Universidad de las Ciencias Informáticas

Centro de Entornos Interactivos 3D, VERTEX, Facultad 4

Videojuego El mundo de Wumpus basado en técnicas de Inteligencia Artificial

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

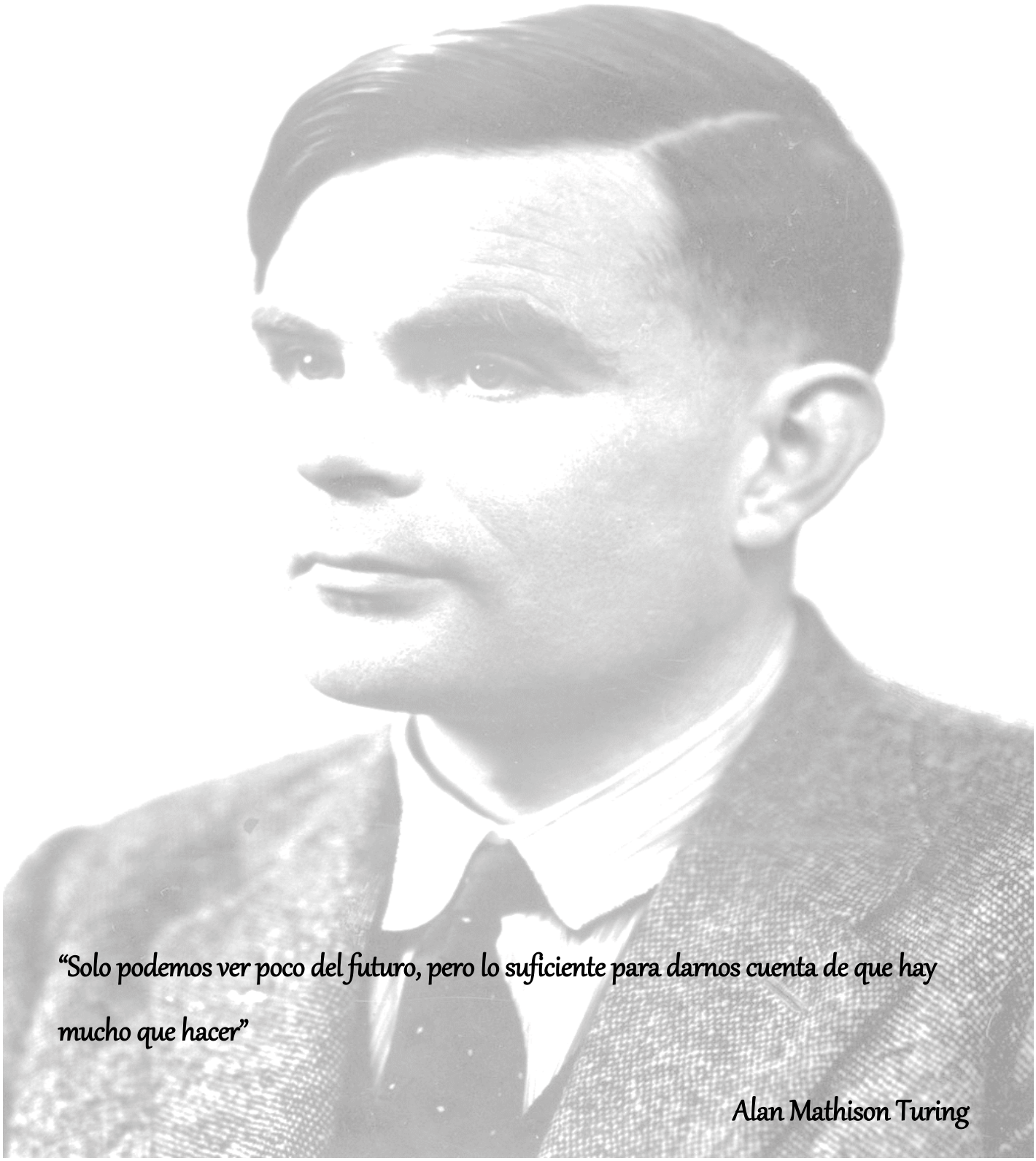
Autor:

Cinthia Cuza Soca

Tutor:

Msc.Yuniesky Coca Bergolla

La Habana, 2019



"Solo podemos ver poco del futuro, pero lo suficiente para darnos cuenta de que hay mucho que hacer"

Alan Mathison Turing

Dedicatoria

A todos aquellos que de una forma u otra me apoyaron en mi carrera. A mis padres por su amor y por creer siempre en que puedo superarme a mí misma.

Agradecimientos

Agradezco a la vida el haberme dado un padre que me enseñó desde pequeña que el verdadero significado de la felicidad está en nosotros, en lo que somos capaces de hacer con nuestra mente. Gracias por tus consejos, por tus exigencias y por tu perseverancia.

A mi madre quien a pesar de todo siempre está a mi lado y me ha demostrado que ama a sus hijos más que a nada en el mundo. Gracias por darme ese toqué práctico de ver la vida, de ser realista, proactiva y reactiva ante las circunstancias.

A mi hermano, el mayor admirador de mi trabajo. Gracias por sentirte orgulloso de tu hermana y por motivarme en la realización de este videojuego.

A mis abuelos, en especial a mi abuela quien me ha ayudado en todo lo que ha podido a lo largo de mi carrera.

A mi pareja por apoyarme desde que estamos juntos, gracias porque siento que he madurado y aprendido mucho a tu lado, gracias por tus consejos y tu amor.

Agradecer de forma especial a mi tutor. Gracias por todos los consejos dados que me han ayudado mucho para culminar el trabajo de diploma de forma satisfactoria.

A mis amigos Leyna, Irian, Juan Luis, Milena, Andy y otros tantos que ya forman parte de mi vida. Gracias por sus consejos y por los momentos que hemos compartido juntos.

A los desarrolladores del centro VERTEX del laboratorio 305 que han estado ahí cada vez que los he necesitado.

A los profesores que me han impartido clase, a todos los respetos y admiro muchísimo.

A todos los que de una forma u otra me han ayudado y apoyado durante estos cinco años: **¡Muchas gracias!**

Declaración de autoría

Declaro por este medio que yo, Cinthia Cuza Soca soy el autor principal del trabajo de diploma para optar por el título de ingeniero en ciencias informáticas: “Videojuego El mundo de Wumpus basado en técnicas de Inteligencia Artificial”, y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales con carácter exclusivo.

Para que así conste firmo la presente declaración de autoría en La Habana, a los ____ días del mes de _____ del año _____.

_____ _____

Autor: Cinthia Cuza Soca **Tutor:** Msc.Yuniesky Coca Bergolla

Resumen

El campo de la Inteligencia Artificial se nutre de varias ramas de la ciencia para ofrecer nuevas soluciones aplicables a disímiles áreas de la vida actual. Una de estas áreas es la de los videojuegos. El presente trabajo surge a partir de la necesidad de incentivar la aplicación de diversas técnicas de Inteligencia Artificial en el desarrollo de los videojuegos realizados en el Centro de Entornos Interactivos 3D VERTEX, perteneciente a la Universidad de las Ciencias Informáticas en Cuba. Se ha identificado poco uso de estas técnicas en los videojuegos que se realizan y falta de conocimiento sobre su empleo. El objetivo de la investigación es desarrollar un videojuego que contribuya a la aplicación de diversas técnicas de Inteligencia Artificial. En la presente investigación se realiza un análisis de la aplicación de diversas técnicas de la Inteligencia Artificial a los videojuegos. Se define para guiar el análisis y diseño el marco de trabajo ingenieril para el proceso de desarrollo de videojuegos, el motor de videojuego Unity y como lenguaje de programación C#. La aplicación ofrecerá la opción de jugar y un Modo Agente que permite visualizar a un agente inteligente que, utilizando las tareas generales de representación del conocimiento, razonamiento, tratamiento de la incertidumbre y aprendizaje, logra alcanzar los objetivos del juego, similar a como lo haría un jugador humano. El producto obtenido podrá ser utilizado por los desarrolladores del centro como ejemplo de su uso de la Inteligencia Artificial mediante la consulta a su código fuente.

Palabras claves: Inteligencia Artificial, videojuegos.

Índice General

Índice de figuras

Índice de tablas

Introducción

Los juegos fueron creados o llevados a cabo desde sus inicios con el propósito de brindar relajación y entretenimiento. Los videojuegos son juegos electrónicos con el mismo fin que los tradicionales, en el que el jugador puede interactuar con un dispositivo electrónico. Con el paso del tiempo y la creación de determinados videojuegos, así como el mal uso de los mismos, la perspectiva de los videojuegos ha tomado un curso equivocado. Suelen verse como un elemento dañino y obsesivo, que afecta la sicología y las emociones. Estos criterios tan excesivamente generalizadores llegan a ser erróneos al analizar la gran cantidad de juegos enfocados a la instrucción e introducción de valores positivos en los jugadores.

Una de las clasificaciones de los videojuegos son los juegos serios, los cuales tienen un fin educativo, de entrenamiento o de información, sin dejar de un lado el entretenimiento que debe proporcionar (1). Los juegos serios han tratado de recuperar una tarea antigua nombrada ludificación, la cual tiene como objetivo convertir una tarea que sería aburrida en una mucho más placentera, pero con el mismo fin de aprender (2).

Dentro de los juegos serios se encuentran los juegos educativos. Se ha demostrado que estos ayudan a la resolución creativa de problemas, al desarrollo del pensamiento reflexivo y del razonamiento, de la capacidad de atención y la memoria, de las habilidades necesarias para resolver conflictos y al desarrollo de la capacidad de superación. Se ha comprobado que su utilización en las aulas propicia la socialización, equidad, igualdad, participación y progreso del estudiante. En la actualidad, la inclusión de videojuegos en las aulas se ha hecho una realidad, tanto es así que se ha convertido en un elemento asociado a las tecnologías educativas (3).

En la búsqueda de mayor calidad, realismo, libertad e interactividad con el jugador, los videojuegos se han aliado con una disciplina tan potente y en constante desarrollo como lo es la Inteligencia Artificial (IA). Esta disciplina intenta entender las capacidades de procesamiento de información de la mente humana para explicar y modelar sistemas

inteligentes. Busca dotar al computador de capacidades propias de la inteligencia humana como son la percepción, el razonamiento y la toma de decisiones.

En 1980 se incluyó por primera vez la Inteligencia Artificial como asignatura electiva del plan de carrera de Computador Científico. Luego se fue incorporando a otros planes de carreras relacionados con la computación con la intención de que los estudiantes pudieran identificar técnicas y métodos de la Teoría de Sistemas Inteligentes necesarios para resolver problemas de procesamiento de información. Los contenidos que abarcó esta asignatura estaban vinculados con el Aprendizaje Automático, Redes Neuronales, Algoritmos Genéticos, Sistemas Inteligentes Autónomos, Sistemas Expertos, entre otros (4).

La IA, para su formación y consolidación, ha heredado técnicas de otras disciplinas como la filosofía, la sicología, la lingüística, la matemática, la ingeniería de computación, entre otras. Su utilidad se ha visto reflejada en áreas de control de sistemas, planificación automática, reconocimiento de escritura, del habla y de patrones; se ve presente en la economía, medicina, ingeniería y en la industria de videojuegos logrando un gran auge y ampliando sus fronteras de conocimiento. En los videojuegos se utilizan diversas técnicas de IA como la búsqueda de caminos, las máquinas de estados finitos, agentes inteligentes, el tratamiento de la incertidumbre, entre otras que le proporcionan al juego y a los no jugadores (NPC) características propias del pensamiento humano (5).

La Universidad de las Ciencias Informáticas[1] (UCI) es una institución con la misión de formar profesionales calificados en la rama de la informática y la producción de aplicaciones y servicios informáticos mediante el vínculo docencia-investigación-producción. La UCI cuenta con varios centros de desarrollo, uno de ellos es el Centro de Entornos Interactivos 3D VERTEX, el cual se encarga de la creación de productos y servicios relacionados con los Entornos Interactivos 3D y contiene entre sus líneas de desarrollo principales la realización de videojuegos enfocados tanto al disfrute del usuario como al aprendizaje. VERTEX ha desarrollado videojuegos en los que se hace uso de

[1] https://www.uci.cu

algunas técnicas de inteligencia artificial como las máquinas de estados, ejemplos de estos videojuegos son: Especies Invasoras, Kuba Kart y Coliseum.

A pesar de los avances alcanzados por el centro VERTEX, son pocos los videojuegos que realiza donde se evidencia el uso de la IA, factor que podría aportar una mayor calidad a los productos desarrollados. Muchas veces se emplean técnicas de IA de forma inconsciente, pues los programadores las comprenden como parte de la implementación del videojuego. Esto se debe además por el poco conocimiento que existe acerca de qué técnicas y cómo se aplican en el desarrollo de videojuegos. No obstante, es importante destacar que los programadores no cuentan con un software que sirva de base y ejemplo de cómo se aplica IA en videojuegos o que facilite el trabajo con estas técnicas.

Un juego clásico para la enseñanza de la IA es El mundo de Wumpus, el cual se utiliza en libros de esta disciplina como *Artificial Intelligence. A modern Approach* (6), los autores lo emplean como ejemplo en varios de sus capítulos. Contar con una implementación básica de este juego que permita el estudio y la modificación de su código, y que cumpla con las especificaciones de un software libre, brindará una herramienta interesante y útil para los desarrolladores del centro VERTEX y que podría ser utilizada, además, para la enseñanza en las clases de esta rama. A continuación, se resume la situación presente en el centro VERTEX explicada anteriormente:

- Falta de conocimiento acerca de la aplicación de técnicas de IA en el proceso de desarrollo de videojuegos.
- Poco uso de técnicas de IA en los videojuegos que se desarrollan en el centro VERTEX.
- Uso inconsciente de técnicas de IA en los videojuegos.
- Ausencia de herramientas que sirvan de ejemplo del uso de las diversas técnicas de IA que se pueden aplicar en los videojuegos.

Teniendo en cuenta la situación antes descrita, se presenta el siguiente **problema de investigación**: ¿Cómo contribuir a la aplicación de técnicas de IA en el desarrollo de videojuegos? Se define como **objeto de estudio** el proceso de desarrollo de videojuegos, quedando enmarcado en el **campo de acción** aplicación de técnicas de

Inteligencia Artificial en el proceso de desarrollo de videojuegos. Se declara como **objetivo de la investigación**: Desarrollar un videojuego que contribuya a la aplicación de diversas técnicas de Inteligencia Artificial en los videojuegos producidos por el centro VERTEX. Como **resultado** se obtendrá el videojuego El mundo de Wumpus, el cual mostrará, a modo de ejemplo, varias técnicas de Inteligencia Artificial.

Para dar cumplimiento al objetivo trazado se definen como tareas de investigación las siguientes:

1. Análisis y síntesis de los elementos relacionados con el objeto de estudio y el campo: concepto de videojuego y elementos a tener en cuenta para el diseño de videojuegos, uso de técnicas de IA en videojuegos, aplicación de IA en los videojuegos realizados por el centro VERTEX y descripción del juego El mundo de Wumpus.
2. Identificación de las herramientas y método ingenieril para el desarrollo de software a utilizar en la elaboración de la propuesta de solución.
3. Elaboración de artefactos según el método ingenieril para la definición y descripción de la propuesta de solución.
4. Desarrollo de la propuesta de solución al problema planteado teniendo en cuenta los resultados de las tareas previas.
5. Validación del cumplimiento de los requerimientos de la propuesta de solución mediante la aplicación de pruebas de software.

Para la realización de las tareas de investigación se emplearon los métodos de investigación científica que se describen a continuación:

- Métodos teóricos:
  - Histórico-Lógico: método que se utiliza para el estudio de los aspectos de interés a tratar en la presente investigación acerca del desarrollo de videojuegos que utilizan técnicas de IA. Se estudió la definición de videojuegos, la estructura que presentan, los tipos de videojuegos y las técnicas de IA presentes en los videojuegos.

- Analítico-Sintético: empleado con el propósito de estudiar la bibliografía necesaria para este tema, de la cual se extrajeron y sintetizaron elementos relacionados con los videojuegos, la IA en videojuegos y El mundo de Wumpus.
- Modelación: permite confeccionar el videojuego El mundo de Wumpus a través de la modelación del diagrama de los paquetes de mecanismos, arquitectura del videojuego, los diagramas de clases de la solución y los diagramas de estado para representar el comportamiento de los mecanismos. Todos estos elementos pertenecen al método ingenieril que se utiliza.

- Métodos empíricos:
  - Consulta bibliográfica: empleado en la elaboración del marco teórico de la investigación.
  - Pruebas: utilizado para comprobar que la propuesta elaborada satisface el objetivo presentado, así como su correcto funcionamiento.

Para abarcar el contenido a abordar el documento contiene los siguientes capítulos:

- **Capítulo 1: Fundamentación teórica:** capítulo que contiene los elementos relacionados con el objeto de estudio y el campo de acción. Se define el método ingenieril de desarrollo de software a utilizar y las herramientas y tecnologías para desarrollar la solución propuesta.
- **Capítulo 2: Caracterización y diseño del videojuego El mundo de Wumpus:** capítulo que abarca la descripción de la propuesta de solución y los artefactos ingenieriles elaborados (diseño del videojuego, especificación de mecanismos y paquete de mecanismos, diagramas de clase, diagramas de estado para cada mecanismo y definición de requisitos no funcionales).
- **Capítulo 3: Implementación y pruebas del videojuego El mundo de Wumpus:** capítulo que incluye el estándar de codificación utilizado y la implementación de la propuesta de solución, teniendo en cuenta la aplicación de las técnicas de IA. Se realiza una representación de los mecanismos del videojuego a nivel de componente y contiene además los resultados de las pruebas de validación realizadas al videojuego.

Capítulo 1

Fundamentación teórica

En el presente capítulo se exponen los conceptos relacionados con la investigación y conocimientos vinculados con estos conceptos que sirvieron para su elaboración, los cuales son: los videojuegos y sus géneros, así como la IA, su utilización en los videojuegos a partir de determinadas técnicas y el empleo de la misma en los videojuegos desarrollados por el centro VERTEX. Se da a conocer el método ingenieril que guía el desarrollo de la investigación, así como las herramientas y tecnologías a utilizar para desarrollar la solución propuesta.

1.1 Videojuegos

Los videojuegos constituyen una parte importante de la industria del entretenimiento pues influyen en el desarrollo de habilidades sicológicas, cognitivas, físicas y sociales (7). Comenzaron desde la creación de los ordenadores con el único fin de brindarle entretenimiento al jugador. Actualmente se encuentran en disímiles ambientes sociales debido a que han evolucionado de tal forma que su objetivo va más allá de alcanzar el entretenimiento del jugador. Un ejemplo de esto son los videojuegos educativos, los cuales pueden ser usados como medios en las clases para la enseñanza.

Ha sido complicado realizar una definición exacta del término videojuego, pues cada autor tiene un concepto distinto del mismo. Stahl enumera cinco tipos de juegos: los juegos de entretenimiento, donde todos los resultados positivos derivados del juego se obtienen durante el juego, los juegos educativos que aportan beneficios de aprendizaje a largo plazo, juegos experimentales para probar hipótesis, juegos de investigación para obtener material empírico y juegos operativos que ayudan a la toma de decisiones. Es por esta cuantía de propósitos que para realizar una definición de videojuego no es recomendable tener en cuenta el tipo de videojuego. Por otro lado, los videojuegos están determinados por reglas, a pesar de que el jugador pueda creer que tiene total libertad en realidad se está rigiendo por las reglas del juego. Se puede decir entonces que un

videojuego es un juego electrónico que recrea una situación ficticia en la cual el jugador puede interactuar, sometido a determinadas reglas, para alcanzar un objetivo específico (8).

1.1.1 Jugabilidad

Debido a que los videojuegos se enfocan principalmente en el disfrute del jugador, es necesario saber la experiencia de los mismos al jugar un determinado videojuego. En este aspecto es cuando toma lugar el concepto de jugabilidad, la cual abarca los factores que satisfacen a los jugadores mientras juegan. Para caracterizar las experiencias de un jugador ante el videojuego se necesitan determinados atributos, estos son (7):

- Satisfacción: cuánto el juego satisface al jugador.
- Aprendizaje: cuán rápido el jugador comprende las mecánicas del juego.
- Eficiencia y efectividad: cuánto tiempo y recursos requiere el jugador para alcanzar sus objetivos.
- Inmersión: cuánto logra el jugador introducirse en el juego.
- Motivación: cuán motivado está el jugador y los elementos del juego que ayudan a su motivación.
- Emoción: sentimientos que transmite el juego.
- Socialización: cómo se siente el jugador al jugar en compañía y los elementos del juego que influyen en este aspecto.

1.1.2 Diseño del videojuego

Para crear un videojuego se debe tener en cuenta los elementos que lo conforman, estos son los elementos formales, vinculados a la estructura del juego, los dramáticos, relacionados con la jugabilidad y los elementos dinámicos, donde se especifican los mecanismos del juego (9):

- Elementos formales: jugadores, objetivos, acciones a ejecutar para alcanzar los objetivos (procedimientos), reglas, conflicto[2], elementos presentes en el

[2] Conflicto: derivado de situaciones que producen las reglas y los procedimientos.

videojuego que ayudan a alcanzar el objetivo (recursos), límites y la condición para saber si el juego se ganó o no (resultados).

- Elementos dramáticos: desafíos, el juego, las condiciones que dan inicio al juego (premisa), historia[3] y arco dramático[4].
- Elementos dinámicos: vinculan a los elementos dramáticos y formales y dependen de cada juego. Se puede decir que son los recursos presentes en el juego que puede utilizar el jugador para realizar determinada acción y con esto lograr el objetivo o los objetivos propuestos. Ejemplo de elementos dinámicos es que el jugador obtenga poderes que lo ayuden a luchar y así ganar determinado nivel, o que el jugador pueda disparar, saltar o que se puedan mover en determinada dirección las fichas de un tablero.

1.1.3 Tipos de videojuegos

Los géneros de los videojuegos son una forma de clasificarlos en función fundamentalmente de su mecánica de juego. No existe una clasificación universal de los videojuegos, esto se hace en dependencia de diversos criterios como son: el tipo de interacción jugador-juego, el tipo de representación gráfica utilizada, las reglas que lo definen y su argumentación y discurso narrativo. Es importante destacar que actualmente es muy complejo definir el género de un videojuego pues los juegos que se crean hoy en día son cada vez más avanzados llegando a mezclarse en un mismo juego varios géneros. A continuación, se muestran algunas de sus clasificaciones (10):

- Aventura: Juegos caracterizados por una historia interactiva donde se combina la narración, la exploración y la resolución de puzles. Ejemplos: *Broken Sword*, *Monkey Island* y *The Whispered Word*.
- Deportivo: Juego donde se simula un deporte real. Ejemplos: *FIFA*, *Pro Evolution Soccer*, *NBA* y Tenis.

[3] La historia le proporciona un sentido al juego y lo ubica en un lugar determinado donde se desarrollará. Sabiendo la historia el jugador tendrá un propósito por el cual jugar y se verá motivado a alcanzar el objetivo del videojuego.
[4] Arco dramático: son los momentos de tensión del videojuego, este elemento se puede representar mediante una curva de tensión donde se verá en cada parte del videojuego el nivel de tensión del mismo.

- Disparo o shooter: Juegos basados en la utilización de armas. Ejemplo: *Call of Duty*, *Borderlands* y *Duke Nuken 3D*.
- Lucha: Juegos donde se simula una lucha cuerpo a cuerpo entre dos o más oponentes. Ejemplos: *Street Fighter*, *Soul Calibur*, *Tekken* y *Boxing*.
- Puzles o rompecabezas: Juegos basados en identificar y relacionar formas geométricas. Requieren un razonamiento lógico y espacial por parte del jugador que le ayuda a completar los distintos retos. Ejemplos: *Tetris*, *Bejeweled*, *Super Puzzle Fighter* y *Lemmings*.
- Rol: Se denominan juegos de rol pues el jugador podrá tener uno o más roles en el juego. Permite interactuar con el entorno desarrollando aventuras y adquiriendo o mejorando habilidades. Ejemplos: las zagas de Zelda o *Final Fantasy*, *Dragon Quest* y *Mass Effect*.
- Simulación: Juegos que simulan situaciones, acciones o funcionamientos propios de la vida real. Un tipo de juego de simulación son los simuladores de vehículos los cuales recrean la experiencia de manejar un vehículo. Ejemplos: *SimCity* y *Ace Combat*.
- Estrategia: Juegos donde el jugador debe decidir una estrategia para lograr el o los objetivos del juego. La estrategia se basa en tomar decisiones consecutivas con un orden y sentido lógico que conduce a ganar el juego. Ejemplos: Empire: *Total War*, Civilización, *The Sims* y *Theme Hospital*.
- Carreras: Juegos que consisten básicamente en conducir vehículos y donde el jugador se ve involucrado en una carrera. Transmiten una sensación de velocidad y espíritu competitivo en el jugador. Ejemplos: *Need For Speed*, *Burnout*, *Forza Horizon* y *Mario Kart*.
- Arcade: Juegos de acción carentes de un hilo argumental elaborado y con una curva de aprendizaje bastante baja. Presentan una sucesión de metas que repercuten a alcanzar el objetivo final, por lo que presentan varios niveles de rápida completitud. Ejemplos: *Pacman* y *Donkey Con*.
- Juegos de plataforma: Juegos que consisten en atravesar diferentes plataformas mientras se superan obstáculos, estos obstáculos incluyen a los propios enemigos. Ejemplos: *Super Mario Bros*., *Sonic the Hedgehog* y *Crash Bandicoot*.

El videojuego El mundo de Wumpus presenta los géneros de disparo y estrategia. El jugador podrá dispararle a su oponente, pero también debe realizar una estrategia basada en decisiones que deberá tomar en cada momento sobre a donde ir y qué hacer para no morir y lograr los objetivos definidos.

1.1.4 Videojuegos serios

Los videojuegos no solo se enfocan en el entretenimiento del jugador, son muchos los autores que expresan la importancia de los videojuegos en el aprendizaje y su papel como vehículo en el desarrollo integral. Algunas habilidades que se adquieren solo jugando son el desarrollo de la coordinación ojo-mano, mayor agudeza visual, rapidez de reacción y capacidad de atención a múltiples estímulos. Permiten mejorar la socialización, ayudan a la toma de decisiones, a la voluntad de tomar riesgos y lo mejor de todo es que todas estas habilidades se adquieren de una forma entretenida y amena (11).

Teniendo en cuenta lo abordado es que surge un nuevo tipo de videojuego: Los videojuegos serios. Estos pueden abarcar una o varias de las clasificaciones abordadas anteriormente con la diferencia de que están enfocados en el aprendizaje del jugador. Los videojuegos serios se utilizan en distintos ámbitos como la educación, salud, marketing, el ámbito militar, entre otros. En el marco educativo su uso ayuda a dominar contenidos y adquirir determinadas habilidades, también permite ejercitar la fantasía y creatividad, favorece la repetición ayudando a asimilar cada vez más rápido los contenidos y mejora la atención. En el área de la salud se utilizan los videojuegos para la rehabilitación de pacientes ya sea guiándolos a realizar determinados ejercicios, desarrollándoles de una forma secuencial habilidades perdidas como por ejemplo la visión, o tratando trastornos del cerebro. En cuanto a la publicidad, se han creado juegos con el fin de darle propaganda a una determinada marca, producto, organización o incluso una idea. En el aspecto militar los videojuegos se crean para ayudar a las prácticas militares recreando ambientes reales de forma tal que puedan conocer el terreno y sus particularidades antes de enfrentarse a una batalla real en el mismo.

El videojuego El mundo de Wumpus es un videojuego serio pues está creado no solo para el entretenimiento del jugador sino para que sirva de herramienta en la aplicación de diversas técnicas de IA en el desarrollo de videojuegos y puede ser usado como ejemplo en las clases relacionadas con esta asignatura y por los desarrolladores del centro VERTEX.

1.2 IA para videojuegos

El surgimiento de la IA vino dado por la idea de resolver problemas tanto fáciles como tan complejos que para el ser humano fuera imposible de resolver o no pudiera darle solución en un tiempo razonable. La IA es aplicable a cualquier ámbito de la actividad intelectual humana, lo cual viene dado de su propia naturaleza y propósito (12). Basándose en la definición de IA que realizan Peter Norvig y Stuart Russell en su libro *Artificial Intelligence: A Modern Approach* (6), la IA incluye el diseño de agentes capaces de percibir su entorno, procesar los datos percibidos y llevar a cabo acciones de forma autónoma a partir de dichas percepciones.

Con el pasar de los años los videojuegos han mejorado en aspectos como gráfico, variedad, libertad del jugador y realismo. Este último ha sido de gran importancia para ampliar las investigaciones de IA donde se persigue la realización de juegos mejores y más inteligentes. La relación entre la IA y los videojuegos es doble pues, por un lado, los videojuegos son un mercado en constante evolución que encuentra en las técnicas de IA un factor diferenciador y, por otro lado, los videojuegos suponen una fuente inagotable de nuevos problemas que ponen a prueba los límites de la IA, favoreciendo la búsqueda de nuevas soluciones (13).

La IA brinda a las computadoras la habilidad de simular el razonamiento humano, de aprender a través de la ejercitación, permite la construcción de conocimiento y comunicarse con un lenguaje natural. En el caso de los videojuegos no solo imitan estas habilidades si no también el comportamiento del mundo real, aportando así un realismo superior al videojuego.

A partir de 1950 se comienza la realización de juegos como el ajedrez, las damas, Go, ente otros más que han logrado una IA consolidada al punto de ganarle a los campeones mundiales de estos juegos de mesa. Estos juegos han sido un reto para la IA debido a

que incluyen un nivel de dificultad alto, los movimientos que puede ejecutar el jugador son de más de 50, lo cual conlleva a un árbol de búsqueda de cientos de nodos o más. Además, se ve presente, como en el mundo real, la necesidad de tomar decisiones y que estas sean óptimas (6).

1.2.1 Técnicas de IA en videojuegos

Históricamente los videojuegos han utilizado diversas técnicas y estrategias para lograr un comportamiento inteligente. Las técnicas de la IA hacen referencia a diferentes campos de investigación y de desarrollo de aplicación de la IA. Inicialmente se utilizaban "trampas", es decir, los personajes del juego tenían información que el jugador no conocía. Con el devenir del tiempo y el desarrollo en esta rama se han ido incorporando técnicas mucho más sofisticadas. Algunas de las estrategias y técnicas más usadas son (13):

- Comportamientos de locomoción: persiguen lograr tareas básicas de movimiento en el entorno. Uno de los comportamientos más utilizados es el de capturar y huir. Es muy sencillo pues persigue solo el objetivo de capturar al jugador y huir de él. Primeramente, la IA debe decidir si está en condiciones de perseguir al jugador y luego lleva a cabo la acción mediante una estrategia que se trace.
- Búsqueda de caminos: es una técnica clásica de IA. se utiliza para determinar el camino a seguir de un punto a otro teniendo en cuenta las características del escenario.
- Redes neuronales: sigue el modelo de las redes neuronales biológicas pues se compone de entradas a la red, salidas correspondientes y neuronas que conectan las entradas y salidas.
- Máquinas de estados finitos: son sistemas formados por un conjunto de diferentes estados y las transiciones entre cada uno de ellos producidas por eventos.
- Agentes inteligentes: sistema capaz de percibir su entorno y actuar en consecuencia. Su comportamiento se determina por la percepción, planificación y actuación.

- Toma de decisiones: a partir de determinadas condiciones el sistema debe tomar la mejor decisión para lograr el objetivo del juego. Para esto se debe introducir determinadas sentencias en cada caso que podría ocurrir dentro del mundo del videojuego.

Estas y varias otras técnicas específicas utilizadas en el desarrollo de videojuegos tributan a los 4 núcleos generales de la Inteligencia Artificial:

- Representación del conocimiento: el sistema debe tener de forma organizada una representación del conocimiento, lo cual incluye reglas del juego, límites, recursos y cómo usarlos, condiciones para hacer o lograr algo, los actores involucrados en el juego, etcétera.
- Tratamiento de incertidumbre: se utiliza cuando no se dispone de todo el conocimiento de lo que sucede. Las acciones a realizar dependen de tomar una decisión a partir del conocimiento que se tiene, por lo tanto, el sistema debe buscar la forma de tratar con esa incertidumbre.
- Razonamiento: es el proceso que ayuda a la toma de decisión y consiste en deducir información a partir de la ya existente, la cual debe ser verídica.
- Aprendizaje: consiste en que el sistema aprenda qué es lo correcto que debe hacer a partir de experiencias pasadas. El sistema aprende a partir de su interacción con un ambiente y las consecuencias de las acciones realizadas.

En el videojuego El mundo de Wumpus se utilizarán como técnicas específicas de IA las máquinas de estados finitos y los comportamientos de locomoción, además contará con varias estrategias que tributan a los 4 núcleos de la Inteligencia Artificial. El agente deberá representar el conocimiento que tiene de su entorno para razonar bajo condiciones de incertidumbre y aprender.

1.3 Aplicación de la IA en los videojuegos desarrollados en VERTEX

El centro de Entornos Interactivos 3D VERTEX presenta entre sus líneas de desarrollo la realización de videojuegos. Según datos consultados por el centro, tres de los videojuegos realizados han implementado comportamientos inteligentes, estos son:

- Kuba Kart: videojuego de carrera donde el jugador deberá competir en distintas pistas contra varios oponentes para obtener el primer lugar. Los oponentes llegarán a la meta guiados por una serie de *way points* que le indicarán el camino a seguir. Para evitar salirse de la pista cuentan con varios *Raycast* que les indican cuando se desvían. El juego presenta además un mecanismo para los poderes que se encuentran a lo largo de la carrera que tanto el jugador como los no jugadores pueden obtener. Si el jugador se halla entre los últimos lugares el poder que obtendrá será bastante potente y lo contrario sucede con los que se encuentran ganando la carrera.
- Especies invasoras: es un videojuego de estrategia que tiene como objetivo recuperar todo el terreno de islas Casabe y desplazar las especies invasoras del territorio que han ocupado. En el juego se encuentran las casas que ocupa el jugador, terrenos neutrales y casas ocupadas por el enemigo. El enemigo tendrá el mismo objetivo del jugador, expandir su territorio, por lo que enviará unidades a los terrenos neutrales y a las casas del jugador para luchar.
- Coliseum: videojuego de lucha donde el jugador deberá enfrentarse a varios enemigos con el fin de proteger a la tierra de una futura invasión inter-dimensional. Para esto se implementaron dos niveles de IA, el primero consiste en que el oponente perseguirá y atacará al jugador. El segundo nivel tiene el mismo objetivo que el primero, pero tendrá en cuenta el nivel de vida. Si el jugador tiene mayor nivel de vida el oponente deberá buscar primero una gema de vida y luego luchar. Se está trabajando actualmente con el desarrollo de un tercer nivel de IA en el que varios enemigos realizarán ataque en conjunto con un plan determinado para acabar de forma más rápida con el jugador. También los no jugadores cuentan con una máquina de estado para perseguir, atacar o buscar gema de vida.

El centro VERTEX ha desarrollado gran cantidad de videojuegos, sin embargo, pocos cuentan con mecanismos que presentan IA. Se evidencian comportamientos inteligentes, pero no una realización sólida de las técnicas de IA para videojuegos. El uso de técnicas de IA brindará mayor calidad, realismo y garantizará que el juego exija un mayor esfuerzo del jugador para ganarlo. Ejemplo de esto es el éxito que ha tenido el

videojuego Coliseum el cual hasta el momento es en el que se ha implementado de una mejor forma la IA.

1.4 El mundo de Wumpus

Uno de los métodos para la enseñanza de la IA es la realización de problemas donde se apliquen técnicas de IA para resolverlos. Ejemplo de esto es El mundo de Wumpus en el cual se emplean conceptos de IA como agentes inteligentes, algoritmo de búsqueda, tratamiento de incertidumbre, aprendizaje y representación del conocimiento. En varias publicaciones se muestra dicho juego de distintas formas, con elementos nuevos o diferentes, pero la lógica del juego en sí, así como el uso de la IA en el mismo, es similar en todas. Es por esto que se dará una explicación general del juego para su entendimiento:

El mundo de Wumpus consiste en un tablero formado por cuadrículas, las cuales podrán contener al jugador, el Wumpus, el tesoro, precipicios, brisa, hedor y la entrada y salida. El objetivo principal del agente inteligente es obtener el oro y salir vivo (6). El diseño del tablero puede realizarse con las dimensiones y la colocación de los elementos en las cuadrículas que se desee, excepto el hedor que debe ubicarse en las casillas adyacentes al Wumpus y la brisa en las adyacentes a los precipicios. El agente podrá moverse hacia arriba, abajo o hacia los lados y también podrá disparar. Muere al caer en un precipicio o si cae en manos del Wumpus. El agente debe memorizar cada información que perciba del entorno, lo cual forma parte de su conocimiento, esto será vital para llegar a lograr su objetivo (14).

El videojuego a desarrollar está basado en el juego el Mundo de Wumpus, es un entorno en el que se muestra la forma de actuar de un agente basado en conocimiento. El problema del Mundo de Wumpus es uno de los métodos más utilizados para la docencia sobre IA pues abarca a partir de un ejemplo sencillo aspectos tan importantes como el razonamiento, el aprendizaje, el tratamiento de la incertidumbre y la representación del conocimiento.

1.5 Metodologías para el desarrollo de software

En el proceso de desarrollo de software se debe asegurar la calidad del producto, debe atender las necesidades requeridas y cumplir con los requisitos que los usuarios demandan. Debido a la complejidad de incorporar los estándares de calidad solicitados por los diversos usuarios, surgen las metodologías de desarrollo de software. Estas se aplicarán acorde al contexto y dominio de la aplicación a desarrollar. Metodología de desarrollo de software no es más que el estudio y determinación del método más adecuado para el desarrollo de un software en específico (15). En Ingeniería de Software es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo.

Las metodologías de desarrollo de software están divididas en dos grupos: las tradicionales y las ágiles. Las primeras buscan imponer disciplina al proceso de desarrollo de software para que sea predecible y eficiente. Las metodologías ágiles surgen luego como reacción a las metodologías existentes con el fin de disminuir la burocracia que implicaba ejecutar estas metodologías tradicionales; tienen como características que son adaptativas y orientadas a las personas (16).

1.5.1 Método ingenieril para el desarrollo de videojuegos

El desarrollo de un videojuego requiere conocer ciertas disciplinas y diversas ciencias. Un videojuego es un tipo especial de software, por lo que la estructura del mismo se debe llevar a cabo a partir de un esquema básico único. En los videojuegos los requisitos funcionales serán llamados mecanismos, estos conforman las partes del producto y permiten la interacción videojuego-jugador para desencadenar, durante el proceso, acciones de entrada o salida. Los elementos básicos que incluye un mecanismo son: objetos, propiedades, comportamientos y relaciones. En el diseño se deberá tener en cuenta los elementos formales y dramáticos para crear no solo un producto sino elementos dinámicos para el entretenimiento del jugador. Dentro de los elementos formales se encuentran los jugadores, los objetos, los procedimientos, reglas, recursos, conflicto, fronteras y resultado. Dentro de los dramáticos están el desafío, juego, premisa, caracteres, historia y arco dramático.

A continuación, se enuncian algunas de las metodologías o marcos de trabajo usadas para el desarrollo de videojuegos (17):

- Cascada: diseño perfecto y completo antes de comenzar a implementar. Sus requisitos son el diseño, la implementación, la verificación y el mantenimiento. Como desventaja se tiene que no responde bien ante los cambios y presenta problemas de comunicación e interpretación.
- SCRUM: metodología ágil basada en compromiso, enfoque, sinceridad, respeto y coraje. Las actividades que lo definen son planificación iteración, ejecución iteración, inspección y adaptación. Como desventaja se hace difícil definir la descripción de las funcionalidades de forma sencilla, es decir, no tan amplia ni específica.
- Programación XP: presenta un enfoque iterativo y se centra en productos de trabajo como las historias de usuario. Incluye las actividades de exploración, planificación de la entrega, iteración, producción, mantenimientos y muerte del proyecto. Su desventaja es que está pensado especialmente para programadores sin contemplar otros roles y, además, su aplicación es difícil en equipos de desarrollo de gran cantidad de personas.
- SUM: es una adaptación de SCRUM y XP para lograr videojuegos de calidad en tiempo y costo. Las fases que presenta son: concepto, planificación, elaboración, beta y cierre, además de la gestión de riesgo realizada a todo lo largo del proyecto. Su uso es recomendable en equipos pequeños, proyectos cortos, equipo multidisciplinario y alta participación del cliente.

Debido a las particularidades que presenta el desarrollo de videojuegos las actividades ingenieriles que se ejecutan generarán resultados diferentes al de los softwares tradicionales. Las especificaciones o entregables (requisitos, análisis, diseño, implementación, pruebas y despliegue) se realizan también de forma diferente. Para intentar lograr seguir determinados estándares se tiende a aplicar una combinación de

metodologías ágiles y tradicionales que al final no logran realmente el resultado esperado.

1.5.2 Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos

El Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos fue realizado en la UCI con el objetivo de contribuir con el proceso ingenieril para el desarrollo de videojuegos, y es utilizado en el Centro de Entornos Interactivos 3D VERTEX. Dicho marco se compone de cinco etapas las cuales presentan en total 20 actividades, dichas etapas son: conceptualización, diseño, implementación, prueba y mantenimiento. Presenta, además, técnicas para la ejecución de las distintas actividades e incorpora actividades de calidad definidas en el Sistema de Gestión de Calidad para los proyectos de desarrollo de la UCI.

Este marco de trabajo presenta como salidas el diseño del videojuego, criterios para validar mecanismos, especificación de mecanismos, modelo de diseño, arquitectura de software, modelo de implementación, guía de jugabilidad, registro de defectos y reporte de *postmortem*. Estas salidas constituirán las entradas para las próximas etapas (17).

Para el desarrollo de software existen diversas metodologías ágiles y tradicionales; pero ninguna logra adecuarse a las funcionalidades de las mecánicas de videojuegos, siendo un impedimento para su aplicación en esta área. Guiar el desarrollo de la propuesta de solución a través del marco de trabajo ingenieril para el proceso de desarrollo de videojuegos *Framework Engineering*, posibilita de una mejor forma la organización y estructura en el proceso de desarrollo de videojuegos. Su utilización permite alinear el trabajo realizado con la forma de trabajo en el centro VERTEX. El cumplimiento de las etapas que propone garantiza un producto de calidad, con artefactos generados que pueden ser consultados fácilmente por otro equipo de trabajo en un evento futuro. Es por esto que se escoge para el desarrollo del videojuego El mundo de Wumpus el marco de trabajo ingenieril para el proceso de desarrollo de videojuegos, realizado y puesto en práctica en la UCI.

1.6 Herramientas y tecnologías

Creadas las bases de la investigación a raíz del estudio de aspectos fundamentales relacionados con el tema a abordar, corresponde seleccionar las herramientas que se utilizarán para llevar a cabo la propuesta de solución. Para esto se determinarán las herramientas para el diseño 2D y el modelado 3D, el motor de videojuego y el lenguaje de programación a utilizar, así como la herramienta para el modelado de los elementos relacionados con la ingeniería para la creación del software. Para esto se realizará un estudio de las ventajas de la utilización de estas herramientas teniendo en cuenta el proyecto a desarrollar.

1.6.1 Herramienta para modelado de software

Visual Paradigm es una herramienta de Ingeniería de Software Asistida (CASE, por sus siglas en inglés) para el desarrollo con un Lenguaje de Modelado Unificado (UML, por sus siglas en inglés). Esta herramienta permitirá la representación de los requerimientos y estructuras del software a crear antes de su propia realización, posibilitando tener mayor organización y evitar tener que cambiar la implementación del sistema por no haber analizado de forma correcta lo que se quiere lograr. Por lo tanto, facilitará y acelerará el trabajo de desarrollo de software. Cuenta con patrones de diseño que podrán ser entendidos en un evento futuro por cualquier otro desarrollador o analista y muestra de una forma bien clara el contenido. Herramienta con soporte multiplataforma que permite a partir de códigos fuentes de programas, archivos ejecutables y binarios crear modelos UML inmediatamente. Se centra en automatizar todo el ciclo de desarrollo de software por lo que posibilita la captura de requisitos, análisis, diseño e implementación. [5]

Se selecciona como herramienta para modelar los artefactos que requiere el método ingenieril seleccionado el Visual Paradigm 8.0 por ser multiplataforma que se caracteriza por ser fácil de usar; pero a la vez es bastante potente pues soporta el ciclo de vida completo del desarrollo de software. Visual Paradigm presenta una estabilidad de

[5] Sitio oficial: https://www.visual-paradigm.com

ejecución en diferentes sistemas operativos permitiendo trabajar con un modelo UML utilizando el mismo programa sin importar el sistema operativo. Además, destacar que esta herramienta guarda todo el modelo en un solo fichero y trabaja de forma aceptable en ordenadores poco potentes.

1.6.2 Herramienta de diseño 2D

La utilización de programas propiamente creados para diseño 2D permite crear una ilustración visual con mucha mejor calidad. Son muchos los programas enfocados a este aspecto los cuales brindan variadas paletas de colores, herramientas disímiles para potenciar el trabajo creativo, así como la incorporación de efectos y estilos que lo agilizan.

Adobe Illustrator es un software basado en vectores ideal para crear ilustraciones que pueden escalarse e imprimirse en cualquier tamaño y resolución, manteniendo al mismo tiempo todos los detalles y la calidad de imagen. Es, junto con la gran familia de Adobe, uno de los programas más importantes y reconocidos de esta firma y uno de los más populares en el mundo del diseño gráfico profesional (18). Programa creado por Adobe System en 1987 y destinado a diseñadores gráficos e ilustradores. Se basa en el dibujo vectorial proporcionando un alto nivel de libertad creativa. Esta herramienta ha adicionado la posibilidad de crear múltiples mesas de trabajo, con lo cual logró una mayor aceptación. Permite además la creación de logotipos, íconos, dibujos, tipografías e ilustraciones para impresión, web, video y dispositivos móviles; manteniendo las imágenes nítidas. Brinda la posibilidad de agregar efectos, administrar estilos y editar personajes individuales para crear diseños tipográficos con un estilo propio y único[6].

Por lo antes dicho se selecciona Adobe Illustrator como programa a utilizar para el diseño 2D de la propuesta de solución, lo cual incluye las pantallas del videojuego. Se selecciona dicha herramienta pues su diseño se basa en vectores y se caracteriza por su completitud y robustez. Adobe Illustrator es compatible con el sistema operativo Windows y permite exportar e importar imágenes en diversos formatos. Su utilización permitirá brindarle mayor personalidad al videojuego con diseños propios y afines a la temática a tratar.

[6] Sitio oficial https://www.adobe.com

1.6.3 Herramienta de modelado 3D

Las herramientas de diseño 3D han cobrado un gran auge en la actualidad. Los tiempos cambian y junto a estos las tecnologías. Las herramientas 3D han participado de manera destacada en este aspecto. Por ejemplo, en la realización de dibujos animados, para programas médicos, videojuegos, para impresiones 3D, entre otros campos.

Autodesk Maya 2018 es un software ampliamente utilizado en animación y diseño industrial. Permite, mediante una variedad creativa de funciones, realizar animaciones, modelados, simulaciones, texturizado, *rigging*[7] y renderizaciones[8] en 3D. Se caracteriza por su potencia, así como por la posibilidad de expansión y personalización de la interfaz y recursos. Brinda herramientas de personajes, de simulación de ropa y cabello, efectos y dinámicas (simulación de fluidos) que presentan una gran calidad. Permite exportar los modelos en todos los formatos posibles utilizados en 3D. Maya cuenta además con MEL (*Maya Embedded Language*), el código que forma el núcleo de Maya, el cual posibilita crear *scripts* y personalizar el paquete. Es importante destacar también que Maya es el único software de 3D acreditado por un Oscar debido al gran impacto que ha tenido en la industria cinematográfica para efectos visuales (19). En cuanto a su interfaz presenta herramientas, controles y datos muy bien organizados, facilitando el trabajo con el software. Su interfaz actualmente tiene un diseño aplanado, los íconos poseen un aspecto de 2D logrando la simplicidad visual. En Maya generalmente hay más de una manera de realizar una tarea, permitiendo que el usuario pueda escoger las ventanas de edición que se adapten mejor a su estilo de trabajo (20).

Por otra parte, MakeHuman es una herramienta de código abierto que permite crear seres humanos de una forma mucho más rápida y sencilla, pero a la vez bastante personalizada y con alta calidad que pueden ser exportados para usar en proyectos como videojuegos. El usuario podrá crearle al personaje un sistema de huesos de forma inmediata y brinda opciones específicas para la exportación de los personajes para los

[7] Rigging: es el proceso mediante el cual se le adiciona un sistema de controles digitales al modelo 3D creado para que pueda ser luego animado.
[8] Renderización: proceso que consiste en llevar modelos 3D a imágenes 2D con efectos fotorrealistas 3D.

juegos evitando que el motor de videojuego a utilizar colapse por la cantidad de polígonos.[9]

Se seleccionan las dos herramientas mencionadas para el modelado de los personajes y de los objetos presentes en los distintos escenarios del videojuego a realizar. El uso de la herramienta MakeHuman en su versión 1.1.1 posibilitará crear los personajes de una forma bastante rápida. Los modelos generados por la herramienta podrán ser exportados con el mínimo de polígonos para ser usados en videojuegos. Las distintas posibilidades de operar en la herramienta Autodesk Maya 2018 y su capacidad de realizar modelado y texturizado permitirán agregar elementos en el entorno con un diseño a fin a la trama del videojuego, logrando compenetrar el concepto con el diseño de la propuesta de solución.

1.6.4 Motor de videojuego y lenguaje de programación

Unity 3D es un motor de videojuego cuya función principal es la integración de gráficos 3D y 2D, sonidos, animación, física, *scripting* e inteligencia con el fin de crear un videojuego. Permite compilar para varias plataformas como son Microsoft Windows, Linux, web, dispositivos móviles, entre otros. Cuenta con un sistema de Interfaz de Usuario (UI) como medio de comunicación rápido entre el jugador y el equipo. En este motor de videojuegos se construyen *assets*, que pueden ser texturas, modelos 3D, archivos de audio, prefabricados (*prefabs*), materiales, animaciones, *scripts*, entre otros. Con estos *assets* se crean módulos integrables a la interfaz de usuario lo cual permite la reducción del tiempo de desarrollo, la reutilización y adaptación más fácil ante futuros cambios.

Para la implementación de la propuesta de solución se utilizó como motor de videojuego Unity 3D en su versión 5.6, herramienta utilizada por el centro VERTEX para el desarrollo de videojuegos. Es multiplataforma y presenta una interfaz sencilla y flexible. Permite generar sistema de huesos en los personajes, paquetes para la navegación de los

[9] http://www.makehumancommunity.org

mismos y mecánicas de partículas que permitirán realizar el trabajo propuesto de una forma más rápida.

La implementación de los *scripts* proporciona las mecánicas del juego. Unity permite programar en lenguaje C# o Java Scripts. Para el desarrollo de El mundo de Wumpus se utilizará el lenguaje C# el cual es un lenguaje orientado a objetos bastante sencillo y brinda diversas librerías. Se utilizará como Entorno de Desarrollo Integrado (IDE) MonoDevelop en su versión 5.9.6.

1.7 Conclusiones parciales

El análisis llevado a cabo sobre los videojuegos y la utilización en los mismos de la Inteligencia Artificial, permitió identificar las características del juego a desarrollar, así como las técnicas de Inteligencia Artificial a utilizar. Estas técnicas, junto a otras estrategias específicas, deberán aportar a los 4 núcleos de conocimiento de la Inteligencia Artificial, la representación del conocimiento, el razonamiento, el tratamiento de la incertidumbre y el aprendizaje.

Mediante la consulta de diferentes bibliografías, se determinaron las características principales del juego El mundo de Wumpus, lo cual constituye las bases para el desarrollo de la propuesta de solución.

El análisis de diversas herramientas y metodologías para el desarrollo de videojuegos permitió determinar las tecnologías y el marco de trabajo a utilizar en el diseño y desarrollo del videojuego. La selección del marco de trabajo ingenieril permitió delimitar las etapas de desarrollo del software y los artefactos que se generarán. Este marco se presenta como alternativa a una metodología de desarrollo de software, el cual se adecua más a este tipo especial de software.

Capítulo 2

Caracterización y diseño del videojuego El mundo de Wumpus

En el capítulo se presenta un breve resumen acerca de la propuesta de solución. Se exponen los artefactos ingenieriles de la metodología seleccionada. Abarca aspectos del diseño del videojuego como las metas para la experiencia del jugador, los elementos formales y dramáticos; la descripción de las pantallas, especificación de mecanismos, la descripción de los patrones de diseño aplicados y la representación de los diagramas de clases del diseño. Estos elementos forman parte de las dos primeras etapas del marco de trabajo ingenieril para el desarrollo de videojuegos: conceptualización y diseño. El cumplimiento de estas dos etapas permite comenzar la tercera la cual abarca la implementación.

2.1 Conceptualización de la propuesta de solución

La conceptualización constituye la primera etapa definida por el marco de trabajo ingenieril para el proceso de desarrollo de videojuego, la cual abarca las actividades siguientes:

1. Definición del género del videojuego: se define el o los géneros del juego, abarcando además un pequeño resumen o descripción del videojuego a realizar.
2. Descripción de las mecánicas: esta actividad se lleva a cabo en la etapa de diseño como parte del artefacto Diseño del videojuego, por lo que no se realizará en la etapa de conceptualización.
3. Especificación de las metas para la experiencia del jugador.

A continuación, se da cumplimiento a las actividades definidas anteriormente.

2.1.1 Descripción de la propuesta de solución. Género del videojuego

El videojuego transcurre en las áreas del mundo de Wumpus. El jugador tendrá como objetivos matar al Wumpus y encontrar el tesoro. Para lograr el primer objetivo debe

encontrar una poción que se encuentra en algún lugar del mapa y lanzársela al Wumpus. Existirán tres posibles terrenos que se presentarán de forma aleatoria cada vez que se comience a jugar.

El juego contará con un Modo Agente y la opción de jugar.

- Modo agente: Jugará un personaje definido ya por el juego. El agente buscará todo el tiempo el tesoro y la poción para luego matar al Wumpus. En la pantalla se muestra el botón de opciones, la imagen de la poción para el wumpus y un mapa con la ubicación del jugador. Gana si logra los objetivos definidos.
- Jugar: El jugador avanzará por el camino y se guiará por el ambiente para saber por dónde ir. En la pantalla se ve el nivel de vida, el botón de opciones y la imagen de la poción para el Wumpus. Para proteger el mundo de Wumpus existe un guardián que será un agente inteligente con la misión de matar a todo aquel que entre en este mundo por lo que si ve al jugador lo perseguirá y atacará. Tanto el guardián como el jugador cuentan con un arma con la que se podrán disparar, esto provoca que el nivel de vida baje y al agotarse pues pierde el juego o simplemente el guardián muere. El jugador gana si logra los objetivos y pierde si muere porque se agotó su nivel de vida, por haber caído en un precipicio o por haber sido asesinado por el Wumpus.

El videojuego presenta el género disparo debido a que el jugador cuenta con un arma para dispararle al guardián. Teniendo en cuenta que la propuesta de juego a realizar está enfocada a ayudar a la aplicación de técnicas de IA en el proceso de desarrollo de videojuego se está en presencia también de un videojuego serio. Además, presenta el género estrategia pues el jugador necesitará trazar una estrategia para tomar la decisión correcta y no morir.

2.1.2 Metas para la experiencia del jugador

- El jugador podrá decidir el orden en que cumplirá las misiones de matar al Wumpus y obtener el tesoro.
- El jugador debe memorizar el terreno para poder desempeñarse de mejor forma mientras juega.

- El jugador adquirirá conocimiento de cómo jugar al observar como juega el agente inteligente en el modo agente.

2.2 Diseño del videojuego

Luego de definir la conceptualización del videojuego corresponde la segunda etapa de la metodología: Diseño. Las actividades que la definen son las siguientes:

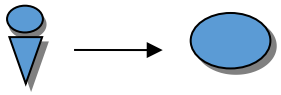
1. Descripción de los elementos formales: donde se especifican jugadores, objetivos del juego, procedimientos, reglas, recursos con que cuenta el jugador, conflictos, límites y resultado.
2. Descripción de los elementos dramáticos: se define la premisa del juego, la historia y los retos que debe afrontar el jugador.
3. Diseño de las pantallas gráficas elementales: se describen las pantallas del videojuego.
4. Validación de mecanismos: esta actividad se llevará a cabo mediante la revisión y aprobación de los mecanismos por el cliente.
5. Descripción de los elementos dinámicos: abarca los mecanismos definidos para el videojuego.
6. Descripción de las características no funcionales: consiste en definir los requisitos no funcionales del videojuego.
7. Administración de los cambios mediante matrices de trazabilidad: Los cambios se identificaron, evaluaron, reportaron y aprobaron en cada encuentro con el cliente. Esta actividad permitió conocer las modificaciones realizadas a un requisito.
8. Representación de la concepción arquitectónica de los mecanismos: se define y modela el paquete de mecanismos. Se obtiene la arquitectura de software.
9. Diseño de la estructura de los mecanismos: se modelan los diagramas de clases según mecanismos definidos.
10. Modelado del comportamiento de los mecanismos: modelado de los diagramas de estado para cada mecanismo.

A continuación, se da cumplimiento a las actividades vistas anteriormente.

2.2.1 Elementos formales

Los elementos formales definen la estructura del videojuego, estos son:

Jugador:

Aspecto	Descripción
Invitación a Jugar	Botón de comienzo: “Jugar”.
Cantidad	Un jugador.
Roles	Es quien interactúa con el mundo del Wumpus para matarlo y obtener el tesoro.
Patrón de Interacción	Individual vs Juego. 

Objetivos:

Objetivo	Descripción
Matar al Wumpus	Para lograr dicho objetivo el jugador debe encontrar la poción que se encuentra en otra habitación de la cueva y lanzársela al Wumpus.
Tipo	Explícito.
Categoría	Exploración.

Objetivo	Descripción
Obtener el tesoro	El jugador debe encontrar el tesoro que se haya en la cueva.
Tipo	Explícito.
Categoría	Exploración.

Procedimientos:

- Caminar: el jugador podrá desplazarse por el terreno presionando flecha arriba, abajo, izquierda y derecha para ir hacia adelante, hacia atrás y hacia los lados. Puede también presionar las teclas W, S, A y D respectivamente.
- Girar: el jugador podrá girar desplazando el mouse.
- Lanzar poción: el jugador podrá lanzar la poción arrastrándola con el clic sostenido hasta el cartel que se le muestra en pantalla.
- Disparar: el jugador podrá disparar presionando la barra de espacio o el clic izquierdo del mouse.

Reglas

- El jugador para matar al Wumpus debe haber encontrado primero la poción.
- El jugador muere si su nivel de vida llega a cero por haber sido atacado por el guardián, si cae en un precipicio o si entra en la casilla del Wumpus.
- El jugador pierde si muere o si utilizó la poción de forma incorrecta.
- El jugador avanza de nivel si mata al Wumpus y encuentra el tesoro.

Recursos:

- Poción con la que podrá matar al Wumpus.
- Nivel de vida del jugador.

Conflictos:

Si el jugador es atacado por el guardián el nivel de vida disminuye haciendo que este muera y que el juego se termine.

Frontera o Límite:

Haber alcanzado los dos objetivos propuestos: matar al Wumpus y encontrar el tesoro.

Resultado:

El juego se gana al haber alcanzado los dos objetivos propuesto: matar al Wumpus y encontrar el tesoro.

2.2.2 Elementos dramáticos

Los elementos dramáticos definen el entretenimiento y nivel de inmersión de los jugadores al jugar, estos son:

Premisa:

El juego surge por la necesidad de contribuir a la aplicación de diversas técnicas de IA en el desarrollo de videojuegos.

Historia:

El jugador es una especie de aventurero y su misión actual es entrar en las fortalezas del Wumpus para matarlo y llevarse el tesoro que ahí se encuentra.

Reto:

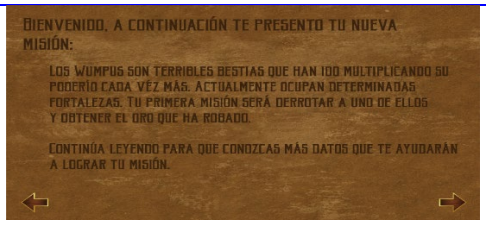
El jugador debe encontrar el tesoro y matar al Wumpus evitando morir asesinado por el Wumpus o por el guardián, o cayendo en un precipicio.

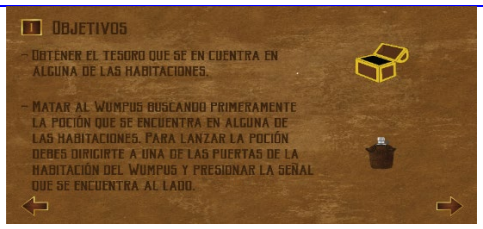
2.2.3 Diseño de las pantallas gráficas

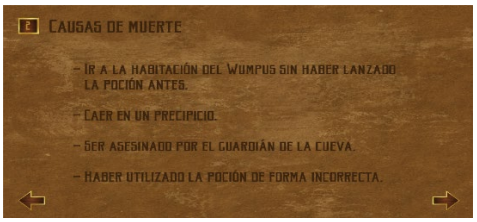
A continuación, se expone una descripción de las pantallas del videojuego las cuales constituyen los elementos visuales que el usuario apreciará al interactuar con el videojuego.

Imagen	Descripción
	Pantalla principal: Contiene el nombre del videojuego y los botones: Jugar, Modo Agente, Ayuda, Ajustes, Créditos y Salir.

	Ajustes: Permite ajustar volumen y sonido.
	Créditos: Se muestran los créditos: realizador y director del proyecto.
	Salir: Permite salir del juego.
	Modo Agente: Se muestra cómo juega el agente inteligente. En la pantalla se observa el botón de menú, la imagen de la poción para el Wumpus, los objetivos que deben ser completados y dos mapas: uno que muestra el análisis realizado por el agente y su ubicación, y otro que muestra el mapa real del terreno. La poción se representa como una sombra cuando el agente no la ha encontrado, en caso de que cuente con la poción se mostrará. Los objetivos completados se marcarán con una palomita.

	Jugar: En esta pantalla es donde tiene lugar el juego. Se observan los mismos elementos de la pantalla del Modo Agente; pero a diferencia de esta en el mapa no se muestra ningún elemento, solo la ubicación del jugador. Además, se puede ver la barra de vida del jugador.
	Menú: Permite ajustar la música y el sonido y acceder a los botones - Inicio: ir a la pantalla principal. - Salir: salir del juego. - Reanudar: reanuda el juego que estaba en ejecución.
	Lanzar poción: Es el espacio donde se lanzará la poción. Brinda una explicación de qué debe hacer el jugador y presenta: - Espacio que enmarca la sombra de la poción para lanzarla. - Botón Cerrar: cerrar de la ventana.
	Ayuda: Contiene toda la información necesaria para el jugador: 1. Introducción: le presenta al jugador el juego y la razón del mismo.

<table>
<tr>
<td> 




</td>
<td>2. Objetivos: explica cuáles son los objetivos y qué debe hacer para lograrlos.
3. Causas de muertes: explica las posibles formas de que el jugador pierda el juego.
4. Información útil: brinda datos importantes acerca del terreno como son la ubicación del precipicio y del hedor.
5. Información en pantalla: explica los elementos presentes en la pantalla al jugar.
6. Controles: da a conocer los controles necesarios para el movimiento del jugador y el disparo.</td>
</tr>
<tr>
<td> 
</td>
<td>Felicitaciones Objetivo Obtener Tesoro:
Muestra un cartel de felicitación por haber completado el objetivo obtener el tesoro.
- Cerrar: cerrar el cartel de felicitaciones y continuar el juego.</td>
</tr>
</table>

	Felicitaciones Objetivo Matar al Wumpus: Muestra un cartel de felicitación por haber completado el objetivo matar al Wumpus. - Cerrar: cerrar el cartel de felicitaciones y continuar el juego.
	Avance de nivel: a través del hermano del wumpus asesinado por el jugador se da información de haber completado el nivel. El jugador puede escoger continuar o ir a inicio. - Inicio: ir a la pantalla principal. - Continuar: que el jugador avance de nivel.
	Game Over: Muestra un cartel informando la causa de perder el juego: haber sido asesinado, por caer en un precipicio, caer en manos del Wumpus o haber utilizado mal la poción. - Inicio: ir a la pantalla principal. - Reanudar: volver a jugar.

Tabla 1. Descripción de las pantallas

2.2.4 Especificación de mecanismos

Los mecanismos son una forma de plasmar las funcionalidades que tendrá determinado videojuego, estos reflejan la relación del usuario con el sistema. En este apartado se especificarán los mecanismos que conforman el videojuego, lo cual incluye a los objetos que presenta y sus propiedades, el comportamiento del mecanismo, las relaciones con otros y la organización arquitectónica (OA) a la que pertenecen.

No.	Nombre	Descripción	OA
1	Mecanismo control del Menú Principal	**Objetos:** botones (Jugar, Modo Agente, Ayuda, Ajustes, Créditos, Salir). **Propiedades:** - Jugar: comenzar a jugar. - Modo Agente: comenzar el modo agente. - Ayuda: muestra las instrucciones del juego. - Ajustes: ver descripción del Mecanismo Ajustes. - Créditos: muestra los créditos. - Salir: permite salir del videojuego. **Comportamientos:** Es la primera pantalla del videojuego. Permite acceder a ajustes, créditos, modo agente y ayuda; así como salir o comenzar el juego. **Relaciones:** M4, M7	Mecanismo núcleo
2	Mecanismo búsqueda de la poción	**Objetos:** jugador, poción. **Propiedades:** - Jugador: debe buscar en el mapa la poción. - Poción: se encuentra en algún lugar del mapa y debe ser encontrada por el jugador. **Comportamientos:** Para que el jugador pueda matar al Wumpus, que es uno de sus objetivos, debe buscar la poción que se encuentra en algún lugar del mapa. Al tener la poción podrá lanzársela al Wumpus y este muere al instante. **Relaciones:** M8, M10	Mecanismo núcleo
3	Mecanismo Vida	**Objetos:** jugador, barra de vida. **Propiedades:** - Jugador: presenta un nivel de vida. - Barra de vida: al agotarse el jugador muere.	Mecanismo núcleo

		Comportamientos: El jugador presenta una barra de vida que puede disminuir al ser este atacado por el guardián. El jugador muere al agotarse el nivel de la vida. **Relaciones:** M6	
4	Mecanismo Ajustes	**Objetos:** volumen de música, volumen de sonido y botón cerrar. **Propiedades:** - Volumen de música: permite subir o bajar el volumen de la música. - Volumen de sonido: permite subir o bajar el volumen del sonido. - Botón cerrar: permite salir de la opción Ajustes volviendo a la pantalla principal. **Comportamientos:** Se accede a la opción ajustes desde el menú principal. Esta opción permite ajustar el volumen de la música y el sonido. **Relaciones:** M1, M9	Mecanismo alternativo
5	Mecanismo búsqueda del tesoro	**Objetos:** jugador, tesoro. **Propiedades:** - Jugador: debe buscar en el mapa el tesoro. - Tesoro: se encuentra en algún lugar del mapa y debe ser encontrado por el jugador. **Comportamientos:** Uno de los objetivos es encontrar el tesoro, por esto el jugador debe buscar el tesoro que se encuentra en algún lugar del mapa. **Relaciones:** M10	Mecanismo núcleo
6	Mecanismo Disparar	**Objetos:** jugador, guardián, pistola. **Propiedades:**	Mecanismo alternativo

		- Jugador: puede disparar usando una pistola. - Guardián: puede disparar usando una pistola. - Pistola: cada jugador tiene una. La cantidad de balas es ilimitada. **Comportamiento:** el jugador y el guardián cuentan con una pistola con la que podrán disparar logrando disminuir el nivel de quien es atacado. Si el jugador es atacado y se queda sin nivel de vida muere y pierde el juego. Las balas son ilimitadas por lo que podrá disparar las veces que considere necesario. **Relaciones:** M3	
7	Mecanismo Modo Agente	**Objetos:** Poción, Mapas y Botón Menú. **Propiedades:** - Menú: ver descripción del Mecanismo Menú. - Poción: indica si el agente cuenta con la poción o no. En caso de no tenerla se muestra solo la sombra de la imagen de la poción. El agente deberá utilizar la poción para matar al Wumpus. - Mapa1: mapa que muestra la ubicación del jugador y el análisis que realiza del terreno. - Mapa2: Imagen real del terreno. **Comportamiento:** El agente buscará todo el tiempo el tesoro y la poción para luego matar al Wumpus. Gana si obtiene el tesoro y mata al Wumpus. **Relaciones:** M1, M9.	Mecanismo alternativo
8	Mecanismo matar al Wumpus	**Objetos:** jugador, Wumpus. **Propiedades:** - Jugador: debe buscar en el mapa al Wumpus. - Wumpus: se encuentra en algún lugar del mapa y debe ser encontrado por el jugador. **Comportamiento:**	Mecanismo núcleo

		Uno de los objetivos del jugador es matar al Wumpus. El jugador, luego de haber buscado y obtenido la poción, tiene que buscar al Wumpus y lanzársela para que muera. Si el jugador entra en la habitación donde está el Wumpus sin tener la poción o sin lanzársela morirá inmediatamente y el juego termina. **Relaciones:** M2, M10	
9	Mecanismo Menú	**Objetos:** volumen de música, volumen de sonido, botones (inicio, salir, reanudar). **Propiedades:** - Volumen de música: permite subir o bajar el volumen de la música. - Volumen de sonido: permite subir o bajar el volumen del sonido. - Botón reanudar: permite salir de la opción Menú volviendo a la pantalla del juego. - Botón inicio: permite salir de la pantalla juego para ir al menú principal. - Botón salir: permite salir del videojuego. **Comportamientos:** Se accede a la opción Menú desde la pantalla del juego y la pantalla Modo Agente. Esta opción permite ajustar el volumen de la música y el sonido, salir del juego o ir al menú principal. **Relaciones:** M1, M4, M7	Mecanismo núcleo
10	Mecanismo Mapa	**Objetos:** mapa. **Propiedades:** - Mapa: permite visualizar la ubicación del jugador en el terreno.	Mecanismo alternativo

		Comportamientos: En la pantalla de juego el jugador podrá visualizar en todo momento el mapa. Este solo contiene la información de la ubicación del jugador en el mapa. **Relaciones:** M2, M5, M8	

Tabla 2. Especificación de mecanismos

2.2.5 Requisitos no funcionales

Los requisitos no funcionales describen las propiedades que el producto debe tener para ser usable y confiable. A continuación, se especifican los requisitos no funcionales:

RnF 1: Jugabilidad y Usabilidad

- RnF1.1: El jugador se desplaza a través de las pantallas desde una PC haciendo clic en los botones definidos por el videojuego.
- RnF 1.2: El jugador avanzará de nivel a medida que alcance los objetivos propuestos en cada nivel.
- RnF 1.3: El videojuego podrá ser jugado por cualquier usuario, pero se recomienda ser usado por mayores de 12 años debido a que presenta el género shooter.
- RnF 1.4: No se requiere de conocimientos sobre informática ni sobre IA para usar el videojuego.
- RnF 1.5: El juego cuenta con la opción de ayuda para que el usuario comprenda sus objetivos y las reglas del juego.

RnF 2: Rendimiento

- RnF 2.1: El tiempo en ejecutarse las diferentes acciones será como mínimo de 30 frame por segundos.

RnF 3: Apariencia o interfaz gráfica

- RnF 3.1: En las interfaces predominan los colores oscuros como el carmelita y el negro, en contraste con el color dorado, teniendo en cuenta la temática a tratar.
- RnF 3.2: El tamaño de letra se encuentra entre 15 y 30.
- RnF 3.3: Los botones mantienen un mismo estilo de diseño y son bastante intuitivos, su proporción les brinda consistencia a las interfaces.
- RnF 3.4: La resolución de la pantalla es ajustable según el dispositivo donde se ejecute el videojuego.
- RnF 3.5: Se tuvo en cuenta para el diseño de las interfaces que fueran intuitivas además se cuenta con la opción de ayuda en caso de existir alguna duda con el videojuego por parte del usuario.

RnF 4: Software

- RnF 4.1: La aplicación debe ser compatible con los sistemas operativos Windows en su versión 8.0 o superiores y Linux.

RnF 5: Hardware

- Para ejecutar la aplicación de forma satisfactoria el dispositivo donde corra debe contar con un microprocesador Intel o AMD a partir de 2.0 GHz y más de 1GB de memoria RAM.

RnF 6: Restricciones de diseño e implementación

- RnF 6.1: Como motor de videojuego se usará la herramienta Unity 3D en su versión 5.6.
- RnF 6.2: La metodología a utilizar para el proceso de desarrollo del videojuego es Framework Ingenieril: marco de trabajo ingenieril para el proceso de desarrollo de videojuegos.
- RnF 6.3: El IDE de desarrollo será ModoDevelop 5.9.6.
- RnF 6.4: El lenguaje de programación será C#.
- RnF 6.5: La herramienta a utilizar para CASE será Visual Paradigm.

- RnF 6.6: La herramienta para realizar el diseño 2D de las pantallas será Adobe Illustrator 2018.
- RnF 6.7: Las herramientas para realizar el modelado 3D serán Autodesk Maya 2018 y MakeHuman versión 1.1.1.

2.2.6 Concepción arquitectónica de los mecanismos

La elaboración de un paquete de mecanismos para los videojuegos permite organizar los mecanismos presentes y cómo estos se relacionan entre sí para comprender la estructura del sistema. Para elaborar el paquete de mecanismos es importante conocer que un paquete puede contener otros paquetes, se debe mostrar la relación de dependencia entre los mecanismos y que estos se diferenciarán, en este caso, por dos grupos:

- Mecanismos núcleos: son aquellos mecanismos indispensables para el videojuego. Constituyen el sentido de motivación del jugador para alcanzar su propósito. Dentro de esta clasificación entran además aquellos mecanismos que permiten desplazarse por las distintas interfaces realizando acciones necesarias para el fin propio del videojuego.
- Mecanismos alternativos: son aquellos mecanismos no indispensables para lograr el objetivo del videojuego. Se relacionan con los mecanismos núcleos y permiten incorporar mayores funcionalidades al videojuego.

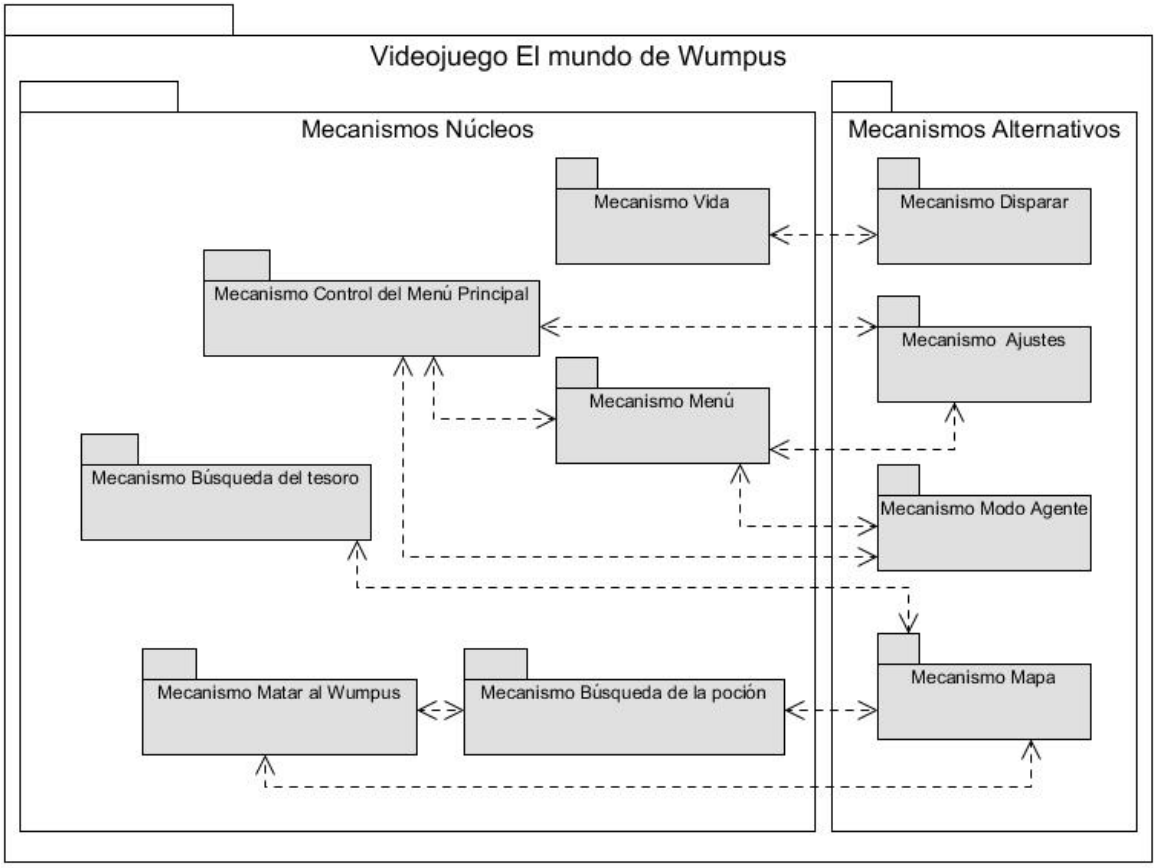


Figura 1. Paquete de mecanismos

2.2.7 Arquitectura del software

Como arquitectura del sistema se propone una basada en componentes. Esta arquitectura se basa en la descomposición del diseño en componentes funcionales o lógicos. Se tuvo en cuenta para la selección que está enfocada a sistemas formados por componentes individuales diseñados para cumplir con cierto propósito. Su uso permite la reutilización de los componentes y facilita la gestión de la complejidad, además, estos pueden relacionarse entre ellos (21). Para su diseño se tuvo en cuenta el punto de vista orientado a objeto que define que un componente es un conjunto de clases que colaboran (22). A continuación, se definen los componentes para el software en cuestión:

Componentes de Interfaz y sonido: posee los elementos esenciales del videojuego.

- Game_Manager: controla los paneles que se encuentran en las diferentes escenas y se encarga de pasar de una escena a otra y de salir del juego.

- SetMusic: controlador del sonido para la **Pantalla Principal** y la opción **Menú**.

Componentes del jugador: se encarga del manejo del personaje.

- FPSInputControl: permite controlar los movimientos de desplazamiento del jugador.
- MouseLook: permite controlar los movimientos de giro del jugador.
- Disparo: le permite al jugador disparar.

Componentes del guardián: controla las acciones del guardián.

- GuardianController: controla el movimiento del guardián. Se encarga de los estados perseguir y disparar.
- DisparoEnemigo: realiza el disparo del guardián.

Componentes de la IA: se encarga del manejo del agente inteligente en el **Modo Agente**.

- IAController: gestiona lo referente al agente inteligente del **Modo Agente**. Permite que este actualice su base de conocimiento, razone y tome una decisión.

Componentes de los objetos de las escenas: posee los distintos objetos de los escenarios presentes en el juego.

- Poción: controla el estado de la poción permitiendo saber si está activa o no, o si ya se utilizó.
- Tesoro: controla el estado del tesoro, si no se encuentra activo es que el jugador ya alcanzó el objetivo Obtener el tesoro.
- Puerta: controla el estado de las puertas permitiendo abrir y cerrarlas para permitir el paso del jugador o el agente.
- Celda: contiene la información de cada celda lo cual será útil para que el agente inteligente del Modo Agente tome una decisión y alcance los objetivos.
- BarraVida: tiene en cuenta los disparos recibidos por el jugador para disminuir el nivel de vida. Si llega a cero el juego se termina.

- Precipicio: se encarga de saber si el precipicio colisiona con el jugador. En caso de que colisione termina el juego.
- Mapa: referente al mapa que se muestra en pantalla para conocer la ubicación del jugador.

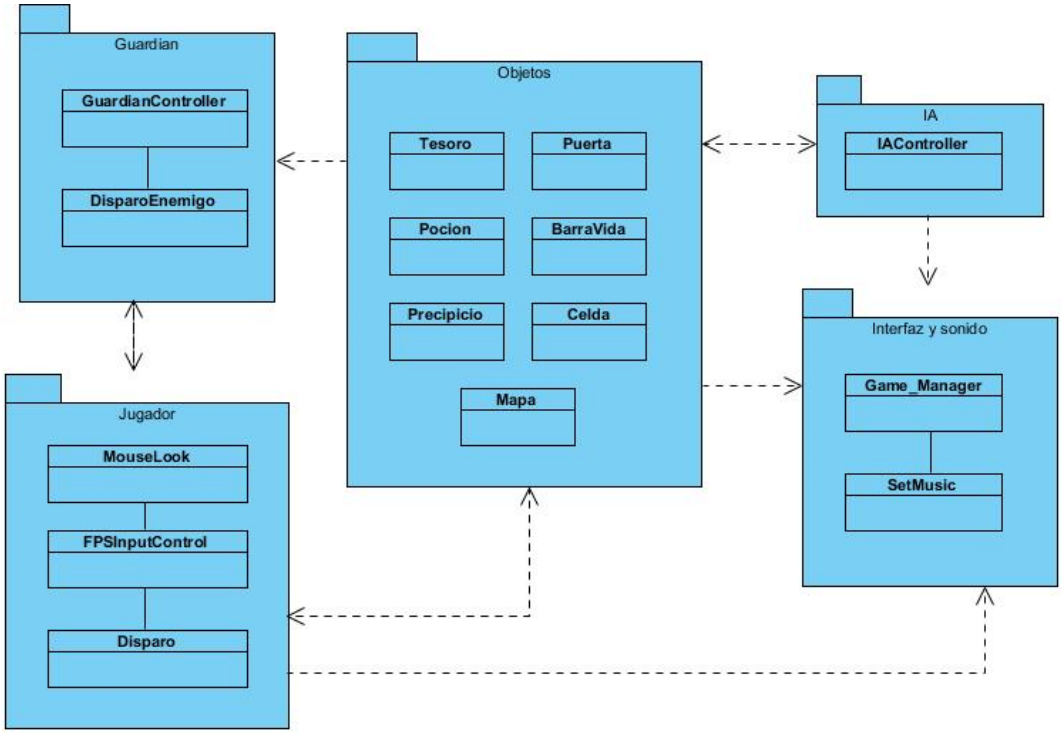


Figura 2. Arquitectura de Software

2.2.8 Diseño de la estructura de los mecanismos

Los diagramas de clases permiten mostrar la relación entre los diferentes *scripts* utilizados en el videojuego que dan cumplimiento a los mecanismos definidos. Cada clase contiene nombre, atributos y métodos que presentan. A continuación, se presenta el diagrama de clases para el mecanismo **Modo Agente** (ver anexos 1, 2 y 3 para conocer los diagramas de clase de los demás mecanismos)

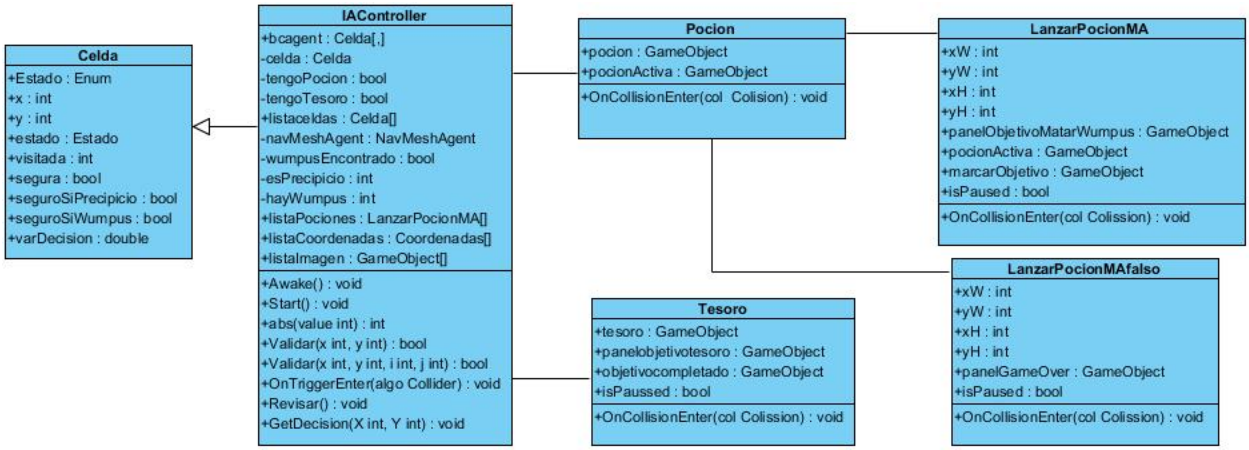


Figura 3. Diagrama de clases Mecanismo Modo Agente.

2.2.9 Patrones de diseño

Los patrones de diseño abarcan descripciones de clases y objetos que se comunican entre sí de forma tal que resuelvan un problema de diseño general según el contexto dado. Los patrones de diseño favorecen la reutilización del código, permiten dar soluciones concretas siendo utilizados en situaciones frecuentes y permitiendo la conexión de componentes.

Para la propuesta de solución se utilizan los Patrones generales asignación de responsabilidades (GRASP por sus siglas en inglés) para describir los principios fundamentales de la asignación de responsabilidades a objetos, y la Pandilla de Cuatro (GOF (*Gang of Four Book*)) definen solo tres tipos de patrones: de creación, estructurales y de comportamiento (22) (23).

Patrones GRASP utilizados:

- Experto: clase con información necesaria para cumplir con determinada responsabilidad. Ejemplo de su utilización es la clase **Game_Manager** encargada de la navegabilidad y el tránsito a escenas, para lo cual presenta todos los paneles y escenas que necesita.
- Bajo Acoplamiento: asigna responsabilidades de forma tal que las clases se comunican con el menor número de clases posibles. En un principio se desarrolló para el sistema un *script* para abrir y cerrar cada panel existente, luego con la utilización de este patrón de diseño se logró unir estos *scripts*

creando solo una clase. También se evidencia esto en los *scripts* utilizados en el modo agente y jugar que presentaban la misma funcionalidad.

- Alta cohesión: asigna a las clases responsables que trabajen sobre un área específica para evitar la complejidad. Para las acciones que puede realizar el jugador se utilizan tres *scripts* uno encargado del desplazamiento, otro del giro a través del mouse y otro para el disparo.
- Controlador: objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. En este caso la clase **Game_Manager** actúa como controlador.
- Creador: consiste en asignarle a una clase la responsabilidad de crear instancias de otra clase. En el videojuego se puede ver a la clase **Game_Manager** y la clase **IA_Controller** como creadoras.

Patrones GOF utilizados:

- Observador: se evidencia en las dependencias entre clases. En el caso del videojuego se utiliza cuando el sistema debe determinar si se ha ganado el juego, para esto debe notificarse que se lograron los dos objetivos propuestos.
- Fachada (*Facade*): el **Game_Manager** funciona como Fachada pues provee las interfaces para una serie de clases.
- Instancia única (*Singleton*): la clase **Game_Manager** permite agrupar determinadas funcionalidades que se acceden desde momentos y partes distintas del videojuego evitando sobrecargar de *scripts* al juego.

2.2.10 Modelado del comportamiento de los mecanismos

Para representar el comportamiento del videojuego se utilizarán los diagramas de estados, donde se podrá visualizar un diagrama por cada mecanismo definido. A continuación, se presenta el diagrama de estados para el mecanismo **Modo Agente** (ver anexos 4, 5, 6, 7, 8, 9, 10, 11 y 12 para conocer los diagramas de estados de los demás mecanismos).

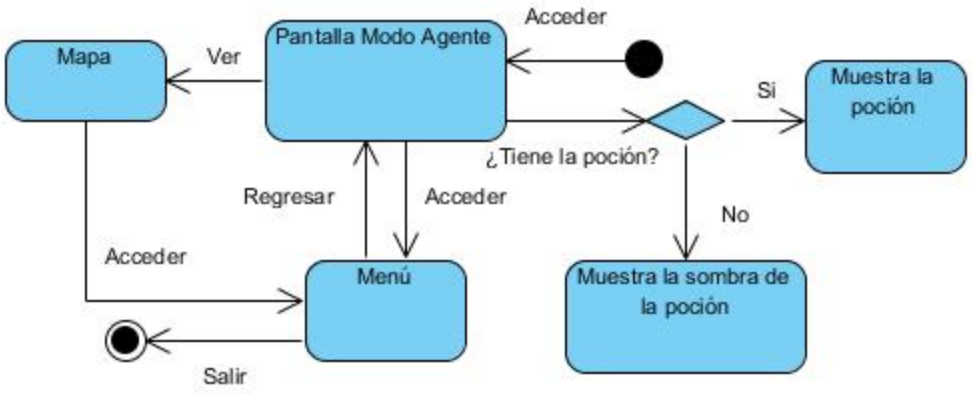


Figura 4. Diagrama de estados Mecanismo Modo Agente.

2.2 Conclusiones parciales

Con el presente capítulo se da cumplimiento a las dos primeras etapas que propone el método ingenieril utilizado permitiendo concretar la propuesta de solución. El cumplimiento de las acciones de la etapa de conceptualización permitió definir las ideas específicas para el desarrollo del videojuego.

La etapa de diseño permitió describir los elementos formales y dramáticos, así como el diseño de pantallas, con lo cual se logra una correcta representación del juego. La elaboración de mecanismos agrupados y relacionados a través de los artefactos generados: paquete de mecanismos y los diagramas de estados, permitieron definir funcionalidades del videojuego.

El completamiento de las etapas de Conceptualización y Diseño, presentes en el marco de trabajo ingenieril para el desarrollo de videojuegos, deja el camino listo para la implementación del videojuego El mundo de Wumpus. Se demuestra además las ventajas que proporciona el uso del marco de trabajo seleccionado para el proceso de desarrollo de videojuegos.

Capítulo 3

Implementación y pruebas del videojuego El mundo de Wumpus

Este capítulo está enfocado a presentar los elementos esenciales relacionados con la implementación del videojuego y a mostrar los resultados de las pruebas de aceptación realizadas a la propuesta de solución. Los elementos a tratar darán cumplimiento a las etapas de Implementación y Prueba teniendo en cuenta el marco de trabajo ingenieril para el desarrollo de videojuegos. Para ello el capítulo abarca el estándar de codificación utilizado y la implementación de la IA. Se define la estructura del videojuego a nivel de componentes y se muestran los resultados obtenidos en las pruebas de aceptación.

3.1 Estándar de codificación

Los estándares de codificación son un conjunto de convenciones que se establecen para la escritura del código en programación. Estos estándares dependen del lenguaje de programación a utilizar y forman parte de las buenas prácticas a la hora de programar. El uso de un estándar de codificación para el presente proyecto permite una mayor calidad y posibilidad de que el código sea mantenible y que pueda ser entendido en menor tiempo. A continuación, se muestran algunos de los estándares de codificación que se tuvieron en cuenta en la implementación del videojuego:

- Definición de clases: el nombre de las clases debe empezar por letra mayúscula. Si es compuesto, cada palabra debe empezar por mayúscula, pero no debe contener espacios (los espacios se pueden sustituir por un guión bajo). Se abre la llave ¨ { ¨ al final de la línea de declaración y la llave de cierre ¨ } ¨ debe aparecer en línea aparte con la misma indentación que el método o clase que cierra. Ejemplo:

```
public class Game_Manager : MonoBehaviour
```

- Declaración e inicialización de variables: cada variable se declara en una línea distinta y la inicialización se realiza junto a la declaración en los casos en el que se tenga un valor inicial. El nombre de la variable debe comenzar con minúscula, ser representativo y no contener espacios. Ejemplo:

```
public float rangodeJugador = 5.0f;
```

- Métodos: la forma de nombrarlos es igual a como se nombran las clases. Ejemplo:

```
public void OnDrag(PointerEventData eventData)
```

- Indentación: la indentación permite que sea más legible el código. Se determina por corchetes que separan a los bloques lógicos de código. Ejemplo:

```
if (col.gameObject.tag == "Player" || col.gameObject.tag == "IA")
{
   pocion.gameObject.SetActive (false);
   pocionActiva.gameObject.SetActive (true);
   pocionNoActiva.gameObject.SetActive (false);
}
else
{
   pocionActiva.gameObject.SetActive (false);
   pocionNoActiva.gameObject.SetActive (true);
}
```

3.2 Diagramas de componentes

La primera actividad definida en la etapa de implementación del marco de trabajo que guía el desarrollo de la solución propuesta es el diseño de los componentes que encapsulan la implementación. Un componente es la parte modular, desplegable y reemplazable de un sistema que encapsula implementación y expone un conjunto de interfaces. Para la definición de componentes se tendrá en cuenta el criterio orientado a objeto el cual define que un componente contiene un conjunto de clases que colaboran entre sí.

Diagrama de componentes Mecanismos Control del menú principal, Ajustes y Menú:

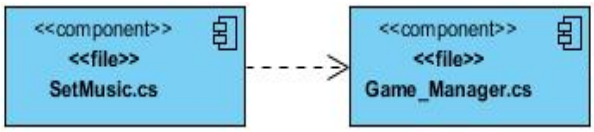


Figura 5. Diagrama de componentes Mecanismos Control del menú principal, Ajustes y Menú.

Diagrama de componentes Mecanismos Búsqueda de la poción, Búsqueda del tesoro y Matar al Wumpus:

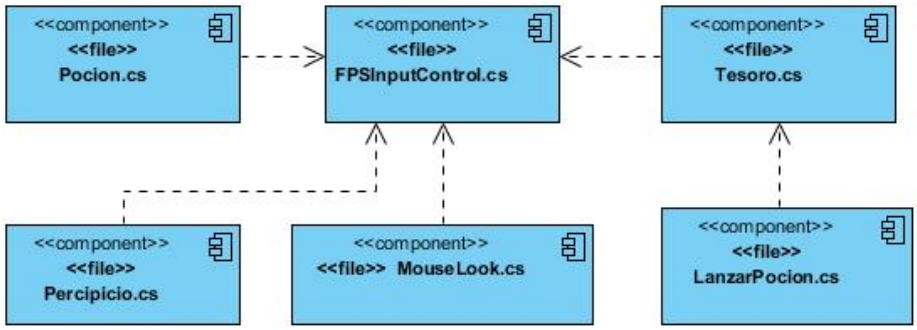


Figura 6. Diagrama de componentes Mecanismos Búsqueda de la poción, Búsqueda del tesoro y Matar al Wumpus.

Diagrama de componentes Mecanismos Disparar y Vida:

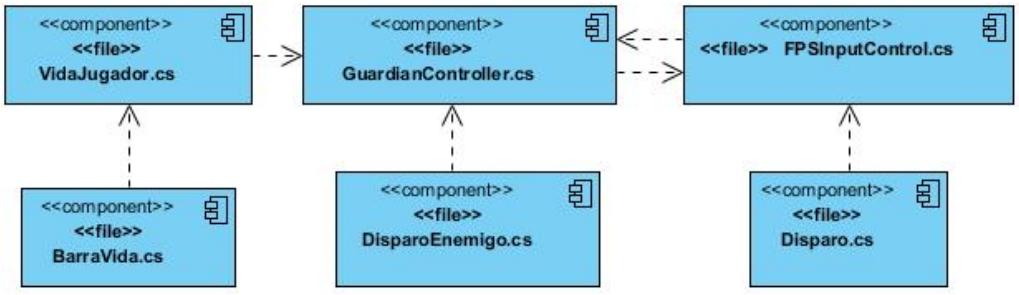


Figura 7. Diagrama de componentes Mecanismos Disparar y Vida.

Diagrama de componentes Mecanismo Modo Agente:

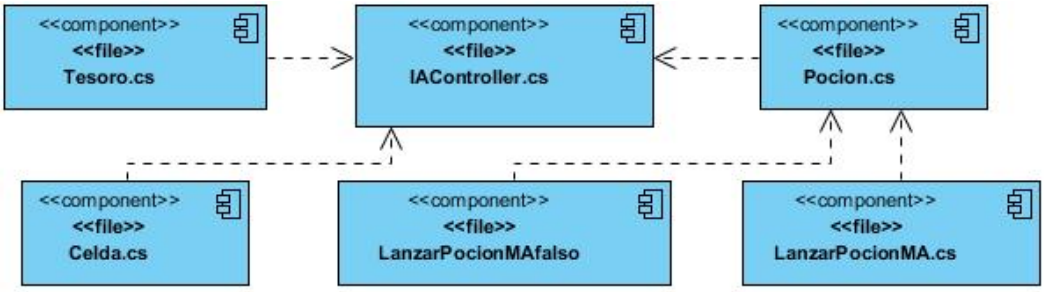


Figura 8. Diagrama de componentes Mecanismo Modo Agente.

3.3 Implementación de la IA en el videojuego

En el videojuego realizado se ponen en práctica algunas técnicas específicas y varias estrategias para satisfacer los cuatro núcleos fundamentales de IA: representación del conocimiento, tratamiento de la incertidumbre, razonamiento y aprendizaje.

Una de las técnicas específicas implementada es una máquina de estados para el guardián del mundo del Wumpus la cual cuenta con los estados:

- Perseguir: el guardián tendrá como destino la posición del jugador. En caso de que colisione con el jugador entonces pasará al estado atacar.
- Atacar: el guardián disparará al jugador hasta que el jugador no colisione con él o hasta que el jugador muera y termine el juego.

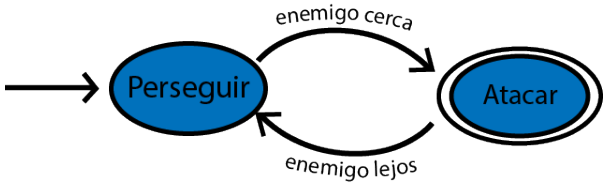


Figura 9. Máquina de estados

Otra de las técnicas son los comportamientos de locomoción. Las versiones de Unity más recientes incorporaron la librería **UnityEngine.AI** la cual da acceso a diversos elementos relacionados con la IA. De la librería se utiliza el **NavMeshAgent** el cual se encarga del movimiento. Permite que el objeto que se moverá llegue al punto especificado por el camino más corto. A continuación, se especifica su uso en el videojuego propuesto:

- En la opción **Jugar** el guardián debe perseguir al jugador. Se utiliza el NavMeshAgent para especificarle que se dirija a la posición del jugador:
  `navMeshAgent.destination = player.position;`
- En el **Modo Agente** el personaje debe avanzar hacia determinada celda. Se emplea el NavMeshAgent para especificarle que se dirija hacia la celda adyacente a él que cumpla con cierta condición:
  `navMeshAgent.destination = listaceldas [l].transform.position;`

Un agente inteligente utiliza la percepción para obtener información del entorno y responder en consecuencia según el propósito que este tenga. En el videojuego se hace uso de *colliders* para garantizar la percepción:

- En la opción **Jugar** el guardián debe detectar cuándo el jugador se encuentra cerca de él para comenzar a atacar. Si el jugador entra en el *collider* del guardián este entonces se detendrá y comenzará a dispararle al jugador, al salir del *collider* entonces continuará persiguiéndolo.
- En el **Modo Agente** el personaje presenta un *collider*. Cuando el agente llega a una habitación y colisiona con el objeto tipo celda entonces obtendrá la información que le brinda esa celda para almacenarla en su base de conocimiento y poder decidir hacia qué habitación dirigirse.

3.3.1 Representación del conocimiento

Para representar el conocimiento en el videojuego se asigna un *script* llamado **Celda** en cada habitación. Este *script* contiene una variable para la coordenada x y otra para la y que permiten saber la posición de dicha habitación. También contiene una variable de

tipo enum para determinar si en la habitación se encuentra brisa, hedor, resplandor, precipicio, Wumpus, poción o null. Este *script* también presenta otras variables, las cuales son:

- VACIA de tipo enum: es el estado que presentan las celdas de la matriz que constituye la base de conocimiento en un comienzo. A medida que el agente las visita o realiza suposiciones del estado que tendrán las celdas este estado cambiará.
- visitada: permite saber si una casilla fue visitada o no.
- segura: son aquellas casillas no visitadas pero que el agente sabe que no presentan un peligro para él.
- seguroSiPrecipicio: marca la casilla con la posibilidad de que haya en ella precipicio.
- seguroSiWumpus: marca la casilla con la posibilidad de que en ella se encuentre el Wumpus.
- varDecision: según el estado se denota un valor, la casilla de menor valor será la más segura.

El agente inteligente cada vez que llega a una habitación consulta el *script* **Celda** para saber qué información tiene de la habitación donde se encuentra, la marca como visitada y guarda esta información en una matriz. Esta matriz es el conocimiento que va adquiriendo el agente sobre el entorno y se irá actualizando a medida que el agente avance por las distintas habitaciones.

3.3.2 Razonamiento

El agente tomará las decisiones a partir del razonamiento que realice con los datos que presenta. El razonamiento consiste en deducir información a partir de la ya existente, es por esto que el agente, luego de almacenar en su base de conocimiento (matriz) la información de la casilla visitada, realiza conjeturas de los posibles estados en las casillas adyacentes teniendo en cuenta las reglas del juego y los valores de las casillas visitadas anteriormente. Este análisis permite marcar casillas no visitadas como: seguro si precipicio, seguro si Wumpus y segura.

Para comenzar se tuvo en cuenta mediante el método **Validar** que el personaje no saliera del tablero con posiciones no válidas como por ejemplo x igual a un número negativo. Cada vez que el agente llega a una habitación y su estado sea **Pocion** o **Resplandor** debe notificarlo mediante las variables **tengoPocion** y **tengoTesoro**. En estos casos y cuando el estado sea **Null** marca las casillas adyacentes a esta como seguras y actualiza la información de la base de conocimiento.

Si el estado de la casilla visitada es **Brisa** y una de las casillas adyacentes a ella no ha sido visitada ni está marcada como **segura** se marcarán como **seguroSiPrecipicio**. En caso de que una casilla adyacente a esta esté marcada como **segurosiWumpus** entonces se marcará como **segura**. Lo mismo sucede para el caso de que el estado de la casilla visitada sea **Hedor**.

Luego de realizar este análisis se invoca al método **Revisar** para garantizar que la información actualizada no contradiga las reglas del juego. Entonces se llama al método **GetDecision**, el cual se encarga de decidir hacia donde debe ir el jugador. Para esto tendrá en cuenta la variable **varDecision** que se actualiza en el método **Revisar**. El método encuentra la casilla adyacente con menor **enteroDecision** y dirige al agente hacia esa casilla.

En caso de que estén verdaderas las variables **wumpusEncontrado, tengoPocion** y **tengoTesoro** entonces el jugador avanzará hacia una casilla donde haya hedor y se aproximará al botón que se encarga de lanzar la poción. Para garantizar que el agente se dirija a casillas seguras se tiene en cuenta que avance hacia casillas que no implican un peligro, y en caso de que una casilla adyacente sea hedor irá hacia esa y buscará el botón de poción para lanzársela al Wumpus. Para asegurar que el botón sea el correcto se le asignó un *script* a cada uno que tiene como variables la posición de la celda donde se encuentran (celda con estado **Hedor**) y la posición de la celda donde lanzará la poción.

3.3.3 Tratamiento de incertidumbre

Debido a que el agente no conoce todo el terreno debe tener implementado un mecanismo de tratamiento de incertidumbre para decidir hacia qué habitación dirigirse

en cada momento. El agente en caso de encontrarse ante el problema de no tener casillas seguras en las posiciones adyacente entonces debe seleccionar la opción que más posibilidades de ganar tenga. Para esto tendrá en cuenta las clasificaciones realizadas en el razonamiento. Este análisis se lleva a cabo en el método **Revisar.**

Cada celda tendrá tres variables booleanas (segura, seguroSiWumpus y seguroSiPrecipicio) y ocho posibles estados:

E_1: vacía E_3: brisa E_5: poción E_7: Wumpus

E_2: null E_4: hedor E_6: resplandor E_8: precipicio

El índice de peligrosidad (indp) de cada celda será determinado de la forma siguiente:

$$indp = \begin{cases} 2 & \text{para } E_i \setminus i = \{2, \ldots, 6\} \\ 5 & \text{para } E_i \setminus i = \{7, 8\} \\ 1 & \text{para } E_1 \setminus \text{segura} = \text{true} \\ 4 & \text{para } E_1 \setminus \text{seguroSiWumpus V seguroSiPrecipicio} = \text{true} \end{cases}$$

Para evitar que el jugador realice el mismo camino de forma infinita con el objetivo de no morir, se define la variable **Visitada** como un entero. Cada vez que se visite dicha casilla el valor de esta variable aumenta. Por lo tanto, se determina que el valor de la variable **varDecision** es el índice de peligrosidad visto anteriormente más el valor de la variable visitada por 0.5.

varDecision = indp + visitada * 0.5

Se multiplica por 0.5 para disminuir el valor de la variable visitada y que al comparar en el método **GetDecision** la cantidad de veces visitada no tenga mayor peso que el estado de la celda.

3.3.4 Aprendizaje

A medida que el agente realiza el razonamiento para tomar una decisión tendrá en cuenta lo aprendido. Si visitó una casilla la información de la misma se almacena en su base de conocimiento y con esto podrá determinar qué acción realizar. Se muestra entonces que

la información que conoce es aprendida por el agente y utilizada para lograr el objetivo del juego. El aprendizaje se ve más concretamente en el método **Revisar** el cual permite eliminar ruido en el análisis realizado. Se determinan las inconsistencias según las reglas del juego:

1. Al definirse que una casilla no tendrá más de dos estados, si encuentra una casilla de la base de conocimiento marcada como **seguroSiWumpus** y una de sus adyacentes con estado **Brisa**, **Null**, **Resplandor** o **Pocion** entonces la casilla analizada pasa a estar marcada como **segura**. Si dos de las adyacentes a esta presentan como estado **Hedor** entonces la celda tendrá estado **Wumpus** y la variable **wumpusEncontrado** será marcada como verdadera. Lo mismo sucede para una casilla marcada como **seguroSiPrecipicio**; pero en este caso se marcará el estado **Precipicio** si la celda presenta tres celdas adyacentes con precipicio, pues pueden existir más de uno.
2. En el juego solo existe un Wumpus, por lo tanto, si ya se definió en qué casilla está el Wumpus las demás que estén marcadas como **seguroSiWumpus** estarán ahora marcadas como **segura**. También se modificará la variable **wumpusEncontrado** como verdadera.
3. En las posiciones adyacentes al Wumpus hay hedor y en las adyacentes al precipicio se percibe brisa, por lo tanto, al definirse la casilla del Wumpus entonces las adyacentes a él, aunque no hayan sido visitadas tendrán ahora como estado **Hedor**. Lo mismo sucede para el caso del precipicio.

3.4 Pruebas de aceptación de la propuesta de solución

Para dar cumplimiento a la cuarta etapa propuesta por el método ingenieril seleccionado se llevan a cabo las pruebas necesarias para validar el videojuego creado. Realizarle pruebas de aceptación al software garantiza la calidad del mismo. Consisten en evaluar el diseño e implementación del sistema en pos de disminuir errores. Para realizar las pruebas se cuenta con el diseño del videojuego, las especificaciones de mecanismos y el compilado del videojuego. A continuación, se presentan las dos actividades a realizar que constituyen la elaboración de las pruebas de validación (9):

1. Desarrollar pruebas Alfa: el juego será evaluado por sus desarrolladores para registrar los errores y problemas que observa.
2. Desarrollar pruebas Beta: el juego será jugado por varios usuarios en los lugares de trabajo de los clientes sin la presencia activa del desarrollador. El usuario es quien registra los problemas vistos en el juego.
3. Registrar defectos durante las pruebas realizadas: esta actividad se lleva a cabo en las actividades 1 y 2.

3.4.1 Desarrollo de pruebas Alfa

Teniendo en cuenta el documento de diseño de videojuego se llevan a cabo las pruebas alfa. Se realiza una primera iteración obteniéndose 8 no conformidades que son resueltas para una segunda iteración. A continuación, se muestran las no conformidades obtenidas.

No conformidades de tipo estética y diseño:

NC1: En la pantalla de **Ayuda** la flecha de siguiente está un poco desproporcionada.

- Solución: se rediseña la flecha de siguiente para las pantallas de **Ayuda**.

NC2: La palomita que señala que se logró el objetivo no se ve bien.

- Solución: se modifica el color de la palomita para que contraste con el fondo oscuro del videojuego.

NC3: Realizar mejor la representación del hedor.

- Solución: se adicionan árboles a las habitaciones que representan brisa. Dichos árboles contienen una animación que les permite moverse al sentir la brisa que proporciona un componente de Unity con esta función.

NC4: Mejorar la ambientación de las habitaciones.

- Solución: se incorporan nuevos modelos a las distintas habitaciones.

No conformidades de usabilidad:

NC5: El giro del personaje es muy sensible.

- Solución: se disminuye la variable de la velocidad de giro.

NC6: Poner sonido de brisa para poder identificar mejor las habitaciones con brisa.

- Solución: se crea un *Audio Source con un sonido de viento* en cada habitación con brisa. El audio se activa cuando el jugador está en la habitación y se desactiva al salir de esta.

NC7**:** El personaje camina un poco lento.

- Solución: se aumenta la variable relacionada con la velocidad del jugador.

No conformidades de funcionalidad:

NC8**:** La guardiana no dispara.

- Solución: se modifica el *script* de disparo para la guardiana.

Segunda iteración de pruebas:

En una segunda iteración se presenta el videojuego y como resultado no se obtienen no conformidades. En la Figura 10 se puede observar el resultado de la realización de las pruebas alfa:

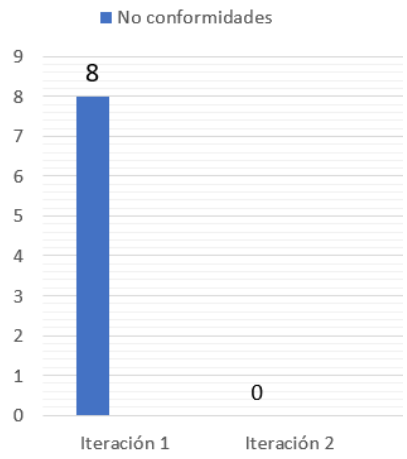


Figura 10. Resultado de las pruebas alfa.

3.4.2 Desarrollo de pruebas Beta

Para la realización de las pruebas beta se tuvo en cuenta opiniones de variados tipos de usuarios. Es por ello que el grupo escogido integra a personas que han desempeñado diferentes roles en el proceso de desarrollo de videojuegos, estudiantes y otros usuarios sin experiencia en el desarrollo de videojuegos, siendo un total de 10 personas. Se realiza una primera iteración donde se obtienen 12 no conformidades:

Primera iteración de pruebas:

No conformidades de tipo estética y diseño:

NC1: Poca visibilidad del botón cerrar en los carteles de felicitación por lograr el objetivo.

- Solución: se rediseña el botón cerrar para estos carteles.

No conformidades de tipo ortográfico:

NC2: Redacción confusa en el cartel que se muestra para lanzar la poción.

- Solución: se redacta de forma más detallada y entendible la información.

NC3: Error de redacción en las pantallas de **Ayuda**.

- Solución: se corrigen los errores de redacción especificados.

No conformidades de funcionalidad:

NC4: El jugador a veces puede alejarse de la habitación del precipicio luego de abrirla y no perder el juego.

- Solución: se aumenta el *collider* del objeto precipicio.

No conformidades de usabilidad:

NC5: En las pantallas de **Ayuda** no hay una opción para ir hacia atrás.

- Solución: se adiciona una flecha que permita ir para la pantalla anterior a la presente dentro de la opción **Ayuda**.

NC6: El mapa no muestra hacia dónde se dirige el jugador.

- Solución: se rediseña la imagen del jugador en el mapa para que sea una flecha, lo cual permite saber la dirección del jugador, aunque este no haya cambiado de posición.

NC7: El usuario nunca sabe quién es el Wumpus y el porqué de la misión.

- Solución: se agrega una pantalla en la opción de **Ayuda** donde se explican estos elementos.

NC8: Se muestra el mismo cartel de **Game Over** para las distintas muertes.

- Solución: se diseña un cartel distinto para cada caso en el que el jugador pierde el juego.

NC9: Mejorar el disparo, hacerlo más potente.

- Solución: se le incorpora un sonido al disparo y partículas que simulen el efecto de fuego. Por último, se aumenta la variable relacionada con la velocidad del disparo.

NC10: El mapa del **Modo Agente** no muestra el análisis realizado por el agente.

- Solución: se diseña una imagen para cada variable de la celda y según el análisis realizado por el agente se activan o desactivan las imágenes. Se logra con esto que el usuario pueda ver en tiempo real cómo piensa el agente inteligente.

NC11: Cambiar los controles para poder tener un mejor desenvolvimiento para girar y disparar.

- Solución: se utiliza del paquete **Standard Assets** el **FirstPersonCharacter** mediante el cual el jugador gira con el movimiento del ratón, se mueve hacia izquierda y derecha con las flechas izquierda y derecha o las teclas A y D respectivamente, se mueve hacia adelante y hacia atrás con las flechas correspondientes o con las teclas W y S respectivamente; y dispara con el botón izquierdo del mouse, a lo que se le agrega la posibilidad de disparar con la barra de espacio.

NC12: Que la pantalla **Game Over** tenga la opción de comenzar a jugar de nuevo.

- Solución: se adiciona un botón en la pantalla **Game Over** que permite comenzar a jugar sin necesidad de ir a la pantalla principal de nuevo.

Segunda iteración de pruebas:

En una segunda iteración se presenta el videojuego con las no conformidades de la iteración anterior resueltas y como resultado no se obtienen no conformidades. En la Figura 11 se puede observar el resultado de la realización de las pruebas beta:

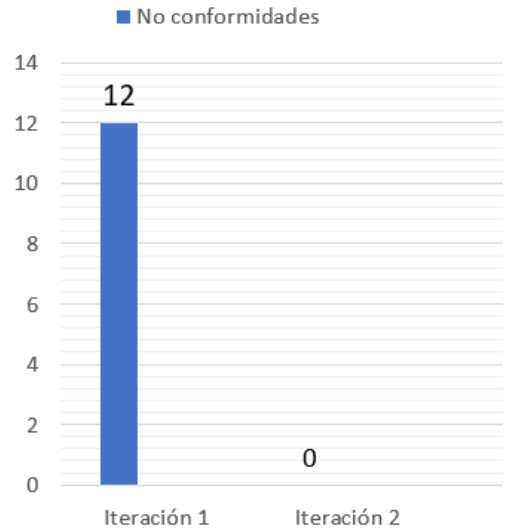


Figura 11. Resultados de las pruebas beta.

3.5 Conclusiones parciales

Mediante el presente capítulo se da cumplimiento a la propuesta de solución realizada, garantizando su culminación y validación. El estándar de codificación utilizado, así como los diagramas de componentes referentes a los mecanismos establecidos, contribuyeron a la correcta implementación del videojuego, específicamente a una buena organización y comprensión del código.

Las estrategias de Inteligencia Artificial utilizadas en la implementación y explicadas de forma detallada, garantizan el cumplimiento a los 4 núcleos de conocimiento de esta rama de las Ciencias de la Computación. Este es un material de consulta para el entendimiento del videojuego y el empleo de diversas técnicas de IA en el proceso de desarrollo de los videojuegos.

Las pruebas de tipo Alfa y Beta permitieron eliminar los errores presentes en la aplicación para obtener un juego con alta calidad y aceptación por parte de los usuarios.

Conclusiones

Las técnicas de Inteligencia Artificial, herramientas, tecnologías y metodología seleccionadas permitieron el desarrollo del videojuego El mundo de Wumpus. Un videojuego de código abierto que aplica técnicas y estrategias de Inteligencia Artificial, útiles para estudiar y reutilizar el código fuente y la documentación, en la realización de nuevos videojuegos.

La incorporación de técnicas tradicionales de Inteligencia Artificial para videojuegos, como las máquinas de estados y los comportamientos de locomoción, ya incluidas en el motor Unity, permitieron crear un oponente en el modo jugar con comportamiento adecuado.

El modo agente implementado, permite visualizar cómo un agente inteligente logra los objetivos del juego, mediante el cumplimiento de los 4 núcleos de conocimiento de la Inteligencia Artificial: la representación del conocimiento, el razonamiento, el tratamiento de la incertidumbre y el aprendizaje.

Como resultado final de la investigación realizada se obtuvo el videojuego El mundo de Wumpus. El cual cumple con los requerimientos definidos inicialmente, está avalado por el conjunto de pruebas realizadas y la satisfacción de los usuarios, con lo cual se da cumplimiento al objetivo inicial de la investigación.

Recomendaciones

Como recomendaciones para el desarrollo de futuras versiones se tienen las siguientes:

- Adicionar nuevos niveles y mapas para cada nivel.
- Habilitar la aplicación para utilizarse en el sistema operativo Android.

El mundo de Wumpus es un juego clásico para la enseñanza de la IA, el cual se utiliza en libros de esta disciplina. Por ejemplo, en el libro *Artificial Intelligence. A modern Approach*, los autores lo emplean como ejemplo en varios de sus capítulos. Es por esto que a pesar de que el videojuego no presenta un fin didáctico se recomienda ser utilizado por los profesores que imparten las asignaturas relacionadas con la disciplina de IA.

Referencias bibliográficas

1. **Javier, Díaz, Queiruga, Claudia y Fava, Laura.** *Juegos serios y educación.* La Plata : Facultad de Informática. Universidad Nacional de La Plata , 2015.

2. **Prieto, Rafael y Medina, Nuria.** *Juegos serios: mapeo sistemático y taxonomías para su clasificación.* Granada : Universidad de Granada, 2015.

3. **Sampedro, Begoña y McMullin, Karen.** *Videojuegos para la inclusión educativa.* Barcelona : Digital Education Review, 2015.

4. **García, Ramón, y otros.** *Propuesta de Articulación de Temas de Sistemas Inteligentes en la Currícula de Licenciatura en Sistemas.* Lanús, buenos Aires : Universidad Nacional de Lanús, 2016.

5. *La Inteligencia Artificial y sus contribuciones a la Física Médica y la Bioingeniería.* **Chacón, José, Flórez, Anderson y Rodríguez, Enrique.** 9, La Playa, Colombia : MUNDO FESC, 2015, Vol. 1. 2216-0353.

6. **Russell, Stuart y Norvig, Peter.** *Inteligencia Artificial Un enfoque moderno.* Madrid : Pearson Educación, 2004.

7. *De la Usabilidad a la Jugabilidad: Diseño de Videojuegos Centrado en el Jugador.* **González, Jose Luis, y otros.** Albacete : Grupo LoUISE, 2008. IX Congreso Internacional Interacción.

8. **Stenros, Jaakko.** *The Game Definition Game: A Review.* Tampere : University of Tampere, 2016.

9. **Hernández, Andy.** *Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos.* La Habana, Cuba : Universidad de las Ciencias Informáticas, 2017.

10. **Tejeda, Albert.** *Introducción al Diseño de Videojuegos.* Cataluña : Universidad Abierta de Cataluña.

11. *El videojuego como herramienta educativa. Posibilidades y problemáticas acerca de los serious games.* **López, Cristian.** 1, Guadalajara : Universidad Pedagógica Nacional, 2016, Vol. 8. 2007-1094.

12. **Gutiérrez, Álvaro.** *Hibridación de bots 'humanizados' de Unreal Tournament 2004 mediante técnicas de Inteligencia Computacional.* Málaga : Escuela Técnica Superior de Ingeniería Informática, 2017.

13. **Parra, Álvaro.** *Búsqueda de caminos en mapas de videojuegos.* Madrid : Universidad Carlos III de Madrid, 2015.

14. **Chicas, Rudy Wilfredo, y otros.** *Investigación aplicada al área de Inteligencia Artificial y desarrollo de un sistema experto.* San Salvador : Escuela de Ingeniería de Sistemas Informáticos de El Salvador, 2004.

15. *Metodologías actuales de desarrollo de software.* **Rivas, Carlos Ignacio, y otros.** 5 980-986, Pachuca, México : Instituto Tecnológico de Pachuca, 2015, Vol. 2.

16. **García, Manuel José.** *Estudio comparativo entre las metodologías ágiles y la metodologías tradicionales para la gestión de proyectos software.* Oviedo : Universidad de Oviedo, 2015.

17. *Marco de trabajo ingenieril para el proceso de desarrollo de videojuegos.* **Hernández, Andy, Pérez, Karina y Correa, Omar.** 1, La Habana, Cuba : Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software, 2017, Vol. 7. 2248-7441.

18. **Indrie, Liliana y Buzle, Marius.** *Adobe Illustrator and GIMP an approach garment design.* Oradea : Universidad de Oradea, 2016.

19. *Recorrido Virtual en tercera dimensión de la sede principal en una universidad de Bogotá.* **Nieto, Yuri Vanessa, López, José Fernando y González, Claudio Camilo.** , Bogotá : Revista Especializada en Ingeniería (UNAD), 2015, Vol. 10. 1900-6608 .

20. **Palamar, Todd.** *Mastering Autodesk Maya 2016.* Forid : Autodesk official press, 2016. 978-1-119-05982-0 .

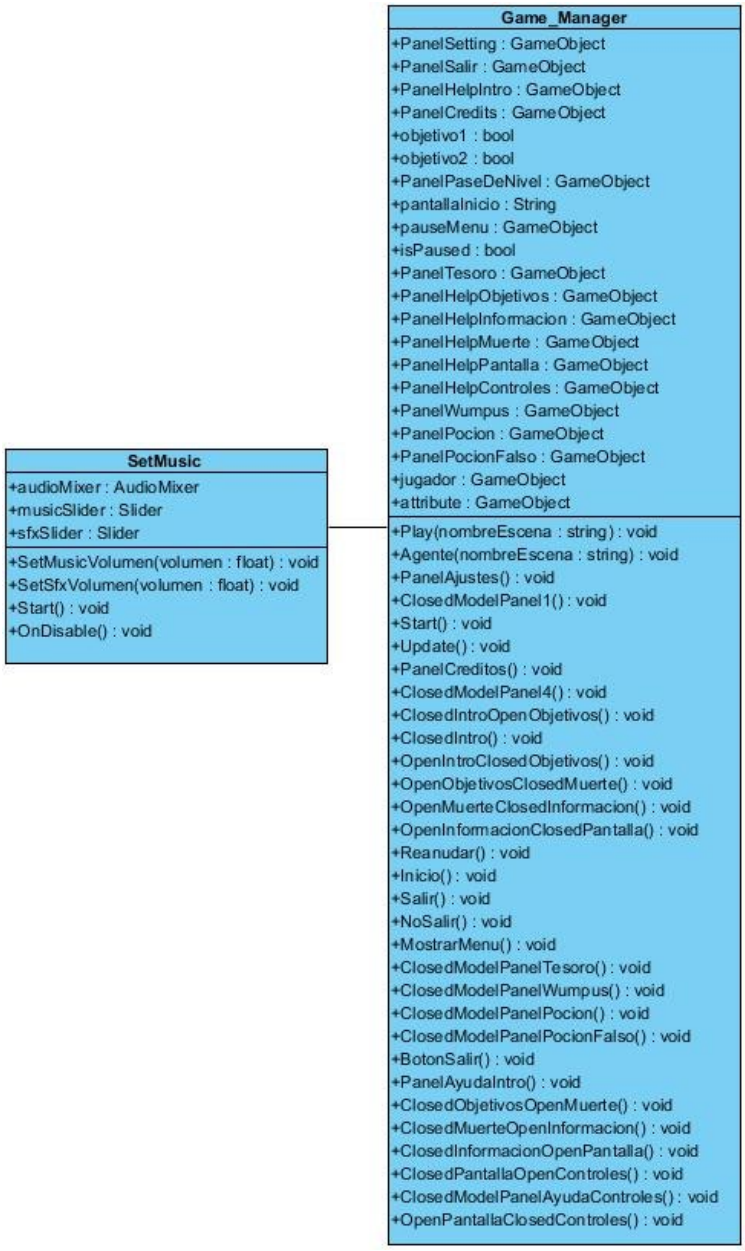
21. **Nieto, Alba C.** ARQUITECTURA POR COMPONENTES JEE, UN CASO PRÁCTICO. *Gerenc. Tecnol. Inform.* Trimestral, 2014, Vol. 14, 38.

22. **Pressman, Roger.** *Ingeniería de Software Un enfoque práctico.* Connecticut : University of Connecticut, 2010. 9786071503145.

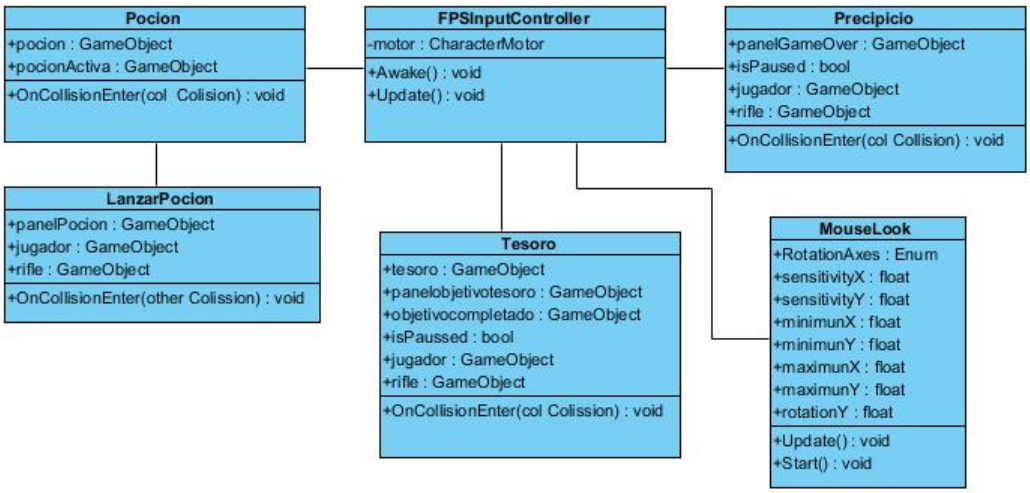
23. **Sommerville, Ian.** *Ingeniería de Software.* s.l. : Pearson Education, 2011. 9780137035151.

Anexos

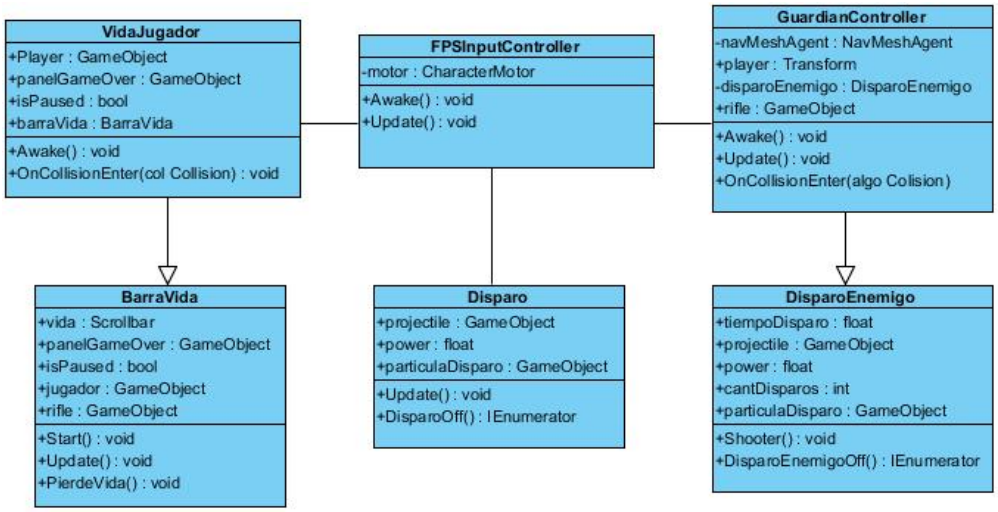
Anexo 1 Diagrama de clases Mecanismo Control del menú principal, Ajustes y Menú



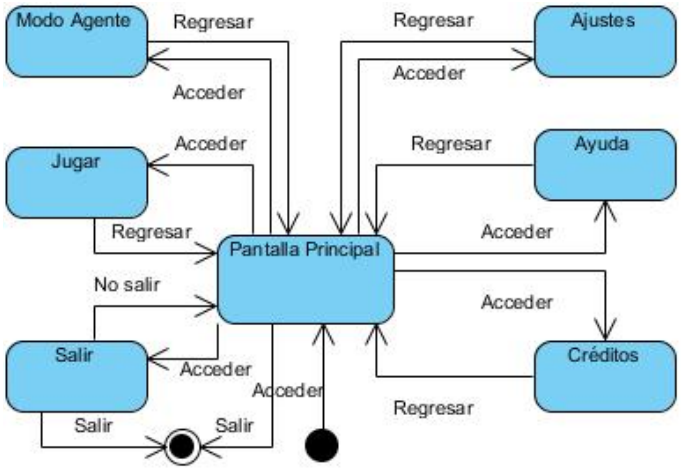
Anexo 2 Diagrama de clases Mecanismos Búsqueda de la poción, Búsqueda del tesoro, Matar al Wumpus



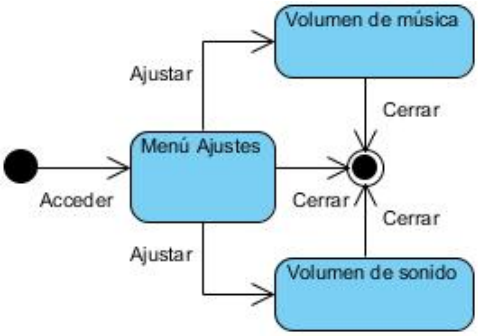
Anexo 3 Diagrama de clases Mecanismos Disparar y Vida



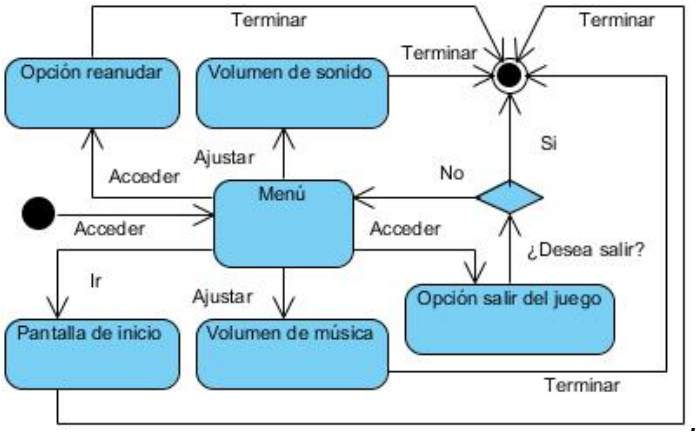
Anexo 4 Diagrama de estados Mecanismo control del menú principal



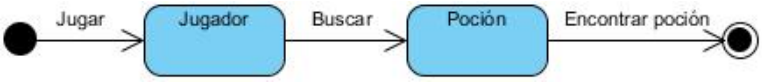
Anexo 5 Diagrama de estados Mecanismo Ajustes



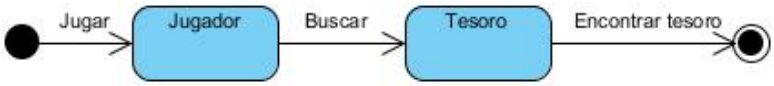
Anexo 6 Diagrama de estados Mecanismo Menú



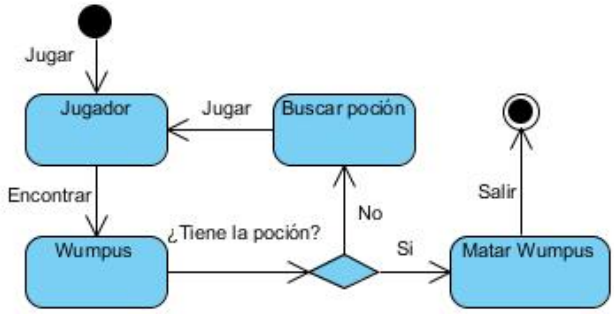
Anexo 7 Diagrama de estados Mecanismo búsqueda de la poción



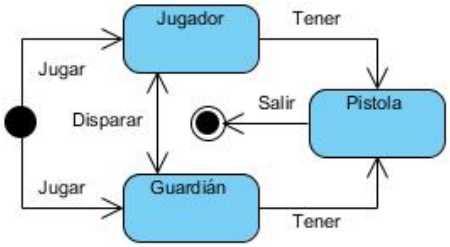
Anexo 8 Diagrama de estados Mecanismo búsqueda del tesoro



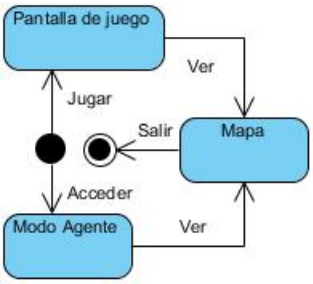
Anexo 9 Diagrama de estados Mecanismo matar al Wumpus



Anexo 10 Diagrama de estados Mecanismo disparar



Anexo 11 Diagrama de estados Mecanismo Mapa



Anexo 12 Diagrama de estados Mecanismo Vida