Universidad de las Ciencias Informáticas

Facultad 5

Estructura para la incorporación de Inteligencia Artificial en videojuegos y simuladores

Trabajo de diploma para optar por el título de Ingeniero

en Ciencias Informáticas

Autor(a): Lisbeth Marina Velázquez Benítez.

Tutor(a): MsC. Marvyn Amado Márquez.

Cotutor(a): Ing. Andy Trujillo.

La Habana, 2011.

DECLARACIÓN DE AUTORÍA

Yo, Lisbeth Marina Velázquez Benítez, declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____	_____
Firma del Autor	Firma del Tutor
Lisbeth Marina Velázquez Benítez	MsC. Marvyn Amado Márquez

Lo importante en ciencia no es tanto obtener nuevos hechos como descubrir nuevas formas de pensar sobre ellos.

William Lawrence Bragg

Datos del contacto

MsC. Marvyn Amado Márquez Rodríguez (mamarquez@uci.cu).

Máster en informática aplicada, líder de la línea de desarrollo "Navegación y Comportamiento Inteligente" enero-junio 2011. Profesor instructor, graduado de Ingeniero en Ciencias Informáticas en el 2006, 3 años de experiencia impartiendo la asignatura de Inteligencia Artificial en la Facultad 5.

AGRADECIMIENTOS

A mi familia, por apoyarme siempre.

A mi tutor, por el tiempo y la paciencia.

A Aroldo, por tratarme como una hija.

A los Pelusas, por soportarme en el cuarto.

A los profes Yadira y Yenifer, por ayudarme cuando lo necesité.

A Sari, Lupe, Dennis por ser como unas madres para mí en todos estos años.

A mis amigos de la UCI Anay, Yadira, Yoandra, Daylen, Maylín, Dalia, Gretel, Daisy y Gustavo.

A mis amigos de Banes Gabdiel, Yucelis, Yuliet y Katia por no dejar que la distancia afectara nuestra amistad.

A todos aquellos que de cierta forma contribuyeron a hacer de mí la joven que soy ahora.

A todos ellos gracias.

DEDICATORIA

A mis padres, por apoyarme a pesar de mis errores y no matar mis sueños.

A Gelson, por aguantarme en los momentos buenos y malos, por siempre estar.

RESUMEN

Los videojuegos y simuladores producidos en la Facultad 5 tienen incluida Inteligencia Artificial en diferentes grados de complejidad. Pero se hace imposible la reutilización del código de los mismos, porque todos tienen mucha dependencia del videojuego o simulador para los que fueron creados. Además, en la línea “Navegación y Comportamiento Inteligente” no existe ninguna documentación que sirva de guía para que los nuevos desarrolladores de videojuegos o simuladores puedan orientarse. Por lo que se propone como objetivo fundamental el diseño de una estructura adecuada para el desarrollo de módulos de IA en los videojuegos y simuladores a desarrollar en la Facultad 5.

Para obtener los resultados esperados se hizo un análisis de las características que deben tener los motores de Inteligencia Artificial, así como las ventajas y dificultades presentadas en los ya desarrollados. Como fruto de esta investigación se obtuvieron los conocimientos necesarios para desarrollar la solución.

Se diseñó un modelo de clases, que permite la incorporación de Inteligencia Artificial, no solo a un agente independiente sino a un grupo de ellos. Además, la estructura resultante puede ser adaptable a distintos tipos de entidades en los entornos virtuales.

Palabras clave: agente, Inteligencia Artificial, simulador, videojuegos

ÍNDICE DE CONTENIDO

INTRODUCCIÓN **1**

CAPÍTULO 1 ..... **4**

FUNDAMENTACIÓN TEÓRICA **4**

1.1. Inteligencia Artificial 4
1.2. Agentes en videojuegos y simuladores 5
1.2.1. Sistemas multiagentes(MAS): 5
1.3. Las Inteligencia Artificial en los distintos tipos de videojuegos 6
1.4. Principales módulos que pueden conformar el diseño de una biblioteca de Inteligencia Artificial 10
1.5. Navegación en agentes inteligentes 12
1.5.1. Representación del entorno 12
1.5.2. Búsqueda de caminos 13
1.5.3. Técnicas para definir el movimiento de los agentes inteligente 14
1.6. Sistema sensorial 16
1.7. Toma de decisiones 16
1.7.1. Técnicas utilizadas para la inteligencia de táctica y estrategia 17
1.7.2. Otras técnicas de toma de decisiones 17
1.8. Motores de IA utilizados en videojuegos y simuladores 20
1.9. Patrones de diseño 21

CAPÍTULO 2 ..... **24**

PRINCIPALES COMPONENTES DE LA ESTRUCTURA **24**

2.1 Partes fundamentales que debe tener un sistema inteligente 24
2.2 Estructura básica 25
2.2.1 Forma de comunicación entre cada uno de los componentes 28
2.3 Modificaciones para la inclusión de inteligencia táctica, de estrategia y aprendizaje en los videojuegos o simuladores 31
2.4 Otras consideraciones válidas a la hora de diseñar un juego o simulador 37

CAPÍTULO 3 ..... **44**

RESULTADO DE LA INVESTIGACIÓN **44**

3.1 Resultado de la investigación realizada 44
3.1.1 Patrones de diseño empleados en la elaboración de resultado 48
3.2 Herramienta y lenguaje de modelado utilizados en el desarrollo de la propuesta de solución 49
3.3 Diferencias fundamentales entre la estructura propuesta y las estudiadas en el Capítulo 1 49
3.2 Aplicación del diseño base extendido a Videojuegos Conocidos 53

CONCLUSIONES **58**

RECOMENDACIONES **59**

REFERENCIAS BIBLIOGRÁFICAS **60**

ANEXOS 62

ANEXO 1 : ARQUITECTURA DE FRANK PUIG PARA UN SISTEMA DE PERCEPCIÓN 62
ANEXO 2: DIAGRAMA DE CLASES PARA EL PAQUETE GESTIÓN DEL MUNDO JUEGO "LABERINTO DEL SABER" 63
ANEXO 3: DIAGRAMA DE CLASES DEL PAQUETE GESTIÓN DE ACTOR VIDEOJUEGO "LABERINTO DEL SABER" 64
ANEXO 4: DIAGRAMA DE CLASES DE LA BIBLIOTECA "LIBRARY ON CROWD BEHAVIORS FOR VIDEOGAMES" 65
ANEXO 5: DIAGRAMA DE CLASES DEL MÓDULO DE IA "SIMULADOR ADUANA" 66

GLOSARIO DE TÉRMINOS 67

ÍNDICE DE FIGURAS

Figura 1 : La inteligencia artificial y el cerebro humano.....11

Figura 2: Modelo de IA propuesto por Ian Millington 11

Figura 3: Relación entre los diferentes componentes de un sistema inteligente 26

Figura 4: Ambiente del juego "Salamina".....27

Figura 5: Estructura básica para la introducción de inteligencia en agentes inteligentes.....29

Figura 6: Diagrama de clases para el ejemplo del perro.....30

Figura 7: Estructura para la incorporación de inteligencia de estrategia en agentes inteligentes.....33

Figura 8: Diagrama de clases para el Caso de estudio # 3.....34

Figura 9: Estado del campo 35

Figura 10: Estado del campo en el momento que el jugador ocho tiene la pelota.....36

Figura 11: Inclusión de los conceptos memoria y búsqueda de caminos 39

Figura 12: Diagrama de clases para el Caso de estudio #4.....41

Figura 13: Modelo de clases del sistema.....44

Figura 14: Representación de los Modelos Cognitivos 45

Figura 15: Diagrama de clases de una entidad inteligente.....46

Figura 16: Secuencia de acciones de una entidad inteligente 47

Figura 17: Juego Assassin´s Creed II 54

Figura 18: Tom Clancy's Splinter Cell, Conviction.....56

INTRODUCCIÓN

El desarrollo del software ha tenido un avance vertiginoso en los últimos tiempos. Cada día son más las industrias que desean automatizar los procesos dentro de sus empresas, con el propósito de agilizarlos y hacerlos menos trabajosos. Bajo esta premisa, lograr un producto informático de calidad es una nueva pauta que se ha de trazar y debe seguir todo desarrollador de sistemas.

Actualmente se incluyen temáticas como los gráficos por computadora y la Inteligencia Artificial, a las aplicaciones que se producen para varias esferas, como la salud, la educación y el entretenimiento. Ejemplo de ello son los videojuegos y simuladores que constituyen una actividad compleja en el desarrollo de software y tienen gran aceptación por parte de los usuarios.

Cuba, se dedica a la realización de proyectos informáticos que incluyen la realidad virtual y dentro de ella, como objetivo primordial, la producción de videojuegos y simuladores. La Universidad de las Ciencias Informáticas (UCI) ha visto un renacer en la producción de juegos de computadoras. Algunos de ellos pertenecientes a la Facultad 5, han sido premiados en el Festival Nacional de Videojuegos, de agosto del año 2009, entre los que se encuentran “Fernanda” y “Rápido y Curioso”.

Los videojuegos y simuladores producidos por la Facultad 5, incluyen en su estructura Inteligencia Artificial (IA) como componente fundamental. Los integrantes de la línea “Navegación y Comportamiento Inteligente”, quienes se encargan de incorporarle la Inteligencia Artificial a los mismos, se han percatado que muchos de ellos implementan arquitecturas similares pero usando un motor de inteligencia específico para cada producto. Las arquitecturas implementadas se vuelvan casi imposibles de reutilizar en su mayoría, porque las clases de inteligencia dependen, en gran medida, con las clases del videojuego o simulador para las que fueron creadas. Las dificultades anteriores implican un gasto innecesario de tiempo y esfuerzo por parte de los nuevos programadores cuando se enfrentan a problemas similares.

Por todo lo antes expuesto, se formula el siguiente **problema científico**: ¿Cómo incorporarle Inteligencia Artificial (IA) a los agentes empleados en los videojuegos y simuladores a desarrollar en la Facultad 5?

El cual delimita como **objeto de estudio:** La Inteligencia Artificial en videojuegos y simuladores.

El objetivo general de este proyecto es: Proponer un diseño adecuado para el desarrollo de módulos de IA en los videojuegos y simuladores a producir en la Facultad 5.

Lo que determina que se proponga como **campo de acción:** El análisis de las características funcionales empleadas en los motores de Inteligencia Artificial desarrollados hasta el momento en videojuegos y simuladores.

Para ello se formulan las siguientes:

Tareas de investigación:

1. Elaboración del marco teórico de la investigación a partir del estado del arte existente sobre el tema.
2. Identificación de los principales tipos de videojuegos y sus características.
3. Estudio y descripción de los diferentes módulos que conforman una biblioteca de IA para videojuegos.
4. Estudio bibliográfico para determinar cuáles son las experiencias más generalizadas en la UCI sobre la realización de videojuegos y otros productos informáticos donde está presente la IA.
5. Elaboración de la propuesta de estructura de un módulo de IA para ser utilizada en la incorporación de inteligencia a agentes empleados en videojuegos y simuladores.
6. Descripción de los componentes de la estructura propuesta.
7. Elaboración de las variaciones principales, que conformaran la estructura de acuerdo a los diferentes niveles de complejidad de los videojuegos y simuladores.

En el trabajo se utilizan los siguientes **métodos de investigación**:

Métodos Teóricos

Histórico-lógico: Con el fin de profundizar en el desarrollo de las tendencias actuales en la elaboración de software de IA.

Análisis – síntesis: Con el propósito de determinar cuáles son las experiencias más generalizadas sobre la realización de software de IA.

Métodos empíricos:

Consulta de fuentes de información: se realizó la lectura de varias bibliografías del tema para tener conocimiento de lo que se ha realizado actualmente sobre lo referente a la problemática abordada.

Observación: A los estudiantes en el proceso de utilización de los productos informáticos elaborados para verificar las funcionalidades que ofrece la aplicación.

A partir de lo anterior se propone como **idea a defender:** El diseño de una estructura para incorporarle IA a los agentes empleados en los videojuegos y simuladores, facilitará el trabajo de los diseñadores en su construcción.

Estructura de los capítulos:

Capítulo 1: Fundamentación teórica: En este capítulo se relacionan los principales conceptos y categorías sobre el tema, como fundamentación para brindar los conocimientos necesarios para la comprensión del diseño que se propone.

Capítulo 2: Principales componente de la estructura: Describirá los conceptos fundamentales, que debe incluir un módulo de IA. Además, proporcionará una guía acerca de cómo se van incluyendo conceptos a la estructura principal, de acuerdo a la complejidad que se desea lograr en el videojuego o simulador.

Capítulo 3: Resultado de la investigación: Se plantea el resultado de la investigación, que da como fruto un diseño de clases, que permite el trabajo con entidades y grupos de entidades en diferentes grados de complejidad.

CAPÍTULO 1
FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se presenta un estudio valorativo sobre el concepto de Inteligencia Artificial y técnicas utilizadas para la implementación de los principales módulos que debe tener una biblioteca de IA para los videojuegos y simuladores. Además, se presentan ejemplos de juegos y simuladores que tienen incluida IA.

1.1. Inteligencia Artificial

En la definición de Inteligencia Artificial que concibe el diccionario de la Real Academia de la Lengua Española, aparece que es "*el desarrollo y utilización de ordenadores con los que se intenta reproducir los procesos de la inteligencia humana"* (Real Academia de la Lengua Española, 2011). En el libro (Russel, 1995) similar a la definición de la Real Academia plantea que "*la Inteligencia Artificial como la creación de programas de computadoras que emulen la forma de actuar y de pensar del ser humano, así como actuando y pensando racionalmente "*, pero no existe en realidad un concepto único sobre el tema.

Para lograr la incorporación de IA en los productos informáticos, se han desarrollado un grupo de técnicas como pueden ser los árboles de decisión, las máquinas de estado finitas, comportamientos de locomoción y redes neuronales. Las mismas se emplean en el campo de los videojuegos y simuladores para simular en los agentes un comportamiento natural.

La utilización de un grupo de técnicas no asegura siempre que la entidad va a actuar de forma creíble. Por lo que, si se programa un agente para que dado un problema tome una única decisión, eso no hace a la entidad comportarse de forma inteligente. Ejemplo, del logro de ilusión de inteligencia en los juegos fue el resultado del descubrimiento de los diseñadores de Halo que:

> *"Se* dieron cuenta que incrementando el número de puntos requeridos para matar a un agente los hacía ver más inteligentes. Para ello realizaron un test, que reveló en su primera ronda que el 36 por ciento de las personas creían que la inteligencia implementada en el juego era fácil y después de este pequeño cambio 0 por ciento de las personas creyeron que la inteligencia era fácil y un 43 por ciento pensaron que eran muy inteligentes". (Buckland, 2005)

Entonces, se puede decir que a diferencia de lo que muchos creen, tener una entidad que tenga inteligencia no se limita a que su comportamiento parezca natural, sino que se centra en qué hacer cuando tiene múltiples opciones para una determinada situación.

Si intentamos que un videojuego o simulador sea lo más cercano a la vida real, según la circunstancia que se presente, hay dos cosas que se deben tener muy claras, la primera: una aplicación bruta es lo menos que quieren los jugadores. La segunda: tampoco una tan inteligente que no le permita ganar o cometer errores.

1.2. Agentes en videojuegos y simuladores

El Dr. Nicholas Jennings en su discurso al recoger el premio nobel al mejor investigador en 1999 planteaba que "*los agentes constituyen el próximo avance más significativo en el desarrollo de sistemas y pueden ser considerados como la nueva revolución en el software"*

Un agente se define como una "*entidad que actúa 'de manera autónoma' en un entorno, transformándolo mediante la interrelación con otras entidades"* (Agentes Inteligentes. Aplicación a la Realidad Virtual). Otras definiciones de agente inteligente lo sitúan como" *cualquier cosa que pueda percibir el mundo mediante sensores y actuar sobre ese mundo mediante efectores*" (Rodríguez, 2011). Realizando una unión de estas definiciones podemos definir un agente como una entidad que percibe el entorno utilizando sensores, actúa de forma autónoma y racionalmente a los cambios del mismo, mediante la relación con otras entidades.

Se caracterizan por la relación que tienen con el entorno virtual a través del cual se mueven, mediante la utilización del sistema de software. También están concebidos para realizar tareas de forma autónoma a partir de la información que recoge y deben ser capaces de aprender e interactuar con otros agentes, de forma que su manera de actuar sea racional, es decir, tratando de maximizar el resultado esperado y de cumplir el fin para el cual fueron creados.

En los videojuegos y simuladores cuando se programa una entidad inteligente, la misma sería en un sistema sencillo la clase que controla la computadora en una interacción jugador – máquina. En sistemas donde es necesario manejar más de una entidad o agente se convertiría en una clase abstracta de donde heredarían cada uno de los diferentes tipos de entidad.

1.2.1. Sistemas multiagentes(MAS)

Los videojuegos y simuladores en ocasiones no necesitan controlar solamente una entidad única, sino más de un número de ellas dentro del entorno virtual. La estructura de un sistema multiagentes tiene seis componentes principales (Ferber, 1999):

- Un entorno.
- Un conjunto de objetos: Que pueblan el entorno e interactúan con los agentes.
- Un conjunto de agentes: Este conjunto está formado por un grupo de entidades, que requiere el sistema.
- Un conjunto de relaciones: Las relaciones están establecidas entre los objetos y los agentes.
- Un conjunto de operaciones: Este conjunto de operaciones son las utilizadas por las entidades para lograr la manipulación de los objetos.
- Operadores: Representan la aplicación de operaciones sobre el mundo y la reacción de éste al ser alterado. Estos operadores se pueden entender como las leyes del universo.

Los agentes que conforman el sistema deben ser capaces de tomar la iniciativa si la situación lo requiere, compartir el conocimiento, cooperar y negociar con los demás agentes; estas son características importantísimas para el buen funcionamiento de los MAS dado que todas las entidades deben tener al menos una meta en común y lograr una buena comunicación ante ellos.

El comportamiento de entidades o MAS en videojuegos y simuladores es muy semejante. Esta investigación se enfocará a estudiar el mismo en los distintos videojuegos producto de la similitud que presenta su comportamiento tanto en simuladores como en videojuegos, dado que en definitiva son agentes que se interactúan en un entorno virtual.

1.3. Las Inteligencia Artificial en los distintos tipos de videojuegos

Los videojuegos son todos aquellos software cuyo objetivo principal es el entretenimiento, pueden ser jugados por una o varias personas dependiendo de su tipo. El nivel de complejidad de los mismos varía, desde aquellos que no poseen grandes cantidades de gráficos y sonido como el Tetris, hasta otros tan complejos como *Assassin´s Creed*: *Brotherhood*, que incluye un gran número de agentes que la inteligencia debe controlar y animaciones con un elevado nivel de detalle.

Clasificación de los videojuegos:

Juegos basados en roles (RPG): Warren Spector (2004), productor de videojuegos como Ultima Underworld 1 y 2, da una definición muy acertada de los mismos: "se refieren a cómo juega un usuario con un carácter determinado en la narración de una historia" (Bates, 2004). Es decir, el jugador va a tener un rol dentro del mundo donde se desarrolla la historia y tiene que cumplir determinadas metas.

Para la implementación de estos tipos de videojuegos se utilizan varias técnicas como, scripting, máquinas de estado y sistemas de mensajería que logren la comunicación entre los miembros de su grupo de forma fácil y rápida.

En el juego *World of Warcraft* se pone de manifiesto lo expresado con anterioridad, cuando un carácter pasa de nivel, si se miembro en algún grupo, el videojuego lo comunica a los demás miembros aunque se encuentren alejados del mismo. Otros como *Wizardy*, "tienen estados estáticos para sus enemigos con un pequeño código especial, que es activado en el lugar y momento dado dentro del juego" (Schwab, 2004).

Aventura: En estos tipos de videojuegos el jugador debe realizar una serie de acciones durante el desarrollo de la historia hasta lograr un objetivo final. "En ellos se combina la exploración con la solución de *puzle* en una historia donde el jugador es el héroe "(Bates, 2004).

Actualmente, este tipo de videojuego ha evolucionado, llegándosele a incorporar elementos en tiempo real o situaciones de combates, entre ellos esta *Tom Raider* como uno de los más representativos. Los enemigos se crean con inteligencia artificial, como cuando un *not player character* (NPC) pretende haber oído algo cuando el jugador se acerca o cuando llama refuerzos. Técnicas como las máquinas de estado finitas, los script y la lógica difusa se emplean en estos tipos de juegos para lograr determinados grados de realismo en los mismos. Ejemplo de ello es *Herzog Swei* que "utiliza como técnica una máquina de estado simple con los estados *begin, get Money, attack*, defensa "(Schwab, 2004).

Estrategia: Los videojuegos de estrategia plantean un escenario donde el jugador debe decidir la estrategia que seguirá para conseguir un objetivo. Están conformados por dos equipos opuestos que tienen la misma oportunidad de ganar. "La IA del juego no puede hacer que el oponente haga lo mismo siempre o que lo haga muy difícil o muy fácil "(Bates, 2004). La Inteligencia Artificial de estos juegos es realmente compleja, por que envuelve gran cantidad de unidades y tecnologías que coordinar. Además la misma debe "estar cercana al nivel de jugadores humanos con calidad, tiene que manejar información incompleta y correr en tiempo real" (Schwab, 2004).

El ajedrez es un ejemplo de juego de estrategia clásico donde algunos implementan el algoritmo Minimax con poda alfa-beta para llegar a las soluciones o algoritmos genéticos para facilitar la búsqueda de nuevas soluciones.

Ftps (primera persona disparando/tercera persona disparando): Estos juegos incluyen herramientas que las personas pueden utilizar para adicionar niveles y cambiar armas. Se caracterizan por tener gran cantidad de enemigos, la inteligencia de estos es primordial. Los enemigos poco inteligentes "creados en la mayoría de los géneros deben ser remplazados por sistemas que sean capaces de casi todo lo que el jugador humano pueda hacer" (Schwab, 2004).

La mayoría de estos juegos requieren que el jugador dispare primero y en el transcurso se le proporcionan armas cada vez más poderosos o mayor salud que depende de su estructura, podemos mencionar entre ellos a *House of the Death*, *Time Crisis, Point Blank. Doom*, en ocasiones, da la oportunidad de correr con la mejor arma y destruir todo a su paso que es lo que algunos jugadores quieren.

Simulación: Como su nombre lo indica pretende competir ante los jugadores con situaciones lo más realistas posibles. "Su meta es cumplir al jugador los deseos que no podría en la vida real. Los jugadores serios quieren una simulación de la vida real y los casuales las quieren como en las películas. (Bates, 2004). Los mismos se pueden clasificar de acuerdo a lo que deseen simular:

- Deportivos: FIFA, NBA, Tenis.
- Juegos de mesa y recreativos: Trivia, Monopolio, ChessMaster, Pinball.
- Vehículos: Simulador de camión MAZ-700 expuesto por el proyecto HDSRV en la feria de Informática 2007.
- Lucha: Street Fighter, Boxing.
- Sociales: simulan empresas u otras organizaciones con el fin de conseguir su optimización y/o supervivencia. Ejemplos: SimCity, Transport Tycoon, Antz.
- Bélicos. Ejemplos: WarCraft, Age of Empires, Command and Conquer.

Entre las técnicas de inteligencia que pueden ser de utilidad para la implementación de estos videojuegos están: máquina de estados difusa porque son fáciles de tratar y proveer patrones de comportamiento; red neuronal para encontrar mejores formas de entrenamiento del juego y de ver cuando un comportamiento es erróneo y algoritmo genético, pues facilitan engendrar programas y ayuda a la generación de los caracteres del videojuego para que evolucionen en varias formas (Schwab, 2004).

Deportivos: Como su nombre lo indica son videojuegos que recrean un deporte, como fútbol, baloncesto, tenis. Estos juegos pueden clasificarse “como juegos de deportes fluidos donde el juego es rápido, dinámico y continua por un largo período, con algunas paradas en el camino y juegos de restablecimiento que se hacen cuando el juego para y se restablece después de un periodo de tiempo, ejemplo Baseball” (Schwab, 2004).

En este tipo de juego los personajes fundamentales son el entrenador y los jugadores. El primero de estos se encuentra dentro del nivel de inteligencia de estrategia, pues debe saber a quién llamar en un momento dado o sustituir. Sin embargo, el jugador debe poseer niveles de inteligencia que lo hagan moverse rápido si es necesario, dejar la bola pero siempre teniendo en cuenta el movimiento escogido y hacia donde se va a dirigir. Los personajes secundarios se incluyen en el módulo de inteligencia porque las mascotas y el público deben actuar acorde al desarrollo del juego.

Carreras: Recrean una sensación de velocidad en el jugador. Se pueden conducir todo tipo de vehículos, desde coches hasta aviones, pasando por lanchas motoras o naves espaciales. Las carreras también se han visto con elementos agregados, por ejemplo: cuando se le ponen obstáculos o cuando corren en medio de disparos. Algunos juegos son construidos en ciudades reales por lo que la IA de los mismos debe controlar el tráfico de la ciudad, esto implica simular semáforos, luces de la carretera, además de los vehículos que, normalmente, transitan por ella. Entre las técnicas de inteligencia que pueden ser usadas están las máquinas de estados, el uso de script, algoritmos genéticos. Muchos de estos juegos “manejan gran cantidad de vehículos por lo que algunas compañías usan los algoritmos genéticos para automatizar este proceso” (Schwab, 2004).

Lucha: Son videojuegos donde se realiza una simulación de una lucha cuerpo a cuerpo. Este es uno de los géneros que “asume que los jugadores están sentados uno al lado del otro y hablan entre ellos. Cada jugador debe tener una serie de movimientos distintos que lo hagan interesante para mirar. Debe ofrecer entretenimiento en cortos periodos de tiempo y ganar la atención suficiente del jugador juegue otro combate” (Bates, 2004).

La mayor característica que debe tener la inteligencia de este tipo de videojuegos es el balance, pues el oponente no puede ser ni muy fácil, ni imposible de vencer. Le dan al usuario un nivel de control de sus caracteres que los demás juegos no. Implementan técnicas como la detección de colisiones y la utilización de “boss” para lograr la inteligencia.

Puzzle: Son videojuegos de habilidad que tiene como reto probar las capacidades del usuario. La meta de los mismos es retar al jugador para que pruebe sus talentos, pero también ayudarlo a ganar de forma tal que no se vea como estúpido.

Hay 5 tipos de *puzzle*: de acción (Tetris, Pipe Dream), de historia (en este juego el talento es resolver el *puzzle* dentro de la historia fácilmente), competitivos (Boggle, Wheel of Fortune), construcción (The Incredible Machine), puros (largas colecciones de *puzzle* de la misma clase).

Juegos de plataforma (*Plataform Games*): Los videojuegos de esta categoría se caracterizan por la rapidez de respuestas requeridas por el usuario. Con una sucesión de metas o etapas previas para alcanzar el objetivo final. Hoy día estos juegos se basan en 3 elementos: exploración, resolución de *puzzle*, talentos físicos.

La inteligencia de estos juegos se muestra en diferentes formas, Abuse (1996) videojuego "realizado en 2d, que emplea básicamente para la inteligencia de la criatura una máquina de estado en la que se definen un estado seleccionado con varios estados incluidos. *Donkey Kong, Castlevania* usan enemigos basados en estados, a menudo basados en solo uno de ellos" (Schwab, 2004).

1.4. Principales módulos que pueden conformar el diseño de una biblioteca de Inteligencia Artificial

Lo que debe llevar un videojuego o simulador para que tenga un buen diseño, ha sido tema de discusión entre un sinnúmero de desarrolladores. John Davis Funge en su libro "*Artificial inteligentes for computers games*" (Funge, 2004) plantea que una arquitectura para videojuegos debe ser capaz de representar el estado del entorno y brindar información del mismo; contener las reglas sobre cómo deben hacerse los cambios en el estado del juego, así como poseer una estructura que permita renderizar el estado del juego.

Como se explicó anteriormente la inteligencia que se proporciona a las entidades dentro de un videojuego debe emular el comportamiento del cerebro, por lo que el sistema debería ser capaz de desarrollar algunas funcionalidades que se procesan en el cerebro humano. Brian Schwab (Schwab, 2004) plantea que "la división del cerebro, nos permite identificar cuáles serían los diferentes submódulos de un sistema inteligente".

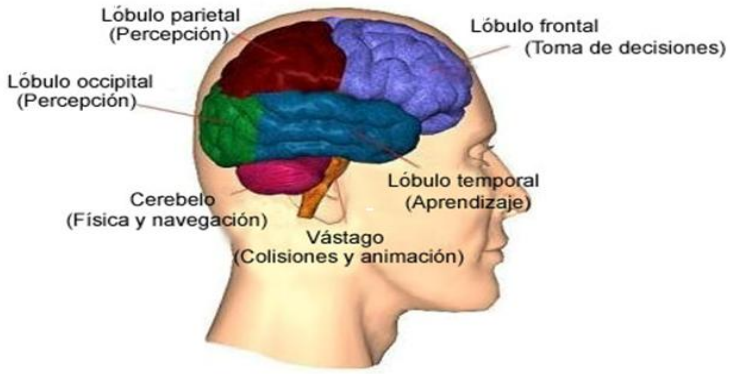


Figura 1 : *La inteligencia artificial y el cerebro humano*

Un sistema inteligente no necesariamente debe conformarse con la cantidad de módulos anteriores; sino "*lo que se debe garantizar es que tengan un tipo de infraestructura que sea capaz de administrar los comportamientos inteligentes, un sistema para garantizar información del mundo dentro de la IA y un medio para asimilar los cambios que se hagan en la pantalla. Además, necesita un estándar entre cada uno de los géneros de los videojuegos, aunque sea imprescindible agregarle un algoritmo o dos para cada uno"* (Millington, 2006). Un modelo que garantiza esta infraestructura es el propuesto en la Figura 2, que se muestra a continuación.

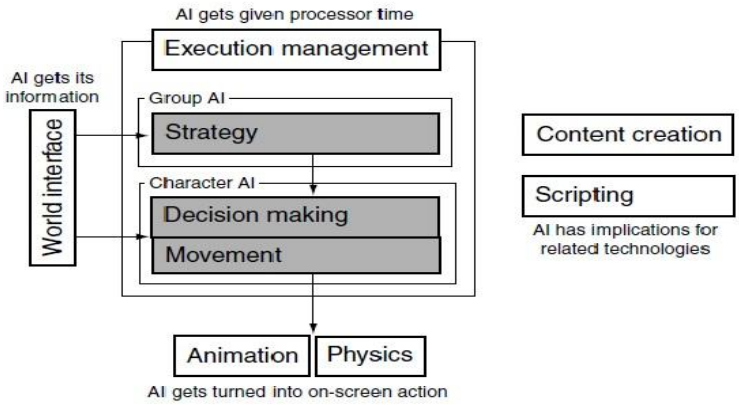


Figura 2: *Modelo de IA propuesto por Ian Millington*

Podemos concluir que se emplea como sistema básico: un sistema de navegación para mover un carácter de un lugar a otro, uno de percepción para asegurar un reconocimiento de los objetos del entorno y la

retroalimentación con el sistema de toma de decisiones e inferencia, este último sería el encargado de decidir qué acción se va a realizar y en el caso de existir más de una con posibilidades elegir la solución.

1.5. Navegación en agentes inteligentes

Los agentes inteligentes tienen movilidad y autonomía, porque deciden a donde moverse y qué algoritmo ejecutar para ello. Sin embargo, para el logro de este movimiento, no se utilizan únicamente los algoritmos de traslado que están adaptados a observar. Además de ellos, intervienen técnicas de representación del entorno y algoritmos de búsqueda de caminos, a los cuáles se dedicará también un estudio en este apartado.

1.5.1. Representación del entorno

Para que una entidad inteligente interactué solamente con los obstáculos dinámicos o estáticos del mundo que están a su alrededor, es necesaria la discretización del mismo; esto consiste en la división del entorno virtual en nodos de grafo de navegación. Para poder garantizarlo existen diversas técnicas como: grafo de camino, grafo finamente granulado, rejilla regular y mallas de navegación.

Grafo de camino

Un grafo se representa mediante una serie de puntos (los vértices) conectados por líneas (las aristas). Cada vértice en el grafo representa un estado libre y cada arista representa un camino libre de colisiones entre dos estados. Estas estructuras tienen el inconveniente de que el movimiento de los agentes se ve supeditado al camino generado (Campos, 2010).

Grafo finamente granulado

Es una asociación de nodos y aristas que permite representar relaciones binarias. Tienen la ventaja de que mejoran los puntos ciegos que generan los grafos de visibilidad y mejoran el suavizado de los caminos. Al hacer uso de este tipo de grafo se presenta el inconveniente de que el número de nodos aumenta considerablemente y por consiguiente la búsqueda de caminos se hace más lenta. (Campos, 2010)

Rejilla regular

Se utiliza para dividir el mundo de polígonos iguales, donde cada una de ellas comparte un borde a lo sumo con la otra. El centro de la rejilla representa un punto en el camino y las rejas vecinas forman las

conexiones con el punto. Existen 3 tipos de rejillas: triangulares, cuadradas o de hexágonos regulares (Smed, 2006).

Las rejillas generalmente soportan consultas de acceso aleatorio, ya que cada celda debe ser accesible en cualquier instante de tiempo. No tiene en cuenta o no presta atención la geometría actual del mundo, esto provoca que puedan quedar desconectadas algunas partes del mundo si la granularidad de la rejilla regular no es lo suficientemente fina. Además, para el almacenamiento de la información de la rejilla regular se necesita memoria, aunque esto puede eliminarse haciendo uso de tablas de consultas jerárquicas. (Campos, 2010)

Malla de Navegación

Es una partición convexa en la geometría del mundo del juego. Es decir, es una serie de polígonos donde ninguno solapa al otro y están unidos en solo dos puntos y un borde. Cada polígono adyacente representa que un punto tiene conexiones con ellos, el centro de cada polígono es un punto en el camino y el centro de cada borde que se comunica también representa uno (Smed, 2006).

Entre las ventajas que nos brinda la malla de navegación está el conocimiento completo sobre las zonas navegables lo que permite que los agentes sean capaces de evitar obstáculos o seleccionar una ruta alternativa mostrando así un comportamiento autónomo (Campos, 2010).

1.5.2. Búsqueda de caminos

En un videojuego o simulador generalmente tenemos agentes virtuales que necesitan moverse por el mundo, por lo que es imprescindible implementar una manera de hacerlo. Muchos son los algoritmos que nos sirven para realizar este trabajo, pero en ocasiones no se trata solo de buscar un camino sino que este debe ser fiel a lo que el agente quiere lograr.

Los caminos dependen de un sin número de variables como son la existencia de obstáculos, si es destino estático o no, e incluso una variable muy importante es la característica del terreno sobre el cual va a moverse el carácter. Por lo que métodos simples cuya programación básica sea a base de if y else no resolverían la mayoría de nuestros problemas y menos teniendo en cuenta como están diseñados los mundos de hoy en juegos y simuladores.

Algunos algoritmos básicos para implementar la búsqueda de caminos dentro de un grafo podrían ser los llamados recorridos a lo ancho y en profundidad. Aunque existen otros con mayor complejidad, los más empleados en juegos y simuladores son los siguientes:

Dijkstra: Este algoritmo fue nombrado por el matemático Edsger Dijkstra (1959). No fue creado originalmente como un algoritmo de búsqueda de camino como normalmente se utiliza en los videojuegos, sino para resolver un problema en teoría de grafos matemáticos (Gavaldà, y otros, 2011). Encuentra el camino mínimo desde un vértice hasta los demás, pero es muy costoso porque calcula todos los caminos posibles a partir de un nodo.

Algoritmo A*: Presentado en 1968 por Peter E. Hart, Nils J. Nilsson y Bertram Raphael, es uno de los más empleados en Inteligencia Artificial, permite el trabajo con grafos dinámicos. Este es muy eficaz ya que expande la menor cantidad de nodos, fácil de implementar y tiene muchas posibilidades de optimización, además puede ser empleado para planes de movimiento complejos (Gavaldà, y otros, 2011). Es un algoritmo de búsqueda heurística, sus variaciones se usan en la implementación de búsqueda de caminos, entre ellas podemos mencionar: IDA*, SMA* y RTA*.

1.5.3. Técnicas para definir el movimiento de los agentes inteligente

Para lograr el movimiento de agentes en un entorno, todo algoritmo por sencillo que sea, tiene como entrada la posición actual del agente y hacia donde desea trasladarse, teniendo como salida una manera de representar el movimiento que se desea hacer. Los algoritmos, a pesar de que tiene el mismo objetivo, se separan en dos grupos fundamentales debido a su complejidad: los que realizan "*movimiento cinemático, este tipo de movimiento no tiene en cuenta la manera en que los caracteres aceleran y reducen la velocidad, solamente se preocupan por la dirección y llegar al blanco".* Mientras que en el segundo grupo *"están los que logran un comportamiento dinámico necesita saber además de la posición la velocidad, y tiene como salida una fuerza o aceleración que le permiten cambiar la velocidad de la entidad"* (Millington, 2006).

El epígrafe siguiente se enfocará en el estudio de los algoritmos de movimiento dinámico, por tener un grado mayor de complejidad y ser más utilizados.

Algoritmos para lograr el movimiento dinámico

Los agentes inteligentes se pueden trasladar de dos maneras: de forma individual o en manada, sea cual sea la forma en que lo hagan es necesario que tengan una dirección determinada. Estos movimientos de dirección son a los que llamamos comportamientos de locomoción.

Las diferencias entre el movimiento de los agentes de forma individual y en grupos, está dada principalmente por la información que manejan. *Craig Reynolds (1987)* nos propone en *"su algoritmo más básico para el comportamiento de manadas la visión de un agente como una unidad individual, que posee información sobre los vecinos y de acuerdo a ella se mueve"* (Gavaldà, y otros, 2011).

Tanto una unidad individual como un colectivo pueden realizar los mismos movimientos de dirección. Solamente, hay que incorporarle en la implementación de los miembros del colectivo tres parámetros fundamentales:

- **Separación:** Para evitar colisiones entre los elementos o que estos se acerquen demasiado**.**
- **Cohesión:** Se utiliza una fuerza que mantiene unido al grupo, la misma está dirigida al centro de masa de sus vecinos.
- **Alineamiento:** Su función es que los individuos vayan todos en la misma dirección.

Todos estos parámetros se implementan de forma sencilla apoyándose en el trabajo con vectores de la física.

A continuación se explicaran algunos de ellos:

Comportamientos de locomoción:

- **Flee y Seek:** La lógica de estos comportamientos es análoga, pues *Seek* se usa para dirigir un agente hacia el objetivo especificado y el *Flee* se utiliza para simular una sencilla forma de evasión, el objetivo puede ser un punto fijo o en movimiento. Todo se realiza utilizando trabajos con vectores (Buckland, 2005).
- **Arrive:** Su objetivo como el del comportamiento *Seek* es llegar a la meta, pero con la diferencia que este mientras se va acercando al agente va disminuyendo la velocidad hasta que arriba al mismo.
- **Wander:** El propósito de este comportamiento es que el agente se mueva de forma indefinida, su resultado puede simular características del terreno. Su implementación se basa como en los anteriores en el trabajo con vectores, lo que se generan son fuerzas con direcciones aleatorias.

Otros comportamientos que valdría mencionar serían el que se encargan de la persecución o el de la evasión de obstáculos, es válido aclarar también que la combinación de varios puede implementar más complejos.

1.6. Sistema sensorial

La percepción *"es mucho más de simplemente ver y oír. Abarca todas las maneras que un NPC recoge los datos sobre el mundo, la información sobre las características tácticas del ambiente, los datos topológicos y medio ambientales como esconder las manchas, situaciones de la emboscada, y las posiciones de paredes y obstáculos son muy importantes"* (Placeres). Está estrechamente ligada a los caracteres del juego y de ella depende el modo en que los mismos sienten el mundo que los rodea. Es necesaria para muchos géneros de videojuegos donde la decisión de un agente está supeditada al entorno donde se desarrolla. Hay que tener en cuenta cuando se desarrolla un sistema de percepción que la visión y el oído tienen distancias limitadas para ser percibidas, por lo que se debe filtrar sino el jugador lo verá irrealista.

Existen diversos paradigmas sobre *"cómo se implementa un sistema de percepción uno de ellos es el polling el cual plantea que el sistema realiza el mismo los cálculos de las entradas y el event donde las entradas son quienes le dicen al sistema en cuanto varió respecto al estado del juego y el sistema solo registra y anota los cambios"* (Schwab, 2004).

Algunas implementaciones comunes sobre el sistema de percepción son el *ray casting*, la generación de datos manuales, el scripting e incluso en videojuegos sencillos se pueden generar imágenes adicionales, pero un sistema de percepción en un videojuego o simulador sería la solución ideal, por lo que los informáticos se dedican a construir los mismos. Un ejemplo de ellos es el ingeniero Frank Puig cuyas investigaciones incluyen la realización de una arquitectura genérica de un sistema de percepción. **(Anexo 1)**

1.7. Toma de decisiones

La toma de decisiones es fundamental para el desarrollo de un videojuego o simulador, pues tomar una mala opción puede definir qué tan estúpida es nuestra Inteligencia Artificial. Podemos dividirla en dos ramas fundamentales de acuerdo al grado de dificultad: están las decisiones tácticas y las estratégicas que involucran mayor cantidad de agentes o las más triviales como cuando en un NPC desea alimento en el juego de los Sims.

1.7.1. Técnicas utilizadas para la inteligencia de táctica y estrategia

***Waypoint*:** Son puntos marcados por el diseñador dentro de la geometría del juego o simulador que almacenan información, dependiendo para que se esté usando (Millington, 2006). La unión de más de un punto se puede emplear también para montar ataques, teniendo a las entidades en puntos estratégicos. Además los *waypoints* pueden representar lugares de refugio, o dónde se puede lograr mayor alcance en los disparos en un combate e incluso donde una entidad perdedora pueda escapar.

Análisis táctico: Existen diferentes vías para realizar el análisis táctico de un juego de acuerdo a su complejidad. Dos de ellas son: los mapas de influencia, usado en los simuladores bélicos, pues su objetivo es determinar la influencia militar en una localización y el análisis del terreno que determina la influencia de las dificultades del terreno en una parte del mundo (Millington, 2006). El análisis puede ejecutarse antes de que el juego o simulador empiece o lentamente durante la ejecución del juego o simulador y aquellas cuyos cambios se realizan de forma muy rápida. En ocasiones, es necesario utilizar más de una de estas técnicas para lograr una simulación adecuada.

Búsqueda de caminos táctica: La base de la búsqueda de caminos táctica se realiza con los algoritmos normales, lo que a la hora de escoger el camino se vinculan las características tácticas del ambiente. Se siguen utilizando las mismas formas de discretizar el mundo que en la búsqueda regular, lo que se incorporan los conceptos de mapas de influencia, *waypoints* en la elaboración de mapas estratégicos. La única variación dentro de la implementación de los algoritmos es el cambio de la función relacionada con el costo del camino, que se extiende para que se incluyan las variaciones tácticas, así como de distancia y tiempo (Millington, 2006).

Acciones Coordinadas: Esta técnica se implementa, cuando tenemos un videojuego donde el jugador tiene un rol en el que tiene varios NPC bajo su cargo y los mismos deben actuar según las decisiones de él. Para ello algunos diseñadores agrupan la IA por niveles siendo la inteligencia del carácter el nivel más bajo, después le seguiría la del grupo y la inteligencia del jugador se pondría a la cabeza de la inteligencia, permitiendo que las decisiones de las unidades estén condicionadas por la del jugador. Otros, sin embargo programan la inteligencia de la entidad y guían al cliente para que crean que la situación que ellos programaron es esencialmente la mejor (Millington, 2006).

1.7.2. Otras técnicas de toma de decisiones

Sistemas basados en reglas: Como su nombre lo indica son sistemas que siguiendo un grupo de reglas determinados, dicen qué decisión es la más acertada. Tienen dos componentes fundamentales: la

memoria de trabajo y la memoria de las reglas. La primera de estas tiene la función de guardar los hechos conocidos y aciertos hechos por las reglas y la segunda contiene una serie de reglas if-then que operan sobre el conocimiento guardado en la memoria de trabajo (Gavaldà, y otros, 2011).

Un ejemplo de los llamados sistemas de expertos, los mismos están conformados para resolver problemas que son resueltos por expertos humanos. Poseen tres componentes: una interfaz, una base de conocimientos y un motor de inferencia (Césari, 2007).

El problema fundamental con este tipo de sistemas es que si se presentan situaciones que las reglas no contemplan el agente no sabe qué hacer, además que si son muchas reglas se vuelve engorroso de implementar.

Redes Bayesianas: Utilizan el teorema de Bayes, los grafos dirigidos y acíclicos. Se pueden construir de dos formas teniendo en cuenta la distribución de la probabilidad de las variables representadas en la red o por un conjunto de reglas asociadas a las variables que están representadas en cada arco (Gavaldà, y otros, 2011).

Para diseñar una red bayesiana se deben evaluar tres características fundamentales: definir las interacciones entre los diferentes elementos de la red, asignar los valores a cada una de las probabilidades de los posibles eventos, decidir qué preguntas se quiere formular a la red. Las redes bayesianas tienen un nivel de complejidad elevado por lo que se recomienda su uso solo en problemas específicos, siempre que no se encuentren soluciones más simples e igual de efectivas.

Árboles de decisión: Es un árbol donde se colocan las decisiones y sus consecuencias. Son utilizados en cualquier proceso que implique toma de decisiones como pueden ser los sistemas de expertos.

El mismo consta de cuatro partes fundamentales:

- Nodos internos deterministas: Nos permitirá decidir cuál rama elegir.
- Nodos internos probabilísticos: De acuerdo a un evento aleatorio, saber por dónde se va a dirigir.
- Hojas del árbol: Representan el resultado que devolverá el árbol de decisión.
- Ramas del árbol: Describen los posibles caminos.

Este algoritmo se aplicó para el demo de un juego de Tres Rayas hecho en la Facultad 5 donde mientras más difícil sea el nivel del juego mayor profundidad de exploración se realiza en el árbol, hasta llegar al ocho que es el máximo (Sardiñas)**.**

Lógica difusa: Sus bases teóricas fueron enunciadas en 1965 por Lotfi A. profesor de Ingeniería Eléctrica en la Universidad de California en Berkeley (Martín del Brío, 2001). Es un sistema que trabaja con información imprecisa o borrosa, como por ejemplo la estatura. Las entradas son un conjunto de variables que están combinadas mediante un grupo de reglas que pueden dar uno o varios resultados de salida. Esta técnica generalmente es combinada con las demás para que le sirva de apoyo a la hora del trabajo con valores borrosos. Se comenzó a aplicar lógica difusa a los sistemas con Inteligencia Artificial con el objetivo de que tuvieran más realismo y humanidad.

Máquinas de estado finitas: Es una representación abstracta de una máquina, la cual tiene estados predefinidos y cuya misión, es decir, en qué momento se cambia de uno a otro de acuerdo a un grupo de condiciones. Es fácil de programar, divide el problema en pequeños sub-problemas. Una característica fundamental es que no requiere de grandes cantidades de memoria o poder computacional, quizás por esa razón es utilizado en la mayoría de los videojuegos que se han hecho hasta el momento.

Algoritmo genético: Es un algoritmo bioinspirado, su funcionamiento se basa en el sistema genético de los seres vivos. Simula los procesos de selección, cruzamiento y mutación que tienen los seres vivientes No es conveniente su aplicación para juegos que se desarrollan en tiempo real (Rodríguez, junio 2008). En los videojuegos se puede emplear cuando existe bastante incertidumbre sobre las características del jugador. Ejemplo de ellos es en los juegos de rol si al principio el jugador es mago, los adversarios pueden evolucionar hacia técnicas que esquiven la magia, pero si de repente el jugador cambia y se hace guerrero, veremos que el comportamiento de los agentes cambiará para adaptarse al nuevo tipo de jugador (Gavaldà, y otros, 2011).

Redes neuronales: Al igual que los algoritmos genéticos están inspirados en el comportamiento de los sistemas biológicos, esta vez sobre el sistema nervioso central. Sus componentes fundamentales son las neuronas que a partir de una entrada proporciona una salida, un conjunto de entradas, una función de activación, reglas de propagación y una función de salida. Su arquitectura está compuesta por una o más capas, generalmente una capa de entrada, una oculta y una de salida y su programación tiene dos fases fundamentales la fase de entrenamiento y la de explotación. En videojuegos pueden sustituir a las máquinas de estado o cualquier sistema de reglas cuando este se hace bastante complejo. Además, es

utilizado para lograr la adaptación de los agentes inteligentes a medida que se desarrolle el videojuego o simulador. Un ejemplo son los videojuegos de coches (Colin McRae Rally) que utilizan las redes neuronales para implementar el modo como conducen los oponentes (Gavaldà, y otros, 2011)

1.8. Motores de IA utilizados en videojuegos y simuladores

La fabricación de videojuegos y simuladores se ha vuelto una práctica relevante en muchas esferas, y el realismo de ellos depende de la Inteligencia Artificial que se le imprime a los mismos. Los motores de IA son los módulos dentro del sistema que se encargan que las entidades simulen un comportamiento natural. La UCI incursiona en estos temas y elabora de forma exitosa varios de estos productos.

A continuación se presentan algunos módulos de IA implementados:

Juego Laberinto del Saber: Tiene implementadas tres técnicas: Sistema de percepción, comportamientos de locomoción, máquina de estado finita **(Anexo 2, 3)**. Existen dos técnicas de percepción predefinidas que son Datos Generados Manualmente y Scripting, pero solo es utilizada la primera. Se puede decir que por su estructura permite la incorporación de nuevas técnicas de IA. Del total de clases que posee 14 dependen de alguna manera de clases del juego, lo que representa un 44 por ciento del total, lo que imposibilita su reutilización; otra característica a destacar es que la MFS no está implementada de forma completa y el manejo de script es inexistente. (Gómez, junio, 2009.)

Rápido y Curioso: El módulo de IA que implementa este juego emplea como técnicas una máquina de estado, comportamientos de locomoción, una red neuronal. Los comportamientos sólo permiten tres movimientos básicos: evitar un obstáculo, adelantarse o seguir una trayectoria dentro del cual está implementada la red. El mismo está construido específicamente para este juego por lo que no es posible reutilizar el código dentro de él. El agente inteligente posee sensores, que es el que le permite al agente obtener los datos del mundo, pero que no llega a ser un sistema, ni incluye percepción auditiva.

***Library on Crowd Behaviors for Videogames*:** Está hecha exclusivamente para la utilización de comportamientos de locomoción. Además, incluye la funcionalidad de particionar el espacio *GridShapedPartitionUtility*, el cual lo hace en forma de rejilla, acelerando la búsqueda de los vecinos cuando se implementa el movimiento de grupos. La estructura de la misma **(Anexo 4)** permite la inclusión de extensiones futuras en caso de que sea necesario (Arias, 2010.).

Simulador Aduana: El módulo de aduana es la arquitectura más completa de todas, aunque se debe trabajar más en la organización de las interfaces. El sistema sensorial que tiene implementado no lo es

tanto, sino un grupo de sensores que el agente posee sin llegar a implementar percepción auditiva, ni la obtención de datos medioambientales. Posee una arquitectura **(Anexo 5)** que permite incorporarle más de una manera de discretizar el mundo. Como la mayoría de las estructuras vistas la toma de decisiones la realiza una máquina de estados. Los agentes dentro del entorno se mueven mediante comportamientos de locomoción. No tiene implementada ninguna forma de aprendizaje.

A nivel mundial los productores de juegos y simuladores son en su mayoría privados por lo que hay poco conocimiento de cómo se realizan los módulos o bibliotecas. Sin embargo, existen algunos como *Aglib y Gaul bibliotecas* que trabajan sólo con algoritmo genético*, Free Fuzzy Logic Library* con lógica difusa *o OpenSteer* creada por Craig Reinolds que trabaja con comportamientos (Arias, 2010). Como se puede apreciar todas estas laboran con una técnica específica dentro de la rama de IA.

1.9. Patrones de diseño

Un patrón de diseño describe un problema en especial y su solución que se pueda presentar como una representación de clases, objetos y sus relaciones (Erick Gamma, 1997). Los mismos pueden clasificarse en creacionales, estructurales o de comportamiento; cada uno se puede aplicarse de variadas formas. Su función principal es que permiten la reutilización del modelo que se vaya a diseñar. En este epígrafe se mostraran algunos patrones teniendo en cuenta cuando debemos utilizarlos.

Algunos de los patrones de diseños son:

Creacionales

- Solitario: Como su nombre lo indica garantiza que una clase solo tenga una instancia, lo que implica que se crea un punto de acceso a la misma desde cualquier otra parte del diseño.
- Método de la fábrica: El uso del mismo permite flexibilidad, además de que facilita futuras ampliaciones, su propósito es tener una clase capaz de encargarse de la creación de un objeto en concreto para permitir a las subclases decidir de cuál de ellas va a instanciarlo.
- Fábrica abstracta: Uno de sus usos más relevantes es proporcionar el sistema independencia sobre la manera en que sus productos son tratados o cuando se quiere revelar solo la interfaz sin llegar a la implementación. Por lo que se puede decir que crea objetos que se relacionan de alguna forma sin especificar sus clases concretas.

- Prototipo: El objetivo del mismo es crear un prototipo para los objetos que queremos en el sistema, para que cuando los vayamos a usar solo se copie el mismo. Esto implica que podamos configurar la aplicación en tiempo de ejecución.

Estructurales

- Composición: Trabaja a nivel de objetos, favorece el entendimiento de los clientes, además de que le hace el trabajo más simple. Pues no tienen que tener en cuenta cuando un objeto es simple o compuesto, ya que estos de tratan uniformemente.
- Adaptador: La misión de este patrón es que la interfaz de una clase se adapte a lo que el cliente necesita, permitiendo que clases de interfaces incompatibles trabajen unidas.
- Fachada: Es la cara de un subsistema, pues su estructura brinda una interfaz que satisface a la mayoría de los clientes que van a trabajar con el mismo y es con ella que interactúan. Una de las ventajas de su aplicación es que el cliente no tiene que trabajar con un gran número de objetos, por que la “fachada” es la responsable de tramitar las peticiones.
- Puente: Separa la abstracción del subsistema de su implementación, posibilitando que se cambie la implementación sin que el personal que lo está usando lo note.
- Decorador: Se utiliza cuando queremos modificar un objeto de forma dinámica, es una alternativa a la herencia, pero es muy costosa su aplicación si se implementan componentes complejos.

Comportamiento:

- Observador: Logra una dependencia de “uno a muchos” entre objetos, por lo que un cambio manifestado en uno es informado a los restantes. Un detalle que es válido mencionar es que cuando se reporta el cambio los restantes objetos son libres de hacerle caso o no. Además de que una pequeña modificación puede provocar costosas actualizaciones.
- Estrategia: Su funcionamiento es a nivel de objetos y permite a los algoritmos variar con independencia quienes los usan, por lo que se usa cuando necesitamos variantes diferentes de un mismo algoritmo. Hace el diseño más entendible y la posibilita futuras extensiones.
- Comando: Resuelve el problema de la redundancia en aplicaciones donde existen más de una manera de ejecutar una función, tomando la acción que se va a ejecutar como un objeto.

- Cadena de responsabilidades: Forma una cadena con los objetos, donde si uno no es capaz de responder a una petición dada la transmite al siguiente para que este lo haga. Proporcionando un mayor acoplamiento y flexibilidad.

CAPÍTULO 2
PRINCIPALES COMPONENTES DE LA ESTRUCTURA

Introducción

En este capítulo podremos conocer cuáles son los componentes fundamentales que debe tener un módulo de IA para que una entidad actúe de forma parecida a la humana, teniendo en cuenta los conceptos mencionados en el Capítulo 1. Además se propondrá una estructura básica de cómo debe ser el sistema y como varía la misma teniendo en cuenta los diferentes grados de complejidad que el videojuego o simulador pueda requerir.

2.1 Partes fundamentales que debe tener un sistema inteligente

Después del análisis realizado en el Capítulo 1, se puede afirmar que poseer una estructura fija de Inteligencia Artificial general se ha convertido en una tendencia creciente, de forma tal que permita a las entidades ser diseñadas adecuadamente, en consonancia con las características que deban poseer cada una de ellas. La estructura, además debe dar la posibilidad de construir herramientas que puedan reutilizarse fácilmente, por lo que tiene que adecuarse a todos los géneros de los juegos o los tipos de simuladores.

Un sistema de software para proveer de inteligencia agentes dentro en un videojuego o simulador debe constar con tres componentes fundamentales: un componente encargado de la navegación, un componente encargado de la toma de decisiones y uno encargado de que el agente obtenga la precepción del mundo.

Componente de navegación: Es quien va a calcular los vectores necesarios para que el NPC se mueva por el mundo de acuerdo a la decisión tomada por el sistema de toma de decisiones. Para que el agente controlado por la computadora tenga una buena navegación debe conocer su ubicación así como la posición de las otras entidades.

En este componente estarían incluidas las implementaciones para lograr el movimiento tanto cinemático como dinámico. Traducido esto a las fórmulas físicas utilizadas para el trabajo con vectores; además se colocarían los comportamientos de locomoción o como también se conocen los *steering behaviors*, en dependencia del software que se va a implementar.

Componente de toma de decisiones: Este componente es esencial para que el sistema que se está desarrollando parezca inteligente, por lo que los diseñadores deben prestarle especial atención.

Supongamos que tenemos un sistema donde el agente se mueve de forma extraordinaria y posee una información elevada sobre el medio ambiente, pero si escoge mal la forma en que debe actuar; esto provocaría que fuese ineficiente para el cliente.

Puede diseñarse de dos formas fundamentales: para simuladores o juegos que requieran inteligencia de estrategia y para los que no. Aunque las técnicas utilizadas para el logro de cada uno de estos diseños son básicamente las mismas y tanto uno como otro incluirían las condiciones necesarias para la ejecución del juego según sus reglas. En este componente estarían incluidas además las implementaciones necesarias para el desarrollo del aprendizaje en el caso que un juego o simulador lo requiera.

Componente de percepción: Se encarga de todo lo que el agente es capaz de obtener del entorno virtual. Su estructura puede ir desde una clase que capture solo datos topológicos hasta un conjunto de clases complejas que capturen estos y además los datos medioambientales y las características tácticas del ambiente. Es muy importante para que una buena decisión sea tomada, que los datos incorporados sean los correctos y que estos se hagan con la rapidez necesaria. No obstante, existen software que trabajan con información ruidosa, para ello emplean lógica difusa.

No todos los sistemas inteligentes necesitan para su desarrollo que los componentes implementados tengan alto grado de complejidad. Ejemplos de ellos puede ser un sencillo juego de Ajedrez o el Salamina, los cuales no utilizan un sistema de navegación muy complejo.

Aquellos videojuegos donde las acciones del agente se realizan de forma lineal, generalmente utilizan los comportamientos de locomoción para dirigir sus acciones de acuerdo a una situación previamente determinada. Por lo que el trabajo del componente de decisión se resumiría a identificar la situación en que se encuentra la entidad y enviar cual es el comportamiento ya predefinido.

2.2 Estructura básica

En el epígrafe anterior vimos cuales deben ser los componentes principales que debe tener un software para incluir inteligencia a una entidad o agente en un juego o simulador. ¿Ahora, cómo sería la relación entre ellos y que representaría cada uno dentro de un videojuego o simulador determinado?

La lógica de esta estructura quedaría de la siguiente manera, el sistema de percepción obtendría la información del medio donde la entidad se desarrolla, esta información se le daría al encargado de tomar la decisión y después que esta sea tomada pasaría al encargado de navegación y este a su vez determina cómo la entidad va a seguir el movimiento.

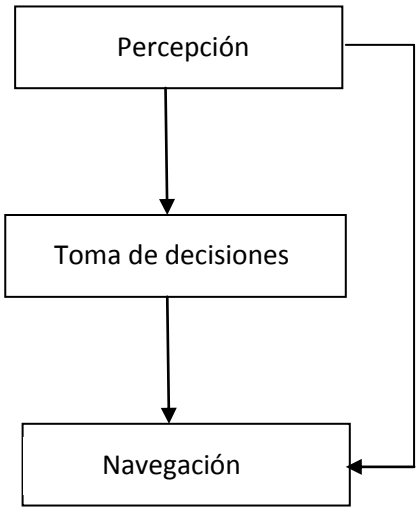


Figura 3: *Relación entre los diferentes componentes de un sistema inteligente*

Si observamos la relación entre los componentes, existe una relación establecida entre el componente de navegación y el de percepción, la misma está dada por que en el momento en que una entidad se traslada de un lugar a otro, debe tener en consideración características del mundo.

Es válido aclarar, también, que cuando se va utilizar la estructura anteriormente presentada debe pensarse de antemano cómo se va a incluir en el sistema que estamos diseñando, por lo que debe hacerse un trabajo de diseño antes de ir directo al código, con el fin de permitir la reutilización del mismo y evitar los errores costosos en la etapa de desarrollo. Para ello la inclusión de patrones de diseño sería una buena opción para la realización de los diagramas de clases.

A continuación se describirá cómo quedaría la estructura, para la implementación del juego Salamina.

Caso de estudio # 1:

Es juego para 2 jugadores (la computadora y un jugador humano), que recrea la histórica batalla naval entre griegos y persas durante las llamadas Guerras Médicas, en el 480 antes de nuestra era.

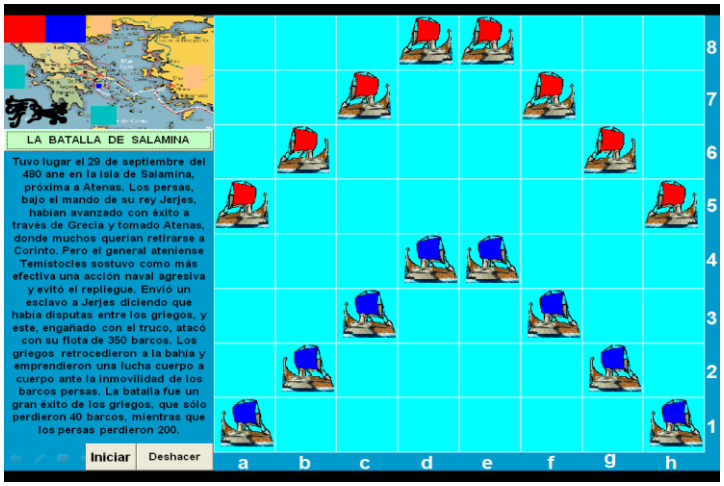


Figura 4: *Ambiente del juego "Salamina"*

En su turno cada jugador podrá mover uno de sus barcos, una o dos casillas en cualquier dirección, avanzando o retrocediendo, horizontal o verticalmente (nunca en diagonal), hasta una casilla vacía. Al moverse dos casillas una ficha podrá pasar sobre otra de su mismo bando, pero no sobre una contraria. Ninguna ficha podrá detenerse entre 2 contrarias.

El movimiento de las fichas estará determinado por la dirección del viento dominante en ese turno, del siguiente modo:

- Si la ficha se mueve en la misma dirección del viento dominante, podrá trasladarse a una o dos casillas, como se desee.
- Si se mueve en una dirección que no sea la del viento dominante, sólo podrá trasladarse a 1 casilla.
- Si en un turno no aparece flecha en ninguna dirección, significará que en ese turno no existe viento dominante y el jugador no podrá mover ninguna de sus fichas.

Un barco será hundido cuando quede entre dos contrarios, formando los tres una misma fila, horizontal, vertical o diagonalmente. Así también se pueden hundir varios al mismo tiempo, por ejemplo: formando un triángulo, donde los tres vértices los ocupen fichas de un mismo bando, y las dos atacadas queden envueltas entre ellas. Pierde el primero que se quede con solamente dos barcos.

La implementación del juego se realizó con una clase que es la que controla la Inteligencia Artificial del contrario dentro del juego, en la que se escoge la jugada que va hacer la computadora. Las imágenes que se colocan en el tablero y el usuario ve, son controladas por una matriz que guarda las coordenadas donde deben ser cargadas por el juego. En este tipo de videojuego tan sencillo no es necesario tener múltiples clases, pues la inteligencia se puede programar con un simple algoritmo, en este caso se

escogió el Mini-Max. La percepción se logra mediante un único método que es el encargado de capturar las variaciones en la matriz del juego cada vez que se realiza una jugada. Quedaría, solamente por decir cómo se logra la navegación de las embarcaciones dentro del tablero después de tomada la decisión. Pues bien, el algoritmo de toma de decisiones se encarga el mismo de devolver una matriz organizada con la jugada escogida al método encargado de mostrar las animaciones.

Como se ha podido observar en esta manera de implementar el mismo, el componente que gestione la navegación solo se encargaría de actualizar la matriz.

2.2.1 Forma de comunicación entre cada uno de los componentes

La estructura propuesta para la investigación introducen dos conceptos: el de sensores y el de cerebro, los cuales están encargados de establecer la comunicación entre los demás componentes. En este momento, cuando mencionan comunicación, se refiere a la parte encargada transmitir información de un componente a otro, dentro de la estructura.

Cerebro: Controla como la entidad va a realizar una acción. Su función es la de comunicar lo que la entidad percibe del mundo con el componente encargado de la toma de decisiones, y a su vez el resultado de la decisión dárselo al componente que se encarga de la navegación, quien en definitiva realiza el movimiento teniendo en cuenta la acción seleccionada.

Sensores: Su función es la de transmitir al agente la información que requiere sobre el mundo que los rodea. Están constantemente, pidiéndole al componente de percepción las modificaciones que se realizan en el mundo. Al igual, que los humanos las entidades pueden tener más de un tipo de sensor según la necesidad de la aplicación.

Introducidas estas nuevas partes al diagrama inicial, quedaría conformado un modelo más completo que el anterior.

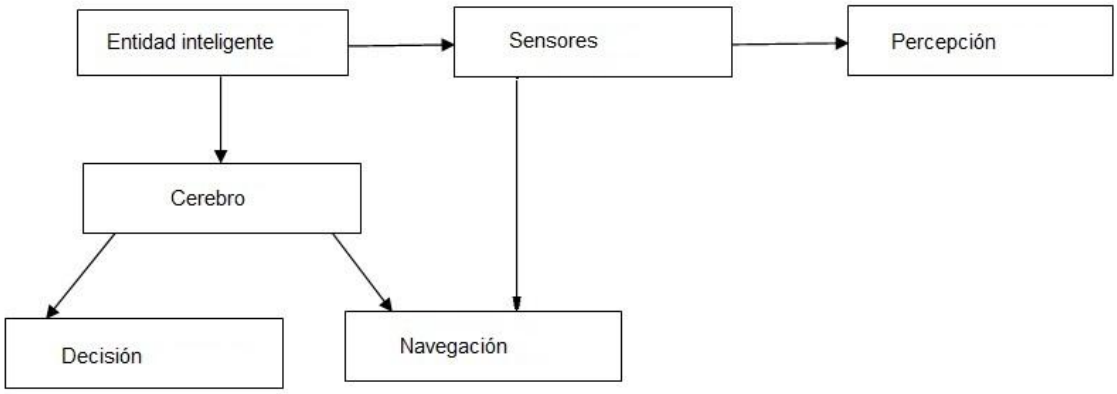


Figura 5: *Estructura básica para la introducción de inteligencia en agentes inteligentes*

Las relaciones entre las diferentes partes del modelo anterior se pueden describir con un ejemplo sencillo.

Caso de estudio # 2:

Un perro constituye el agente o entidad inteligente que queremos modelar. El mismo se encuentra a unos metros de distancia de una persona. Un animal en una situación similar puede actuar de maneras muy diferentes entre ellas: correr, atacar, mover la cola, todas ellas supeditadas a condiciones del ambiente que la rodea y las reglas del juego. Para la elección de la acción se puede programar una máquina de estado finita, que será la encargada de elegir como va actuar de acuerdo a la persona que tenga delante y a la distancia exacta de la misma. Además de que para moverse se programarían los comportamientos necesarios.

El diagrama de clases del modulo de IA, para la situación anterior se diseñaría de esta manera. Es importante señalar que en el mismo se tienen en cuenta exclusivamente las operaciones principales:

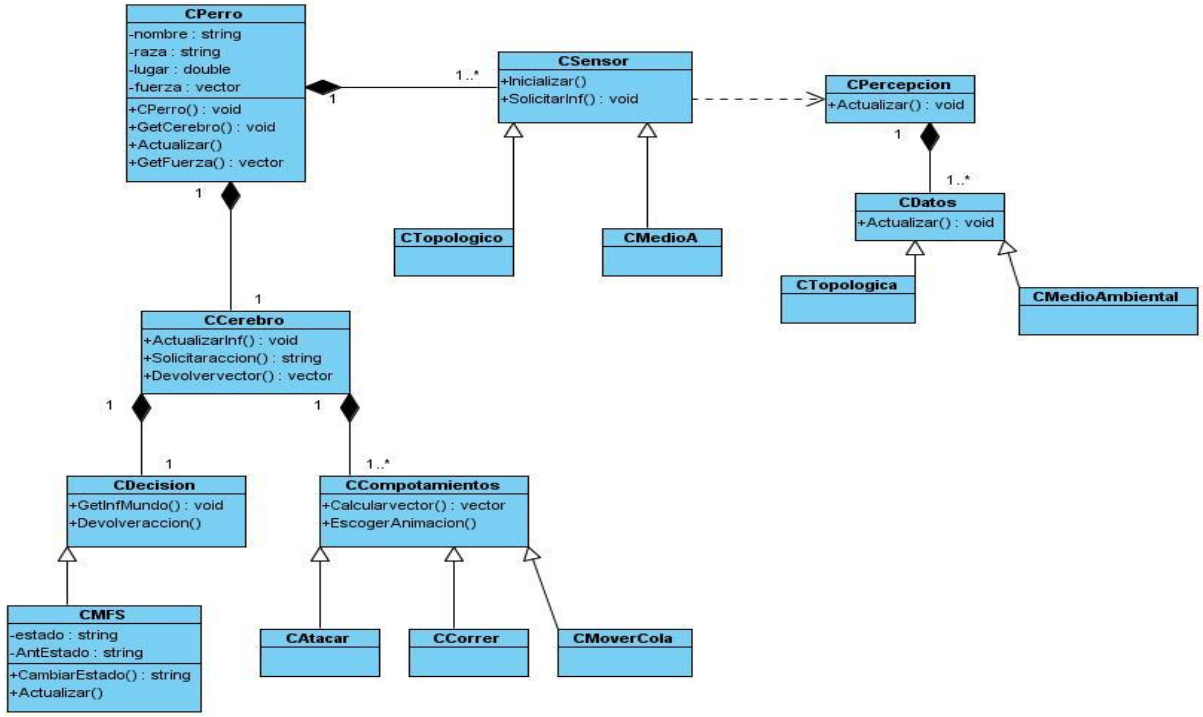


Figura 6: *Diagrama de clases para el ejemplo del perro*

La secuencia de pasos que seguiría el modulo sería la siguiente:

1. La entidad inteligente, en este caso el perro, enviaría a los sensores una señal pidiendo que actualice su estado.(Generalmente esta señal se envía de forma periódica, de acuerdo a un tiempo determinado por el diseñador)

2. Los sensores obtendrían los datos actualizados del componente de percepción.

3. La entidad actualizaría los datos del entorno en el cerebro de la misma.

4. El cerebro, con los datos obtenidos de los sensores, utiliza la máquina de estado finita que tiene el componente de decisión implementada y selecciona la acción que va a ejecutar.

5. Una vez, seleccionada la acción, el cerebro según el comportamiento escogido calcula el vector director necesario para mover el perro mediante el componente de navegación, y escoge la animación que debe ejecutarse.

6. El cerebro actualiza a la entidad inteligente con el vector resultante y la animación, la cuál es mostrada en pantalla.

2.3 Modificaciones para la inclusión de inteligencia táctica, de estrategia y aprendizaje en los videojuegos o simuladores

En epígrafes anteriores se expusieron los componentes de la estructura básica, así como la forma en que podrían comunicarse para un videojuego o simulador dado, pero no siempre nos enfrentamos a situaciones sencillas como el ejemplo el del juego Salamina o el ejemplo del perro.

En ocasiones, las entidades que manejamos no solo controlan lo que ellas hacen, sino también las actuaciones de las restantes. Los realizadores, en ese momento, se enfrentan a un reto diferente: el de incorporarle a los mismos la capacidad de tomar de decisiones estratégicas, de pensar maniobras tácticas y adaptarse a las situaciones inesperadas, algo que no han visto en las estructuras anteriores.

No solo los simuladores bélicos o de lucha, necesitan inteligencia de estrategia o táctica; los videojuegos deportes también la utilizan en la implementación del entrenador o pequeños juegos donde se programe una trampa para el jugador, que involucre más de una entidad. Inclusive, situaciones de guerra donde el general de batalla muere, el segundo al mando debe tener la capacidad de incluirla en su programación, para que pueda asumir el mando en ese momento.

La inclusión de técnicas de táctica o estrategia no significa tener que implementar nuevos algoritmos, sino incluir las tácticas que se implementaron en los mismos. Si se remiten al Capítulo 1, al observar las técnicas que allí se presentan, se pueden percatar que es fácil incluirlas entre las consideraciones de las técnicas restantes. Las variaciones a realizar serían mínimas, quizás modificar una función heurística o incorporar un camino adicional.

La colocación de *waypoints* permiten especificar al diseñador puntos de cobertura, o puntos donde el alcance del fuego es mayor; por lo que en medio de un ataque la decisión no se basaría exclusivamente en si se tiene arma o no, sino tendría en cuenta si es más beneficioso moverse a un punto de cobertura cercano para alejarse del fuego enemigo.

De acuerdo al producto que se esté construyendo la secuencia que debe seguir la incorporación de inteligencia de estrategia varía, pues las decisiones en un juego en ocasiones dependen de la estrategia que se va a usar o viceversa.

Si de la decisión que se va a tomar depende la estrategia que el agente va a utilizar, se debe dar al proceso de decisión acceso a las consideraciones tácticas, que estén implementadas, pues de no ser así, una mala opción puede generar gasto innecesario de memoria al tener que re-evaluar la opción escogida

primeramente, cuando el sistema inteligente se dé cuenta que si hubiese tomado el otro camino, la decisión era mejor. Esto sucedería en el mejor de los casos, en el que el sistema se percate que la acción elegida no fue la mejor, en caso de que no sea así puede correrse el riesgo de que el juego se vea poco inteligente o el simulador poco realista.

Hay sistemas en que existe una jerarquía de mando para elegir la acción que se va a ejecutar, por ejemplo un simulador bélico. En ese caso el agente al mando deberá estudiar las condiciones del ambiente en que se esté desarrollando la acción, seguidamente dentro de las estrategias que tenga a su conocimiento escoger y enviarle un mensaje a las demás entidades involucradas y cada uno entonces, jugaría su rol en la simulación. Para elaborar el diseño del simulador sería bueno hacer un estudio de las estrategias que las personas han llevado a cabo en situaciones similares y programárselas a la entidad.

En algunos videojuegos, el jugador es quien escoge la decisión, en ese caso los NPC deberán actuar acorde a la misma, por lo que la decisión del jugador prima por encima de las restantes. Los comportamientos de los caracteres en esta situación son más sencillos, pueden ser no alejarse mucho del grupo, proteger al jugador, o servir de cebo para que haga una misión determinada. Además de que se pone de manifiesto la otra variante de incorporación de estrategia y táctica en el componente de toma de decisiones, pues los comportamientos de los caracteres están supeditados por la táctica escogida por el jugador.

La toma de decisiones debe manejarse por niveles. Las estrategias y tácticas del grupo, deben colocarse por encima de las decisiones individuales o en caso que el jugador sea el jefe del mismo se sitúan ellas en los niveles más altos. Dentro de un mismo agente las decisiones estratégicas se colocarían en la cima, pues son quienes tienen un efecto en un largo período de tiempo. Por ejemplo en un juego de futbol sería si jugar a la ofensiva o no. El otro nivel estaría formado por las tácticas, que es el encargado de aplicar la estrategia escogida y por último en el nivel más bajo se colocaría las de efecto corto, como disparar el balón a la portería o hacer un pase.

La adaptabilidad de los agentes creíbles, se manejaría como una técnica mas dentro del nivel de decisión que la necesite, para ello se utilizan algoritmos como las redes neuronales que emulan el comportamiento de los seres vivos. Puede realizarse de dos formas: el aprendizaje supervisado, que se realiza cuando se conocen las posibles entradas y salidas. La otra vía es el aprendizaje no supervisado, que se utiliza cuando no se tiene un conocimiento antes de ejecutar la técnica.

Durante el desarrollo del epígrafe se han planteado diferentes maneras en que se le puede agregar inteligencia en un videojuego o simulador determinado, lo que restaría ahora es incluir estas consideraciones en la estructura básica que fue presentada anteriormente, para ellos veamos la Figura 7.

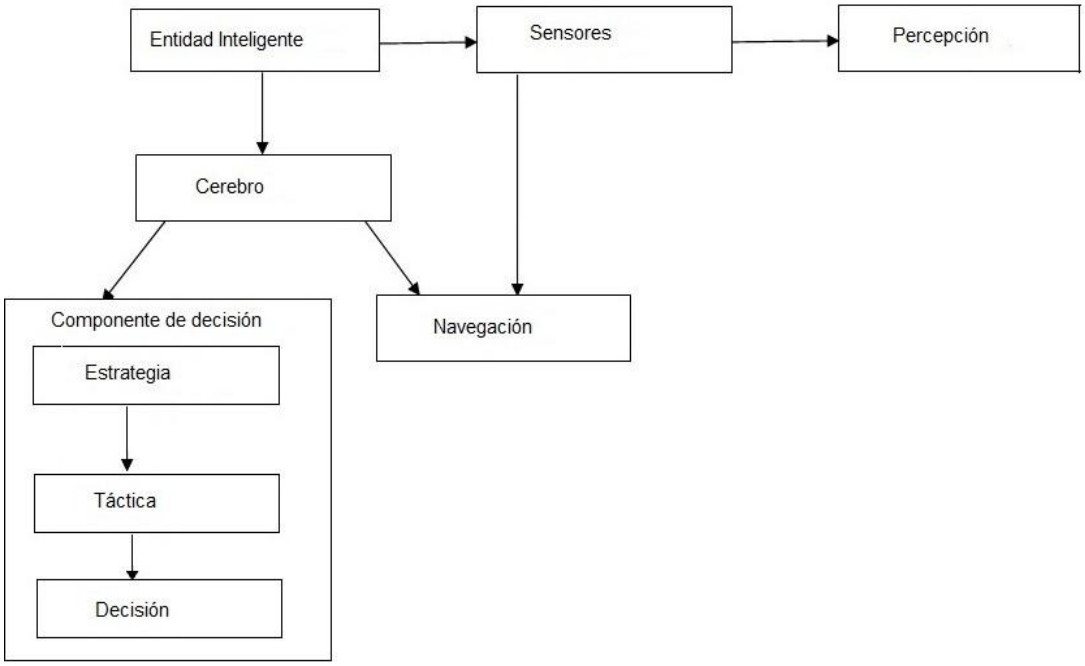


Figura 7: *Incorporación de inteligencia de estrategia en agentes inteligentes*

Caso de estudio # 3:

Un jugador de futbol cuando tiene la pelota en sus pies para ejecutar una acción, tiene que conocer la ubicación de los restantes jugadores, el grado de presión que ejerce el equipo contrario sobre ellos y la estrategia optada por su equipo. Además de saber una serie de técnicas para la realización de los pases y evitar que el jugador contrario le arrebate el balón.

Llevando esto a técnicas concretas de programación, para los movimientos de los jugadores se deberá utilizar el movimiento dinámico, pues la velocidad es un factor determinante en este tipo de juego. El componente de navegación, tendría incluidos los comportamientos necesarios para la recepción de la bola, el tiro a la portería, y el movimiento en el terreno. Teniendo en cuenta, que para el traslado por el campo, es imprescindible conocer la velocidad de la pelota, en momentos como la recepción de un pase largo, donde el jugador debe llegarle al balón.

Un mapa de influencia, podría adaptarse para la localización de zonas donde el jugador este menos marcado. Aunque se había mencionado, que se emplea en las tácticas militares, podría calcularse la influencia de unos jugadores sobre otros, teniendo en cuenta aspectos como: rapidez del jugador y domino del balón.

Las decisiones del partido podrían elegirse utilizando cualquiera de las técnicas estudiadas, en este caso un árbol de decisión sería una opción a considerar, donde las ramas o caminos a seguir podrían, ser en un pase, los jugadores posibles y en los nodos internos se pondrían las condiciones a evaluar para elegir a quien se va a pasar. O en caso de que el jugador tenga la pelota, las ramas podrían enfocarse a si va a realizar un pase o no y de decidir no hacerlo, se evaluaría si está en posición para disparar a la portería o no. Para este caso la estrategia optada por el equipo es la ofensiva. La discretización del videojuego se puede realizar con una malla de navegación. El diagrama de clases para esta situación sería el siguiente:

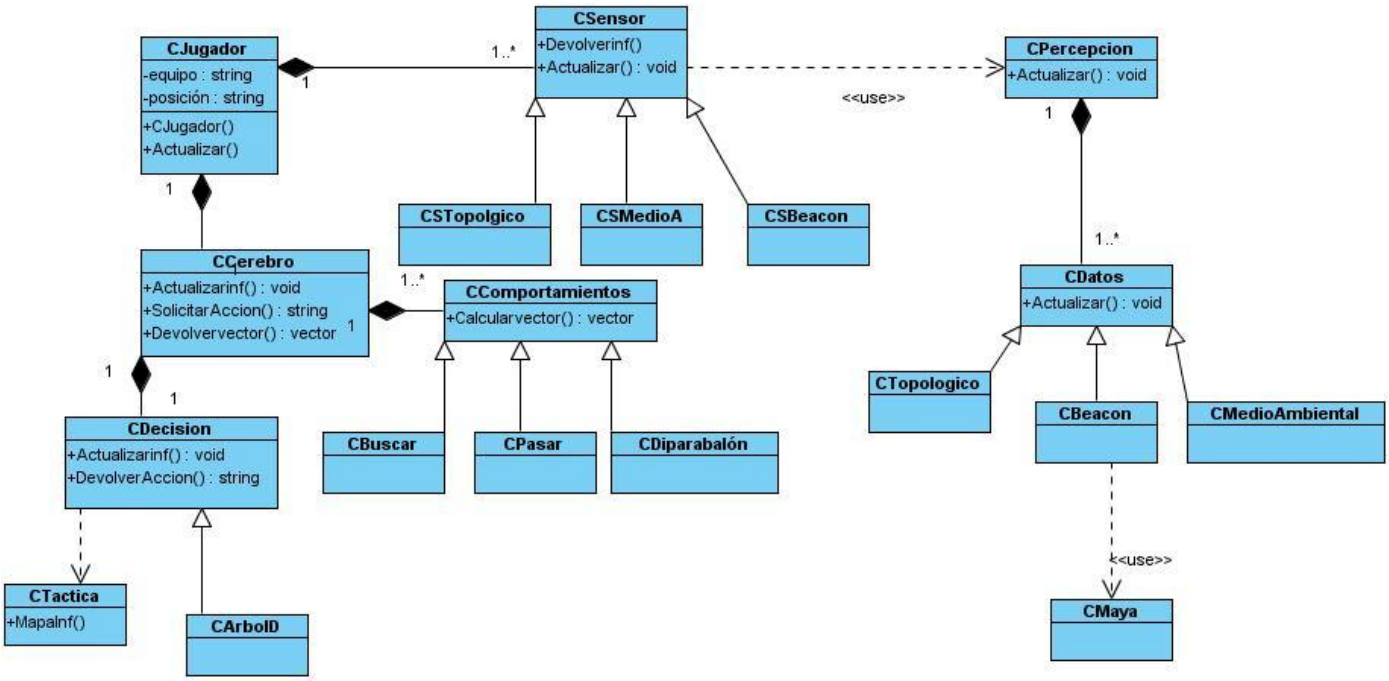


Figura 8: *Diagrama de clases para el Caso de estudio # 3*

Continuamos ubicados en el campo de futbol anterior, en este caso explicaremos cómo se reaccionaría el módulo de inteligencia en la realización de un tiro de esquina, cuando el jugador se encuentra cerca de su portería. El conjunto de pasos para que él tome una decisión sería:

1. Pedir al sistema de percepción, a través de la clase sensores, la posición de los restantes jugadores, en el terreno.

2. El cerebro transmitiría la información obtenida a la clase encargada de tomar la decisión y ordenaría a esta llegar a una conclusión.

3. En dicha clase el árbol de decisión implementado decidiría, si es mejor tirarla al centro de la portería para que uno de su equipo cabecee e intente meter gol, o utilizando el mapa de influencia evaluar cuál es el jugador que tiene menos posibilidades que le quiten el balón.

4. De acuerdo a la decisión tomada, el cerebro le pasaría la opción escogida a la clase encargada de la navegación, quien devolvería cuál es la animación escogida y el vector director con que se debe ejecutar el tiro.

5. El sistema se encargaría de visualizar la decisión tomada.

En el siguiente ejemplo dentro de este mismo caso de estudio se puede observar cómo se combinan la acción de más de un jugador para lograr la realización de una jugada.

Imaginen que el balón se encuentra en la parte del campo correspondiente al equipo manejado por la computadora (blanco) y que lo tiene. La acción a desarrollar en ese caso sería ofensiva, es decir, ir hacia la otra parte del campo e intentar anotar.

Figura 9: *Estado del campo*

La pelota la recuperó el jugador uno del equipo blanco, este recibe la posición de los jugadores y la presión del equipo amarillo, de acuerdo a ello el sistema decide hacerle el pase largo al jugador número ocho, que se encuentra en la banda izquierda cerca del área y no tiene sobre el ninguna presión, porque el equipo contrario ha dejado descubierta esa zona (el mapa de influencia en este caso daría la banda como una zona segura dentro del campo).

El jugador ocho recepciona la pelota, el cerebro le orienta que debe avanzar con la pelota, debido a que no puede disparar a puerta. El estado del campo sería el siguiente:

Figura 10: *Estado del campo en el momento que el jugador ocho tiene la pelota*

Otra vez el sistema discretiza el mundo para obtener el estado de los jugadores, informándole al jugador que hay dos compañeros cerca del área de la portería, la decisión estaría en saber a cuál de los dos disparar.

Mientras esto ocurre el jugador número 10, recibe información mediante sus sensores de la posición del jugador ocho y que él tiene la bola, por lo que el árbol de decisión le orienta retrasarse, para que se desmarcarque del jugador amarillo que lo estaba presionando y así el ocho pueda pasarle.

Después, de estar libre el 10, el ocho realiza el pase, el 10 la recepciona. La jugada, ha sido concluida, el equipo dirigido por la computadora, tiene la posibilidad de anotar.

En este caso han descrito jugadas aleatorias, donde se auxilia de una técnica de toma de decisiones de estrategia, pero se pueden crear una base de conocimientos con las jugadas más destacadas y programar a los jugadores, para si las condiciones son favorables, se ejecuten.

Por supuesto, todas estas acciones simples requieren de que la información del mundo se actualice de forma constante, que el árbol de decisión evalué, todas las opciones posibles y se visualicen las mismas.

2.4 Otras consideraciones válidas a la hora de diseñar un juego o simulador

En la investigación realizada, sobre los aspectos que se debían tener en cuenta, para la confección de un módulo de Inteligencia Artificial, cuando se estudiaron los diseños hechas con anterioridad, se manejaron dos ideas que se consideran interesantes poner a su consideración.

La primera idea que se manejó fue la de incluir en las entidades inteligentes, un componente que se encargue de proporcionarle algún tipo de memoria a corto plazo. Este planteamiento es válido, en la realización de software, donde el agente se enfrente a situaciones similares en cortos períodos de tiempo.

Las fuentes bibliográficas estudiadas, la sitúan como un componente más de la entidad, que es empleada para escoger las decisiones. Pero el cerebro del agente es a quien se propone tenga a su cargo el control de la misma, puesto que él puede darle la información que debe almacenar. También ella no es una habilidad, que llevan todos los agentes inteligentes dentro de un mismo juego, por lo que al relacionarla directamente con la clase que controla las entidades inteligentes, imposibilitaría que todos los agentes hereden de ella.

Un robot, que se construya para sustituir un dependiente en una tienda, puede utilizar este componente para recordar, que le dio una pieza de ropa a un cliente para que se la probara, aun y cuando continuara atendiendo al resto, permitiendo que cuando el vuelva atenderlo de primero.

La segunda idea, se refería a incluir la búsqueda de caminos como un componente más de la estructura, por su puesto la cuál es desechada para lograr que la estructura fuese genérica, debido a que no en todos los juegos o simuladores está técnica es utilizada. No obstante, en este epígrafe expondremos algunas recomendaciones sobre la forma en que debe manejarse en caso de que la aplicación que se vaya a desarrollar la necesite.

Cuando se emplea búsqueda de caminos en la programación de un software, se piensa en un algoritmo capaz de devolver el camino más corto de un punto a otro. Aunque no siempre es necesario el camino más corto, sino el más ventajoso. Además, se debe poner atención en que una entidad que actue a la perfección no resulta atractiva en la mayoría de las situaciones.

Existen dos tipos de vías para encontrar un camino: las que incluye las consideraciones tácticas y las que no la incluyen. Ambas implementan básicamente los mismos algoritmos; la diferencia en cada una está en el momento en que el diseñador las emplea y en las consideraciones que se tienen en cuenta a la hora de construir la función heurística, porque la búsqueda que incluye táctica emplea una función más compleja.

Muchas han sido las variantes por las que han optado los diseñadores, para incluirla en sus diagramas: algunos han optado por darle esta responsabilidad al modelo cognitivo que emplean (Aduana tiene implementado el algoritmo A* dentro de la malla de navegación), otros le dan esta responsabilidad a la clase encargada de la navegación. Sin embargo, si estudiamos el funcionamiento del cerebro no es el componente de navegación, quien elige el camino, el solamente se encarga de ejecutarlo y muchísimo menos es el modelo cognitivo empleado, cuya única función conceptualmente es la de discretizar el mundo. El cerebro, es el encargado de escoger el camino, aunque tampoco es una técnica de toma de decisiones.

Para comprender mejor lo planteado con anterioridad, para ellos se describirá un ejemplo de la vida cotidiana, cuando una persona quiere moverse hacia un lugar y tiene dos calles por donde llegar, una de la forma más rápida que otra. El cerebro, evalúa primero que todo el grado de prisa que tenemos para llegar a la meta, después si es de noche cuál está más iluminado y basado en esas consideraciones, escoge un camino. Seguidamente, emite una señal que ordena a los músculos, seguir esa dirección.

Ahora, una entidad inteligente como realizaría esto: le enviaría una señal al cerebro diciéndole que debe escoger un camino, el mismo evalúa las condiciones del ambiente y decide qué tipo de ruta va a emplear, es decir, si se va por el camino más corto, si se va realizar de manera táctica o si puede ser cualquier otro. De acuerdo al tipo de camino escogido, se ejecuta el algoritmo. El conjunto de puntos resultante, se le pasaría al encargado de la navegación, para calcular el movimiento del agente.

Como se pudo constatar, es un poco contradictorio ubicar la búsqueda de caminos como parte del componente de navegación o el de toma de decisiones, aunque sí ha quedado demostrado que la misma

es ejecutada por el cerebro. Por todo lo antes planteado, se propone que a la hora de diseñar un sistema que la incluya, se modele como una clase amiga relacionada con el cerebro de la entidad.

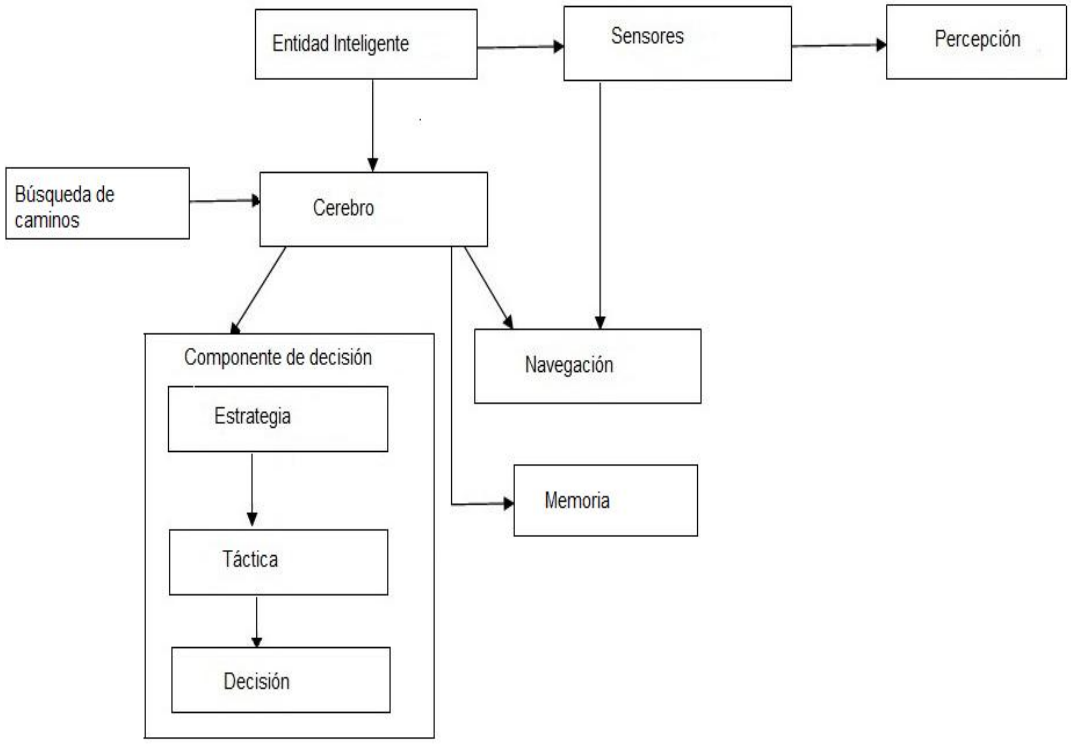


Figura 11: *Inclusión de los conceptos memoria y búsqueda de caminos*

Caso de estudio # 4

El agente que estamos modelando, tiene que fugarse de un banco, que acaba de asaltar. La policía empieza a acorrarlo, alcanzando a colocar a tres oficiales entre la salida y el asaltante. La primera reacción del asaltante (en este caso nuestro agente inteligente) debería ser encontrar la ruta más segura y rápida para llegar a la salida, e impedir que sigan entrando oficiales. La segunda acción que debe pensar acto seguido de haber llegado cerca de la puerta es derribar a todo el que le impida le paso hacia la salida.

Para llevar a cabo dicha situación, el módulo inteligente debe buscar un camino que cumpla esas condiciones, por lo que se incluirán algoritmos tácticos. Escogimos la colocación de puntos de interés, en la zona del banco, que especificaran lugares donde se puede proteger de los disparos, además otros

donde su poder de disparo será mayor. El algoritmo A* podré ser quien escoja el conjunto de puntos finales, lo que se tomaran en consideración los puntos colocados a la hora de elegir el mejor camino.

Además, es necesaria para la elección de las decisiones una buena información de lo que está ocurriendo en el exterior, para lo que recomendamos el empleo del sistema de percepción basado en la arquitectura propuesta por Frank Puig.

El agente debe tener dos tipos de habilidades: audición y la visión, para lo cual se equipara con sensores que controlen la información topológica, medio ambiental y los llamados *beacon*. Los *beacon* recibirían la información de los objetos dinámicos del área.

Existirá una clase cerebro que será, como en los ejemplos anteriores de comunicar la información al componente de decisión. El mismo podrá implementarse de forma factible con una máquina de estado finita, que podría tener como uno de sus estados combatir o salir, en el cual se encontraría ahora. La navegación debe realizarse con comportamientos de locomoción dinámicos entre los que se encontrarían: buscar, arribar, evasión de obstáculos.

Con estas técnicas es suficiente, para ejecutar la búsqueda de caminos, pero que pasa cuando el asaltante llega al último punto antes de la salida y tiene que enfrentarse a los policías.

Lo primero que se tendría que saber es si hay algún punto de fuego o de cobertura cerca de su posición, en el caso de que no se encuentre ya en uno. Después, de tomada esa consideración, el tiroteo comenzaría, hasta que los policías o él terminen muertos. Asumiendo, que la entidad logre derribar a uno de ellos, para evitar que se mueva de forma imprudente olvidando a los restantes, es posible la elaboración de una clase memoria, que almacenaría la información reciente capturada por los sensores. La memoria se activaría inmediatamente, después de que uno de los policías ha muerto, para recordarle al asaltante que todavía quedan dos que debe vencer.

El diagrama de clases resultante para este caso de estudio se diseñaría de la siguiente manera:

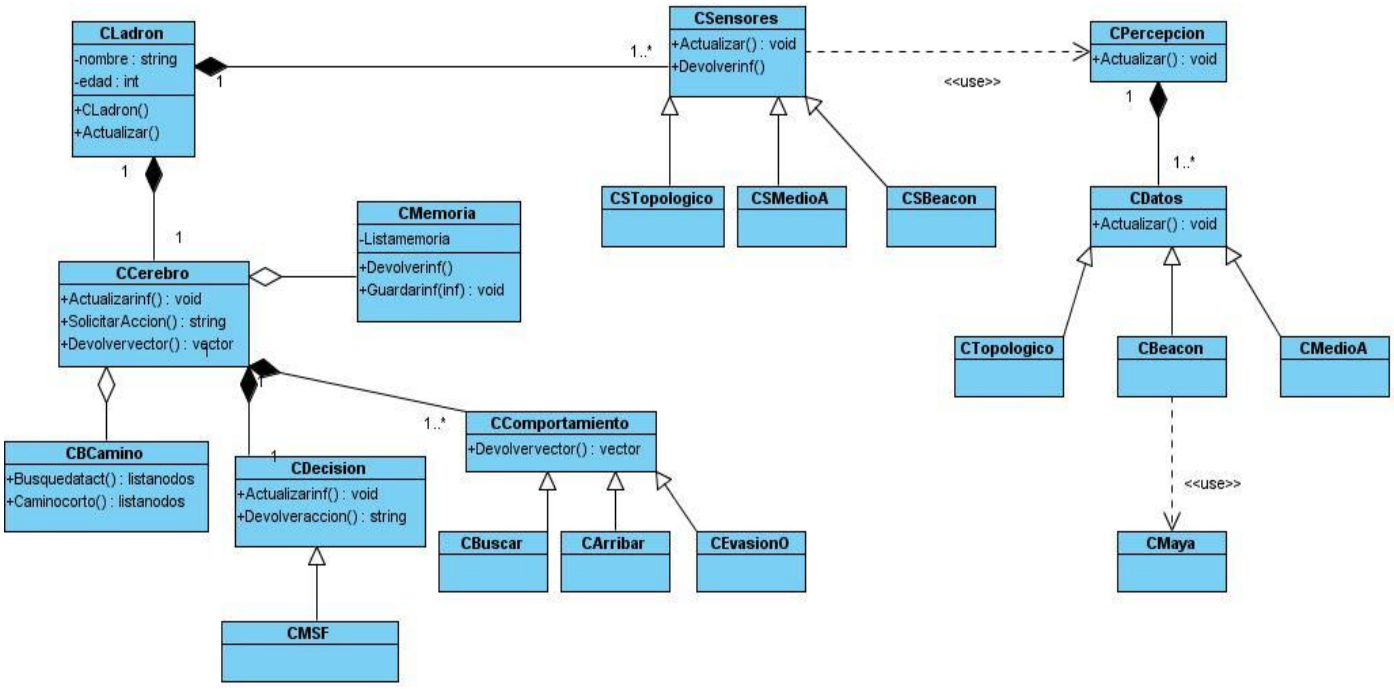


Figura 12: *Diagrama de clases para el Caso de estudio #4*

Secuencia de pasos del sistema para la realización de la búsqueda de caminos del caso de estudio # 4:

1. El sistema se percata que el agente debe buscar una forma salir del banco.
2. Los sensores envían la información del ambiente al cerebro.
3. El cerebro, le pide a la clase decisión, el camino que deben tomar y le da la información entregada por los sensores.
4. La clase decisión utilizando la máquina de estado finita, que actualmente está en el estado salir, envía al cerebro para que determine el camino más corto y menos peligroso.
5. La clase cerebro orienta a la clase de búsqueda de caminos, para que encuentre el mismo.
6. La clase de búsqueda de caminos, utilizando la información de la clase estrategia y la obtenida por el agente mediante los sensores, devuelve el conjunto de puntos al cerebro.
7. Este se los da a la clase encargada de la navegación, quien sigue el camino.

Secuencia de pasos para la ejecución del enfrentamiento con la policía:

1. Cuando el agente se encuentre en el final del camino, el cerebro enviaría a la clase encargada de la decisión la posición de los guardias y el estado de la máquina cambiaría a combate.

2. El encargado de la decisión utilizaría la información que brinda la clase de táctica, para averiguar si existe algún punto de fuego o de cobertura cercano y de ser así, mandaría al cerebro el comportamiento que ejecutaría la entidad.

3. La clase de comportamientos haría los cálculos pertinentes y los enviaría al cerebro.

4. El cerebro nuevamente, pediría una acción a la clase decisión y esta devolvería que se debería disparar hacia uno de los enemigos.

5. La clase comportamientos, después de recibir las orientaciones necesarias por parte del cerebro, realizaría los cálculos.

6. Inmediatamente, después de muerto ese guardia, el cerebro pediría la siguiente acción.

7. La clase decisión, antes de cambiar de estado pregunta a la memoria, si queda alguien más.

8. Esta clase le devuelve la posición de los guardias y la cantidad.

9. Al darse cuenta que le faltan dos, la clase decisión indicará disparar al otro guardia.

10. Así sucesivamente, se repetirán las pasos del seis al nueve hasta que todos los oficiales o el agente mueran.

Es importante decir que, en la situación que se está describiendo, no se tuvo en cuenta, que las personas dentro del banco pudieran intentar algo más o que la policía lograra entrar más personal o inclusive que los oficiales de la puerta quisieran intentar algo. El objetivo no es diseñar una entidad perfectamente inteligente, sino darles la idea de cómo podría funcionar su módulo y darle una vía para el diseño de sus clases, que después de acuerdo a las exigencias de su software usted adaptaría. Además, en el conjunto de pasos se obviaron los necesarios para que el resultado del comportamiento se mandara al sistema de animaciones, porque se centra solamente en mostrar cómo se relacionan los componentes encargados de la inteligencia, no en la relación del módulo con el juego o simulador.

Normalmente, si la simulación o el juego que vamos a hacer, queremos hacerlo muy complejo se debería, en cada punto, ver si la posición de las personas circundantes ha cambiado. De ser así, quizás habría que

modificar la ruta o la decisión antes tomada. Inclusive, se podría añadir un mapa de influencia, para averiguar la desventaja que tu agente posee respecto a la fuerzas policiales. En el peor de los casos, en que hayan logrado entrar, entonces, la rendición sería una posibilidad.

CAPÍTULO 3
RESULTADO DE LA INVESTIGACIÓN

Introducción

En capítulos anteriores se explicó cómo quedaría una estructura que permita el logro de IA en las entidades inteligentes y también se caracterizaron las distintas clasificaciones de los videojuegos. En este capítulo se expondrá el resultado de la investigación realizada. Se darán a conocer las principales diferencias entre las arquitecturas estudiadas y la estructura que se propone en la solución. Además de que se observará la forma de comportarse de algunas entidades en videojuegos y simuladores desarrollados actualmente, para comprobar que la estructura propuesta puede ser aplicada a los mismos.

3.1 Resultado de la investigación realizada

Después de la investigación realizada y la identificación de los principales conceptos que debe incluir un agente para que se comporte de forma inteligente, se diseñó un diagrama de clases, que reúne las características necesarias para lograrlo. La inteligencia se construye en dos diagramas uno que representa la controlada por el juego o simulador en general, que es la que obtiene la percepción y encargada de controlar los agentes inteligentes. El otro recoge el funcionamiento de la entidad inteligente, en su interior.

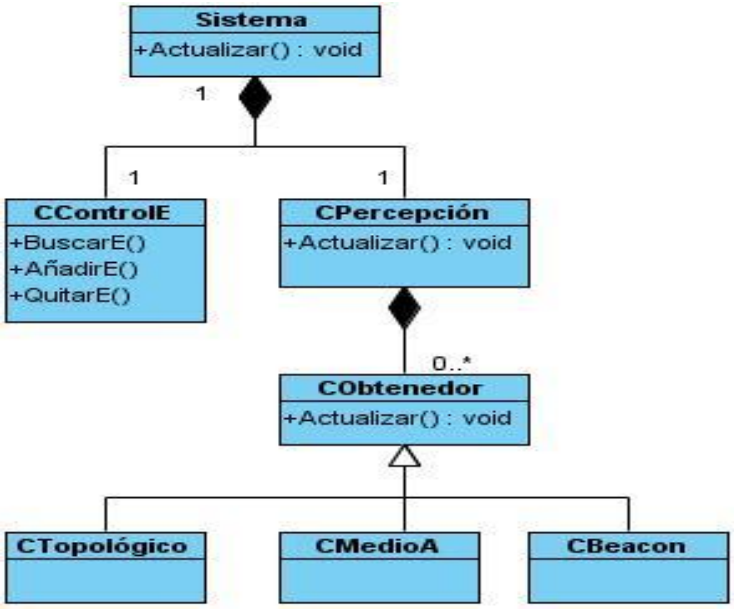


Figura 13: *Modelo de clases del sistema*

Sistema: Clase que representa el juego o simulador, donde se va a incluir la inteligencia artificial.

CPercepción: Es una clase abstracta, con la que se relaciona el sistema. Da la cara a un conjunto de clases, que se dedican a captar la información del mundo que los rodea.

CObtenedor: De ella heredan todo los tipos de obtenedores de datos. Ello proporciona extensibilidad, pues permite incluirle otros obtenedores de ser necesario.

CTopológico: Se encarga de recoger los datos topológicos del entorno virtual, es decir la información de puntos de cobertura, fuego, amenazas, esquinas, paredes

CMedioA: Obtiene la información referente a la estructura del mundo que lo rodea, generalmente representado por círculos o líneas donde están los objetos.

CBeacon: Su objetivo es recopilar los datos móviles dentro del escenario.

CControlE: Es quien controla las entidades inteligentes. Se construye para permitir fundamentalmente el trabajo con las entidades individuales o los grupos de entidades, según el software requiera.

Para el buen funcionamiento de estas clases, es necesario que se tenga una vía para discretizar el mundo, por lo que son utilizados modelos cognitivos. El diseño de los mismos se plantea como una clase abstracta **CModCong** de la que heredaría cada uno de los modelos, que el juego emplee. Esta clase tendría de función de ser la fachada hacia las restantes partes de la estructura, manteniendo una relación estrecha con el componente de percepción.

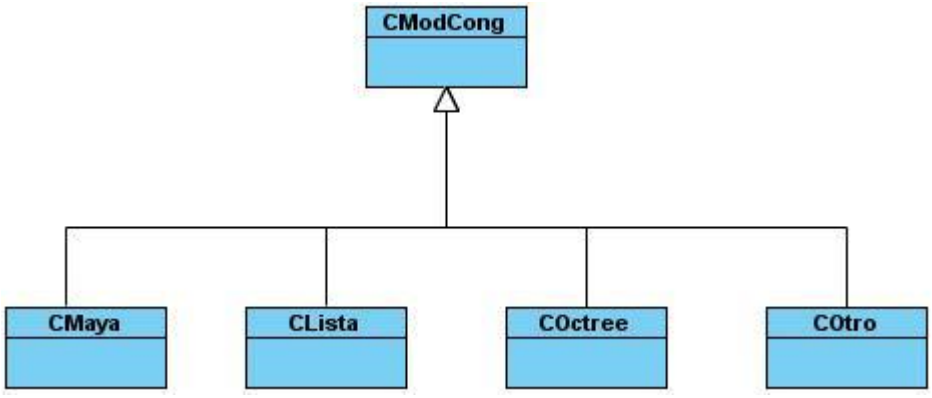


Figura 14: *Representación de los Modelos Cognitivos*

A continuación se expondrá brevemente el conjunto de clases que conforman al agente inteligente:

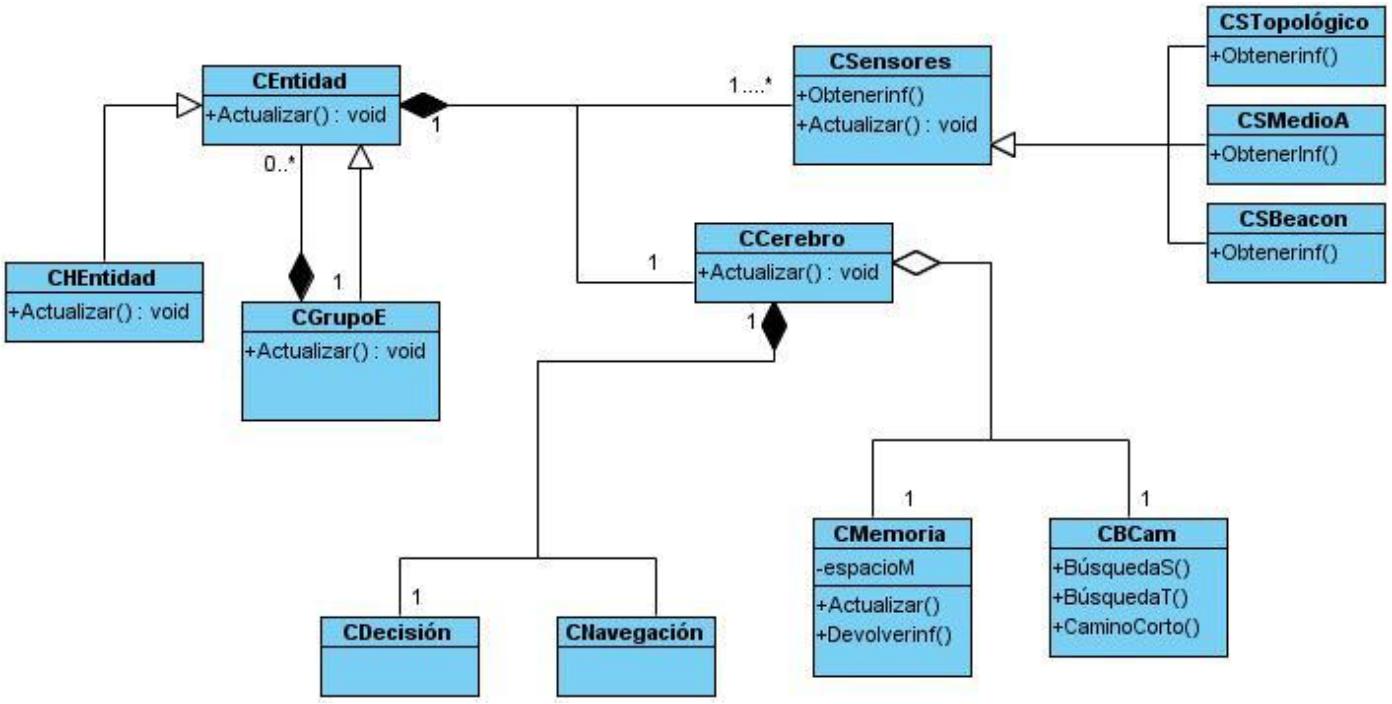


***Figura 15: Diagrama** de clases de una entidad inteligente*

CEntidad: Clase padre de la cual heredaran tanto los grupos de entidades, como las entidades individuales.

CHEntidad: Representa a una entidad en su forma simple.

CGrupoE: Representa un grupo de entidades inteligentes.

CSensores: Es una abstracción de la cual heredaran cada uno de los tipos de sensores, que el agente posea. Serán quienes realicen los cálculos necesarios para informar al agente solamente con lo que él necesita

CSTopológico: Sensor encargado de encuestar al componente de percepción sobre la información topológica del sistema.

CSMedioA: Encuesta al componente de percepción sobre la información medio ambiental del sistema.

CSBeacon: Sensor que pregunta al componente de percepción sobre los objetos móviles de la escena.

CCerebro: Clase interfaz que comunica las restantes clases de sistema con la información obtenida de los sensores y envía una respuesta a la entidad inteligente.

CMemoria: Almacena los datos importantes que obtuvo el agente del entorno, en un periodo de tiempo determinado.

CBCam: Encargada de realizar los distintos tipos de búsqueda de caminos, que pueda necesitar el agente virtual.

CDecision: Clase interfaz encargada del módulo toma de decisiones, dependiendo de la complejidad del juego o simulador, puede estar relacionada con una a más clases.

CNavegacion: Clase interfaz que representa el componente de navegación. El diseño del componente variara según la concepción del videojuego o simulador,

COtro: Esta clase simboliza cualquier otro componente que se necesite para lograr la inteligencia del agente inteligente

Para dar una idea de cómo se relacionan estas clases seguidamente se mostrará un diagrama con la secuencia de acciones que se desarrollan cuando un agente virtual necesita tomar una decisión. En el diagrama estarán presentes solamente las clases principales.

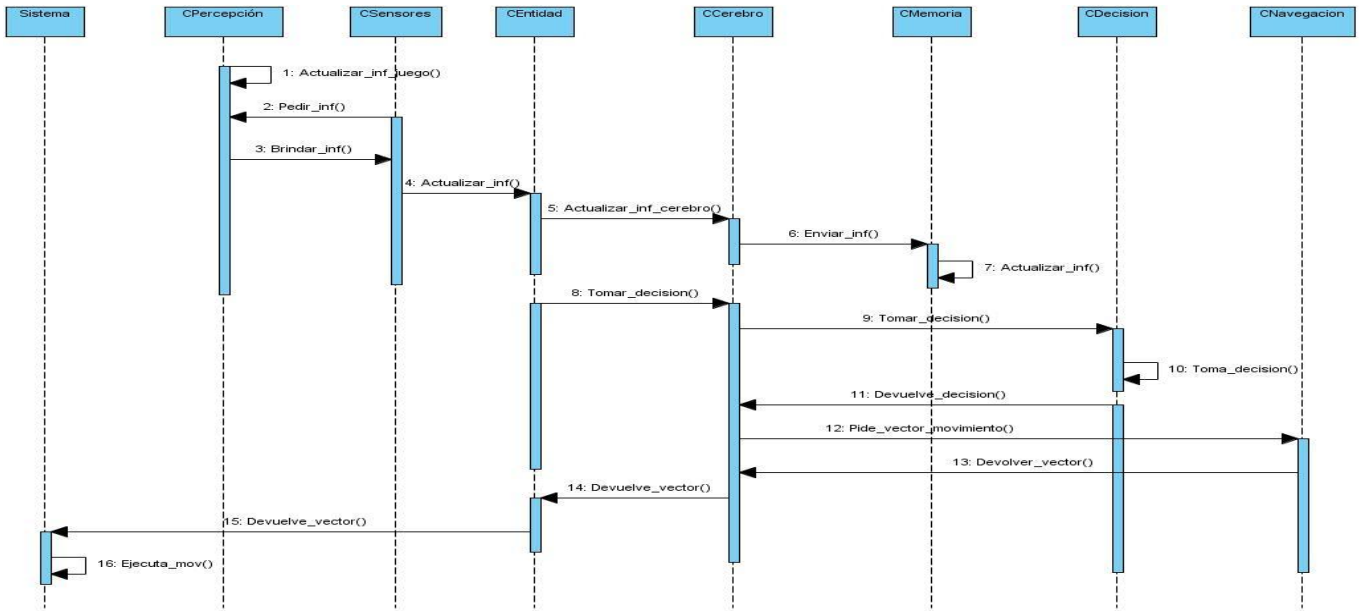


Figura 16: *Secuencia de acciones de una entidad inteligente*

En el caso de la búsqueda de caminos que no se muestra en el diagrama de secuencia, estas serán activadas por la clase **CCerebro,** cuando el agente lo requiera de acuerdo a la acción que deba realizar. Para mayor profundidad, en el Capítulo 2, Epígrafe 2.4 se muestran ejemplos de donde se aplica la búsqueda en la construcción de una entidad inteligente.

3.1.1 Patrones de diseño empleados en la elaboración de resultado

El resultado alcanzado anteriormente utiliza tres patrones de diseño en su elaboración, para lograr que el mismo cumpla con los objetivos definidos. Los patrones empleados son el solitario o *singleton*, adaptador o *adapter* y composición o *composite*.

El solitario se emplea para las clases **CPercepción** y **CControlE**, porque en ambos casos se brinda un punto de acceso global a las mismas, impidiendo que se creen otras instancias de ese tipo. La facilidad del patrón de brindar una vía única para acceder a estas clases, posibilita que pueda ser utilizado desde cualquier lugar dentro de la lógica del videojuego o simulador que se necesite. Además de que, mantiene el concepto de que aunque en el sistema se encuentren muchos agentes inteligentes, solamente debe encontrase una entidad que los controle.

El patrón adaptador se empleó para el diseño de los modelos cognitivos, porque permite convertir la interfaz de una clase para que se adapte a lo que el cliente que la usa necesita. Los modelos cognitivos no tiene todos las mismas entradas en los datos por lo que requieren interfaces diferentes, este problema se solucionó con la inclusión de este patrón. La clase **CModCong** sería la clase a adaptar, pues no resulta práctico adaptar cada uno de los modelos por separado. La utilización del adaptador brinda flexibilidad al diseño, para que un adaptador trabaje con muchas clases a adaptar.

El mayor problema que se materializó en la construcción del resultado, fue el tratamiento de las decisiones de forma jerárquica, debido a que en ocasiones hay que poner las decisiones de un grupo por encima de las individuales. Además, de que necesitábamos una manera de lograr la comunicación entre los agentes creíbles. Para resolver esta dificultad se empleó el patrón composición. El mismo posibilita el tratamiento de objetos simples y compuestos en una estructura jerárquica recursiva. También, es más cómodo para el programador, porque pueden tratar uniformemente objetos simples y compuestos. Los grupos de entidades se pueden manejar como entidades individuales, por lo que permite la toma de decisiones de grupo y el movimiento grupal. La reducción de cálculos, es otra de las ventajas pues si un agente en un grupo buscó un camino por donde se pueden ir, ya no es necesario que los restantes repitan los mismos

cálculos. Con la introducción de este patrón la estructura propuesta puede funcionar como un sistema multi-agente, lo que la coloca por encima de las ya vistas hasta el momento.

3.2 Herramienta y lenguaje de modelado utilizados en el desarrollo de la propuesta de solución

Visual Paradigm

Es una herramienta de modelado de software, con la cual se puede exportar código a partir de los diagramas y generar documentación. Se escoge principalmente por ser multiplataforma, lo que está muy ligado a la política seguida por la UCI. También por brindar la posibilidad de modelar el ciclo completo de vida del software, entre ellos los diagramas de clases, que son los que se van a emplear en el transcurso del capítulo.

Una de las ventajas que posee es "s*oportar todas las versiones del UML hasta el 2.1 y la realización de ingeniería inversa"* (Luis Giraldo, septiembre, 2005). Se ajusta a como se lleva el trabajo en los proyectos donde varias personas están trabajando juntas en un mismo software, permitiendo que se realicen cambios en un diagrama desde distintas fuentes y visualiza estos en tiempo real.

Lenguaje Unificado de Modelado (UML)

Es empleado el UML como lenguaje de representación visual, permite un entendimiento entre las personas y las computadoras."*Es un estándar industrial ampliamente utilizado, tanto que algunos conocedores expresan que en vez de competir por desarrollar una herramienta que soporte muchos lenguajes, se han centrado en crear una que soporte mejor UML"* (Ivan Jacobson, 2000). Se emplea en el diseño de los ejemplos por estar creado para el modelado de programación orientada a objetos, que es con la que se implementa en la Facultad 5. Encuentra una manera expresiva y simple de mostrar al cliente la manera en qué se va diseñando el software.

3.3 Diferencias fundamentales entre la estructura propuesta y las estudiadas en el Capítulo 1

Este epígrafe da a conocer las dificultades encontradas en los diseños estudiados en el Epígrafe 1.8 del Capítulo 1. Además de enfatizar en aquellos conceptos que no se tienen en cuenta en las mismas. El estudio está basado fundamentalmente en los videojuegos realizados en la Facultad 5 producto a que la documentación sobre este tema es escasa. Los productos analizados son: "Rápido y curioso", "Aduana", "Library on Crowd Behaviors for Videogames", "Laberinto del Saber", este último fue diseñado con la arquitectura que propuso Frank Puig para una biblioteca de Inteligencia Artificial. Es importante aclarar

que la estructura propuesta tiene como objetivo dar una idea de cómo deben estar los diferentes componentes dentro de un módulo inteligente de acuerdo a diferentes NPC.

Los conceptos a tener en cuenta para realizar la comparación entre la estructura propuesta y las arquitecturas desarrolladas son los siguientes: percepción, sistema de decisiones, ubicación de la búsqueda de caminos, memoria, aprendizaje, inteligencia de estrategia, sensores, cerebro.

3.1.1 Percepción

Las arquitecturas analizadas, no presentan un sistema de percepción completo, incluso algunas ni siquiera llegan a tenerlo. Si analizamos el juego “Rápido y curioso”, este no posee un sistema de percepción propiamente dicho, sino un conjunto de sensores que encuestan al mundo para obtener información. Además de no incluir información auditiva.

Library on Crowd Behaviors for Videogames posee un componente cuya ventaja principal es la de ser independiente a la lógica del agente, por lo que permite la inclusión de nuevos mecanismos de percepción sin afectar la lógica de los mismos. Su implementación reduce los cálculos innecesarios al sistema, pues no se actualiza de no cambiar informaciones relevantes.

El simulador Aduana no presenta un componente de percepción sino un grupo de sensores, que no incluyen la percepción auditiva, ni la obtención de datos medio ambientales.

Otra de las arquitecturas estudiadas es la propuesta por Frank Puig (Placeres) para un sistema de percepción, esta es la más cercana a lo que un sistema de percepción debe tener. Permite el tratamiento de información topológica, medio ambiental y de información dinámica, pero al igual que el resto de las estudiadas no incluye un sistema de comunicación entre agentes inteligentes.

En el videojuego “Laberinto del Saber “no se especifica en la bibliografía consultada que maneje los datos topológicos y medio ambientales, solo hay evidencia de la existencia de una clase cBeaconGathered que posee una lista donde se registran eventos dinámicos y estáticos que ocurren en el mundo.

Por lo que para darle solución a este problema el diseño propuesto incluye una clase que permite la formación de grupos de agentes, en la cual se podrá implementar un sistema de comunicación entre los miembros de ese grupo. La implementación de la misma, no debe permitir solamente la comunicación desde un único agente al resto, sino de todos con todos. El componente de percepción también al igual que el de la biblioteca *Library on Crowd Behaviors for Videogames* es independiente al agente inteligente, por lo que un cambio en el mismo no constituye un problema. Además, de posibilitar la obtención de los datos topológicos, medio ambientales y los objetos dinámicos del entorno.

3.1.2 Toma de decisiones

Las arquitecturas analizadas en su mayoría tienen la misma estructura, emplean una máquina de estado finita, que hereda del cerebro del agente inteligente. Sin embargo, la biblioteca *Library on Crowd Behaviors for Videogames*, no posee un componente de toma de decisiones propiamente dicho, la información obtenida del mundo es enviada mediante una señal que es la que dice que comportamiento va a realizar el agente, pero una colección de datos solo puede activar un comportamiento en concreto. Como se puede constatar, la mayoría de estos videojuegos y simuladores, utilizan una sola técnica, además de que ven la toma de decisiones como sub-cerebros, cuando conceptualmente ella es una parte de este. Incluso, la arquitectura propuesta por Frank Puig para una biblioteca que está siendo utilizada en la Facultad 5 en estos momentos no centraliza la toma de decisiones, sino la técnica hereda directamente de la clase Brain. Sin embargo, a diferencia de las arquitecturas obtenidas, se incluyó una clase interfaz que maneja todas las técnicas de toma de decisiones que el simulador o juego necesiten y que tendrá una relación de composición con la clase encargada de manipular las funcionalidades del cerebro de la entidad. Esta interfaz, permitirá que de ella hereden las diferentes interfaces pertenecientes a las técnicas de decisión escogidas; para proporcionar que se adapten al sistema de decisión tanto maquinas de estados como redes neuronales y puedan trabajar juntas. El enfoque que se le dio a la propuesta realizada fue inspirado en el cerebro humano, por lo que la toma de decisiones es una parte más dentro de los componentes del cerebro y no un sub-cerebro.

3.1.3 Ubicación de la búsqueda de camino

A excepción de Aduana cuya búsqueda de caminos es responsabilidad de la malla de navegación que implementan, las demás arquitecturas no tienen evidencia de la presencia de la misma. La disconformidad fundamental entre las arquitecturas y la búsqueda de caminos ha estado en el lugar donde se ubica. El videojuego “Laberinto del Saber” no tiene una clase que maneja la búsqueda de camino para los agentes inteligentes, por lo que no se sabe a ciencia cierta cómo Frank Puig trabaja con la ella dentro de la arquitectura. A diferencia de ellas, la propuesta establecida por el presente trabajo de diploma la coloca como una funcionalidad del cerebro de la entidad, que es donde conceptualmente se considera debería colocarse. Producto a que no es una manera de seleccionar una acción aunque ayuda a ello, ni plantea como se va a mover la entidad inteligente sino por donde hacerlo.

3.1.4 Memoria

La memoria no es menester que la tengan todos los agentes inteligentes, pero ayuda a que un agente en simuladores o videojuegos como los de estrategia y lucha tomen las decisiones. En cuanto a memoria se

refiere el único juego que lo posee es el “Laberinto del Saber”, donde el diseñador ve a la memoria como un componente de la entidad inteligente, conectada directamente con los sensores del mismo. En cambio, el diseño propuesto relaciona la memoria con la clase cerebro, que es donde realmente se utiliza para evitar sobrecargar al agente inteligente con información innecesaria, pues se como había planteado no todas las entidades la requieren. Además el cerebro mismo es el encargado de actualizar la memoria, con la información relevante obtenida por los sensores.

3.1.5 Aprendizaje

Los productos realizados en la facultad generalmente no incluyen aprendizaje, sino son pensados para sistema de causa- efecto, es decir cómo responder a un grupo de cambios, pero con comportamientos predefinidos. Solamente en el juego “Rápido y Curioso”, quien posee una red neuronal que permite al automóvil actuar sobre situaciones que la máquina de estado finita no contempla evitando así un comportamiento rígido. La estructura de la solución está pensada para que se le puedan añadir aprendizaje, dentro de la interfaz de toma de decisiones, pues los algoritmos empleados para el logro del mismo tributan a las decisiones que debe tomar el agente inteligente. Aunque, a la propuesta de Frank Puig utilizada en el videojuego “Laberinto del Saber”, se le puede incluir técnicas de aprendizaje como un sub-cerebro más dentro de la misma.

3.1.6 Inteligencia de Estrategia

Ninguna de las arquitecturas anteriores tiene incorporada inteligencia de estrategia. La arquitectura propuesta permite la inclusión dentro del sistema de decisiones de inteligencia táctica, proponiendo una separación entre ellas, ya que la misma puede lograrse desde dos visiones diferentes.

Como se había planteado en el Capítulo 1 cuando hablamos de este tipo de inteligencia, la entidad no solo tiene que preocuparse de ella, sino que prima la decisión del grupo. Para lograrlo se utilizó el patrón composición, que permite tratar a un grupo de entidades, como un agente creíble, dándole al grupo un cerebro propio desde el cual puede planificar una estrategia y comunicarle a cada entidad individual, el rol que desempeña dentro de ella para que el actué.

3.1.7 Sensores

La utilización de sensores es un denominador común entre todas las arquitecturas las que generalmente poseen un grupo de sensores, que además de darle la información al agente son quienes la obtienen, dentro de ese caso tenemos a la de “Rápido y curioso” y “Aduana”. Los dos motores restantes, si emplean los sensores al igual que la arquitectura propuesta como la clase encargada de encuestar al mundo y

realizar los cálculos requeridos para brindar a la entidad la información que necesita. Se puede decir además que a diferencia de “Laberinto del Saber” y la arquitectura propuesta el resto no poseen sensores que le permitan captar la información auditiva, por lo que el agente no tiene este tipo de información. El diseño de los sensores en el videojuego “Laberinto del Saber”, los relaciona con la memoria mediante composición, no obstante, el diseño elaborado no propone esta relación sino que relaciona los sensores únicamente con la entidad inteligente.

3.1.8 Cerebro

Todas las arquitecturas menos la biblioteca *Library on Crowd Behaviors for Videogames* que no lo posee, utilizan el cerebro como la clase que controla las acciones, de la cual heredan dos sub-cerebros uno que se encarga de definir la acción y otro que se encarga de la navegación. La solución expresada en el trabajo sigue viendo al cerebro como la clase que controla la navegación y las decisiones, pero estas partes no la ve como sub-cerebros, sino la relación es de asociación (composición y agregación). Además, de que permite que en caso de aparecer un componente nuevo agregarlo relacionandolo al cerebro de la entidad, lo que también se puede realizar con el diseño propuesto por Frank Puig para una biblioteca de IA para videojuegos. El enfoque que se le dio es diferente porque las relaciones que se establecen con los componentes restantes, no se manifiestan como sub-cerebros, sino como componentes del mismo.

3.1.9 Navegación

La navegación en los productos estudiados se logra mediante comportamientos de locomoción para lograr el movimiento de los agentes en el área, pero la no existencia de una interfaz impide la inclusión de nuevas formas de navegación como el movimiento cinemático. Para ello se incluye una nueva clase interfaz con la cual se relacionarán las formas de movimiento para agentes que el diseñador del juego defina.

3.2 Aplicación del diseño base extendido a Videojuegos Conocidos

Desafortunadamente, la documentación sobre la arquitectura presente en la Inteligencia Artificial de los videojuegos y simuladores más utilizados actualmente es mínima. Quizás, debido a que gran parte de los mismos se realizan con licencias propietarias, que no revelan este tipo de detalle, otro de los inconvenientes es que nuestro país no tiene acceso a muchas de las fuentes donde estos resultados son publicados.

Producto a que esta documentación es inexistente, para la validación de esta investigación se han observado los comportamientos de los agentes autónomos en diferentes tipos de videojuegos. Todo ello

con el objetivo de conocer el grado de inteligencia que poseen los mismos, para poder verificar si la estructura propuesta en el Capítulo 2, puede lograr que los agentes se comporten de esa manera.

El estudio incluirá juegos como RPG y Shoteer. A continuación describiremos algunos de ellos:

3.2.1 Videojuego de rol

Este tipo de juego, se caracteriza por la inclusión de un gran número de NPC a lo largo del juego. Como el jugador tienen un papel a través de la historia, la inteligencia del mismo debe centrarse en hacer el mundo lo más realista posible. Para ello los NPC completan el mundo de ficción creado para el juego y se dedican a desempeñar cualquier papel, desde un transeúnte hasta el enemigo principal de la trama.

El *Assassin's Creed* II perteneciente a la zaga de los *Assassin's Creed*, que ya tiene su tercera entrega en este año, fue el escogido para la realización del estudio de sus entidades inteligentes. Es un videojuego de acción/aventura histórica, desarrollado por Ubisoft. La trama comienza cuando Desmond Miles, un barman secuestrado por los templarios en el año 2012, consigue huir de la prisión de Abstergo gracias a Lucy Stillman, una asesina actual. Con el uso de una máquina llamada Animus, es transportado al renacimiento donde conoce a su antepasado Ezio Auditore De Florencia. La acción se desarrolla desde el año 1476 hasta 1499 durante este tiempo Ezio Auditore dará caza a todos los responsables de la muerte de su familia.

Figura 17: *Juego Assassin´s Creed II*

En el desarrollo del juego se pueden observar gran cantidad de NPC que recrean la vida de la ciudad de Florencia. Se podrán contratar a ladrones, cortesanas, y mercenarios para ayudar en las misiones.

Además de un gran número de transeúntes que reaccionan antes acciones del jugador tales como: arrojar dinero al suelo o contratar los servicios de los citados anteriormente e incluso lanzar una bomba de humo para dejarlos inutilizados por unos instantes. Los soldados, son también un grupo fundamental dentro de la trama, pues constituyen enemigos de Ezio dejándonos ver situaciones de combate. De esta gama de personajes escogimos a los transeúntes para la modelación de la estructura, debido a que son las entidades que más se utilizan para recrear la vida de la ciudad.

Transeúntes: Reaccionan ante las apariciones sorpresivas de Desmond se molestan o cambian de posición e incluso le hablan. Otras de las acciones que realizan estos agentes es discutir contra su oponente si detectan que han sido robados. Son capaces de mostrar sensación de dolor una vez que son vencidos en una pelea. Cuando se arroja bombas de humo a su alrededor, estos agentes tienen la habilidad de reaccionar ante este suceso dando la ilusión de inteligencia.

Después del estudio realizado se observó que el diseño propuesto se adapta al comportamiento del personaje anterior. La estructura del componente de percepción permite se le brinde al agente información visual y auditiva, para que pueda responder a las provocaciones del jugador. Los sensores pueden ser capaces de manejar la cantidad de informaciones que requiera. La navegación requiere de movimientos dinámicos por lo que con la clase interfaz **CNavegacion** se relacionaran comportamientos de locomoción como el seguimiento y la evasión. Estos algoritmos deben incluir comportamientos grupales, porque ninguno de los NPC choca o se separa demasiado cuando caminan al lado o en grupo. Es probable que en estos agentes autónomos se haya empleados técnicas de búsqueda de caminos, funcionalidad que también sustenta el diseño elaborado en la clase **CBCam**. La toma de decisiones, es fácilmente manejada en el diseño, pues a la clase interfaz **CDecision** tendrá relación directa con la o las técnicas que el juego requiera. Se puede afirmar que el diseño propuesto, está construido para entidades más complejas, porque estas no requieren la formación de grupos. Tampoco la elaboración de las decisiones es por niveles, pues no incluyen inteligencia de estrategia, ni aprendizaje; por lo que un simple sistema causa-efecto podría cumplir con los requerimientos de implementación de los mismos.

3.2.2 Videojuego tipo Shooter

La trama de este tipo de videojuego, está caracterizada por una serie de misiones, en el cual el jugador tendrá que demostrar todas sus habilidades de combate para llevarlo por un mundo de ficción con versátiles escenarios, en ellos hay una gran cantidad de enemigos acerca de los cuales se centra el trabajo de la inteligencia artificial.

Tom Clancy's Splinter Cell: Conviction es el quinto episodio de la serie de videojuegos Splinter Cell desarrollado y publicado por Ubisoft. Está dentro de los videojuegos de tipo shooter en este caso en tercera persona. La entrega de la serie se puso a la venta en abril del 2010.

Figura 18: *Tom Clancy's Splinter Cell, Conviction*

El videojuego cuenta la historia venganza de Sam Fisher, por la muerte de su hija. El personaje es ayudado a capturar al asesino por una ex compañera llamada Anna. A las funcionalidades anteriores se le agregan dos más: “la última posición conocida”, que hace que cuando descubren al jugador aparezca su silueta en el último lugar donde fue visto y la segunda es que permite marcar a varios enemigos y dispararles de una sola vez.

Los agentes inteligentes que se muestran en el videojuego, tienen como función la de atacar al personaje principal. Los que presentan mayor nivel de inteligencia son los comandos militares que existen en la escena, por lo que el análisis se lo realizaremos a los mismos.

Militares: el nivel de percepción de los mismos es elevado, producto a que cuentan con una visión que les permite observar a través de las paredes. Una de las acciones que realizan es la reacción ante el ruido que el jugador provoca. Existen niveles de comportamiento de grupo, porque cuando se activa “la última posición conocida”, se dirigen hacia la silueta e incluso disparan sobre ella. Además se observa, que deben tener implementados conocimientos tácticos para la entrada a las habitaciones. La cooperación entre ellos se nota producto a que se cubren unos a otros, se dispersan, dan voces de mando y si encuentran un compañero muerto, empiezan a buscar al jugador.

El diseño propuesto permite el trabajo de grupos de entidades con la utilización del patrón composición, el mismo permite que la clase **CEntidad** trate a un comando militar como una sola. Debido a esto se puede implementar las estrategias para las entradas a las habitaciones, donde cada jugador en específico, desarrolla un papel dentro de ella. La clase **CGrupoE** tendrá contenida las implementaciones necesarias dentro de su clase **CDecision** planifique que hacer cada vez. Independientemente de la estrategia escogida cada soldado en específico seleccionara como va a realizar el movimiento para cumplir con ella. Además, de posibilitar este diseño la comunicación entre los miembros del comando. La estructura propuesta, para el logro de la percepción se aplica perfectamente a los niveles que estas entidades requieren. Porque permite obtener los datos topológicos, medio ambientales y los objetos que están en movimiento, dentro de la escena, información suficiente para el logro de las decisiones y la navegación. Los comportamientos observados son de naturaleza dinámica por lo que se implementaran directa con la interfaz **CNavegacion,** ya sea heredando de ella directamente o mediante la utilización de una clase padre de la cual heredaran los comportamientos de locomoción que implemente el juego.

CONCLUSIONES

Para el cumplimiento de los objetivos de esta investigación, fue necesario hacer un estudio de las técnicas de inteligencia artificial y cómo podrían ser incluidas en videojuegos y simuladores. El estudio realizado facilitó la creación del diseño elaborado, pues permitió cambiar el enfoque del trabajo no solo a un agente individual, sino a nivel de sistema de software.

El diseño elaborado posibilita, además, el trabajo tanto con agentes individuales como con un grupo de ellos, pudiéndosele incorporar todas las funcionalidades que poseen los sistemas multiagentes.

El módulo diseñado propone un componente de percepción global y una serie de sensores para que las entidades inteligentes puedan interactuar con el entorno del videojuego o simulador.

Como resultado se obtuvo un modelo que puede adaptarse a las necesidades que el videojuego o simulador que se vaya a desarrollar requiriera, pues el sistema permite incorporarle técnicas de toma de decisiones, modelos cognitivos y técnicas para la ejecución de la acción de manera flexible y fácil.

RECOMENDACIONES

- Incluir de técnicas de comunicación entre sistemas multiagentes al diseño elaborado.
- Aplicar el modelo propuesto a los módulos de IA que se vayan a realizar en la Facultad 5.

REFERENCIAS BIBLIOGRÁFICAS

Agentes Inteligentes. Aplicación a la Realidad Virtual. **Bergolla, Yuniesky Coca.** Cuba : s.n.

Arias, Proenza, Martin. 2010. *Library on Crowd Behaviors for Videogames.* La Habana : s.n., 2010.

Bates, Bob. 2004. *Game Design, Segunda edición.* Estados Unidos : Thomson Course Technology PTR, 2004. 1-59200-493-8.

Buckland, Mat. 2005. *Programming Game AI by Example.* Estados Unidos : Wordware, 2005. 1556220782.

Césari, Matilde. 2007. *Métodos de adquisición de conocimiento y modelado de sistemas expertos bayesianos de ayuda a la toma de decisiones.* Mendoza : s.n., 2007.

Erick Gamma, Richard Helm,Ralp Jhonson, Jhon Vlissides. 1997 . *Desing Patters: Elements od Reusable Object - Oriented software.* s.l. : Addison Wesley Longman, 1997 . 0-201-63498-8.

Ferber, J. 1999. *"Multi-Agent Systems".* s.l. : Addison-Wesley, 1999.

Funge, John David. 2004. *Artificial Intelligence for Computer Games.* Massachusetts : A K Peters, 2004. 1-56881 - 208-6.

Gavaldà, Duch, Jordi y Navarro, Tejedor Heliodoro. 2011. *Inteligencia Artificial.* Universidad Oberta de Catalunya : s.n., 2011.

Gómez, Cangas, Aliuchy, Prieto, Pulido, Reinier. junio, 2009.. *Ingeniería Inversa a Biblioteca de Inteligencia Artificial para Videojuegos.* La Habana : s.n., junio, 2009.

Ivan Jacobson, Grady Booch, James Rumbaugh. 2000. *El Proceso Unificado de desarrollo de software.* Madrid : Addison Wesley, 2000.

Lianet Campos, Reinier Melian. 2010. *Herramienta para la creación de mallas navegación en entornos virtuales.* La Habana : s.n., 2010.

Luis Giraldo, Luisiana Zapata. septiembre, 2005. *Herramientas de desarrollo de ingeniería de software para linux.* septiembre, 2005.

Martín del Brío, Bonifacio, Molina, Sanz, Alfredo. 2001. *Redes Neuronales y Sistemas Difusos, Segunda Edición.* Zaragoza, España : s.n., 2001.

Millington, Ian. 2006. *Artificial intelligent for games .* San Francisco : Elsevier Inc, 2006. 978-0-12-497782-2.

Placeres, Frank Puig. *Advanced Perception System For Games.* Universidad de las Ciencias Informáticas, Cuba : s.n.

Real Academia Española. 2011. Real Academia Española. [En línea] IBM, 2011. [Citado el: 15 de 2 de 2011.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=inteligencia%20artificial..

Rodríguez, Reyes, Dairo. junio 2008. *Algoritmos bioinspirados.* junio 2008.

Rodríguez, Wladimir. 2011. Inteligencia Artificial Clase #2 :Agentes Inteligentes. Venezuela : s.n., 2011.

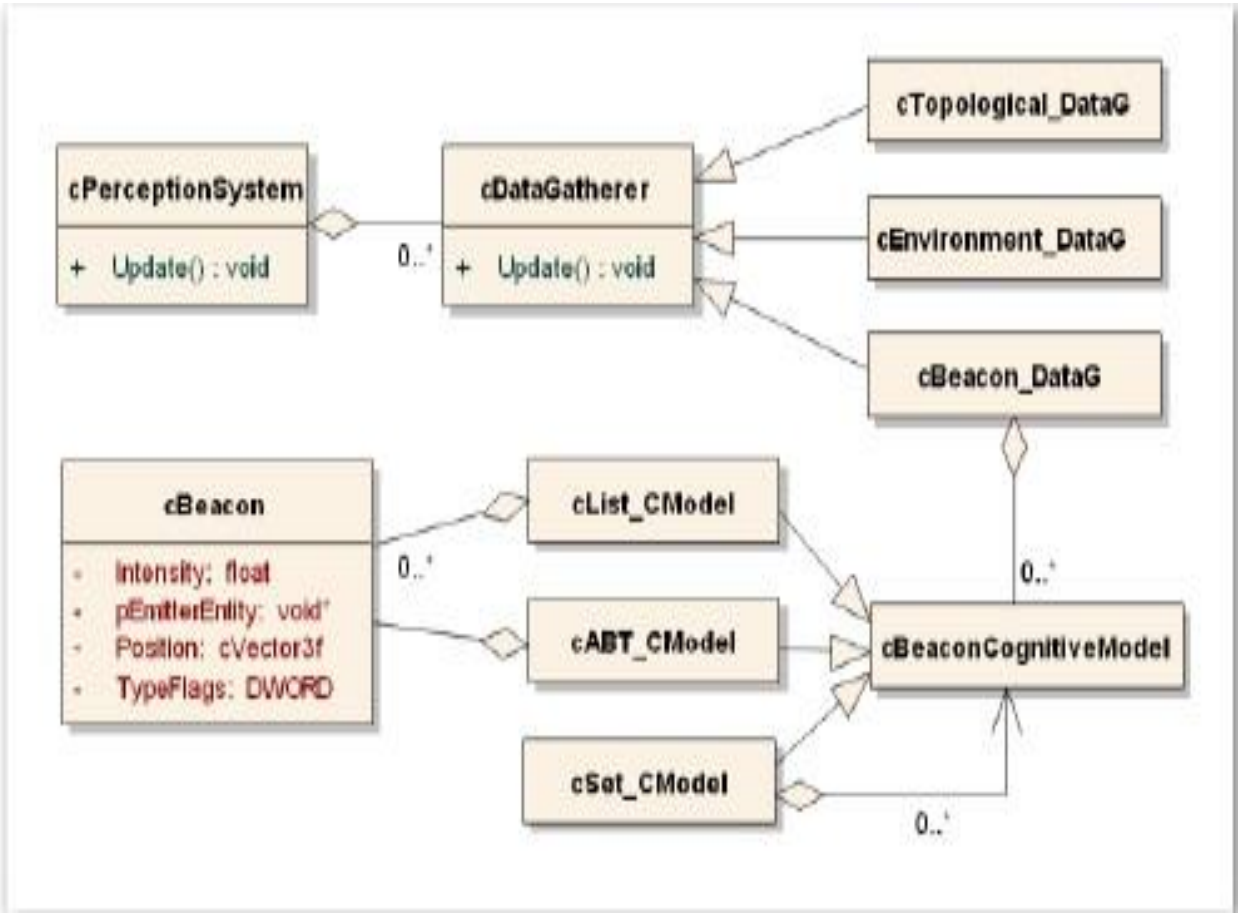
Russel, Norving. 1995. *IA Bible Artificial Intelligente: A Modern Aproach.* 1995.

Sardiñas, Alexander. *Aplicación del algoritmo Min -Max en el desarrollo de juegos de contrincantes en un entorno virtual.* La Habana : s.n.

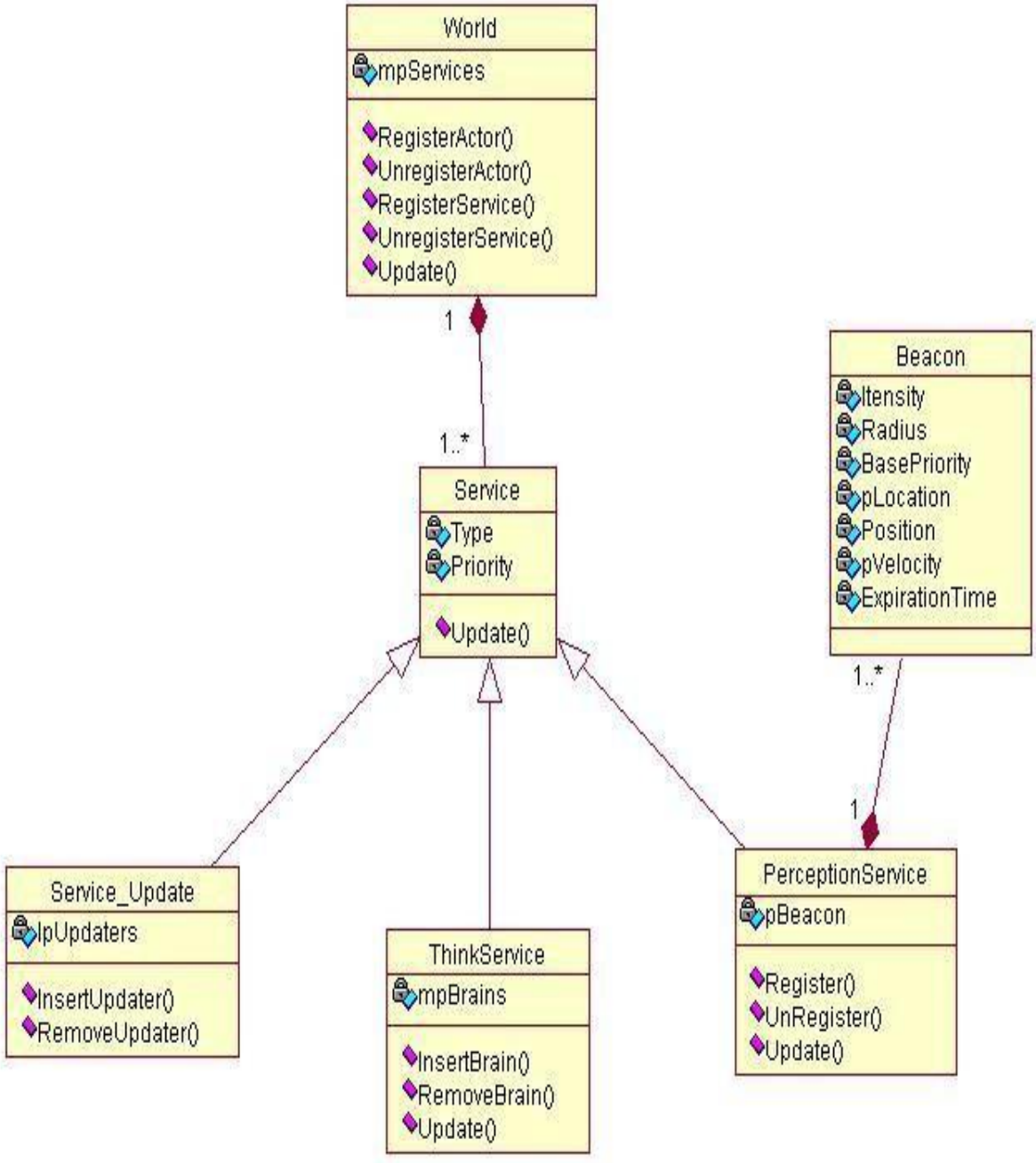
Schwab, Brian. 2004. *IA Game Engine Programming.* Massachusetts : Charles River Media, 2004. 1-58450-344-0.

Smed, Jouni, Hakonen Harri. 2006. *Algorithms and networks computer.* Finlandia : John Wiley & Sons, 2006. 978-0-047-01812-5.

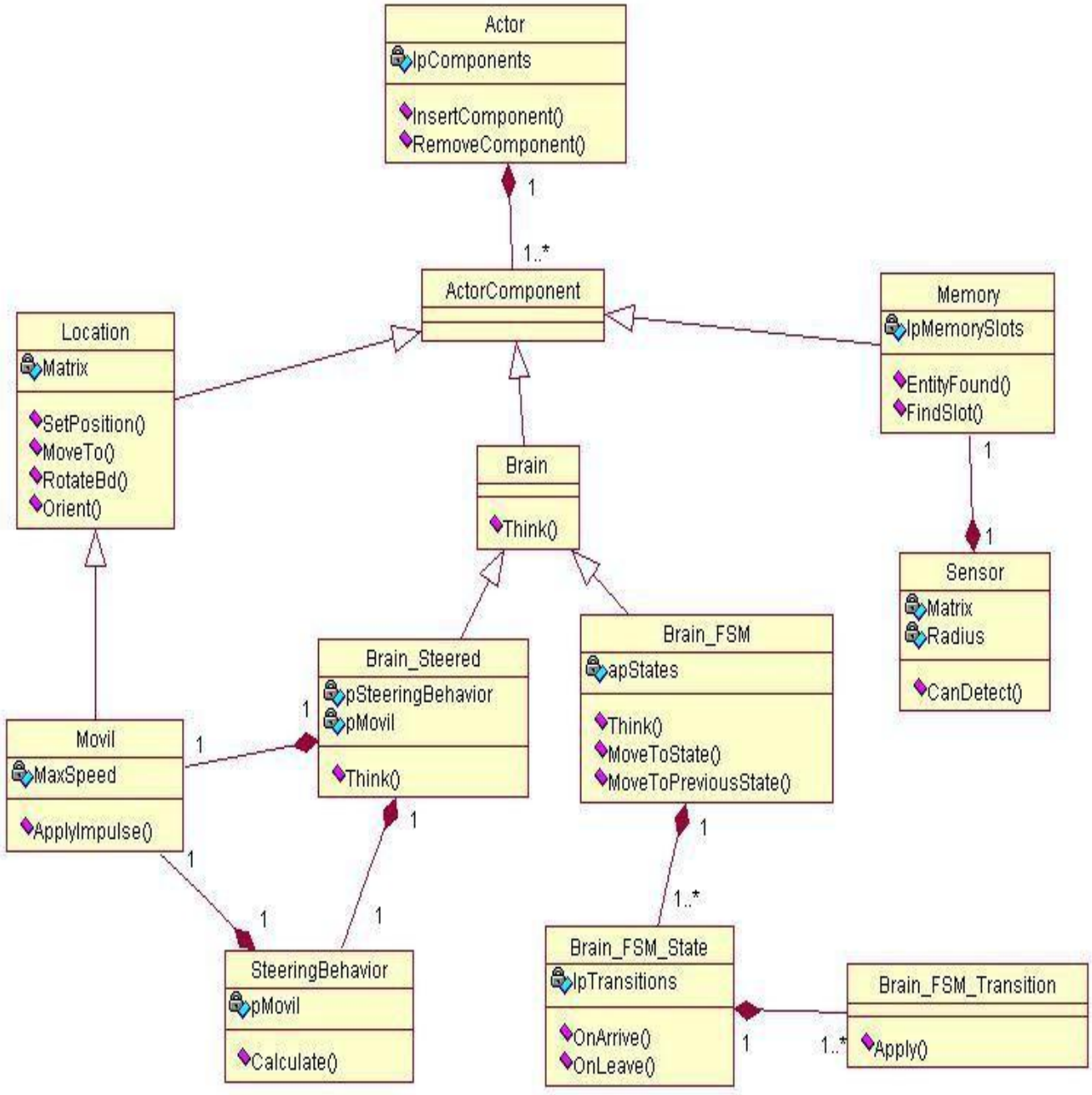
ANEXOS



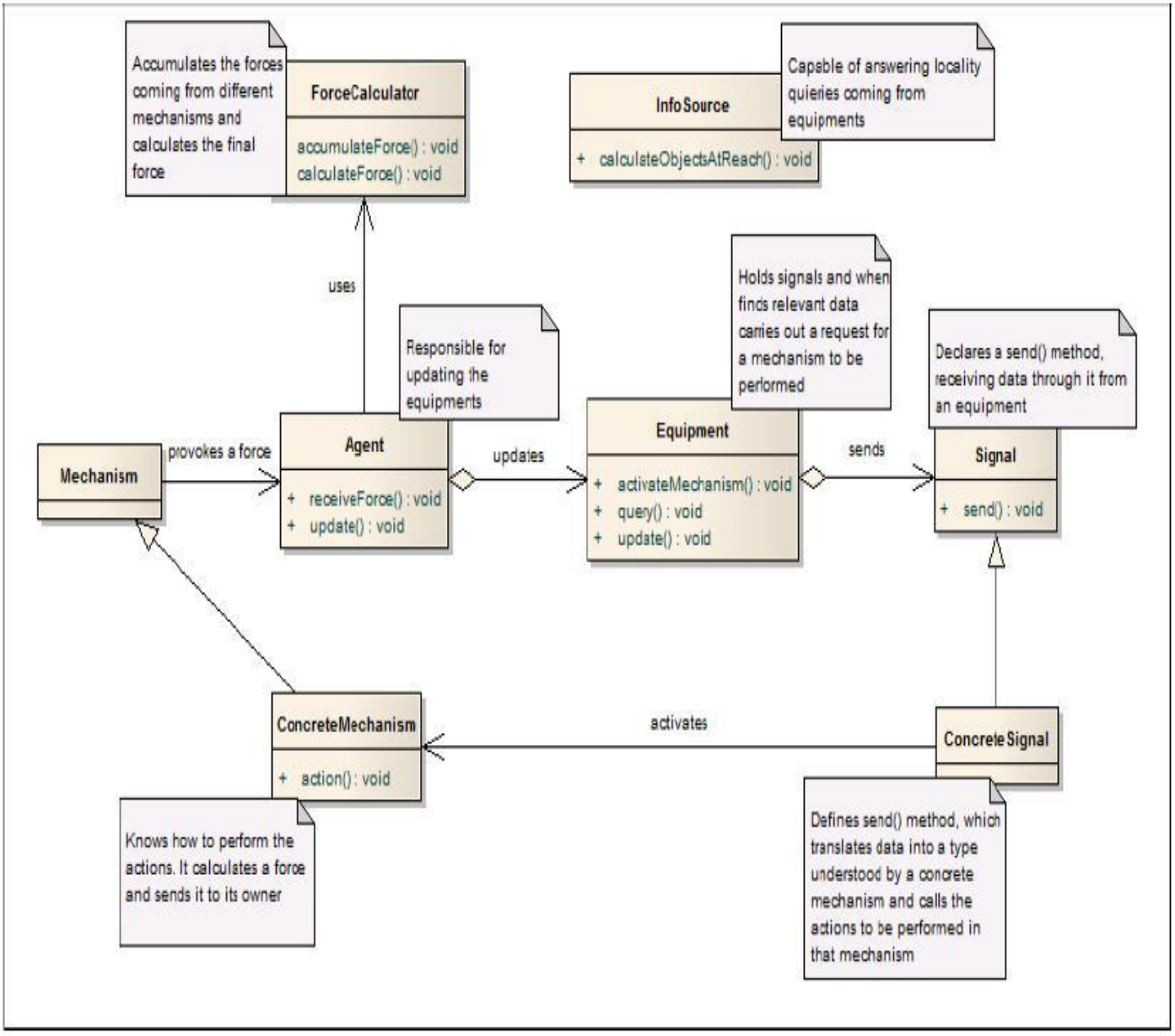
Anexo 1 : *Arquitectura de Frank Puig para un sistema de percepción*



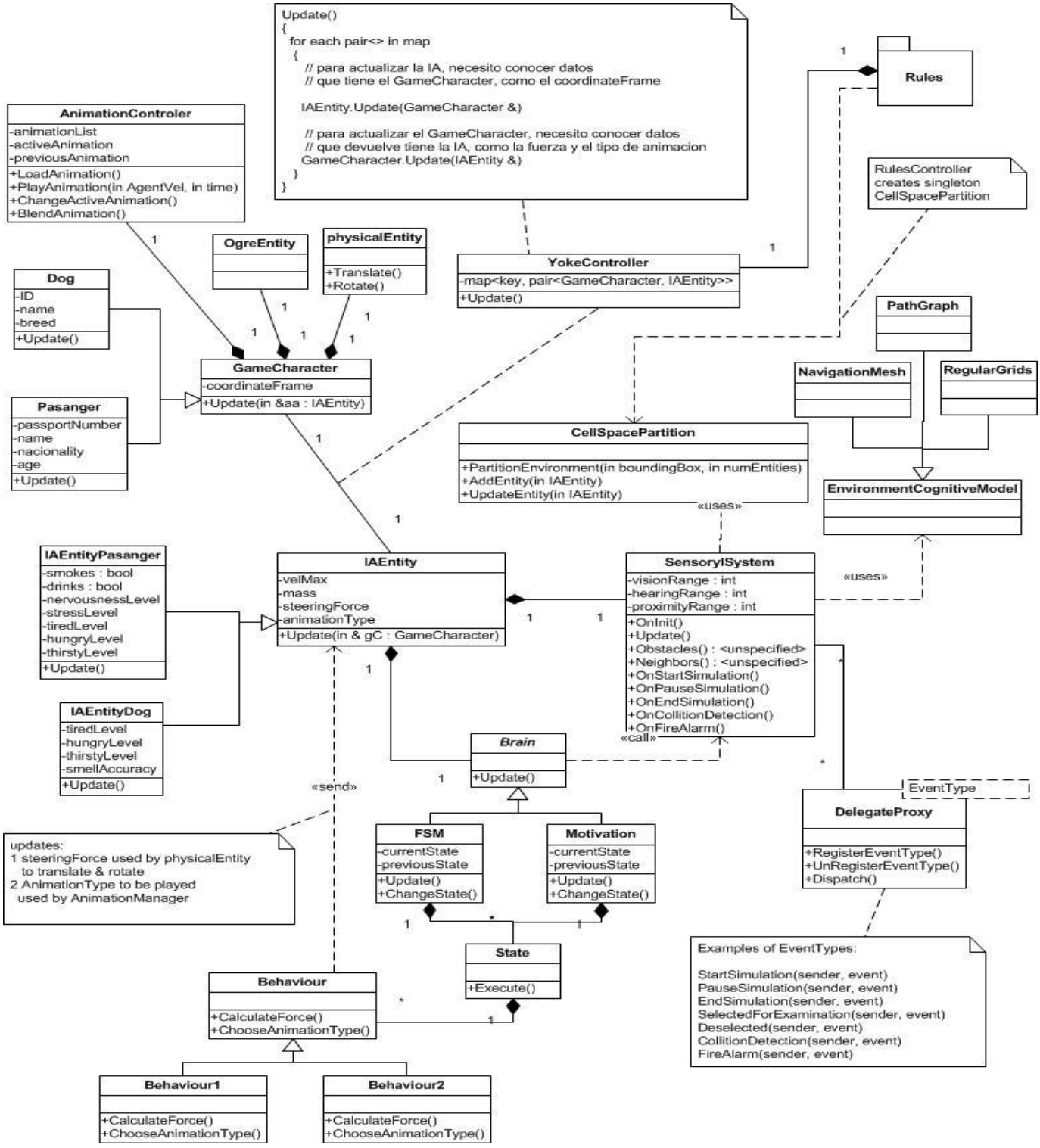
Anexo 2: *Diagrama de clases para el paquete Gestión del mundo juego "Laberinto del saber"*



Anexo 3: *Diagrama de clases del paquete Gestión de actor videojuego "Laberinto del Saber"*



Anexo 4: *Diagrama de clases de la biblioteca "Library on Crowd Behaviors for Videogames"*



Anexo 5: *Diagrama de clases del módulo de IA "Simulador Aduana"*

Glosario de términos

Entidad inteligente: Agente capaz de simular el comportamiento de humanos, animales u otros objetos en un entorno virtual.

Mundo: Entorno generado por computadoras inspirado o no en la realidad.

Estructura: Distribución y orden de las partes importantes que componen un todo, que pueden o no estar interrelacionadas entre sí.

Componente: Entiéndase por aquello que forma parte de la composición de un todo. Se trata de elementos que, a través de algún tipo de asociación, dan lugar a un conjunto.

NPC: Entidades que no están bajo el control directo de los jugadores.

Scripting: Se trata de un conjunto de líneas de código que realizan una determinada función, en un momento determinado que el diseñador del sistema considere.

Máquina de estados difusa: Técnica de inteligencia artificial que a diferencia de la máquina de estado, están construidas con una noción de lógica difusa en su implementación.

Renderizar: Término usado en jerga informática para referirse al proceso de generar una imagen desde un modelo.

Discretización: Proceso que convierte la geometría del mapa de juego a nodos de grafo de navegación.

Reglas if-then: Conjunto de reglas de producción que constituyen primordialmente los sistemas de expertos basados en reglas.

Lenguaje de Modelado: Referido al lenguaje UML, que es utilizado para visualizar, especificar, construir y documentar los artefactos de un sistema.

Programación orientada a objetos: Es un paradigma de programación que usa objetos y sus interacciones, para diseñar aplicaciones y programas informáticos.

Entorno virtual: Dígase el mundo por el cual se mueven las entidades inteligentes.

Waypoints: Refiérase a los puntos de interés tácticos para un agente dentro de un entorno virtual.

Módulo de inteligencia artificial: Término que se refiere a la parte de un sistema de software que le incorpora al mismo Inteligencia Artificial.