ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN SISTEMAS

"DISEÑO E IMPLEMENTACIÓN DE UN SERIOUS GAMES CON TÉCNICAS DE INTELIGENCIA ARTIFICIAL PARA EL DISEÑO DE UN CURSO INTERACTIVO 3D DE INTRODUCCIÓN A UNITY"

TRABAJO DE TITULACIÓN

Tipo: **PROYECTO TÉCNICO**

Para optar para al Grado Académico de:

INGENIERA EN SISTEMAS INFORMÁTICOS

AUTOR: CHRISTIAN DAVID VIRACOCHA SUNTAXI

TUTOR: ING. VICTOR FERNANDO PROAÑO BRITO

Riobamba – Ecuador
2018

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO
FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
ESCUELA DE INGENIERÍA EN SISTEMAS

El Tribunal del Trabajo de Titulación certifica que: El Proyecto Técnico: "DISEÑO E IMPLEMENTACIÓN DE UN SERIOUS GAMES CON TÉCNICAS DE INTELIGENCIA ARTIFICIAL PARA EL DISEÑO DE UN CURSO INTERACTIVO 3D DE INTRODUCCIÓN A UNITY", de responsabilidad del señor Christian David Viracocha Suntaxi, ha sido minuciosamente revisado por los miembros del Trabajo de Titulación, quedando autorizada su presentación.

	FIRMA	FECHA
Ing. Alonso Álvarez **DELEGADO VICEDECANO**	_____	_____
Ing. Fernando Proaño **DIRECTOR DEL TRABAJO DE TITULACIÓN**	_____	_____
Ing. Patricio Moreno **MIEMBRO DEL TRIBUNAL**	_____	_____

"Yo Christian David Viracocha Suntaxi soy el responsable de las ideas, doctrinas y resultados expuestos en este Trabajo de Titulación y el patrimonio intelectual del mismo pertenecen a la Escuela Superior Politécnica de Chimborazo".

Christian David Viracocha Suntaxi

DEDICATORIA

Dedico este trabajo a mi madre Marlene Suntaxi, mi padre Manuel Viracocha y a DIOS por sobre todas las cosas, por ser parte de mi vida y ser los pilares fundamentales para la construcción de mi carrera profesional. A ellos, porque sembraron en mí los valores de la responsabilidad y el deseo de superación para realizar todo lo que me he propuesto y llegar a la meta con éxito.

Christian David

AGRADECIMIENTO

Mi agradecimiento siempre a DIOS por ser mi guía espiritual e iluminarme con su sabiduría y llegar alcanzar mi sueño y ahora hecho realidad.

Gracias a mis padres por su apoyo incondicional y confianza que me dieron al estar lejos de ellos.

A la ESCUELA SUPERIOR POLITÉCNICA DEL CHIMBORAZO y a mis Maestros por enseñarme e inculcar en mí, los conocimientos necesarios para ser una persona útil para la sociedad.

Christian David

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

ÍNDICE DE ANEXOS

ÍNDICE DE ABREVIATURAS

ESPOCH	Escuela Superior Politécnica del Chimborazo
NPC	Non Person Controller
IA	Inteligencia artificial
RPG	Role Playing Game
MEF	Máquinas de Estado Finito
DEF	Diagramas de Estado Finito
RNA	Redes Neuronales Artificiales
GOAP	Goal Oriented Action Planning
RV	Realidad Virtual
PC	Personal Computer
CPU	Central Processor Unit
RAM	Random Access Memory

RESUMEN

El propósito del presente trabajo de titulación es el diseño e implementación del Serious Game ¨Unity City¨ el cual incorpora elementos de inteligencia artificial dentro de la lógica y el ambiente del juego, para impartir un curso interactivo en 3D de Introducción a Unity 3D, con la finalidad de que los estudiantes aprendan el manejo y desarrollo de videojuegos en el motor de Unity.
El proyecto está destinado para el Centro de Tutorías y la materia de Interfaces y Multimedia de la Escuela de Informática y Electrónica de la ¨ESPOCH¨, para lo cual se aplicó la metodología SUM, diseñado para grupos pequeños y la mejora en tiempos y calidad de desarrollo, el cual se divide en fases, que va desde el planteamiento del concepto del juego hasta llegar a la fase del Producto Beta y la liberación del juego. Se creó una ciudad virtual con la información más relevante para mejorar la capacidad de aprendizaje de los estudiantes, el cual está habitado por personajes no jugables, que tienen implementado máquinas de estados finitos con el fin de lograr una interacción con el jugador. El Serious Game fue evaluado con un grupo de estudiantes, en el cual se vio los resultados de usar el juego, visto que los estudiantes adquirieron nuevos conocimientos en el manejo y desarrollo de videojuegos y no tuvieron problemas al seguir la lógica del juego. En conclusión, la implementación del Serious Game ayudan a los estudiantes a adquirir nuevos conocimientos de forma indirecta y divertida.

Palabras clave: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <INGENIERÍA DE SOFTWARE>,<PROGRAMACIÓN DE VIDEOJUEGOS>, <SERIOUS GAMES>,<METODOLOGÍA SUM>,<TÉCNICAS DE IA>.

SUMMARY

The purpose of this work is to design and implement the Serious Game “Union City” which incorporates artificial intelligence elements within the game´s logic and environment, to teach an3D introductory interactive course on *Unity 3D*, with the aim to train students on the management and development of videogames using the video game engine. The project is for the “Centro de Tutorías”, for the subject of Interfaces and Multimedia of the Escuela de Informática y Electrónica at “ESPOCH”. SUM methodology for videogames, which is designed for small groups, time enhancement, and for development quality, was applied. The methodology is divided in different development stages –from the game concept setting up, the Beta Version production, to the game release. A virtual city, taking into account the most relevant information, was created to improve students´ learning capacity. The virtual city is inhabited by non-playable characters, which have finite state engines with the purpose of achieving interaction with the players. The Serious Game was evaluated with the participation of a group of students, which showed positive results because the students acquired new knowledge on videogame management and development. The students did not have problems to follow the game logic. Thus, it can be concluded that the implementation of educational games, or Serious Games helps students to attain new knowledge in a direct and fun fashion.

Key words: <TECHNOLOGY AND ENGINEERING SCIENCE>, <SOFTWARE ENGINEERING>, <VIDEOGAME PROGRAMMING>, <SERIOUS GAMES>, <SUM METHODOLOGY>, <IA TECHNIQUES>

INTRODUCCIÓN

Hoy en día, el avance tecnológico en el área de aprendizaje se ha incrementado rápidamente dentro de la nueva generación de estudiantes, por tal razón los profesores han tenido que capacitarse en la utilización de nuevas tecnologías. Es por ello, que se eligió el recurso de los videojuegos en el ámbito del aprendizaje para la enseñanza en el uso del motor de videojuegos Unity 3D que tiene la finalidad de crear mundos virtuales interactivos para el usuario.

El mundo de los videojuegos se divide en diversas categorías. Este proyecto se centra en la categoría de los Serious Games que son aquellos juegos que tienen como finalidad divertir y enseñar a las personas diversas propuestas educativas, con la finalidad de que él estudiante se divierta y no le resulte tan cansado al momento de estudiar, por lo que se intenta incentivar al estudiante a repasar en su tiempo libre y mejorar sus habilidades cognitivas.

Por tanto, este documento se encuentra dirigido al desarrollo del Serious Game ¨Unity City¨ mediante la metodología SUM, el cual tiene como objetivo, que los jóvenes aprendan a usar y desarrollar videojuegos mediante el motor Unity 3d, en donde se construye una ciudad virtual en la que existen personajes que enseñan los conceptos básicos para crear su propio mundo virtual y así poder determinar los beneficios y desventajas de los Serious Games.

Y por último, el presente trabajo se encuentra dividido en 3 capítulos: El capítulo I se basa en el marco referencial de los temas y herramientas que se usan en el proyecto. El capítulo II se centra en la creación, diseño y desarrollo del Serious Game ¨Unity City¨. Y el capítulo III se comprueba y se analiza los resultados obtenidos del juego dentro de un entorno orientado al segmento juvenil.

Planteamiento del Problema

Antecedentes

Debido a la revolución tecnológica en el ámbito del aprendizaje en estos últimos años, se ha tenido que implementar nuevas metodologías para los jóvenes, por lo que aparece el término "Serious Games" como un calificativo que denota un componente educativo, el cual nace con el objetivo de impartir conocimientos mediante videojuegos en donde se fortalecen las habilidades latentes de los jugadores a través de la práctica y la resolución de los problemas.

Lo que da lugar a que los jugadores adquieran competencias mediante actividades basadas en el videojuego, debido a su carácter lúdico e interactivo que conlleva a motivar el aprendizaje gracias a la mejora en el desempeño y en las experiencias adquiridas, como por ejemplo el uso de la realidad virtual que puede simular situaciones realistas en donde los estudiantes no corran ningún riesgo durante su enseñanza (Romero y Turpo Gebera 2012a, p. 10).

Hoy en día, existe la demanda de tecnologías con algoritmos basados en la inteligencia artificial como son: Los tutores virtuales o sistemas que se encargan de realizar el seguimiento del estudiante, analizar el grado de atención y el nivel del aprendizaje ganado, gracias a la utilización del sistema informático utilizando técnicas de personalización que se encuentran fundamentados en modelos de sistemas con agentes inteligentes (Urretavizcaya, 2001, p. 2).

Ahora el mundo de los videojuegos ya no solo es para entretener sino también para adquirir habilidades y destrezas necesarias en el mundo real. Los videojuegos deben poseer principios pedagógicos, cognitivos y de aprendizaje, por lo que es un gran reto, por esta razón existen diversas investigaciones y desarrolladores que estudian este tema para mejorar el campo de la enseñanza.

Por lo que se ha visto la necesidad de buscar nuevas maneras de enseñanza en materias de un contenido de grado de dificultad alto. En este contextose ha identificado la materia de Interfaces y Multimedia dictada en la Escuela de Ingeniería en Sistemas de la Facultad de Informática y Electrónica de la ESPOCH.

Esta materia comprende en unos de sus capítulos el diseño de sistemas interactivos multimedia basado en diseño de animaciones 3D, juegos por computadora y simuladores. El diseño e implementación de la aplicación o proyecto final es tarea sistemática y compleja, por lo que se requiere una guía o tutorial para completar el sistema.

Si bien el estudiante tiene acceso a proyectos de semestres anteriores, no existe un tutorial con ambientes 3D y con aspectos de juego dentro de la plataforma Unity3D para que se pueda lograr un buen diseño y desarrollo de la aplicación final.

En el proceso de enseñanza y aprendizaje se lo realiza la mayor parte con técnicas tradicionales, pero en los últimos años se ha implementado en parte un modelo lúdico de aprendizaje utilizando dinámicas y artes manuales para mejor comprensión del estudiante.

En consecuencia, se plantea atacar el problema de la falta de conocimientos mediante la implementación de un Serious Game, que se basa en el viaje por una ciudad en 3D que es para la plataforma de PC, para que los estudiantes obtengan conocimientos previos para el manejo correcto de la plataforma Unity.

Para lo cual se medirá la usabilidad y funcionabilidad que presta el Serious Game a los estudiantes, en el que se medirá si el juego cumple con los objetivos planteados de enseñanza y a la vez el esfuerzo que se necesita para la utilización de Unity 3D.

Algunos autores como Armas Lasso manifiestan que, aunque los videojuegos tengan una mala reputación a consecuencia de que se cree que causan casos de extrema violencia, atrofiamiento de habilidades de la inteligencia, factores de sociabilidad, cambios de conducta y comportamientos negativos sobre los chicos, indican que, esto se debe más bien al tipo de videojuego o al ambiente familiar en el que viven.

Según la investigación realizada por Estallo y Márquez señalan que los videojuegos incentivan a los jóvenes a crear, a imaginar, hacer estrategas, a ganar o perder en un mundo en el cual siempre se está compitiendo para ser eficientes y competitivos en el mundo moderno. (Armas Lasso 2016, p.28)

El estado del arte

Dentro de la creación de los Serious Games existen una gran variedad de proyectos similares que intentan cubrir casi los mismos objetivos del proyecto que se han desarrollado en el ámbito del aprendizaje siguiendo diferentes metodologías.

Por lo que se optado en el proyecto el uso de la metodología SUM que se encuentra fuertemente ligado con el desarrollo de videojuegos, que ha dado tan buenos resultados en proyectos anteriores

y en el uso de la plataforma Unity en el que se integra el Serious Game, que a su vez dará vida a un ambiente de inmersión multimedia al usuario con el fin de enseñar el uso las herramientas básicas para la plataforma Unity.

- El proyecto ¨ **Serious Games orientado al aprendizaje de la física**¨ consiste en el análisis e implementación del Serious Game el cual está orientado al aprendizaje de la física, como herramienta didáctica complementaria en el contexto del proceso de enseñanza aprendizaje de la física a nivel superior en la carrera de ingeniería mecatrónica en UPIITAIPN.

  La finalidad del proyecto es promover el aprendizaje de la física en forma de habilidades y competencias para un estudiante de ingeniería mecatrónica (Pretelín Ricárdez, 2016, p. 28).

- El proyecto ¨ Inteligencia artificial del videojuego la dama ¨ tiene como finalidad el desarrollo e implementación de un Serious Game enfocado en las leyendas de Quito en la cual se implementará agentes de la inteligencia artificial como son Plath Planning y la máquina de estados con el fin de que ciertos elementos tengan un comportamiento más realista dentro de la plataforma Android con lenguaje en C# y usando el motor de juegos Unity 3D.

  Para la implementación de los agentes se creó diferentes tipos de comportamientos personalizados, para cada uno de los NPC's (Non Player Character) que componen el videojuego, algunos de los cuales se los implementa usando Agentes (NavMeshAgent), herramienta que es proporcionada por el Game Engine de Unity (Castro y Patricio, 2016, p. 5).

- El proyecto ¨Llumpak ¨ realizado por el Ingeniero Juan Moreno, es un juego que intenta llegar a las personas de una forma dinámica y a la vez divertida, para hacer que las personas recapaciten sobre el medio ambiente, y como nosotros podemos ayudar a salvarlo, mediante la recolección de basura o como erradicar los incendios, dentro del juego; el desarrollo del juego se basó en la implementación de la metodología SUM y el uso de Unity (Arias y Carlos, 2016, p. 30).

- El proyecto y estudio desarrollado por el Licenciado Carlos Yaguana, se basa en la relación que tiene el jugador con el mundo de los videojuegos, mediante un sistema de realidad virtual con la Kinect, en la que se simula la realidad a través de entornos lúdicos, con lo que concluyo que los participantes perciben el mundo virtual como un ¨cuasi otro¨, en la que se crea una comunicación entre hombre y máquina (Padilla y Daniel, 2014, p. 5).

- El proyecto que realizó el Ingeniero Jonnathan Cóndor, se basa en el desarrollo del demo de un videojuego en el que se representa la mitología ecuatoriana descrita en el ejemplar ¨La historia del Reino de Quito¨ basándose en metodologías ágiles para su desarrollo y el uso de leguaje C# y JavaScript, por lo que obtuvo como resultados los rasgos principales para el desarrollo de un prototipo valido para el videojuego, de un género estratégico (Alvarado y Eduardo 2015, p. 14).

Todos los proyectos que involucran videojuegos que se basan en las áreas de enseñanza, se pueden clasificar como Serious Games, en vista de que intentan cambiar y a la vez enseñar mediante la práctica, rasgos principales dentro de nuestra sociedad, como al intentar que seamos más responsables con el medio ambiente o enseñarnos más acerca de la historia de nuestro país.

Formulación del problema

¿Permitirá la aplicación de un Serious Game mejorar el aprendizaje de los aspectos básicos de Unity?

Sistematización del problema

- ¿Qué son los Serious Games y cuáles son las ventajas y desventajas de su uso?
- ¿La metodología SUM abarca todos los aspectos para el desarrollo de videojuegos?
- ¿Cuál es la importancia de la inteligencia artificial dentro de los Serious Games?
- ¿Cuáles son las técnicas de Inteligencia artificial que contribuyen más al desarrollo de los Serious Games?
- ¿Cuáles son los aspectos que se deben tomar al desarrollo de los Serious Games?
- ¿Qué plataforma contribuye más al desarrollo de los Serious Games?
- ¿Qué nuevas estrategias educativas se están usando para integrar las tecnologías emergentes como la realidad virtual y aumentada?

Justificación

Justificación Teórica

Los Serious Games se han diseñado para mejorar ciertos aspectos de aprendizaje debido a que los usuarios de este tipo de juegos inician su manejo con grandes expectativas las mismas que no siempre son llenadas, por lo que los juegos son usados especialmente para la capacitación de servicios como son los de emergencias, entrenamiento militar, educación corporativa, cuidado de la salud, con el fin de contribuir al desarrollo del conocimiento (Derryberry 2017, p. 7).

También se los puede ver como un método o técnica de enseñanza que puede ser aplicable para la formación en empresas hasta en adultos para el desarrollo de habilidades y aptitudes que están relacionados con tareas profesionales especificas dentro de un ambiente formativo (Sánchez Gómez 2017, p. 10).

El uso de la plataforma Unity en el desarrollo de juegos ha abarcado el mercado de los juegos por ser una herramienta muy potente, fácil de usar por su capacidad de aprendizaje para la creación de juegos básicos y sencillos en 2D y 3D, otra de sus bondades es la utilización de tres lenguajes para los scripts entre los cuales tenemos el Javascript y C#.

Pero uno de los aspectos más importantes de Unity es ser una herramienta multiplataforma, por lo que se lo puede usar en diversos sistemas operativos, pero existen desventajas dado que al manejar al mayor rendimiento éste causa un sobrecalentamiento en el sistema que lo puede volver lento, otro problema es el espacio que ocupa en los proyectos pues puede expandirse en su tamaño de acuerdo a lo que se implemente, pero esto no afectará al producto final (Fenrir, 2016, p. 8).

Al usar la tecnología de Realidad virtual podrá permitir al usuario sumergirse en una simulación gráfica 3D generada por el ordenador y navegar e interactuar en ella en tiempo real, desde una perspectiva centrada en el usuario por lo que un Serious Game tendría un mayor impacto en el estudiante, en donde se pueda sustituir la realidad física por un entorno ficticio generado por el ordenador.

Por lo que se obtiene una experiencia más real del aprendizaje con el fin de que se quede grabado en la mente de la persona como por ejemplo si pensamos que la realidad virtual es lo más parecido que tenemos a la máquina del tiempo, dado que nos permite recrear virtualmente cualquier tipo de espacio en tres dimensiones y situarlo en cualquier época, incluso en el futuro.

A consecuencia de la realidad virtual, ésta posee un grado de realismo completamente creíble que deja una huella dentro de la persona.(Perez Martinez, 2016, p. 3), por lo que habitar dentro de un entorno virtual habitado por los avatares, se buscará apoyar las diversas actividades relacionadas con el aprendizaje, debido a que al interactuar con ellos se podrá implementar un entorno de aprendizaje constructivista favorable con el estudiante (Ménendez Escobar, 2017, p. 5).

Mediante un entorno de aprendizaje constructivista dentro del juego se busca la necesidad de entregar a cada uno de los estudiantes las herramientas necesarias para poder construir sus propios procedimientos y encontrar la solución a los problemas, por lo que se adoptará un proceso participativo por parte de los estudiantes que interactuarán con el entorno virtual a fin de resolver el problema que se plantea (Gamelearn, 2015, p. 10).

Justificación Aplicativa

La manera de enseñar que se ha dado hasta ahora ha provocado en los estudiantes la falta de interés por aprender, pero ahora en estos tiempos la tecnología ha tomado un lugar importante en el mundo moderno, mediante el cual el aprovechamiento de las tecnologías emergentes dentro de los Serious Games en un ambiente educativo, ha demostrado que tiene un mayor efecto en los estudiantes dado que recrea una sensación de inmersión multimedia dentro del mundo virtual.

La implementación de una aplicación con un tipo de enseñanza constructivista ayudará a los estudiantes adquirir y construir sus propias herramientas de aprendizaje. Dado que al enfocar su tiempo en jugar videojuegos podrá aumentar la habilidad cognitiva del estudiante y captar mejor los conocimientos que provee el juego de manera que el estudiante no sienta el estrés de estudiar.

El Serious Games ¨Unity City¨ tiene el propósito de adentrar al estudiante a un mundo virtual usando el ordenador en donde le permita interactuar con una ciudad ficticia, para lo cual deberá completar diversos objetivos en donde se pueda observar la información concerniente para el desarrollo de un videojuego en Unity3D.

OBJETIVOS

Objetivo General

Diseñar e implementar un Serious Games para el diseño de un curso interactivo 3D de Introducción a Unity con apoyo de la Inteligencia Artificial.

Objetivos Específicos

- Determinar el estado del arte a nivel local e internacional de Serious Games.
- Determinar las características de la metodología SUM en el desarrollo de Serious Games.
- Desarrollar el entorno Virtual para el Serious Game con algoritmos inteligentes.
- Evaluar la usabilidad y funcionalidad del Serious Game.
- Documentar las técnicas de Inteligencia Artificial para el Serious Game.

CAPITULO I

1. MARCO TEÓRICO REFERENCIAL

1.1. Definición de un Serious Games

Un Serious Game o también conocido como juego serio puede ser definido como aquel juego que tiene como principal objetivo el aprendizaje de sus jugadores. Aunque algunos autores sostienen que los juegos serios tienen intrínsecamente propósitos, otros por el contrario los relacionan con distintas áreas comunicativas como: la información, la persuasión y la educación (Quintero, 2012, p. 3).

La modalidad de videojuego denominada Serious Game se caracteriza por ser diseñado específicamente con objetivos educativos, de entrenamiento e información. Esto no implica que el juego no tenga que ser divertido, sino que se diseña el entretenimiento del juego para educar de forma que el aprendizaje del estudiante pase a ser divertido.

Los Serious Games para la modelización formal se ha considerado los siguientes componentes estructurales:

- **Objetivos:** Deben estar claramente definidos y ser conocidos en todo momento por el jugador. En este contexto de un Serious Games de ámbito educativo universitario, quedan especificados en los objetivos de aprendizaje de las competencias que se requiera trabajar.
- **Reglas:** Es el componente donde queda determinado el orden de juego, los derechos y responsabilidades de los jugadores y los objetivos que debe perseguir cada jugador para alcanzar el reto que se propone.
- **Reto:** Determina cuándo se da por completado el juego. El jugador se enfrenta a problemas para los cuales tiene que buscar soluciones, de forma que una vez resueltos todos, dará por alcanzado el reto. Para este Serious Games que se propone, los criterios de cierre de los escenarios parciales y del final (juego terminado), se concretan en resultados de aprendizaje.
- **Interacción:** Es el componente que surge de la propia mecánica y dinámica del juego y que da lugar a todas las experiencias que tenga el jugador (Romero y Turpo Gebera 2012, p. 7).

1.1.1. Ventajas de los Serious Games

Las principales ventajas de los Serious Games, se centran en su finalidad, que son:

- Permiten estimular la mente de los jugadores al impulsarlos en la toma de decisiones y la mejora de las funciones cognitivas de sus usuarios.

- El aprendizaje obtenido durante el juego es aplicable en el mundo real, en vista de que permite adquirir conceptos y capacidades necesarias como el manejo de aviones gracias a los simuladores que existen.
- Los Serious Games permiten que sus jugadores desarrollen habilidades durante el juego de forma continua, al momento de repetir el juego, gracias al enfoque lúdico que poseen.
- Existe un entorno de colaboración entre el jugador y el juego con el fin de cumplir objetivos que favorezcan a las dos partes, a consecuencia de que el juego busca compartir sus conocimientos mientras que el jugador busca adquirir los mismos.

1.1.2. Desventajas de los Serious Games

Las desventajas más comunes dentro del Serious Games se presenta al momento de su desarrollo, debido a que al crearlo se tiene que invertir mucho tiempo y dinero, además estos tipos de juegos no tienen mucha demanda en el mercado.

1.2. Inteligencia artificial en los Videojuegos

La inteligencia artificial se puede ver como una simulación del comportamiento de los personajes no manejados por el jugador es decir de los NPCs (Non Person Controller) que son Personajes no jugables como son los enemigos, jefes finales, animales (Alcalá, 2016, p. 6).

Se los puede ver como agentes electrónicos que pueden pensar, evaluar y actuar en ciertos principios de la optimización y la coherencia para cumplir con una meta o propósito dentro del mundo del juego (Alcalá, 2016, p. 23).

La aplicación de la Inteligencia Artificial (IA) en los videojuegos va más allá de la inteligencia que posean los NPCs, que es la adaptación del juego en función de los gustos personales de cada jugador, algo que se está potenciando hoy en día y que supone una de las experiencias que gracias a esta técnica son más gratificantes.

Un claro ejemplo de ello es el conocido videojuego World of Warcraft, en el cual se van recopilando datos del usuario a lo largo del juego y mediante minería de datos se extrae información sobre los gustos del mismo, para posteriormente potenciar el juego (Martínez, 2016, p. 11).

1.2.1. Definición de Inteligencia Artificial en videojuegos

Es la simulación de comportamientos de los personajes no manejados por el jugador: NPCs, enemigos, jefes finales, animales, etc... Es un agente electrónico que puede pensar, evaluar y

actuar en ciertos principios de la optimización y la coherencia para cumplir con una meta o propósito. (Alcalá, 2011, p.2)

1.2.2. Historia de la inteligencia artificial en los videojuegos

Los primeros sistemas de inteligencia artificial se originaron en los años 50, se aplicaron a juegos de mesa: damas (Arthur Samuel) y ajedrez (Claude Shannon). No es sino, en los años 60 se desarrollaron juegos como el Pong o Spacewar, basados en la lógica.

Años después por los años 70 surgieron juegos de un jugador contra enemigos que se movían mediante patrones almacenados. Space Invaders en 1978 añadió dificultad creciente y respondía a las acciones del jugador. Pac-Man en 1980 incorporó algoritmos de búsqueda en laberintos. Dragon Warrior en 1990 fue el primer RPG, permitía variar las rutinas de la inteligencia artificial de los enemigos durante las batallas.

En los años 90 se produjo un boom de nuevos géneros y nuevas técnicas de inteligencia artificial

- Máquinas de estados finitos
- Redes de neuronas
- Computación evolutiva
- Lógica difusa

1.2.3. Técnicas de inteligencia artificial

1.2.3.1. Path Planning (Búsqueda de caminos)

La búsqueda de caminos tiene como objetivo encontrar una secuencia de acciones que permitan al jugador moverse desde el punto A hasta otro punto B del mapa.
La información que necesita este tipo de inteligencia artificial es:

a) Representación del mapa

Este elemento proporciona toda la información del juego, de cómo está estructurado y una representación que puede ser tridimensional visualmente, pero realmente funciona como una imagen bidimensional y está constituida por una matríz.

La información que se proporcione en este punto es muy importante, por ejemplo: Si se restringiera la iteración del jugador con el entorno; es decir, definir en un punto en el mapa en el cual la inteligencia artificial no pudiera interaccionar, como fuera el caso de una roca gigante, entonces la inteligencia artificial debería esquivarla. Para ver un ejemplo de las diferentes representaciones de mapas (Castro y Patricio, 2016, p. 8).

b) Movilidad de las unidades o el jugador

Se toma en cuenta la movilidad que deben tener las unidades o el jugador, si se tratara de un juego en primera persona como un juego de disparos, normalmente se controla a un único jugador.

Pero si se tratara de un juego de estrategia, en el cual hay un conjunto de unidades, las cuales todas y cada una de éstas es un jugador independiente, tiene que calcular su camino, para llegar a un punto determinado (Castro y Patricio, 2016, p. 8).

c) Coste de las acciones de las unidades o del jugador

Cada una de las acciones de la inteligencia artificial tiene un coste (valor), por ejemplo: Se define que si la inteligencia artificial se mueve en diagonal tiene un valor de 1, si salta tiene el valor de 50, estos valores son los que influyen en que un algoritmo encuentre o no un camino más rápido o más lentamente (Castro y Patricio, 2016, p. 8).

El uso de esta técnica se puede ver en el videojuego de Unity City para los personajes no jugables, en el cual se crea un Mapa de Navegación del terreno del juego en donde se identifican todos los objetos estáticos como también las elevaciones del terreno, con la finalidad de que los personajes no jugables puedan calcular el mejor camino para llegar desde un punto a otro, teniendo en cuenta variables de costo en sus movimientos, que son calculados por Unity 3D.

1.2.3.2. Máquina de Estados Finitos

Una Máquina de Estado Finito o también llamada Autómata Finito es una abstracción computacional que describe el comportamiento de un sistema reactivo mediante un número determinado de Estados y un número determinado de Transiciones entre dicho Estados (Castro y Patricio, 2016, p. 9).

Las Transiciones de un estado a otro se generan en respuesta a eventos de entrada externos e internos; a su vez estas transiciones y/o subsecuentes estados pueden generar otros eventos de salida.

Esta dependencia de las acciones o respuestas del sistema a los eventos de entrada hace que las Máquinas de Estado Finito (MEF) sean una herramienta adecuada para el diseño de Sistemas Reactivos y la Programación Conducida por Eventos (Event Driven Programming), cual es el caso de la mayoría de los sistemas embebidos basados en microcontroladores o microprocesadores.(Castro y Patricio, 2016, p. 10)

Las MEF se describen gráficamente mediante los llamados Diagramas de Estado Finito (DEF), llamados también Diagramas de Transición de Estados (Torrico, 2016, p. 3).

a) Sistemas Reactivos

El sistema reactivo es aquel que interactúa con el medio ambiente, con el fin de ser conducido por eventos para la obtención de respuestas para permanecer en un estado de forma indefinida, estos sistemas deben tener la capacidad de responder de forma inmediata, que no debe durar más de milisegundos o microsegundos como dentro de un sistema de soporte vital dentro de un hospital (Torrico, 2016, p. 3).

b) Sistemas Transformacionales

A diferencia de los Sistemas Reactivos un Sistema Transformacional es aquel que recibe cierta información de entrada, realiza una cierta cantidad de cómputo, produce cierta información de salida y luego termina. No muchos sistemas embebidos caen en esta categoría; ejemplos más típicos son las aplicaciones para PC, como por ejemplo: Un procesador de texto (Castro y Patricio 2016, p. 10).

c) Diagrama de Estado Finito o Diagrama de Transición de Estados

El Diagrama de Estado Finito es la gráfica que se encarga de representar los estados y las transiciones entre cada uno de ellos de forma simplificada de una máquina de estados finitos, como por ejemplo: El sistema de control de un ascensor que posee tres estados como se muestra en la Figura 1-1:

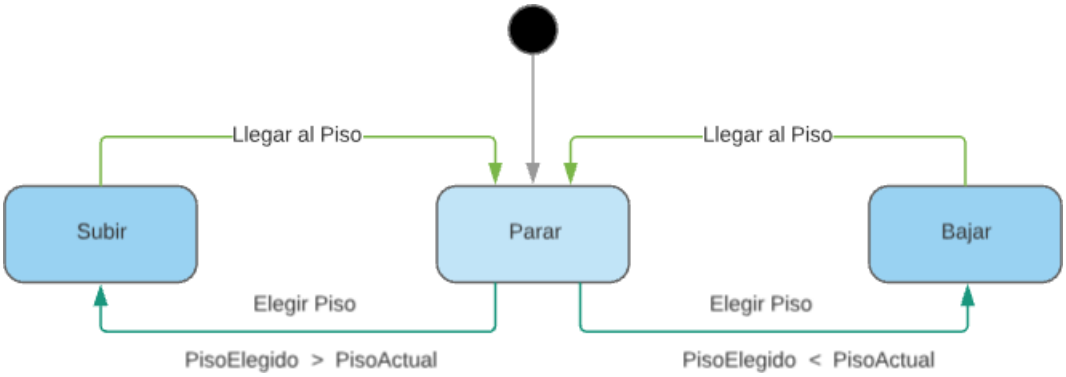


Figura 1-1: Diagrama de Estados Finitos

Realizado por: Christian Viracocha, Año 2018.

Transiciones: Las transiciones se las representa mediante flechas que indican la dirección de transición de un estado a otro.

Eventos: Los eventos son los siguientes:

- Elegir piso: Es un evento externo que se genera cuando el usuario llama al ascensor y elige un piso.

- Llegar al piso: Es un evento interno que se genera cada vez que se llega al piso elegido por el usuario.

Los eventos se escriben por encima de las flechas de transición.

Condiciones de Transición: Las transiciones que se muestran en este ejemplo tienen asociadas sus respectivas Condiciones de Transición, se debe decir que no todas las transiciones poseen condiciones de Transición.

PisoElegido > PisoActual: Es la condición necesaria para que el ascensor pase del estado de Detenido a Subir al piso indicado.

PisoElegido < PisoActual: Es la condición necesaria para que el ascensor pase del estado de Detenido a Bajar al piso indicado.

Las Condiciones de Transición se escriben por debajo de las flechas de transición (Torrico, 2016, p. 3).

El uso de esta técnica se puede ver el videojuego de Unity City para los personajes no jugables que interactúan con el jugador directamente, en donde los estados cambian de acuerdo con la distancia que hay entre ellos, con el fin de crear una relación entre el jugador y el juego.

Esto se presenta cuando el personaje no jugable a localizado al jugador, este cambia de un estado de reposo a un estado de caminar y cuando está frente a él, este cambia automáticamente a un estado de interacción, en donde el personaje no jugable cambia de animación y a la vez muestra la información respectiva que se encuentra almacenado internamente y así dar un poco más de vida y toma de decisiones a aquellos personajes no jugables dentro del juego.

1.2.3.3. Árbol De Decisiones

Los Árboles de Decisiones son rápidos, fácil de implementarlos y sencillos de entender.

Aunque sean el tipo de técnicas de toma de decisión más sencillas, pueden resultar bastante sofisticados con algunas extensiones/variaciones.

Se puede visualizar en la Figura 2-1, está formado por un punto de decisión, que se denomina la raíz. Cuando se llega a un punto de decisión, la bifurcación que se elige depende del estado de entrada, esto es, de la información de la que se dispone en el momento en el que se ha iniciado el proceso de toma de decisiones (Castro y Patricio, 2016, p. 12).

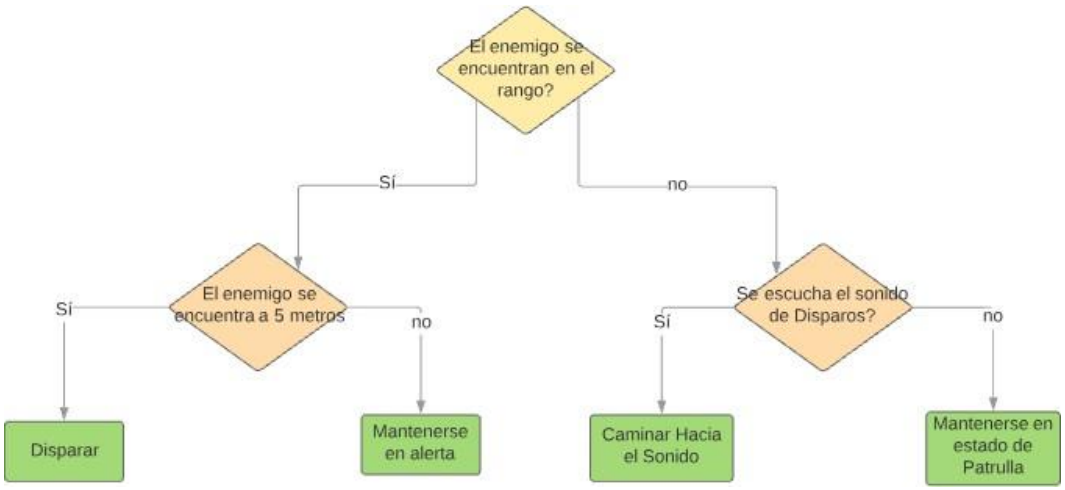


Figura 2-1: Diagrama del Árbol de Decisiones

Realizado por: Christian Viracocha, Año 2018.

1.2.3.4. Redes Neuronales Artificiales (RNA)

Las RNA (Redes Neuronales Artificiales) son un sistema que sigue el modelo de las redes neuronales biológicas, está conformado por las entradas a la red, las salidas y los nodos o neuronas junto con las interconexiones entre ellas como se muestra en la Figura 3-1 (Mallo y Ruiz, 2016, p. 5).

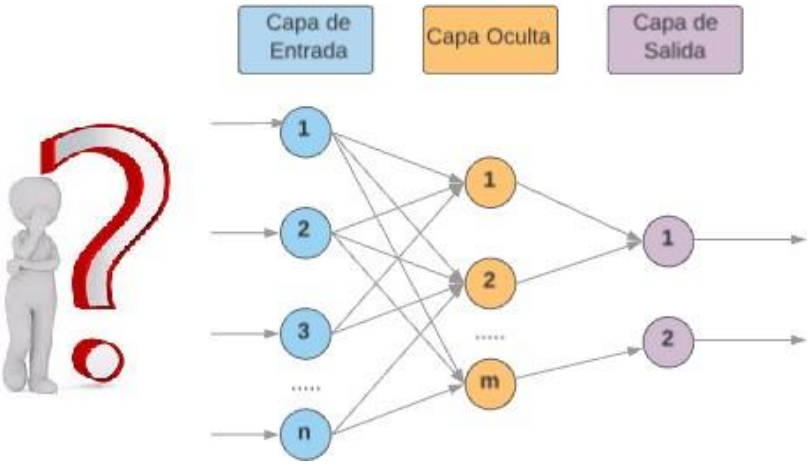


Figura 3-1: Diagrama Red Neuronal Artificial

Realizado por: Christian Viracocha, Año 2018.

La principal ventaja de este tipo de redes puede conseguir un sistema que proporcione salidas adecuadas en función de unas entradas sin saber exactamente la función total de transferencia, sino simplemente entrenándola bien, generalizando el reconocimiento en patrones. Como punto

negativo se requiere un entrenamiento supervisado exhaustivo, cuanto más profundo sea el entrenamiento mejor reaccionará la red (Mallo y Ruiz, 2016, p. 5).

Los ejemplos típicos de RNA´s son el perceptrón simple y la multicapa, aunque en la mayoría de los casos actuales la multicapa es usado. Esta capacidad de devolver el resultado más correcto ante una entrada desconocida convierte a las RNA´s en un modelo muy productivo para la inteligencia artificial en los videojuegos.

Una de las aplicaciones más comunes encontramos en las técnicas de reconocimiento de obstáculos y objetivos, así como en el movimiento autónomo (Mallo y Ruiz, 2016, p. 5).

El RNA se usa como un ejemplo de perceptrón multicapa para un sistema de avance con obstáculos, es decir recibe mediciones de distancia por la izquierda, derecha y centro hasta llegar a un obstáculo; el mismo que dirigirá si hay que seguir de frente, girar a la derecha o a la izquierda (Mallo y Ruiz, 2016, p. 5).

En el caso de que la población para entrenamiento no sea lo suficientemente grande se pueden aplicar técnicas más avanzadas, que permiten adaptarse más fácilmente durante el juego a los cambios del entorno (Mallo y Ruiz, 2016, p. 5).

1.2.3.5. Goal Oriented Action Planning (GOAP)

Este método consiste en la planificación de acciones orientadas a las metas, como los seres humanos, visto que tenemos objetivos al realizar ciertas acciones. Cuando se piensa en inteligencia artificial puede crear ciertas acciones que le permitan llegar a los objetivos, por lo que el objeto debe elegir las mejores acciones para cumplir la meta y pueda ser mucho más interactivo, como se muestra en la Figura 4-1.

El GOAP se basa en que cada NPC del juego y se les otorga de:

- Un conjunto de metas que son alcanzables.
- Un conjunto de condiciones asociadas a cada una de las metas.
- Un conjunto de acciones formadas por dos elementos:
  - Precondiciones
  - Efectos

Figura 4-1: Ejemplo de Precondiciones y Efectos

Realizado por: Christian Viracocha, Año 2018.

El uso de esta técnica se ve en el videojuego de Unity City, en donde es usado para el Wizard para mandar mensajes de acuerdo con como haya llenado los test que se encuentran alrededor del juego. También se encuentra presente en el sistema de finalización del juego, ya que la precondición para terminar es haber cumplido con todos los objetivos, con el fin de mostrar un video en donde se agradece por haber jugado.

1.3. Realidad Virtual

La Realidad virtual comprende la interface hombre-máquina (human-machine), que permite al usuario sumergirse en una simulación gráfica 3D generada por ordenador y navegar e interactuar en ella en tiempo real, desde una perspectiva centrada en el usuario.
Es una experiencia sintética mediante la cual se pretende que el usuario sustituya la realidad física por un entorno ficticio generado por ordenador. "La Realidad Virtual es lo más parecido que tenemos a la Máquina del Tiempo, en tanto que nos permite recrear virtualmente cualquier tipo de espacio en tres dimensiones y situarlo en cualquier época, incluso en el futuro, con un grado de realismo completamente creíble" (Martínez 2011, p. 10).

1.3.1. Sistemas de Realidad Virtual

Sistemas Desktop de realidad virtual
Muestra una imagen 2D ó 3D en un monitor, "casco" o pantalla de proyección. Son la mayoría de los videojuegos para PC's o consolas de los hogares. El usuario ve la imagen en primera persona (Martínez 2011, p. 14).

RV en segunda persona
Es un integrante "visible" del mundo virtual porque ve la proyección de su imagen en un fondo o ambiente (Martínez 2011, p. 14).

Sistema de Tele presencia
Son los elementos que están ligados bajo un control remoto que permite manipular los dispositivos robóticos como por ejemplo en la medicina al momento de realizar una cirugía y los doctores se encuentran a cada extremo del mundo (Martínez 2011, p. 14).

Sistema de inmersión de realidad virtual
Es la tecnología que permite al usuario adentrarse dentro del mundo virtual al engañar a sus sentidos utilizando sistemas de tipo CAVE (Martínez 2011, p. 14).

1.4. Entorno de aprendizaje constructivista

El enfoque constructivista para el aprendizaje se establece, que el conocimiento es elaborado individualmente y socialmente por los estudiantes con el fin de ganar experiencias y representaciones del mundo y sobre la base de los conocimientos declarativos ya conocidos (Esteban, 2000, p. 2) .

Al hablar del desarrollo de un entorno de aprendizaje constructivista también se hablará del modelo para el diseño de entornos que comprometan a los estudiantes a la elaboración del conocimiento dado que consiste en una solución que parte de un problema, pregunta o proyecto como núcleo del entorno para ofrecer al estudiante varios sistemas de interpretación y de apoyo intelectual derivado de su alrededor.

Por lo que se le otorga al estudiante las herramientas necesarias para resolver problemas o hallar la respuesta a diversas preguntas.

Los elementos constitutivos del modelo son a) las fuentes de información y analogías complementarias relacionadas; b) las herramientas cognitivas; c) las herramientas de conversación/colaboración; y d) los sistemas de apoyo social/contextual (Esteban, 2000, p. 5).

1.5. Tutorial

Un tutorial es un material educativo que permite impartir conocimientos a cualquier persona visto que se percibe como un recurso de mucha ayuda, que puede ser visto como un curso corto y que se centra en aspectos principales de los temas en lo que se basa.

Por lo general sirven como ayuda a los estudiantes que tiene mucha dificultad en temas específicos, según Medway (MEDWAY, F. J. 1485, p. 5315), ubica 3 modalidades que posee la tutoría en la enseñanza superior:

- La primera modalidad se centra en el apoyo a los cursos que se imparten como por ejemplo la explicación del contenido de parte de un profesor a sus estudiantes cuando poseen dudas del mismo.
- La segunda modalidad se centra en preparar a los estudiantes en temas específicos para rendir un exámen.
- La última modalidad se basa en el uso de material previamente estructurado y computarizado para los alumnos (Ordóñez et al. 2005, p. 2).

Al impartir tutorías se debe crear una relación pedagógica ante un gran grupo de estudiantes, por lo que un profesor se convierte en un consejero o un hermano mayor en la cual su autoridad se suaviza para crear un ambiente relajado y tranquilo. Y de esta manera se pueda impartir las clases,

que deben ser dadas en espacios en los cuales el estudiante se encuentre a gusto, como en un lugar físico o virtual.(Ordóñez et al: 2005, p. 5)

Se debe tener en cuenta varios requisitos para que las tutorías funcionen correctamente entre las cuales se identifican:

- El creador de las tutorías debe tener las suficientes habilidades y capacidades del tema que se desea impartir, con la finalidad de que inspire confianza y seguridad para que sus estudiantes puedan eliminar todas sus dudas.
- El tutor debe estar capacitado en varias técnicas de enseñanza y en manejo de grupos.
- Las tutorías no deben ser de larga duración.(Ordóñez et al:. 2005, p. 5)

1.5.1. Orientación para organizar un sistema de tutorial

Dentro de las Instituciones de Educación Superior se deben seguir varias orientaciones que generalizan las tutorías educativas, para que puedan mejorar su comprensión entre los estudiantes de nivel superior:

- Los seminarios de tutorías se dividen en 3 modalidades que son:
    - Los seminarios de apoyo académico, que se centran en el repaso de los cursos regulares.
    - Los seminarios de desarrollo de habilidades básicas, las cuales se ve orientado a la mejora en las habilidades de lectura, hábitos de estudio, redacción entre otros.
    - Los seminarios de desarrollo de la personalidad, que se ubica en la orientación humana y profesional de los alumnos.
- Los profesores que imparten las tutorías se deben dedicar un 50% de su tiempo libre, pero si existen más profesores que dominen el tema, el tiempo de dedicación se reducirá en un 33% de su tiempo, por lo que se recomienda por lo menos 2 horas por semana.
- Cada clase no deberá pasar de los 12 estudiantes con la finalidad de captar las deficiencias y ayudar a todos los estudiantes.
- Al momento de empezar el sistema de tutorías en forma experimental, se recomienda que se utilicen profesores que se ofrezcan de voluntarios para impartir las clases.
- Los directivos de la escuela debe realizar un seguimiento del sistema de tutorías hacia los profesores y estudiantes, con la finalidad de mejorar la calidad de las clases.(Ordóñez et al: 2005, p. 6)

1.6. Unity 3D

Unity 3D es un motor de videojuegos desarrollado por la empresa de Unity Technologies, que se encuentra disponible para los sistemas operativos de Windows, Linux y MacOS, que maneja diversas herramientas, diseñadas para el desarrollo de videojuegos en diversas plataformas mediante lo siguiente:

- El Editor visual de Unity posee herramientas intuitivas que desarrollan juegos inmersivos para mejorar la capacidad en el manejo del programa.
- El ambiente de desarrollo MonoDevelop sirve para la creación de nuevos Scripts el cual combina con éxito las habilidades de un editor de texto con las herramientas y ayudas de Unity mediante el uso de los Lenguajes C# y UnityScript el cual se basa en el lenguaje JavaScript.
- También posee herramientas para el diseño de juegos en 2D, 3D, el manejo de los Storytelling, de las cinemáticas, de la iluminación, del sistemas de audio, para la gestión de Sprites, para los efectos de partículas y el sistema de animación (Unity Technologies: 2018, p. 2).

1.6.1. Ventajas de Unity 3D

Las ventajas de usar Unity 3D son:

- Unity es una herramienta fácil de usar y muy intuitiva dado que su interfaz es simple y muy versátil para el uso de los usuarios, además posee manuales y tutoriales que abarcan casi todo el manejo del sistema, aunque también por su gran extensión y herramientas se requieren años de experiencia para usarlo a su máxima capacidad.
- El uso en el desarrollo de scripts con lenguajes nativos como son el C# y JavaScript, proporciona una gran ventaja para los desarrolladores a consecuencia de que mucha gente conoce y lo ha usado alguna vez en sus proyectos.
- Unity tiene una gran variedad de herramientas para el desarrollo de videojuegos en el ámbito de la animación, las cinemáticas entre otras, que ayudan a elevar el potencial de la calidad de los juegos.
- Posee una tienda llamada Asset Store, que contiene múltiples y muy variados assets como pueden ser objetos 3d que ayudan a la ambientación del juego, como también herramientas y múltiples scripts que ayudan a mejorar el rendimiento y la lógica del juego.
- Cuenta con la habilidad de crear juegos para múltiples plataformas, por lo que es conocida como un herramienta de multiplataformas, desde juegos para celular con diversos sistemas operativos, juegos para Pc, juegos para diversas consolas que pueden

manejar dispositivos como son la Kinetec o el Oculus Rift que dan una nueva perspectiva de los juegos hacia las personas (Fenrir, 2016, p. 2).

1.6.2. Desventajas de Unity 3D

Las desventajas que se encuentran al momento de manejar Unity 3D son varias debido a que la herramienta es muy extensa y por su complejidad aumenta en proporción al desarrollo del juego. Los más relevantes problemas se refieren al estudio "Cinco cosas buenas y cinco no tan buenas de Unity 3D" (Fenrir, 2016, p. 3).

- El espacio de memoria que ocupan los juegos, visto que cada proyecto posee un gran número de archivos y estos aumentan exponencialmente referente a la cantidad de elementos que existen en el mundo virtual, como también el aumento de herramientas y assets dentro del mismo.
- Otro problema es el rendimiento al desarrollar los diversos juegos, a consecuencia de que el sistema está ejecutando en tiempo real diversas tareas de renderización del mundo virtual, en cada cambio que se realice. Por lo que el CPU y la RAM se disparan, por lo que es recomendable tener una computadora con una RAM de 16 Gigas y una buena tarjeta de video para evitar el sobrecalentamiento de sus máquinas.
- Uno de los problemas más comunes en los desarrolladores antiguos es el gestionamiento de las versiones, debido a que muchos objetos y herramientas fueron actualizados y otros fueron dejados como obsoletos, por lo que los juegos que fueron realizados en una versión más antigua van a tener problemas al ejecutarse en una versión actual.
- Otra desventaja al desarrollar los juegos es la alimentación de los recursos de la versión gratuita pero aún así es una herramienta muy poderosa, aunque también existe una versión de pago la cual es hecha específicamente para los profesionales la misma que tienen diversos beneficios como son el uso de herramientas adicionales y descuentos en la tienda de Asset.

1.7. Metodología SUM para Videojuegos

La metodología SUM tiene como objetivo desarrollar videojuegos de calidad en tiempo y costo, así como la mejora continua del proceso para incrementar su eficacia y eficiencia. Pretende obtener resultados predecibles, administrar eficientemente los recursos y riesgos del proyecto y lograr una alta productividad del equipo de desarrollo.

SUM es concebida para que se adapte a equipos multidisciplinarios pequeños (de tres a siete integrantes que trabajan en un mismo lugar físico o estén distribuidos) y para proyectos cortos (menores a un año de duración) con alto grado de participación del cliente (Gemserk, 2016, p. 2).

Esta metodología se adapta para videojuegos de estructura y roles de Scrum que se define en cuatro: equipo de desarrollo, productor interno, cliente y verificador beta. El productor interno y el cliente se corresponden en forma directa con los roles de Scrum Master y Product Owner de Scrum respectivamente (Gemserk, 2016, p. 2).

1.7.1. Fases de SUM

El proceso de desarrollo se divide en cinco fases iterativas e incrementales que se ejecutan en forma secuencial con excepción de la fase de gestión de riesgos que se realiza durante todo el proyecto. Las cinco fases secuenciales son: concepto, planificación, elaboración, beta y cierre, como se muestra en la Figura 5-1 (Gemserk 2016, p. 3).

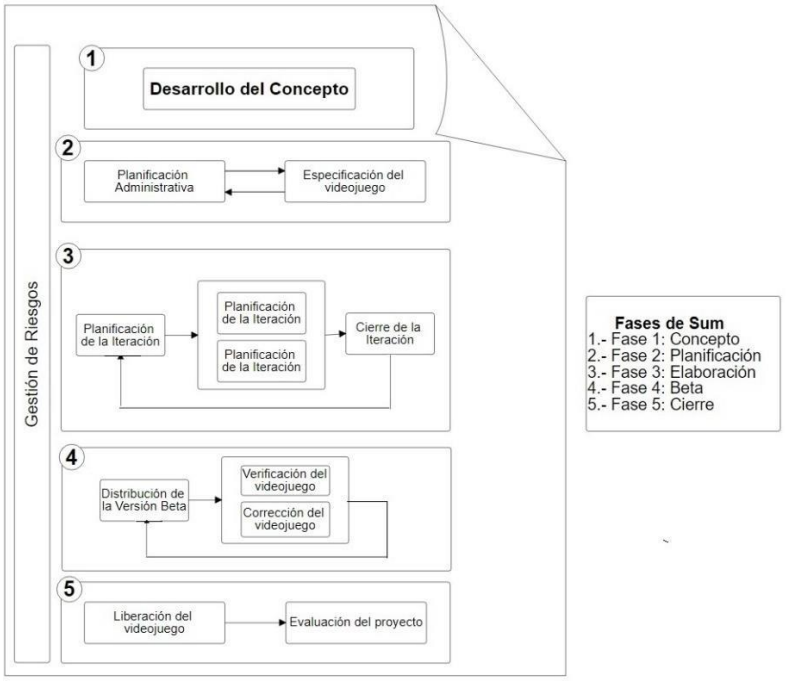


Figura 5-1: Procesos de la metodología SUM

Realizado por: Christian Viracocha, Año 2018

Fase 1: Concepto

En la fase de concepto se define los aspectos técnicos como son: La elección de las herramientas, la tecnología que se usa durante el juego, a qué plataforma está destinada, el aspecto del negocio en donde se ve el objetivo del juego, a qué grupo de personas va dirigido y cuál es el modelo de negocio así como también los elementos, sus personajes y ambientación del juego (Gemserk 2016, p. 5).

Fase 2: Planificación

La fase de planificación tiene dos objetivos principales: 1.- Planificar el resto de las fases del proyecto y 2.- Especificar las características a implementarse en el videojuego. Para ello se realizan dos actividades cuyos resultados componen el plan del proyecto.
Estas se ejecutan en paralelo a consecuencia de que las salidas que generan dependen entre sí, por ejemplo el cronograma debe ser coherente con el tiempo estimado y para realizar las características del videojuego (Gemserk 2016, p. 5).

Fase 3: Elaboración

El objetivo de esta fase es implementar el videojuego. Para conseguirlo se trabaja en forma iterativa e incremental cuyo fin es lograr una versión ejecutable del videojuego al finalizar cada iteración (Gemserk, 2016, p. 5).

Fase 4: Beta

La fase Beta tiene como objetivo evaluar y ajustar distintos aspectos del videojuego como por ejemplo Gameplay, diversión, curva de aprendizaje y curva de dificultad y además elimina la mayor cantidad de errores detectados. Se trabaja en forma iterativa liberando distintas versiones del videojuego para verificar.
En cada ciclo primero se planifica y distribuye la versión beta para ser verificada. Mientras ésta se verifica, se envían reportes con los errores o evaluaciones realizadas. Estos reportes son analizados para ver la necesidad de realizar ajustes al videojuego (Gemserk, 2016, p. 6).

Fase 5: Cierre

Los objetivos del Cierre es poner a disposición del cliente la versión final del videojuego y evaluar el desarrollo del proyecto. Se compone de dos actividades que se ejecutan en forma secuencial, liberación del videojuego y evaluación del proyecto (Gemserk 2016, p. 6).

Gestión de Riesgos

Esta fase se realiza durante todo el proyecto con el objetivo de minimizar la ocurrencia e impacto de los problemas. Esto se debe a que distintos riesgos pueden ocurrir en cualquiera de las fases por lo cual siempre debe existir un seguimiento de los mismos (Gemserk, 2016, p. 6).

CAPITULO II

2. MARCO METODOLÓGICO

2.1. Actividades de la Metodología SUM

2.1.1. Concepto

El Serious Game se desarrolla para la plataforma Pc, el cual será jugable mediante el teclado y el mouse, que tendrá como objetivo la enseñanza y el manejo del programa para la creación de videojuegos en Unity 3D, destinado para los jóvenes que desean aprender a manejar el Sistema.

El juego transporta al jugador a un mundo virtual dentro de una ciudad ficticia en la cual existen 2 tipos de NPC´s; el primero, es el NPC que habita y recorre la ciudad independientemente del jugador, mientras que el otro tiene como objetivo visualizar los aspectos básicos del manejo del programa UNITY 3D y pequeños test para medir el aprendizaje del jugador.

Por lo que su jugabilidad en el que se centra, es el desarrollo y cumplimiento de los objetivos de cada área en donde se encuentra el jugador, el cual va a aprender el manejo del sistema según cómo vaya jugando y explorando el mundo virtual. El participante podrá ver su progreso mediante su Score y la Barra de progreso del juego que se encuentra en la parte superior del panel del visor.

Para el desarrollo del Serious Game se usa el motor de videojuegos Unity 3D, ya que es una herramienta fácil de usar y muy intuitiva, como también de una laptop con la potencia necesaria para el procesamiento de información concerniente al juego.

2.1.2. Planificación

La planificación que se implementa para el desarrollo del proyecto se basa en diez iteraciones con una duración de veinte días por iteración, debido al tiempo de investigación y desarrollo del videojuego, en donde se crean los modelos en 3D y se comprueba el funcionamiento de la inteligencia artificial dentro de los personajes no jugables.

Con la finalidad de mejorar el aspecto jugable e investigativo en la implementación de los temas que se tratan para el aprendizaje del estudiante así como también el uso de elementos de inteligencia artificial dentro del sistema.

2.1.3. Análisis

2.1.3.1. Definición del Juego

El Serious Game ¨Unity City¨ es un juego enfocado a la enseñanza en un ambiente virtual en primera persona, la cual crea un ambiente inmersivo para el jugador, mediante el desplazamiento y la interacción con la ciudad virtual y sus habitantes mediante la PC.

La jugabilidad se centra en el aprendizaje del jugador, en donde se dispone de videos tutoriales, como también de la teoría, las herramientas relacionadas con el proceso y el procedimiento de cómo usar el programa.

También se puede realizar pequeños test, el cual tiene como objetivo medir los conocimientos adquiridos en la utilización del programa Unity 3D para el desarrollo de videojuegos.

El desarrollo del juego se basa en un solo jugador y se reiniciará cada vez que inicie una partida, esto con el fin de que tenga que realizar nuevamente las misiones. Tomando en cuenta que, cuando juegue por 2da vez, el jugador tendrá la experiencia necesaria, debido a que conoce la ciudad a la perfección lo que le permite acceder al material educativo con facilidad.

2.1.3.2. Ambiente del juego

El Serious Games se ubica en una ciudad ficticia conformado por casas, edificios basados en la arquitectura de Riobamba y un Parque Central, el cual está poblado de personajes no jugables, conocidos en la jerga de juegos como los NPCs, los cuales están divididos en dos tipos:

- El primero, son aquellos que se mueven a través de la ciudad de manera independiente con la finalidad de que la ciudad tenga vida.
- El segundo, son aquellos que contienen el material educativo que se divide en: Teoría, Procedimientos, Menús, Videos y un test el cual mide el grado de aprendizaje de que obtuvo el jugador sobre el uso el programa Unity 3D.

La ciudad se divide en 6 sectores los cuales son:

- Área de aprendizaje para la creación de un Terreno.
- Área de aprendizaje para la implementación del Skybox.
- Área de aprendizaje para la implementación de los Objetos de Luz.
- Área de aprendizaje para la implementación del Personaje Jugable.
- Área de aprendizaje para la creación de Scripts.
- Área de aprendizaje de la Estructura de Unity.

Cada uno de estos contiene 5 Personajes no Jugables los cuales impartirán el curso de Unity 3D.

2.1.3.3. Vista principal del juego

La vista principal que posee el jugador será en primera persona el cual cuenta con un visor que contiene los siguientes elementos:

- En la parte superior izquierda de la pantalla se encuentra el puntaje y debajo de ella se encuentra las calificaciones de los test que hay en el juego,
- En la parte superior derecha de la pantalla se encuentra la barra de progreso del juego,
- En la parte inferior izquierda se encuentra el mini mapa del juego
- Y en la parte inferior derecha se encuentra el panel de los objetivos.

En la Figura 1-2 se muestra la imagen del visor que posee el jugador al momento de movilizarse por la ciudad, en donde cada uno de los elementos del juego van actualizándose mientras juega.

Figura 1-2: Pantalla del Visor del Juego

Realizado por: Christian Viracocha, Año 2018

2.1.3.4. Actividad principal del jugador

La actividad principal del jugador dentro del juego es el aprendizaje mediante el desplazamiento por la ciudad virtual y la interacción con los NPC´s de cada sector. Los cuales contienen diferentes temas del uso de Unity 3D que tiene como finalidad obtener el mayor conocimiento posible durante el juego.

2.1.3.5. Paper prototiping

El prototipo del videojuego se basa en la construcción de una ciudad en 3D donde encontramos NPC´s que manejan información acerca del uso de Unity, como se muestra en la Figura 2-2, en la que se ve el prototipo de la ciudad y los NPC´s donde el jugador sigue una ruta por toda la ciudad encontrándose con las misiones del juego.

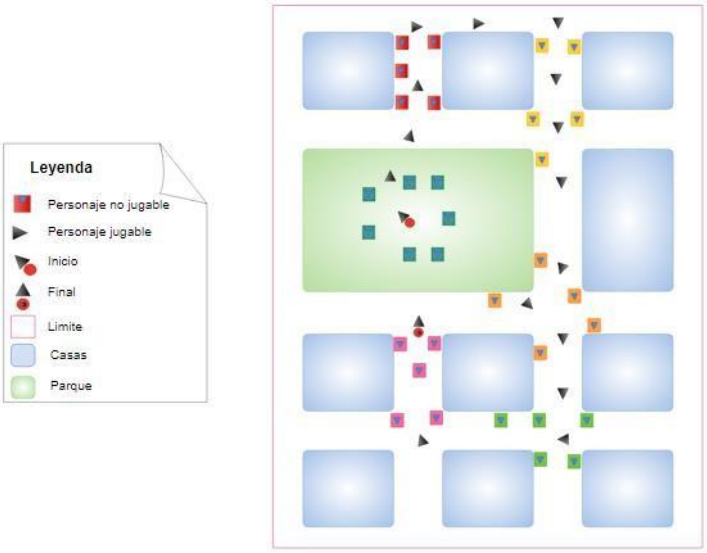


Figura 2-2: Prototiping de la ciudad virtual

Realizado por: Christian Viracocha, Año 2018

2.1.3.6. Assets

Los Assets son paquetes de archivos que proporciona Unity para el desarrollo del juego, que va desde objetos en 3D hasta scripts de desarrollo los cuales ayudan a mejorar el rendimiento y la lógica del juego. Los assets se consiguen del Asset Store de Unity como se muestra en la Figura 3-2.

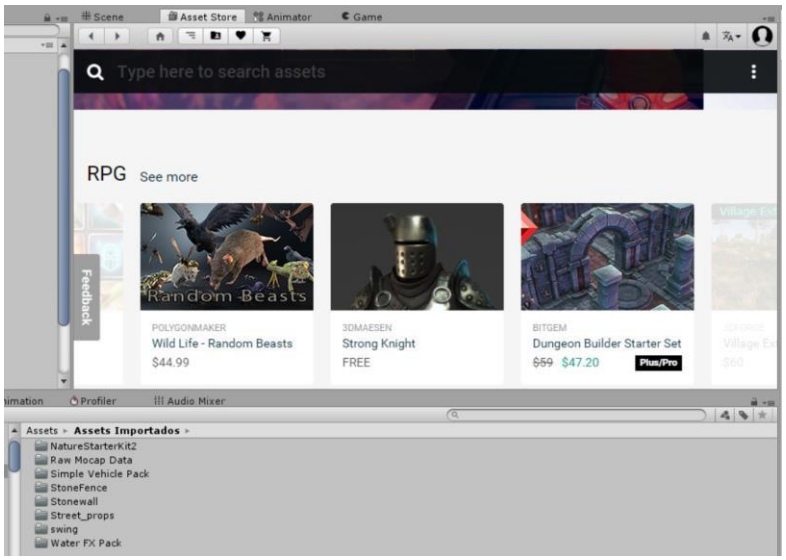


Figura 3-2: Pantalla del Asset Store

Realizado por: Christian Viracocha, Año 2018

Los Assets que se consiguieron de la tienda son:

Simple Cars Pack: En el interior del paquete de Asset se encuentran ocho diferentes modelos de carros que sirven como decoración dentro de la ciudad.

Swing: Es un objeto 3D en forma de un columpio de madera que sirve como decoración en el parque.

Park Props Pack: Es un paquete de objetos diseñados como decoración para un parque en el cual constan de sillas de parque y postes de luz.

GAZ Street Props: Es un paquete con objetos 3D con decoraciones que se encuentran en las calles de la ciudad, como por ejemplo los hidrantes, basureros y postes con diferentes modelos.

Dry Stone Wall with leafy Vines: Es el paquete con objetos 3D el cual cuenta con diferentes versiones de muros de piedra con vegetación para la ambientación del mundo virtual.

Stone Fence: Es el paquete con objetos 3D que consta de diferentes versiones de las vallas con cerramientos y sus respectivos postes para la ambientación del mundo virtual.

Simple Modular Street Kit: Es el paquete con diferentes objetos que servirán para la implementación de las calles dentro de la ciudad, con diferentes diseños.

Standard Assets: Es uno de los paquetes más importantes dentro de Unity, dado que contiene los diferentes scripts más básicos para la creación de un videojuego, como son las varias texturas que se usan en el mundo virtual, así como también la implementación de personajes jugables con sus respectivos scripts, entre muchos más.

Raw Mocap Data for Mecanim: Es el paquete de las animaciones para los personajes jugables y no jugables dentro del mundo virtual.

Nature Starter kit 2: Es el paquete de naturaleza el cual contiene los diferentes modelos 3D para la creación de un bosque como son los diseños de los árboles, hierbas, arbustos entre muchos más.

Water FX Pack: Es un paquete en el cual se encuentran los scripts que darán las animaciones de las partículas de agua, como también existen las texturas necesarias para dar el efecto de agua.
QS MAterials Nature – Pack Grass vol. 1: Es el paquete que consta de texturas de hierbas para el terreno.

2.1.3.7. Avatares

Dentro del videojuego existen dos tipos de Avatares: El principal es el jugador en 1ra persona el cual puede interactuar con el mundo virtual mediante el manejo del cursor de los dispositivos; el otro tipo de avatares, son los personajes no jugables los cuales se crean personas con diferentes tipos de modelos en 3D, donde habitan en el mundo virtual del juego.

2.1.3.8. Inteligencia artificial

Las técnicas de inteligencia artificial que se implementa dentro del juego, son las Máquinas de Estado Finito que se basa en una estructura del programa que permite determinar el comportamiento de un objeto o de la lógica del juego que tiene como finalidad crear un ambiente más inteligente y el uso del Path Planning o búsqueda de caminos que se utiliza dentro del juego, para dar independencia a los personajes no jugables que recorren la ciudad.

Dentro del juego se incorpora un Wizard, que usa Goal Oriented Action Planning, que se basa en el cumplimiento de metas del juego, el cual maneja los datos del jugador referente al puntaje que va ganando durante la partida, lo que permite informar al jugador, cuales son los test que aún no se han desarrollado o faltan de completar. Además otorga tips al jugador para manejar correctamente el juego.

Por lo tanto, dentro del proyecto se implementa la Inteligencia artificial dentro de los personajes no Jugables y Vehículos que mueven e interactúan con el jugador dentro de la ciudad virtual, los cuales caminan por la ciudad y podrán dar su conocimiento al usuario, así como también el manejo de la información del puntaje del jugador, el cual ayuda al momento de dar consejos al jugador.

2.2. Elaboración

2.2.1. Elaboración del Ambiente Virtual

Para la elaboración del ambiente virtual se toma como referencia los edificios de la ciudad de Riobamba creando una ciudad ficticia, en el que se implementa objetos en 3D diseñados desde

Sketchup y adquiridos mediante la tienda de Asset Store para la ambientación de la naturaleza y de la ciudad, como se muestra en la Figura 4-2. Ver **Anexo C**.

Figura 4-2: Ambiente de la Ciudad Virtual

Realizado por: Christian Viracocha, Año 2018

Para la creación de los personajes no jugables se utiliza el programa MakeHuman, que se basa en la construcción de personajes, que habitan el mundo virtual, por lo que se tomó en cuenta que todos los personajes en 3D deben tener un esqueleto base para la implementación de las animaciones del Juego, como se muestra en la Figura 5-2. Ver **Anexo C**.

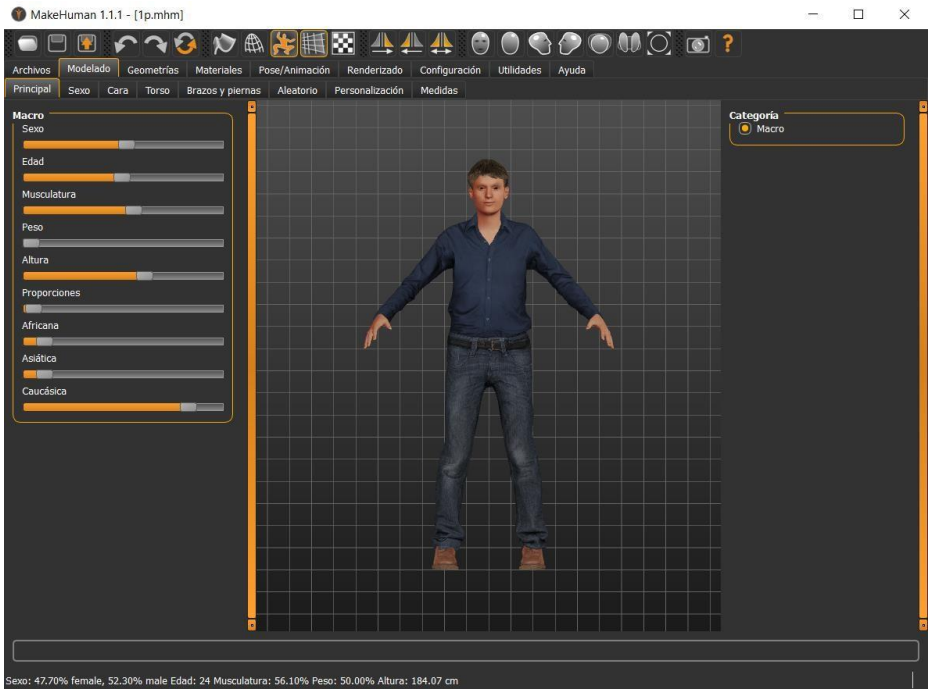


Figura 5-2: Pantalla de Makehuman

Realizado por: Christian Viracocha, Año 2018

Para la creación final del Ambiente se procede a colocar objetos en 3D así como la implementación del Skybox según se muestra en la Figura 6-2 para imitar el paisaje de un medio día en el mundo real mediante su configuración. También se toma en cuenta el sonido de la ciudad al momento del juego.

Figura 6-2: SkyBox del Mundo Virtual

Realizado por: Christian Viracocha, Año 2018

2.2.2. Elaboración de los Elementos 3D

La elaboración de los Objetos 3D que usa la ciudad virtual, se desarrolla a partir de imágenes de edificios reales de la Ciudad de Riobamba, utilizando el programa de SketchUp, el cual es un programa de diseño y modelado en 3D como se muestra en la Figura 7-2, los cuales fueron importados a Unity con la finalidad de crear el aspecto de la ciudad. Ver **Anexo C**.

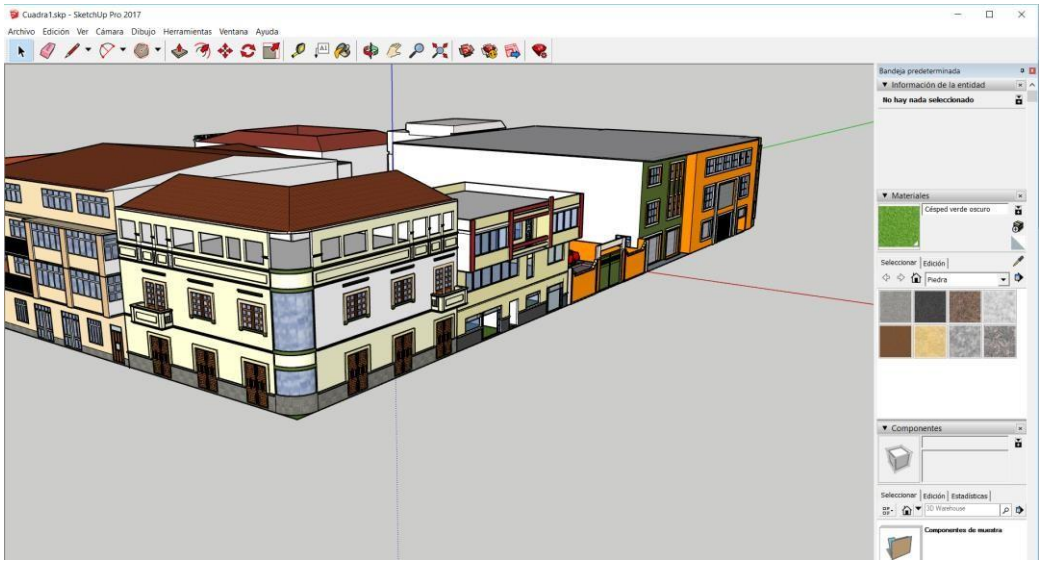


Figura 7-2: Modelo de la Cuadra en 3D

Realizado por: Christian Viracocha, Año 2018

Para elaborar los personajes no jugables se utiliza el programa MakeHuman el cual es un programa diseñado para la creación de avatares en 3D para videojuegos, en donde se crea 20 modelos diferentes de personajes llamados habitantes de la ciudad como se ve en la Figura 8-2.

Figura 8-2: Personajes no jugables en la ciudad virtual

Realizado por: Christian Viracocha, Año 2018

Las Texturas de los personajes son modificados al ser implementados en el mundo virtual, debido a que al importarlos a Unity se crean texturas metálicas, que no son aptas para el juego, por lo que se usan herramientas de Unity para configurarlas, con el fin de que los personajes se vean más reales como se muestra en la Figura 9-2. Ver **Anexo C**.

Figura 9-2: Pantalla de configuración de los NPCs

Realizado por: Christian Viracocha, Año 2018

2.2.3. Elaboración de los Personajes No Jugables

Para la elaboración de los personajes no jugables dentro del juego se utiliza los modelos en 3D, que se realizan en el programa MakeHuman, los cuales incorporan un esqueleto interno que permite manipular los movimientos de los avatares dentro del juego con la ayuda del sistema Animator de Unity, como se muestra en la Figura 10-2. Ver **Anexo C**.

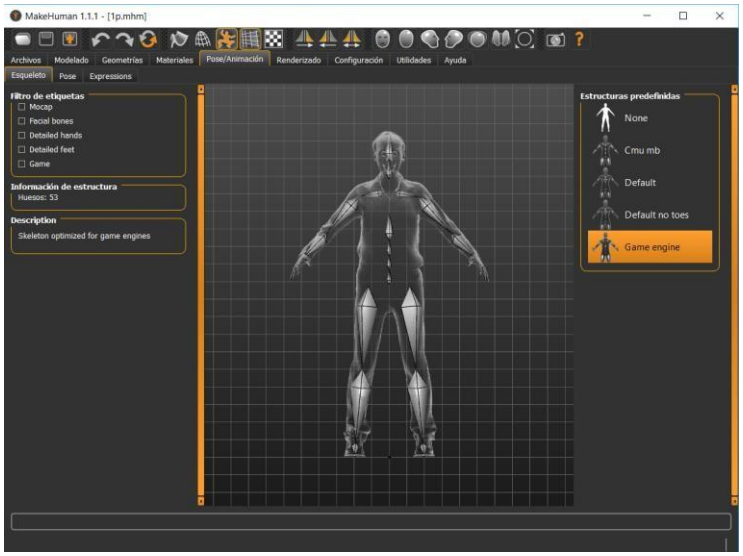


Figura 10-2: Pantalla de la Estructura de los NPC en MakeHuman

Realizado por: Christian Viracocha, Año 2018.

Se crea el archivo Animator Controller para la creación de las animaciones de los NPCs, con el fin de diseñar los diferentes tipos de comportamientos de los personajes, teniendo en cuenta el uso de la programación en base a la Máquina de Estados para la implementación de la Inteligencia Artificial dentro del personaje no jugable como se muestra en la Figura 11-2 y donde se crea la animación con diferentes estados. Ver **Anexo C**.

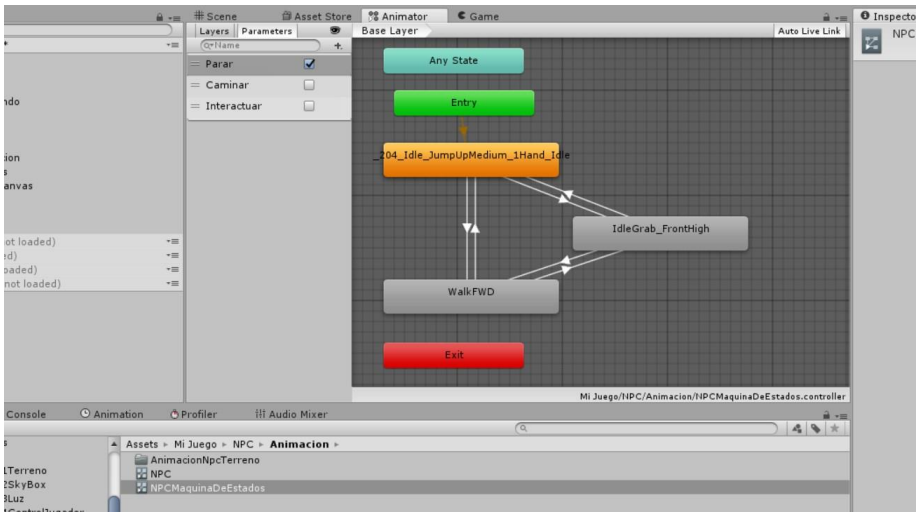


Figura 11-2: Animator Controller de la animación para los NPCs

Realizado por: Christian Viracocha, Año 2018.

Para Implementar la Inteligencia Artificial básica basada en la creación de máquinas de estado, se necesita crear el Script que controla el cambio de eventos del personaje basado en los Triggers, donde se encuentran los estados del personaje, el cual mide la distancia entre el jugador y lo ubica de forma automática, con la finalidad de que se logre la interacción entre el jugador y el NPC como se muestra en la Figura 13-2. Ver **Anexo C**.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MaquinaEstadoIA : MonoBehaviour {
    public Transform player;
    static Animator anim;


    void Start () {
        anim = GetComponent<Animator> ();

    }

    // Update is called once per frame
    void Update () {
        if (Vector3.Distance (player.position, this.transform.position) < 10) {

            Vector3 direccion = player.position - this.transform.position;
            direccion.y = 0;

            this.transform.rotation = Quaternion.Slerp
(this.transform.rotation, Quaternion.LookRotation (direccion), 0.1f);

            anim.SetBool ("Parar", false);

            if (direccion.magnitude > 5) {

                anim.SetBool ("Caminar", true);


                anim.SetBool ("Interactuar", false);


            } else {
                anim.SetBool ("Interactuar", true);
                anim.SetBool ("Caminar", false);
            }
        } else {
            anim.SetBool ("Parar", true);
            anim.SetBool ("Caminar", false);
            anim.SetBool ("Interactuar", false);
        }


    }
}
```

Figura 13-2: Script para el manejo del cambio de estados con la animación del NPC

Realizado por: Christian Viracocha, Año 2018.

Para que el personaje jugable pueda moverse con facilidad en el mundo virtual se crea un mapa de navegación del juego como se muestra en la Figura 14-2, en el que se definen los objetos estáticos del juego y las elevaciones máximas que pueda el personaje acceder, con todo esto el personaje podrá movilizarse y elegir el mejor camino para llegar a sus objetivos en el mundo virtual.

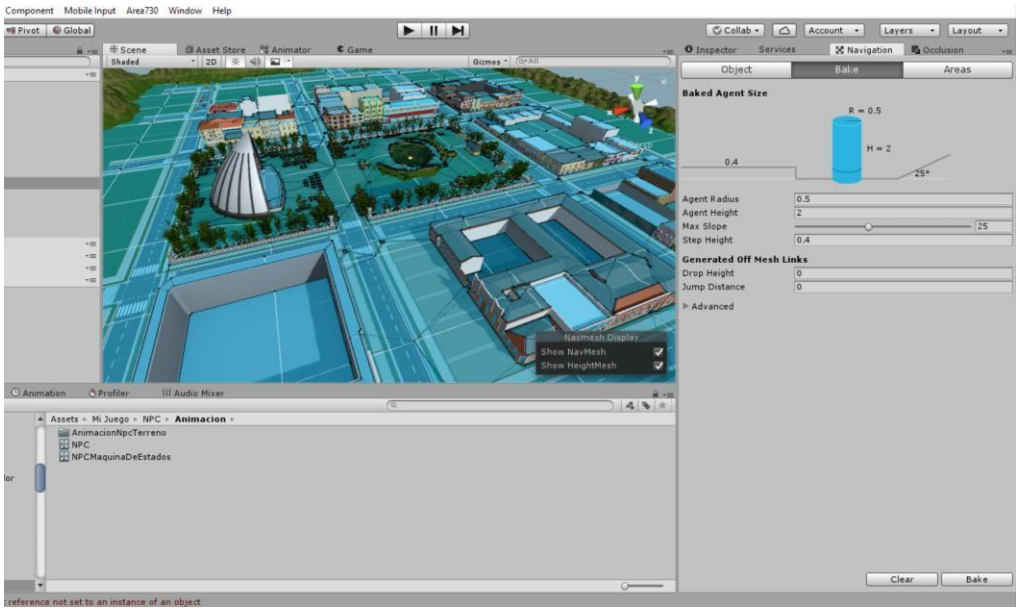


Figura 14-2: Mapa de Navegación

Realizado por: Christian Viracocha, Año 2018

2.2.4. Elaboración de la Información del Tutorial

Para la elección del material educativo implementado en el juego, se identifica los principales temas de utilización y creación de juegos en el motor de videojuegos Unity, basándose en tutoriales realizados por los programadores de videojuegos, por lo cual se clasifica en seis temas básicos, los cuales son:

- **Introducción a Unity:** Se centra en los menús y en la estructura de la interfaz de Unity necesarias para su utilización, también sus características en la creación de juegos.
- **Creación del Terreno:** Es el primer paso para diseñar un videojuego en 3D, en el que se diseña el terreno para el mundo virtual. Aquí se enseña cómo modificar la arquitectura base del terreno, las elevaciones o gradientes del terreno y la implementación de las texturas primarias y secundarias así como también la importación de objetos en 3D.
- **Implementación del Skybox:** Es el segundo paso para la creación de un videojuego en 3D, en el que se basa en la implementación del cielo, el cual dará el efecto de la iluminación, con el fin de observar el fondo de un cielo más allá del horizonte del mundo virtual.
- **Implementación de Objetos de Luz:** Es el tercer paso para el desarrollo de un videojuego el mismo que permite implementar diferentes efectos de luz para cualquier escena, como la presencia de un sol, la luz de los faros de los carros, explosiones de luz, entre otros, con el fin de mejorar el ambiente del juego.
- **Implementación de Personaje Jugable:** En este paso se crea el personaje jugable que puede ser en primera persona o en tercera persona, de acuerdo al tipo de juego que se

desarrolle, en el que se observa los diferentes tipos de configuración que se puede cambiar para el manejo del personaje.

- **Creación básica de los Scripts:** En este paso se identifica como crear los scripts para la implementación de la lógica y comportamiento de los objetos dentro del juego, teniendo en cuenta las principales características de un Script basado en el código de Unity.

2.2.5. Elaboración de los scripts

Los Script son desarrollados dentro del Ambiente de Desarrollo integrado MonoDevelop de Unity 3D, basados en los lenguajes nativos de JavaScript para la creación del Menú de Acceso y C# para la creación de la lógica y del comportamiento de los personajes no jugables como también los jugables.

Dentro del juego, los Scripts se usan para implementar las diferentes técnicas de Inteligencia Artificial como también el control de la información de los datos de aprendizaje y las calificaciones, que sirven para ver el avance del juego y a la vez su nivel de aprendizaje que adquiere el jugador dentro del juego mediante los Test como se muestra en la Figura 15-2, si desea ver más acerca de los scripts ver el **Anexo C**.

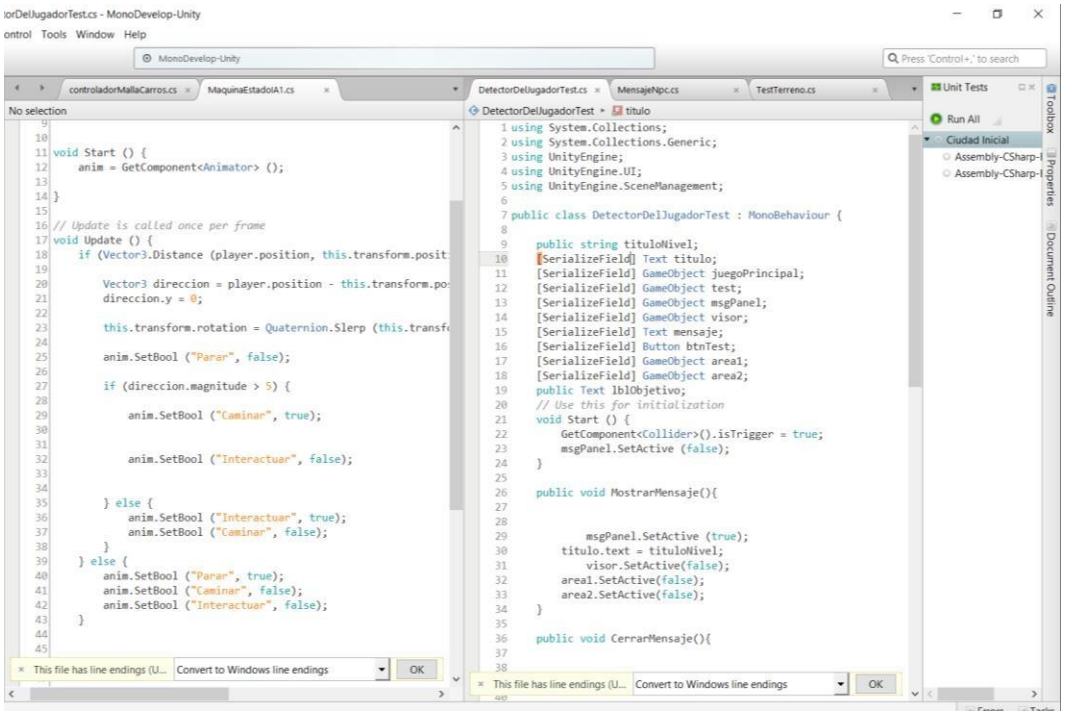


Figura 15-2: Pantalla de Scripts para la Maquina de Estados

Realizado por: Christian Viracocha, Año 2018

En la Figura 16-2 se observa parte del código que se usa para la construcción del menú en la que se establece las opciones de modificación básica para el juego, como son el manejo del

audio del videojuego o la calidad de las gráficas que se basan en el manejo de las texturas y luces del juego, si desea ver más acerca del Script ver el **Anexo C**.

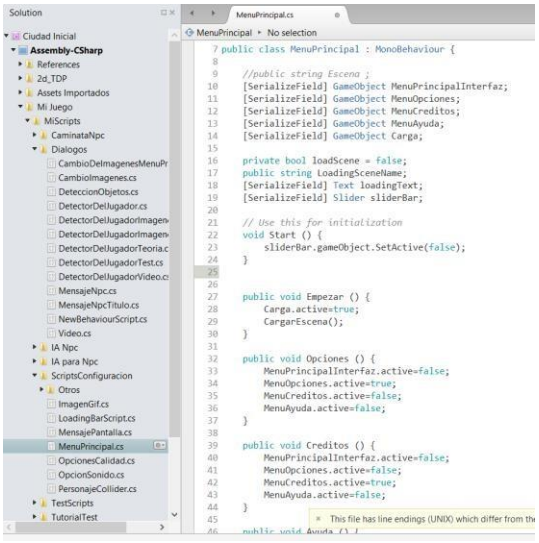


Figura 16-2: Pantalla de Scripts para el Menú Principal

Realizado por: Christian Viracocha, Año 2018

El script que se usa para la implementación de la inteligencia artificial, basado en una máquina de estados, se conforma por varias configuraciones dentro del sistema, como son:

- La creación de la animación básica del personaje no jugable, la cual sirve para dar movimiento al personaje
- La construcción del archivo Animator Controller la cual establece los estados del personaje no jugable, en la que se elige la animación a realizarse.
- El diseño del mapa de navegación del terreno busca identificar los objetos estáticos, con la finalidad de que el personaje no jugable pueda tomar sus propias decisiones al momento de caminar por la ciudad virtual.
- La creación del Script tiene como objetivo reconocer diversos eventos que se presentan en el mundo virtual y que conlleva a integrar elementos creados anteriormente. Cada personaje conoce su entorno, lo que le permite tomar decisiones cuando ocurran diversos eventos a su alrededor, si desea saber más ver **Anexo C**.

En la Figura 17-2 se visualiza el diagrama de la máquina de estados que se usa para los personajes no jugables especiales que habitan dentro del juego, en donde se vizualizan los diferentes estados en los que se encuentra cada personaje, por lo que dependiendo de cada jugador el estado del personaje cambia, mientras que, para los personajes no jugables normales, su comportamiento cambia de acuerdo al lugar en dónde se encuentren.

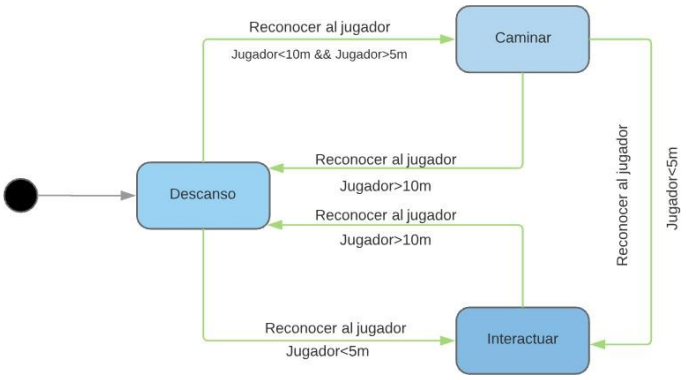


Figura 17-2: Diagrama de la Maquina de Estados

Realizado por: Christian Viracocha, Año 2018

El manejo de los Paneles que muestra la información de Unity para el aprendizaje se necesita de la creación del Script del Controlador de Colliders que reconoce el tipo de mensaje que se deberá mostrar en el juego y el Script de manejo de Canvas que se encuentran dentro de cada uno de los personajes no jugables, los cuales se guían por la tabla de calificaciones que muestran el progreso del juego.

2.2.6. Creación del Producto Alfa

El Producto Alfa es la primera versión, que se utiliza para establecer las características y aspectos que faltan en el juego para la creación del producto final, a consecuencia de que esta versión no es completamente funcional.

El juego está constituido por el menú principal que se muestra en la Figura 18-2 y consta de 5 opciones principales que son:
Iniciar Juego: Permite ingresar directamente al juego.
Configuraciones: Se muestran las configuraciones básicas del juego.
Ayuda: Visualiza los aspectos básicos del juego.
Créditos: Muestra la información básica del juego.
Salida: Permite salir del juego.

Figura 18-2: Menú Principal

Realizado por: Christian Viracocha, Año 2018

El juego inicia en primera persona, en el que se muestra el visor del personaje, en donde se ubica El Score, La barra de Progreso, El mini mapa, La lista de la calificación de los Test y la Guía de Objetivos que se tienen que cumplir como muestra la Figura 19-2.

Figura 19-2: Visor del Jugador, Versión Alfa

Realizado por: Christian Viracocha, Año 2018

La ciudad virtual en el que habita el jugador se encuentra dividido en sectores como se observa en la Figura 20-2, el cual contiene las diferentes áreas en donde se encuentra ubicado el material educativo y son representados por los NPC´s.

Figura 20-2: Sectores de la Ciudad Virtual

Realizado por: Christian Viracocha, Año 2018

Cuando el personaje se acerca a los NPC´s de los objetivos, éstos muestran diferentes tipos de material educativo que poseen. Además su comportamiento difiere de los demás ya que poseen diferentes características en su configuración base.

Cuando el jugador realiza alguna interacción con los NPC´s que sea parte de los objetivos muestra un panel con la información de Unity como se puede ver en la Figura 21-2, el cual aumenta el Score, Barra de Progreso y además actualiza la bandeja de objetivos del juego.

Figura 21-2: Sectores de la Ciudad Virtual

Realizado por: Christian Viracocha, Año 2018

En la reunión de la presentación del producto alfa, se vio la necesidad de mejorar la interfaz del usuario y la incorporación de algoritmos inteligentes para la creación de un auxiliar, que ayude al jugador en el trascurso del juego, tomando en cuenta los resultados que va adquiriendo el juego mientras juega.

2.3. Producto Beta

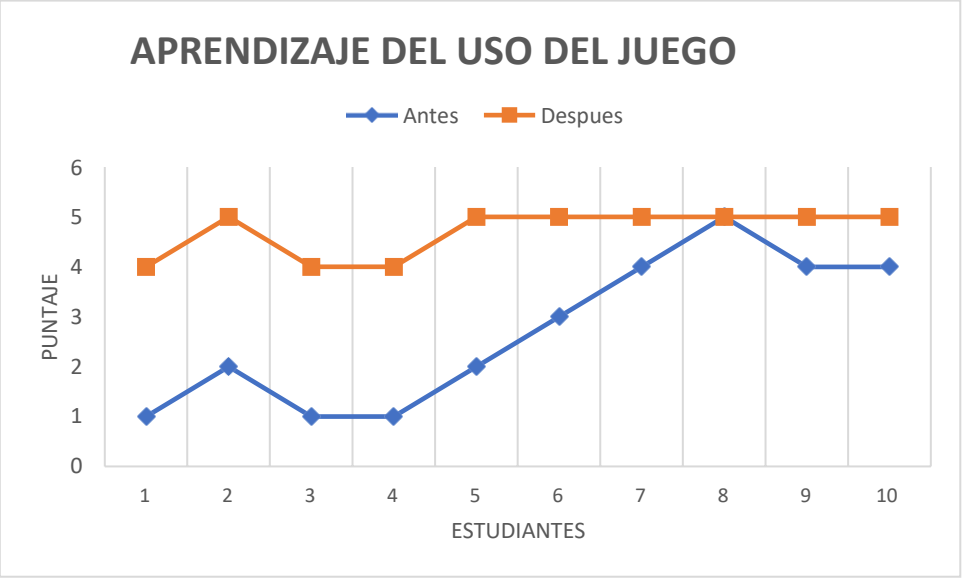
El Producto Beta final que se obtuvo, es el juego para la etapa de pruebas con los estudiantes para medir la usabilidad y funcionalidad del Serious Game. Debido a que se arregló la interfaz del juego al ambientarlo con más vida a la ciudad virtual en 3D y la mejora en la parte del material educativo que se usa en el juego.

Además se implementa un ayudante quien otorga al jugador mensajes de ayuda para manipular el juego y avisos para identificar objetivos que no se han completado aún, mediante el puntaje que se almacena dentro del juego, como se muestra en la Figura 22-2.

Figura 22-2: Visor del Jugador, Versión Beta

Realizado por: Christian Viracocha, Año 2018

Para lo cual se prueba el juego con estudiantes en donde se evalúan los conocimientos del antes y después de usar el juego, con el fin de obtener resultados para la curva de aprendizaje como se muestra en la Gráfica1-2 y en donde se observa como han incrementado sus conocimientos.



Gráfica 2-2: Gráfica de Aprendizaje

Realizado por: Christian Viracocha, Año 2018

Dentro de la curva de aprendizaje se determinó un incremento en los conocimientos de los jóvenes al usar el juego, visto que al principio no tenían idea de cómo usar el sistema Unity y ahora ya poseen los conocimientos básicos para crear un mundo virtual.

La curva de dificultad que se muestra en la Gráfica 2-2, muestra que al inicio del juego los participantes tienen dificultades al comenzar a usarlo, pero mientras más utilizado sea se hace más simple su manejo por lo que mejora su habilidad en la jugabilidad e interacción dentro del mundo virtual.

Gráfica 2-2: Gráfica del Nivel de Dificultad

Realizado por: Christian Viracocha, Año 2018

2.4. Cierre

Para la liberación del juego se corrige todos los errores que se presentan en la fase del producto Beta, donde se prueba el juego con dos grupos de estudiantes y se evalúan los diferentes aspectos de usabilidad, funcionalidad en su jugabilidad y manejo, mediante encuestas y pruebas del juego.

El cual es supervisado por el docente a cargo, una vez de haber cumplido con todos los requisitos del juego. Para más información ver el Capítulo 3 donde se explica cómo se realizó la prueba del sistema y que resultados se obtuvo de ello.

Después de haber culminado con éxito las pruebas necesarias, el Serious Game queda liberado para la entrega y demostración al responsable del proyecto. Evidenciando de esta manera los resultados obtenidos de la evaluación del producto, con la finalidad que de que el responsable observe el resultado y beneficios que atrae el uso de los Serious Game en la enseñanza.

2.5. Gestión de Riesgos

La fase de gestión de riesgos se mantiene activo durante todo el proceso del proyecto con el fin de minimizar y gestionar posibles riesgos que puedan presentarse en el proyecto. Para lo cual se hace el análisis de los posibles riesgos con la finalidad de evitarlos si es posible, caso contrario gestionar de la mejor manera para minimizar el impacto en el desarrollo del proyecto. Para mayor información ver el **Anexo D.**

CAPÍTULO III

3. Marco de Resultados

En este capítulo se muestran los resultados obtenidos en las pruebas que se realizaron del Serious Game a los grupos de estudiantes de la Facultad de Informática y Electrónica, mediante una sesión, en las que se midió el grado de los conocimientos de los jugadores, la usabilidad y funcionalidad del Serious Game.

3.1. Recolección de Datos

Para recolectar los datos resultantes del uso del Serious Game se utilizaron dos técnicas de recolección de datos como es la Observación, en la que se vio como los estudiantes manipulaban el juego y el cuestionario para recopilar información referente al conocimiento y funcionalidad del mismo, posteriormente se realizó una sesión de pruebas con un grupo de estudiantes que se dividió en cuatro fases:

1. **Fase 1**: Se realizó una Introducción del Serious Game.
2. **Fase 2:** Se llenó la primera sección del Test, para medir sus conocimientos básicos acerca de Unity.
3. **Fase 3**: Se realizó la prueba del Serious Game, cumpliendo ciertos objetivos durante el juego.
4. **Fase 4**: Se terminó de llenar las demás secciones del Test, en la cual se midió la usabilidad, funcionalidad y el nivel de conocimientos adquiridos después de jugar.

Para ver los formatos de la encuesta y ficha de la observación ver el **Anexo B**.

3.2. Fases de la Recolección de Datos

3.2.1. Fase 1

En la fase 1 de la recolección de datos, se reunió a un grupo de diez estudiantes para la prueba del Serious Game ¨Unity City¨ en la cual se dio la explicación del objetivo donde se centra en el aprendizaje y manejo del motor de videojuegos Unity 3D, mediante la información relevante y pruebas que medirán su nivel de conocimientos; además se explicó cuáles son los controles necesarios que deben saber para jugar.

3.2.2.Fase 2

En la fase 2 de la recolección de datos, se les otorgó a los estudiantes un Test el cual estuvo dividido en cuatro secciones, los cuales ayudarían a medir las diferentes variables como el nivel de conocimientos hasta llegar a la usabilidad que presenta el juego.

Por lo que se solicitó a los estudiantes llenar la primera sección del test, el cual se midió el nivel de conocimientos básicos de lo que es Unity, con el fin de comparar los resultados de la fase 4 y ver si hay un aumento en los conocimientos del jugador después de haber probado el Serious Game.

3.2.3. Fase 3

En la fase 3 de recolección de datos, se centró en el uso del Serious Game por parte de los estudiantes, en el que se les pidió cumplir diferentes objetivos dentro del juego, con el fin de poder identificar la existencia de los problemas y contribuir de esta manera a la toma de datos en base a las experiencias de los jugadores.

A los participantes se les pidió como objetivos, lo siguiente:

- Recorrer la ciudad virtual revisando cada uno de los NPC´s objetivos de la primera área del juego.
- Identificar al NPC que tiene la información de la Teoría del Área Seleccionada.
- Identificar al NPC que tiene el video tutorial de alguna de las áreas.
- Realizar un Test de una de las áreas seleccionadas.
- Identificar los Tips del Juego dentro de la ciudad virtual.

3.2.4. Fase 4

En la fase final de recolección de datos, se pidió a los estudiantes completar el Test que se les entregó al principio de la sesión de prueba, con la finalidad de medir la usabilidad, funcionalidad y el conocimiento del estudiante después de haber participado en el juego, para ser comparados con la información que se adquirió al inicio de la sesión.

3.3. Planificación

Para la recolección de datos se planteó usar dos grupos de estudiantes con y sin experiencia enel manejo de diseños de programas, por lo que se hizo una planificación como se muestra en la Tabla 1-3.

Tabla 1-3: Tabla de Planificación para la Recolección de Datos

	Grupo A Sin conocimientos de Unity	**Grupo B Conocimientos en Programación**
Tema	Serious Games ¨Unity City¨	Serious Games ¨Unity City¨
Hora	10:00 am hasta las 12:00	14:00 am hasta las 16:00
Fecha	06-Julio-2018	06-Julio-2018
Números de Estudiantes	5	5
Encargado	Ing. Fernando Proaño	Ing. Fernando Proaño

Realizado por: Christian Viracocha, 2018

3.4. Resultados y Análisis de la Recolección de Datos

Después de haber recolectado los datos mediante la sesión de pruebas del Serious Game, usando el test y una hoja de control para la observación, se analizó los datos tomando en cuenta la secuencia de cada sección del test.

Los resultados que se obtuvieron al inicio de la primera sección del test tuvieron como finalidad medir el nivel de conocimientos de cada jugador acerca del uso del motor gráfico de Unity 3D, por lo que se obtuvo la valoración como se muestra en la Tabla 2-3.

Tabla 2-3: Valoración de los Resultados de la Sección 1 del Test.

Grupo A Sin conocimientos de Unity	**Grupo B Conocimientos en Programación**
1	3
2	4
1	5
1	4
2	4

Realizado por: Christian Viracocha, 2018

Durante la realización de la etapa de la recolección de datos en la que se prueba el Serious Game, se obtuvo los resultados de la observación, donde se midió el nivel de abstracción del juego por parte de los estudiantes al momento de jugar, dado que se observó cómo ellos interactuaban y comprendían el material educativo que se les presentó, como se muestra en la Tabla 3-3.

Tabla 3-3: Valoración de los resultados de la observación

Parámetros de Evaluación	**Grupo A Sin conocimientos de Unity**	**Grupo B Conocimientos en Programación**
Manejo del Sistema	3	4
Emoción	4	3
Lógica de Manejo	5	5

Localización de Objetivos	5	5
Encontrar soluciones	5	5
Identificación de personajes	5	5
Cumplimiento de objetivos	4	4
Facilidad en el manejo de controles	4	5
Manejo de la Interfaz	5	5
Adquisición de conocimientos	5	5

Realizado por: Christian Viracocha, 2018

Una vez terminado la etapa en la que se prueba el sistema, se ordenó terminar el Test, que llenaron al inicio de la recolección de datos, donde se mide la usabilidad, funcionalidad y nivel de conocimientos adquiridos al usar el juego, por lo que se obtuvo los resultados como se muestran en la Tabla 4-3.

Tabla 4-3: Valoración del Promedio de los Resultados de la Sección 2 al 3 del Test

Parámetros de Evaluación	**Grupo A Sin conocimientos de Unity**	**Grupo B Conocimientos en Programación**
Sección 2 (Sobre 10 Puntos)		
Diversión	8.6	8.4
Útil	9.6	9.2
Adaptación del jugador	9.4	9.6
Dificultad	6.4	3.2
Aprobación del jugador	9	82
Aceptación	9.8	9
Sección 3 (Sobre 5 Puntos)		
Efectividad	4.4	4.4
Eficiencia	4.2	4.4
Inmersión	4	3.4
Motivación	4.2	3.4
Aprendizaje	4.8	4.4
Emoción	4.2	4.2
Satisfacción	4.2	4.2

Realizado por: Christian Viracocha, 2018

En la Tabla 5-3 se visualizan los resultados obtenidos de la sección 4 del Test en el que se realiza un cuestionario, donde se mide el nivel de conocimientos que tienen los estudiantes después de haber jugado y así poder compararlos con los resultados que se obtuvieron al inicio de la toma de resultados.

Tabla 5-3: Valoración de los Resultados de la Sección 4 del Test

Grupo A **Sin conocimientos de Unity**	**Grupo B** **Conocimientos en Programación**
4	5
5	5
4	5
4	5
5	5

Realizado por: Christian Viracocha, 2018

Una vez obtenidos los resultados de la prueba del Serious Game se realizó la prueba estadística de significancia con el objetivo de encontrar las diferencias medias entre los dos grupos mediante los puntajes que se muestran en la Tabla 6-3.

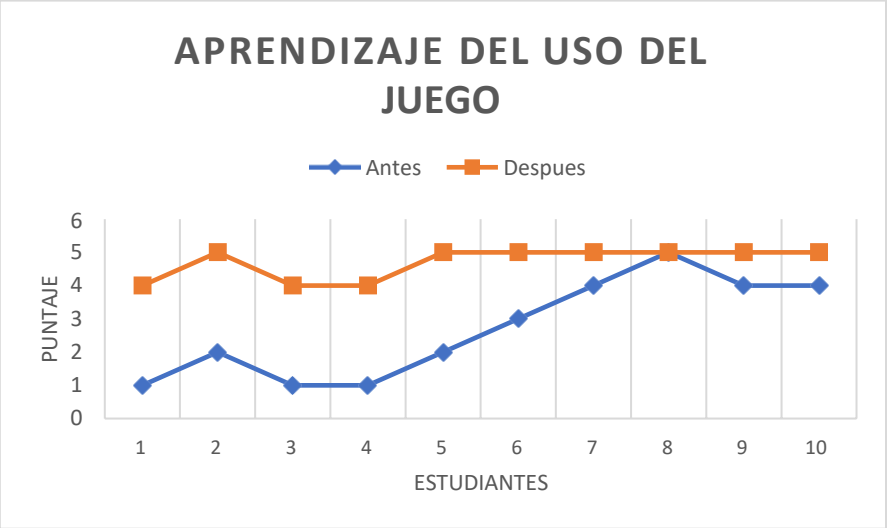
Tabla 6-3: Diferencia de los puntajes obtenidos de la Sección 1 y Sección 4

Sección 1			**Sección 4**		
Grupo A Sin conocimientos de Unity	**Grupo B Conocimientos en Programación**	**Diferencia**	**Grupo A Sin conocimientos de Unity**	**Grupo B Conocimientos en Programación**	**Diferencia**
1	3	2	4	5	1
2	4	2	5	5	0
1	5	4	4	5	1
1	4	3	4	5	1
2	4	2	5	5	0

Realizado por: Christian Viracocha, 2018

De los resultados de la Tabla 6-3 se concluyó que la diferencia significativa entre el Grupo A y el Grupo B es mayor, antes de probar el Serious Game ya que los conocimientos entre ambos grupos son grandes, pero al haber usado el juego se logró minimizar la diferencia significativa entre ambos, ya que los conocimientos de los grupos A y B se igualaron.

En la Gráfica 1-3 se pudo observar comó el nivel de aprendizaje de los estudiantes fue en ascenso correspondiendo al objetivo del Serious Game, a consecuencia de que antes de usar el juego, los niveles de conocimientos eran muy bajos y de acuerdo con el pasar del tiempo estos fueron incrementándose exponencialmente.

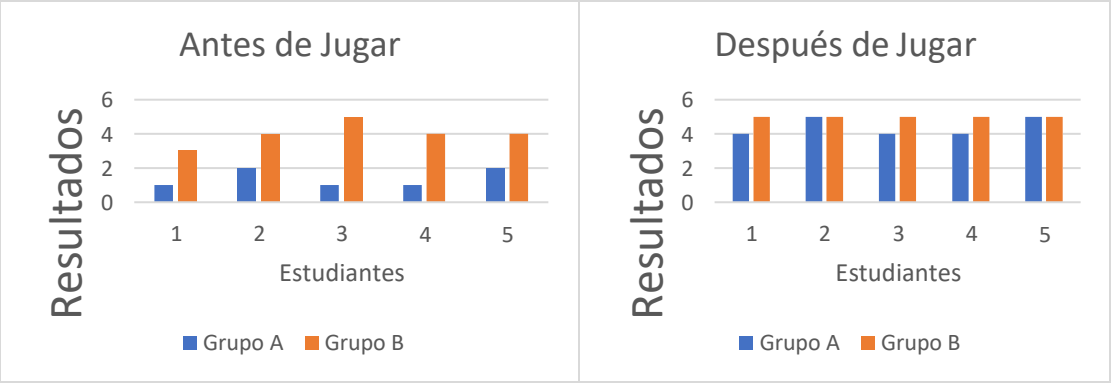


Gráfica 1-3: Nivel de aprendizaje del Serious Game

Realizado por: Christian Viracocha, Año 2018

De acuerdo con la Gráfica 2-3, en la que se observan los resultados que se obtuvieron del antes y después de jugar se puede concluir que antes de jugar la diferencia de conocimientos entre los dos grupos es notable, visto que el Grupo A no posee ningún conocimiento de Unity mientras que el Grupo B tiene algo de experiencia.

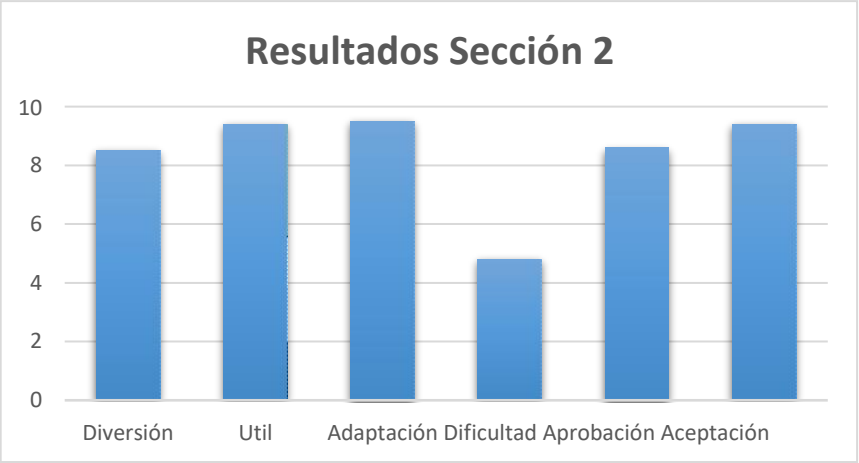
Después de jugar el Serious Game el Grupo A igualó también sus conocimientos, casi al mismo nivel que el grupo B, aunque ellos también incrementaron sus conocimientos, por lo que se concluye que los dos grupos elevaron en gran medida sus conocimientos, al casi igualarse en sus resultados.



Gráfica 2-3: Comparación del Nivel de Aprendizaje del Serious Game

Realizado por: Christian Viracocha, Año 2018

En relación con la funcionalidad y usabilidad se utilizó los resultados obtenidos del test después de que los estudiantes jugaron, en la cual se comprobó el funcionamiento y adaptabilidad del Serious Game, mediante los promedios de cada parámetro de la Tabla 4-3, en la cual se obtuvieron los resultados que se muestran en el Figura 3-3.

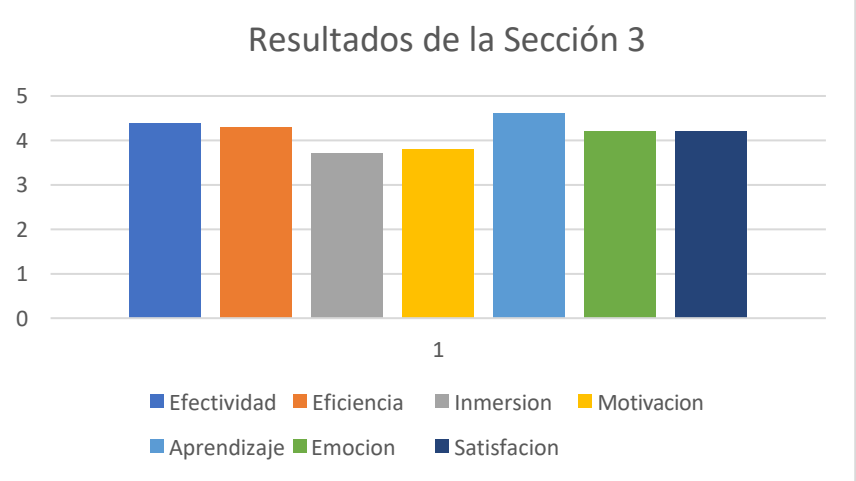


Gráfica 3-3: Resultados de la Sección 2 del Test de Usabilidad

Realizado por: Christian Viracocha, Año 2018

En la Gráfica 3-3 se pudo observar la experiencia que tuvo el estudiante al manipular el juego en donde se concluyó que, de acuerdo a los resultados obtenidos, el jugador ha sabido comprender con éxito los controles e identificar el objetivo primario del juego. También se pudo apreciar que el jugador se adaptó rápidamente al mundo virtual y no encontró desbalance alguno en la interfaz del juego.

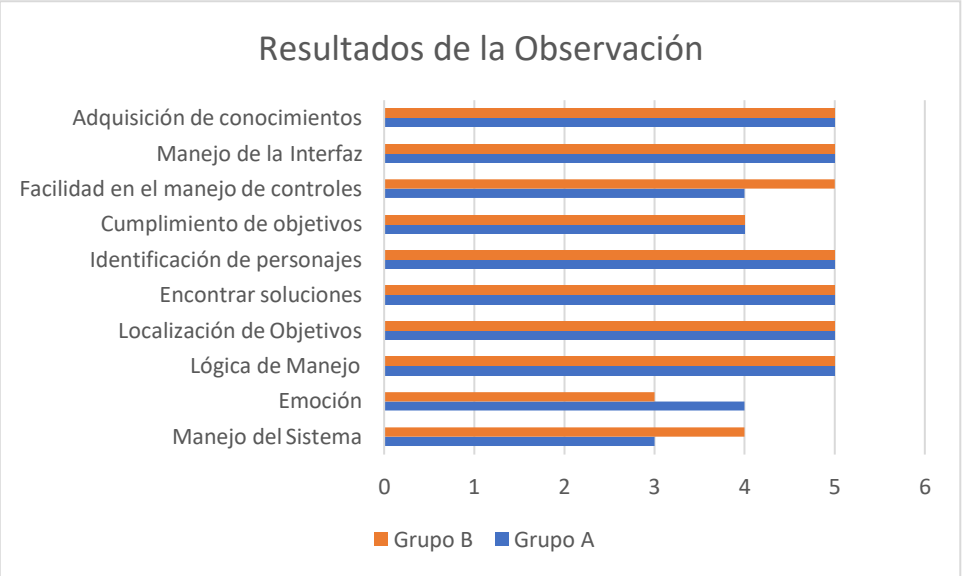
De acuerdo con la Gráfica 4-3, donde se muestra los resultados de la sección 3 del test realizado, se analizaron los parámetros con el que fue evaluado, determinando así la efectividad y eficiencia del juego y de esta manera da lugar a que el juego cumpla con el objetivo base de los Serious Games, a consecuencia de que el juego no es tan complicado al momento de usarlo, se adapta al jugador, es divertido y el objetivo principal es la enseñanza del motor de videojuegos Unity 3D.



Gráfica 4-3: Resultados de la Sección 4 del Test de Usabilidad

Realizado por: Christian Viracocha, Año 2018

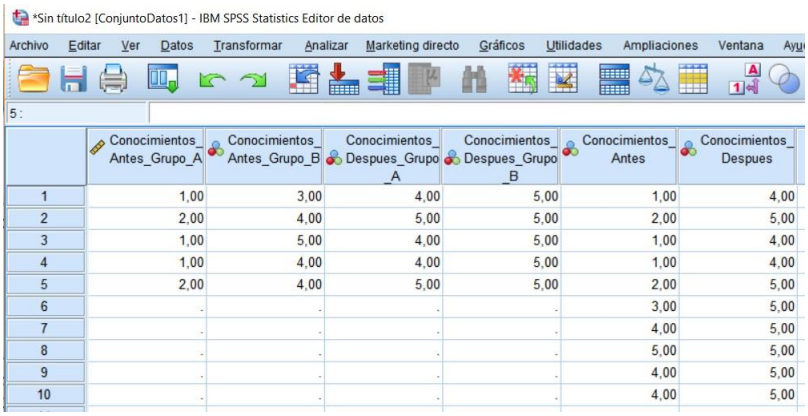
Durante la recolección de datos usando el método de Observación, se obtuvieron los resultados que se muestran en la Tabla 3-3 en donde se puede constatar que los jugadores al principio del juego no dominaban los controles con fluidez, pero al pasar el tiempo, los estudiantes pudieron dominar los controles del juego y adaptarse a la interfaz como se muestra en la Gráfica 5-3.



Gráfica 5-3: Resultados de la de la Ficha de Observación

Realizado por: Christian Viracocha, Año 2018

Para el análisis a nivel estadístico de los resultados obtenidos del test que se realizó al probar el juego utilizando a dos grupos de estudiantes: antes y después de jugar se utilizó la técnica de T-Student de muestras relacionadas, utilizando el programa IBM SPSS STAT para calcular los resultados que se muestra en la Figura 1-3.


*Sin título2 [ConjuntoDatos1] - IBM SPSS Statistics Editor de datos

Archivo Editar Ver Datos Transformar Analizar Marketing directo Gráficos Utilidades Ampliaciones Ventana Ayu

5 :

	Conocimientos_Antes_Grupo_A	Conocimientos_Antes_Grupo_B	Conocimientos_Despues_Grupo_A	Conocimientos_Despues_Grupo_B	Conocimientos_Antes	Conocimientos_Despues
1	1,00	3,00	4,00	5,00	1,00	4,00
2	2,00	4,00	5,00	5,00	2,00	5,00
3	1,00	5,00	4,00	5,00	1,00	4,00
4	1,00	4,00	4,00	5,00	1,00	4,00
5	2,00	4,00	5,00	5,00	2,00	5,00
6	.	.	.	.	3,00	5,00
7	.	.	.	.	4,00	5,00
8	.	.	.	.	5,00	5,00
9	.	.	.	.	4,00	5,00
10	.	.	.	.	4,00	5,00

Figura 1-3: Pantalla de los datos del Test de Conocimientos en el Programa IBM SPSS

Realizado por: Christian Viracocha, Año 2018

Hipótesis del antes de jugar ¨Unity City¨

Hipótesis Nula

Ho: No existen diferencias entre los conocimientos de los estudiantes de los dos grupos que fueron asignados para probar el juego ¨Unity City¨.

Hipótesis de investigación

H1: Existen diferencias entre los conocimientos de los estudiantes de los dos grupos que fueron asignados para probar el juego ¨Unity City¨.

$\propto$=**Porcentaje de error=** 0.05

Para el cálculo de los datos descriptivos de los resultados que se obtuvieron de la sección 1 del test, se utilizó el programa de IBM como se muestra en la Figura 2-3, donde se muestra la información necesaria para el cálculo de la prueba de T-Student.

Descriptivos

			Estadístico	Error estándar
Conocimientos_Antes_Grupo_A	Media		1,4000	,24495
	95% de intervalo de confianza para la media	Límite inferior	,7199	
		Límite superior	2,0801	
	Media recortada al 5%		1,3889	
	Mediana		1,0000	
	Varianza		,300	
	Desviación estándar		,54772	
	Mínimo		1,00	
	Máximo		2,00	
	Rango		1,00	
	Rango intercuartil		1,00	
	Asimetría		,609	,913
	Curtosis		-3,333	2,000
Conocimientos_Antes_Grupo_B	Media		4,0000	,31623
	95% de intervalo de confianza para la media	Límite inferior	3,1220	
		Límite superior	4,8780	
	Media recortada al 5%		4,0000	
	Mediana		4,0000	
	Varianza		,500	
	Desviación estándar		,70711	
	Mínimo		3,00	
	Máximo		5,00	
	Rango		2,00	
	Rango intercuartil		1,00	
	Asimetría		,000	,913
	Curtosis		2,000	2,000

Figura 2-3: Pantalla de los resultados descriptivos de los conocimientos antes de probar el juego

Realizado por: Christian Viracocha, Año 2018

Con la información anterior se puede calcular el valor base para identificar la hipótesis válida como se muestra en la Figura 3-3 en donde se visualiza toda la información necesaria para la prueba del T-Student con muestras relacionadas.

Estadísticas de muestras emparejadas

		Media	N	Desviación estándar	Media de error estándar
Par 1	Conocimientos_Antes_Grupo_A	1,4000	5	,54772	,24495
	Conocimientos_Antes_Grupo_B	4,0000	5	,70711	,31623

Prueba de muestras emparejadas

		Diferencias emparejadas					t	gl	Sig. (bilateral)
		Media	Desviación estándar	Media de error estándar	95% de intervalo de confianza de la diferencia Inferior	Superior			
Par 1	Conocimientos_Antes_Grupo_A - Conocimientos_Antes_Grupo_B	-2,60000	,89443	,40000	-3,71058	-1,48942	-6,500	4	,003

Figura 3-3: Pantalla de resultados para la prueba de las muestras emparejados de los conocimientos antes de probar el juego

Realizado por: Christian Viracocha, Año 2018

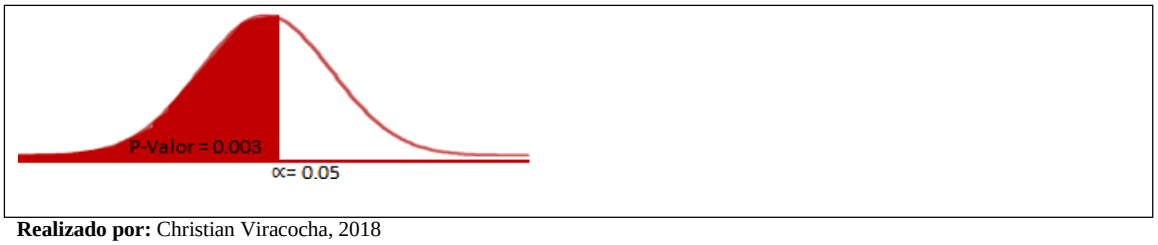
Criterios para decidir la hipótesis:

Para la toma de la hipótesis correcta en el proceso de validación del proyecto antes de ser probado, se utilizan los siguientes criterios, los cuales son analizados en la Tabla 7-3.

- Si la probabilidad obtenida es P-valor <= ∝ se rechaza la Ho.
- Si la probabilidad obtenida es P-valor > ∝ no se rechaza la Ho.

Tabla 7-3: Tabla de la decisión estadística final del antes de probar el juego

Decisión Estadística		
P-Valor = 0.003	<	∝= 0.05
Conclusión: Se acepta la hipótesis de investigación, por lo que si existen diferencias de conocimientos de los estudiantes entre los dos grupos que fueron asignados para probar el juego ¨Unity City¨. Por lo que la diferencia de conocimientos que existen entre los dos grupos es de 2.6 tal como se muestra en la Figura 3-3.		
Gráfica		



Realizado por: Christian Viracocha, 2018

Hipótesis del después de jugar ¨Unity City¨

Hipótesis Nula

Ho: No existen grandes diferencias de conocimientos de los estudiantes entre los dos grupos que jugaron el juego ¨Unity City¨.

Hipótesis de investigación

H1: Existen grandes diferencias de conocimientos de los estudiantes entre los dos grupos que jugaron el juego ¨Unity City¨.

∝=**Porcentaje de error=** 0.05

Para el cálculo de la segunda hipótesis se obtuvieron los resultados descriptivos usando los valores de la tercera sección del test con el programa de IBM que se muestra en la Figura 4-3 con el fin de calcular la prueba de T-student de las muestras relacionadas.

Descriptivos[a]

			Estadístico	Error estándar
Conocimientos_Despues_Grupo_A	Media		4,4000	,24495
	95% de intervalo de confianza para la media	Límite inferior	3,7199	
		Límite superior	5,0801	
	Media recortada al 5%		4,3889	
	Mediana		4,0000	
	Varianza		,300	
	Desviación estándar		,54772	
	Mínimo		4,00	
	Máximo		5,00	
	Rango		1,00	
	Rango intercuartil		1,00	
	Asimetría		,609	,913
	Curtosis		-3,333	2,000

a. Conocimientos_Despues_Grupo_B es constante. Se ha omitido.

Figura 4-3: Pantalla de los resultados descriptivos de los conocimientos después de probar el juego

Realizado por: Christian Viracocha, Año 2018

Con los resultados anteriores se obtiene el valor base para decidir cuál hipótesis es válida, como se muestra en la Figura 5-3 en donde se visualiza toda la información para la prueba del T-Student con muestras relacionadas.

Estadísticas de muestras emparejadas

		Media	N	Desviación estándar	Media de error estándar
Par 1	Conocimientos_Despues_Grupo_A	4,4000	5	,54772	,24495
	Conocimientos_Despues_Grupo_B	5,0000	5	,00000	,00000

Prueba de muestras emparejadas

		Diferencias emparejadas					t	gl	Sig. (bilateral)
		Media	Desviación estándar	Media de error estándar	95% de intervalo de confianza de la diferencia Inferior	Superior			
Par 1	Conocimientos_Despues_Grupo_A - Conocimientos_Despues_Grupo_B	-,60000	,54772	,24495	-1,28009	,08009	-2,449	4	,070

Figura 5-3: Pantalla de resultados para la prueba de las muestras emparejados de los conocimientos después de probar el juego

Realizado por: Christian Viracocha, Año 2018

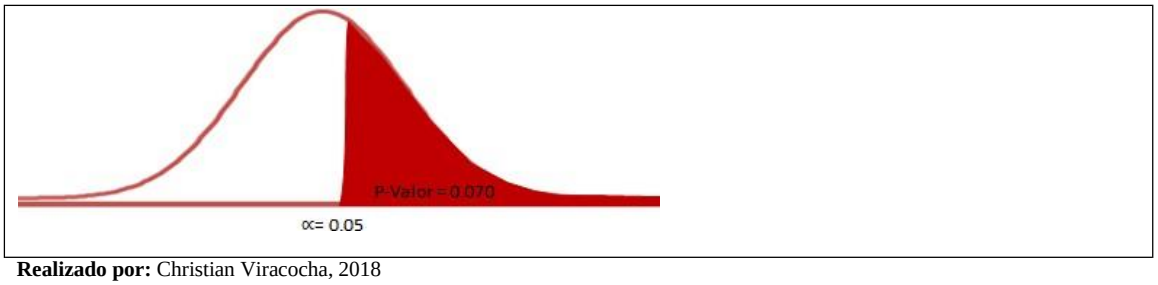
Criterios para decidir la hipótesis:

Para la toma de la hipótesis correcta en el proceso de validación del después, de ser probado el juego por los dos grupos de estudiantes, se utilizan los siguientes criterios, los cuales son analizados en la Tabla 8-3.

- Si la probabilidad obtenida es P-valor <= ∝ se rechaza la Ho.
- Si la probabilidad obtenida es P-valor > ∝ no se rechaza la Ho

Tabla 8-3: Tabla de la decisión estadística final del después de probar el juego

Decisión Estadística		
P-Valor = 0.070	>	∝= 0.05
Conclusión: Se acepta la hipótesis nula por lo que no existen grandes diferencias de conocimientos entre los estudiantes de los dos grupos que probaron el juego ¨Unity City¨. Por lo que la diferencia de conocimientos que existen entre los dos son de 0.6 tal como se muestra en la Figura 5-3.		
Gráfica		



Realizado por: Christian Viracocha, 2018

Hipótesis del Nivel de Aprendizaje al Usar el Juego de ¨Unity City¨

Hipótesis Nula

Ho: El nivel de aprendizaje de los estudiantes no aumentó después de haber jugado el Serious Game Unity City de acuerdo con los puntajes obtenidos mediante el test, que se realizó al finalizar el proceso de pruebas.

Hipótesis de Investigación

H1: El nivel de aprendizaje de los estudiantes aumentó después de haber jugado el Serious Game Unity City de acuerdo con los puntajes obtenidos mediante el test, que se realizó al finalizar el proceso de pruebas.

∝=**Porcentaje de error=** 0.05

Para el cálculo de la hipótesis final se comparó todos los resultados obtenidos de la sección 1 y 4 del test para obtener los datos descriptivos con el programa de IBM como se indica en la Figura 6-3 para el cálculo de la prueba de T-Student de las muestras relacionadas.

Descriptivos

			Estadístico	Error estándar
Conocimientos_Antes	Media		2,7000	,47258
	95% de intervalo de confianza para la media	Límite inferior	1,6309	
		Límite superior	3,7691	
	Media recortada al 5%		2,6667	
	Mediana		2,5000	
	Varianza		2,233	
	Desviación estándar		1,49443	
	Mínimo		1,00	
	Máximo		5,00	
	Rango		4,00	
	Rango intercuartil		3,00	
	Asimetría		,140	,687
	Curtosis		-1,622	1,334
Conocimientos_Despues	Media		4,7000	,15275
	95% de intervalo de confianza para la media	Límite inferior	4,3544	
		Límite superior	5,0456	
	Media recortada al 5%		4,7222	
	Mediana		5,0000	
	Varianza		,233	
	Desviación estándar		,48305	
	Mínimo		4,00	
	Máximo		5,00	
	Rango		1,00	
	Rango intercuartil		1,00	
	Asimetría		-1,035	,687
	Curtosis		-1,224	1,334

Figura 6-3: Pantalla de los resultados descriptivos de los conocimientos adquiridos por el juego

Realizado por: Christian Viracocha, Año 2018

Con los resultados anteriores se obtiene el valor base para decidir cuál hipótesis es válida, como se muestra en la Figura 7-3 en donde se visualiza la información para la prueba del T-Student con muestras relacionadas.

Estadísticas de muestras emparejadas

		Media	N	Desviación estándar	Media de error estándar
Par 1	Conocimientos_Antes	2,7000	10	1,49443	,47258
	Conocimientos_Despues	4,7000	10	,48305	,15275

Prueba de muestras emparejadas

		Diferencias emparejadas					t	gl	Sig. (bilateral)
		Media	Desviación estándar	Media de error estándar	95% de intervalo de confianza de la diferencia Inferior	Superior			
Par 1	Conocimientos_Antes - Conocimientos_Despues	-2,00000	1,15470	,36515	-2,82602	-1,17398	-5,477	9	,000

Figura 7-3: Pantalla de resultados para la prueba de las muestras emparejados de los conocimientos adquiridos al probar el juego

Realizado por: Christian Viracocha, Año 2018

Criterios para decidir la Hipótesis Final:

Para la toma de decisión de la hipótesis final en el proceso de validación del proyecto del antes y después de ser probado el juego, se utilizaron los siguientes criterios, los cuales son analizados en la Tabla 9-3 en donde se pone en contexto la hipótesis final.

- Si la probabilidad obtenida es P-valor <= $\propto$ se rechaza la Ho.
- Si la probabilidad obtenida es P-valor > $\propto$ no se rechaza la Ho

Tabla 9-3: Tabla de la decisión estadística final al probar el juego

Decisión Estadística		
P-Valor = 0.000	<	$\propto$= 0.05
Conclusión: Se acepta la Hipótesis de Investigación por lo que el nivel de aprendizaje en los estudiantes aumentó, después de haber jugado el Serious Game Unity City de acuerdo con los puntajes obtenidos mediante el test, que se realizó al finalizar el proceso de pruebas. Por lo que la diferencia de conocimientos que existen entre los dos grupos son de 2 tal como se muestra en la Figura 7-3.		
Gráfica		

Realizado por: Christian Viracocha, 2018

El resultado final del análisis estadístico del antes y después de usar el Serious Game dio como resultado el aumento de un 33.33% respecto a los conocimientos que tienen los estudiantes sobre el manejo del motor de videojuegos Unity 3D, sabiendo que el resultado máximo de las pruebas fue de 6 y la diferencia estadística de 2.

El resultado final del proyecto fue la elaboración de un Serious Game con la implementación del motor de videojuegos Unity 3D, destinado a la enseñanza de los conocimientos básicos para el uso y diseño de videojuegos en Unity 3D. Donde se incorporó la inteligencia artificial para el diseño del ambiente del juego, así como también algoritmos que manejaron los resultados del juego el cual permitió visualizar mensajes que ayudaron a la jugabilidad del mismo.

Al ser expuesto al uso del Serious Game por un grupo de estudiantes, se pudo comprobar que el juego ayuda a incrementar los conocimientos del jugador. Además con las encuestas que se realizaron, nos permitieron ver resultados favorables donde el jugador se siente conforme y no tiene dificultad al usarlo, por lo que los jugadores desearían tener una nueva versión del juego.

CONCLUSIONES

- La utilización de los Serious Games permite mejorar el nivel de aprendizaje en los jóvenes y niños sin distinción de edad, debido a que los participantes se divierten y aprenden jugando. Con estos juegos se pueden impartir nuevos y diversos conocimientos, que van desde como sumar hasta como manejar un avión.
- Con el uso de la Metodología SUM lo que permite es crear juegos a corto tiempo, debido a que piensa más en la calidad del tiempo y costos durante el desarrollo del proyecto. Gracias a la división de las fases, se enfocan primero en abstraer información básica del juego para luego centrarse en su desarrollo mediante iteraciones, con el fin de obtener un producto que se use en las pruebas y corrección de errores, para crear el producto final.
- El uso de las técnicas referentes a la Inteligencia Artificial dentro de los juegos como son las máquinas de estados o el Path Planning para los videojuegos, es una buena forma de incursionar en la inteligencia *artificial* dentro de los Serious Game que conforman la base

para la creación de un entorno más real para los jugadores, en donde podemos otorgarles el don de decisión a los personajes no jugables, aunque ésta sea limitada.

- El buen manejo de la inteligencia artificial dentro de los Serious Games puede contribuir al aprendizaje de los jugadores, debido a que se puede realizar un seguimiento de los resultados y decidir cuál es el mejor camino que puede seguir el jugador, para que el nivel de aprendizaje se eleve y el resultado sea el objetivo final del juego, que se basa en el adiestramiento y mejora de las habilidades del jugador en diferentes campos de la sociedad.
- En estos tiempos el desarrollo de los Serious Games va en aumento, gracias a los diferentes estudios y proyectos realizados por las universidades y empresas, debido a que estamos en una nueva era, donde se busca implementar el deseo de aprender de forma indirecta a través de juegos educativos que sean divertidos para los estudiantes o participantes del juego.
- El motor de videojuegos UNITY 3D se ha comprobado que es una herramienta útil gracias a su amplia gama de opciones para la creación de mundos virtuales de aprendizaje. Debido a su interfaz intuitiva nos permite crear un juego simple como sumar dos números hasta uno complejo como es la simulación completa de una ciudad con independencia propia.
- El Serious Game ¨Unity City¨ es un juego fácil de manejar por su interfaz intuitiva y porque sus controles son estándares en el mundo de los juegos, con el fin de que cualquier persona lo domine de forma inmediata. Además, los objetivos dentro del juego son fáciles de cumplir y muy valiosos dado que poseen información relevante para la enseñanza en la utilización de Unity 3D.
- El análisis estadístico del antes y después de usar el Serious Game dio como resultado el aumento de un 33.33% en los conocimientos que tienen los estudiantes sobre el manejo del motor de videojuegos Unity 3D, lo que significa que los Serious Games si favorecen en mejorar y aumentar el nivel de aprendizaje en los participantes.

RECOMENDACIONES

- El Serious Game ¨Unity City¨ se puede ampliar a nuevas versiones que pueden incluir la ejecución en red, en donde se pueden incrementar diferentes áreas de aprendizaje y a la vez interactuar con otros jugadores.
- Para la creación de nuevos proyectos basados en Serious Games se recomienda usar tecnologías emergentes como la realidad virtual o la realidad aumentada, con la finalidad de crear un ambiente más cercano al jugador como es la proyección de objetos digitales en el mundo físico o vivir una simulación de una explosión de una supernova en la comodidad de nuestra casa.

- También se puede proyectar esta aplicación como un proyecto piloto para que se pueda implementar en otras asignaturas de la carrera de Ingeniería en Sistemas, tales como, en Inteligencia Artificial, Proyectos integradores, SIG, entre otros.
- Se recomienda usar modelos matemáticos para el uso de la inteligencia artificial ya que al usar formulismos matemáticos permite estudiar, simplificar y a la vez pronosticar comportamientos de sistemas complejos.
- Se aconseja a las personas que vayan a usar el motor de videojuegos Unity 3D, realizar varios cursos referentes al manejo de ésta, debido a su gran cantidad de herramientas que posee, a causa de que algunas de ellas no se encuentran a simple vista o no se sabe cómo utilizarlas, por lo que se pierde la oportunidad de obtener un juego de alta calidad.

BIBLIOGRAFÍA

Abadía, I. "*Juegos serios para televisión digital interactiva: revisión de literatura y definiciones.*", 2015, Revista S&T, 10(22*), Memorias: 5to Encuentro Internacional de Investigación en Diseño*, pp. 9.

Alcalá, J. *Inteligencia Artificial en Videojuegos.* [En línea]. 2015. [Consulta: 11 diciembre 2017]. Disponible en: http://www.flasentertainment.com/blog/ia.pdf.

Alvarado, E. & Campuzano, G. *Representación específica de la mitología ecuatoriana descrita en una obra histórica, a través del demo jugable de un videojuego de estrategia.* [En línea]. (Tesis).(Ingeniería) Universidad Central del Ecuador. 2015. [Consulta: 2 julio 2018]. Disponible en: http://www.dspace.uce.edu.ec/handle/25000/4306

Álvarez, R., *Introducción a las Máquinas de Estado Finito*. [en línea], [Consulta: 8 junio 2018]. Disponible en: http://www.tecbolivia.com/?option=com_content&view=article&id=13&fontstyle=f-larger.

Arias, J. & Nieto, G. *Programación del videojuego Llumpak para dispositivo móvil de sistema operativo Android.* [En línea]. (Tesis).(Ingeniería) Universidad Central del Ecuador. 2016. [Consulta: 2 julio 2018]. Disponible en: http://www.dspace.uce.edu.ec/handle/25000/6554

Armas, R. *Lúdica final.* [En línea]. 2015. [Consulta: 8 enero 2017]. Disponible en: http://dspace.ups.edu.ec/bitstream/123456789/2433/1/Ludica%20final.pdf.

Castro, P. & Nieto, B. *Inteligencia artificial del videojuego La Dama*. [En línea]. (Tesis).(Ingeniería) Universidad Central del Ecuador. 2015. Consulta: 12 diciembre 2017]. Disponible en: http://www.dspace.uce.edu.ec/handle/25000/6611

Derryberry, A. *Serious games: online games for learning.* [En línea]. 2015. [Consulta: 11 diciembre 2017]. Disponible en: https://iktmangud.files.wordpress.com/2014/09/online-games-for-learning.pdf

Esteban, M. *El diseño de entornos de aprendizaje constructivista.* [En línea]. 2000. [Consulta: 11 enero 2018]. Disponible en: http://www.um.es/ead/red/6/documento6.pdf

Fenrir, A. *Cinco cosas buenas y cinco no tan buenas de Unity 3D.* [En línea]. 2016. [Consulta: 13 enero 2017]. Disponible en: https://laleyendadedarwan.es/2016/06/25/cinco-cosas-buenas-y-cinco-no-tan-buenas-de-unity-3d/.

GAMELEARN. *La teoría del game-based learning. Gamelearn: Game-based learning courses for soft skills training.* [En línea]. 2015. [Consulta: 6 enero 2017]. Disponible en: https://laleyendadedarwan.es/2016/06/25/cinco-cosas-buenas-y-cinco-no-tan-buenas-de-unity-3d/.

GEMSERK, G. *SUM para Desarrollo de Videojuegos.* [En línea]. 2016. [Consulta: 12 diciembre 2017]. Disponible en: http://www.gemserk.com/sum/.

Mallo, J.S. y Ruiz, C.M. *Inteligencia Artificial en los Videojuegos.* [En línea]. 2016. [Consulta: 10 febrero 2018]. Disponible en: http://www.academia.edu/22310357/Inteligencia_Artificial_en_los_Videojuegos

Martinez, C. *Inteligencia Artificial en los Videojuegos.* [En línea]. 2016. [Consulta: 5 febrero 2018]. Disponible en: http://www.it.uc3m.es/jvillena/irc/practicas/13-14/03.pdf.

Martinez, F. *Presente y Futuro de la Tecnología de la Realidad Virtual. Creatividad, TICs y sociedad de la información.* [En línea]. 2011. [Consulta: 15 febrero 2018]. Disponible en: http://creatividadysociedad.com/articulos/16/4-Realidad%20Virtual.pdf

Medway, F. J. *Tutoring as a teaching method*. 2, Oxford-N. York, Pergamon Press: The International Encyclopedia of Education. , *1485*, pp 5315.

Ménendez, A. *Mundos virtuales y educación. Mundos virtuales y educación.* [En línea]. 2017. [Consulta: 22 diciembre 2017]. Disponible en: http://revistas.lasalle.edu.co/index.php/ls/article/viewFile/2385/2130.

Ordóñez, A., Ibeth, G., Villegas, O. & Balmes, H. *El acompañamiento tutorial como estrategia de la formación personal y profesional: un estudio basado en la experiencia en una institución de educación superior. vol* 4, no. 1, pp. 31-42.

Padilla, C. & López, F., *Videojuegos y comunicación: estudio de la relación comunicacional del jugador con el mundo del videojuego, en el sistema de realidad virtual Kinect.* [En línea].

(Tesis).(Licenciatura) Universidad Central del Ecuador. 2014. [Consulta: 2 julio 2018]. Disponible en: http://www.dspace.uce.edu.ec/handle/25000/3198

Perez, F.X., *Presente y Futuro de la Tecnología de la Realidad Virtual. Presente y Futuro de la Tecnología de la Realidad Virtual.* [En línea]. 2016. [Consulta:11 diciembre 2017]. Disponible en: http://www.creatividadysociedad.com/articulos/16/4-Realidad%20Virtual.pdf.

Pretelín, A., *Serious Game orientado al aprendizaje de la física.* [En línea]. (Tesis).(Licenciatura) Instituto Politécnico Nacional. 2010. [Consulta: 2 febrero 2018]. Disponible en: http://www.cicata.ipn.mx/OfertaEducativa/MFE/Estudiantes/Documents/Angel_Pretelin_2010_MCFE.pdf.

Romero, M. y Turpo, O., " *Serious Games para el desarrollo de las competencias del siglo XXI. RED.*", 2014, Revista de Educación a Distancia, pp. 1–22.

Sánchez, M., *Buenas Prácticas en la Creación de Serious Games (Objetos de Aprendizaje Reutilizables) [*En línea]. 2016. [Consulta: 5 enero 2017]. Disponible en: http://ceur-ws.org/Vol-318/Sanchez.pdf.

UNITY TECHNOLOGIES, *Products - Unity*. [En línea]. 2018. [Consulta: 8 junio 2018]. Disponible en: https://unity3d.com/es/unity.

Urretavizcaya, M., *" Sistemas Inteligentes en el ambito de la Educacion. INTELIGENCIA ARTIFICIAL ",* vol. 5, no. 12. [Consulta: 9 junio 2018]. ISSN 1988-3064, 1137-3601. DOI 10.4114/ia.v5i12.703. Disponible en: http://journal.iberamia.org/index.php/ia/article/view/292.

ANEXOS

Anexo A. Gestión de Riesgos

GESTIÓN DE RIESGOS

Para la gestión de riesgos realiza el siguiente análisis para determinar los factores de riesgo que potencialmente tendrían un mayor efecto sobre el desarrollo del Serious Game, por lo que serán gestionados antes, durante y después de desarrollo del juego

Los valores de probabilidad e impacto de cada uno de los riesgos indicaran que tan probable es que ocurra ese riesgo, por lo que se cuantifica las probabilidades de riesgo de acuerdo con los siguientes criterios:

Tabla 1: Rango de Probabilidades

RANGO DE PROBABILIDADES	DESCRIPCIÓN	VALOR
1% - 33%	BAJA	1
34% – 67%	MEDIA	2
68% -99%	ALTA	3

Realizado por: Christian Viracocha, 2018

El impacto del riesgo es valorado en función de aspectos que retrasan la entrega del producto e impacto técnico de acuerdo con los siguientes parámetros:

Tabla 2: Impacto del Riesgo

IMPACTO	RETRASO	IMPACTO TÉCNICO	VALOR
BAJO	1 semana	Ligero	1
MODERADO	2 semanas	Moderado	2
ALTO	1 mes	Severo	3
CRÍTICO	Más de un mes	Critico	4

Realizado por: Christian Viracocha, 2018

PRIORIZACIÓN DE RIESGOS

Para priorizar los riesgos del proyecto se analizo la exposición de cada uno de ellos, por lo que nos permitirá cuantificar por prioridad cada riesgo, con el fin de saber cuál es el riesgo más latente dentro del desarrollo del proyecto y así realizar un plan de contingencia sobre el riesgo, para lo cual se realizó la Tabla 3 en donde se observan los posibles riesgos y cual es la prioridad de cada uno, teniendo en cuenta que la prioridad 1 es la más alta y la 5 es la más baja.

Tabla 1: Priorización de Riesgos

ID	DESCRIPCIÓN	PROBABILIDAD			IMPACTO		PRIORIDAD
		PORCENTAJE	PROBABILIDAD	VALOR	IMPACTO	VALOR	

R01	La falta de un consulto para el manejo de Unity	70%	Alta	3	Alto	3	1
R02	Mala planificación de las actividades en el desarrollo del proyecto	60%	Media	2	Alto	3	3
R03	Perdida de información	20%	Bajo	1	Critico	4	3
R04	Cambio total o parcial de la Temática del Juego.	20%	Bajo	1	Alto	3	1
R05	Robo de los equipos.	20%	Bajo	1	Critico	4	3
R06	Daño del equipo de trabajo debido al uso o falta de mantenimiento.	20%	Bajo	1	Critico	4	3
R07	Ausencia de miembros del Equipo.	80%	Alta	3	Critico	4	1

GESTIÓN DE RIESGOS

La gestión de riesgos nos sirve para poder prevenir que un riesgo ocurra, en el caso de que ocurra nos ayudará a saber cómo gestionar ese riesgo.

Para el desarrollo de la gestión se utilizará la Tabla 4 la cual se encuentra estructurada de la siguiente manera:

- En los campos de Probabilidad, Impacto y Prioridad se ingresan los datos que se obtuvieron como resultado en la Tabla 3.
- **DESCRIPCIÓN:** Se realiza una breve descripción de riesgo a gestionar.
- **REFINAMIENTO:** Se subdivide este campo en Causas y Consecuencias
    - Causas: son los motivos por los que el riesgo se puede dar.
    - Consecuencias: son los resultados una vez que ya haya ocurrido del riesgo.
- **REDUCCIÓN:** Actividades que se pueden ejecutar para poder prevenir que ese riesgo ocurra.
- **SUPERVISIÓN:** Actividades que se deben ejecutar continuamente para poder prevenir que ese riesgo ocurra.
- **GESTIÓN:** Las acciones que se tomaran para poder resolver las consecuencias que trajo ese riesgo.

Cada riesgo tiene una hoja de riesgo que las puede observar a continuación:

Tabla 2 Hoja de Gestión de Riesgo

HOJA DE GESTIÓN DEL RIESGO		
ID. DEL RIESGO: R_01		**FECHA**: 02/05/2018
Probabilidad: Alta **Valor:** 3	**Impacto:** Alto **Valor:** 3	**Prioridad:** 1

DESCRIPCIÓN: La falta de un consultor en el manejo de Unity, puede acarrear tiempos mas largos, en encontrar soluciones a problemas de desarrollo.

REFINAMIENTO

Causas

- Falta de comunicación con el consultor.
- Arrogancia por parte del equipo desarrollador.

Consecuencias

- Alargamiento en tiempos de desarrollo.
- El juego poseerá errores internos.

REDUCCIÓN

- Tener un contrato con un consultor que pueda ayudar las 24 horas del día.
- Encontrar las carencias en el equipo de desarrollo con respecto a la programación en Unity.

SUPERVISIÓN

- Mantenerse al tanto del progreso del proyecto.
- Verificar los posibles bugs que pueden presentarse en el juego.

GESTIÓN

- Contratar a un consultor y mantenerse informado del progreso del proyecto.

HOJA DE GESTIÓN DEL RIESGO

ID. DEL RIESGO: R_02		**FECHA**: 02/05/2018
Probabilidad: Media **Valor:** 2	**Impacto:** Alto **Valor:** 3	**Prioridad:** 3

DESCRIPCIÓN: Mala planificación de las actividades en el desarrollo del proyecto por parte del equipo desarrollador.

REFINAMIENTO

Causas

- Falta de comunicación con el cliente al plantear la solución a su problema.
- Falta de experiencia al planificar proyectos de juego.

Consecuencias

- Incumplimiento en los factores de costos, fechas y esfuerzo previstos en el desarrollo del proyecto.
- Incumplimientos de los requerimientos del cliente.
- Inconformidad por parte del cliente.

REDUCCIÓN

- Capacitar al equipo de desarrollo
- Mantener un control en desarrollo del proyecto

SUPERVISIÓN

- Mantener al tanto al cliente en el desarrollo del proyecto mediante reuniones.
- Controlar el proceso del desarrollo de la aplicación.

GESTIÓN

- Replanificar el proyecto con el cliente.

HOJA DE GESTIÓN DEL RIESGO

ID. DEL RIESGO: R_03		**FECHA**: 02/05/2018
Probabilidad: Bajo **Valor:** 1	**Impacto:** Critico **Valor:** 4	**Prioridad:** 3

DESCRIPCIÓN: Perdida de información del proyecto por causas internas del software

REFINAMIENTO

Causas

- No poseer un antivirus.
- No haber realizado backups del proyecto.

Consecuencias

- Daño de la información del proyecto por causas de virus.
- Perdida de la confianza por parte del cliente.
- Daños en la planificación del proyecto.

REDUCCIÓN

- Crear un sistema para la creación de los Backups.
- Mantener siempre actualizado él antivirus.

SUPERVISIÓN

- Verificar las últimas actualizaciones de los Backups del proyecto.
- Mantener informes de los Backups, para identificar su porcentaje de progreso.

GESTIÓN

- Comprar antivirus de última generación.
- Mantener los Backups en la nube o discos externos

HOJA DE GESTIÓN DEL RIESGO

ID. DEL RIESGO: R_04		**FECHA**: 02/05/2018
Probabilidad: Bajo **Valor:** 1	**Impacto:** Alto **Valor:** 3	**Prioridad:** 1

DESCRIPCIÓN: El cliente decide de improviso cambiar la temática del juego.

REFINAMIENTO

Causas

- Falta de comunicación con él cliente.
- No haber identificado cual era el deseo del cliente al plantear su proyecto.

Consecuencias

- Retraso en la ejecución del proyecto debido a cambios en la planificación.
- El juego final no cumplirá con todas las funcionalidades requeridas por el cliente.
- Desacuerdos con el cliente.

REDUCCIÓN

- Realizar un contrato identificando el resultado final del proyecto.
- Mantener constantes reuniones con el cliente, para mantenerlo en conocimiento del progreso del proyecto.

SUPERVISIÓN

- Mantenerse al tanto de los deseaos y necesidades del cliente.
- Verificar el progreso del proyecto.

GESTIÓN

- Tratar de llegar a un acuerdo con el cliente o redefinir las condiciones del contrato.

HOJA DE GESTIÓN DEL RIESGO

ID. DEL RIESGO: R_05		**FECHA**: 02/05/2018
Probabilidad: Bajo **Valor:** 1	**Impacto:** Critico **Valor:** 4	**Prioridad:** 3

DESCRIPCIÓN: Los equipos del equipo de desarrollo fueron robados

REFINAMIENTO

Causas

- Falta de seguridad por parte de la empresa.

- Descuido por parte de los desarrolladores.

Consecuencias

- Retraso en la ejecución del proyecto.
- Disminución en la confianza por parte del cliente.

REDUCCIÓN

- Obtener un buen sistema de seguridad para la empresa
- Poseer backups en la nube, con el fin de evitar perdida de información.

SUPERVISIÓN

- Mantenerse informado de todo lo que ocurre fuera y dentro de la empresa.
- Verificar que se realicen continuos respaldos del proyecto.

GESTIÓN

- Cobrar el seguro de las maquinas robadas.

HOJA DE GESTIÓN DEL RIESGO

ID. DEL RIESGO: R_06		**FECHA**: 02/05/2018
Probabilidad: Bajo **Valor:** 1	**Impacto:** Critico **Valor:** 4	**Prioridad:** 3

DESCRIPCIÓN: Los equipos del área de desarrollo dejaron de funcionar por mal uso de ellos y por falta de mantenimiento.

REFINAMIENTO

Causas

- Falta de mantenimiento del equipo.
- Carencia en el buen uso de los equipos.

Consecuencias

- Daño parcial o total del Equipo.
- Perdida de información.

REDUCCIÓN

- Realizar mantenimiento continuo a los equipos.
- Evitar factores externos que puedan dañar a los equipos como líquidos.

SUPERVISIÓN

- Monitorear los equipos del área de desarrolló con respecto a su funcionamiento.
- Verificar los respaldos de la información del proyecto.

GESTIÓN

- Cobrar el seguro de las máquinas y recuperar la información de los respaldos del proyecto.

HOJA DE GESTIÓN DEL RIESGO

ID. DEL RIESGO: R_07		**FECHA**: 02/05/201
Probabilidad: Alta **Valor:** 3	**Impacto:** Critico **Valor:** 4	**Prioridad:** 1

DESCRIPCIÓN: Parte del equipo de desarrollo se ausentan continuamente durante el desarrollo del proyecto.

REFINAMIENTO

Causas

- Falta de comunicación.
- Falta de responsabilidad.

Consecuencias

- Retraso en la ejecución del proyecto.
- Desacuerdos con el equipo de desarrollo.

- Incumplimiento del contrato.
REDUCCIÓN - Realizar convivencias con el equipo de desarrollo. - Otorgar incentivos por el trabajo bien hecho.
SUPERVISIÓN - Mantenerse al tanto de la situación proyecto. - Verificar que todos los miembros del equipo estén al tanto de lo que realizan sus compañeros.
GESTIÓN - Tratar de llegar a un acuerdo con los miembros del equipo. - Contratar a un programador extra en el proyecto.

Se han encontrado 7 riesgos más importantes para el desarrollo del proyecto, los cuales fueron gestionados en las Hojas de Gestión de Riesgos. Terminando el Análisis de Gestión de Riesgos los responsables del proyecto podrán tomar las acciones necesarias para prevenir y gestionar cualquiera de los riesgos que se presentaron anteriormente.

Los riesgos que han sucedido hasta el momento son:

- La falta de un consultor de Unity 3D. Como acción a la aparición de este riesgo se busco contratar la ayuda de un experto para identificar y solucionar los problemas que van apareciendo durante el proyecto
- Daño del equipo de trabajo debido al uso o falta de mantenimiento. Como acción a la aparición de este riesgo se realizó un respaldo de toda la información y una limpieza completa del hardware utilizado.

Anexo B. Formato de la Encuesta

Modelo de la Encuesta

El modelo que se utiliza para la toma de resultados en la prueba del Serious Game se divide en 4 secciones, que se llenaran en un orden secuencial, de acuerdo como se vaya realizando la prueba.

- La primera sección se realizó antes de probar el juego, con la finalidad de conocer cuáles son los conocimientos básicos que poseen los estudiantes antes de jugar.
- La segunda y tercera sección se realizó después de haber jugado el juego con el fin de medir la usabilidad y funcionalidad del Serious Game teniendo en cuenta los criterios de cada uno de los estudiantes que usaron el juego.
- La cuarta sección se realiza al finalizar la prueba, ya que tiene la finalidad de medir los nuevos conocimientos adquiridos por los estudiantes después de haber jugado el juego.

Los datos que se obtuvieron de la prueba del Serious Game, gracias a la encuesta, nos sirven para comparar los resultados del antes y después del uso del juego, y así conocer si existe algún aumento en el conocimiento de los estudiantes, al mismo tiempo obtener diversos criterios y opiniones en función a la usabilidad y funcionalidad del juego por parte de los estudiantes.

A continuación, se muestra la encuesta que se uso en la prueba del Serious Game.

Test de Usabilidad y Funcionalidad del Serious Games "Unity City"

Objetivo

El objetivo de este test es medir la usabilidad y funcionalidad del jugador al momento de manejar el Serious Games que fue diseñado por ¨Christian Viracocha ¨.

Datos Personales

Nombre: …….

Edad: ……

Carrera: …….

Fecha: ……

Notas

- El test se divide en 4 sectores, que deben llenarse de forma secuencial, por lo que la sección 1 se llenara antes de jugar y las secciones siguientes se las llenara al finalizar de jugar.

Le agradezco por adelantado su disposición de participar en Test de Usabilidad, que nos ayudara en la detección de problemas en juego, si es que los tuviera, por lo que vamos a comenzar con preguntas simples.

Sección 1

Esta sección es un cuestionario que mide el nivel de conocimientos de Unity 3D.

¿Ha utilizado alguna vez Unity 3D?

Si No

¿A creado juegos para computadora?

Si No

¿Qué es Unity 3D?

……..

¿Sabe cómo crear un Terreno, con todos sus elementos en el área de Trabajo de Unity?

Si No

¿Sabe que es un Script?

Si No

¿Reconoce los elementos de Luz que se pueden implementar dentro del proyecto de Unity?

Si No

¿Sabe cómo implementar los personajes jugables dentro del proyecto en Unity?

Si No

Sección 2

En esta sección se calificará en una escala en la que el menor puntaje es 0 y el mayor es 10, marque con una X su respuesta.

Preguntas	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
¿El juego es divertido?										
¿El juego provee de información valiosa?										
¿Qué tan facil fue adaptarse al controlar el juego?										
¿Qué tan difícil fue cumplir con los objetivos del juego?										
¿Lo jugarías de nuevo?										
¿La interfaz del juego es aceptable?										

Sección 3

En esta sección se calificara en una escala: Totalmente de Acuerdo, Muy de Acuerdo, Neutral, Muy Desacuerdo, Totalmente en desacuerdo, su respuesta será marcada con una X.

Preguntas	**Totalmente de Acuerdo**	**Muy de Acuerdo**	**Neutral**	**Muy Desacuerdo**	**Totalmente en desacuerdo**
El juego es difícil de jugar (Efectividad)					
El juego es agradable con el teclado (Eficiencia)					
Se sintió dentro del juego al momento de usarlo (Inmersión)					
Deseaba volver a visualizar la información del juego (Motivación)					

No tuvo problemas al cumplir con los objetivos (Aprendizaje)					
Quiero seguir jugando para aprender más (Emoción)					
Esperaría una nueva versión					

Sección 4

Esta sección es un cuestionario que mide el nivel de conocimientos de Unity 3D después de haber usado el Serious Game.

¿Qué es Unity 3D?

……..

¿Sabe cómo crear un Terreno, con todos sus elementos en el área de Trabajo de Unity?

Si No

¿Sabe que es un Script?

Si No

¿Reconoce los elementos de Luz que se pueden implementar dentro del proyecto de Unity?

Si No

¿Sabe cómo implementar los personajes jugables dentro del proyecto en Unity?

Si No

¿Usted recomendaría de este tutorial para la enseñanza de otra herramienta informática?

Si No

……

Firma

Para la toma de resultados en el método de la observación se utilizo una ficha de observación, en donde los resultados que se van obteniendo son mediante la visualización de los estudiantes al momento de usar el Serious Game, en donde se ve como ellos manipulan y se adaptan a la lógica

del juego, como también el tiempo y gusto de los participantes al jugar, con el objetivo de obtener nuevo conocimientos a partir del juego.

A continuación, se muestra la ficha de observación que se usó, para la toma de datos:

Ficha para evaluación por Observación del Serious Games "Unity City"

Objetivo

El objetivo de este de esta ficha es recopilar la información del proceso de prueba del Serious Game ¨Unity City¨

Datos Personales

Nombre: ……..……

Fecha: ……..……

Institución: ……

Docente a Cargo: ……..……

Cantidad de Alumnos: …

Referencias:

1	**2**	**3**	**4**	**5**
Nada	Poco	Aceptable	Casi Siempre	Siempre

Ficha

	1	2	3	4	5
Manejan el juego de forma fluida los estudiantes					
Transmiten entusiasmo al jugar					
Siguen la secuencia lógica del juego					
Saben cómo cumplir con los objetivos					
Encuentran soluciones a las preguntas del Test					
Identifican a los Personaje no-jugables especiales en el juego					
Buscan cumplir los objetivos que se les presentan en el juego					
Se les dificulta aprender el manejo de los controles del juego					
Les agrada la interfaz del juego					
Han adquirido nuevos conocimientos al jugar					

Anexo C. Manual de Desarrollo

MANUAL DE DESARROLLO DEL SERIOUS GAME ¨UNITY CITY¨

Autor: Christian Viracocha

Contenido

1. Introducción

El presente manual proporciona las principales características y los procesos necesarios para la creación del Serious Game ¨Unity City¨, como por ejemplo la creación del ambiente del juego y la configuración necesario para los personajes no jugables, mediante el motor gráfico de Unity 3D, ScketchUp y MakeHuman entre otros programas, los cuales nos ayudarán a la creación del juego, con la finalidad de que los usuarios puedan crear cualquier tipo de juego, tomado como referencia Unity City.

2. Requerimientos

Los requisitos principales para el desarrollo del juego en el área de Hardware son:

- Una Laptop o PC de escritorio
- 16 Gb de RAM como mínimo
- 4 Gb de Tarjeta de video superior al Nvidia GTX 970
- Procesador Intel I7

Los programas necesarios para el desarrollo del juego son:

- Unity 3D
- Scketchap
- Maya
- MakeHuman
- Autodesk FBX Converter x64 2013
- Movie Maker
- Inno Setup Compiler

3. Descripción del Serious Game

El Serious Game se basa en la construcción de una ciudad virtual utilizando la arquitectura de algunos edificios de la ciudad de Riobamba, para su ambientación como también la implementación de personajes no jugables, que se encuentran dividido en dos personajes:

- Los personajes no jugables especiales: Estos poseen información relevante para la jugabilidad del jugador.
- Los personajes no jugables simples: Son aquellos personajes que contribuyen a dar vida a la ciudad, por lo que poseen una independencia propia dentro del juego.

El jugador posee un avatar en 1ra persona, el cual le ayudará a interactuar en la ciudad virtual, en donde posee un visor, el cual le muestra la información del juego, como su puntaje, el minimapa de la ciudad, la barra de progreso, los objetivos que se deben cumplir.

La interacción que tiene el jugador dentro del juego se basa en la movilidad e interacción con los personajes no jugables especiales que poseen información relevante, para cumplir con los objetivos del juego.

La finalidad del juego es la adquisición de información para la manipulación y construcción de juegos mediante el motor de videojuegos Unity 3D, de parte de los jugadores.

4. Desarrollo

El desarrollo del juego se basó en la Metodología SUM que contribuye a la creación de juegos, para grupos pequeños con el fin de mejorar la calidad y tiempos en la creación de los juegos, para mayor información de la metodología revisar el siguiente link ¨ **http://www.gemserk.com/sum/**¨ en donde se explica con mayor amplitud el manejo de la metodología.

Antes de desarrollar el Serious Game, se debe tener claros los aspectos principales para la creación de un videojuego educativo como:

- La historia del juego,
- La jugabilidad,
- El tipo de arte que se va a usar en todo el proyecto,
- La tecnología que se usará en el desarrollo del juego,
- El material educativo que se implementará y
- La metodología de desarrollo.

Estos aspectos son los más importantes que siempre se deben tomar en cuenta antes de desarrollar cualquier juego educativo.

4.1. Creación del concepto del Juego

El Serious Games ¨Unity City¨ es un juego enfocado a la enseñanza en un ambiente virtual en primera persona, en la cual se crea un ambiente inmersivo para el jugador, mediante el desplazamiento y la interacción con el mundo virtual el cual puede ser mediante una pantalla o el uso del Oculus Rift.

La jugabilidad se centra en el aprendizaje del jugador, en donde se dispondrá de videos tutoriales, como también de la teoría, las herramientas relacionadas con el proceso de cómo usar el programa, los cuales el jugador podrá encontrar en la ciudad 3D. También se puede realizar pequeños test los cuales tendrán como objetivo medir los conocimientos adquiridos de la utilización del programa Unity 3D para el desarrollo de videojuegos.

El juego se basará en un solo jugador y se reiniciará cada vez que se juegue para que el jugador tenga que realizar nuevamente las misiones, teniendo en cuenta que puede viajar por toda la

ciudad y acceder a cualquier área en el que se encuentre el material que él desea, con el fin de que el jugador pueda acceder al conocimiento, sin la necesidad de repetir el juego nuevamente.

4.2. Creación del Prototyping

El prototipo del videojuego muestra la ciudad en 3D en el cual existen NPC´s que manejan información acerca del uso de Unity, también señala la posible ruta que puede seguir el jugador para cumplir todas las misiones que se presentan en el juego, como se muestra en la Figura 1.

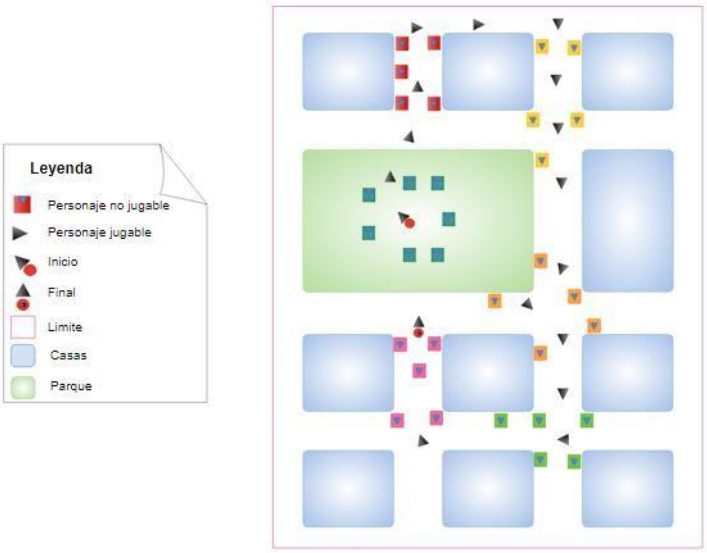


Figura 1: ProtoTyping de la ciudad virtual

Realizado por: Christian Viracocha, Año 2018

4.3. Descripción del Ambiente del juego

El Serious Games se ubica en una ciudad ficticia conformado por casas, edificios basados en la arquitectura de Riobamba y un parque central, el cual está poblado de personajes no jugables, conocidos en la jerga de juegos como los NPC´s, los cuales están divididos en dos tipos:

- El primero, son aquellos que se mueven a través de la ciudad de manera independiente con la finalidad de que la ciudad tenga vida
- El segundo, son aquellos que contienen el material educativo que se divide en: Teoría, Procedimientos, Menús, Videos y un test el cual mide el grado de aprendizaje de que obtuvo el jugador sobre el uso el programa Unity 3D.

La ciudad se divide en 6 sectores los cuales son:

- Área de aprendizaje para la creación de un Terreno.
- Área de aprendizaje para la implementación del Skybox.
- Área de aprendizaje para la implementación de los Objetos de Luz.
- Área de aprendizaje para la implementación del Personaje Jugable.

- Área de aprendizaje para la creación de Scripts.
- Área de aprendizaje de la Estructura de Unity.

Cada uno de estos contiene 5 Personajes no Jugables los cuales impartirán el curso de Unity 3D.

4.4. Desarrollo del Ambiente 3D

4.4.1. Creación de los Objetos 3D

Para crear cada uno de los Objetos en 3D se necesita imágenes de los objetos reales, para que sean diseñados en los programas de Maya, 3D Max, o SketchUp. Al momento de diseñar objetos para incorporar a los juegos, es necesario minimizar la cantidad de polígonos de cada objeto ya que esto puede aumentar, el procesamiento interno, por lo que el juego se vuelve lento.

4.4.2. Requisitos para construir los Objetos 3D

Los requisitos básicos para el diseño en 3D son:

- Obtener imágenes de los objetos en 3D.
- Tener los Colores o Texturas de los objetos originales.
- Tener instalados programas de Diseño.
- Manejar un sistema de escalas para los objetos.

Se recomienda al momento de diseñar objetos usar los programas de 3D Max o Maya ya que proveen de herramientas necesarias para minimizar polígonos de las figuras.

4.4.3. Proceso de construcción de los Objetos 3D

Para la creación de los objetos para el juego de Unity City se utilizó el programa de Sketchup por lo que es un programa básico para la creación de objetos 3D.

Se accede al programa de SketchUp y tener imágenes de la arquitectura que se va a crear, como se muestra en la Figura 2.

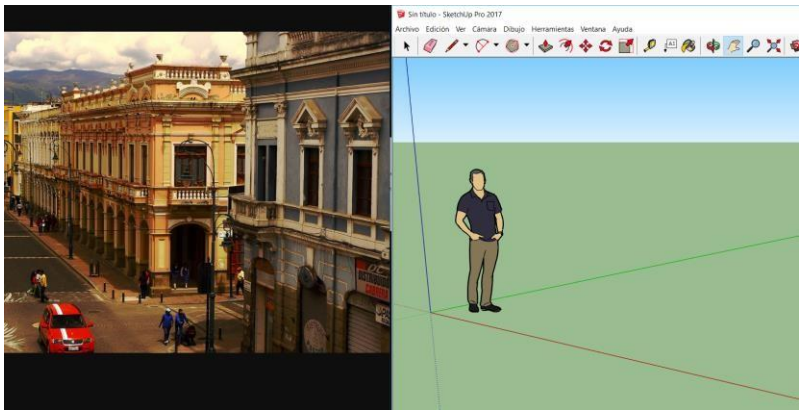


Figura 2: Pantalla del SkectchUp

Realizado por: Christian Viracocha, Año 2018

Se recomienda ver un tutorial para el manejo de SketchUp llamado ¨ **Tutorial: Aprende SketchUp en 10 minutos**¨ con el Link ¨ https://www.youtube.com/watch?v=9Nj2uDg3Yg0¨, como se muestra en la Figura 3.

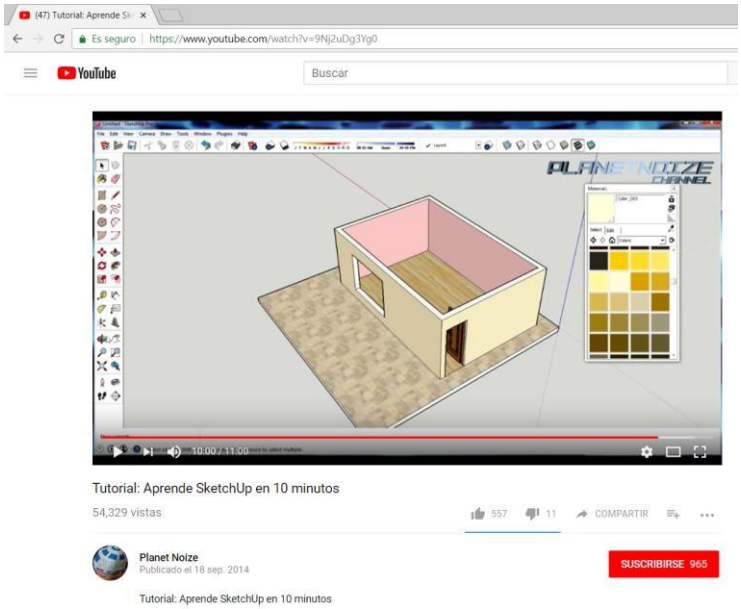


Figura 3: Pantalla del Video Tutorial de SketchUp

Realizado por: Christian Viracocha, Año 2018

Una vez terminado de crear el modelo en 3D de la ciudad, éste se verá de la siguiente forma, en donde se deberá exportar el modelo a Unity como se muestra en la Figura 4.

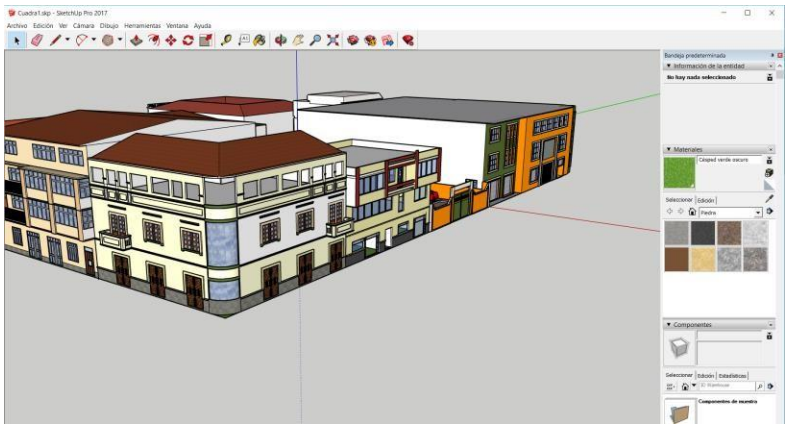


Figura 4: Pantalla del SkectchUp con un Modelo en 3D

Realizado por: Christian Viracocha, Año 2018

4.4.4. Convertir los Objetos para Unity 3D

Cuando se haya terminado de crear todos los modelos en 3D de la ciudad, estos deberán ser exportados al formato ¨Fbx¨ en el SketchUp, para que sea reconocido dentro de Unity 3D, como se ve en la Figura 5.

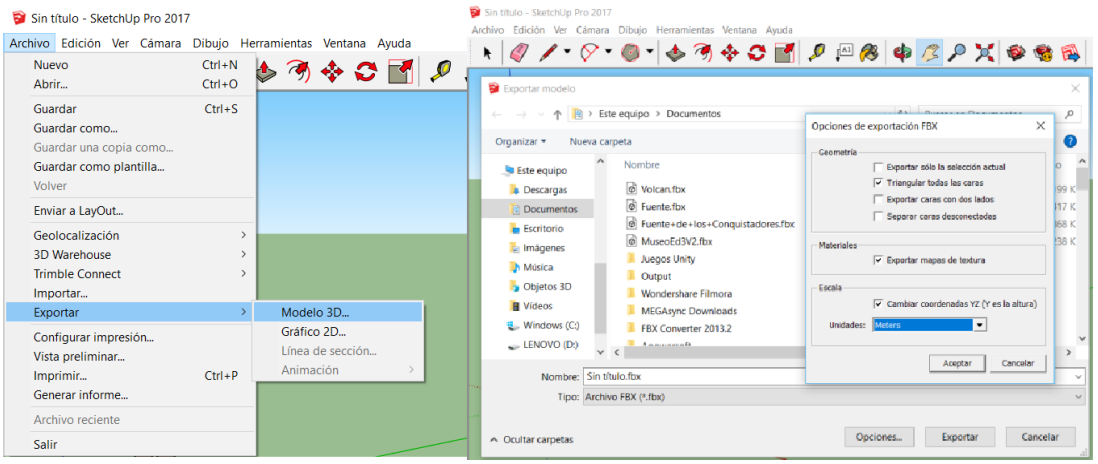


Figura 5: Pantalla del Proceso de Exportación en SketchUp

Realizado por: Christian Viracocha, Año 2018

Si los polígonos de la figura en 3D son demasiados se recomienda utilizar programas que puedan ayudar a reducirlos, como es el programa de Maya o el programa Autodesk FBX Converter x64 que ayuda a convertir y reducir el tamaño de los archivos como se ve en la Figura 6.

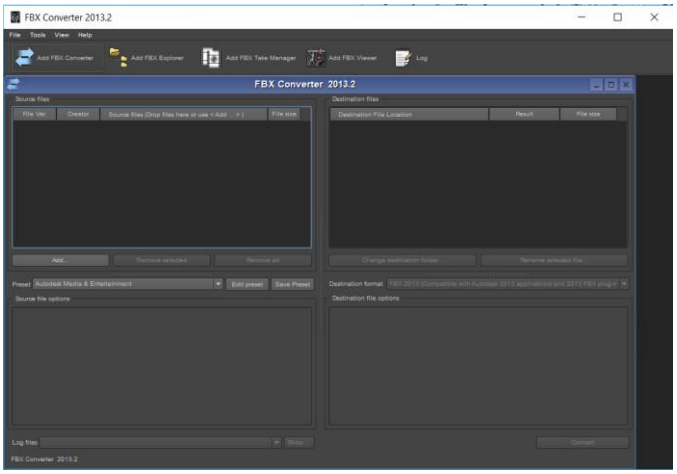


Figura 6: Pantalla del Autodesk FBX Converter

Realizado por: Christian Viracocha, Año 2018

4.4.5. Creación de los NPC del juego

Para crear los NPC o personajes no jugables es necesario tener el programa ¨MakeHuman¨, para lo cual se recomienda ver el video tutorial ¨**Como Utilizar MakeHuman para Unity**¨ con el link

¨https://www.youtube.com/watch?v=yJnpBz2up3o¨ en donde se enseña los aspectos básicos para su uso como se ve en la Figura 7.

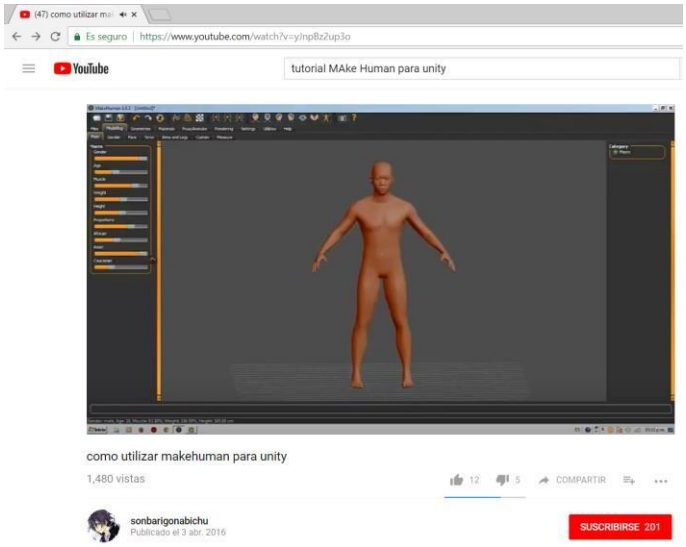


Figura 7: Pantalla del Tutorial de MakeHuman

Realizado por: Christian Viracocha, Año 2018

Una vez que se ha creado el modelo del personaje siguiendo los pasos del tutorial sobre MakeHuman se tendrá un personaje como se ve en la Figura 8.

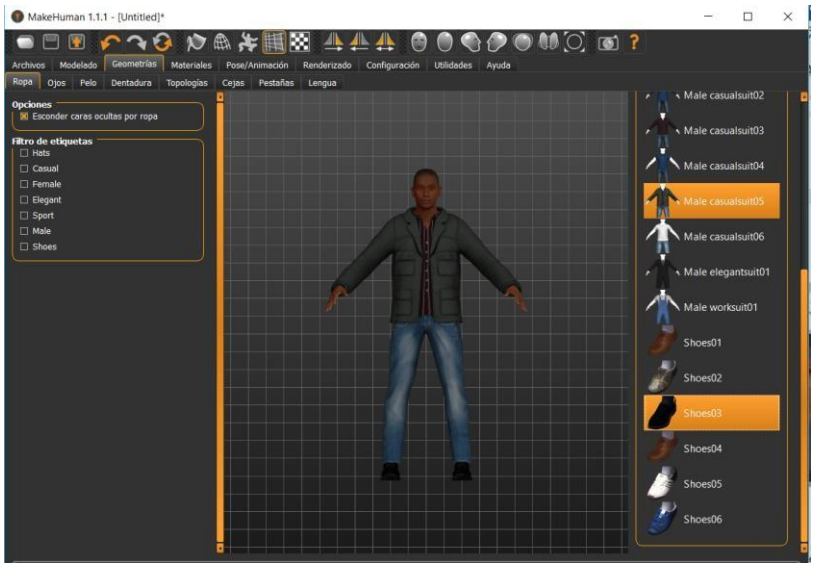


Figura 8: Pantalla de MakeHuman

Realizado por: Christian Viracocha, Año 2018

Para poder implementar las animaciones del personaje en Unity 3D es necesario implementar un esqueleto, con la estructura Game Engine que es la base para las animaciones como se muestra en la Figura 9.

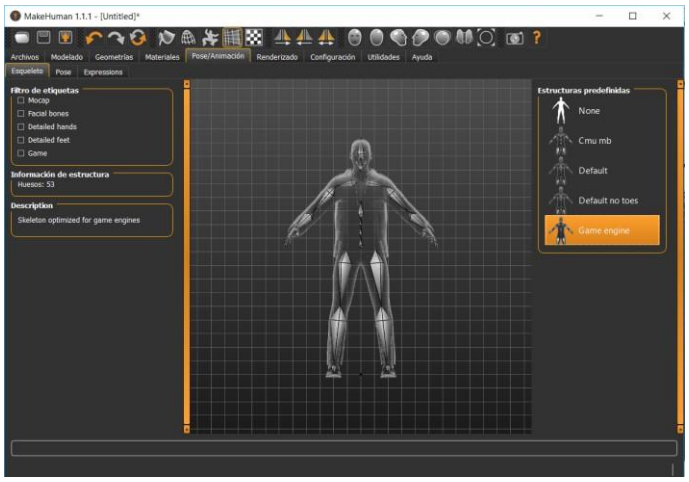


Figura 9: Pantalla de MakeHuman con el Esqueleto Game

Realizado por: Christian Viracocha, Año 2018

Una vez implementado todo el esqueleto se tendrá que exportar el personaje en el formato ¨Collada¨ para que sea reconocido por Unity y poder incorporar animaciones que se adquirieron por la tienda Asset Store como el nombre ¨Raw Mocap¨ que contiene las animaciones básicas para los personajes del juego.

4.5. Desarrollo del Mundo Virtual

4.5.1. Crear el proyecto en Unity

Para crear el juego en Unity, es necesario que, al inicio del programa se debe dar clic en New, en donde se muestra el formulario inicial del juego como también se puede incluir paquetes de los Assets del juego, que ya se encuentran instalados por default como se muestra en la Figura 10.

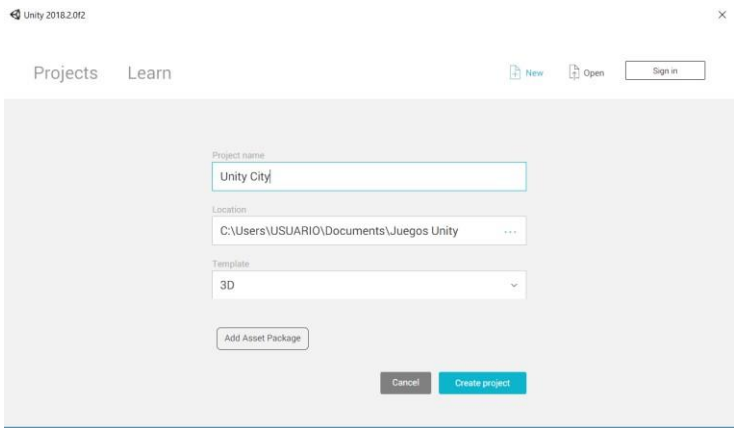


Figura 10: Pantalla de Unity 3D para Crear un Nuevo Proyecto

Realizado por: Christian Viracocha, Año 2018

4.5.2. Implementar los Assets Principales para el Juego

Los Assets son paquetes de archivos que proporciona Unity para el desarrollo del juego, que va desde objetos en 3D hasta scripts de desarrollo los cuales ayudan a mejorar el rendimiento y la lógica del juego. Los assets se consiguen del Asset Store de Unity como se muestra en la Figura 11 los cuales contribuirán en la ambientación del juego.

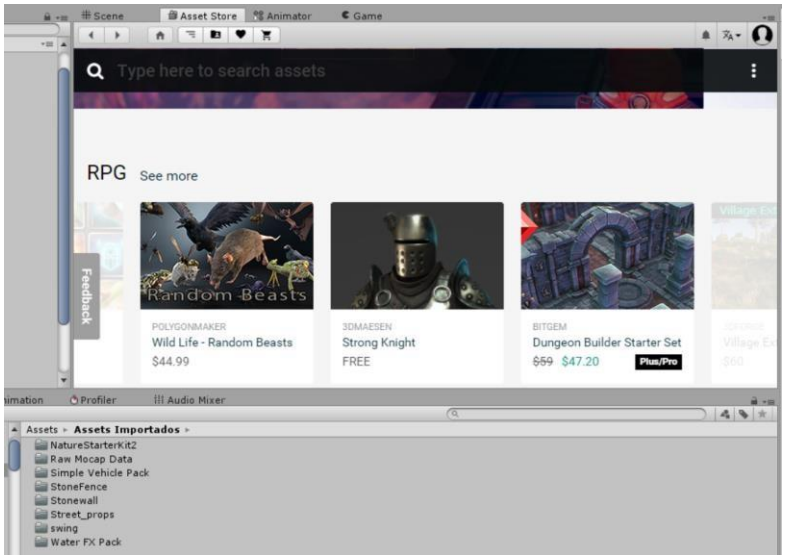


Figura 11: Pantalla del Asser Store

Realizado por: Christian Viracocha, Año 2018

Los Assets que se usan para crear el juego son:

- **Simple Cars Pack:** En el interior del paquete de Asset se encuentran 8 diferentes modelos de carros que sirven como decoración dentro de la ciudad
- **Swing:** Es un objeto 3D en forma de un columpio de madera que sirve como decoración en el parque.
- **Park Props Pack:** Es un paquete de objetos diseñados como decoración para un parque el cual consta de sillas de parque y postes de luz.
- **GAZ Street Props :** Es un paquete con objetos 3D con decoraciones que se encuentran en las calles de la ciudad, como por ejemplo los hidrantes, basureros y postes con diferentes modelos.
- **Dry Stone Wall with leafy Vines:** Es el paquete con objetos 3D el consta con diferentes versiones de muros de piedra con vegetación para la ambientación del mundo virtual.
- **Stone Fence:** Es el paquete con objetos 3D el consta con diferentes versiones de las vallas con cerramientos con sus respectivos postes para la ambientación del mundo virtual.
- **Simple Modular Street Kit:** Es el paquete con diferentes objetos que servirán para la implementación de las calles dentro de la ciudad, con diferentes diseños

- **Standard Assets:** Es uno de los paquetes más importantes dentro de Unity ya que contiene los diferentes scripts más básicos para la creación de un videojuego, como son las varias texturas que se usaran en el mundo virtual, como también la implementación de personajes jugables con sus respectivos scripts, entre muchos más.
- **Raw Mocap Data for Mecanim:** Es el paquete de las animaciones para los personajes jugables y no jugables dentro del mundo virtual.
- **Nature Starter kit 2:** Es el paquete de naturaleza el cual contiene los diferentes modelos 3D para la creación de un bosque como son los diseños de los árboles, hierba, arbustos entre muchos más.
- **Water FX Pack:** Es el paquete en el cual se encuentran los scripts que darán las animaciones de las partículas de agua, como también existirán las texturas necesarias para dar el efecto de agua.
- **QS MAterials Nature – Pack Grass vol. 1:** Es el paquete que consta de texturas de hierbas para el terreno.

4.5.3. Crear un Terreno y el ambiente de la Ciudad Virtual

Una vez que se ha abierto el proyecto de Unity, se creará el terreno en donde se podrá implementar la ciudad ficticia y utilizando los assets que se descargaron en la tienda se podrá dar la estructura base de la ciudad como se muestra en la Figura 12.

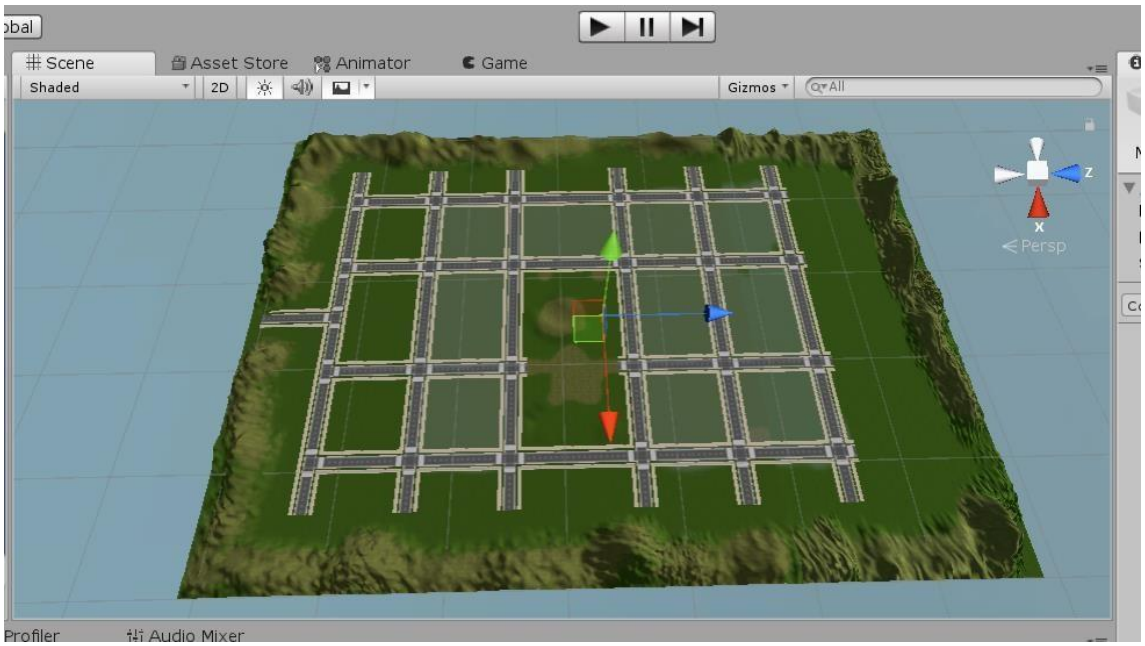


Figura 12: Pantalla de Terreno en Unity

Realizado por: Christian Viracocha, Año 2018

4.5.4. Importar los Objetos en 3D

Para importar los objetos que se realizaron anteriormente en el programa de SketchUp y MakeHuman, es necesario crear una carpeta especial dentro del directorio del proyecto, para almacenar los archivos.

Una vez seleccionado los archivos y carpetas de los modelos, estos deberán ser arrastrados al directorio del juego como se muestra en la Figura 13.

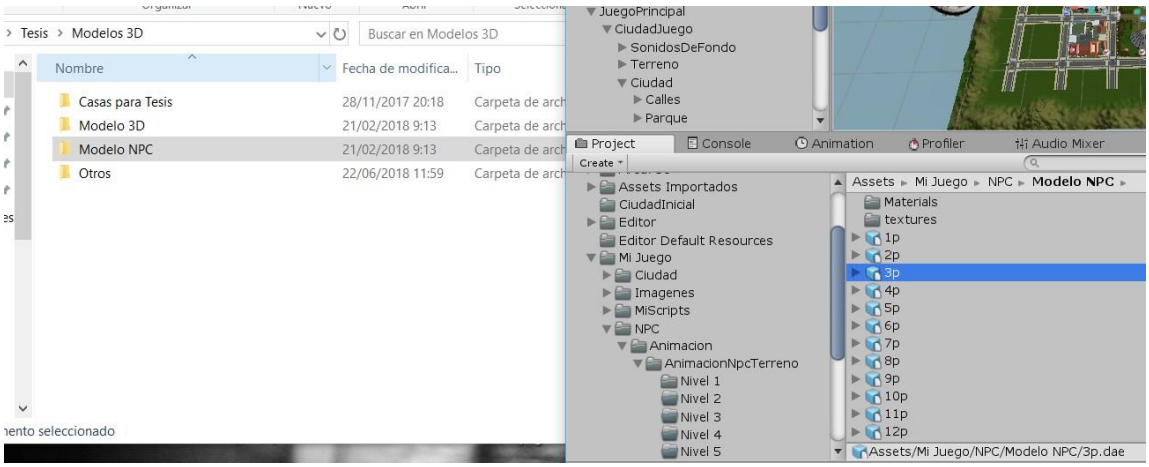


Figura 13: Pantalla para Importar Objetos a Unity 3D

Realizado por: Christian Viracocha, Año 2018

Para implementar las cuadras en 3D del juego, es necesario arrastrar los objetos desde el directorio del panel de Project hacia el Terreno, una vez ahí se los debe ubicar en el lugar apropiado y se debe tomar en cuenta la escala de la cuadra para que concuerde con los demás objetos, como se ve en la Figura 14.

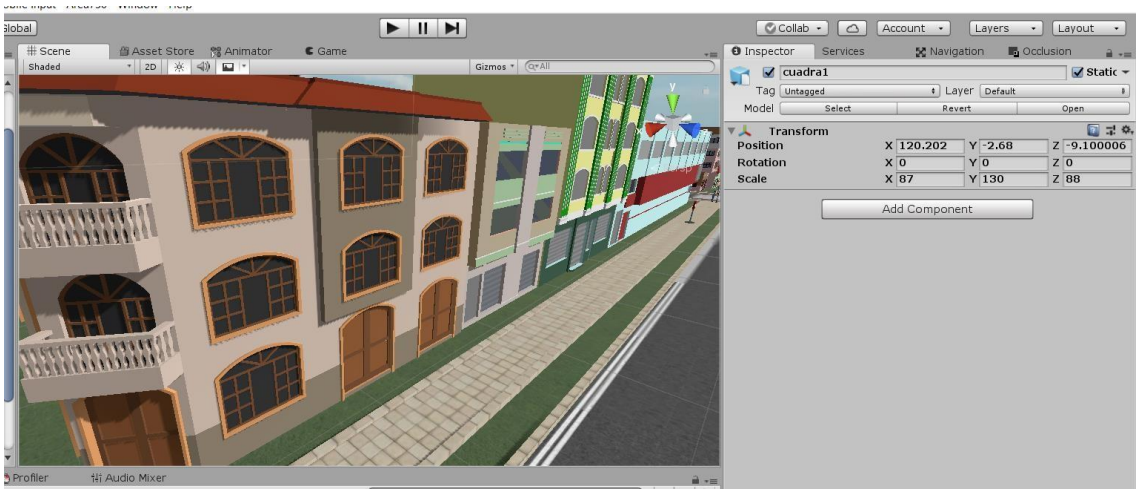


Figura 14: Pantalla Objetos en Unity

Realizado por: Christian Viracocha, Año 2018

4.5.5. Implementar los NPC dentro del juego

Para implementar los personajes en 3D que se realizaron en el programa de MakeHuman, deben ser arrastrados al terreno, en donde se dividen en componentes en el panel Hierarchy y se toma en cuenta que las texturas de los componentes de los personajes son metalizadas por lo que tienen que ser adaptadas al mundo virtual, como se muestra en la Figura 15.

Figura 15: Pantalla de Implementación de NPC´s

Realizado por: Christian Viracocha, Año 2018

4.5.6. Proceso de Animación de los NPC Normales

Para el proceso de animación es necesario poseer o haberse descargado el Asset ¨Raw Mocap Data¨ donde se encuentran las animaciones básicas para los personajes de un juego, como se muestra en la Figura 16.

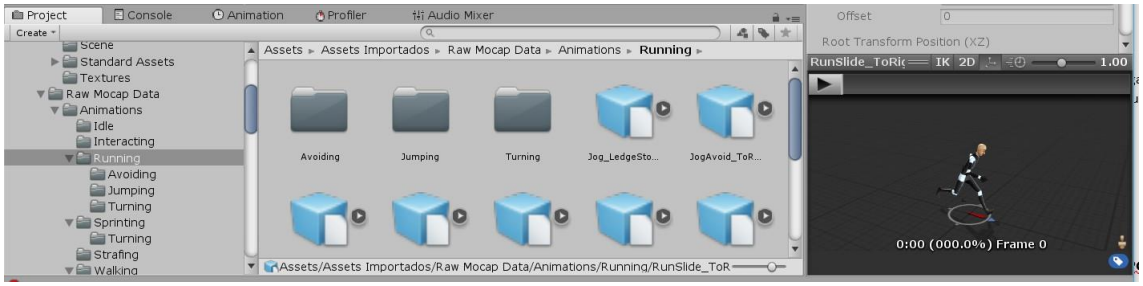


Figura 16: Pantalla de Animación de Raw Mocap

Realizado por: Christian Viracocha, Año 2018

Una vez seleccionado la animación para el personaje, es necesario crear un Animator Controller, con el fin de implementar un control de animaciones, como se muestra en la Figura 17.

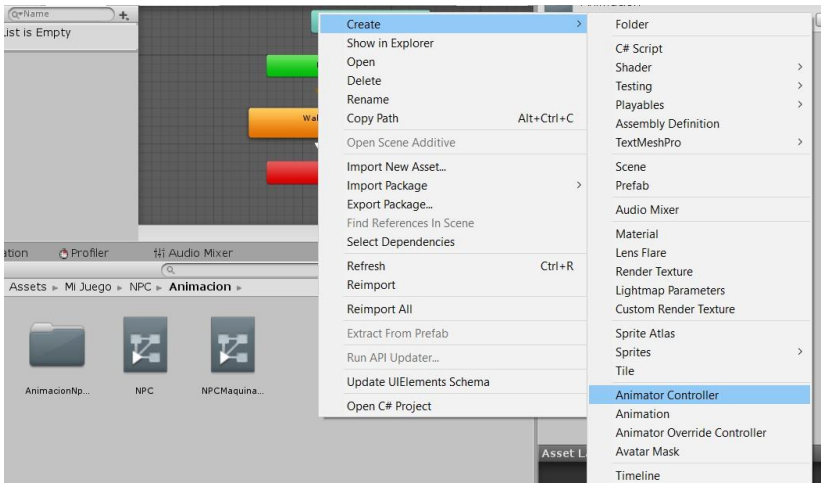


Figura 17: Pantalla de Creación de Animator Controller

Realizado por: Christian Viracocha, Año 2018

Después de haber creado el controlador de la animación es necesario arrastrar el archivo de animación hacia el panel del Animator, este panel aparece cuando se da doble clic en el archivo que se creó, estando en el panel es necesario crear una estructura de la animación, para que se sepa cuando entrar y cuando salir de la animación, como se muestra en la Figura 18.

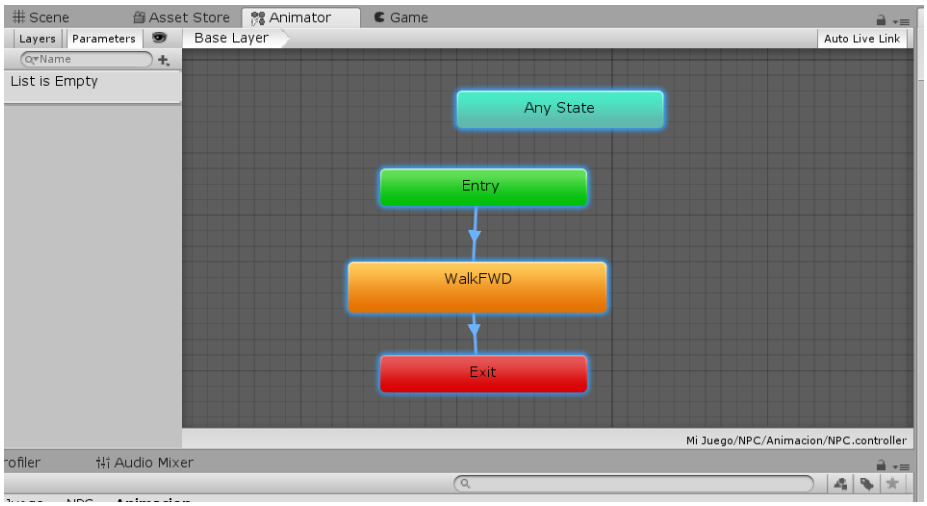


Figura 18: Pantalla de Animator

Realizado por: Christian Viracocha, Año 2018

Una vez terminado de crear la estructura de animación es necesario seleccionar nuevamente al personaje en el panel de Hierarchy para visualizar el componente Animator en el panel Inspector, para arrastrar el controlador de la animación en la opción Controller, para completar la animación del personaje, como se muestra en la Figura 19.

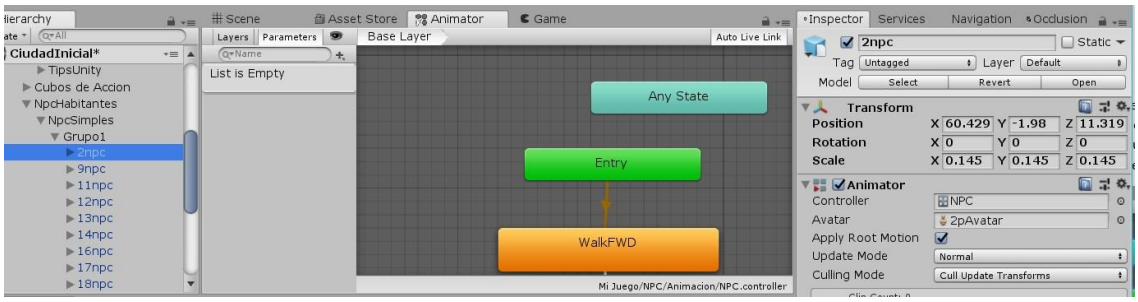


Figura 19: Pantalla de Implementación del Animator Controller

Realizado por: Christian Viracocha, Año 2018

4.5.6.1. Creación del Mapa de Navegación

Para que los personajes que hemos implementado anteriormente en el juego puedan moverse por la ciudad, es necesario crear un mapa de navegación. Se debe convertir todos los objetos del mapa en Static del juego para crear el mapa de navegación, como se muestra en la Figura 20.

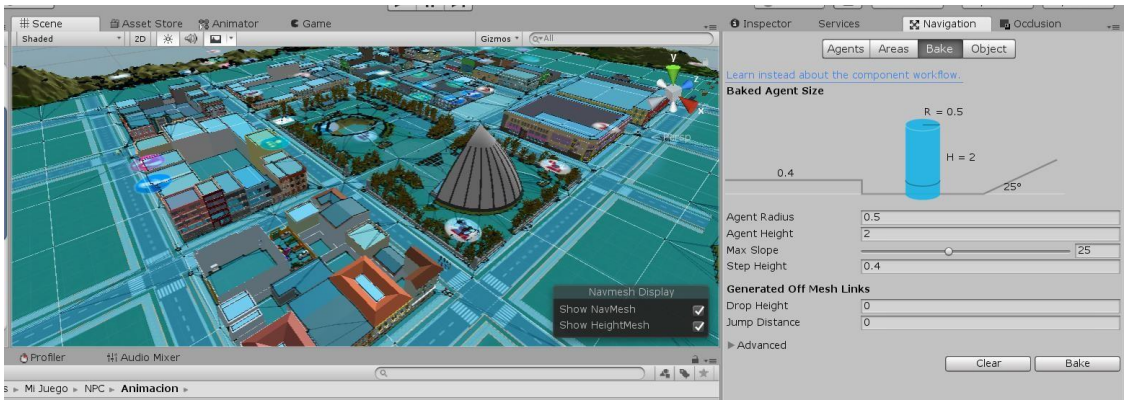


Figura 20: Pantalla del Mapa de Navegación

Realizado por: Christian Viracocha, Año 2018

Se recomienda ver los video tutoriales con el título ¨ **Tutorial Crear NPC - Navmesh** ¨ con el link ¨ https://www.youtube.com/watch?v=4BeSlmx-WwE¨ con el que nos ayuda crear los personajes no jugables en el juego.

4.5.6.2. Creación de los Scripts para los NPC

Para que los NPC´s puedan moverse con libre voluntad por el mundo virtual es necesario implementar el componente Nav Mesh Agent y un Script para el control de mallado, en donde podemos asignar el personaje y los puntos claves para poder desplazarse por el mundo virtual, mediante el siguiente código.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class controladorMalla : MonoBehaviour {
```

```
    public UnityEngine.AI.NavMeshAgent npc;
    public Transform objetivo;
    public Transform[] Nodos;
    public float offSet = 1;
    private int posicionActual = 0;

    // Use this for initialization
    void Start () {
        if (npc == null) {
            npc = this.gameObject.GetComponent<UnityEngine.AI.NavMeshAgent> ();
        }
        objetivo=Nodos[0];
    }

    // Update is called once per frame
    void Update () {
        npc.SetDestination (objetivo.position);
        Vector3 distancia;
        distancia = objetivo.position - transform.position;
        if (distancia.magnitude <= offSet) {
            posicionActual++;
            if (posicionActual >= Nodos.Length) {
                posicionActual = 0;
            }
            objetivo=Nodos[posicionActual];
        }

    }


}
```

Una vez terminado el código, éste debe ser implementado en el personaje, como un componente de éste. Los Target que se muestran en la Figura 21 del personaje, son objetos vacíos que se encuentran ubicados por toda la ciudad, para ser puntos de control para el desplazamiento del personaje.

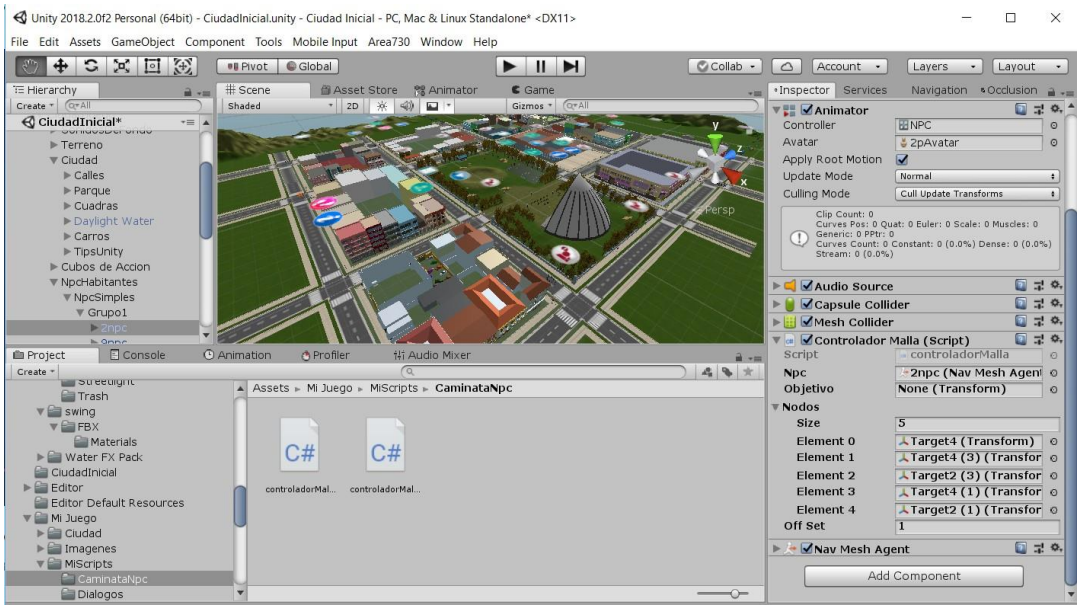


Figura 21: Pantalla de Implementación del Script para el NPC

Realizado por: Christian Viracocha, Año 2018

4.5.6.3. Implementación de la Animación de los NPC´s

Una vez terminado todos los pasos anteriores, el NPC o personaje no jugable puede moverse por toda la ciudad libremente, sin la necesidad de que el atraviese las edificaciones del juego, como se muestra en la Figura 22.

Figura 22: Pantalla de Animación de los NPC

Realizado por: Christian Viracocha, Año 2018

4.5.7. Proceso Animación para los NPC´s Especiales

El proceso de animación es idéntico al anterior proceso, sólo cambia la estructura de animación.

Se recomienda visualizar el video ¨ **Basic Artificial Intelligence for a Non-Player Character with Unity 5**¨ con el link ¨ https://www.youtube.com/watch?v=gXpi1czz5NA&list=PL5ej1bTaHVGo2Xrpsa4KHOCmxIFT7oYY-&index=26¨ en donde se muestra el proceso para la implementación de la inteligencia artificial que vamos a crear con la máquina de estados del personaje.

El animator controller que se creó para la máquina de estados, en donde se identifica tres diferentes estados el personaje no jugable, con sus respectivas animaciones, los cuales van a ser identificados en el script de control de mallado para la máquina de estados como se muestra en la Figura 23.

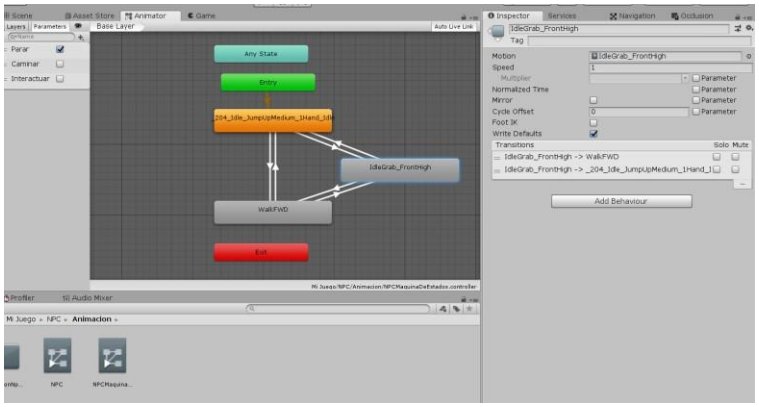


Figura 23: Pantalla de Animator para el NPC especial

Realizado por: Christian Viracocha, Año 2018

4.5.7.1. Creación de los Scripts de Maquina de Estados

El script para la máquina de estados tiene el objetivo de cambiar los estados del personaje no jugable, al contacto con el jugador ya que este puede identificar al jugador y distancia que se encuentra de él, para implementar diferentes animaciones ya que posee una relación con el con el Animator Controller, se recomienda crear un script por cada Personaje no jugable ya que puede existir Bugs dentro del juego y los npc no funcionar, el Script que se uso es:

```
using System.Collections;
        using System.Collections.Generic;
        using UnityEngine;

        public class MaquinaEstadoIA1: MonoBehaviour {
            public Transform player;
            static Animator anim;

            void Start () {
                anim = GetComponent<Animator> ();

            }

            // Update is called once per frame
            void Update () {
                if (Vector3.Distance (player.position, this.transform.position) < 10) {
                    Vector3 direccion = player.position - this.transform.position;
                    direccion.y = 0;

                    this.transform.rotation = Quaternion.Slerp (this.transform.rotation, Quaternion.LookRotation (direccion), 0.1f);

                    anim.SetBool ("Parar", false);

                    if (direccion.magnitude > 5) {

                        anim.SetBool ("Caminar", true);


                        anim.SetBool ("Interactuar", false);


                    } else {
                        anim.SetBool ("Interactuar", true);
                        anim.SetBool ("Caminar", false);
                    }
                } else {
                    anim.SetBool ("Parar", true);
                    anim.SetBool ("Caminar", false);
                    anim.SetBool ("Interactuar", false);
                }


            }

        }
```

En este script Podemos identificar que mide la distancia del jugador y de acuerdo con eso el código cambia de forma automática los estados del jugador ya que si el personaje no ve al jugador va a estar en estado de reposo, pero si el jugador está a menos 10 m, este va a caminar hacia él y si está a menos de 5m va a que cambiar de estado al estado de interactuar.

4.5.7.2. Implementación de la animación de los NPC para cada estado

Para finalizar la animación con la máquina de estados se implementará el Script de IA e identificaremos el player dentro del componente del Script como se muestra en la Figura 24.

Figura 24: Pantalla de Implementación de la Animación del NPC con la Maquina de Estados

Realizado por: Christian Viracocha, Año 2018

4.5.8. Manejo de la información educativa puntajes del juego

Para manejar los puntajes juego al realizar alguna acción como el cumplimiento de objetivos, es necesario manejar una base de datos interna del juego donde se almacenan los puntos, para poder usarlos en todo el transcurso del juego, el script que se uso es el siguiente:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class GenerarData {

   public static GenerarData instance;

   public int currentScore = 0;
   //Nivel 1
   public float score1=0.0f;
   public float score2=0.0f;
   public float score3=0.0f;
   public float score4=0.0f;
   public float score5=0.0f;

   private GenerarData(){
      currentScore = PlayerPrefs.GetInt("score", 0);
      Debug.Log(currentScore);
   }
   public static GenerarData getInstance(){
      if (instance == null)
      {
         instance = new GenerarData();
      }
      return instance;
   }

}
```

Este Script no es asignado a ningún objeto, por lo que es usado por otros scripts que manejan información como el script del Test del juego, en donde al contestar preguntas aumentan su puntaje y estos son almacenados en este Script.

4.5.8.1. Creación del Canvas

El canvas es una espacie de interfaz en 2D que sirve para crear los Menús o el visor de elementos del jugador que usa el juego de UnityCity para visualizar los objetivos o scores del juego, como se muestra en la Figura 25.

Figura 25: Pantalla del Canvas del Juego

Realizado por: Christian Viracocha, Año 2018

Para mejorar el manejo de los canvas es necesario la manipulación de las anclas de cada elemento que se usa dentro de él, se recomienda ver el video ¨ **Hagamos Videojuegos - Anclas y organización de UI**¨ con el Link ¨ https://www.youtube.com/watch?v=S73dCxpC81E¨, en donde nos ayuda a entender el manejo de los canvas para el juego.

Para el juego Unity City es necesario tener varios canvas que serán las interfaces que mostrarán, los videos, los pasos, los Test entre otras pantallas con el fin de otorgar al jugador el material educativo como se muestra en la Figura 26.

Figura 26: Pantallas de los Canvas del Juego

Realizado por: Christian Viracocha, Año 2018

Para la creación de los Test dentro del canvas se usaron diferentes elementos y scripts para su funcionamiento como se muestra en la Figura 27, con el fin de manejar y almacenar la información de las calificaciones de los Tests.

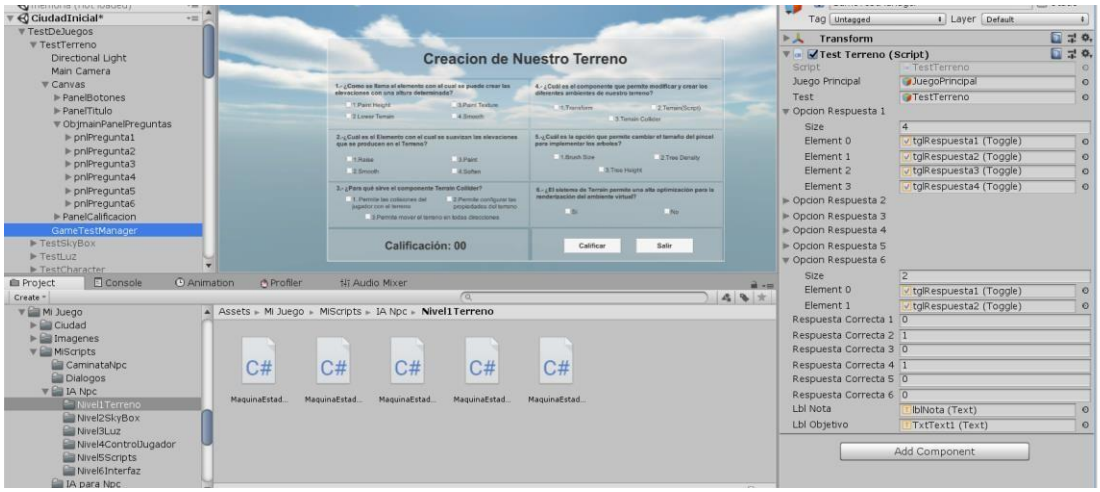


Figura 27: Pantalla del Canvas del Test

Realizado por: Christian Viracocha, Año 2018

Para crear el mini mapa es recomendable ver el siguiente video ¨**Minimap Circular en Unity3D**¨ con el Link ¨https://www.youtube.com/watch?v=2aMpxKUpnKM¨ para saber cómo hacer un mapa estable como se muestra en la Figura 28.

Figura 28: Pantalla del Mini Mapa

Realizado por: Christian Viracocha, Año 2018

4.5.8.2. Creación del Script para el Canvas

Para la implementación de los canvas dentro del juego es necesario que los personajes no jugables especiales posean un objeto en su interior de forma rectangular, en donde se implementen los Tiggers que permiten identificar al cuando el jugador está en frente del personaje y así mostrar el canvas respectivo, como se muestra en la Figura 29.

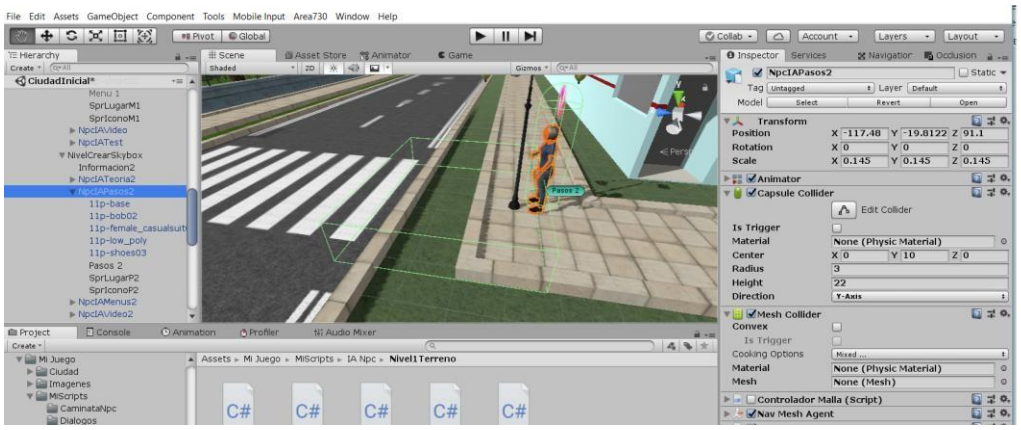


Figura 29: Pantalla donde se Muestra el Area del Trigger

Realizado por: Christian Viracocha, Año 2018

Cada uno de los objetos en forma rectangular poseen diferentes scripts que permiten almacenar información como texto e imágenes, los cuales son procesados por el script principal, con el fin de mostrar los canvas con la información específica de cada NPC, como se muestra la Figura 30.

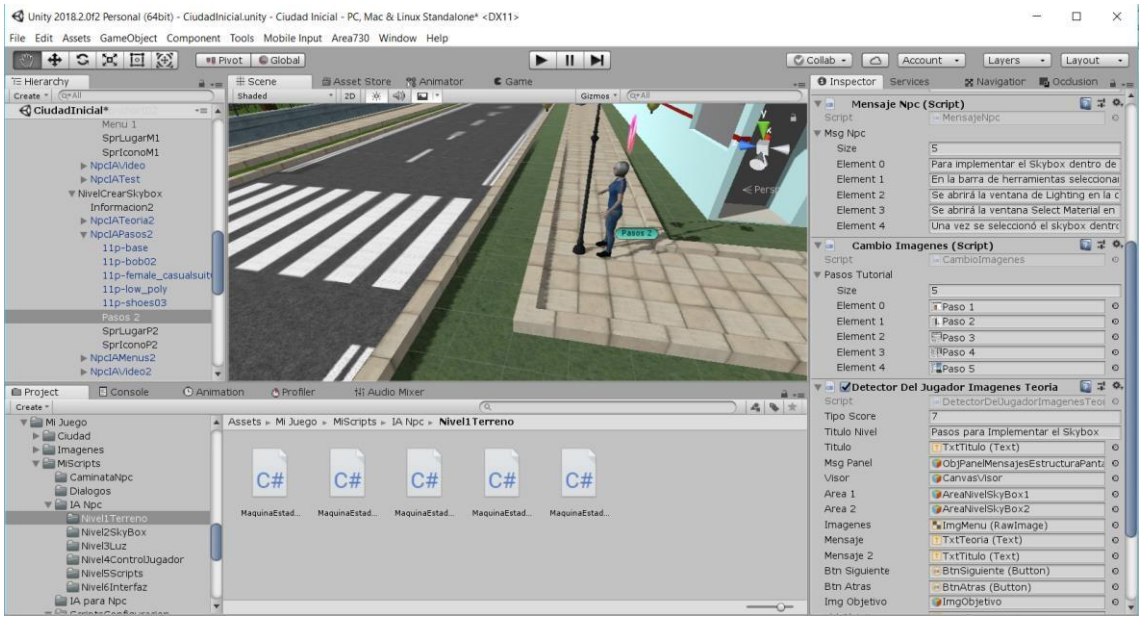


Figura 30: Pantalla de la información de cada NPC

Realizado por: Christian Viracocha, Año 2018

El Script principal tiene el nombre ¨Detector del Jugador¨ y dependiendo de su canvas maneja diferente información, a continuación, se presenta los Scripts para el manejo de cada uno de los Canvas y de su información. Este script nos permite mostrar el canvas de la introducción al Test, además transporta al canvas de las preguntas del test mediante los botones.

Script Mensaje NPC

Este script nos permite almacenar información de tipo texto para ser procesado en el canvas de información.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MensajeNpc : MonoBehaviour {

    int index =0;
    public string[] msgNpc ;


    public string GetMsg(){

        if (index == msgNpc.Length) {
            index = msgNpc.Length-1;
            return msgNpc [msgNpc.Length-1];
        } else {
            return msgNpc [index];
        }


    }

    public void BtnSiguiente()
    {
        if(index <= msgNpc.Length){
            index++;
        }
    }

    public void BtnAtras()
    {

        index = index - 1;
        if (index == -1) {
            index = 0;

        } else {
            index=index;
        }

    }

    public void BtnReiniciar()
    {
        index=0;
    }

}
```

Script Cambio de Imágenes

Este script nos permite almacenar imágenes para ser procesado en el canvas de información mediante sus diferentes funciones.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class CambioImagenes : MonoBehaviour {

    int index=0;
    public Texture[] pasosTutorial;

    // Use this for initialization
    public Texture GetPasos() {

        if (index == pasosTutorial.Length) {
            index = pasosTutorial.Length-1;
            return pasosTutorial [pasosTutorial.Length-1];
        } else {
            return pasosTutorial [index];
        }
    }

    public void BtnSiguiente()
    {
        if(index <= pasosTutorial.Length){
            index++;
        }
    }

    public void BtnAtras()
    {

        index = index - 1;
        if (index == -1) {
            index = 0;

        } else {
            index=index;
        }

    }

    public void BtnReiniciar()
    {
        index=0;
    }
}
```

Script Detector del jugador Teoría

Este script nos permite manejar la gestión de objetos e información del juego, con el fin de apagar y encender objetos de la interfaz del jugador.

```
using UnityEngine;
using UnityEngine.UI;

public class DetectorDelJugadorTeoria : MonoBehaviour {
    public int tipoScore;
    public string tituloNivel;
    private int cont=0;
// [SerializeField] GameObject scriptMensaje;
    [SerializeField] GameObject msgPanel;
    [SerializeField] GameObject visor;
    [SerializeField] GameObject area1;
    [SerializeField] GameObject area2;
    [SerializeField] Text mensaje;
    [SerializeField] Text titulo;
    [SerializeField] Button btnSiguiente;
    [SerializeField] Button btnAtras;
    public Text lblObjetivo;
    [SerializeField] GameObject imgObjetivo;
    [SerializeField] GameObject estructuraVisor;
    // Use this for initialization


    void Start () {
        GetComponent<Collider>().isTrigger = true;
        msgPanel.SetActive (false);
        btnSiguiente.onClick.AddListener(BtnSiguiente);
        btnAtras.onClick.AddListener(BtnAtras);
    }


    public void MostrarMensaje(){
        BtnReinicio();

//          string msg = GameObject.Find(scriptMensaje).Get
Msg ();
            string msg = GetComponent<MensajeNpc> ().GetM
sg ();
            mensaje.text = msg;
            titulo.text = tituloNivel;
            msgPanel.SetActive (true);
            visor.SetActive(false);
            area1.SetActive(false);
            area2.SetActive(false);

    }

    public void CerrarMensaje(){

        BtnReinicio();


            GetComponent<MensajeNpc>().BtnReiniciar();
            string msg = GetComponent<MensajeNpc>().GetMs
g();
            mensaje.text = msg;
            msgPanel.SetActive (false);
            visor.SetActive(true);
            area1.SetActive(true);
            area2.SetActive(true);
            PorcentajeObjetivo();
            CambioDeMensaje();
        ApagarEstructura();
        PrenderImagen();
        Invoke("ApagarImagen", 2f);
        Invoke("PrenderEstructura", 2f);
    }

    public void BtnSiguiente(){
        GetComponent<MensajeNpc>().BtnSiguiente();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        mensaje.text = msg;

    }

    public void BtnAtras(){
        GetComponent<MensajeNpc>().BtnAtras();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
         mensaje.text = msg;


    }

    public void BtnReinicio(){
        GetComponent<MensajeNpc>().BtnReiniciar();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        mensaje.text = msg;


    }

    void Update () {
        if (Input.GetKeyDown (KeyCode.E))
            BtnSiguiente ();

        if (Input.GetKeyDown (KeyCode.Q))
            BtnAtras();
    }


    public void PorcentajeObjetivo(){
        GenerarData gd = GenerarData.getInstance();
        // Nivel 1
        ////////////////////////////////////////
        if(tipoScore==1)
        {
            gd.score1 = 1;
            Debug.Log(gd.score1);
        }

        if(tipoScore==2)
        {
            gd.score2 = 1;
            Debug.Log(gd.score2);
        }

        if(tipoScore==3)
        {
            gd.score3 = 2;
            Debug.Log(gd.score3);
        }

        if(tipoScore==4)
        {
            gd.score4 = 2;
            Debug.Log(gd.score4);
        }

        if(tipoScore==5)
        {
            gd.score5 = 10;
            Debug.Log(gd.score5);
        }

        ////////////////////////////////////////

        // Nivel 2
        ////////////////////////////////////////
        if(tipoScore==6)
        {
            gd.score6 = 1;
            Debug.Log(gd.score6);
        }

        if(tipoScore==7)
        {
            gd.score7 = 1;
            Debug.Log(gd.score7);
```

```
}

if(tipoScore==8)
{
   gd.score8 = 2;
   Debug.Log(gd.score8);
}

if(tipoScore==9)
{
   gd.score9 = 2;
   Debug.Log(gd.score9);
}

if(tipoScore==10)
{
   gd.score10 = 10;
   Debug.Log(gd.score10);
}

//////////////////////////////////////////

// Nivel 3
//////////////////////////////////////////
if(tipoScore==11)
{
   gd.score11 = 1;
   Debug.Log(gd.score11);
}

if(tipoScore==12)
{
   gd.score12 = 1;
   Debug.Log(gd.score12);
}

if(tipoScore==13)
{
   gd.score13 = 2;
   Debug.Log(gd.score13);
}

if(tipoScore==14)
{
   gd.score14 = 2;
   Debug.Log(gd.score14);
}

if(tipoScore==15)
{
   gd.score15 = 10;
   Debug.Log(gd.score15);
}

//////////////////////////////////////////

// Nivel 4
//////////////////////////////////////////
if(tipoScore==16)
{
   gd.score16 = 1;
   Debug.Log(gd.score16);
}

if(tipoScore==17)
{
   gd.score17 = 1;
   Debug.Log(gd.score17);
}

if(tipoScore==18)
{
   gd.score18 = 2;
   Debug.Log(gd.score18);
}

if(tipoScore==19)
{
   gd.score19 = 2;
   Debug.Log(gd.score19);
}

if(tipoScore==20)
{
   gd.score20 = 10;
   Debug.Log(gd.score20);
}

//////////////////////////////////////////
// Nivel 5
//////////////////////////////////////////
if(tipoScore==21)
{
   gd.score21 = 1;
   Debug.Log(gd.score21);
}

if(tipoScore==22)
{
   gd.score22 = 1;
   Debug.Log(gd.score22);
}

if(tipoScore==23)
{
   gd.score23 = 2;
   Debug.Log(gd.score23);
}

if(tipoScore==24)
{
   gd.score24 = 2;
   Debug.Log(gd.score24);
}

if(tipoScore==25)
{
   gd.score25 = 10;
   Debug.Log(gd.score25);
}

//////////////////////////////////////////

// Nivel 6
//////////////////////////////////////////
if(tipoScore==26)
{
   gd.score26 = 1;
   Debug.Log(gd.score26);
}

if(tipoScore==27)
{
   gd.score27 = 1;
   Debug.Log(gd.score27);
}

if(tipoScore==28)
{
   gd.score28 = 2;
   Debug.Log(gd.score28);
}

if(tipoScore==29)
{
   gd.score29 = 2;
   Debug.Log(gd.score29);
}

if(tipoScore==30)
{
   gd.score30 = 2;
```

```
            Debug.Log(gd.score30);
        }

        if(tipoScore==31)
        {
            gd.score31 = 2;
            Debug.Log(gd.score31);
        }

        if(tipoScore==32)
        {
            gd.score32 = 10;
            Debug.Log(gd.score32);
        }
        ////////////////////////////////////////


    }

    void CambioDeMensaje()
    {
        lblObjetivo.text = lblObjetivo.text.Replace("#323232F
F","#ff0000ff");
    }

    void PrenderImagen()
    {
        if (cont == 0)
        {
            imgObjetivo.SetActive(true);
            cont = 1;

        }
    }

    void ApagarImagen()
    {

        imgObjetivo.SetActive(false);

    }

    void PrenderEstructura()
    {
        estructuraVisor.SetActive(true);
    }

    void ApagarEstructura()
    {
        estructuraVisor.SetActive(false);
    }
}
```

Script Detector del jugador Imágenes Teoría

Este script nos permite manejar la gestión de objetos e información del juego, con el fin de apagar y encender objetos de la interfaz del jugador.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
public class DetectorDelJugadorImagenesTeoria : MonoBeh
aviour {
    private int cont=0;
    public int tipoScore;
    public string tituloNivel;
    [SerializeField] Text titulo;
    [SerializeField] GameObject msgPanel;
    [SerializeField] GameObject visor;
    [SerializeField] GameObject area1;
    [SerializeField] GameObject area2;
    [SerializeField] RawImage imagenes;
    [SerializeField] Text mensaje;
    [SerializeField] Text mensaje2;
    [SerializeField] Button btnSiguiente;
    [SerializeField] Button btnAtras;
    [SerializeField] GameObject imgObjetivo;
    public Text lblObjetivo;
    [SerializeField] GameObject estructuraVisor;
    // Use this for initialization
    void Start () {
        GetComponent<Collider>().isTrigger = true;
        msgPanel.SetActive (false);
        btnSiguiente.onClick.AddListener(BtnSiguiente);
        btnAtras.onClick.AddListener(BtnAtras);
    }


    public void MostrarMensaje(){
        BtnReinicio();
            titulo.text = tituloNivel;
            Texture imagenPasos = GetComponent<CambioIma
genes> ().GetPasos ();
            imagenes.texture = imagenPasos;
            msgPanel.SetActive (true);
            visor.SetActive(false);
            string msg = GetComponent<MensajeNpc>().GetM
sg ();
            string msgTitulo =GetComponent<MensajeNpcTitu
lo> ().GetMsg ();
            mensaje.text = msg;
            mensaje2.text = msgTitulo;
            msgPanel.SetActive (true);

            visor.SetActive(false);
            area1.SetActive(false);
            area2.SetActive(false);


    }

    public void CerrarMensaje(){
        BtnReinicio();
            GetComponent<MensajeNpc>().BtnReiniciar();
            GetComponent<MensajeNpcTitulo>().BtnReiniciar(
);
            string msg = GetComponent<MensajeNpc>().GetMs
g();
            string msgTitulo = GetComponent<MensajeNpcTitu
lo>().GetMsg();
            mensaje.text = msg;
            mensaje2.text = msgTitulo;
            msgPanel.SetActive (false);

            visor.SetActive(true);
            area1.SetActive(true);
            area2.SetActive(true);

            GetComponent<CambioImagenes>().BtnReiniciar();
            Texture imagenPasos = GetComponent<CambioIma
genes> ().GetPasos ();
            imagenes.texture = imagenPasos;
            msgPanel.SetActive (false);
            visor.SetActive(true);
```

```
            PorcentajeObjetivo();
        CambioDeMensaje();
        ApagarEstructura();
        PrenderImagen();
        Invoke("ApagarImagen", 2f);
        Invoke("PrenderEstructura", 2f);
    }

    public void BtnSiguiente(){
        GetComponent<CambioImagenes>().BtnSiguiente();
        Texture imagenPasos = GetComponent<CambioImage
nes> ().GetPasos ();
        imagenes.texture = imagenPasos;

        GetComponent<MensajeNpc>().BtnSiguiente();
        GetComponent<MensajeNpcTitulo>().BtnSiguiente();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        string msgTitulo = GetComponent<MensajeNpcTitulo
>().GetMsg();
        mensaje.text = msg;
        mensaje2.text = msgTitulo;
    }

    public void BtnAtras(){
        GetComponent<CambioImagenes>().BtnAtras();
        Texture imagenPasos = GetComponent<CambioImage
nes> ().GetPasos ();
        imagenes.texture = imagenPasos;

        GetComponent<MensajeNpc>().BtnAtras();
        GetComponent<MensajeNpcTitulo>().BtnAtras();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        string msgTitulo = GetComponent<MensajeNpcTitulo
>().GetMsg();
        mensaje.text = msg;
        mensaje2.text = msgTitulo;
    }

    public void BtnReinicio(){
        GetComponent<MensajeNpc>().BtnReiniciar();
        GetComponent<MensajeNpcTitulo>().BtnReiniciar();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        string msgTitulo = GetComponent<MensajeNpcTitulo
>().GetMsg();
        mensaje.text = msg;
        mensaje2.text = msgTitulo;

        GetComponent<CambioImagenes>().BtnReiniciar();
        Texture imagenPasos = GetComponent<CambioImage
nes> ().GetPasos ();
        imagenes.texture = imagenPasos;

    }

    void Update () {
        if (Input.GetKeyDown (KeyCode.E))
            BtnSiguiente ();

        if (Input.GetKeyDown (KeyCode.Q))
            BtnAtras();
    }

    public void PorcentajeObjetivo(){
        GenerarData gd = GenerarData.getInstance();
        // Nivel 1
        ////////////////////////////////////////
        if(tipoScore==1)
        {
            gd.score1 = 1;
            Debug.Log(gd.score1);
        }

        if(tipoScore==2)
        {
            gd.score2 = 1;
            Debug.Log(gd.score2);
        }

        if(tipoScore==3)
        {
            gd.score3 = 2;
            Debug.Log(gd.score3);
        }

        if(tipoScore==4)
        {
            gd.score4 = 2;
            Debug.Log(gd.score4);
        }

        if(tipoScore==5)
        {
            gd.score5 = 10;
            Debug.Log(gd.score5);
        }

        ////////////////////////////////////////

        // Nivel 2
        ////////////////////////////////////////
        if(tipoScore==6)
        {
            gd.score6 = 1;
            Debug.Log(gd.score6);
        }

        if(tipoScore==7)
        {
            gd.score7 = 1;
            Debug.Log(gd.score7);
        }

        if(tipoScore==8)
        {
            gd.score8 = 2;
            Debug.Log(gd.score8);
        }

        if(tipoScore==9)
        {
            gd.score9 = 2;
            Debug.Log(gd.score9);
        }

        if(tipoScore==10)
        {
            gd.score10 = 10;
            Debug.Log(gd.score10);
        }

        ////////////////////////////////////////

        // Nivel 3
        ////////////////////////////////////////
        if(tipoScore==11)
        {
            gd.score11 = 1;
            Debug.Log(gd.score11);
        }

        if(tipoScore==12)
        {
            gd.score12 = 1;
            Debug.Log(gd.score12);
        }

        if(tipoScore==13)
        {
            gd.score13 = 2;
```

```
    Debug.Log(gd.score13);
}

if(tipoScore==14)
{
    gd.score14 = 2;
    Debug.Log(gd.score14);
}

if(tipoScore==15)
{
    gd.score15 = 10;
    Debug.Log(gd.score15);
}

//////////////////////////////////////

// Nivel 4
//////////////////////////////////////
if(tipoScore==16)
{
    gd.score16 = 1;
    Debug.Log(gd.score16);
}

if(tipoScore==17)
{
    gd.score17 = 1;
    Debug.Log(gd.score17);
}

if(tipoScore==18)
{
    gd.score18 = 2;
    Debug.Log(gd.score18);
}

if(tipoScore==19)
{
    gd.score19 = 2;
    Debug.Log(gd.score19);
}

if(tipoScore==20)
{
    gd.score20 = 10;
    Debug.Log(gd.score20);
}

//////////////////////////////////////
// Nivel 5
//////////////////////////////////////
if(tipoScore==21)
{
    gd.score21 = 1;
    Debug.Log(gd.score21);
}

if(tipoScore==22)
{
    gd.score22 = 1;
    Debug.Log(gd.score22);
}

if(tipoScore==23)
{
    gd.score23 = 2;
    Debug.Log(gd.score23);
}

if(tipoScore==24)
{
    gd.score24 = 2;
    Debug.Log(gd.score24);
}

if(tipoScore==25)
{
    gd.score25 = 10;
    Debug.Log(gd.score25);
}

//////////////////////////////////////

// Nivel 6
//////////////////////////////////////
if(tipoScore==26)
{
    gd.score26 = 1;
    Debug.Log(gd.score26);
}

if(tipoScore==27)
{
    gd.score27 = 1;
    Debug.Log(gd.score27);
}

if(tipoScore==28)
{
    gd.score28 = 2;
    Debug.Log(gd.score28);
}

if(tipoScore==29)
{
    gd.score29 = 2;
    Debug.Log(gd.score29);
}

if(tipoScore==30)
{
    gd.score30 = 2;
    Debug.Log(gd.score30);
}

if(tipoScore==31)
{
    gd.score31 = 2;
    Debug.Log(gd.score31);
}

if(tipoScore==32)
{
    gd.score32 = 10;
    Debug.Log(gd.score32);
}
//////////////////////////////////////


}
void CambioDeMensaje()
{
    //lblObjetivo.text="Listo";
    lblObjetivo.text = lblObjetivo.text.Replace("#323232F
F","#ff0000ff");
}

void PrenderImagen()
{
    if (cont == 0)
    {
        imgObjetivo.SetActive(true);
        cont = 1;

    }
}

void ApagarImagen()
{

    imgObjetivo.SetActive(false);
```

```
    }

    void PrenderEstructura()
    {
        estructuraVisor.SetActive(true);
    }

    void ApagarEstructura()
    {
        estructuraVisor.SetActive(false);
    }
}
```

Script Detector del jugador Video

Este script nos permite manejar la gestión de objetos e información del juego, con el fin de apagar y encender objetos de la interfaz del jugador para mostrar los videos del tutorial.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class DetectorDelJugadorVideo : MonoBehaviour {

    private int cont=0;
    public int tipoScore;
  // public string tituloNivel;
  // [SerializeField] Text titulo;
    [SerializeField] GameObject msgPanel;
    [SerializeField] GameObject visor;
    [SerializeField] Text mensaje;
    [SerializeField] Button btnSiguiente;
    [SerializeField] Button btnAtras;
    [SerializeField] GameObject area1;
    [SerializeField] GameObject area2;
    public Text lblObjetivo;
    public Video canvasVideo;
    // Use this for initialization
    [SerializeField] GameObject imgObjetivo;
    [SerializeField] GameObject estructuraVisor;
    void Start () {
        GetComponent<Collider>().isTrigger = true;
        msgPanel.SetActive (false);
    }


    public void MostrarMensaje(){



            msgPanel.SetActive (true);
    // titulo.text = tituloNivel;
            visor.SetActive(false);
        area1.SetActive(false);
        area2.SetActive(false);

    }

    public void CerrarMensaje(){


            msgPanel.SetActive (false);
            canvasVideo.StopVideo ();
            visor.SetActive(true);
            PorcentajeObjetivo();
        area1.SetActive(true);
        area2.SetActive(true);
        CambioDeMensaje();
        ApagarEstructura();
        PrenderImagen();
        Invoke("ApagarImagen", 2f);
        Invoke("PrenderEstructura", 2f);
    }

    public void BtnSiguiente(){
        GetComponent<MensajeNpc>().BtnSiguiente();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        mensaje.text = msg;

    }

    public void BtnAtras(){
        GetComponent<MensajeNpc>().BtnAtras();
        string msg = GetComponent<MensajeNpc>().GetMsg()
;
        mensaje.text = msg;

    }

    public void PorcentajeObjetivo(){
        GenerarData gd = GenerarData.getInstance();
        // Nivel 1
        ////////////////////////////////////////
        if(tipoScore==1)
        {
            gd.score1 = 1;
            Debug.Log(gd.score1);
        }

        if(tipoScore==2)
        {
            gd.score2 = 1;
            Debug.Log(gd.score2);
        }

        if(tipoScore==3)
        {
            gd.score3 = 2;
            Debug.Log(gd.score3);
        }

        if(tipoScore==4)
        {
            gd.score4 = 2;
            Debug.Log(gd.score4);
        }

        if(tipoScore==5)
        {
            gd.score5 = 10;
            Debug.Log(gd.score5);
        }

        ////////////////////////////////////////

        // Nivel 2
        ////////////////////////////////////////
        if(tipoScore==6)
        {
            gd.score6 = 1;
            Debug.Log(gd.score6);
        }

        if(tipoScore==7)
```

```
{
    gd.score7 = 1;
    Debug.Log(gd.score7);
}

if(tipoScore==8)
{
    gd.score8 = 2;
    Debug.Log(gd.score8);
}

if(tipoScore==9)
{
    gd.score9 = 2;
    Debug.Log(gd.score9);
}

if(tipoScore==10)
{
    gd.score10 = 10;
    Debug.Log(gd.score10);
}

//////////////////////////////////////////

// Nivel 3
//////////////////////////////////////////
if(tipoScore==11)
{
    gd.score11 = 1;
    Debug.Log(gd.score11);
}

if(tipoScore==12)
{
    gd.score12 = 1;
    Debug.Log(gd.score12);
}

if(tipoScore==13)
{
    gd.score13 = 2;
    Debug.Log(gd.score13);
}

if(tipoScore==14)
{
    gd.score14 = 2;
    Debug.Log(gd.score14);
}

if(tipoScore==15)
{
    gd.score15 = 10;
    Debug.Log(gd.score15);
}

//////////////////////////////////////////

// Nivel 4
//////////////////////////////////////////
if(tipoScore==16)
{
    gd.score16 = 1;
    Debug.Log(gd.score16);
}

if(tipoScore==17)
{
    gd.score17 = 1;
    Debug.Log(gd.score17);
}

if(tipoScore==18)
{
    gd.score18 = 2;
    Debug.Log(gd.score18);
}

if(tipoScore==19)
{
    gd.score19 = 2;
    Debug.Log(gd.score19);
}

if(tipoScore==20)
{
    gd.score20 = 10;
    Debug.Log(gd.score20);
}

//////////////////////////////////////////
// Nivel 5
//////////////////////////////////////////
if(tipoScore==21)
{
    gd.score21 = 1;
    Debug.Log(gd.score21);
}

if(tipoScore==22)
{
    gd.score22 = 1;
    Debug.Log(gd.score22);
}

if(tipoScore==23)
{
    gd.score23 = 2;
    Debug.Log(gd.score23);
}

if(tipoScore==24)
{
    gd.score24 = 2;
    Debug.Log(gd.score24);
}

if(tipoScore==25)
{
    gd.score25 = 10;
    Debug.Log(gd.score25);
}

//////////////////////////////////////////

// Nivel 6
//////////////////////////////////////////
if(tipoScore==26)
{
    gd.score26 = 1;
    Debug.Log(gd.score26);
}

if(tipoScore==27)
{
    gd.score27 = 1;
    Debug.Log(gd.score27);
}

if(tipoScore==28)
{
    gd.score28 = 2;
    Debug.Log(gd.score28);
}

if(tipoScore==29)
{
    gd.score29 = 2;
    Debug.Log(gd.score29);
}
```

```
        if(tipoScore==30)
        {
            gd.score30 = 2;
            Debug.Log(gd.score30);
        }

        if(tipoScore==31)
        {
            gd.score31 = 2;
            Debug.Log(gd.score31);
        }

        if(tipoScore==32)
        {
            gd.score32 = 10;
            Debug.Log(gd.score32);
        }
        ////////////////////////////////////////


    }

    void CambioDeMensaje()
    {
        lblObjetivo.text = lblObjetivo.text.Replace("#323232F
F","#ff0000ff");
    }
```

```
    void PrenderImagen()
    {
        if (cont == 0)
        {
            imgObjetivo.SetActive(true);
            cont = 1;

        }
    }

    void ApagarImagen()
    {

        imgObjetivo.SetActive(false);

    }

    void PrenderEstructura()
    {
        estructuraVisor.SetActive(true);
    }

    void ApagarEstructura()
    {
        estructuraVisor.SetActive(false);
    }

}
```

Script Detector del jugador Test

Este script nos permite mostrar el canvas de la introducción al Test, además transporta al canvas de las preguntas del test mediante los botones

```
public string tituloNivel;
[SerializeField] Text titulo;
[SerializeField] GameObject juegoPrincipal;
[SerializeField] GameObject test;
[SerializeField] GameObject msgPanel;
[SerializeField] GameObject visor;
[SerializeField] Text mensaje;
[SerializeField] Button btnTest;
[SerializeField] GameObject area1;
[SerializeField] GameObject area2;
public Text lblObjetivo;
[SerializeField] GameObject estructuraVisor;
// Use this for initialization
void Start () {
    GetComponent<Collider>().isTrigger = true;
    msgPanel.SetActive (false);
}

public void MostrarMensaje(){


        msgPanel.SetActive (true);
    titulo.text = tituloNivel;
        visor.SetActive(false);
    area1.SetActive(false);
    area2.SetActive(false);
}

public void CerrarMensaje(){



        msgPanel.SetActive (false);

        visor.SetActive(true);
    area1.SetActive(true);
    area2.SetActive(true);
    CambioDeMensaje();
    ApagarEstructura();
    PrenderImagen();
    Invoke("ApagarImagen", 2f);
    Invoke("PrenderEstructura", 2f);
}


// Update is called once per frame
public void BtnTest(){
    SceneManager.LoadScene(2);
}

public void ActivarTest(){
    juegoPrincipal.SetActive(false);
    test.SetActive(true);
}

public void SalirTest(){
    juegoPrincipal.SetActive(true);
    test.SetActive(false);
}

void CambioDeMensaje()
{
    lblObjetivo.text = lblObjetivo.text.Replace("#323232F
F","#ff0000ff");
}

void PrenderImagen()
{
    if (cont == 0)
    {
        imgObjetivo.SetActive(true);
        cont = 1;

    }
}
```

```
    void ApagarImagen()
    {

        imgObjetivo.SetActive(false);

    }

    void PrenderEstructura()
    {
        estructuraVisor.SetActive(true);
    }

    void ApagarEstructura()
    {
        estructuraVisor.SetActive(false);
    }
}
```

Script para la Calificación del Test

Este script nos permite calificar las respuestas del Canvas del Test, además nos permite guardar los resultados dentro de juego

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class TestCharacter : MonoBehaviour {

    [SerializeField] GameObject juegoPrincipal;
    [SerializeField] GameObject test;
    public Toggle[] opcionRespuesta1;
    public Toggle[] opcionRespuesta2;
    public Toggle[] opcionRespuesta3;
    public Toggle[] opcionRespuesta4;
    public Toggle[] opcionRespuesta5;
    public Toggle[] opcionRespuesta6;
    int score =0;

    public int respuestaCorrecta1 = 0;
    public int respuestaCorrecta2 = 2;
    public int respuestaCorrecta3 = 1;
    public int respuestaCorrecta4 = 0;
    public int respuestaCorrecta5 = 2;
    public int respuestaCorrecta6 = 1;

    public Text lblNota;
    public Text lblObjetivo;
    void Start () {

    }

    void Update () {

    }


    public void CheckRespuestas(){
        bool correcta = true;
        score = 0;

        /*----------Pregunta 1----------
--*/
        for(int i=0;i<opcionRespuesta1.Length;i++)
        {
            if (opcionRespuesta1 [i].isOn)
            {
                if (i != respuestaCorrecta1)
                {
                    correcta = false;
                }
            }
        }

        if (correcta)
        {
            score++;
        }

        correcta = true;
        /*----------
*/


        /*----------Pregunta 2----------
--*/
        for(int i=0;i<opcionRespuesta2.Length;i++)
        {
            if (opcionRespuesta2 [i].isOn)
            {
                if (i != respuestaCorrecta2)
                {
                    correcta = false;
                }
            }
        }

        if (correcta)
        {
            score++;
        }

        correcta = true;
        /*----------
*/


        /*----------Pregunta 3----------
--*/
        for(int i=0;i<opcionRespuesta3.Length;i++)
        {
            if (opcionRespuesta3 [i].isOn)
            {
                if (i != respuestaCorrecta3)
                {
                    correcta = false;
                }
            }
        }

        if (correcta)
        {
            score++;
        }

        correcta = true;
        /*----------
*/


        /*----------Pregunta 4----------
--*/
        for(int i=0;i<opcionRespuesta4.Length;i++)
```

```
        {
            if (opcionRespuesta4 [i].isOn)
            {
                if (i != respuestaCorrecta4)
                {
                    correcta = false;
                }
            }
        }

        if (correcta)
        {
            score++;
        }

        correcta = true;
        /*-------------------------------------------------------------
*/


        /*-------------------------Pregunta 5-------------------------
--*/
        for(int i=0;i<opcionRespuesta5.Length;i++)
        {
            if (opcionRespuesta5 [i].isOn)
            {
                if (i != respuestaCorrecta5)
                {
                    correcta = false;
                }
            }
        }

        if (correcta)
        {
            score++;
        }

        correcta = true;
        /*-------------------------------------------------------------
*/


        /*-------------------------Pregunta 6-------------------------
--*/
        for(int i=0;i<opcionRespuesta6.Length;i++)
        {
            if (opcionRespuesta6 [i].isOn)
            {
                if (i != respuestaCorrecta6)
                {
                    correcta = false;
                }
            }
        }

        if (correcta)
        {
            score++;
        }

        correcta = true;
        /*-------------------------------------------------------------
*/
        lblNota.text = "" + score +"/6";
    }

    // Se utiliza cuando se cambia la escena
    public void CambiarEscena(){
        GenerarData gd = GenerarData.getInstance();
        gd.score20 = score;
        SceneManager.LoadScene(1);
    }

    //Se utiliza cuando se activa el objeto
    public void SalirTest(){

        GenerarData gd = GenerarData.getInstance();
        float temp = (10 * score) / 6;
        gd.score20 = temp;
        Debug.Log(gd.score20);
        juegoPrincipal.SetActive(true);
        test.SetActive(false);
        CalcularPorcentajeNota();
    }

    void CalcularPorcentajeNota()
    {
        float scoreParcial;
        scoreParcial = score / 6.0f;
        float porcenteje= 100*scoreParcial;
        lblObjetivo.text="Test Del Los Personajes: " +porcente
je.ToString("F2");
    }
}
```

Script para el Wizard

Este script nos permite manejar la información de las calificaciones de los Test, con el fin de mostrar mensajes que ayuden a la jugabilidad y cumplimiento de los objetivos del juego, al aparecer cada cierto tiempo en la interfaz del jugador.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class ControlMascota : MonoBehaviour {
    public float contTiempo=240.0f;
    public float contAux=0.0f;
     int cont=0;
    int cont2=0;

    [SerializeField] GameObject panelMascota;
    [SerializeField] GameObject panelObjetivos;
    [SerializeField] Text lblMensaje;
    [SerializeField] GameObject btnSi;
    [SerializeField] GameObject btnNo;
    public string[]  MensajeOtros;
    public string[] MensajeInicial;
    public string[] MensajeArea1;
    public string[] MensajeArea2;
    public string[] MensajeArea3;
    public string[] MensajeArea4;
    public string[] MensajeArea5;
    public string[] MensajeArea6;



    // Use this for initialization
    void Start () {
        //contAux = contTiempo;
```

```
 // AsignarInformacion();
   InvokeRepeating("ActivarMascota", 10f, 25f);


}

// Update is ca=lled once per frame
void Update () {

// contTiempo -= Time.deltaTime;
 //  ControlDeTiempoActivar();
}

void ControlDeTiempoActivar()
{
   if (contTiempo > 0)
   {
      DesactivarMascota();

   }
   else
   {

         ActivarMascota();
         Invoke("Reset", 5f);

   }

}



public void ActivarMascota()
{

   panelMascota.SetActive(true);
   ImprimirMensaje();
   panelObjetivos.SetActive(false);
   Invoke("DesactivarMascota", 15f);

}

public void DesactivarMascota(){

   panelMascota.SetActive(false);
   panelObjetivos.SetActive(true);

}

public void Reset(){
      contTiempo = contAux;
   cont = 1;

   if (cont2 == 0)
   {
      cont2 = 1;
   }
}




void ImprimirMensaje()
{
   lblMensaje.text="inicio";
   if (cont == 0)
   {
      MensajeInicio();
      //lblMensaje.text = "inicio";
      cont = 1;

   }
   else
   {

      if (cont2 == 0)
      {
        MensajeAyuda();
       // lblMensaje.text = "ayuda";
        cont2 = 1;
      }
      else
      {
         MensajeTest();
       // lblMensaje.text = "test";
        cont2 = 0;
      }
   }

}

void MensajeTest(){
   GenerarData gd = GenerarData.getInstance();

   btnSi.SetActive(false);
   btnNo.SetActive(false);
   //Para las pruebas
   //Area 1
   if (gd.score5 > 0f && gd.score5 < 6f)
   {
      lblMensaje.text = MensajeArea1[Random.Range(0,
MensajeArea1.Length)];
   }
   else
   {
      lblMensaje.text = "Todavía no has hecho la Prueba
1";
   }

   //Area 2
   if(gd.score10>0f && gd.score10<6f )
   {
      lblMensaje.text = MensajeArea2[Random.Range(0,
MensajeArea2.Length)];

   }
   else
   {
      lblMensaje.text = "Todavía no has hecho la Prueba
2";
   }

   //Area 3
   if(gd.score15>0f && gd.score15<6f )
   {
      lblMensaje.text = MensajeArea3[Random.Range(0,
MensajeArea3.Length)];

   }
   else
   {
      lblMensaje.text = "Todavía no has hecho la Prueba
3";
   }

   //Area 4
   if(gd.score20>0f && gd.score20<6f )
   {
      lblMensaje.text = MensajeArea4[Random.Range(0,
MensajeArea4.Length)];

   }
   else
   {
      lblMensaje.text = "Todavía no has hecho la Prueba
4";
   }

   //Area 5
   if(gd.score25>0f && gd.score25<6f )
   {
      lblMensaje.text = MensajeArea5[Random.Range(0,
```

```
MensajeArea5.Length)];

        }
        else
        {
            lblMensaje.text = "Todavía no has hecho la Prueba
5";
        }

        //Area 6
        if(gd.score32>0f && gd.score32<6f )
        {
            lblMensaje.text = MensajeArea6[Random.Range(0,
MensajeArea6.Length)];

        }
        else
        {
            lblMensaje.text = "Todavía no has hecho ninguna Pr
ueba";
        }
    }

    void MensajeInicio(){
        btnSi.SetActive(true);
        btnNo.SetActive(true);
        // Bienvenido a Unity City, Has utilizado antes Unity?
        lblMensaje.text = MensajeInicial[0];

        //Si
        //Te recomiendo que comiences con la creación de un T
erreno para crear tu terreno.
        // lblMensaje.text = MensajeInicial[1];

        //No

        //Te recomiendo que comiences visitando el parque, pa
ra ver que es Unity.
        // lblMensaje.text = MensajeInicial[2];

    }

    public void MensajeInicioSi(){
        btnSi.SetActive(false);
        btnNo.SetActive(false);
        lblMensaje.text = MensajeInicial[1];

    }
    public void MensajeInicioNo(){
        btnSi.SetActive(false);
        btnNo.SetActive(false);
        lblMensaje.text = MensajeInicial[2];

    }

    void MensajeAyuda(){
        btnSi.SetActive(false);
        btnNo.SetActive(false);
        lblMensaje.text = MensajeOtros[Random.Range(0, Me
nsajeOtros.Length)];

    }

}
```

Cada uno de los scrips anteriormente vistos son implementados en cada uno de sus objetos, excepto el ultimo, por razones de que siempre debe estar activo en todo el juego como se muestra en la Figura 31.

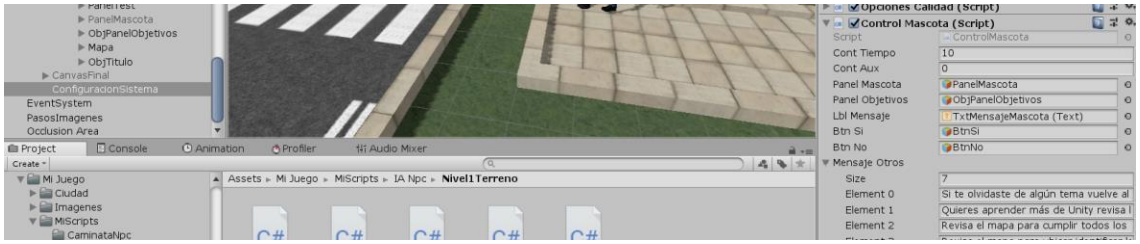


Figura 31: Pantalla de Implementación del Script del Wizard

Realizado por: Christian Viracocha, Año 2018

4.5.9. Creación de los Menús y Escenas

Para implementar las escenas del juego se debe tener en cuenta que el fin de las mismas, es la separación estructural se las secuencias, ya que existe una Escena Principal en donde se encuentra las opciones principales del juego y una Escena de Desarrollo donde ocurre todo el juego, por lo que en la Figura 32 se observan dos escenas principales del juego.

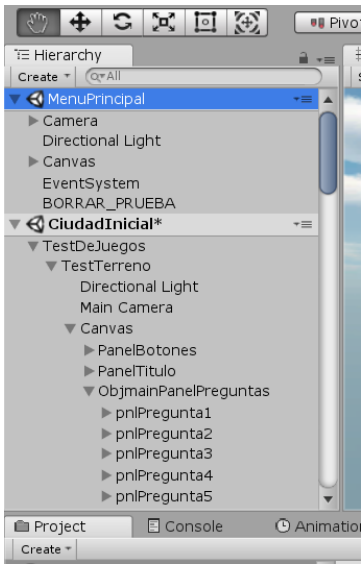


Figura 32: Pantalla de las Escenas del Juego

Realizado por: Christian Viracocha, Año 2018

Las escenas se crean desde la barra de herramientas, en la opción New Scena como se muestra en la Figura 33, es recomendable apagar la escena al dar clic derecho sobre la escena y seleccionar la opción Unload Scena para que al momento de desarrollar el juego no existan bug en las pruebas, ni tampoco fallas en las luces del juego.

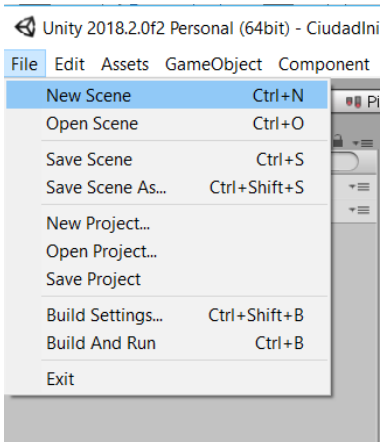


Figura 33: Pantalla de Creación de una Nueva Escena

Realizado por: Christian Viracocha, Año 2018

4.5.9.1. Creación del Canvas para Menús

Para comenzar a armar el menú principal se recomienda ver el video tutorial ¨**Tutorial Unity 5: Menús y Escenas | Mi Aprendizaje [#3]**¨ con el Link ¨ https://www.youtube.com/watch?v=-TobfKcKzhs¨ en donde se ve los aspectos más importantes al crear los menús y escenas principales, para el juego como se ve en la Figura 34.

Figura 34: Pantalla del Menú Principal

Realizado por: Christian Viracocha, Año 2018

Para implementar las opciones es recomendable observar este video ¨**SETTINGS MENU in Unity**¨ con el link ¨https://www.youtube.com/watch?v=YOaYQrN1oYQ&t=14s¨ donde nos muestra las principales configuraciones para el menú, como se muestra en la Figura 35.

Figura 35: Pantalla de la Interfaz del Video Tutorial

Realizado por: Christian Viracocha, Año 2018

4.5.9.2. Creación del Script para los Menús

Los Script que se usaron para la construcción del menú principal y de pausa, manejan los cambios de escena y la configuración básica del juego, basado en el cambio de la calidad de imágenes, y de sonido.

Para el menú principal se usó el siguiente script para el cambio de escenas y uso de la escena asíncrona:

```
 using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;

public class MenuPrincipal : MonoBehaviour {

    //public string Escena ;
    [SerializeField] GameObject MenuPrincipalInterfaz;
    [SerializeField] GameObject MenuOpciones;
    [SerializeField] GameObject MenuCreditos;
    [SerializeField] GameObject MenuAyuda;
    [SerializeField] GameObject Carga;

    private bool loadScene = false;
```

```
public string LoadingSceneName;
[SerializeField] Text loadingText;
[SerializeField] Slider sliderBar;

// Use this for initialization
void Start () {
   sliderBar.gameObject.SetActive(false);
}

// Update is called once per frame
void Update () {

}

public void Empezar () {
   Carga.active=true;
   CargarEscena();
}

public void Opciones () {
   MenuPrincipalInterfaz.active=false;
   MenuOpciones.active=true;
   MenuCreditos.active=false;
   MenuAyuda.active=false;
}

public void Creditos () {
   MenuPrincipalInterfaz.active=false;
   MenuOpciones.active=false;
   MenuCreditos.active=true;
   MenuAyuda.active=false;
}

public void Ayuda () {
   MenuPrincipalInterfaz.active=false;
   MenuOpciones.active=false;
   MenuCreditos.active=false;
   MenuAyuda.active=true;
}

public void Salir () {
   Application.Quit();

}


public void CargarEscena () {

      // If the player has pressed the space bar and a new sce
ne is not loading yet...


      // ...set the loadScene boolean to true to prevent loadin
g a new scene more than once...
      loadScene = true;

      //Visible Slider Progress bar
      sliderBar.gameObject.SetActive(true);

      // ...change the instruction text to read "Loading..."
      loadingText.text = "Loading...";

      // ...and start a coroutine that will load the desired scen
e.
      StartCoroutine(LoadNewScene(LoadingSceneName));



   }

   // The coroutine runs on its own at the same time as Upda
te() and takes an integer indicating which scene to load.
   IEnumerator LoadNewScene(string sceneName) {

      // Start an asynchronous operation to load the scene th
at was passed to the LoadNewScene coroutine.
      AsyncOperation async = SceneManager.LoadSceneAs
ync(sceneName);

      // While the asynchronous operation to load the new sc
ene is not yet complete, continue waiting until it's done.
      while (!async.isDone)
      {
         float progress = Mathf.Clamp01(async.progress / 0.9
f);
         sliderBar.value = progress;
         loadingText.text = progress * 100f + "%";
         print(progress.ToString("F2"));
         yield return null;

      }

   }

}
```

Para el manejo de las escenas Asíncronas se recomienda ver el siguiente video ¨**How to make a Loading Bar & Load Scene Asynchronously in Unity**¨ con el link ¨https://www.youtube.com/watch?v=K0Ni7bmZ3h8¨ con el fin de mejorar el uso de las escenas de carga.

Para el manejo del menú de las opciones es necesario el uso de los siguientes scripts:

Manejo de la Calidad de la imagen

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class OpcionesCalidad : MonoBehaviour {

   public Button btn1;
   public Button btn2;
   public Button btn3;

   // Use this for initialization
   void Start () {
      btn1.interactable = true;
      btn2.interactable = false;
      btn3.interactable = true;
   }

   // Update is called once per frame
   void Update () {
```

```
    }

    public void GraficasAlta () {
        QualitySettings.currentLevel= QualityLevel.Fantastic;

        btn1.interactable  =  true;
        btn2.interactable  =  true;
        btn3.interactable = false;
    }

    public void GraficasMedia () {
        QualitySettings.currentLevel= QualityLevel.Good;
        btn1.interactable = true;
        btn2.interactable = false;
        btn3.interactable = true;
    }

    public void GraficasBaja () {
        QualitySettings.currentLevel= QualityLevel.Fastest;
        btn1.interactable = false;
        btn2.interactable = true;
        btn3.interactable = true;
    }
}
```

Para el manejo de opciones de sonidos solo se usó el Script de sonido que se encuentra ligado al volumen de la cámara, como se muestra en la Figura 36, el jugador puede manipular el sonido con un Slider en el menú de opciones, el script que se uso es:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.Audio;

public class OpcionSonido : MonoBehaviour {

    public AudioMixer audioMixer;

    public void SetVolume (float volumen)
    {
        AudioListener.volume = volumen;
    }

}
```

Figura 36: Pantalla de la Camara Principal para el Script de Audio

Realizado por: Christian Viracocha, Año 2018

4.5.9.3. Implementación de los scripts con Menús

Para la implementación de los scripts que se crearon anteriormente, se debe emplear los canvas que se crearon al inicio de este proceso, en donde se crea un objeto vacío al final de la jerarquía del proyecto, con el fin de almacenar todos los scripts de configuración como se muestra en la Figura 37.

Figura 37: Pantalla del Objeto Vacío Para almacenar los Scripts

Realizado por: Christian Viracocha, Año 2018

Para que luego, el objeto vacío en donde se encuentran los scripts, sea arrastrado a los botones del canvas, en donde se seleccionara las funciones respectivas como se muestra en la Figura 38.

Figura 38: Pantalla del Canvas (Implementación de funciones a los Botones)

Realizado por: Christian Viracocha, Año 2018

4.5.10. Control de Información para visualizar la información

Dentro del juego no es recomendable mantener todos los objetos y elementos activados ya que puede acusar incongruencias en el manejo de la información, por eso en los scripts activan y desactivan objetos durante el juego, de acuerdo con lo que se está realizando, por eso en la Figura 39 de la jerarquía del juego no están todos activados.

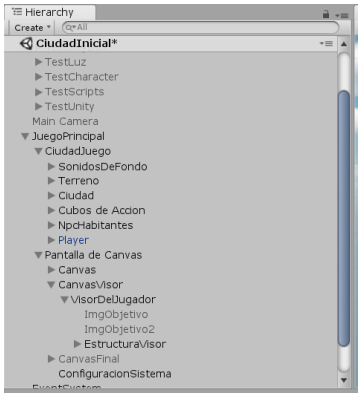


Figura 39: Pantalla de la Jerarquía de los Objetos del Juego

Realizado por: Christian Viracocha, Año 2018

Durante el desarrollo del juego, se tomó en cuenta que al tener activados todos objetos que poseen información como texto o imágenes, causaban conflictos al ejecutarse el juego, por lo que se creó objetos en forma rectangular con los triggers para identificar al jugador cuando este se encontraba en sectores específicos del juego, con la finalidad de activar o desactivar los objetos de información, como se muestra en la Figura 40, para solucionar esos errores.

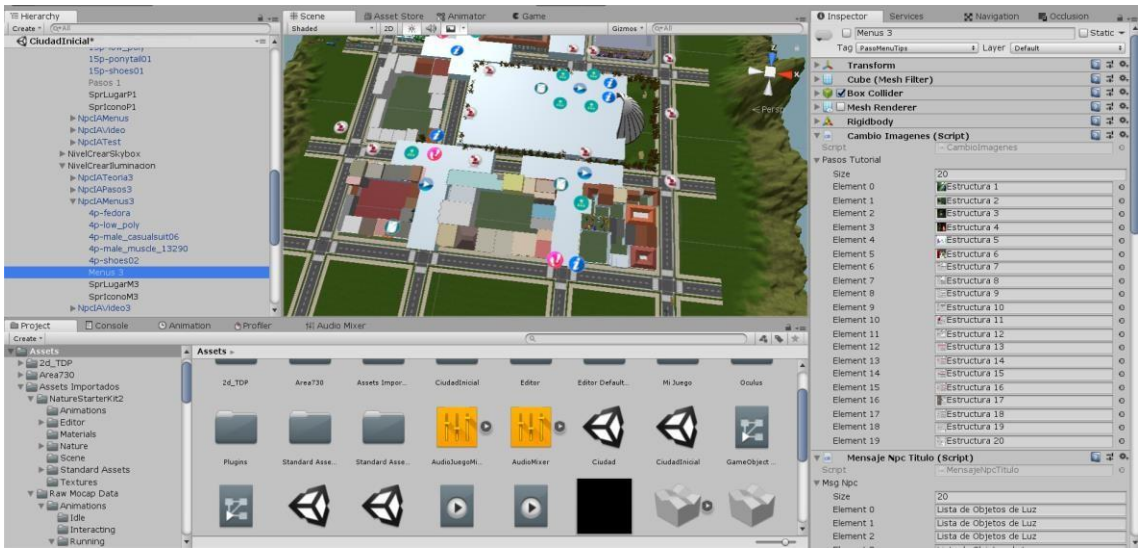


Figura 40: Pantalla de las áreas del Juego

Realizado por: Christian Viracocha, Año 2018

5. Creación del BUILD del Juego

Para crear el ejecutable del juego, tenemos que acceder a la barra de herramientas, en la opción File y seleccionar la opción Build Settings en donde elegiremos las escenas y la plataforma del juego, en donde se va a jugar ,como se muestra en la Figura 41.

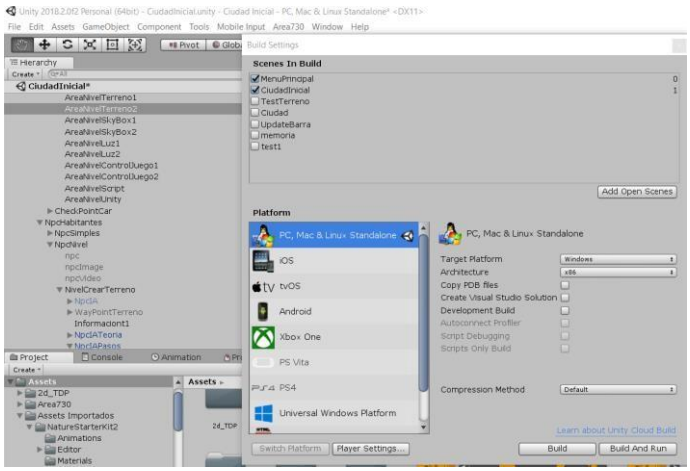


Figura 41: Pantalla para el Build del Juego

Realizado por: Christian Viracocha, Año 2018

Cuando se da clic en build, es recomendable guardar el ejecutable del juego en una carpeta distinta al proyecto principal.

6. Creación del Instalador del Juego

Para crear el instalador de cualquier juego para Unity 3D es recomendable ver el siguiente video ¨ **M7-2016: 18. Crear instalador para un juego Unity3d**¨ con el link ¨ https://www.youtube.com/watch?v=04PlTMf8ihs¨ con el fin de establecer las reglas básicas para realizar un ejecutable viable para el juego, debido a que si no se siguen los pasos exactamente, va a salir error en la ejecución del instalador, para lo cual se usa el programa ¨ Inno Setup Compiler¨ que se muestra en la Figura 42.

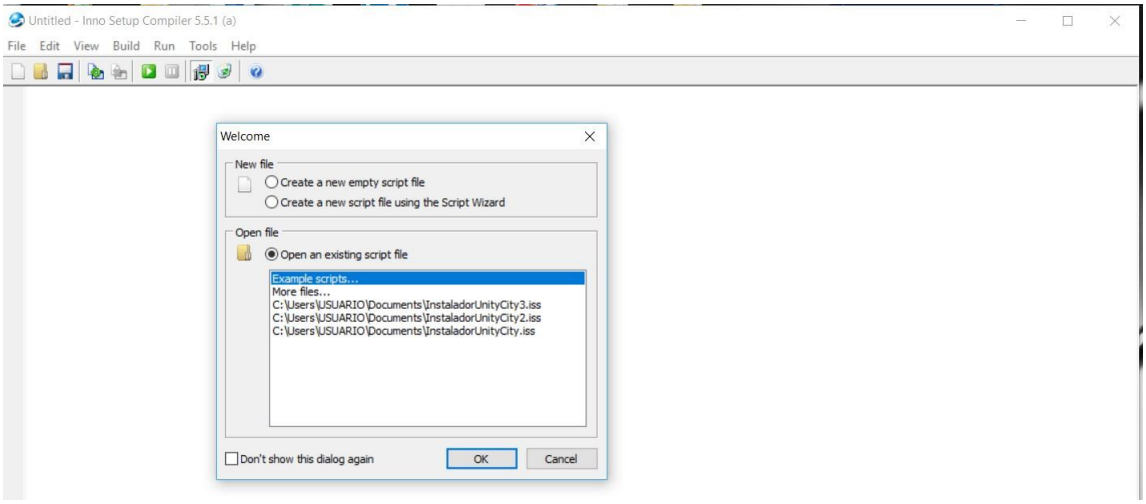


Figura 41: Pantalla del Programa Inno Setup Compiler

Realizado por: Christian Viracocha, Año 2018

7. Recomendaciones

- Al crear el código del juego, es recomendable usar solo un tipo de lenguaje nativo, como el C# para evitar complicaciones en el futuro, por cuanto la mayoría de los tutoriales crean código en C# y muy pocos usan JavaScript.
- Al comenzar a desarrollar algún proyecto en Unity 3D, se debe verificar la versión de Unity por el problema en el manejo de las versiones, porque si se desea pasar el juego a una versión más actual, el proyecto va a presentar errores por el código o herramientas que pueden estar obsoletas.
- Se recomienda que los nuevos desarrolladores que utilicen por primera vez Unity, deben recopilar la suficiente información del proyecto que desean realizar, es decir que si desean implementar escenas asíncronas, primero deben consultar si es posible o no su implementación.
- Los objetos en 3D que Unity utiliza para sus proyectos, deben reducirse sus polígonos para que el sistema del compilador del juego no se sobrecargue y vuelva al juego lento.

- Es recomendable para el diseño de los objetos en 3D usar Maya por cuanto posee las herramientas necesarias para minimizar polígonos.
- Cuando se crea un Script para múltiples objetos dentro del proyecto, se debe tener en cuenta que, según su ubicación, estos pueden causar Bugs dentro del juego, por lo que se recomienda mantenerlos desactivados hasta el momento necesario para su uso.
- Al momento de implementar la Inteligencia artificial, se debe documentarse bien y probarlos fuera del juego principal, con el fin de evitar cualquier contratiempo. Una vez probado ahí, se recomienda usarlo en el proyecto principal con múltiples objetos, para identificar posibles fallas o bugs en el juego.

Anexo D. Manual de Usuario

MANUAL DE USUARIO DEL SERIOUS GAMES ´UNITY CITY´

Autor: Christian Viracocha

Contenido

1. Introducción

Este manual proporciona las principales características del Serious Games ¨Unity City¨ como también su funcionamiento para el uso correcto durante su jugabilidad para todas las personas que quieran aprender a manejar el motor de videojuegos Unity 3D, y saber cómo usarlo para sacar el mayor provecho del juego, ya que tiene como objetivo compartir los diferentes contenidos que ayudaran a introducir al jugador a la creación de videojuegos.

2. Requerimientos

Los requerimientos mínimos para que el Serious Games funcione correctamente, son los siguientes:

- Sistema operativo Windows 8 o superiores.
- Procesador Intel Core I5 o superior
- Memoria RAM de 1 Gb a 2 Gb.
- Tarjeta de video de 1Gb.
- Tarjeta de sonido Compatible DirecX9 o superior
- Espacio de 1Gb

3. Descripción de la aplicación

El Serious Games ¨Unity City¨ fue desarrollado para el departamento de Tutorías de la Facultad de Informática y Electrónica, con la finalidad de introducir a los estudiantes dentro del mundo del desarrollado de videojuegos, el cual consiste en un Tutorial de Introducción a Unity 3D, en el cual se enseña los primeros pasos para crear tu propio videojuego, dentro de una ciudad virtual interactiva, la cual esta dividida en sectores con diferentes temas.

Cada sector del videojuego se compone de la teoría del tema a tratarse, los pasos que se tiene que seguir, las ventanas necesarias para su uso, un video tutorial el cual muestra como usarlos y para finalizar, un pequeño test, el cual medirá el grado de aprendizaje que obtuvo el usuario en el juego.

El usuario siempre va a poder observar el Score que va ganando durante el juego, va a existir una barra de progreso, que le indicara lo que le falta para completar el juego, va a existir un mini mapa para ubicarse en la ciudad y poder identificar a los personajes no jugables, que son los objetivos del juego además va a observar todos los objetivos que se de hacer durante todo el juego, y estos cambiaran de color cuando se hayan finalizado.

Cuando el jugador haya finalizado el juego, automáticamente aparecerá un video agradeciendo por haber jugado, y se regresara a la ciudad para que el jugador pueda seguir explorando la ciudad virtual ya, que existen por todo el juego Tips, que contribuyen al uso de Unity 3D.

4. Instalación e Inicio del Juego

Una vez que se descargue el instalador del juego, se dará doble clic en el archivo ejecutable, y se dará clic en siguiente en la pantalla del instalador, hasta terminar el proceso, una vez terminado el proceso de instalación se visualizara la pantalla para acceder al juego, es recomendable poner la resolución ¨1280x720¨, se dará clic en el botón play para iniciar el juego, como se muestra en la Figura 1.

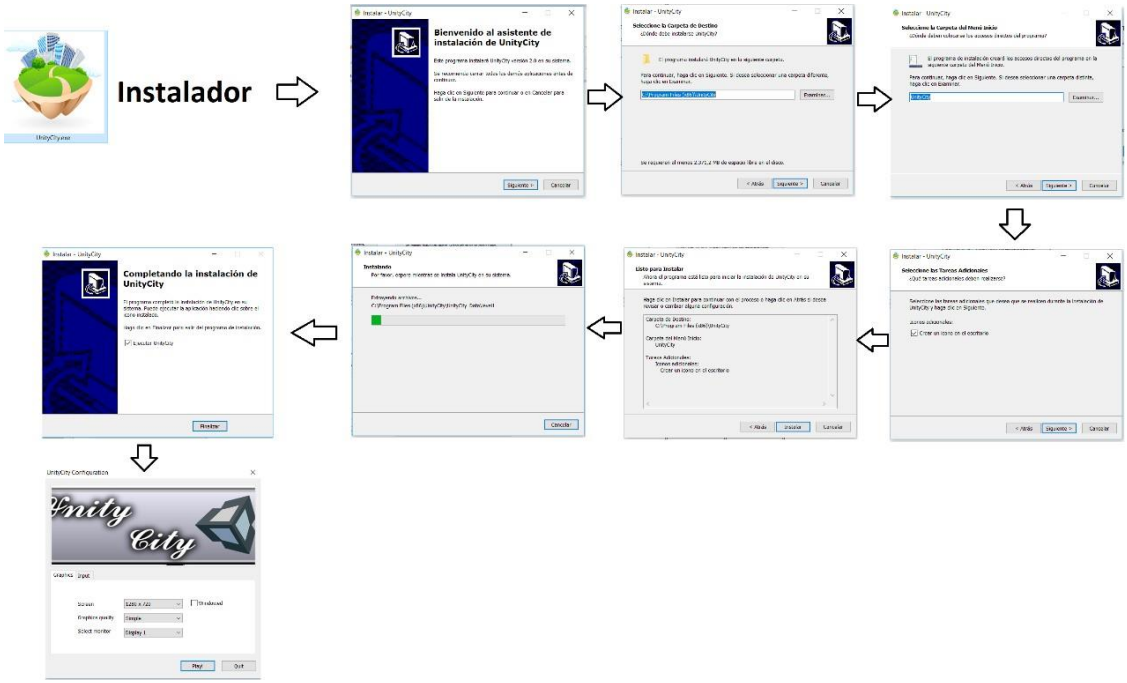


Figura 1 Proceso de instalación (Aplicación de Unity City)

Realizado por: Christian Viracocha, Año 2018

5. Navegación por el Serious Game

Al ingresar al Juego se visualizará la pantalla del Menú Principal que tiene diversas opciones para la configuración del juego y a para acceder al mismo.

5.1. Pantalla del Menú principal

La pantalla que se muestra al inicio del juego es la del Menú Principal, como se muestra en la figura 2, la cual consta con 5 botones principales:

- **Empezar**: Inicial el juego de forma automática.
- **Opciones**: Muestra la ventana de opciones para el juego para su configuración interna, como la calidad de imagen y el volumen del juego
- **Créditos**: Muestra la información general del juego.
- **Ayuda**: Muestra la información referente a la jugabilidad y características del juego.
- **Salir**: Permite cerrar el juego.

Figura 2 Pantalla del Menú Principal del Juego

Realizado por: Christian Viracocha, Año 2018

5.2. Pantalla del Menú Opciones del Menú Principal y del Menú de Pausa

Al seleccionar el botón de Opciones del Menú Principal y de Pausa, este desplegara la ventana que se muestra en la Figura 3, la cual consta de 2 configuraciones principales:

- **Gráficos**: Esta opción nos permite cambiar la calidad de las gráficas del juego.
- **Sonido**: Esta opción nos permite cambiar el volumen del juego.

Figura 3 Pantalla del Menú de Opciones

Realizado por: Christian Viracocha, Año 2018

5.3. Pantalla de Ayuda

Al seleccionar el botón de Ayuda del Menú Principal, este desplegara la ventana que se muestra en la Figura 4, en donde se visualiza una cadena de imágenes que muestra como jugar y las características del juego.

Figura 4 Pantalla del Menú de Ayuda

Realizado por: Christian Viracocha, Año 2018

5.4. Pantalla de Créditos

Al seleccionar el botón de Créditos del Menú Principal, se visualizará una ventana que se muestra en la Figura 5, en donde se visualiza la información básica del juego y de sus creadores, también consta de un botón que le permite regresar al menú principal.

Figura 5 Pantalla del Menú de Créditos

Realizado por: Christian Viracocha, Año 2018

5.5. El Visor del Jugador

Al ingresar al juego se visualizará el visor principal, que tiene la finalidad de mostrar en todo momento el progreso del jugador y a la vez se visualizaran los objetivos que han cumplido con un pequeño ayudante que ayudara al jugador al dar concejos de como jugar como se muestra en la Figura 6.

Figura 6 Pantalla del Visor del jugador

Realizado por: Christian Viracocha, Año 2018

En la Figura 7 nos permite visualizar el Score del Juego, el cual va aumentando cuado finalizan los objetivos del juego, también se observa las calificaciones de los Test que se van realizando en cada sector del juego.

Figura 7 Pantalla del Score y Calificación de los Tests

Realizado por: Christian Viracocha, Año 2018

En la Figura 8 podemos observar la barra de progreso que se va llenando de acuerdo al cumplimiento de los objetivos hasta llegar al final en el cual se visualizara el mensaje de Agradecimiento por haber jugado.

Figura 8 Pantalla de la barra de progreso

Realizado por: Christian Viracocha, Año 2018

En la Figura 9 se visual el panel de objetivos los cuales van cambiando de color de negro a rojo al momento de finalizar ese objetivo, con la finalidad de tener una lista de todos los objetivos restantes para acabar el juego.

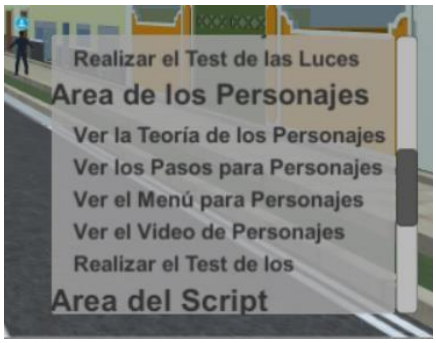


Figura 9 Pantalla del Panel de Objetivos

Realizado por: Christian Viracocha, Año 2018

En la Figura 10 podemos observar el panel donde se visualiza la mascota del juego, esta aparece cada 3 minutos, la cual mostrara información referente al juego, tips, y de acuerdo a las calificaciones de los test otorgara mensajes y avisos para acabar los test.

Figura 10 Pantalla del Ayudante del Juego

Realizado por: Christian Viracocha, Año 2018

5.6. La Ventana Teoría

Cuando se interactúa con los personajes no jugables del juego van a mostrar diferentes pantallas con información referente al manejo de Unity, y uno de ellos es el personaje de Teoría que muestra la teoría del elemento que se está revisando en ese momento como se muestra en la Figura 11.

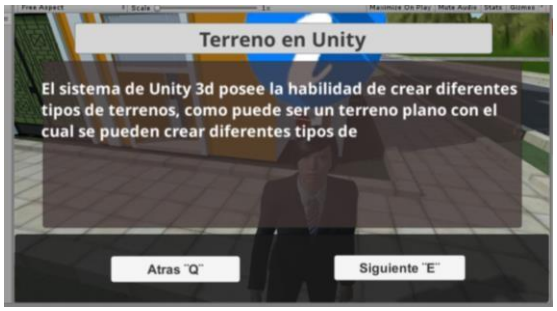


Figura 11 Pantalla del Teoría

Realizado por: Christian Viracocha, Año 2018

5.7. La ventana de Pasos y Menús

En esta ventana podemos ver los procedimientos y menús necesarios para crear diferentes aspectos del juego usando Unity gracias a los personajes no jugables que se encuentran en el juego como se muestra en la Figura 12, ya que estos son interactivos con el jugador.

Figura 12 Pantalla del Procedimientos

Realizado por: Christian Viracocha, Año 2018

5.8. La ventana Video

Esta ventana provee de un video tutorial que es compartido por los personajes no jugables dentro del juego, el cual funciona al clic en los botones de control o simplemente tecleando las letras: F para reproducir el video, R para pausar el Juego y la letra G para el Stop del video como se muestra en la Figura 13.

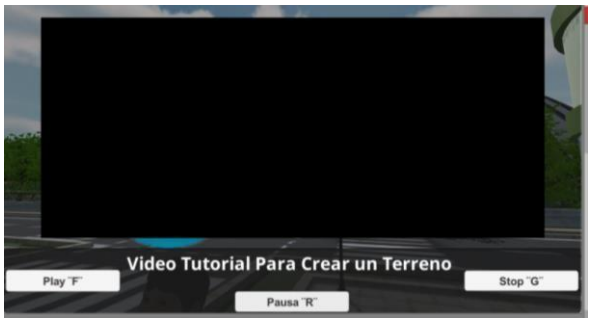


Figura 13 Pantalla de Video

Realizado por: Christian Viracocha, Año 2018

5.9. La ventana Test

Igual a las ventanas anteriores de información, esta funciona mediante la interacción con los personajes no jugables dentro del juego, ya que es una pantalla de acceso al Test del juego, como se muestra en la Figura 14.

Figura 14 Pantalla Para acceder al Test

Realizado por: Christian Viracocha, Año 2018

Cuando se accede al Test se cambiará de pantalla, a una que posee las preguntas necesarias para medir el aprendizaje que se ganó en el área jugada, el cual consta de 2 botones como se muestra en la Figura 15:

- **Calificar**: Este Botón permite calificar y evaluar el test, la calificación se visualiza a la inferior izquierda.
- **Salir**: Permite regresar al juego.

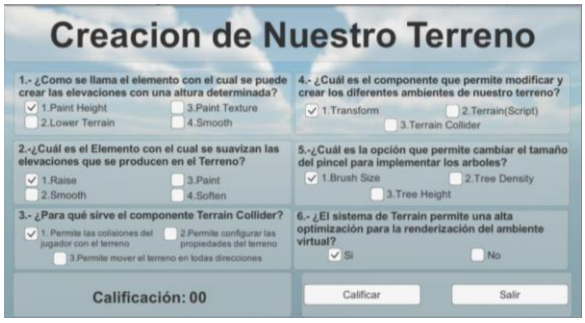


Figura 15 Pantalla del Test

Realizado por: Christian Viracocha, Año 2018

5.10. Pantalla de Pausa

La pantalla que se muestra en la Figura 16, aparece al teclear la tecla P, el cual es el menú de Pausa del juego, como se muestra en la Figura 2, la cual consta con 5 botones principales:

- **Continuar**: Inicial el juego de forma automática.
- **Mapa**: Muestra el mapa del juego.
- **Opciones**: Muestra la ventana de opciones para el juego para su configuración interna, como la calidad de imagen y el volumen del juego

- **Objetivos**: Muestra la información de cuáles son los objetivos y el mapa de la ciudad virtual.
- **Salir**: Permite cerrar el juego.

Figura 16 Pantalla de Pausa

Realizado por: Christian Viracocha, Año 2018

5.11. Pantalla del Mapa

La pantalla que se muestra en la Figura 17 es el mapa de la ciudad virtual del juego con la simbología que se maneja dentro del juego para identificar a los personajes no jugables, ya que estos poseen un icono encima de ellos los cuales los identifican.

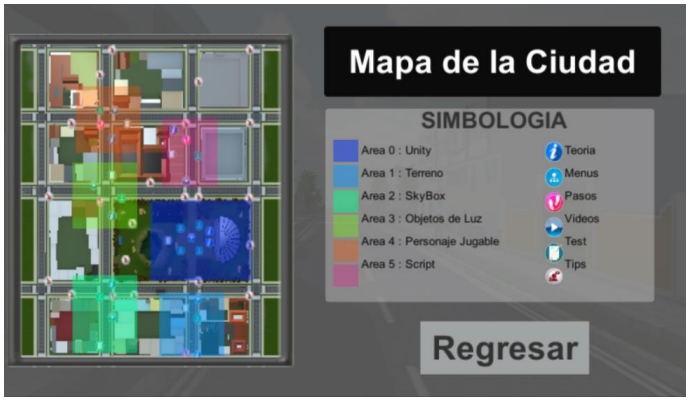


Figura 17 Pantalla de Mapa de la Ciudad

Realizado por: Christian Viracocha, Año 2018

5.12. Pantalla del Objetivos

La pantalla que se muestra en la Figura 18 es el mapa de la ciudad virtual del juego con los objetivos que se tiene que cumplir para finalizar el juego.

Figura 18 Pantalla de Objetivos

Realizado por: Christian Viracocha, Año 2018

5.13. Pantalla de Salir

Para salir del juego es necesario la confirmación del usuario, con la finalidad de que no exista un accidente al momento de seleccionar las opciones del menú de pausa, como se muestra en la Figura 19.

Figura 19 Pantalla de Confirmación de salida

Realizado por: Christian Viracocha, Año 2018

6. Tipos de Npc o Personajes no Jugables

Dentro del juego existen diferentes tipos de personajes no jugables los cuales se encuentran divididos en:

Npc´s Simples: Estos personajes son aquellos que sirven para contribuir al ambiente del juego, entre los cuales tenemos a las personas que están caminando por la ciudad o los autos que están por toda la zona.

Npc´s Especiales: Los Npc especiales poseen un icono en su parte superior los cuales les identifican como objetivos del juego, estos se clasifican en:

 Teoría

 Procedimientos

 Menús

 Videos

 Test

 Tips

7. Áreas del juego

La ciudad virtual en el que habita el jugador se encuentra dividido en sectores como muestra la Figura 22, los cuales contiene las diferentes áreas en donde se encuentra ubicado el material educativo, los cuales serán repartidos por los personajes no jugables, estos sectores se dividen en:

0. Sector de Unity 3D
1. Sector para la construcción del terreno
2. Sector para la implementación del Skybox
3. Sector para la implementación de los objetos de luz
4. Sector para la creación de personajes jugables
5. Sector para la construcción de Scrips

Figura 22 Sectores de la ciudad Virtual

Realizado por: Christian Viracocha, Año 2018