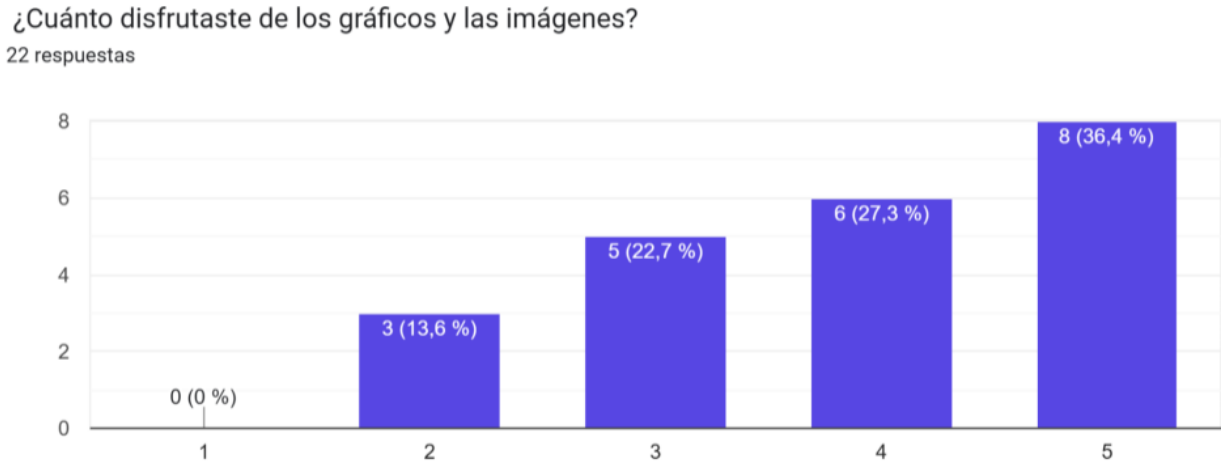


Figura A.63: Respuestas Versión por Voz – Pregunta 27.

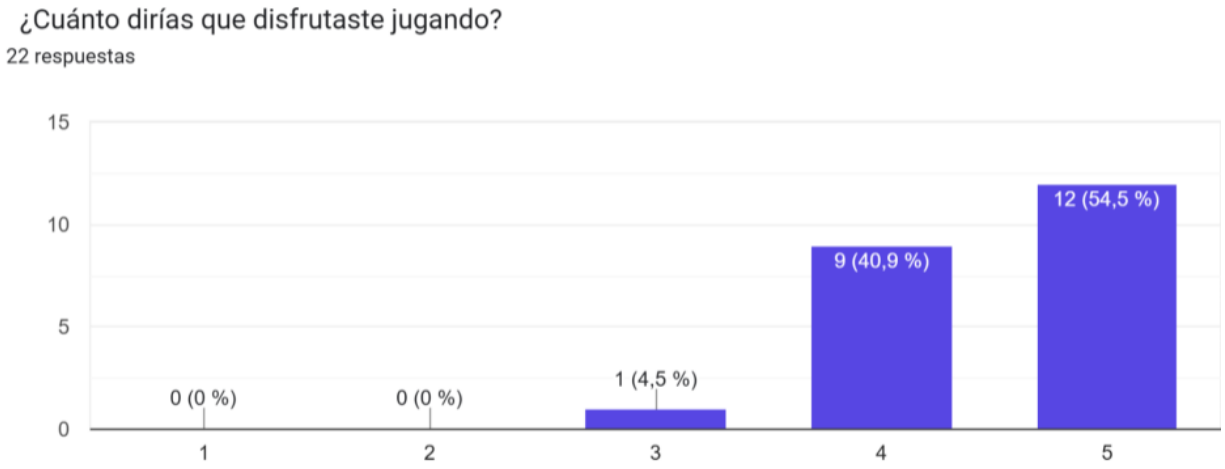


Figura A.64: Respuestas Versión por Voz – Pregunta 28.

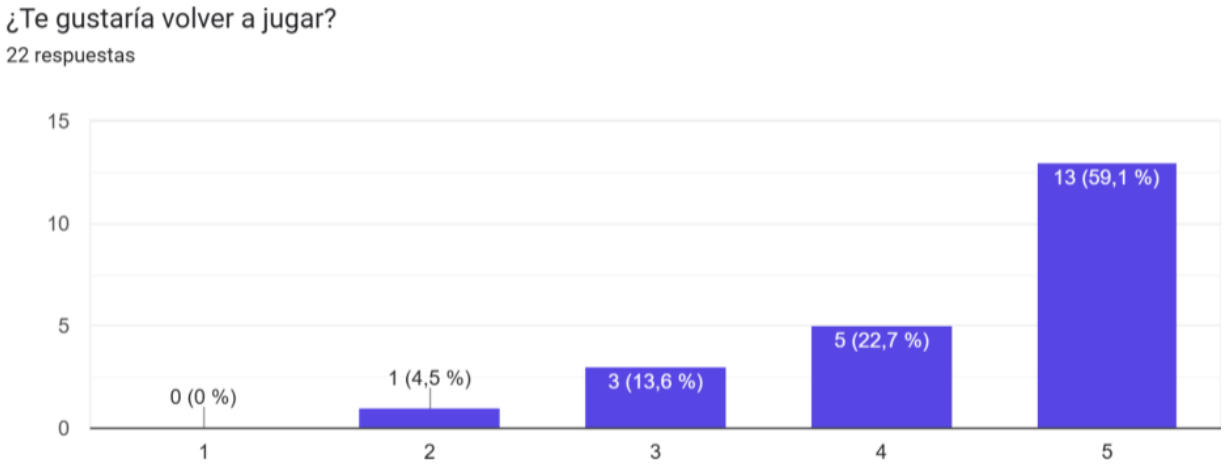


Figura A.65: Respuestas Versión por Voz – Pregunta 29.

A.10.3. Respuestas Versión Teclado y Ratón

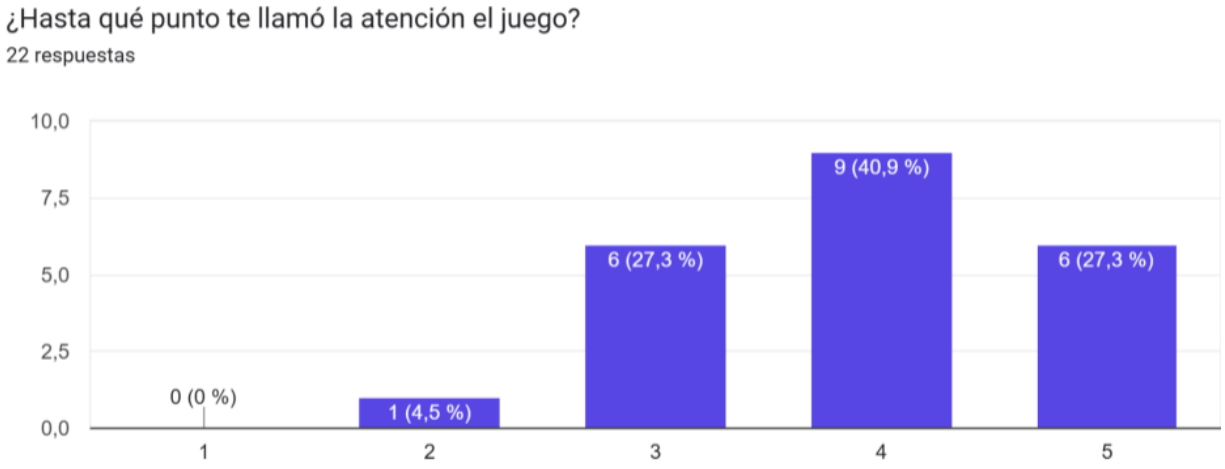


Figura A.66: Respuestas Teclado y Ratón – Pregunta 1.

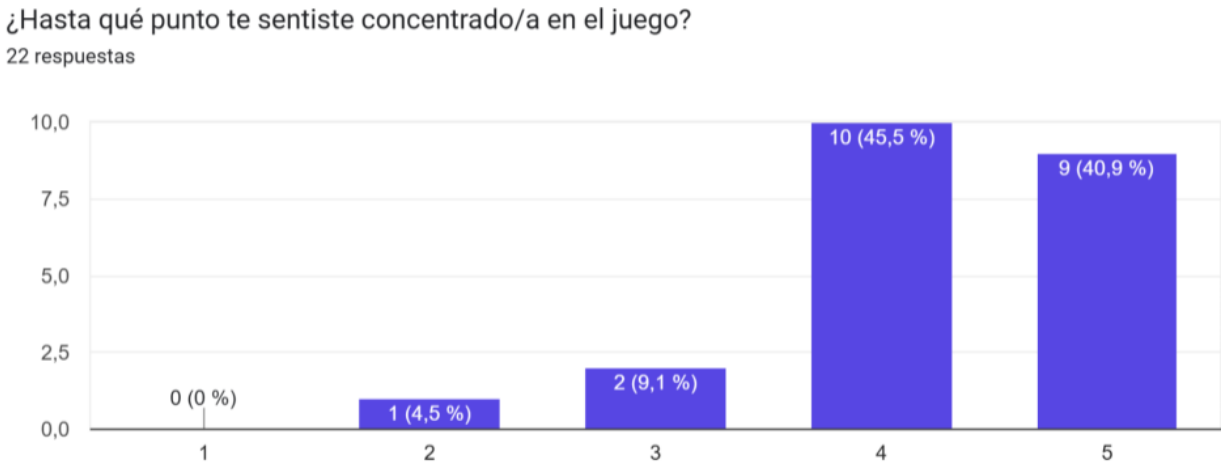


Figura A.67: Respuestas Teclado y Ratón – Pregunta 2.

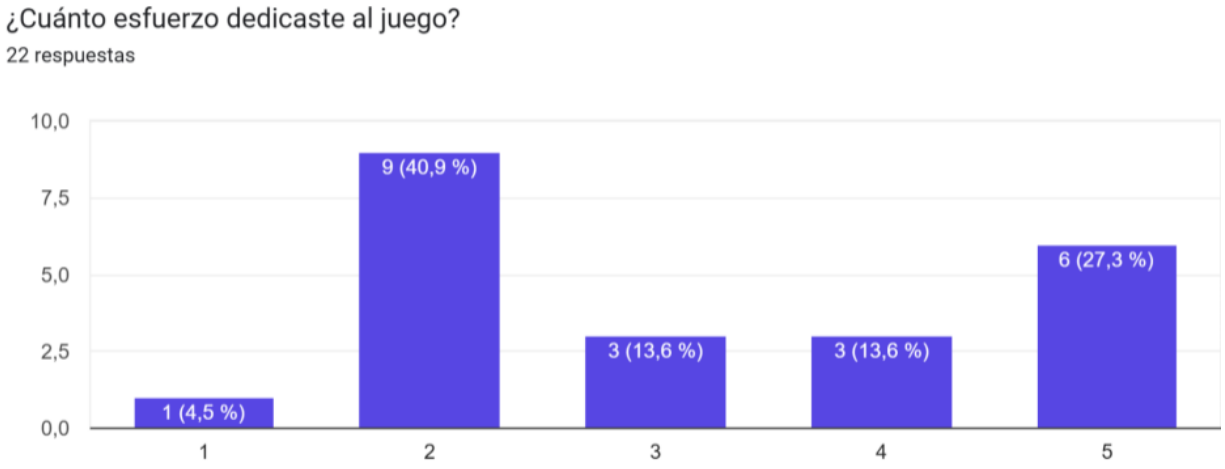


Figura A.68: Respuestas Teclado y Ratón – Pregunta 3.

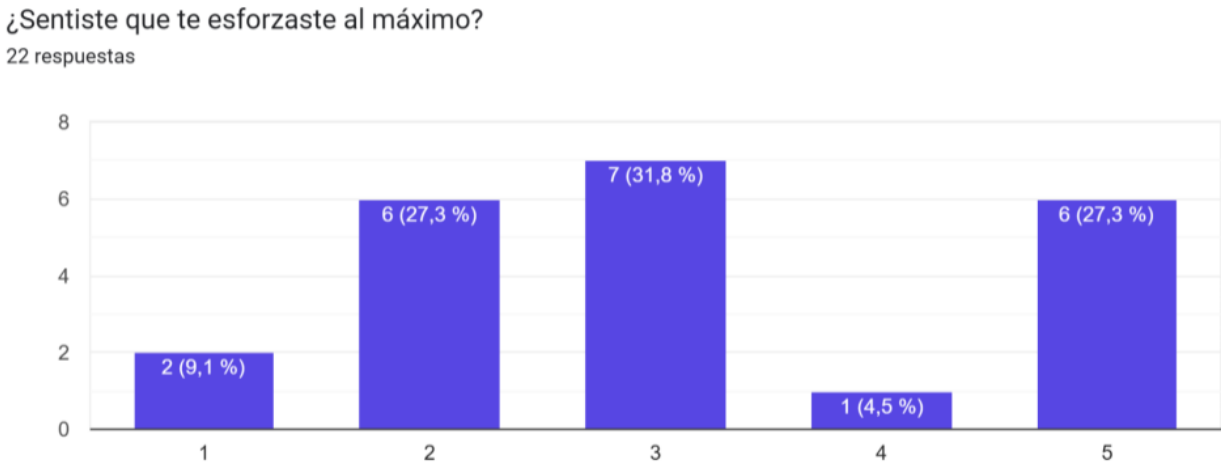


Figura A.69: Respuestas Teclado y Ratón – Pregunta 4.

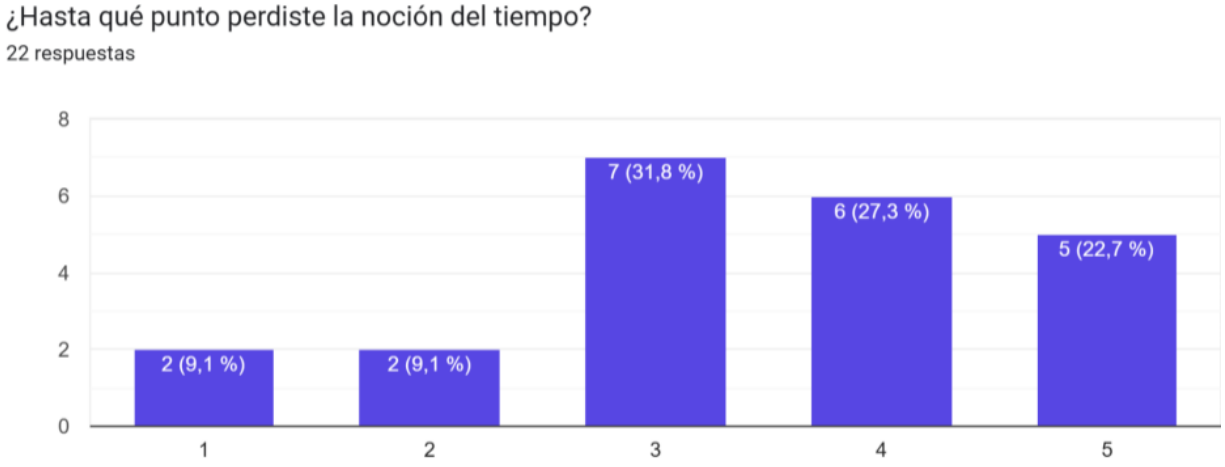


Figura A.70: Respuestas Teclado y Ratón – Pregunta 5.

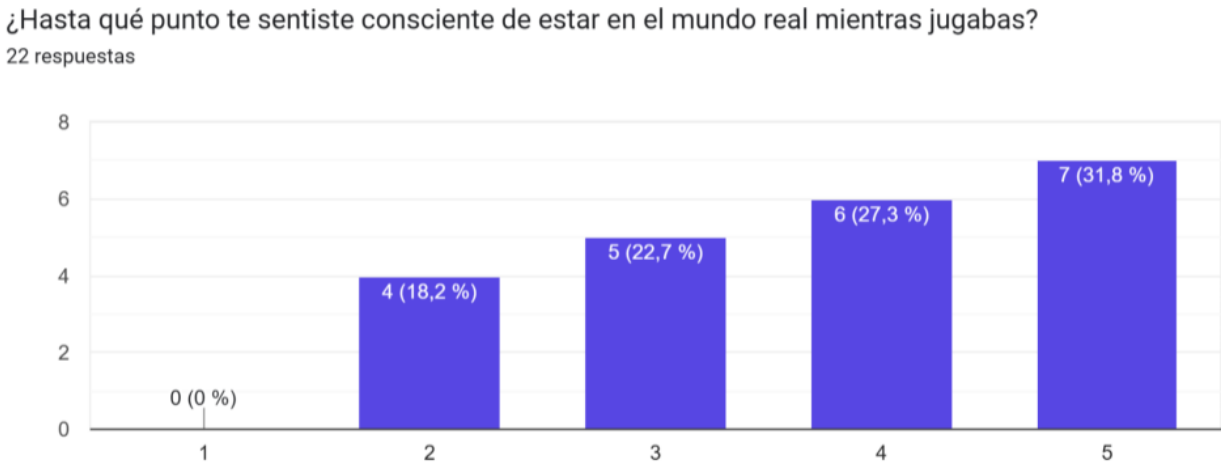


Figura A.71: Respuestas Teclado y Ratón – Pregunta 6.

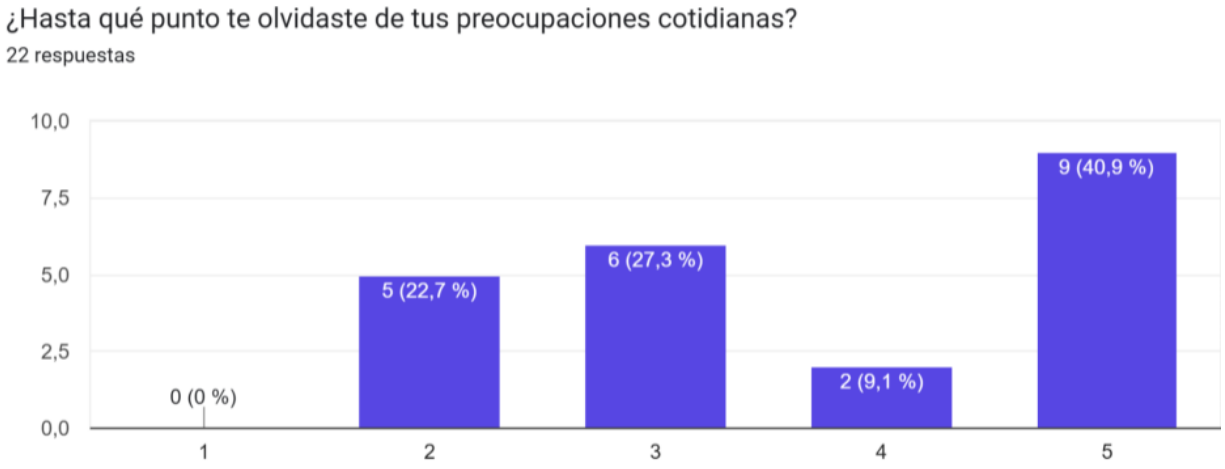


Figura A.72: Respuestas Teclado y Ratón – Pregunta 7.

Figura A.73: Respuestas Teclado y Ratón – Pregunta 8.

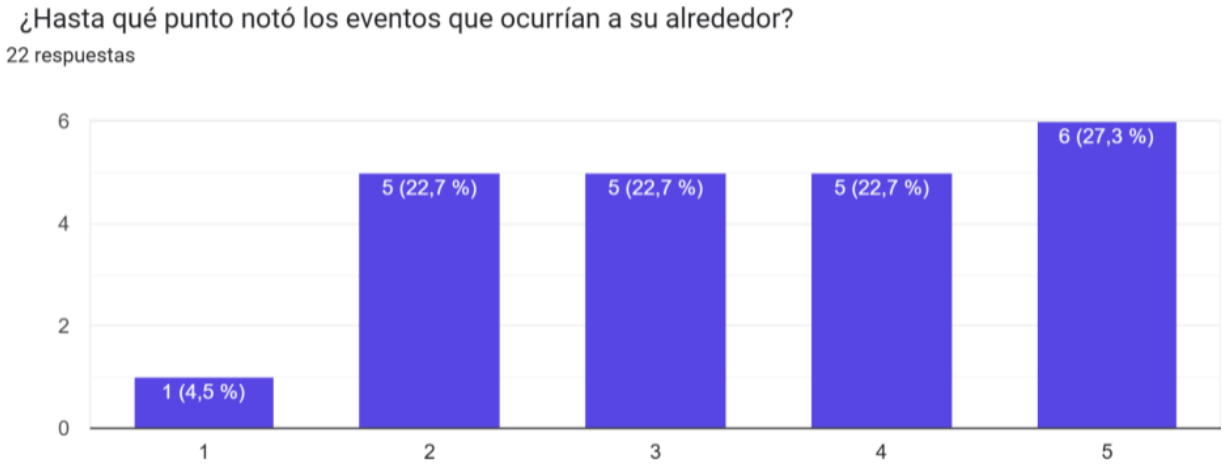


Figura A.74: Respuestas Teclado y Ratón – Pregunta 9.

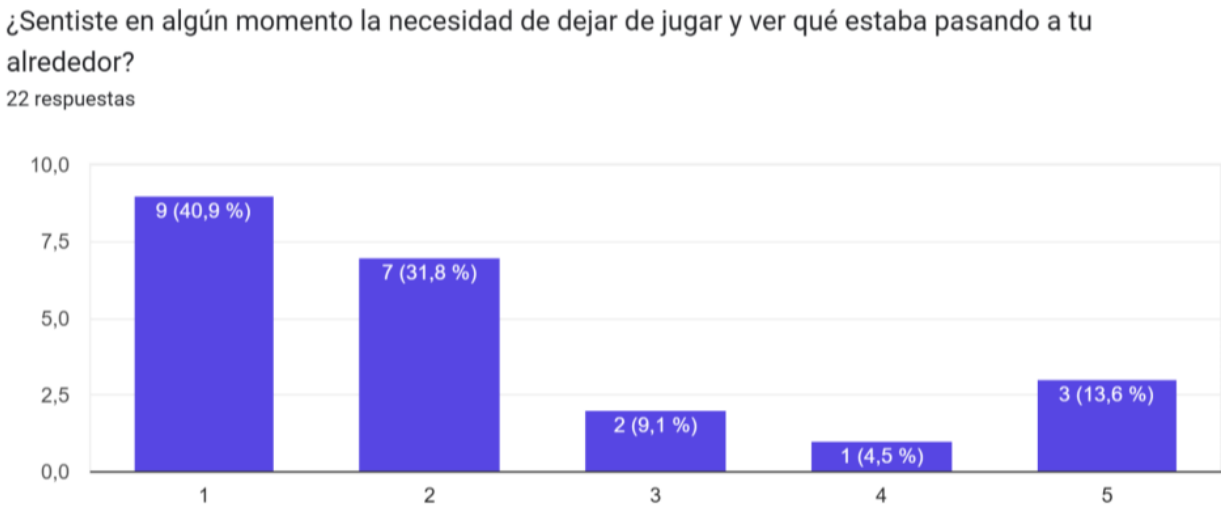


Figura A.75: Respuestas Teclado y Ratón – Pregunta 10.

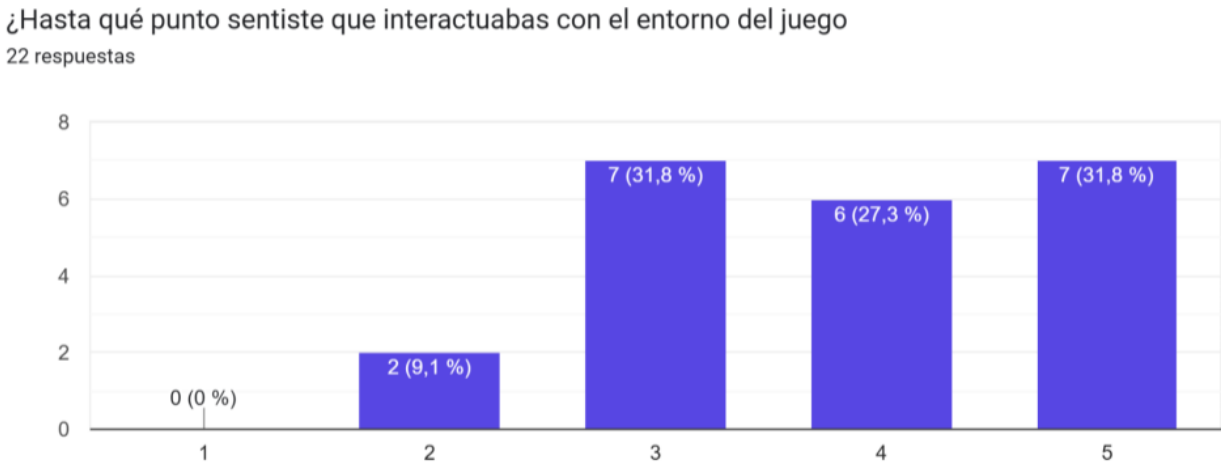


Figura A.76: Respuestas Teclado y Ratón – Pregunta 11.

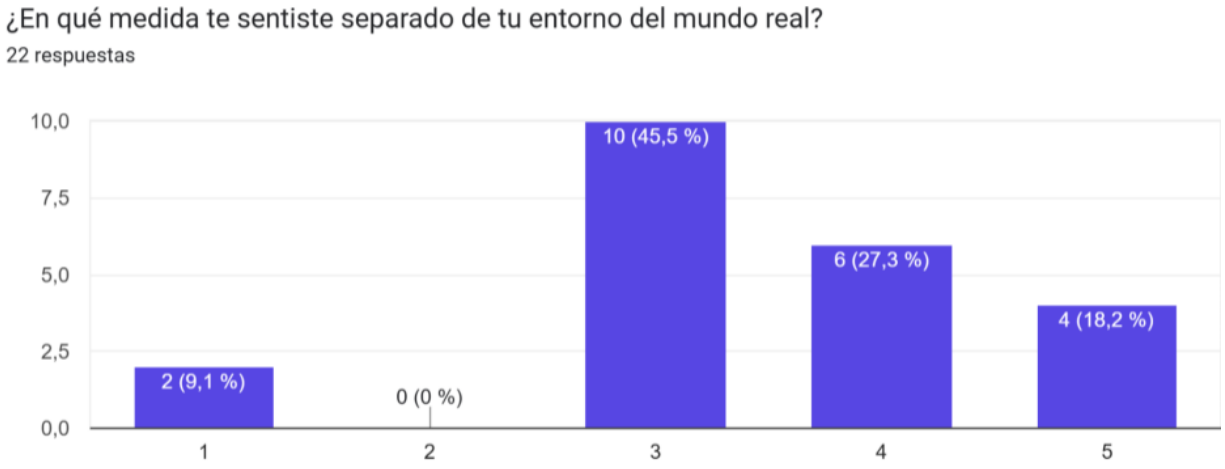


Figura A.77: Respuestas Teclado y Ratón – Pregunta 12.

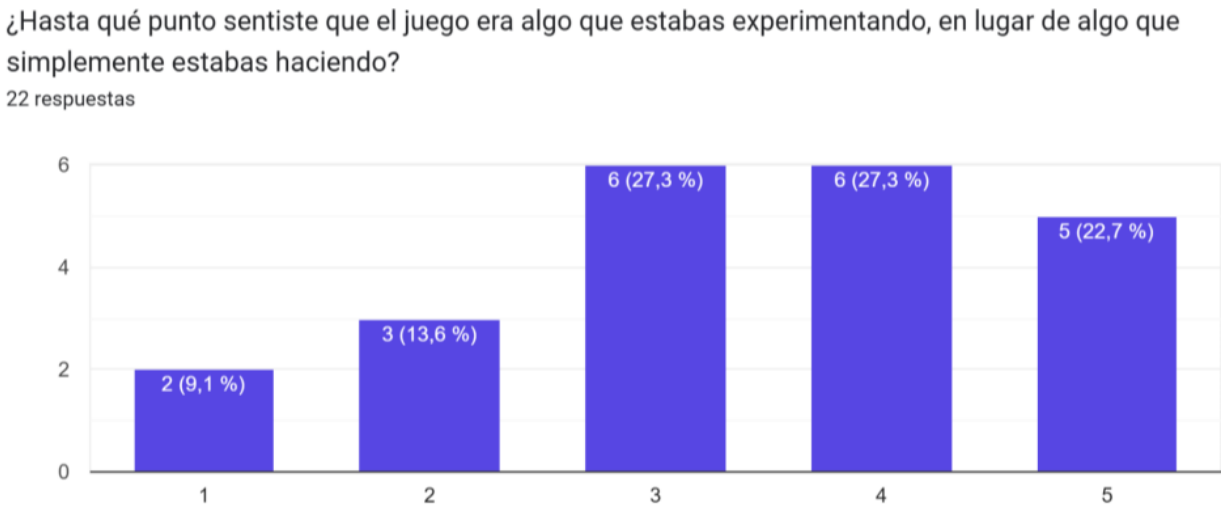


Figura A.78: Respuestas Teclado y Ratón – Pregunta 13.

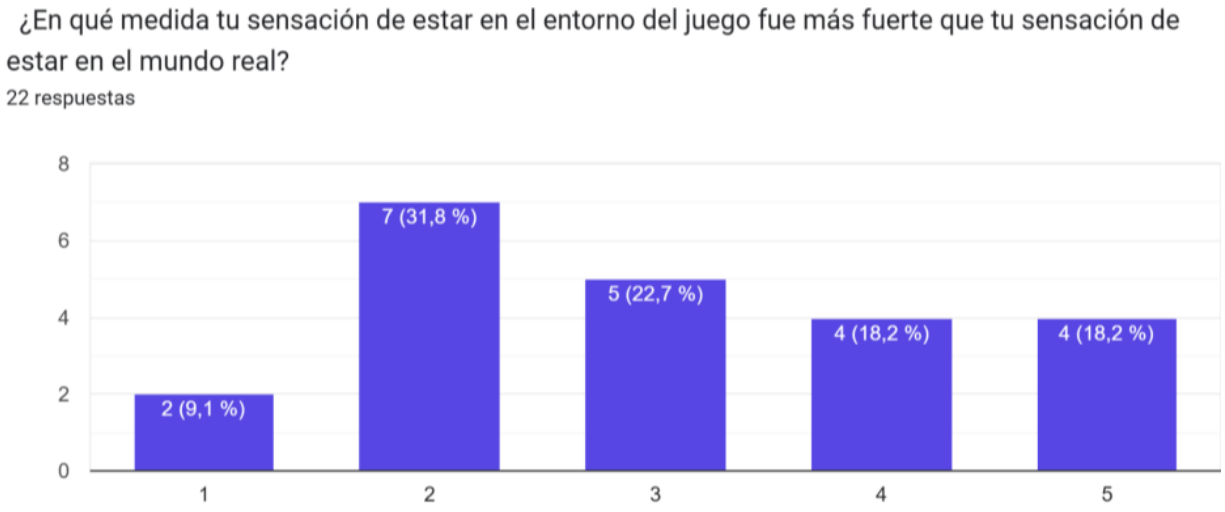


Figura A.79: Respuestas Teclado y Ratón – Pregunta 14.

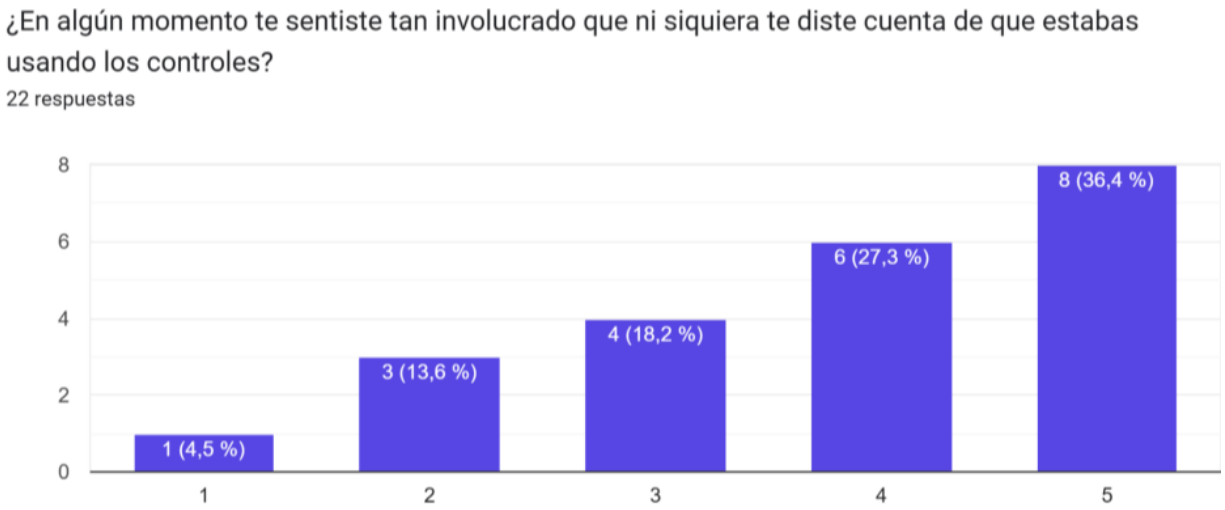


Figura A.80: Respuestas Teclado y Ratón – Pregunta 15.

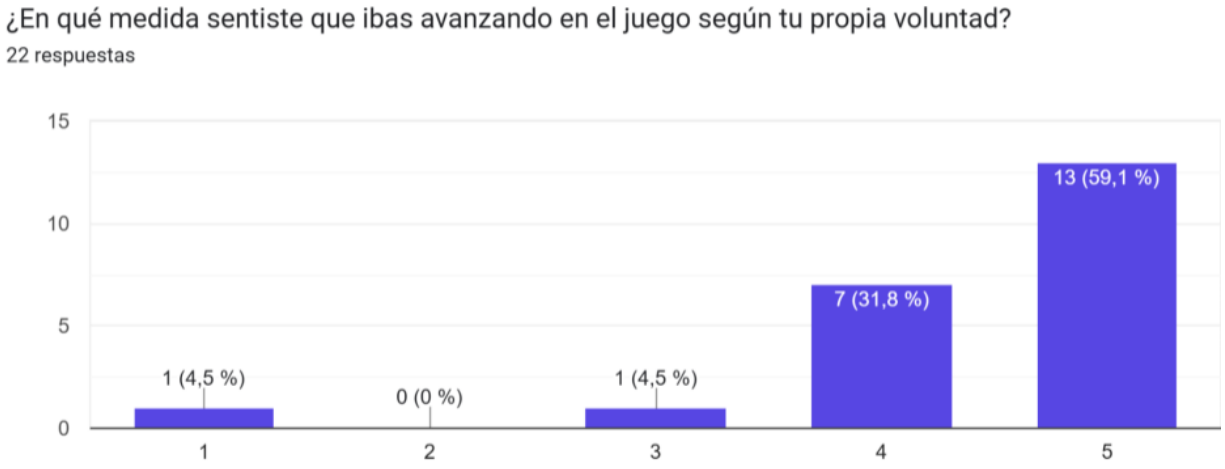


Figura A.81: Respuestas Teclado y Ratón – Pregunta 16.

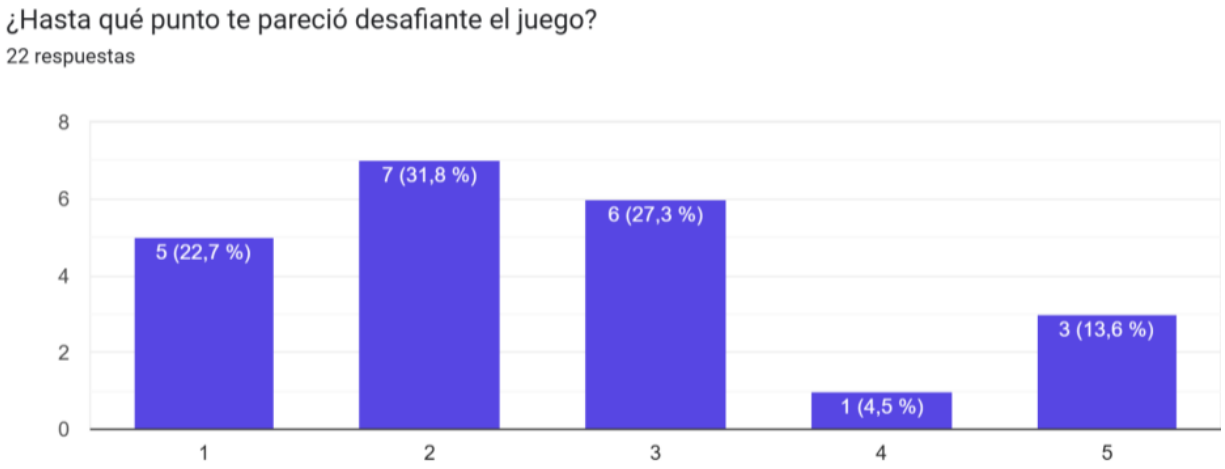


Figura A.82: Respuestas Teclado y Ratón – Pregunta 17.

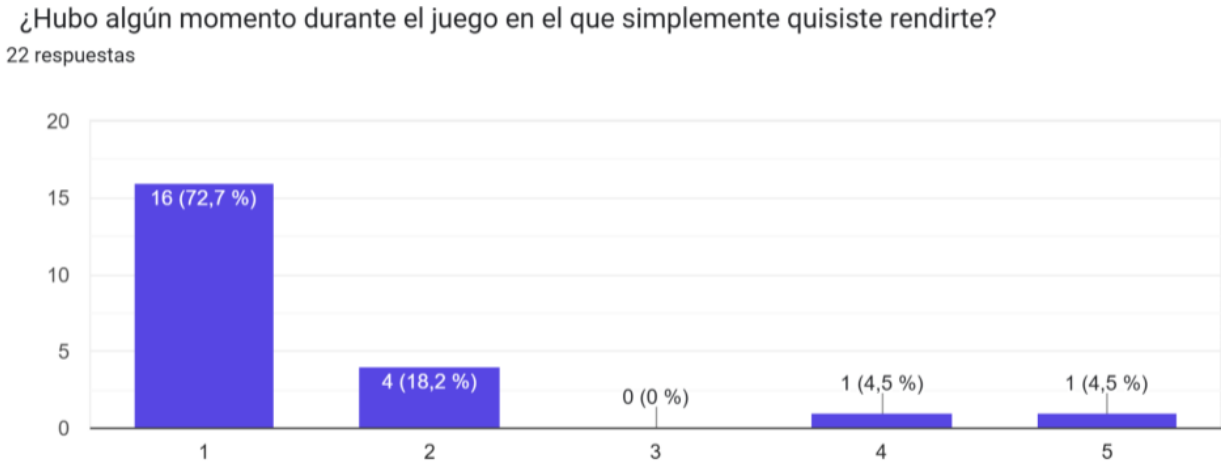


Figura A.83: Respuestas Teclado y Ratón – Pregunta 18.

Figura A.84: Respuestas Teclado y Ratón – Pregunta 19.

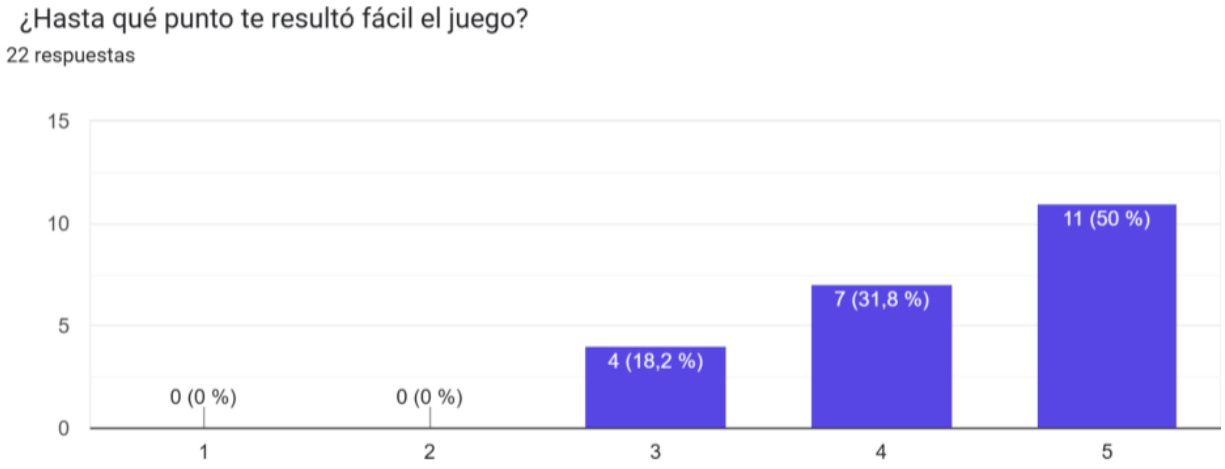


Figura A.85: Respuestas Teclado y Ratón – Pregunta 20.

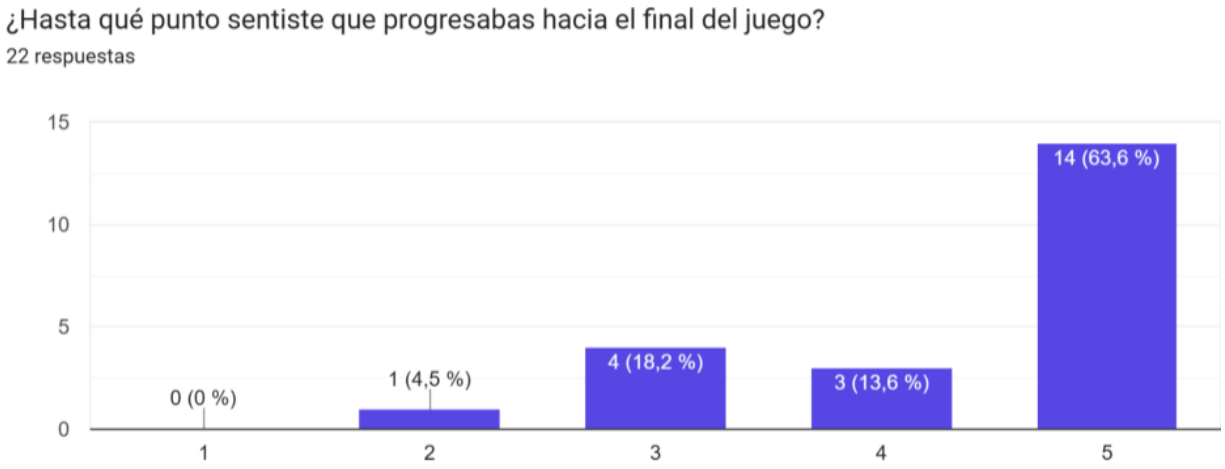


Figura A.86: Respuestas Teclado y Ratón – Pregunta 21.

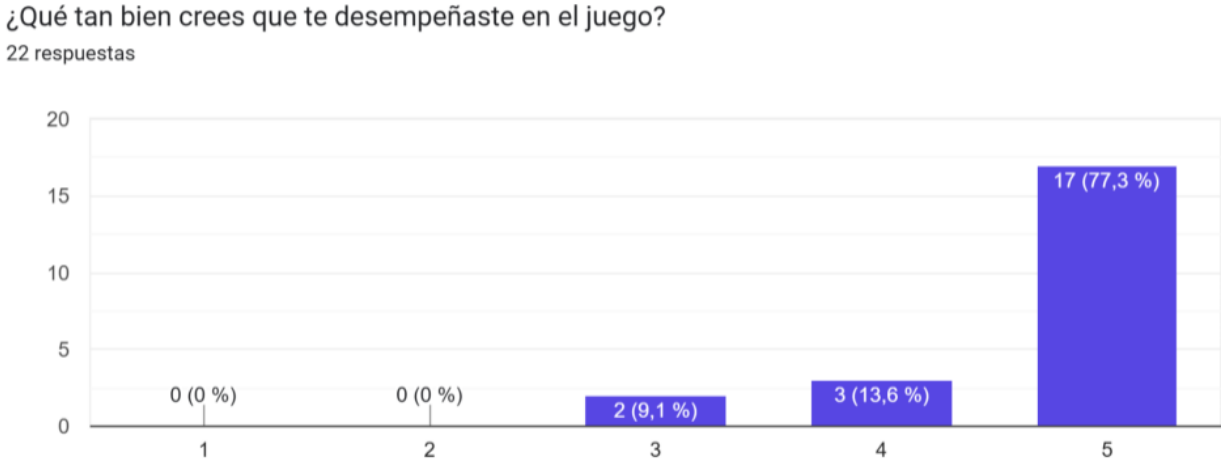


Figura A.87: Respuestas Teclado y Ratón – Pregunta 22.

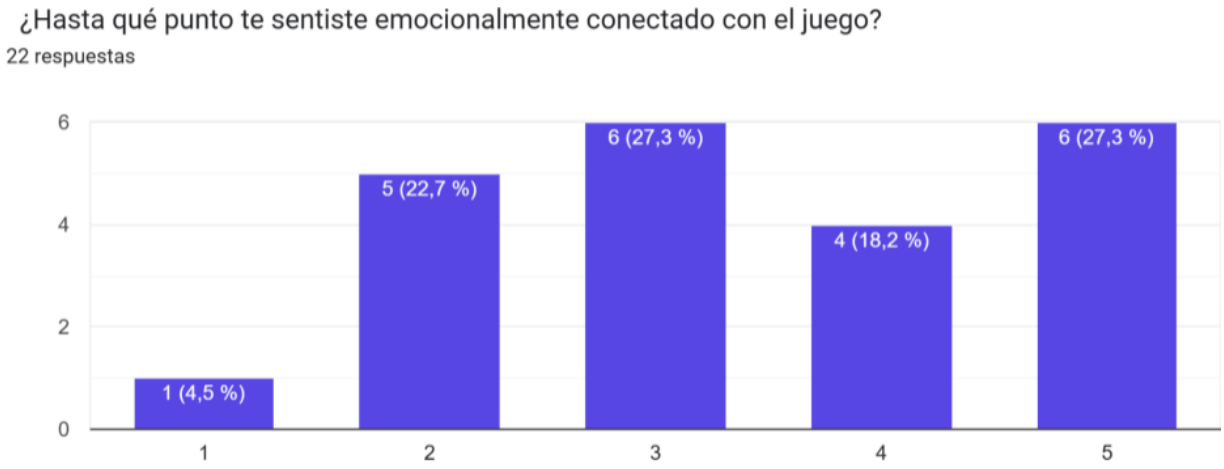


Figura A.88: Respuestas Teclado y Ratón – Pregunta 23.

Figura A.89: Respuestas Teclado y Ratón – Pregunta 24.

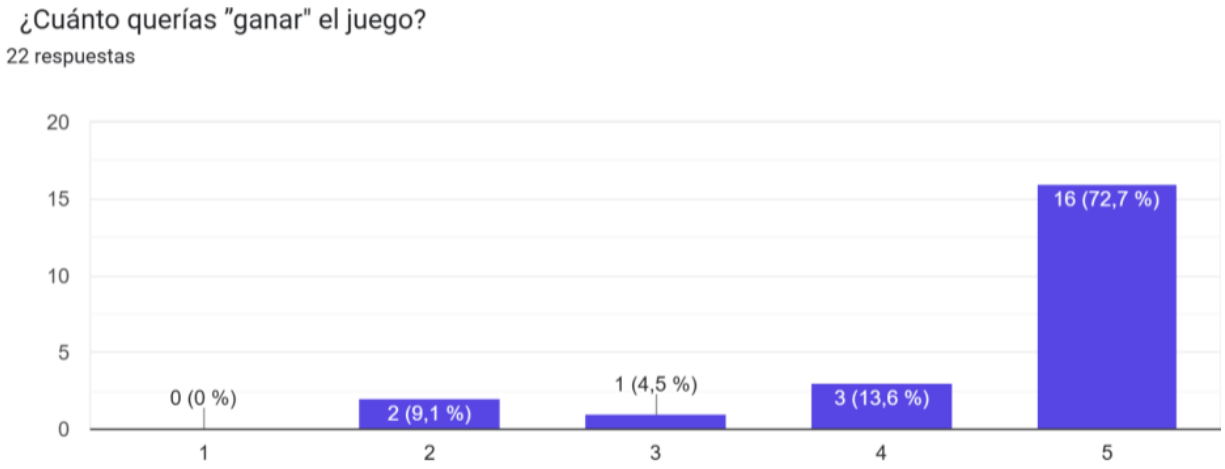


Figura A.90: Respuestas Teclado y Ratón – Pregunta 25.

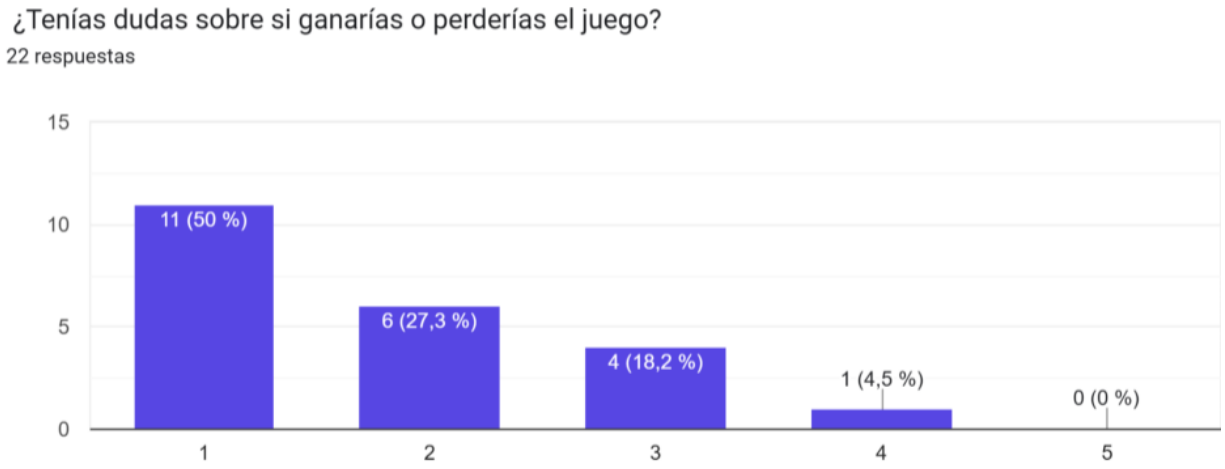


Figura A.91: Respuestas Teclado y Ratón – Pregunta 26.

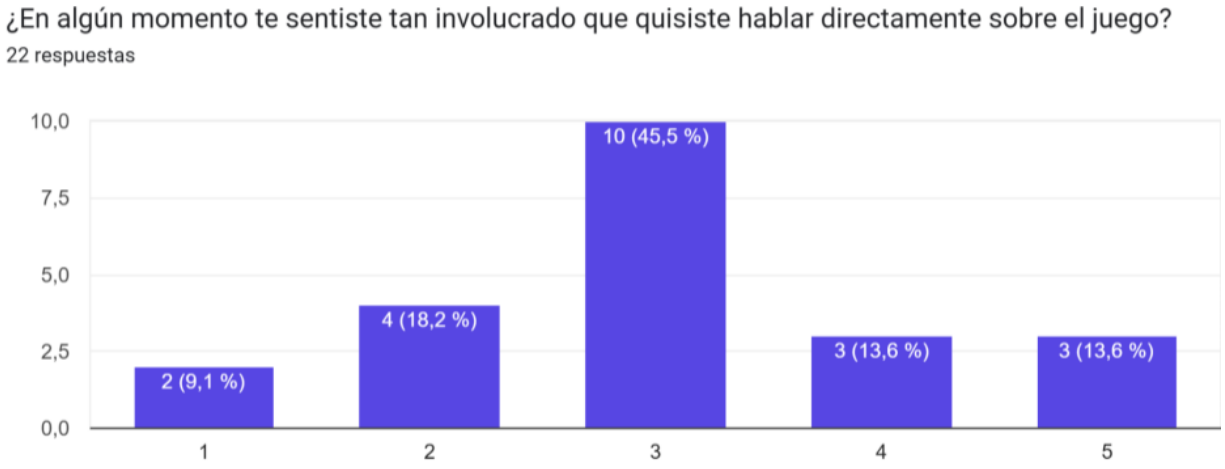


Figura A.92: Respuestas Teclado y Ratón – Pregunta 27.

Figura A.93: Respuestas Teclado y Ratón – Pregunta 28.

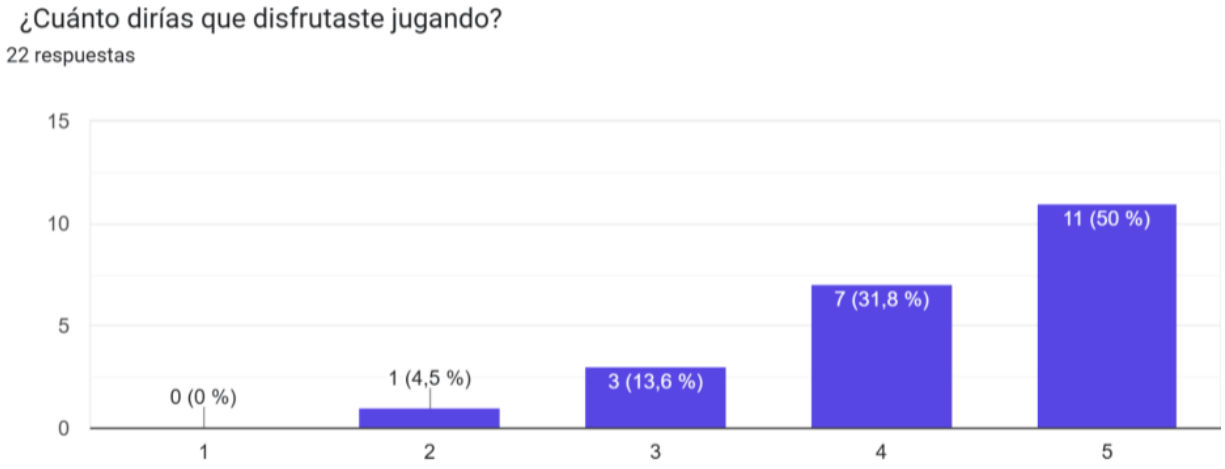


Figura A.94: Respuestas Teclado y Ratón – Pregunta 29.

Figura A.95: Respuestas Teclado y Ratón – Pregunta 30.

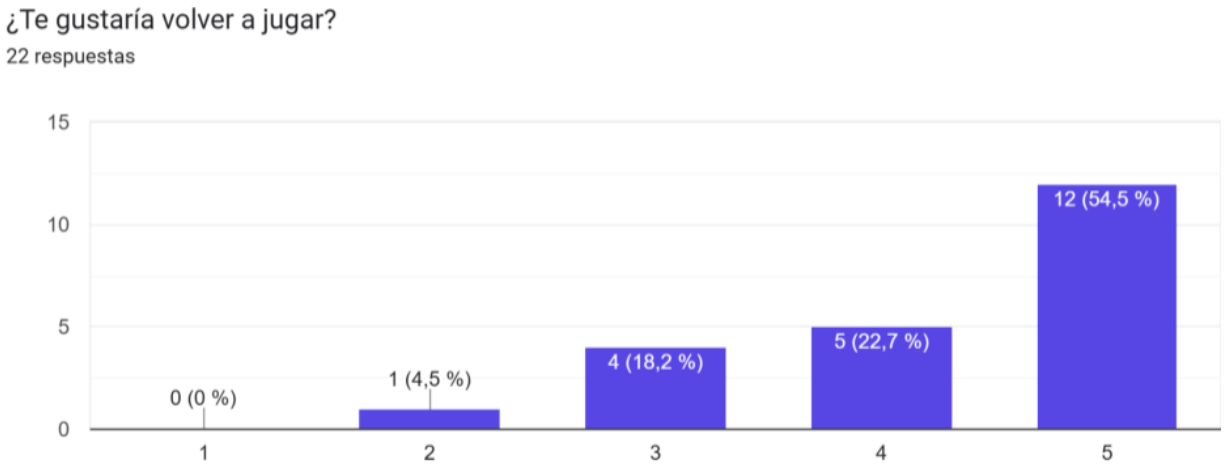


Figura A.96: Respuestas Teclado y Ratón – Pregunta 31.