Pontificia Universidad JAVERIANA Cali | Facultad de Ingeniería y Ciencias

Acta de Correcciones al Proyecto de Grado
Ingeniería de Sistemas y Computacíon

Fecha: 25/08/2023

Autores: Luis Miguel Oviedo Lutkens

Nombre del Proyecto de Grado: Prototipo de un sistema de evaluación del desarrollo cognitivo en niños a través de un videojuego tipo laberinto.

Director: Andrés Navarro Newball

Como indica el artículo 2.27 de las Directrices de Trabajo de Grado, he verificado que los estudiantes indicados arriba han implementado todas las correcciones que los Jurados del Proyecto de Grado definieron que se efectuarán, como consta en el Acta de Calificación correspondiente.

Firma de Director(a) del Proyecto de Grado

Nota de Aceptación

Aprobado por el Comité de Trabajo de Grado en cumplimiento de los requisitos exigidos por la Pontificia Universidad Javeriana para optar el título de Ingeniero de Sistemas y Computación.

Dr. CAMILO ROCHA
Decano de la Facultad de Ingeniería

ING. GERARDO MAURICIO SARRIA
Director Carrera Ingeniería Sistemas y Computación.

ING. ANDRÉS ADOLFO NAVARRO NEWBALL
Director(a) Trabajo

ING. JUAN CARLOS MARTÍNEZ ARIAS
Jurado 1

ING. JUAN PABLO GARCÍA CIFUENTES
Jurado 2

Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias.
Ingeniería de Sistemas y ciencias de la Computación.
Proyecto de Grado.

Prototipo de un sistema de evaluación del desarrollo cognitivo en niños a través de un videojuego tipo laberinto

Luis Miguel Oviedo Lutkens

Director: Dr. Andrés Adolfo Navarro Newball

Lunes 17 de Julio del 2023

Abstract

The extensive deforestation of native forests due to so-called "progress"by humans displaces wildlife and destroys flora, the habitat of endangered species such as the Spectacled Bear and the Cotton-top Tamarin Monkey. Despite various efforts to conserve these ecosystems, the main factor contributing to species loss is the lack of knowledge about them. However, environmental education available for individuals with visual or hearing impairments can be insufficiently inclusive or limited. Children with different visual and hearing disabilities can develop impairments in their thinking, speech, language, behavior, social development, emotional well-being, and ultimately affect their academic and occupational performance from early stages of life. Therefore, this project took the initiative to create a maze-type video game that allows children with visual and hearing impairments to learn about nature interactively. Through this video game, the aim is to create a positive impact in the lives of children, enabling them to learn through inclusive interaction, raise awareness about the conservation of Colombian fauna and flora, and contribute to the assessment of children's cognitive and social development through technology.

Key Words: visual disability, hearing disability, environmental education, video game, maze.

Resumen

La deforestación extensiva de bosque nativo a causa del llamado “progreso” del hombre desplaza la fauna y destruye la flora, hábitat de especies en vía de extinción cómo el Oso de Anteojos y el Mono Tití Cabeciblanco. A pesar de los diferentes esfuerzos para conservar estos ecosistemas de vida el principal factor que contribuye a la pérdida de las especies es el desconocimiento de las mismas. No obstante, la educación ambiental disponible para las personas con discapacidades visuales o auditivas puede llegar a ser poco inclusiva o limitada. Los niños y niñas con diferentes discapacidades visuales y auditivas desde etapas tempranas de la vida pueden desarrollar afectaciones en su pensamiento, habla, lenguaje, conducta, desarrollo social, emocional y finalmente afectar su desempeño escolar y laboral. De modo que por medio de este proyecto se tomó la iniciativa de crear un videojuego de tipo laberinto que permita a los niños con discapacidades visuales y auditivas aprender de la naturaleza de manera interactiva. Por medio de este videojuego se busca crear un impacto positivo en la vida de los niños de modo que puedan aprender por medio de una interacción inclusiva, generar conciencia sobre el cuidado de la fauna y flora colombiana y contribuir a la evaluación del desarrollo cognitivo y social de los niños por medio de la tecnología.

Palabras Clave: Discapacidad visual, discapacidad auditiva, educación ambiental, videojuego, laberinto.

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Planteamiento del Problema

Colombia es un país mega diverso que contiene gran parte de las fuentes hídricas, las selvas y los bosques del mundo. En 2019, el país ocupa el primer lugar en diversidad de especies de aves y orquídeas y el segundo lugar con mayor riqueza de plantas, anfibios, mariposas y peces de agua dulce [1]. Varias de estas especies se encuentran declaradas en vía de extinción, entre las cuales se encuentran el Oso de Anteojos y el Mono Tití Cabeciblanco. Especies que se han visto afectadas y desplazadas por el constante y desmesurado crecimiento del "progreso"humano. Una de las principales causas que pone en peligro la supervivencia de estas es la pérdida de su hábitat natural y la fragmentación de los bosques. La ampliación de los límites agropecuarios, la siembra de cultivos ilícitos, la ganadería extensiva, la tala y la minería, tanto legal como ilegal, los incendios forestales y el crecimiento de las fronteras urbanas contribuyen a la pérdida del hábitat de estas especies [2].

Otro aspecto importante a tener en cuenta es el cambio climático ya que se ha demostrado que algunas especies cambian sus recorridos y habitas solo con presentarse unos cambios mínimos en la temperatura del ambiente, lo que no asegura la completa adaptación y supervivencia. En estos casos algunos grupos o individuos deben ser trasladadas a zonas más estables y adecuadas para asegurar su supervivencia [3]. Por otra parte, la educación ambiental en los sectores tanto urbanos como rurales y cercanos al hábitat de estas especies es de vital importancia para permitir la conservación y el actuar consciente del ser humano frente a los recursos dados por la naturaleza.

La educación ambiental (EA) no defiende opiniones ni procedimientos particulares, en cambio, les enseña a los individuos a sopesar los distintos lados de una problemática ambiental mediante el pensamiento crítico y estimula sus propias habilidades para resolver problemas y tomar decisiones, de [4]. Con esto en mente, la EA se ha convertido en un campo interdisciplinario que constantemente se está desarrollando y formulando iniciativas, políticas y proyectos con el fin de proteger y preservar la naturaleza, además de ser un tema que nos compete a todos ya que compartimos el mismo planeta. Por medio del contacto directo con la naturaleza es como podemos llegar a aprender, apreciar y sensibilizarnos sobre el verdadero valor de esta, pero también es claro que no todas las personas cuentan con las mismas capacidades o medios para conocer directamente las especies en vía de extinción en su hábitat natural o existe alguna discapacidad que lo impide.

También es importante recalcar que el proceso educativo de cada persona es distinto por el hecho de que nuestros sentidos pueden percibir el mundo de diferentes maneras o porque se sufre de alguna discapacidad o limitación que no permite la completa interacción con el entorno. En el Boletín de poblaciones de personas discapacitadas del semestre 2020-I el Ministerio de Salud estimaba que a

agosto de 2020 1,3 millones de personas presentaba alguna discapacidad. Definiendo la condición de discapacidad como las afectaciones en estructuras corporales, así como las limitaciones para realizar alguna tarea o restricciones en la participación, de [5]. De este modo se plantea usar la tecnología para ampliar los medios disponibles permitiendo el fácil acceso al conocimiento ambiental y evaluar la asimilación y concientización de forma interactiva.

Actualmente la Universidad Javeriana de Cali y la Universidad de Sherbrooke trabajan en compañía desarrollando un proyecto que busca mejorar la inclusión y la educación en el ámbito medio ambiental para los niños del Instituto para Niños Ciegos y Sordos del Valle del Cauca. En el instituto se realizan diferentes procesos de habilitación y rehabilitación para niños con discapacidad visual y auditiva con el propósito de brindarles la educación y tratamiento adecuado. En el marco este proyecto se han creado diferentes narrativas las cuales se han plasmado en piezas tecnológicas que permiten la inclusión educativa de los niños. Una de estas narrativas consiste en el conocimiento y conservación del hábitat del Oso de Anteojos y el Mono Tití Cabeciblanco, narrativa para la cual se desarrolló un videojuego en donde el jugador o en este caso el niño debe guiar, al Oso o al Mono según su elección, por un laberinto a su hogar o hábitat, en el caso del Oso el objetivo es llegar a la montaña mientras que el Mono debe ser guiado al bosque. Con base en esto, este proyecto de grado busca realizar un prototipo de sistema de evaluación para medir el progreso y desarrollo de los niños en el juego, además de añadir elementos interactivos como sonidos y partículas para mejorar la experiencia. De esta manera se realiza la siguiente pregunta y sistematización:

1.1.1. Formulación

¿Como desarrollar un prototipo de un sistema que permita evaluar el desarrollo cognitivo en niños a través de un videojuego tipo laberinto?

1.1.2. Sistematización

- ¿Como utilizar los elementos actuales de un juego tipo laberinto para permitir la evaluación del desarrollo cognitivo en niños?
- ¿Como estructurar el videojuego de manera que permita la evaluación del progreso y desarrollo cognitivo de los jugadores?
- ¿Como implementar el videojuego que permita la evaluación del desarrollo de los niños?
- ¿Como validar el videojuego desarrollado con usuarios?

1.2. Objetivos

1.2.1. Objetivo General

Desarrollar un prototipo de un sistema que permita evaluar el desarrollo cognitivo en niños a través de un videojuego tipo laberinto.

1.2.2. Objetivos Específicos

- Identificar y seleccionar los elementos actuales del juego tipo laberinto de manera que puedan ser utilizados para evaluar el desarrollo cognitivo en niños.
- Estructurar el videojuego de manera que permita la evaluación del progreso y desarrollo cognitivo de los jugadores.
- Implementar el videojuego de manera que permita la evaluación del desarrollo cognitivo de los niños.
- Probar y validar el videojuego desarrollado con usuarios.

1.3. Justificacion

El proyecto colaborativo Colombia-Quebec, Narrativa, Realidad virtual y Discapacidad Sensorial valiéndose de la realidad virtual, la realidad aumentada y estrategias transmediales, propone la concepción, validación y creación de narraciones y experiencias multisensoriales e inmersivas para niños con diferentes capacidades sensoriales, acerca de la fauna colombiana, tomado de [6]. Teniendo esto en cuenta, dentro de una de las narrativas ya trabajadas, se desarrolló un videojuego cuyo objetivo es enseñarle al jugador respecto al cuidado y preservación de los hábitats del Oso de Anteojos y el Mono Tití Cabeciblanco de una manera interactiva. Este proyecto de grado tiene como objetivo la creación de un sistema de evaluación del desarrollo y progreso de los niños dentro del juego con el fin de que a posteriori, otros datos relacionados con la cognición y el desarrollo del niño puedan ser evaluados. Por ejemplo, el desarrollo y progreso del niño puede ser medido por medio de la cantidad de tiempo que se demore el jugador en completar un nivel. Además, el juego actualmente carece de un sistema de puntaje, que puede ser dado en relación al tiempo y otros datos, y de una interfaz con la cual llevar el progreso.

El cuidado y conservación del medio ambiente y los recursos es una labor que nos compete a todos ya que compartimos el mismo espacio llamado planeta Tierra. Animales y plantas entran constantemente de una lista llamada "en vía de extinción" y esto es precisamente para llamar la atención de que estamos acabando poco a poco con la diversidad. Conservar la biodiversidad es conservar la estabilidad de los climas, permitir que se desarrolle mayor resistencia a especies invasoras y enfermedades y puede llevar a mayor producción primaria e incluso permitir el uso eficiente de los recursos más limitados, esto a nivel económico, pero al destruir la biodiversidad se impacta directamente en el funcionamiento del ecosistema, en [7]. He aquí la importancia de poder comprender el medio ambiente y la relación entre sus diferentes elementos, entender que al desaparecer una especie tan importante como el Oso de Anteojos o el Mono Tití otras pueden verse afectadas hasta llegar al punto de desaparecer.

Como sabemos que poseemos diferentes capacidades tanto para aprender o realizar cualquier tipo de tarea y todos percibimos el mundo según nuestros sentidos lo permitan, por lo que para las personas con visibilidad o audición reducida la educación tradicional representa un reto. Las estrategias transmediales, como puede ser en este caso un videojuego, apoyan los procesos de aprendizaje aportando medios interactivos y adaptados a las necesidades de los alumnos. Este y otros aspectos adicionales nos llevan a una educación inclusiva en donde a cambio de resolver una relación con un paciente o enfermo, se trata de un ciudadano con derechos tal cual los tiene todo otro ser humano. Superar este sesgo, implicado en la caracterización de las personas a partir de la discapacidad que tienen de manera temporal o permanente, lleva a poner en duda los mecanismos actuales que, bajo esta idea ya normalizada, han cerrado el acceso a la educación a muchas personas [8].

De este modo se plantea contribuir al desarrollo de nuevos entornos de aprendizaje abiertos al público en general, utilizando herramientas tecnológicas con un enfoque de comprensión y de concientización sobre el cuidado y preservación de la biodiversidad y la importancia de las especies además de buscar sensibilizar sobre la problemática social y discriminatoria que viven las personas

con diferentes capacidades sensoriales, principalmente en el contexto educativo.

1.4. Delimitaciones y Alcances

El proyecto consiste en la actualización y mejora de un juego tipo laberinto desarrollado anteriormente, añadiendo funcionalidades para permitir la evaluación del desarrollo cognitivo de los jugadores. Se adicionará un modo de pruebas y un modo aleatorio al momento de jugar. El modo de pruebas permitirá jugar 3 niveles de complejidad de laberinto almacenados previamente y almacenar la información del jugador. Mientras que en el modo aleatorio el jugador debe cruzar un laberinto que se genera aleatoriamente cada vez que se desea jugar. En ambos modos se medirá el tiempo que se demora el jugador o el niño en recorrer el laberinto. El proyecto permitirá obtener y almacenar la información referente al jugador: id, nombre, edad, y sexo. No se adicionarán nuevas mecánicas de juego ni narrativas al desarrollo existente, únicamente se desarrollará el sistema de evaluación e interacciones adicionales del jugador como sonidos o elementos visuales.

El proyecto está dispuesto por el Instituto para Niños Ciegos y Sordos y para los niños del instituto (se cederá el proyecto para su propiedad y uso).

1.4.1. Entregables

- Prototipo funcional del sistema integrado al juego principal
- Trabajo de grado con toda la información relacionada al desarrollo, documentación y pruebas

Capítulo 2

Marco de referencia

2.1. Marco Teórico

En el marco de este proyecto se desarrollará un sistema de evaluación de los jugadores de un videojuego ya existente, que apoya los talleres y clases de los alumnos del Instituto para Niños Ciegos y Sordos, con el fin de poder dar un puntaje o valor a su desarrollo y progreso dentro del juego. Primero debemos definir los conceptos de juego, videojuego, discapacidad y otros.

2.1.1. Laberinto

Un Laberinto [9] es un espacio o lugar formado por un conjunto confuso de caminos y pasadizos que se conectan en el cual es fácil perderse. Normalmente los laberintos pueden ser diseñados para confundir a cualquier persona que ingrese en ellos y evitar que sigan un camino o para evitar que se llegue a un destino o tesoro. También existen laberintos cuyo objetivo es llegar al centro del mismo, donde se encuentra una recompensa.

A lo largo de la historia, los laberintos han sido utilizados también como figuras literarias para representar el alto grado de dificultad al realizar algunas tareas y está presente en diferentes historias de la mitología como por ejemplo en la leyenda del Minotauro [10]. En este caso el laberinto representa las dificultades de los animales para vivir y conseguir alimentos en su hábitat natural.

2.1.2. Juego

Según Huizinga, en [11], nos dice que el juego es una acción u ocupación libre, que se desarrolla dentro de unos límites temporales y espaciales siguiendo reglas obligatorias, aunque libremente aceptadas, va acompañado de un sentimiento de tensión, alegría y de la consciencia de ser de otro modo que en la vida corriente.

A lo largo de la historia, múltiples autores han dado su definición de juego y su impacto el desarrollo de los individuos en la sociedad y el ambiente en el que viven. “El juego es el modo con el que cuenta el niño para asimilar la realidad del mundo que le rodea, tiende un puente entre la actividad sensorio-motriz y la representación del pensamiento; el juego es una actividad feliz que comienza en deleite y termina en sabiduría; No se define por la actividad acometida sino por la actitud distintiva que toma el que juega hacia la actividad.” En [12].

2.1.3. Videojuego

A lo largo de la contemporaneidad las tecnologías de información han potenciado la interacción del ser humano con su entorno y esto se evidencia mucho más en los videojuegos.

Definición: Diego Levis nos define el videojuego como un entorno informático que reproduce sobre una pantalla de televisor o un monitor un juego cuyas reglas han sido previamente programadas, en [13]. Además, da a entender que debido a la capacidad interactiva y dinámica del juego en relación a la tecnología tiene un potencial transformativo en las relaciones tradicionales de los niños con el entorno, lo que puede llevar a replantearnos los presupuestos en los que se basa la educación tradicional.

Como se ha dicho anteriormente, un juego tiene un conjunto de reglas, que en el caso de los videojuegos están programadas en su código fuente, pero también se ven determinadas por la narrativa del mismo. La narrativa define la estructura de la historia y los principales elementos emocionales como el tema, la trama y los personajes.

2.1.4. Discapacidad

Definición: Una discapacidad es una afectación o deficiencia (temporal o permanente) del cuerpo o la mente que limita la realización de ciertas actividades y dificulta la interacción con el mundo que nos rodea, en [14]. Sin duda alguna estas deficiencias afectan los procesos cognitivos y por lo tanto la educación de los niños.

La educación es un mecanismo primordial para fomentar nociones técnicas e instrumentales en todas las áreas de conocimiento y ayudar a desarrollar destrezas que las personas requieren para desenvolverse un mundo competitivo, globalizado e interconectado [15].

Existen múltiples tipos de discapacidad, pero el enfoque de este proyecto es apoyar los procesos de aprendizaje y desarrollo de los niños del Instituto Para Niños Ciegos y Sordos del Valle del Cauca, por lo que el desarrollo de este trabajo se centra en niños con discapacidades visuales y auditivas.

2.1.4.1. Discapacidad Auditiva

Una deficiencia o discapacidad auditiva ocasiona que la persona no pueda oír bien y por esto no esté en capacidad de comprender los sonidos del medio ambiente ni de la lengua oral que se habla en su entorno [16]. Así como existen múltiples tipos de discapacidades, también existen múltiples formas de adquirir una discapacidad y diferentes niveles o tipos de discapacidad auditiva que son [17]:

- Sordera: deficiencia total o profunda del sentido auditivo.
- Hipoacusia: Deficiencia parcial del sentido auditivo. A medida que se avanza en edad, se va perdiendo poco a poco la audición debido a la exposición a ruidos fuertes o daños en el aparato o nervio auditivo.

También se pueden clasificar por su origen:

- Genéticas o hereditarias.
- Adquiridas en alguna etapa de la vida.
- Congénitas: Por una enfermedad adquirida por la madre durante el embarazo, por un trauma o asfixia durante el parto o por partos prolongados.

"La discapacidad auditiva tiene efectos importantes en las etapas tempranas de la vida de una niña o un niño porque afecta su pensamiento, habla, lenguaje, conducta, desarrollo social y emocional, así como su desempeño escolar y laboral." De [18].

2.1.4.2. Discapacidad Visual

La discapacidad visual se puede categorizar en tres grupos: Alteraciones funcionales, alteraciones de estructura y alteraciones del sistema nervioso. Las alteraciones funcionales que implican efectos como visión borrosa, impedimentos para adaptarse a los cambios de luz, dificultad para ver en la oscuridad o con luz tenue, impedimento de identificar ciertos colores. Las alteraciones o trastornos de estructura tienen que ver con afectaciones directas como traumas, impactos presiones (glaucoma), parásitos o infecciones que terminan afectando partes importantes del ojo como los nervios ópticos, la córnea, el cristalino, entre otros. Por último, las alteraciones nerviosas están relacionadas con la atrofia natural al nervio óptico que lleva las imágenes de lo que vemos al cerebro, de [19].

"Los niños pequeños con discapacidad visual grave pueden sufrir retrasos en el desarrollo motor, lingüístico, emocional, social y cognitivo, con consecuencias para toda la vida. Los niños en edad escolar con discapacidad visual también pueden presentar niveles más bajos de rendimiento académico" [20].

2.1.5. Herramientas de desarrollo

Una herramienta es un objeto elaborado con la finalidad facilitar la realización de una labor o tarea. Las herramientas fueron inicialmente metálicas y se usan para realizar cualquier labor. Ejemplos de herramientas puede ser una pala, una escoba o hasta un cepillo de dientes. Así mismo existen herramientas de desarrollo que son programas virtuales que apoyan los procesos de diseño, maquetación, programación y depuración del software [21]. En este caso la herramienta seleccionada para el desarrollo principal del videojuego es Unity. Unity es una plataforma de desarrollo que permite crear experiencias multiplataforma, juegos y más en perspectiva 2D y 3D [22]. En el editor de Unity podemos hacer uso de programas y scripts en lenguaje C#. Este proyecto se desarrolló inicialmente en Unity y seguirá siendo desarrollado en Unity. En la figura 2.1 vemos una captura de pantalla del entorno de desarrollo de Unity.

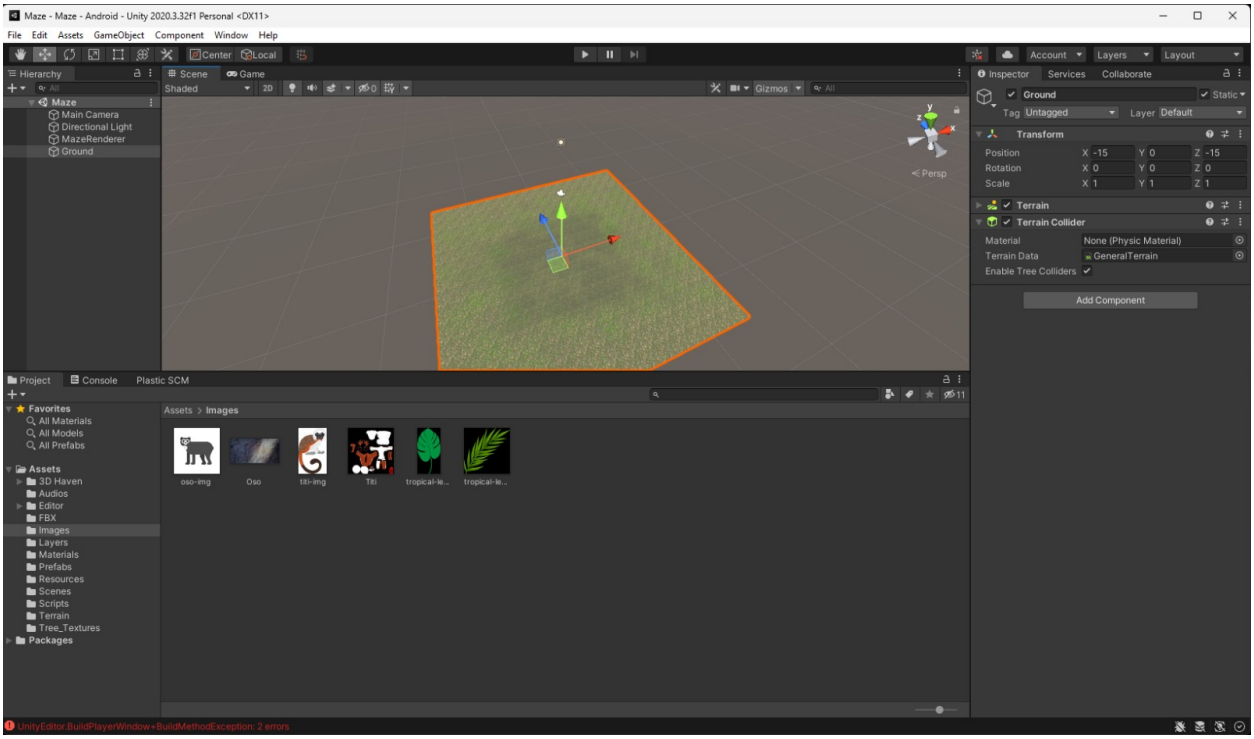

Figura 2.1: Captura de pantalla donde vemos el editor de Unity con los archivos del proyecto.

2.2. Trabajos relacionados

- **Un videojuego para apoyar la terapia del lenguaje en niños con discapacidad auditiva. El caso de la descripción dinámica:** Para este trabajo se desarrolló un videojuego en el cual los jugadores, en este caso niños en proceso de aprendizaje y con alguna afectación auditiva, deben completar y organizar una secuencia de imágenes que narran una historia. El videojuego *Secuencia de Caricaturas* está compuesto de 7 niveles diferentes. En cada uno de los niveles el niño trabaja diferentes habilidades del lenguaje como: organizar secuencias de manera lógica, realizar descripciones por medio de oraciones simples y compuestas y realizar inferencias para desarrollar habilidades para crear narraciones, entre otras [23].

  Observamos que por medio del videojuego se busca que los jugadores desarrollen capacidades y responsabilidades frente al mundo externo y se apropien de una narrativa especifica, como puede ser una de la vida cotidiana o del cuidado del medio ambiente. Junto con este proyecto, ambas son maneras interactivas de aprender y se enfocan en apoyar el aprendizaje de un grupo reducido de personas como lo son los niños con discapacidades auditivas.

- **A serious game for children with speech disorders and hearing problems:** "Into

the forest" es el nombre que recibe el videojuego realizado en este proyecto relacionado. El juego consiste en que el jugador por medio del habla, mueve un personaje a través del entorno para ganar monedas. El videojuego permite recibir comandos de voz sencillos como pueden ser "arriba", "adelante", "atrás", "derecha" o "izquierda" además de contar con un tutor que ayuda a los niños de manera visual por medio de flechas. Este videojuego esta específicamente desarrollado para niños con dificultades auditivas y desordenes del habla [24].

Este proyecto relacionado puede ser muy parecido al videojuego explicado en este documento, ya que el objetivo de ambos es guiar al personaje por medio de comandos; en este trabajo relacionado el usuario o jugador usa comandos de voz para mover el personaje mientras que en este proyecto se usan movimientos del jugador que se detectan por medio del giroscopio del celular. Ambos trabajos tienen el enfoque de apoyar la educación y aprendizaje de niños con discapacidades auditivas por medio de una narrativa diferente.

- **Sistema Informático para Niños Discapacitados Visuales en Etapa Preescolar:** El Centro de Estudios de Rehabilitación Nutricional y Desarrollo Infantil (CEREN) que hace parte de la Comisión de Investigaciones Científicas de la Provincia de Buenos Aires (CIC-PAB) ha implementado "La Valijita Viajera", un sistema informático compuesto por una aplicación multimedial y un conjunto de materiales didácticos que complementan las actividades educativas con el fin de promover un crecimiento integral en los niños con diferentes capacidades visuales y ciegos [25].

  Este trabajo se puede considerar relacionado a este proyecto ya que ambos presentan una herramienta didáctica para que los educadores puedan evaluar de una manera diferente a los niños con discapacidad visual. Este trabajo relacionado no es un videojuego como el desarrollado en este proyecto, sino que presenta múltiples materiales audiovisuales y ejercicios didácticos. Tanto este proyecto como el trabajo relacionado anteriormente descrito buscan una adecuada promoción del desarrollo integral de niños discapacitados visuales.

Capítulo 3

Metodología

3.1. Actividades por objetivo

- **Objetivo 1:** *Identificar y seleccionar los elementos actuales del juego tipo laberinto de manera que puedan ser utilizados para evaluar el desarrollo cognitivo en niños.*
  - Identificar el funcionamiento actual del juego
  - Identificar los elementos interactivos y medibles del juego y que otros se pueden añadir.
  - Evaluar los elementos interactivos que se pueden añadir, como partículas y sonidos.
- **Objetivo 2:** *Estructurar el videojuego de manera que permita la evaluación del progreso y desarrollo cognitivo de los jugadores*
  - Seleccionar los elementos que definirán el sistema de evaluación del jugador y su funcionamiento
  - Diseñar un prototipo del sistema de evaluación y puntaje.
  - Diseñar y seleccionar una interfaz para comunicar el progreso del jugador durante y al final del juego.
- **Objetivo 3:** *Implementar el videojuego de manera que permita la evaluación del desarrollo cognitivo de los niños*
  - Evaluar que programas y archivos deben ser modificados y/o actualizados para implementar el nuevo sistema.
  - Implementar el sistema de evaluación.
- **Objetivo 4:** *Probar y validar el videojuego desarrollado con usuarios*
  - Construir un plan de pruebas.
  - Probar el prototipo con usuarios.
  - Analizar las pruebas y proponer trabajos futuros o mejoras que se pueden realizar.

3.2. Metodologías de desarrollo

3.2.1. Metodología Agil

Una metodología ágil está basada en el desarrollo iterativo, la inspección, adaptación y retroalimentación frecuente, el aumento incremental y evolutivo de la solución planteada y el trabajo organizado en equipos multifunciones e independientes. El trabajo se distribuye en pequeñas ventanas de tiempo conocidas como iteraciones en las cuales de desarrollan secciones del producto final con el objetivo de que, una vez finalizada la iteración, se pueda demostrar avances a los interesados o clientes obteniendo así una retroalimentación constante. De este modo es una metodología flexible que permite modificaciones de los requerimientos durante las etapas de desarrollo y pruebas. Algunas características aplicadas en este proyecto son, de [26]:

- El proceso de desarrollo en iteraciones cortas que reemplaza los largos procesos secuenciales.
- Desarrolladores intentan crear información completa y asertiva al final de cada iteración. Pero una revisión puede llevar a un proceso adicional de desarrollo.
- Los casos de uso y la comunicación oral son los recursos usados frecuentemente más que en especificaciones formales y documentos de diseño.
- Demostraciones y las revisiones por pares (desarrollador - cliente) validan cómo el desarrollo ha logrado los objetivos previstos.

3.2.2. Scrum

Scrum es una estructura de las metodologías ágiles que otorga una flexibilidad en el control y manejo sobre los requerimientos. Los requerimientos se consideran la pila de producto o Product Backlog que describe todas las tareas a realizar en el proyecto. Al iniciar cada Sprint o iteración se toma una cantidad de tareas específicas de la pila de producto y se organizan para ser realizadas en el tiempo determinado de duración de cada Sprint. Scrum es modelo flexible que puede ser aplicados a cualquier tipo de proyecto de cualquier industria y puede ser útil tanto para pequeños cómo grandes proyectos [27].

Este proyecto tomó sprints o iteraciones semanales en donde al inicio de cada iteración se realizaba una reunión con el Dr. Andrés Adolfo Navarro Newball que está en constante contacto con el Instituto para niños Ciegos y Sordos del Valle del Cauca, conociendo así las necesidades más importantes ante el desarrollo del sistema y logrando una verificación y validación continua del cumplimiento de los requerimientos del proyecto.

Capítulo 4

Análisis

4.1. Actores del sistema

Dentro de las personas del Instituto para Niños Ciegos y Sordos del Valle del Cauca se identificaron dos usuarios que interactúan directamente con el sistema que son el Profesional Encargado y el Jugador o Niño

4.1.1. Jugador

El actor o usuario JUGADOR/NIÑO es definido como cualquier niño que estudie en el Instituto para Niños Ciegos y Sordos del Valle del Cauca y que use el videojuego. Este actor puede seleccionar el avatar con el que cruzará el laberinto además de jugarlo, que consiste en direccionar el avatar por medio de la inclinación del dispositivo o tableta y finalmente ver los resultados obtenidos una vez cruce la salida del laberinto.

4.1.2. Profesional Encargado

El actor o usuario PROFESIONAL ENCARGADO es definido como cualquier persona a cargo de la educación y cuidado de los niños dentro del instituto. De este modo el profesional encargado puede ser un profesor, tutor, analista o director del instituto. La función de este usuario es guiar al usuario JUGADOR en la configuración pre-juego, por lo que el sistema debe ser capaz de permitirle crear nuevos jugadores, seleccionar jugadores ya creados, seleccionar el modo de juego, entre otras funciones. Una de las funciones más importantes es seleccionar la dificultad del laberinto que cruzará el jugador según el criterio médico del jugador. Es decir que el profesional encargado puede decidir que dificultad de laberinto (fácil 4x4, intermedio 7x7, difícil 10x10) jugará el niño o niña según las capacidades. Los tamaños de los laberintos de cada nivel de dificultad se acordaron con el director del instituto y el director del trabajo de grado.

4.2. Requisitos

A continuación, se muestra la lista completa de los requisitos del proyecto:

- RF-01 Interfaz de usuario:
  - Descripción: El sistema debe permitir la interacción por medio de una interfaz entre el usuario y el sistema.

- RF-02 Menú inicial:
  - Descripción: Al iniciar el juego, el sistema debe mostrar un menú inicial para guiar al usuario.
- RF-03 Modos De Juego:
  - Descripción: El sistema en el menú inicial debe permitir al usuario profesional encargado seleccionar entre los diferentes modos de juego. (Pruebas o Práctica)
- RF-04 Laberintos de prueba:
  - Descripción: En el modo de pruebas el sistema debe permitir jugar tres (3) laberintos estáticos previamente almacenados, uno 4x4, uno 7x7 y uno 10x10
- RF-05 Información del jugador:
  - Descripción: El sistema debe permitir al usuario guardar la información de los jugadores del modo de pruebas por medio de una interfaz. La información necesaria es: Id del jugador, nombre del jugador, edad y sexo.
- RF-06 Laberinto de práctica:
  - Descripción: En el modo de práctica el sistema debe permitir al usuario jugador jugar tres laberintos de diferente dificultad que son generados aleatoriamente.
- RF-07 Puntaje:
  - Descripción: El sistema debe mostrar por medio de la interfaz un puntaje por cada laberinto completado por el usuario jugador.
- RF-08 Tiempo:
  - Descripción: El sistema debe tomar el tiempo que se demora el usuario jugador en completar cada laberinto.
- RF-09 Selección del avatar:
  - Descripción: El sistema debe permitir al usuario seleccionar un avatar con el que va a jugar por medio de una imagen.
- RF-10 Información de modo pruebas:
  - Descripción: El sistema debe permitir almacenar los diferentes puntajes, tiempos e información de los laberintos completados por los jugadores del modo de pruebas.
- RF-11 Impactos de pared:

  - Descripción: El sistema debe generar partículas y un sonido cuando el avatar del jugador impacta las paredes del laberinto durante el juego.

- RF-12 Aura del avatar:
  - Descripción: El sistema debe mostrar un aura alrededor del avatar para apoyar la visualización del usuario jugador durante el juego.
- RF-13 Imagen del piso:
  - Descripción: El sistema en la escena de laberinto debe cargar fondo plano (piso / suelo del laberinto) que contraste con los otros elementos del videojuego.
- RF-14 Resumen de partida:
  - Descripción: Al finalizar cada modo de juego sistema debe mostrar un resumen de puntaje obtenido por el usuario jugador al igual que desplegar un botón para volver al menú inicial.
- RF-15 Persistencia de la información:
  - Descripción: El sistema debe permitir persistir la información de los jugadores y partidas de los laberintos de prueba.
- RF-16 Menú modo pruebas:
  - Descripción: El sistema, en el menú inicial debe permitir al usuario profesional encargado seleccionar un jugador que ya existe o crear uno nuevo para jugar el modo de pruebas.
- RF-17 Creación nuevo jugador:
  - Descripción: El sistema debe contar con una interfaz que permita al usuario profesional encargado ingresar la información de un nuevo jugador en el modo de pruebas.
- RF-18 Contraste de elementos y videojuego:
  - Descripción: Los elementos visuales del sistema deben tener alto contraste para facilitar la visualización.
- RF-19 Visualización de resultados:
  - Descripción: El sistema debe permitir al usuario profesional encargado visualizar los resultados de las partidas del modo de pruebas realizadas por cada jugador.
- RF-20 Dificultades de juego:
  - Descripción: Sin importar el modo escogido, el usuario profesional encargado debe poder seleccionar la dificultad de la partida /juego antes de empezar.

4.3. Diagrama de casos de Uso

En un diagrama de casos de uso se tienen Actores y Casos. Un Actor en el diagrama es representado por una persona hecha de palitos la cual es análoga al rol de usuario que tiene dentro del sistema. Una asociación de un actor con un Caso de uso significa que el actor es el que lleva a cabo ese caso de uso. De este modo un caso de uso indica de manera general los hechos que ocurren en la interacción Actor - Caso, lo que pasa después y lo que podría llegar a pasar, de [28].

En la figura 4.1 podemos observar el diagrama de clases resultante que describe la interacción de los diferentes usuarios del sistema con el videojuego. Además, se puede diferenciar una secuencia en las acciones que dan un resultado observable en la interfaz. Podemos observar que el Actor Profesional Encargado posee la mayor parte de interacción directa con el sistema, ya que el configura el modo de juego, la dificultad, puede ver los resultados y generar nuevos laberintos. Pero por otro lado el Niño o Jugador realiza principalmente dos acciones directas sobre el sistema que es seleccionar el avatar o personaje con la cámara AR y jugar el laberinto.

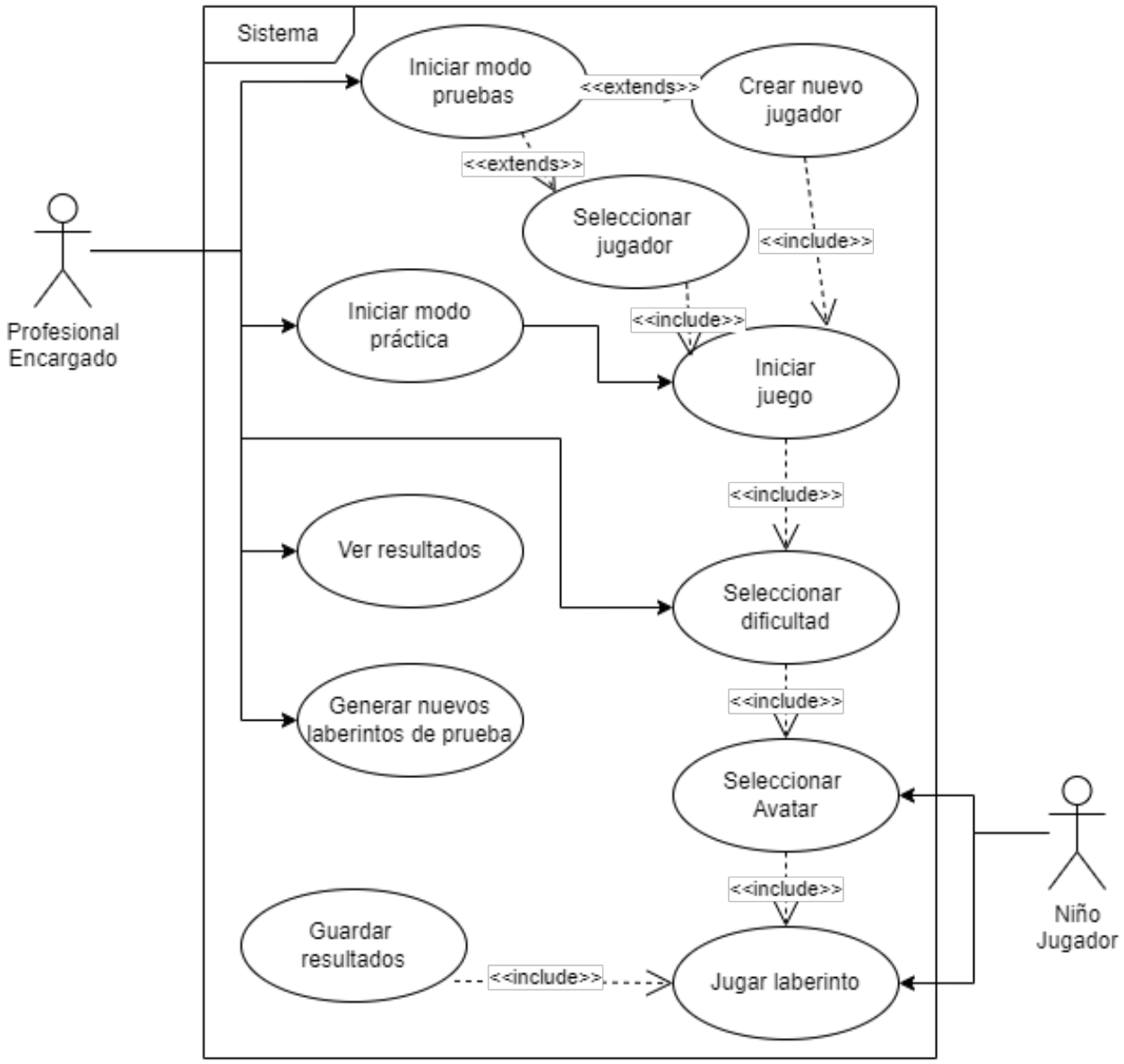

Figura 4.1: Diagrama de casos de uso que representa los actores que interactúan con el sistema.

Capítulo 5

Diseño

5.1. Descripción del modelo del sistema

En el Editor de Unity cada aplicación, juego, etc. que se cree está compuesto por diferentes escenas que son usadas para crear el menú principal, los niveles individuales, formularios y cualquier otro tipo de ambientes [29].

Dentro de cada escena de nuestro proyecto en Unity podemos definir objetos nombrados cómo GameObjects que en sí solos no representan mayor valor, pero con ayuda del motor de Unity podemos añadir propiedades a estos objetos ampliando las interacciones y funcionalidades. Posiblemente la propiedad más importante de los GameObjects son los scripts, lo que nos permite por medio de lenguaje de programación C# cambiar otras propiedades y finalmente generar cualquier tipo de interacción necesaria con el usuario. Toda clase definida en Unity también se conoce cómo script.

Una vez dicho esto podemos explicar que el sistema contiene 3 escenas diferentes: Menu, escena AR o realidad aumentada y escena Maze o de juego. En el menú inicial como su nombre lo indica, se añade diferentes campos, botones y textos que, por medio de la clase MainMenu que se detalla en el diagrama de clases, configura las otras escenas según la elección del usuario. En esta escena el usuario Profesional Encargado puede seleccionar el modo de juego que configura las escenas posteriores, seleccionar o crear el jugador y elegir la dificultad del juego, entre otras acciones. En el modo de pruebas se carga un laberinto almacenado previamente según la dificultad escogida y al final de la partida se guarda la información por medio de la clase SaveSystem. Otro caso que se puede dar es que el usuario profesional encargado puede seleccionar el modo de práctica el cuál siempre carga un laberinto aleatorio de tamaño nxn según la dificultad escogida, sin necesidad de almacenar la información. En el modo de pruebas cómo en el de práctica, el profesional encargado debe seleccionar la dificultad del laberinto. La dificultad define el tamaño del laberinto que se va a jugar.

Dificultades:

- Fácil: Laberinto 4x4
- Intermedio: Laberinto 7x7
- Difícil: Laberinto 10x10

La escena AR es la escena previa al inicio de la partida en cada modo de juego. En esta escena se le solicita al usuario JUGADOR escoger un avatar (Oso de anteojos o Tití Cabeciblanco) por medio de la cámara del dispositivo donde se encuentra instalada la aplicación. El niño o jugador debe

apuntar la cámara del dispositivo a la imagen del avatar que desea escoger. Una vez el dispositivo capta la imagen del avatar por medio de la cámara, (AR por un script) guarda el objeto seleccionado en la clase WorldSingleton y se llama a ejecutar la escena de juego o Maze.

En la escena Maze se instancia la clase MazeRenderer que busca la información del laberinto a dibujar según la configuración previa configurada en SaveSystem e instancia el avatar seleccionado. Una vez esto ocurre se le da comienzo al Juego y se activan el contador de tiempo manejado por la clase TimerScript y el contador de puntos manejado por la clase PlayerController. Una vez se detecta que el avatar a logrado llegar a la salida del laberinto se muestran los resultados del juego y se almacenan por medio de la clase SaveSystem. Una vez finalizada una partida el ciclo se ejecuta nuevamente lanzando la escena Menu.

5.2. Diagrama de Clases

El diagrama presentado en la figura 5.1 podemos observar las principales clases del proyecto a desarrollar. Como bien se ha dicho anteriormente en Unity se manejan escenas donde se muestran y ejecutan las diferentes interacciones de los objetos del videojuego o sistema a desarrollar. De este modo se creó una clase principal para cada una de las escenas del videojuego y adicionalmente una clase que permita la comunicación de la información entre cada escena. MainMenu es la clase que se instancia durante toda la ejecución de la escena Menu. En esta escena se configura el sistema para realizar pruebas, iniciar el modo prácticas, generar nuevos laberintos, seleccionar un jugador o crear uno nuevo, seleccionar la dificultad, ver resultados, entre otras. Todo esto con ayuda de la clase SaveSystem que va almacenando la configuración seleccionada en esta escena además de contar con métodos para leer y guardar información. Por ejemplo, el atributo modoPruebas del tipo booleano en la clase SaveSystem es usado durante toda la ejecución del sistema para almacenar un valor que nos indica si la partida que se va a jugar es una prueba o una práctica. Otro ejemplo del uso de la clase SaveSystem desde MainMenu es el uso del método nuevoMazeLevel3() que llama a el método NewLevelMaze() y LoadMaze3() de SaveSystem para crear, almacenar y cargar un nuevo laberinto de dificultad 3.

Por otro lado, cuando se ejecuta la escena AR se instancia la clase WorldSingleton, que usa el patrón de diseño Singleton como su nombre lo indica y se instancian dos objetos TravelerController que almacena el nombre de los objetivos que se van a detectar por medio de la cámara, Oso y Titi respectivamente. En la clase WorldSingleton se almacena el nombre del Target" u objetivo una vez la cámara AR detecta la imagen del Oso o del Titi. Finalmente se carga la escena Maze desde el objeto TravellerController que detectó la imagen. Se usó el patrón de diseño Singleton para que en el caso de que se detecten las dos imágenes objetivo al tiempo, solo se seleccione un avatar y se llame a ejecutar la escena Maze una única vez.

Finalmente, la clase MazeRenderer es la primera clase que se instancia en la escena Maze, esta clase lee toda la información previamente almacenada en las escenas anteriores desde las instancias de los objetos SaveSystem y WorldSingleton. Con esta información, según la dificultad y personaje seleccionado se dibuja el laberinto por medio del método DrawMaze() o DrawPruebasMaze() de MazeRenderer y se instancia el objeto PlayerController. PlayerController es la clase encargada del

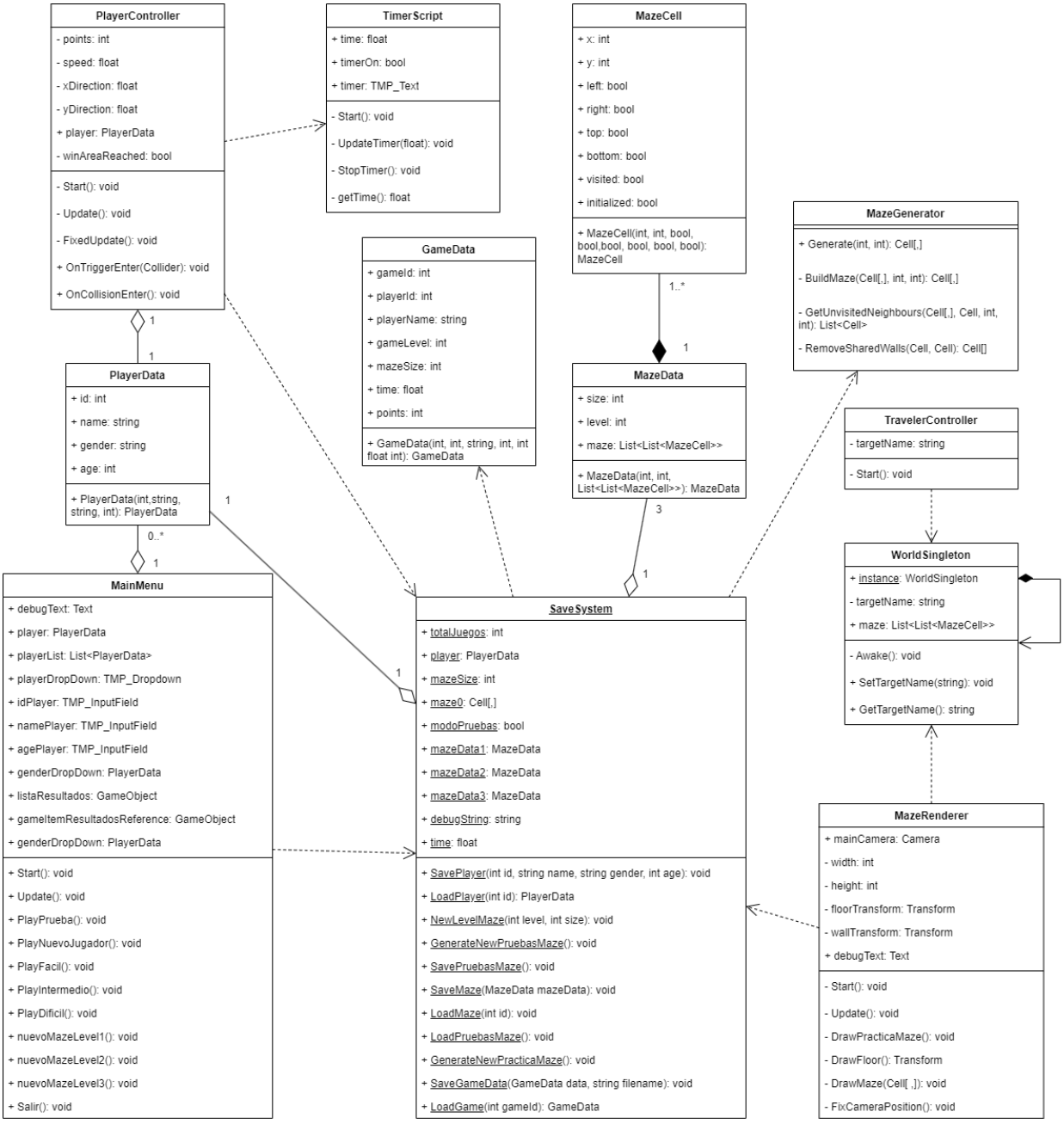

Figura 5.1: Diagrama Clases: Muestra las clases del sistema con sus asociaciones y dependencias

movimiento del personaje por medio de la inclinación del dispositivo, toma de puntaje y tiempo, interacciones y detección de colisiones.

Por otro lado, la clase MazeData es usada para instanciar y almacenar la información de los diferentes laberintos. MazeData tiene los atributos size y level representando así el tamaño y el nivel del laberinto a instanciar. El atributo maze de MazeData es una matriz de objetos MazeCell. Esto se representa con una relación de composición con MazeCell ya que MazeData por sin este último atributo no representa ningún laberinto. La clase MainMenu tiene una relación de agregación con PlayerData ya que se debe instanciar la información de los jugadores que se pueden seleccionar y del jugador seleccionado. La clase SaveSystem tiene referencias a las clases PlayerData para almacenar, durante toda la ejecución del sistema, el jugador seleccionado inicialmente desde MainMenu y tiene además referencias de los laberintos que se cargan durante la inicialización del juego, es decir referencias a MazeData. La clase PlayerController también hace referencia a un objeto PlayerData que se usa para guardar la información del jugador al final de cada partida.

Vale la pena aclarar que las clases PlayerController, TimerScript, MazeRenderer, MainMenu y TravelerController son clases hijas de la clase base MonoBehaviour propia de Unity. La clase MonoBehaviour ofrece funciones del ciclo de vida del motor de videojuego que hacen mucho más sencillo realizar desarrollos en Unity. Algunas de estas funciones que son usadas en este proyecto y que también se ven presentes en el diagrama de la figura 5.1 son Start(), Update() y FixedUpdate(). La función Start() es llamada en el marco (de marco por segundo) cuando se habilita un script justo antes de que se llame por primera vez cualquiera de los métodos Update. En otras palabras cuando un objeto se instancia, primero se llama a la función Start() antes de que ese objeto empiece a interactuar por medio de las funciones update, luego la función Update() es llamada cada marco por segundo definido en el juego y la función FixedUpdate() es llamada en cada frecuencia física del sistema, aproximadamente 50 llamados por segundo. Podemos encontrar más información sobre estas funciones en [30]

5.3. Diagrama de Secuencia

Los diagramas de secuencias describen la interacción entre las instancias de los objetos durante la ejecución de un sistema. En los diagramas representados en las figuras 5.2, 5.3 y 5.4 ignoran las interacciones con las clases propias de Unity. El objeto SceneManager, como su nombre en inglés lo indica, es el encargado de manejar y controlar todos los objetos necesarios en cada escena e incluso por medio de la función LoadScene() cargar una escena con la configuración inicial de la misma.

5.3.1. Escena 1: Menu

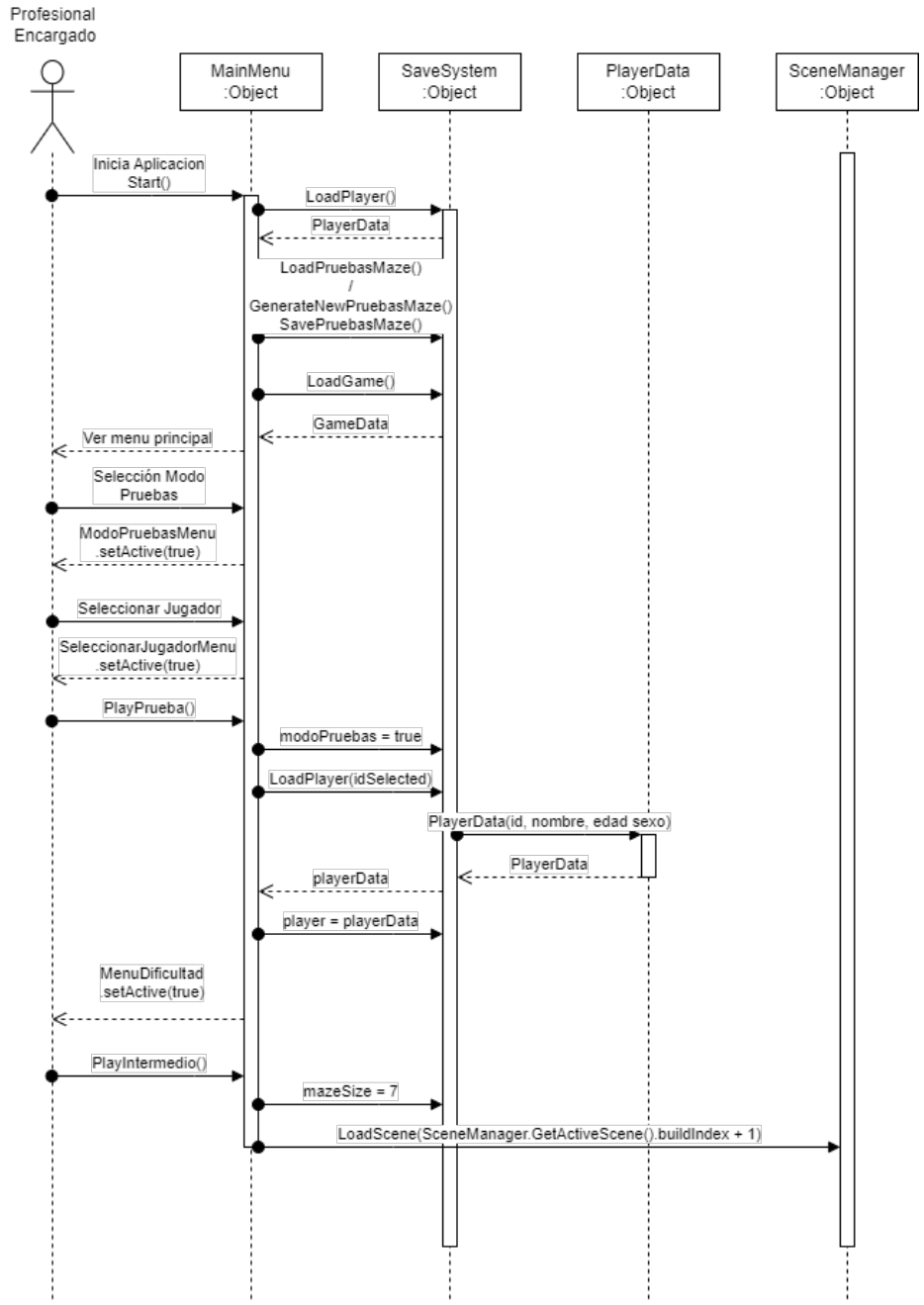

Figura 5.2: Diagrama de secuencia 1: indica la secuencia de ejecución para la escena Menu, en donde se configura y selecciona el jugador para una partida de pruebas.

En el diagrama presentado en la Figura 5.2 se muestra la interacción de los objetos durante la escena Menu. Este diagrama muestra la secuencia en la que el actor Profesional Encargado va a realizarle una partida en modo de prueba de dificultad intermedia (laberinto de tamaño 7x7). El diagrama comienza cuando el actor inicia la aplicación y se instancian los objetos MainMenu y SaveSystem. La función Start() de MainMenu inicializa las listas de los jugadores y la lista de los resultados previamente almacenados si es que existen. De igual manera se revisa si existen laberintos de pruebas previamente almacenados, en caso de que no existan se generan y se guardan por medio de los métodos GenerateNewPruebasMaze() y SavePruebasMaze().

Una vez se carga la información el actor Profesional Encargado puede empezar a interactuar con el menú principal. En el diagrama de la figura 5.2 vemos que el actor selecciona el modo de pruebas y selecciona un jugador ya existente. Además al presionar Jugar desde el menú se ejecuta la función PlayPrueba() de SaveSystem lo que configura el juego y carga el jugador seleccionado por medio de la función LoadPlayer(). Esta función nos retorna un objeto del tipo PlayerData que es almacenado en SaveSystem durante el resto de la ejecución de la partida. El encargado ahora puede seleccionar la dificultad a la que jugará el niño en el diagrama vemos que selecciona la dificultad intermedio (PlayIntermedio()). Esto finalmente configura el tamaño del laberinto que se va a cargar y se llama LoadScene() del objeto SceneManager para continuar a la escena AR.

5.3.2. Escena 2: AR

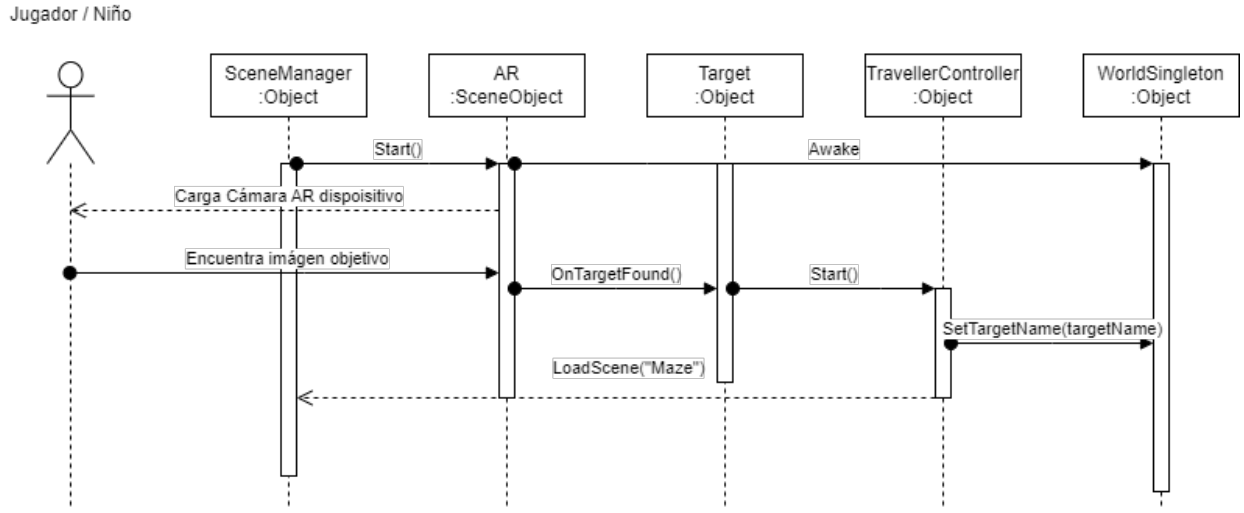

Figura 5.3: Diagrama de secuencia 2: indica la secuencia de ejecución para la escena AR, en donde se busca la imagen objetivo para seleccionar el avatar.

En esta escena el principal actor es el jugador o niño al cual se le entrega el dispositivo ya configurado para realizar un laberinto de práctica o de prueba. En la figura 5.3 vemos que una vez

el jugador encuentra la imagen objetivo para seleccionar su personaje, se llama la función OnTargetFound() de un objeto Target. Este objeto es un GameObject añadido en la escena de modo que almacena la imagen objetivo del personaje a seleccionar y además contiene un objeto TravellerController que se encuentra desactivado, de modo que existen tantos objetos Target según posibles personajes o avatares posibles pueda seleccionar el jugador. Una vez se encuentra una imagen objetivo en el campo de visión de la cámara del dispositivo se activa el objeto TravellerController que almacena el nombre del objetivo seleccionado y desde TravellerController realizan dos cosas, primero se configura el nombre del personaje seleccionado en el objeto WorldSingleton y segundo se realiza el llamado de cargar la escena del laberinto (Maze) por medio del llamado de LoadScene() del manejador de escenas de Unity (SceneManager).

5.3.3. Escena 3: Maze

En esta última escena las clases principales son MazeRenderer y PlayerController. Cuando se carga esta escena la clase MazeRenderer utiliza los valores almacenados previamente para dibujar el laberinto (DrawPruebasMaze()), obtener el nombre del personaje escogido (lee el nombre de WorldSingleton) e iniciar el contador de tiempo (Start() de TimerScript) cómo se muestra al inicio del diagrama de secuencia de la figura 5.4. Luego de esto instancia el GameObject con la textura del personaje escogido dentro del laberinto (Start() de PlayerController) e inicia el juego. Todos estos pasos descritos anteriormente habilitan el movimiento del personaje dentro del laberinto. Cuando el jugador logra llevar al animal o personaje afuera del laberinto, se ejecuta la función OnTriggerEnter() lo que detiene el contador de tiempo y lo obtiene por medio de StopTimer() y getTime() de la clase Timer además de mostrarle los resultados al jugador y guardar la información de la partida con SaveGameData() de SaveSystem. Finaliza el juego cargando nuevamente el menú inicial por medio del SceneManager.

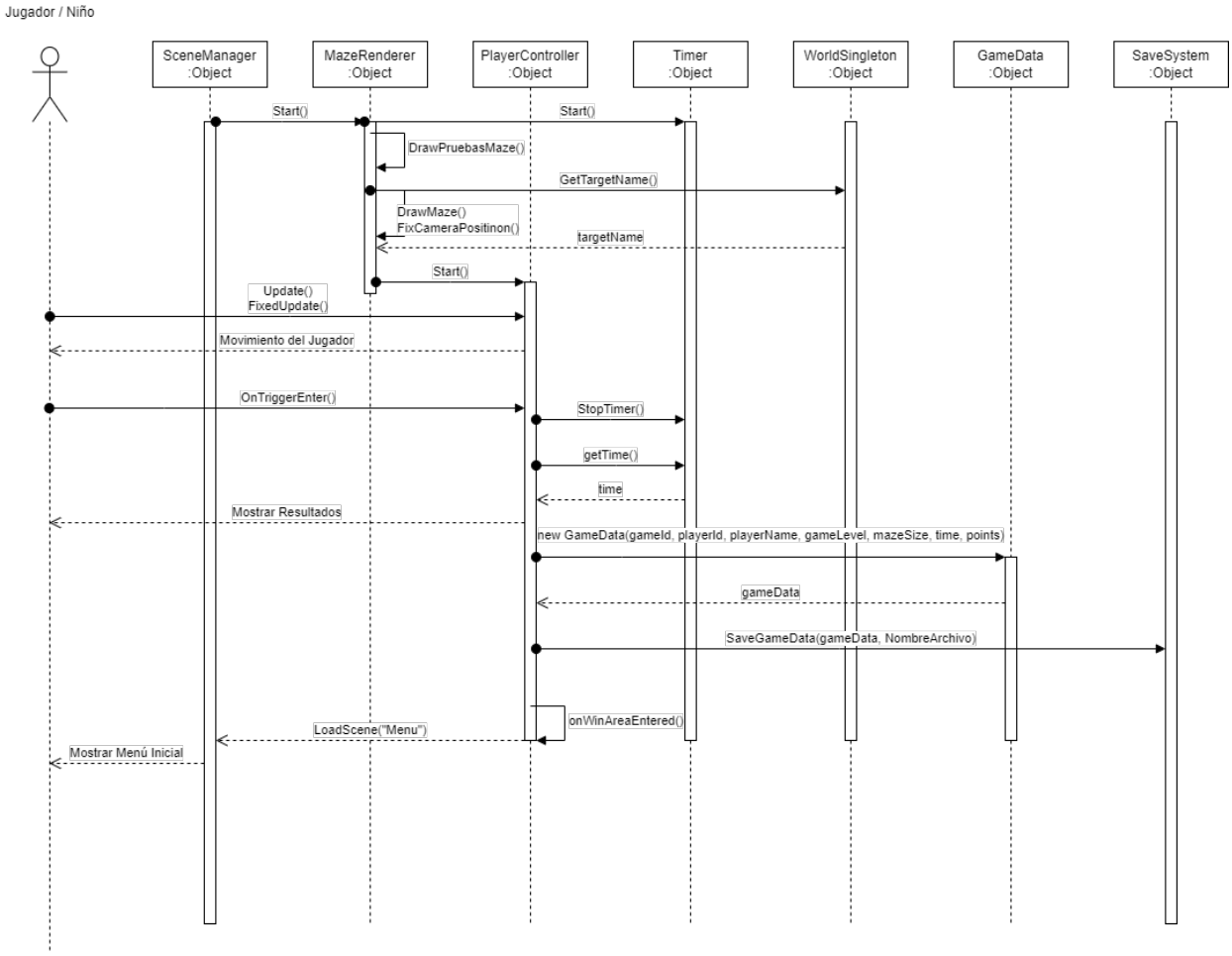

Figura 5.4: Diagrama de secuencia 3: muestra la secuencia de ejecución de los objetos durante la escena Maze

5.4. Diseño de Interfaz

Para el sistema en general se pueden diferenciar tres interfaces importantes al momento de interactuar con el jugador o profesional encargado. La primera interfaz es el menú inicial, es la primera pantalla que se ve una vez el profesional encargado inicia la aplicación. En el menú inicial se encuentran las opciones para empezar partidas o juegos de tipo pruebas o práctica, opciones y salir. Este menú inicial es la primera escena que se carga en el motor de Unity. De este modo a continuación se presenta una lista jerárquica que representa el menú inicial y sus submenús.

- MODO PRUEBAS

- SELECCIONAR JUGADOR
  - JUGAR
    - FÁCIL
    - INTERMEDIO
    - DIFÍCIL
- NUEVO JUGADOR
  - JUGAR
    - FÁCIL
    - INTERMEDIO
    - DIFÍCIL

- MODO PRACTICA
  - FÁCIL
  - INTERMEDIO
  - DIFÍCIL

- OPCIONES
  - GENERAR LAB1
  - GENERAR LAB2
  - GENERAR LAB3
  - RESULTADOS

- SALIR

Cómo resultado al momento de diseñar en el editor de Unity se obtuvo el menú presentado en la Figura 5.5.

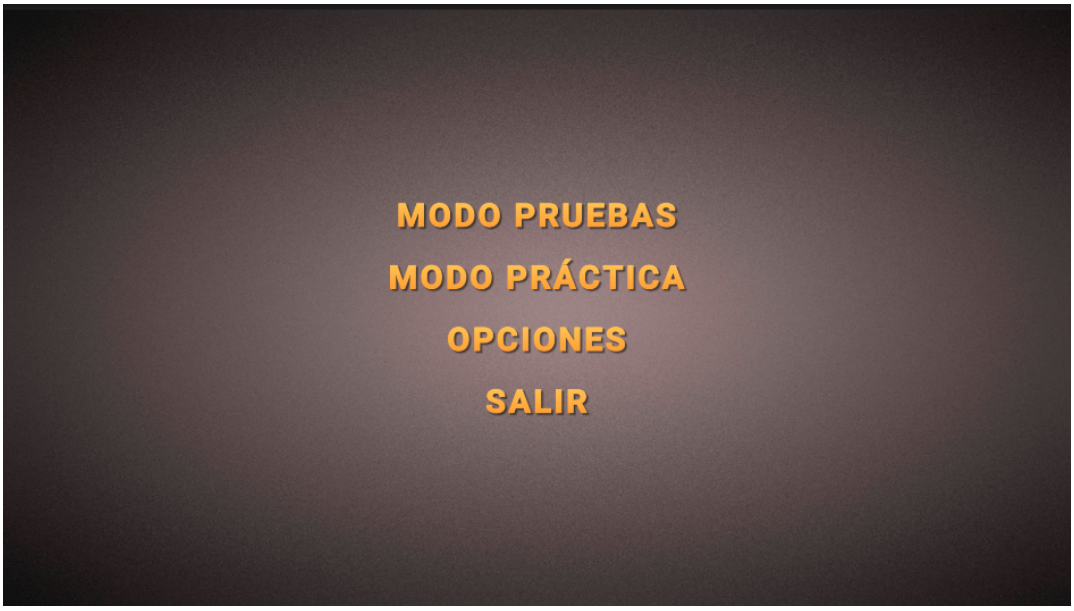

Figura 5.5: Menú con diseño y colores iniciales.

Figura 5.6: Menú inicial del sistema con colores finales.

Se evaluaron otros colores como fondo blanco y letras rojas pero se decidió finalmente usar la imagen del fondo que se ve en la Figura 5.6 y modificar el color de la imagen para obtener mayor

contraste de las letras amarillas. Cada vez que se va a iniciar un nuevo juego se debe seleccionar la dificultad, el menú es presentado en la Figura 5.7.

Figura 5.7: Menú de selección de dificultad.

La segunda interfaz importante que se ve en el orden de ejecución es la cámara de Realidad Aumentada o escena AR, que en este caso se usa como selector del avatar o personaje que usará el niño para recorrer el laberinto. El sistema en esta escena ejecuta la cámara del dispositivo y una vez se identifica alguna de las imágenes que representan cada uno de los avatares en el rango de visión, se lanza la nueva escena (Maze) donde se carga el laberinto según el modo y la dificultad seleccionada. Las imágenes objetivo las podemos encontrar en los anexos 9.1 y 9.2 tanto para el Oso de Anteojos y el Tití Cabeciblanco respectivamente. En este orden de ideas el profesional encargado le entrega al niño el sistema completamente configurado para seleccionar el avatar y continuar con el juego.

De este modo en la última interfaz importante del sistema se muestra la ejecución de la partida. En esta se carga un laberinto nxn según la dificultad escogida: Fácil carga un laberinto 4x4, la dificultad Intermedio carga un laberinto 7x7 y finalmente un laberinto 10x10. Se carga el avatar escogido en la esquina inferior izquierda del laberinto generado o cargado según el modo escogido anteriormente. El juego inicia y el niño debe usar la inclinación del dispositivo para guiar al avatar (animal) a su lugar de hábitat. En la Figura 5.8 se observa la escena de juego con el personaje Oso de Anteojos seleccionado cursando un laberinto de dificultad 10x10.

Al trabajar con niños con discapacidad visual y auditiva, se probaron diferentes opciones para cambiar el color del fondo y aumentar el contraste de los objetos en la escena, al igual la adición de una luz sobre el avatar para indicar y facilitar la ubicación del jugador en el laberinto. Además,

se pensó añadir sonidos que se reproducen a modo de interacción con el jugador cuando el avatar impacta muy fuerte una pared del laberinto. Además de estos cambios escogió la esquina superior izquierda para añadir un contador de tiempo para indicar el tiempo que tarda el jugador en cruzar el laberinto y un contador de puntos que indica los choques del avatar con las paredes del laberinto. Se agregó un mensaje final indicando la terminación de la partida y los resultados obtenidos. En la Figura 5.9 podemos observar los cambios en el fondo y los indicadores de tiempo y puntaje durante el juego y en la Figura 5.10 observamos la finalización del videojuego.

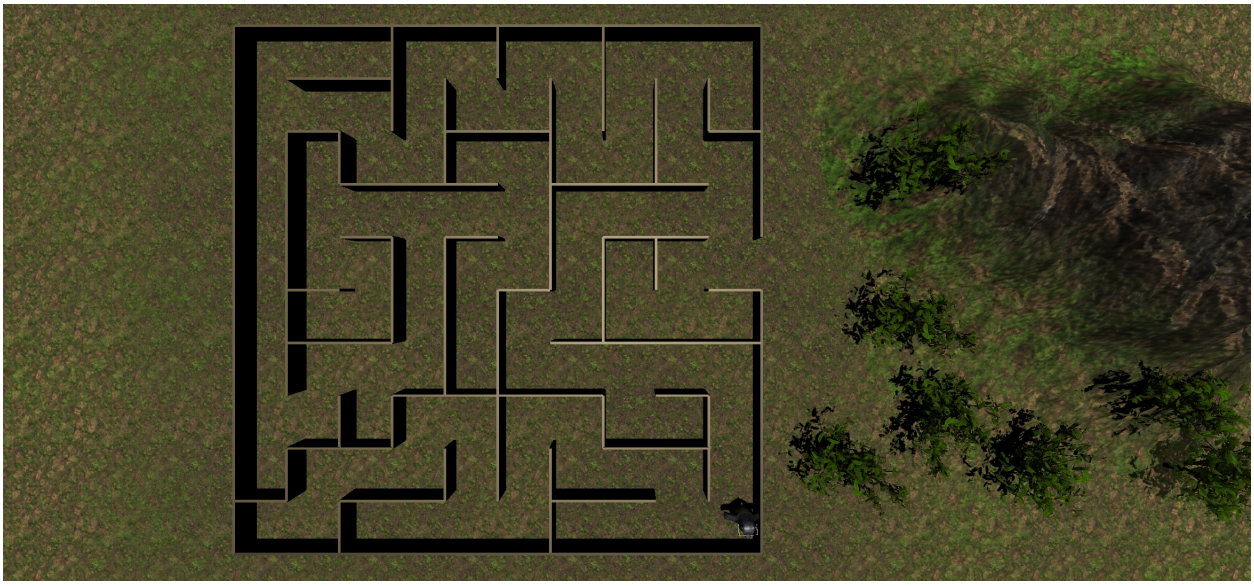

Figura 5.8: Escena Maze donde se cargan los diferentes laberintos en la versión inicial del sistema.

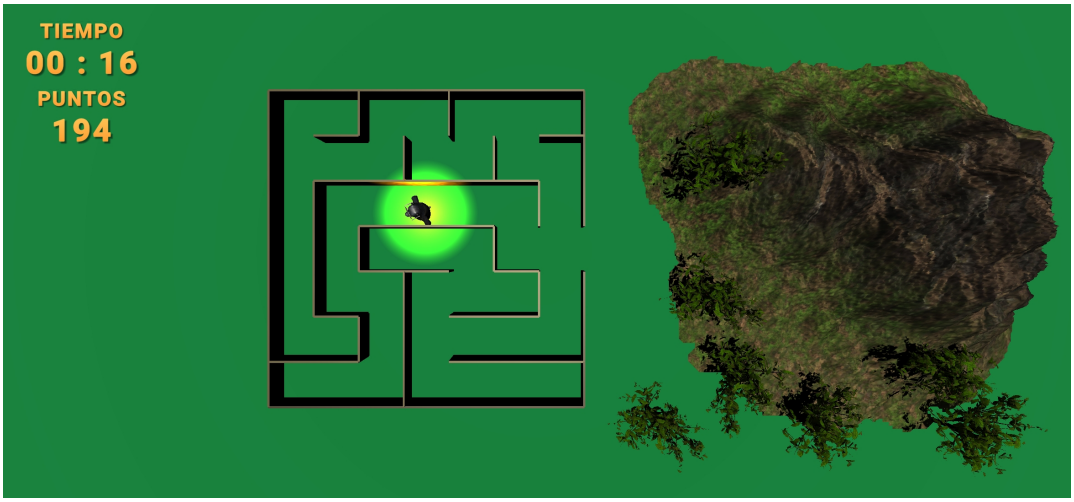

Figura 5.9: Escena Maze de partida con laberinto de nivel intermedio (7x7) con algunos cambios visuales

Figura 5.10: Escena Maze al finalizar una partida

5.5. Modelo de Datos

Debido a los alcances de este proyecto no se usa una base de datos para el almacenamiento de la información que debe persistir en el sistema. En la figura 5.11 podemos observar el modelo de datos relacional diseñado para cumplir con los objetivos de este proyecto. Si bien comparamos el diagrama de clases de la figura 5.1 con el modelo presentado en la figura 5.11, vemos que las tablas

también están presentes cómo clases del sistema, pero sus atributos son usados cómo campos para almacenar la información.

En la figura 5.11 la tabla/clase GameData es aquella almacenará toda la información de las partidas. Cada partida tiene un identificador, un identificador del jugador, el nombre del jugador, el tamaño del laberinto jugado, el nivel, los puntos y el tiempo obtenido en la partida. La información del jugador también debe ser almacenada y para esto se usa la clase PlayerData que almacena el id del jugador, el nombre, género y la edad. La relación entre las partidas y los jugadores es de muchos a uno, es decir que un jugador o registro de PlayerData puede estar relacionado en varias partidas o registros de GameData.

Por otro lado, están las tablas o clases MazeData y MazeCell en las cuales se va a realizar el almacenamiento de la información de cada uno de los laberintos que se deban almacenar. La clase MazeData contiene el valor del tamaño del laberinto, el nivel y una lista de listas del objeto MazeCell, que representa una matriz de celdas o habitaciones que puede tener el laberinto. La tabla o clase MazeCell almacena la posición x, y en la matriz del laberinto, cuatro booleanos que indican las direcciones en las que se encuentra o no una pared y finalmente dos booleanos que se usan para crear e inicializar el laberinto.

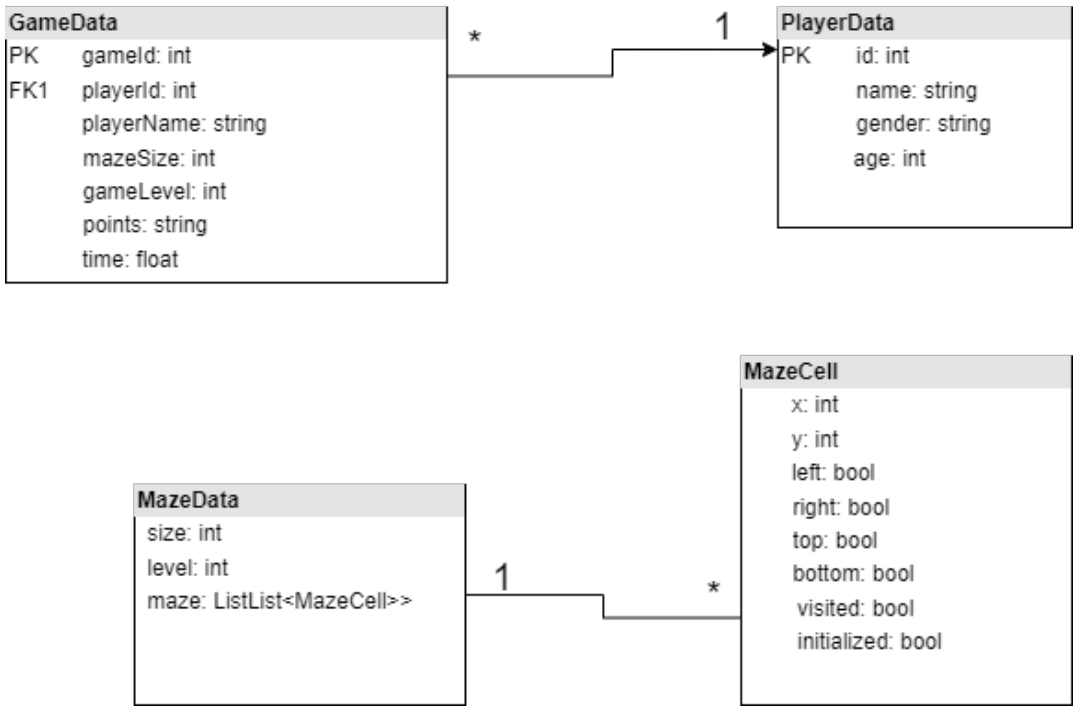

Figura 5.11: Modelo de datos relacional que representa los datos que se almacenan en el sistema.

5.6. Diseño del Algoritmo

El algoritmo de mayor complejidad y más importante en el sistema consiste en la función de generación de laberintos. La funcionalidad se encuentra en la clase MazeGenerator que se observa en el diagrama de clases en la sección de Diseño.

Existen dos tipos de laberintos [31]:

- Laberinto de conexión simple: Son aquellos laberintos que solo tienen un camino único el que llega a la salida es decir tiene una única solución y callejones sin salida. Normalmente la solución suele recorrer gran parte del laberinto, ver la Figura 5.12.

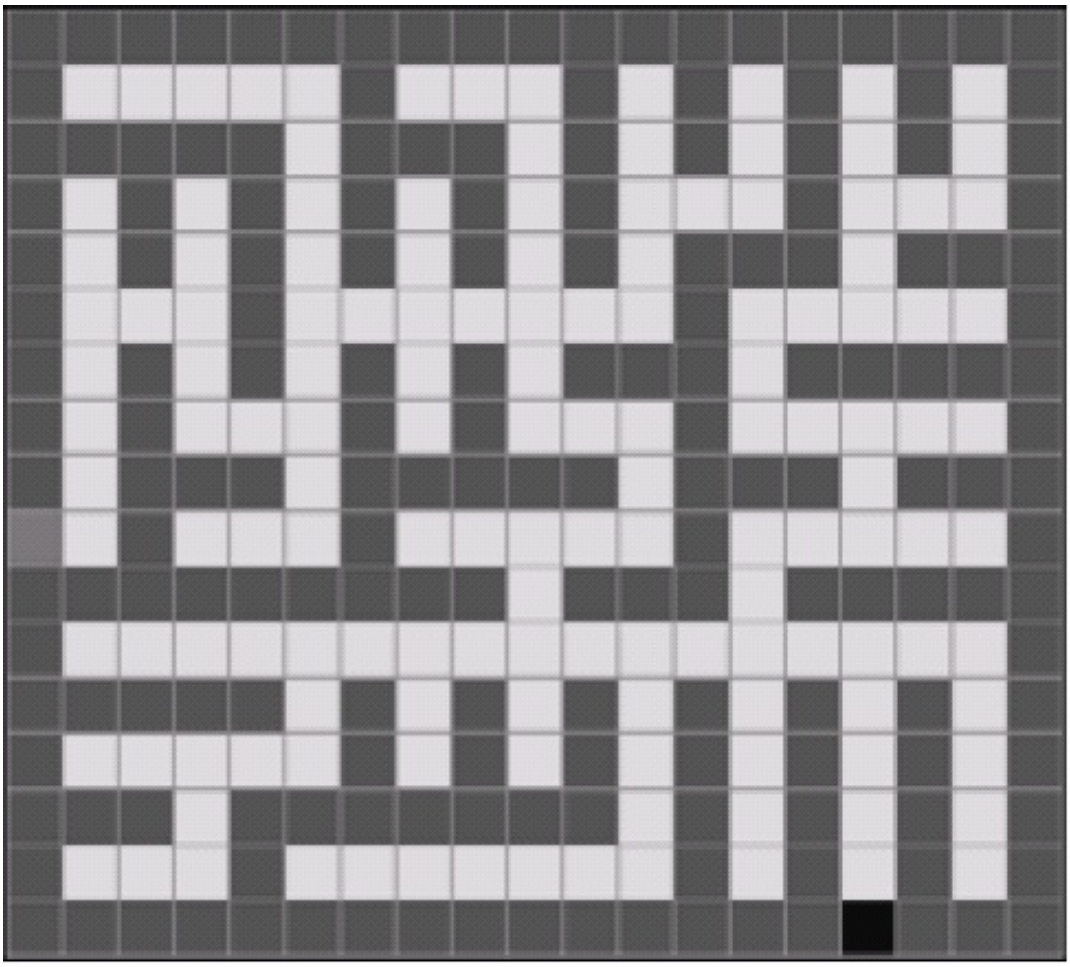

Figura 5.12: Ejemplo de un laberinto de conexión simple o LCS.

- Laberinto de conexión múltiple: Son laberintos que pueden tener múltiples caminos correctos para llegar al punto de salida. En este tipo de laberintos se requiere que el camino seleccionado

sea el más corto que cualquier otra posible solución, además de que existen ciclos que pueden llegar a dificultar la búsqueda, ver la Figura 5.13.

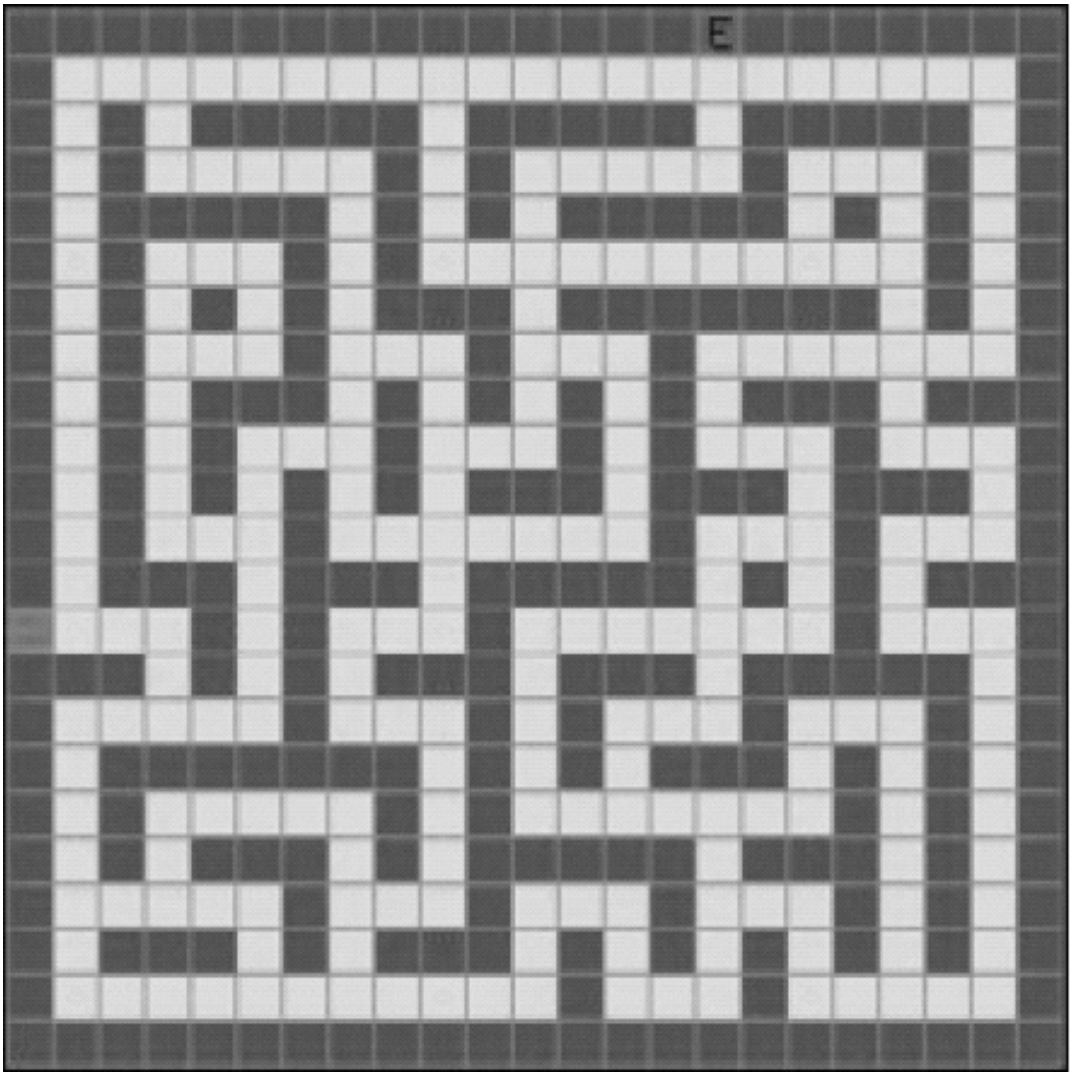

Figura 5.13: Ejemplo de un laberinto de conexión múltiple o LCM.

Para este proyecto se realizará un algoritmo que genera laberintos de conexión simple por lo que debemos abstraer la idea de laberinto a datos controlables. Debemos pensar en un laberinto como un grupo de nodos que se conectan entre sí por lo menos por un camino, de manera que si colocamos muros entre los nodos que no están conectados obtenemos un laberinto de conexión simple. La cantidad de nodos dependerá del tamaño que se desee generar el laberinto. Así podemos empezar a definir características descriptivas a cada nodo para representar su posición en el laberinto y

posibles caminos compartidos con sus nodos vecinos.

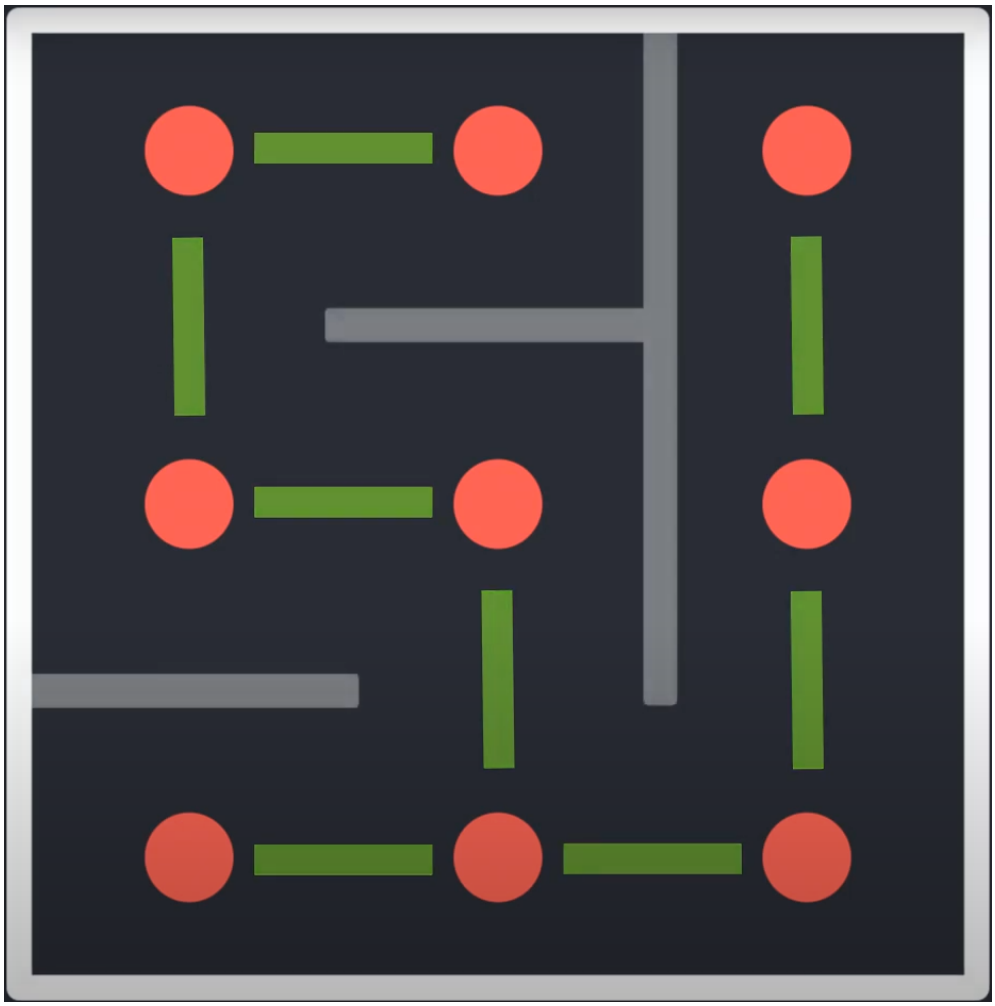

Figura 5.14: Ejemplo visual de generación de laberintos por el método de cavar túneles

En la figura 5.14 los puntos rojos representan los nodos, las líneas verdes representan los caminos entre nodos y las líneas grises entre los nodos representan muros entre nodos no conectados por caminos. El tamaño del laberinto es 3x3 lo que da un total de 9 nodos.

De este modo lo que el programa realizará es usar una estructura o matriz de nodos donde cada nodo es una celda o habitación con cuatro muros, un muro en cada dirección: arriba, abajo izquierda y derecha. Así que debe determinar que muros deben ser eliminados entre cada nodo para generar un camino continuo dentro del laberinto. A este método se le conoce como cavar túneles ya que va abriendo túneles a través de las paredes a lo largo y ancho del laberinto.

Para esta implementación se usará el algoritmo iterativo de backtracking con una pila de celdas.

Aquí el pseudocódigo del algoritmo:

1. De la matriz de celdas seleccionar la primera celda, marcarla como visitada y agregarla a la pila de celdas.
2. Mientras la pila de celdas no este vacía:
    2.1. Sacar una celda de la pila mediante pop y marcarla como celda actual.
    2.2. Verificar que la celda actual tenga celdas vecinas sin visitar. Si tiene celdas vecinas no visitadas entonces:
        2.2.1. Seleccionar una celda vecina aleatoria no visitada.
        2.2.2. Eliminar la pared que comparten la celda actual y la celda vecina seleccionada.
        2.2.3. Marcar celda vecina seleccionada como visitada.
        2.2.4. Agregar la celda actual a la pila.
        2.2.5. Agregar la celda vecina a la pila.

Capítulo 6

Implementación

6.1. Elección del motor de videojuegos

La elección de un motor de videojuegos depende de varios factores, como los requisitos del proyecto, la experiencia del equipo de desarrollo y las necesidades específicas del juego que se va a desarrollar. Unity, Unreal Engine y Godot son tres opciones populares en el ámbito de los motores de videojuegos. Unity destaca por su gran versatilidad, facilidad de uso y gran comunidad de desarrolladores. Es compatible con múltiples plataformas, ofrece una interfaz intuitiva y herramientas visuales. Por otro lado, Unreal Engine se destaca por su potente motor de renderización, efectos visuales avanzados y uso en juegos de mayor presupuesto. Ofrece soporte tanto para C++ como para Blueprints, un sistema de programación visual. Godot siendo de código abierto, ofrece una interfaz amigable, múltiples lenguajes de programación y una comunidad en crecimiento.

Si bien cada motor tiene sus ventajas, la elección de Unity puede ser preferible en ciertos casos. Unity es altamente versátil permitiendo el desarrollo de juegos para una amplia variedad de plataformas cómo: PC, consolas, dispositivos móviles y de realidad virtual o aumentada. Además, cuenta con una gran cantidad de recursos y una extensa comunidad de desarrolladores que pueden dar apoyo. El lenguaje de programación C# es ampliamente utilizado y estructurado lo que facilita el desarrollo y mantenimiento del código. Estas características hacen a Unity una opción sólida para este proyecto.

6.2. Líneas de código relevantes

6.2.1. Tipo abstracto de datos Cell

La función BuildMaze de la clase MazeGenerator recibe como parámetro la matriz de celdas inicializada y los enteros ancho y alto (width, height en inglés). La función retorna una matriz del tipo abstracto de dato definido Cell. Este tipo de dato o struct representará una celda del laberinto.

```
1 public struct Cell {
2     public int x;
3     public int y;
4 
5     public bool left;
6     public bool right;
7     public bool top;
8     public bool bottom;
```

```
9
10      public bool visited;
11      public bool initialized;
12  }
```

6.2.2. Generación del laberinto

La descripción del pseudocódigo del algoritmo se encuentra en la sección 5.6 Diseño del Algoritmo. A continuación, vemos como se usa la estructura Cell para generar el laberinto desde la clase MazeGenerator:

```
1   private static Cell[,] BuildMaze(Cell[,] matrix, int width, int height) {
2       Stack<Cell> cellStack = new Stack<Cell>();
3       Cell current = matrix[0, 0];
4       current.visited = true;
5       cellStack.Push(current);
6       while (cellStack.Count > 0) {
7           current = cellStack.Pop();
8           List<Cell> unvisited = GetUnvisitedNeighbours(matrix, current,
                width, height);
9           if (unvisited.Count > 0) {
10              int randomIndex = Random.Range(0, unvisited.Count);
11              Cell randomUnvisited = unvisited[randomIndex];
12              Cell[] updatedCells = RemoveSharedWalls(current,
                    randomUnvisited);
13              current = updatedCells[0];
14              randomUnvisited = updatedCells[1];
15              randomUnvisited.visited = true;
16              matrix[current.x, current.y] = current;
17              matrix[randomUnvisited.x, randomUnvisited.y] = randomUnvisited
                    ;
18              cellStack.Push(current);
19              cellStack.Push(randomUnvisited);
20          }
21      }
22      return matrix;
23  }
```

La función encargada de generar laberintos se llama BuildMaze, recibe cómo parámetro una matriz de estructuras Cell con todos los valores que representan los muros construidos entre todas las celdas del laberinto, el largo y el alto del largo del laberinto y retorna la matriz con las estructuras Cell inicializadas representando el laberinto final. Entre las líneas 2 y 5 se crea la pila vacía cellStack de objetos Cell, se selecciona la primera celda como la celda actual, se marca como visitada y se agrega a la pila. Luego se crea un ciclo en la línea 6 que se ejecuta mientras la pila contenga elementos. Dentro del ciclo se obtiene la celda seleccionada con una operación pop de la pila y

se obtiene una lista de celdas no visitadas vecinas a la celda seleccionada por medio de la función GetUnvisitedNeighbours. La función GetUnvisitedNeighbours recibe cómo parámetro la matriz que se está modificando, la celda seleccionada, el ancho y el alto del laberinto. En la línea 9 del código se verifica si la lista de celdas vecinas no visitadas contiene elementos; en caso de que el tamaño de la lista de celdas no visitadas sea mayor que cero, se realizan las siguientes operaciones:

- Se selecciona aleatoriamente un vecino de la celda actual no visitada de la lista (líneas 10 y 11)
- Se remueven los muros entre la celda seleccionada actualmente y la celda vecina no visitada seleccionada aleatoriamente. Esto se realiza por medio de la función RemoveSharedWalls que recibe la celda actual y la celda seleccionada aleatoriamente.
- La función nombrada en el punto anterior retorna las celdas que se le pasaron cómo parámetro en una lista con los dos elementos.
- En las líneas 13 y 14 se actualizan las variables que almacenan la celda actual y la celda seleccionada aleatoriamente.
- La celda seleccionada aleatoriamente se marca cómo visitada (línea 15) y se actualizan las celdas sin los muros compartidos en la matriz (líneas 16 y 17).
- Finalmente agregan a la pila la celda actual y la celda seleccionada aleatoriamente en ese orden.

De este modo mientras una celda tenga vecinos no visitados se seguirán añadiendo celdas a la pila. En el caso de que la lista de vecino no visitados de la celda seleccionada está vacía se ha recorrido todo el laberinto y así mismo la pila ha quedado vacía y finalmente se retorna la matriz que representa el laberinto.

6.2.3. Sistema de almacenamiento de información

Para almacenar la información no se desplegó una base de datos, como bien se definió anteriormente, sino que se usaron clases que se serializan a texto en formato binario y se almacenan en archivos planos. A continuación, se presenta el código que se usó para serializar la clase GameData a un formato binario y viceversa:

- Código para serializar el objeto GameData a formato binario y guardar en archivo.

```
1 public static void SaveGameData(GameData gameData, string fileName) {
2     BinaryFormatter formatter = new BinaryFormatter();
3     string filePath = Application.persistentDataPath + "/games/" +
          fileName;
4     FileStream stream = new FileStream(filePath, FileMode.Create);
5     formatter.Serialize(stream, gameData);
6     stream.Close();
7 }
```

El método presentado en el código recibe como parámetros un objeto tipo GameData que será serializado y el nombre del archivo en donde se almacenará la información. En la segunda línea se instancia un objeto BinaryFormatter que nos ayuda con la serialización del objeto más adelante. En la tercera línea del código se crea una variable tipo string que usa la ruta del directorio persistente de la aplicación (Application.persistetDataPath) y la concatena con el directorio "games" y el nombre que toma el archivo que se recibe cómo parámetro. El directorio "games" es donde se almacenarán todas las partidas jugadas en archivos con el nombre iniciado con la palabra "game" seguido por el identificador del juego y finaliza con el tipo de archivo ".game". De modo que el archivo donde se almacena la información de la partida con identificador número tres (3) tendrá como nombre: "game3.game". En la fila 4 del código se crea un objeto FileStream que nos ayuda a establecer una interfaz de lectura y escritura hacia el archivo especificado en el string filePath de la línea 3. Este objeto recibe un parámetro de crear el flujo en modo creación o escritura. Finalmente, en la línea 5 del código objeto tipo BinaryFormatter es usado para serializar el objeto GameData en el stream o flujo establecido hacia el archivo. Al final del método se cierra el archivo.

- Código para leer información de una partida desde un archivo plano en formato binario a un objeto GameData.

```
1  public static GameData LoadGame(int gameId){
2      string path = Application.persistentDataPath + "/games/game" +
           gameId.ToString() +  ".game";
3      if (File.Exists(path))
4      {
5          BinaryFormatter formatter = new BinaryFormatter();
6          FileStream stream = new FileStream(path, FileMode.Open);
7          GameData data = formatter.Deserialize(stream) as GameData;
8          stream.Close();
9          return data;
10     }
11     else
12     {
13         Debug.LogError("Archivo de jugador no encontrado. ");
14         return null;
15     }
16 }
```

Cómo se observa en el código presentado anteriormente la función LoadGame recibe cómo parámetro un entero que representa el identificador de la partida jugada. En la primera línea de código de la función se crea la ruta del archivo que se va a leer usando el parámetro de la función. De este modo en la línea 3 se verifica si existe esta ruta y en caso de que no exista, la función retorna un valor nulo (null, línea 14). En caso de que la ruta del archivo exista se crea un objeto BinaryFormatter, un objeto FileStream en modo lectura y escritura (FileMode.Open, línea 6). Finalmente se decodifica la información del formato binario a un

objeto GameData en la línea 7, se cierra el archivo y se retorna el objeto GameData con la información obtenida del archivo.

6.3. Proceso de desarrollo y detección de errores

El código inicial del sistema fue desarrollado por el ingeniero Martín Vladimir Alonso Sierra Galvis quien integró el código de generación de laberintos en Unity. Se programó una reunión con Martín en la cual se le preguntó respecto al funcionamiento inicial del sistema y como añadir los requerimientos al sistema. Este código se analizó y modificó a medida de los requisitos obtenidos.

Durante todo proceso de desarrollo existen múltiples errores con los que el programador debe lidiar. A medida que se fue adaptando el juego inicial y modificando el código fuente a los requisitos de los usuarios finales, aparecieron errores en el proyecto que no eran evidentes. Debido a que la aplicación fue pensada para dispositivos móviles con sistema operativo tipo Android, no existía otra manera de probar el sistema más que generando APK para instalarla en el dispositivo personal y realizar las pruebas. Aun así algunas veces después de realizar cambios en el código y generar el APK muchos de los cambios y funcionamientos esperados no se evidenciaban o no tenían el funcionamiento correcto pero no existía manera de depurar el código una vez instalado en el dispositivo por lo que se agregó un texto de color rojo a modo de depuración en las versiones que se probaban cómo se muestra en la figura 6.1.

Figura 6.1: Menú inicial con el texto de depuración que se ve en la parte izquierda de la figura

Capítulo 7

Pruebas

7.1. Descripción de las pruebas

Para probar el sistema se realizaron pruebas de usuario en el instituto donde se pudieron realizar 14 pruebas de partidas con cinco profesionales encargados presentes y siete niños del instituto. De estas pruebas se tomaron tres datos importantes que son los resultados de las partidas jugadas, una encuesta a los niños de su opinión del juego y una encuesta para los educadores en donde pueden dar observaciones más puntuales y adaptativas al sistema.

Las pruebas de usuario sobre la aplicación nos ayudan a obtener valoraciones y comentarios adicionales sobre la experiencia de los usuarios y por medio de la observación directa se toma nota de las posibles dificultades y errores que tuvieron durante la ejecución de la aplicación. Los resultados de las partidas jugadas en las pruebas quedan almacenados en el sistema dando así indicadores del desempeño de los niños al cruzar los diferentes laberintos, pero además nos puede llegar a indicar que tan fácil o difícil fue cruzar el mismo laberinto para cada uno de los niños con diferentes discapacidades.

La información recolectada de los jugadores en relación a los resultados obtenidos en las partidas nos puede llegar a indicar cómo las dificultades y deficiencias auditivas o visuales pueden afectar la capacidad de uso de la aplicación.

Finalmente, la encuesta a los educadores y profesionales del instituto nos ayuda a validar la utilidad de la aplicación cómo medio de educación ambiental, uso para evaluar el desarrollo de cognitivo de los niños y la facilidad de uso. Además, la encuesta nos ayuda obtener comentarios de que aspectos del sistema se podría cambiar para mejorar la interacción y el aprendizaje.

7.2. Resultados de partidas

Los resultados de las partidas de los jugadores se puede ver representada en la tabla de la Figura 7.1. En total se realizaron catorce pruebas y en la segunda columna de la tabla vemos el identificador de cada uno de los jugadores que realizaron las pruebas. La información de las discapacidades de los jugadores se presenta en la Figura 7.2. En la tercera columna de la tabla vemos la dificultad en la que se ejecutó la prueba, por ejemplo, el jugador 1 en la prueba 1 jugó en la dificultad Fácil, en la prueba 2 en dificultad Difícil y en la prueba 3 en Fácil. En la penúltima y última columna se ven el tiempo y el puntaje obtenido respectivamente. También podemos obtener el promedio de tiempo y puntaje obtenido por cada nivel de dificultad. El promedio del tiempo de las partidas jugadas en Fácil es de 43 segundos, el de las partidas jugadas en nivel Intermedio fue 182 segundos y la el de

las partidas ejecutadas en Difícil fue 60 segundos. En Fácil el tiempo mínimo fue de 8,6 segundos y el máximo fue 108,3 segundos, esto nos muestra cómo las dificultades de cada niño pueden afectar sus resultados en el juego. Otro ejemplo de esto es evidente al observar las tres partidas jugadas en nivel Intermedio de las cuales un jugador cruzó el laberinto en 29 segundos y los otros dos en 225 y 292 segundos. Respecto a los puntajes podemos decir que para el nivel Fácil el promedio obtenido fue 89 puntos de 100. En nivel Intermedio el promedio de puntaje obtenido fue de 159 de 200 y en nivel Difícil fue de 254 puntos sobre 300. Como bien se ha dicho antes, el puntaje indica la cantidad de golpes que el avatar recibe contra las paredes del laberinto. El promedio total de impactos en nivel de dificultad Fácil fue de 11 impactos aproximadamente, el de dificultad Intermedio fue 40 y el Difícil fue de 46 impactos, si bien es evidente que la dificultad de juego aumenta al añadir esta condición donde los jugadores deben evitar impactar las paredes.

Es importante explicar que todos los niños que jugaron las pruebas no realizaron partidas en todas las dificultades ya que el profesional encargado seleccionaba la dificultad de la partida según las características y capacidades visuales, auditivas y edad del jugador.

# De Juego	ID Niño / Jugador	Nivel de Dificultad	Tiempo (segundos)	Puntaje
1	1	Fácil	13,55481	86/100
2	1	Difícil	68,87345	238/300
3	1	Fácil	8,604568	95/100
4	2	Intermedio	225,0263	132/200
5	2	Fácil	53,31775	79/100
6	3	Fácil	59,35743	92/100
7	3	Intermedio	29,3531	176/200
8	4	Fácil	25,55617	91/100
9	4	Difícil	46,61036	256/300
10	5	Fácil	32,36583	94/100
11	5	Difícil	73,41541	269/300
12	6	Difícil	51,51426	253/300
13	6	Intermedio	29,2437	170/200
14	7	Fácil	108,394	87/100

Figura 7.1: Tabla de resultados obtenidos por los jugadores en las pruebas

7.3. Información jugadores y preguntas

La encuesta sobre la opinión de los niños se hizo de manera directa tomando nota de sus respuestas. En la tabla de la Figura 7.2 podemos ver algunas de sus respuestas más las discapacidades de los niños.

ID Niño / Jugador	¿Te gusto el juego?	¿Te pareció facil, normal dificil?	¿Algúna cosa adicional que te gustaía cambiar o un comentario?	Diagnostico	Edad	Grado
1	Sí	Fácil	Quiere un laberinto más difícl	Retinosquisis, ve mejor por el ojo izquierdo	8 años	1ro
2	Sí	Fácil	Es de concentración.	Cataratas ojo derecho, afasia ojo derecho, hidrocefalia y toxoplasmosis congenito, posible glaucoma	8 años	1ro
3	Sí	Fácil	Sin comentarios	Sin discapacidad	6 años	Transición
4	Sí	Difícil	Sin comentarios	Hipoacusia bilateral moderada	9 años	2do
5	Sí	Difícil	Le gustaron los diferentes animales que tiene el laberinto	Hipoacusia severa	9 años	1ro
6	Sí	Difícil	Agregaría más animales	Discapacidad auditiva moderada	10 años	Transición
7	Sí	Difícil	Sin comentarios	Discapacidad visual por hipoplasia del Nervio óptico	7 años	Transición

Figura 7.2: Tabla con la información de los jugadores y las preguntas realizadas a los jugadores

Cómo podemos observar en la Figura 7.2, a todos los participantes de las pruebas les gustó el juego, a tres les pareció fácil el juego y a cuatro les pareció difícil. Entre los participantes se encontraban las diferentes condiciones: Retinosquisis, cataratas, hidrocefalia, toxoplasmosis, hipoacusia, hipoplasia del nervio óptico, entre otras. El jugador número 3 no tiene ninguna discapacidad. De los resultados de las pruebas de los jugadores y sus discapacidades podemos deducir a modo general que los jugadores con discapacidad auditiva obtuvieron mejores resultados que lo jugadores con visión reducida. Por ejemplo, entre los jugadores 1 y 2 el promedio de los puntos de las partidas en nivel Fácil fue de 86 puntos aproximadamente, ambos jugadores con discapacidades como retinosquisis en el primero y cataratas, afasia e hidrocefalia en el segundo. Mientras que el promedio de puntaje obtenido por los jugadores 4 y 5 en la misma dificultad es de 92,5 puntos. Los jugadores 4 y 5 tienen una discapacidad conocida cómo hipoacusia, severa y moderada respectivamente. También

vale la pena resaltar que para la prueba del jugador 7 fue necesario apagar las luces del cuarto donde se realizaron las pruebas ya que su discapacidad visual implica que su campo de visión es muy reducido los reflejos de las luces afectaban la interacción y visualización de los objetos del videojuego.

7.4. Encuesta a profesionales del Instituto para niños Ciegos y Sordos del Valle del Cauca

Para la encuesta de los profesionales encargados se realizó un formulario de Google para obtener las respuestas fácilmente. A continuación, vemos las preguntas con las respuestas generales de la encuesta:

- La pregunta ¿Considera útil el uso del videojuego cómo medio de educación ambiental? nos ayuda a validar la finalidad del sistema y del proyecto colaborativo Colombia-Quebec, Narrativa, Realidad virtual y Discapacidad Sensorial, que es la creación de experiencias y narraciones inmersivas por medio de tecnologías y estrategias transmediales para promover la educación ambiental y la fauna colombiana en los niños. Los los participantes de la encuesta deben dar una valoración a la utilidad del videojuego como medio educativo, donde 1 es poco útil y 5 es muy útil. Podemos ver el gráfico de respuestas en la Figura 7.3 donde cuatro de cinco personas escogieron la opción 5 y uno escogió la opción 4, considerando así el videojuego útil cómo medio de educación ambiental.

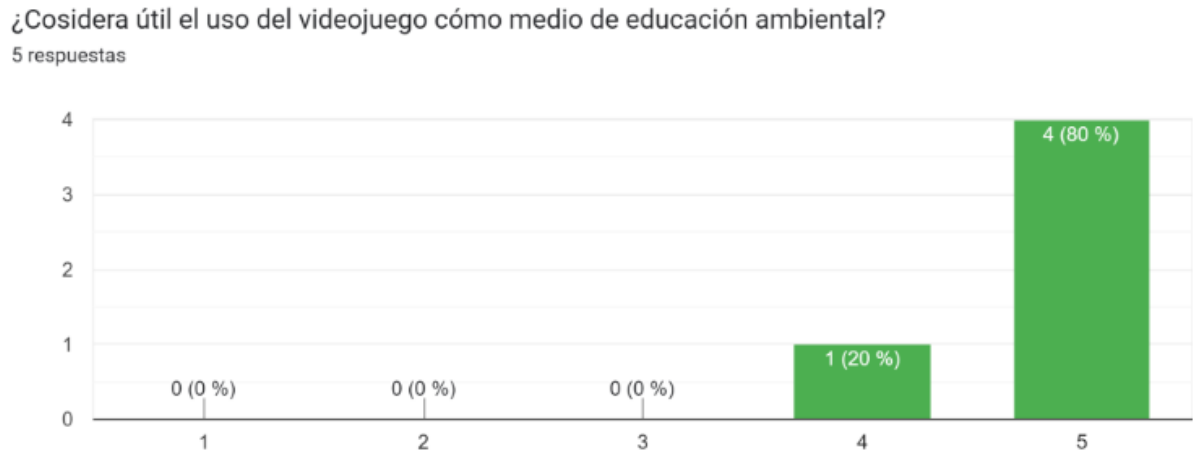

Figura 7.3: Pregunta 1 - encuesta a profesionales encargados

- La pregunta ¿Lo usaría para evaluar el desarrollo cognitivo de los niños? busca validar que los profesionales encargados tengan una herramienta interactiva útil adicional para evaluar

diferentes aspectos cognitivos de los niños con discapacidad. los resultados e evidencian en la Figura 7.4

Figura 7.4: Pregunta 2 - encuesta a profesionales encargados

- A pesar de que las cinco personas que respondieron la encuesta consideran el sistema fácil de utilizar (cómo se observa en la Figura 7.5, a nivel de observación se notó algunas dificultades que se evidencian más adelante la subsección 7.5.

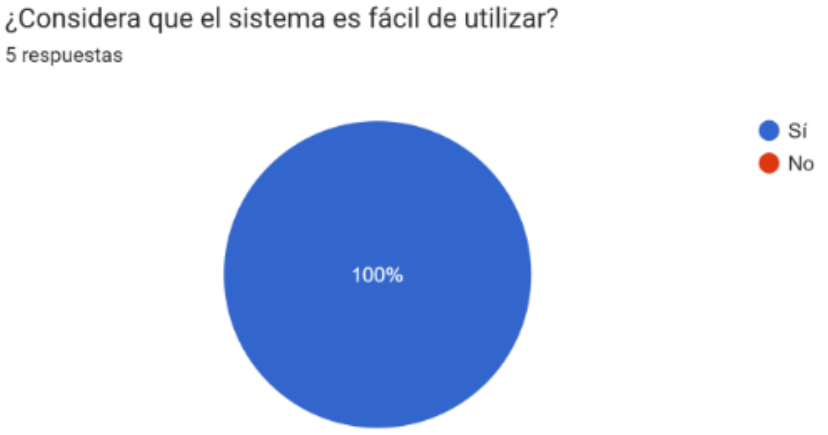

Figura 7.5: Pregunta 3 - encuesta a profesionales encargados

- Para finalizar se les realizo una pregunta abierta a los profesionales encarados presentes en las pruebas donde tenían la libertad de realizar comentarios sobre cambios o mejoras que

se le podrían llegar a realizar al sistema. Las respuestas se evidencian en la Figura 7.6. En estas respuestas también podemos llegar a evidenciar algunas dificultades que tuvieron los jugadores y que se ve representado en la primera respuesta. Al parecer fue evidente la necesidad de una explicación a modo de tutorial dentro del juego para los nuevos jugadores. La segunda respuesta hace referencia a una posible modificación para mejorar el proceso educativo que propone cambiar el final de cada partida de modo que cuando el jugador complete el laberinto aparezcan diferentes opciones de hábitats y el jugador debe llevar al animal al hábitat correcto. Además, propone que el niño deba responder que animal es y dejar registradas estas respuestas. De la tercera respuesta se destaca que debería existir una forma de cambiar los contrastes o colores del juego de modo que se pueda adaptar a niños con diferente capacidad visual. Al igual que la respuesta 5 también propone aumentar el tamaño de los textos y revisar el contraste y los diferentes colores. Finalmente la respuesta 4 en la Figura 7.6 hace referencia al tamaño del laberinto cuando se ejecutan las dificultades Fácil e Intermedio ya que entre el borde de la pantalla y el laberinto queda un espacio vacío.

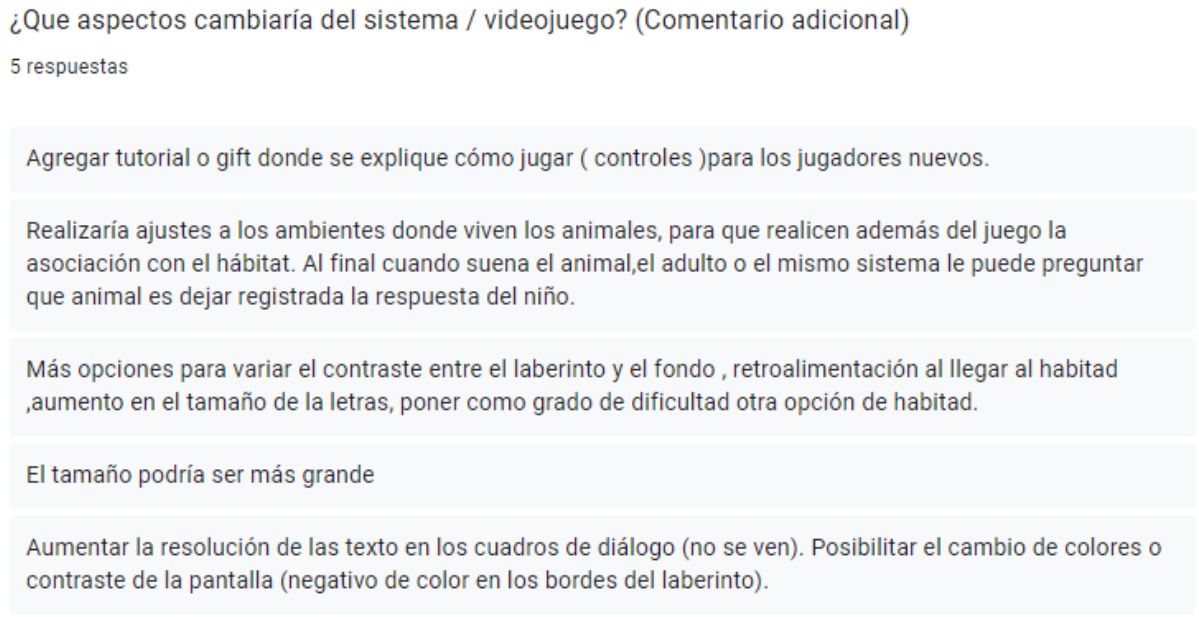

Figura 7.6: Pregunta 4 - encuesta a profesionales encargados

7.5. Comentarios adicionales

Durante las pruebas los observadores presentes se acercaban a dar comentarios frente a las dificultades que presentaban los usuarios y posibles adaptaciones o adiciones a la implementación del sistema. A continuación, la lista de comentarios:

- Aumentar la sensibilidad con la que se reproduce el sonido cuando el avatar se impacta contra las paredes del camino del laberinto.
- Añadir vibración del dispositivo móvil cuando el avatar impacta contra una pared del laberinto.
- Cuando finaliza cada partida o juego, el usuario en vez de esperar un tiempo para volver al menú principal, añadir un botón para volver.
- En el menú inicial, al crear un nuevo usuario, modificar el tamaño de la lista del campo para seleccionar el género del jugador.
- Añadir indicaciones para el usuario profesional encargado en el menú inicial.
- Añadir una opción que permita cambiar los contrastes y los colores de la escena de juego para niños con diferentes discapacidades visuales.
- Añadir al sistema la capacidad de almacenar la trayectoria que toma el jugador dentro del laberinto.
- Añadir indicaciones iniciales de usar la inclinación del dispositivo antes de empezar cada juego.
- Si un jugador queda "atascado" o queda parado mucho tiempo en un lugar del laberinto añadir un indicador de ayuda visual.
- Acercar la cámara de la escena Maze para que el laberinto quede más grande en las dificultades fácil e intermedio.
- Para mejorar el aprendizaje de los niños, modificar el sistema para que cuando el niño termine el laberinto, mostrar dos o tres hábitats diferentes y el jugador debe dirigir el avatar al hábitat correspondiente al animal escogido antes de jugar.

Capítulo 8

Conclusiones y trabajos futuros

8.1. Conclusiones

- A partir del proceso realizado se puede evidenciar que se lograron completar los objetivos propuestos para el proyecto tal como lo indica la sección 1.2.
- Se logró integrar un sistema de generación de laberintos desarrollado en lenguaje de programación C# para el videojuego desarrollado en Unity.
- Por medio de la metodología Scrum fue posible cumplir los objetivos propuestos en la sección 1.2 permitiendo así el desarrollo continuo teniendo en cuenta las limitaciones de tiempo y alcances del proyecto.
- Gracias a la investigación desarrollada, la constante retroalimentación con el Instituto por medio del Dr. Andrés Adolfo Navarro Newball y al uso de las herramientas de desarrollo de entornos visuales como medio interactivo se logró crear un videojuego tipo laberinto inclusivo que, de acuerdo con las observaciones y respuestas de los profesionales del Instituto para Niños Ciegos y Sordos del Valle del Cauca el prototipo tiene el potencial de apoyar el proceso de evaluar el desarrollo cognitivo de los niños.
- El código preexistente realizado por Martín Vladimir Alonso Sierra presentó una ventaja en las etapas de diseño y desarrollo ya que se encontraba debidamente comentado y desarrollad bajo las buenas prácticas de programación lo que permitió el fácil entendimiento y modificación para cumplir los objetivos.
- Se lograron cumplir en su totalidad la mayoría de los requerimientos definidos inicialmente. No se logró crear un sistema de puntaje basado la toma de caminos incorrectos.
- Por medio de las pruebas se concluye que el prototipo del videojuego desarrollado tiene el potencial de evaluar el desarrollo cognitivo de los niños con discapacidades visuales y auditivas del Instituto para Niños Ciegos y Sordos del Valle del Cauca, además de apoyar a los mismos en su proceso de aprendizaje y de adaptación al entorno.

8.2. Trabajos futuros

- Agregar modificaciones y evaluar comentarios detallados en la sección 7.5.
- En conjunto a un colaborador de la salud (psicólogo, profesor, evaluador, pediatra, entre otros) evaluar cómo medir otros elementos cómo la percepción espacial, la coordinación motora del jugador y el proceso de adaptación al sistema.
- La información que se almacenan en archivos puede llegar a ser almacenada en una base de datos o en un sistema en la nube de modo que se use un sistema centralizado de datos y si se instala el prototipo en varios dispositivos se consulte la información a ese sistema centralizado.
- A partir de los datos obtenidos en las pruebas generar diferentes gráficos para que sean analizados por los especialistas del instituto y tener una manera visual de evaluar el progreso.
- Agregar un sistema que dibuje el camino que realiza el jugador a través del laberinto de modo que se pueda a agregar a los datos que revisan los especialistas.
- Finalmente es necesario extender las pruebas por un periodo de tiempo más largo para confirmar si la aplicación realmente permite evaluar y hacer seguimiento del desarrollo de los niños.

Bibliografía

[1] "Colombia, el segundo país más biodiverso del mundo, celebra el día mundial de la biodiversidad," *Ministerio de Ambiente y Desarrollo Sostenible*, 2019. Disponible en: https://www.minambiente.gov.co/bosques-biodiversidad-y-servicios-ecosistemicos/colombia-el-segundo-pais-mas-biodiverso-del-mundo-celebra-el-dia-mundial-de-la-biodiversi

[2] H. G. Romero, "Deforestación en colombia: Retos y perspectivas," *FEDESARROLLO*, 2019. Disponible en: https://www.repository.fedesarrollo.org.co/bitstream/handle/11445/337/KAS%20SOPLA_Deforestacion%20en%20Colombia%20retos%20y%20perspectivas.pdf?sequence=2&isAllowed=.

[3] H. Hoag, "Confronting the biodiversity crisis," *Nature Climate Change*, vol. 1, no. 1, pp. 51–54, 2010. Disponible en: https://www.nature.com/articles/climate.2010.38.

[4] "La importancia de la educación ambiental," Agencia de Protección Ambiental de Estados Unidos (EPA), 2022. Encontrado en: https://espanol.epa.gov/espanol/la-importancia-de-la-educacion-ambiental#es.

[5] "Boletines poblacionales: Personas con discapacidad - pdc," *Ministerio de Salud y Protección Social*, 2020. Oficina de Promoción Social. Disponible en: https://www.minsalud.gov.co/sites/rid/Lists/BibliotecaDigital/RIDE/DE/PS/boletines-poblacionales-personas-discapacidadI-2020.pdf.

[6] L. S. y A. Rodríguez, "Proyecto colaborativo colombia-quebec, narrativa, realidad virtual y discapacidad sensorial." https://www.youtube.com/watch?v=ZDrwQY2YOJA&ab_channel=Direcci%C3%B3ndeInvestigaciones, 2015. Encontrado el 05/04/2023.

[7] D. Tilman, F. Isbell, and J. M. Cowles, "Biodiversity and ecosystem functioning," *Annual Review of Ecology, Evolution, and Systematics*, vol. 45, no. 1, pp. 471–493, 2014.

[8] M. Motta, C. Lucía, and N. A. S. Cardozo, *Educación inclusiva: Conceptos y rutas*. Bogotá, Colombia: Editorial Pontificia Universidad Javeriana, 2013.

[9] Cambridge Dictionary, "labyrinth." https://dictionary.cambridge.org/dictionary/english/labyrinth, (s.f.). Encontrado el 9 de marzo de 2023.

[10] J. M. Sadurni, "La leyenda del minotauro, el terrorífico monstruo mitad hombre y mitad toro." https://historia.nationalgeographic.com.es/a/la-leyenda-del-minotauro-el-terrorifico-monstruo-mitad-hombre-y-mitad-toro-_19205, (s.f.). Encontrado el 9 de marzo de 2023.

[11] J. Huizinga, *Homo Ludens*. Madrid, España: Alianza Editorial/Emencé Editores, 1938.

[12] F. S. Marcos, "Las definiciones del juego," *Revista Española de pedagogía*, vol. 36, no. 142, 1978. Disponible en: https://revistadepedagogia.org/xxxvi/no-142/las-definiciones-del-juego/101400049649/.

[13] D. S. Levis Czernik, "Los videojuegos: cuando mirar también es hacer," *Comunicación y pedagogía: nuevas tecnologías y recursos didácticos*, no. 152, pp. 71–78, 1998.

[14] "Las discapacidades y la salud: Información básica," *Centros para el control y la Prevención de enferemedades*, 2020. Disponible en: https://www.cdc.gov/ncbddd/spanish/disabilityandhealth/disability.html.

[15] A. P. Roncancio-Ortiz, M. F. Ortiz-Carrera, H. Llano-Ruiz, M. J. Malpica-López, and J. J. Bocanegra-García, "El uso de los videojuegos como herramienta didáctica para mejorar la enseñanza-aprendizaje: una revisión del estado del tema," *Revista Ingeniería Investigación y Desarrollo*, vol. 17, no. 2, pp. 36–34, 2017.

[16] Bejarano Bejarano, Olga Lucía and Vargas Díaz, Janeth, "Orientaciones pedagógicas para la atención y la promoción de la inclusión de niñas y niños menores de seis años con discapacidad auditiva." ICBF - recurso virtual:https://www.icbf.gov.co/sites/default/files/cartilla-auditiva-4.pdf, 2010. Pág. 12. Encontrado el 9 de marzo de 2023.

[17] Fundación UNICAP, "Discapacidad auditiva: Pérdida de audición." https://www.fundacionunicap.org/discapacidad-auditiva/, 2017. Encontrado el 9 de marzo de 2023.

[18] Sistema Nacional DIF Gobierno de México, "¿qué es la discapacidad auditiva?." https://www.gob.mx/difnacional/articulos/que-es-la-discapacidad-auditiva, 2017. Encontrado el 9 de marzo de 2023.

[19] A. C. González Saucedo, F. J. García Heredia, and R. Ramírez Martínez, "Discapacidad visual," *Cultura Científica y Tecnológica*, mar. 2016.

[20] World Health Organization, "Ceguera y discapacidad visual." https://www.who.int/es/news-room/fact-sheets/detail/blindness-and-visual-impairment, 2022. Encontrado el 9 de marzo de 2023.

[21] Alarcon Flores, Yael and Garcia Hernández, Araceli, "Apuntes digitales: Programación web." Universidad Autónoma del Estado de Hidalgo: http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro40/index.html, 2021. Encontrado el 9 de marzo de 2023.

[22] Unity, "Unity: Developer tools." https://unity.com/developer-tools. Encontrado el 9 de marzo de 2023.

[23] M. Sierra, J. Martínez, G. Alvarez, and D. Linares, "Un videojuego para apoyar la terapia del lenguaje en niños con discapacidad auditiva. el caso de la descripción dinámica," in *CoTH 2019-Workshop on Computing and Technology in Health*, 2019.

[24] N. Nasiri, S. Shirmohammadi, and A. Rashed, "A serious game for children with speech disorders and hearing problems," in *2017 IEEE 5th International Conference on Serious Games and Applications for Health (SeGAH)*, pp. 1–7, 2017.

[25] J. A. Ferreyra, A. Méndez, and M. Rodrigo, "El uso de las tic en la educación especial: descripción de un sistema informático para niños discapacitados visuales en etapa preescolar," *Revista Iberoamericana de Tecnología en Educación y Educación en Tecnología*, p. p. 55–62, ene. 2009.

[26] "Iso/iec/ieee international standard - systems and software engineering — developing information for users in an agile environment," *ISO/IEC/IEEE 26515:2018(E)*, pp. 1–32, 2018.

[27] F. Hayat, A. U. Rehman, K. S. Arif, K. Wahab, and M. Abbas, "The influence of agile methodology (scrum) on software project management," in *2019 20th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, pp. 145–149, 2019.

[28] D. Rosenberg and M. Stephens, *Use Case Driven Object Modeling with UML: Theory and Practice*. Addison-Wesley Professional, 2005.

[29] Unity Technologies, "Creating scenes." https://docs.unity3d.com/es/530/Manual/CreatingScenes.html, 2016.

[30] "Unity - scripting api: Monobehaviour." https://docs.unity3d.com/ScriptReference/MonoBehaviour.html. Accedido el 12 de abril de 2023.

[31] V. Tomás Mariano, M. Pozas Cárdenas, and J. Hernández Camacho, "Propuesta para la generación de laberintos ampliados en 2d." Página web de la Universidad Autónoma del Estado de Hidalgo, Sin fecha.

Capítulo 9

Anexos

9.1. Anexo 1

En la figura 9.1 podemos observar la imagen objetivo que se usa para seleccionar el avatar Oso de Anteojos desde la escena AR. Imagen obtenida de alamy.com.

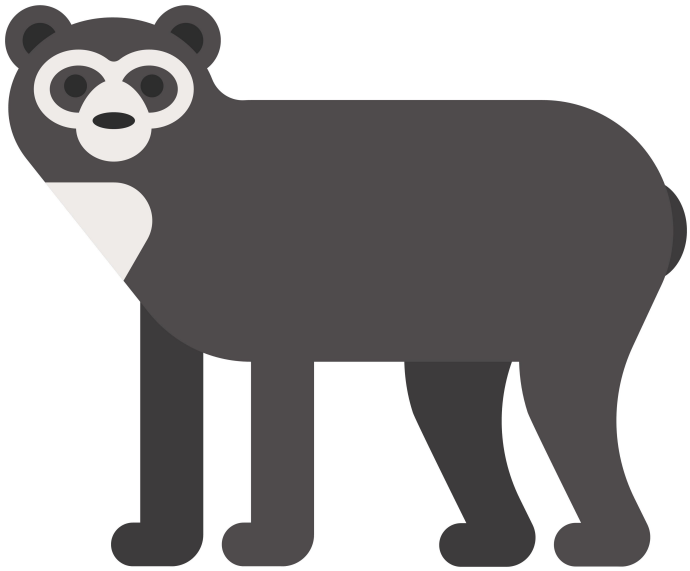

Figura 9.1: Anexo 1: Imagen objetivo Oso de Anteojos

9.2. Anexo 2

En la figura 9.2 podemos observar la imagen objetivo que se usa para seleccionar el avatar del Tití Cabeciblanco desde la escena AR. Imagen obtenida de alamy.com.

Figura 9.2: Anexo 2: Imagen objetivo Tití Cabeciblanco