TÍTULO

Detallando el nombre de la asignatura, el nombre que los alumnos quieran dar a su práctica (opcional), el número de grupo y los nombres de los integrantes.

Comportamiento de personajes

Grupo 2

Mario Belén Rivera, Sergio Cruz Serrano, Mireya Funke Prieto, Samuel Ríos Carlos, Enrique Sánchez de Francisco

# DESCRIPCIÓN GENERAL

Texto libre en el que se explicará el **funcionamiento de la escena** o juego. Es importante detallar por lo menos el **funcionamiento general**, describir los **comportamientos planteados**, **motivaciones e interacciones diseñadas** con las explicaciones pertinentes. Si el programa es interactivo, se deberán **detallar todos los controles**. También se facilitará la dirección del **repositorio de GitHub**.

**DESCRIPCIÓN GENERAL**

*Mashallow: Pilferage in YolkTown* es un juego 3D de puzles e investigación para un jugador ambientado en un poblado que celebra sus fiestas. En cada partida se deberá encontrar a un ladrón que, aprovechando el jolgorio, deambulará por el pueblo robando a los aldeanos. Marshallow, el Marshall, tiene el deber de detener a dicho ladrón, para ello hablará con pueblerinos a los que han robado y otros que han sido testigos de un robo. Ambos perfiles de aldeanos te darán los datos necesarios para encontrar al ladrón mediante descripciones sobre su apariencia, sin embargo, algunos no alcanzarán a verlo bien y sus datos serán de dudosa fiabilidad, dificultando la búsqueda del asaltante.

**ESCENA Y LEVEL DESIGN**

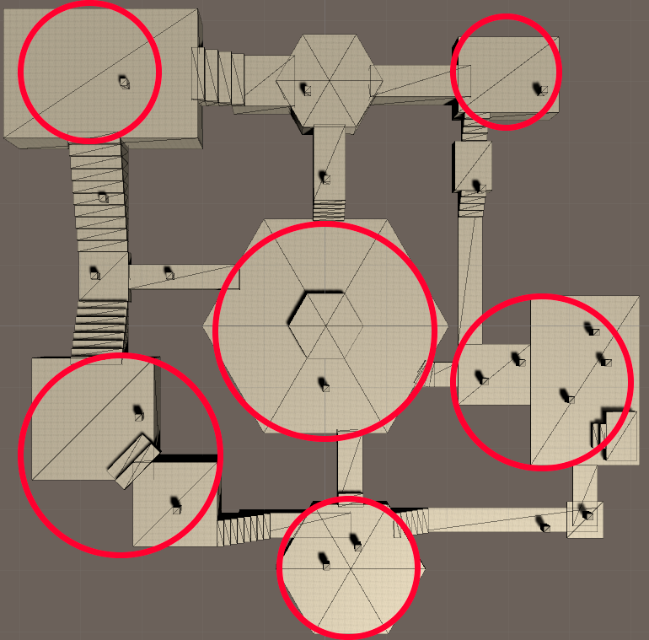
Como se ha comentado antes, el juego transcurre en un poblado, y este está dividido en seis zonas diferenciadas: el parque, el ayuntamiento, la forja, la plaza, el mercado y las viviendas. Cada zona tiene una forma distinta, algunas son más alargadas y estrechas, y otras como la plaza tienen más espacio, pudiendo albergar a un número mayor de pueblerinos. También cabe decir que las hemos repartido en cuatro niveles de altura, estando el parque en la zona más baja y el ayuntamiento en la más alta, añadiendo un componente de verticalidad al juego. Todo lo comentado aporta más variedad no solo visual si no también jugable, ya que cambian tanto las formas de las zonas como los elementos de estas, pudiendo encontrar más maleza en el parque, en contraposición a la plaza donde lo más destacado serán los bancos, las farolas y una fuente central, variando así el movimiento tanto en las zonas como entre ellas.

El diseño de nivel de “Marshallow: Pilferage in YolkTown” es algo que se ha tenido claro desde prácticamente el primer momento, ya que se quería crear un escenario no demasiado grande para concentrar la acción de robo del ladrón y a la vez un lo suficientemente amplio como para dar sensación de libertad al jugador.

Se ha decidido diseñar un nivel en el que las 6 zonas de interés están distribuidas en forma de estrella con una de ellas en el centro (la plaza del pueblo) y las cinco restantes alrededor de esta. Al ser un nivel compacto, se genera una jugabilidad más frenética ya que no hay mucha distancia entre una zona y otra, intentando así no alargar demasiado las partidas.

Esta distribución se aprovecha de tal forma que el número de aldeanos por zona sea equitativo, es decir, que no se dé el caso de que haya un exceso de habitantes o falta de ellos en una zona. También se tendrá en cuenta la distribución a la hora de que el ladrón decida robar, ya que por norma general lo hará dentro de dichas zonas (esto se explicará más a fondo en la descripción detallada de los agentes).

A continuación, se muestra una imagen en vista cenital del blocking del escenario, resaltando en rojo las zonas comentadas anteriormente. Se puede observar cómo encontramos múltiples conexiones entre estas, generando así un gameplay fluido.



Cabe comentar que hemos interconectado todas las zonas desde varios puntos para así aumentar la dificultad a la hora de encontrar al ladrón, debido a que de darse el caso en el que sólo hay un pasillo de salida desde una de las zonas, el ladrón solo puede haber escapado por este y cabe la posibilidad de que se encuentre al jugador, haciendo muy fácil y aburrida la experiencia de juego.

**COMPORTAMIENTOS**

* **Aldeano:**
  + **Aldeano que ha sido robado (víctima):** Este PNJ te dará siempre dos datos sobre el ladrón.
  + **Aldeano que presencia un robo (testigo):** Este PNJ sólo te da un dato sobre el ladrón.
  + **Aldeano viandante:** Simplemente deambula por el escenario e interactúa con sus elementos.
* **Ladrón:** Roba a aldeanos, dejando un margen de tiempo entre robo y robo para no levantar sospechas. Además, puede despistar al Marshall dando datos falsos haciéndose pasar por un testigo.
* **Marshall:** Personaje jugable que se mueve libremente por el escenario pudiendo obtener datos del ladrón, si logra encontrar víctimas de robo y acercándose a estas para obtener dicha información.

**CONTROLES**

Para mover al personaje por el escenario, encontramos dos posibilidades, dependiendo de si se juega a *“Marshallow: Pilferage in YolkTown”* en ordenador o en dispositivo móvil:

* El control en ordenador es mediante clicks de ratón, haciendo click en la parte del escenario a la que se desee moverse. Además, este mismo control es usado para todo, ya sea navegar por la interfaz en los menús o para girar la cámara en mitad de partida.
* En cuanto al control en dispositivos móviles, es exactamente igual que en ordenador, pero sustituyendo los clicks por “taps” o toques en la pantalla.

**REPOSITORIO GITHUB**

https://github.com/lightningopal/Juegos-para-Web-y-Redes-Sociales---Unity

# DESCRIPCIÓN DETALLADA DE LOS AGENTES

En este documento especificaremos detalladamente qué comportamientos van a seguir los personajes de *Marshallow: Pilferage en Yolktown*, el cual cabe destacar que cuenta con dos tipos de PNJs muy diferenciados, los aldeanos y el ladrón. A modo de introducción, explicaremos resumidamente el funcionamiento del videojuego y qué papel tienen dichos PNJs. Todos los personajes cuentan con accesorios, colores del cuerpo y número de ojos distintos al resto, por tanto, cada aldeano tiene sus características distintivas. La gracia es que el ladrón contará con sus propios accesorios, y al ser visualmente igual que un aldeano, se mimetizará entre ellos. Y por si fuera poco, cada partida es diferente al resto, ya que todos los pueblerinos aparecerán con características aleatorias, teniendo en cuenta que no puede haber dos iguales. Para identificar al ladrón, que no es tarea fácil, el usuario hará uso de las declaraciones de los habitantes que han sido tanto víctimas como testigos de un robo, y una vez recabada cierta información el jugador será capaz de diferenciar al ladrón entre todos los aldeanos para así detenerlo y ganar la partida.

A continuación, se explicará en profundidad el comportamiento de todos los personajes que conforman el videojuego.

**\*** Para el comportamiento de los aldeanos y el ladrón, hemos decidido usar *behaviour trees*, ya que son una solución de alto nivel fácil de trabajar y escalables, que permiten cambios de manera rápida y cómoda. Además, nos evita tener que programar la lógica por código y nos lo facilita mediante su herramienta de grafo.

## Aldeanos

**Descripción**: a lo largo del pueblo se podrán encontrar diferentes aldeanos vestidos con diferentes accesorios (sombreros, corbatas...).

* Interacción con el entorno para moverse.
* Interacción con el jugador.
* Interacción con el ladrón cuando este le roba cambiar su estado (emoción, conflicto).
* Dinámica de contenido.
* Análisis --> smart objects.
* Objetivos pequeños.
* Razona: extrae hechos relevantes observados.
* Decidir: que vaya a un sitio y si hay mucha gente que se vaya a otro
* Actuar.

En primer lugar, encontramos a los aldeanos, PNJs que transitan por el escenario recorriendo el mapa de un punto de interés a otro, siendo estos las diferentes partes del pueblo, como por ejemplo el parque, el mercado o el ayuntamiento. El aldeano elegirá aleatoriamente a qué punto va a moverse una vez lleve un tiempo X en el actual, decidiendo una vez en el nuevo punto si se queda o se mueve a otro. Esto sucede para que no se supere el cupo máximo de posibles aldeanos en un mismo punto, consiguiendo de esta forma un mejor reparto de las zonas, ya que nunca se darán los casos donde haya una zona con exceso de habitantes o con ausencia de ellos. Para moverse entre zonas, el agente utilizará una Navmesh.

Como se ha comentado anteriormente, los aldeanos pueden ser tanto víctimas como testigos de robo por parte del ladrón. Por tanto, en total contaríamos con tres tipos de comportamientos de pueblerino: víctima, testigo y viandante. El estado por defecto de un aldeano es el de viandante, pero este cambia al de víctima si le roban o al de testigo si presencia un robo.

Seguidamente se explicarán más a fondo estos posibles estados y cómo se comportan los agentes.

**Flujo en partida del comportamiento de un aldeano (desde el inicio de esta):**

* El habitante aparece de formaaleatoria en una zona (teniendo en cuenta las restricciones de cupo máximo para repartir equitativamente a los agentes).
* Tras pasar X tiempo, elige la zona a visitar a continuación.
* Transita hacia la zona elegida. El agente decide si hacerlo andando o corriendo. No hay problema si durante el trayecto circula por una zona ocupada por el número máximo de aldeanos.
* Al llegar a la zona comprueba si esta cumple el cupo máximo de aldeanos.
  + Si es así, el agente en cuestión elige de forma aleatoria otra zona de interés del escenario, descartando la zona en la que se encuentra actualmente.
  + Si el número de aldeanos es menor al máximo permitido, el agente entrará en la zona.
* Una vez se encuentra en una zona disponible, puede interactuar con los elementos y los aldeanos de esta (la interacción se representa mediante barks en forma de bocadillos que contienen gráficos).
* Vuelta a empezar desde el paso dos.

Este flujo (perteneciente al estado de viandante) se puede ver interrumpido por presenciar o ser víctima de un robo, y su resolución sería la siguiente:

**Aldeano en estado de víctima:**

* Si y solo si el aldeano se encuentra en una zona válida, este es robado.
* Una vez ocurra esto, el agente permanece quieto indicando mediante un bark (en forma de exclamación roja) que ha sido víctima de un robo.
* Cuando el jugador se acerca a la víctima (es decir, se encuentra dentro del radio de interacción del agente), esta le proporciona información referente al ladrón mediante otro bark en forma de bocadillo con dos iconos (pistas) dentro. Llegados a este punto, el agente decide si facilitar un tipo de información u otro:
  + Un 40% de las veces el aldeano recordará sin problemas las dos características del ladrón que proporcionará al jugador, esto se traduce en que la información será fiable y por tanto las dos pistas serán 100% ciertas. Por ejemplo, si en el bocadillo aparecen una mancha de color azul y una corbata, esto querrá decir que el ladrón es de color azul y porta una corbata con un 100% de probabilidad.
  + El otro 60% de las ocasiones, la víctima no será capaz de recordar nítidamente las características del delincuente. Esto se representa visualmente con un simple icono de interrogación encima del bocadillo informativo. En estos casos también se darán dos pistas, pero no estará asegurado el 100% de la fiabilidad en ambas. Esto quiere decir que una de las pistas será veraz y la otra dudosa, lo cual significa que existirá una probabilidad del 70% de que esta última sea falsa (cabe decir que el jugador no tiene forma de saber cuál es cierta y cuál no). Por ejemplo, tomando la misma situación del anterior apartado, en este caso habría una posibilidad del 70% de que o el ladrón no fuera realmente azul o no llevara corbata.
* Cuando o bien el usuario se aleja del aldeano (sale del radio de interacción) o bien han pasado 10 segundos, este vuelve a su comportamiento normal de viandante.

**Aldeano en estado de testigo:**

Todos los habitantes cuentan con un cono de visión que les permite ver un robo si se diera el caso. Por tanto, suponiendo que alguno ve un delito:

* El agente continúa realizando su comportamiento de viandante, pero indica con un bark en forma de interrogación que ha presenciado el hurto.
* Al acercarse el jugador al testigo (es decir, se encuentra dentro del radio de interacción del agente), este se queda quieto y le proporciona información de la misma forma que la víctima, pero con un pequeño cambio. El aldeano decide si:
  + Facilitar información 100% fiable (un 50% de las veces). El testigo habrá visto nítidamente UNA característica del ladrón. Por ejemplo, si en el bocadillo aparecen unos cuernos de cabra esto querrá decir que sin lugar a dudas el delincuente tiene cuernos de cabra.
  + Proporcionar información dudosa (el otro 50% de las ocasiones). Esto se representa visualmente con un simple icono de interrogación encima del bocadillo informativo. Aunque ha presenciado el robo, el testigo no habrá podido discernir bien ninguno de los atributos del ladrón, lo cual quiere decir que la única pista será cierta sólo el 50/60% de las veces. De nuevo, tomando el mismo ejemplo de antes, en este caso la posibilidad de que el ladrón tenga cuernos de cabra es del 50/60%.
* Cuando o bien el usuario se aleja del aldeano (sale del radio de interacción) o bien han pasado 20 segundos, este vuelve a su comportamiento normal de viandante.

**Tabla de percepciones (condiciones?):**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Implementación** | **Acceso** |
| Visual | Cono de visión |  |
| Smart object zona | Área dentro de una zona clave |  |

**Tabla de acciones:**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Implementación** | **Efectos** |
| Dar información |  |  |
| Estar quieto |  |  |
| Merodear zona |  |  |
| Elegir zona |  |  |
| Moverse |  |  |

Diagrama

Descripción generada automáticamente**Behaviour tree (diagrama UML)\*:**

**Condiciones:**

* **Soy víctima:** Si el aldeano ha sido robado.
* **Marshall cerca:** Si el jugador se acerca lo suficiente.
* **Soy testigo**: Si el aldeano ha presenciado un robo.
* **Estoy en zona**: Si el aldeano se encuentra en una zona de interés, visitándola no de paso.
* **No tengo destino:** Si el aldeano no sabe a qué zona moverse.
* **Estoy en destino:** Si el aldeano ha llegado a la zona que tenía previamente elegida.
* **Hay hueco:** Si la zona en la que el aldeano se encuentra tiene hueco suficiente para entrar.

**Acciones:**

* **Dar información:** Un bark con forma de bocadillo que contiene información sobre el ladrón ya sea información verdadera o dudosa.
* **Estar quieto:** El aldeano se queda en una posición fija.
* **Merodear zona:** El aldeano se mueve libremente por la zona con cierta probabilidad de mostrar mediante barks en forma de bocadillo, su estado de ánimo o la interacción con algún elemento.
* **Elegir zona:** Se elige una zona a la cual se desea moverse, pudiendo repetirse la última elegida.
* **Moverse:** Transitar hacia la zona elegida.

**Nodos Decoradores:**

* **Tiempo:** Hacer la acción durante un tiempo determinado previamente.
* **Elegir (Correr o Andar):** Seleccionar de forma completamente aleatoria si para desplazarse a su próximo destino se va a hacer andando o corriendo.

\*Para un mejor visionado del diagrama, se ha adjuntado una imagen en el entregable.

**Diagrama:** [**https://www.teatroabadia.com/en/uploads/documentos/iagramas\_del\_uml.pdf**](https://www.teatroabadia.com/en/uploads/documentos/iagramas_del_uml.pdf)

## Ladrón

Evaluación de la situación y toma de decisiones.

Algoritmo de decisión.

En segundo lugar, encontramos al ladrón, un PNJ que rondará la ciudad con un comportamiento similar al de los aldeanos para así no llamar la atención, sin embargo, cada cierto tiempo, este agente robará a algún pueblerino. Para ello el ladrón deberá encontrarse en una zona de interés y además haber alguna víctima a la que robar dentro de su área de visión, todo esto sin dejar de lado que, si el Marshall se acerca demasiado, el robo se cancela y disimula transitando hacia otra zona. Para moverse entre zonas, el agente utilizará una Navmesh.

El ladrón cuenta con dos estados diferentes, el estado de robo o el estado de disimular, siendo el primero donde se da la situación en la que el ladrón actúa como tal y el segundo un estado donde se comporta como los demás aldeanos.

**Flujo en partida del comportamiento del ladrón (desde el inicio de esta):**

* El ladrón aparece de formaaleatoria en una zona (teniendo en cuenta las restricciones de cupo máximo para repartir equitativamente a los agentes).
* Tras esperar un breve periodo de tiempo, ejecuta un robo (con las condiciones pertinentes, explicadas más adelante).
* Huye de la zona y decide si disimular siendo testigo de robo o simplemente deambular.
* Vuelta al paso dos.

**Ladrón en estado de disimular:**

En el estado de disimular, el comportamiento es exactamente igual al del estado de viandante de un aldeano, teniendo en cuenta que éste no podrá ser robado, por lo tanto, jamás ejercerá como víctima.

**Ladrón en estado de robo:**

* En primer lugar, el estado de robo se activa cuando ha pasado cierto tiempo desde el robo anterior para así no llamar tanto la atención y simular un comportamiento realista. Al principio de la partida también hay un pequeño tiempo de espera para robar.
* En segundo lugar, una vez el modo robo se activa, el ladrón comprueba si se encuentra en un punto de interés. En caso de no ser así, busca dicha zona para así ejecutar su robo. En caso contrario, el ladrón elige una víctima.
* Para elegir víctima primero debe de haber aldeanos dentro de su área de visión. En caso de no encontrar a ningún pueblerino, éste elige otra zona a la que ir a robar. Sin embargo, si hay alguien a quien robar, el ladrón escoge de entre todos ellos al que esté situado a menor distancia de su posición (se puede dar el caso en el que el aldeano elegido justamente esté abandonando la zona en este mismo momento, por lo tanto la víctima se encontraría en un pasillo en vez de en un punto de interés).
* Cuando la víctima esté seleccionada, el ladrón se dirige rápidamente a robarle (a una velocidad mayor que la velocidad máxima del aldeano, para que así el robo se ejecute de la forma más rápida posible).
* Una vez ambos colisionen, sucede el robo y el ladrón decide si actuar como testigo o no.
  + En caso de que el ladrón actúe como testigo, un bark en forma de interrogación aparecerá sobre su cabeza dando a entender al Marshall que ha presenciado un robo (tal y como lo haría el aldeano en estado de testigo). Si el Marshall se acerca a él, sucederá lo mismo que sucede cuando el Marshal se acerca a un aldeano testigo y este le muestra un bark de información dudosa (es decir, el que tiene un interrogante en el bocadillo), la diferencia es que en el caso del ladrón esta información tendrá un 100% de probabilidad de ser falsa. La gracia reside en que el jugador no tiene forma de saber esto y simplemente lo confundirá con un testigo normal y corriente.
  + Si el ladrón decide no actuar como testigo, simplemente ejerce la función de aldeano en estado de viandante de nuevo.

Si durante el proceso de robo el Marshall entra dentro de la zona de visión del ladrón, el robo se cancelará y elegirá otra zona a la que ir a robar, empezando de nuevo el proceso.

**Tabla de percepciones (condiciones?):**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Implementación** | **Acceso** |
| Visual | Cono de visión |  |
| Smart object zona | Área dentro de una zona clave |  |

**Tabla de acciones:**

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Implementación** | **Efectos** |
| Ejecutar robo |  |  |
| Hacerse pasar por testigo |  |  |
| Ir hacia la víctima |  |  |
| Elegir víctima |  |  |
| Merodear zona |  |  |
| Elegir zona |  |  |
| Moverse |  |  |

Diagrama

Descripción generada automáticamente**Behaviour tree (diagrama UML):**

**Condiciones:**

* **Puedo robar:** Si el ladrón se encuentra en estado de robo y quiere robar.
* **Marshall lejos:** Si el jugador se encuentra fuera del área de visión del ladrón.
* **He elegido víctima:** Si el ladrón ya tiene objetivo de robo.
* **Estoy junto a la víctima:** Si el ladrón colisiona con la víctima.
* **Estoy en zona:** Si el ladrón se encuentraen una zona de interés, visitándola no de paso.
* **Aldeano(s) cerca:** Si en la zona actual hay algún aldeano.
* **No tengo destino:** Si el ladrón no sabe a qué zona moverse.
* **Estoy en destino:** Si el ladrón ha llegado a la zona que tenía previamente elegida.
* **Hay hueco:** Si la zona en la que el ladrón se encuentra tiene hueco suficiente para entrar.

**Acciones:**

* **Ejecutar robo:** Robar al aldeano y cambiar el estado de robo a false.
* **Hacerse pasar por testigo**: Hacerse pasar por testigo tras un robo durante un tiempo.
* **Ir hacia la víctima:** Moverse desde la posición actual hasta la víctima elegida.
* **Elegir víctima:** Seleccionar a que aldeano robar.
* **Merodear zona:** El aldeano se mueve libremente por la zona con cierta probabilidad de mostrar mediante barks en forma de bocadillo, su estado de ánimo o la interacción con algún elemento.
* **Elegir zona:** Se elige una zona a la cual se desea moverse, pudiendo repetirse la última elegida.
* **Moverse:** Transitar hacia la zona elegida.

**Nodos Decoradores:**

* **Random:** Elegir de forma aleatoria.
* **Elegir (Correr o Andar):** Seleccionar de forma completamente aleatoria si para desplazarse a su próximo destino se va a hacer andando o corriendo.

a) Nombre y descripción textual detallada.

b) Tabla de percepciones (Nombre, Implementación, Acceso)

c) Tabla de acciones (Nombre, Implementación, Efectos)

d) Diagramas UML.

# FLUJO Y ESTRUCTURA DE DATOS

Aquí se detallará:

a) Cómo fluye la información desde y hacia el entorno (con una tabla).

b) Todas las estructuras de datos relevantes para el comportamiento de los personajes, es decir, atributos de los personajes, jugador y entorno que influyan en el comportamiento.

# DESCRIPCIÓN DE LOS ALGORITMOS

Esta sección es opcional. Aquí los alumnos pueden detallar los algoritmos implementados para todas aquellas percepciones o acciones que no sean triviales (por ej. línea de visión, búsqueda de caminos...). La idea es que podamos valorar todo esfuerzo extra que hayáis realizado, por lo que queda a vuestro criterio qué incluir (si es que queréis incluir algo) en esta sección.

Los agentes se comportan de la siguiente manera:

* Aldeanos:
  + Son agentes de la NavMesh del nivel.
  + Tienen un cono de visión para poder ver robos del ladrón.
  + Almacenan información de lo que han visto.
* Ladrón:
  + Es un agente de la NavMesh del nivel.
  + Tiene un radio de visión para evitar robar con el jugador cerca.
  + Almacena información falsa para despistar.

# COMPORTAMIENTOS EMERGENTES

Esta sección es opcional. Aquí los alumnos pueden detallar los comportamientos emergentes que pueden ocurrir en el proyecto.

# REPARTO DE TAREAS

Detalle de los roles ejercidos por cada miembro del grupo. Se debe detallar quién y en qué medida ha realizado las tareas de programación, tareas artísticas, de diseño del juego y de gestión del grupo de trabajo.

**Tareas de Programación:**

* **Programación e implementación de la IA:** Mario Belén Rivera, Samuel Ríos Carlos y Mireya Funke Prieto.

**Tareas artísticas:**

* **Level Design:** Mireya Funke Prieto.
* **Diseño de personajes:** Enrique Sánchez de Francisco.
* **Modelado, texturizado y animación de Personajes:** Sergio Cruz Serrano.
* **Diseño, modelado y texturizado de escenarios:** Mireya Funke Prieto.

**Diseño de juego:**

* **Level Design:** Sergio Cruz Serrano y Enrique Sánchez de Francisco.
* **Documento de diseño de la IA:** Enrique Sánchez de Francisco y Sergio Cruz Serrano.
* **Documento de Testing:** Enrique Sánchez de Francisco y Sergio Cruz Serrano.
* **Diseño de Behaviour Trees:** Sergio Cruz Serrano.

**Gestión del grupo de trabajo:**

# LECCIONES APRENDIDAS

En este apartado los alumnos detallarán, desde el análisis crítico, los factores que hayan podido afectar positiva o negativamente al desarrollo del proyecto, los puntos fuertes y elementos a mejorar.

# LICENCIAS

En este apartado el alumno detallará todas las fuentes y licencias de los recursos empleados en su proyecto, incluidos los propios.

# REFERENCIAS BIBLIOGRÁFICAS

Es muy importante que todas las páginas de la memoria estén numeradas. Si lo consideran oportuno, los alumnos pueden añadir también al final una sección de referencias bibliográficas.