



Escuela
Politécnica
Superior

Gestión y manejo de comportamientos grupales emergentes de entidades en videojuegos



Grado en Ingeniería Multimedia

Trabajo Fin de Grado

Autor:

Borja Pozo Wals

Tutor/es:

Francisco José Gallego Durán

Octubre 2020



Universitat d'Alacant
Universidad de Alicante

Gestión y manejo de comportamientos grupales de entidades en videojuegos

Subtítulo del TFG

Autor

Borja Pozo Wals

Tutor/es

Francisco José Gallego Durán

Departamento de Ciencia de la Computación e Inteligencia Artificial



GRADO EN INGENIERÍA MULTIMEDIA



Escuela
Politécnica
Superior



Universitat d'Alacant
Universidad de Alicante

ALICANTE, 12 de diciembre de 2020

Resumen

título es una herramienta para simular escenarios bélicos como los que podemos encontrar en juegos del género Real Time Strategy (RTS) como son ‘*They are Billions*’ o la saga ‘*Age of Empires*’, desarrollado completamente en C++ para Personal Computer (PC).

La simulación se compone de un escenario, una unidad controlada por el jugador, una serie de unidades que conforman el ejercito bajo sus ordenes y por último, tenemos una serie de objetivos que abatir. Una vez conseguido el objetivo terminará el juego pudiendo el jugador elegir entre repetir el nivel o volver al menú principal.

Justificación y objetivos

A lo largo de la carrera son muchas las asignaturas que requieren y desarrollan habilidades relacionadas con la programación en diversos lenguajes y usos, pero no es hasta el tercer año que se me presenta la oportunidad de desarrollar un videojuego completo.

Durante la asignatura de ‘Fundamentos de los Videojuegos’ tuve por primera vez la experiencia de enfrentarme al desafío que es crear un videojuego, y fue en ese momento cuando me di cuenta de que a lo que me quería dedicar es a desarrollar juegos de forma profesional. A lo largo del cuarto curso junto a los demás integrantes de ‘*Sunlight Studio*’ desarrollamos ‘*Cyborgeddon*’, y fue en este momento que terminé de decidir que es a lo quiero dedicarme en el futuro.

A lo largo de mi vida he jugando una considerable cantidad de juegos entre los cuales se puede apreciar una inclinación por los juegos de aventura y exploración, los Rol Play Game (RPG) y los RTS o de estrategia en general, es por esto que me hace especial ilusión desarrollar un juego de uno de estos géneros.

Teniendo en cuenta esto la elección de desarrollar un juego del tipo RTS para la realización del Trabajo Final de Grado (TFG) se debe a que según mi criterio es el que tiene mayor potencial para ayudarme a mejorar mis habilidades como programador, ya que se basa en unas mecánicas con las que no he trabajado con anterioridad.

Una vez dicho esto, los objetivos planteados para el proyecto son los siguientes:

- Aumentar mis conocimientos de C++
- Comprender mejor el funcionamiento del PC
- Aprender nuevas técnicas de Inteligencia Artificial (IA)
- Profundizar en la arquitectura Entity-Component-System (ECS)
- Desarrollar un producto mediante el cual mostrar mis habilidades

Agradecimientos

*A mi madre Gabriela,
por el increíble esfuerzo que realiza cada día por mí.*

*A mi hermana Raquel,
por haber sido y ser un faro para mí.*

*A mi buen compañero Jorge Espinosa,
por nuestra convivencia estos años.*

*A mi querido amigo Guillermo y su familia,
por el cariño recibido incluso a pesar de la distancia.*

*Ni los reyes ni los gobernantes llevan el cetro,
sino los que saben mandar*

Sócrates.

Para saber hablar es preciso saber escuchar

Plutarco.

Índice general

| | | |
|----------|---|-----------|
| 1 | Introducción | 1 |
| 2 | Marco teórico | 3 |
| 2.1 | Estudio de mercado | 3 |
| 2.2 | Referentes | 3 |
| 2.2.1 | Age of Empires II: The Age of Kings | 4 |
| 2.2.2 | The Age of Empires II: The Age of Kings | 4 |
| 2.3 | Técnicas de inteligencia artificial | 7 |
| 3 | Documento de Diseño del Juego (GDD) | 11 |
| 3.1 | Características | 11 |
| 3.2 | Descripción general | 11 |
| 3.3 | Mecánicas | 11 |
| 3.4 | Unidades | 12 |
| 3.5 | Controles | 12 |
| 3.6 | Pantallas | 13 |
| 3.7 | Estados del juego | 15 |
| 3.8 | Escenarios | 16 |
| 4 | Metodología y fases del producto | 17 |
| 4.1 | Metodología | 17 |
| 4.2 | Iteraciones | 18 |
| 4.2.1 | Iteración 0 | 18 |
| 4.2.2 | Iteración 1 | 18 |
| 4.2.3 | Iteración 2 | 19 |
| | Bibliografía | 21 |

Índice de figuras

| | | |
|------|---|----|
| 2.1 | Carátula del juego original. | 4 |
| 2.2 | Ciudad siendo asediada. | 5 |
| 2.3 | Imágen promocional del juego. | 5 |
| 2.4 | Ejemplo de unidad mecánica del imperio y estética del juego | 6 |
| 2.5 | Infectado gigante | 6 |
| 2.6 | Separation behavior | 8 |
| 2.7 | Alignment behavior | 8 |
| 2.8 | Cohesion behavior | 8 |
| | | |
| 3.1 | MockUp selección. | 12 |
| 3.2 | MockUp pantalla de inicio. | 13 |
| 3.3 | MockUp pantalla de carga. | 14 |
| 3.4 | MockUp escenario juego. | 14 |
| 3.5 | MockUp victoria. | 14 |
| 3.6 | MockUp derrota. | 14 |
| 3.7 | MockUp menú de pausa. | 15 |
| 3.8 | MockUp escenario juego. | 15 |
| 3.9 | Ejemplo escenario con vegetación. | 16 |
| 3.10 | Ejemplo escenario desértico. | 16 |
| | | |
| 4.1 | Logos de Trello y Toggl. | 18 |
| 4.2 | Lista de tareas realizadas en la iteración 0 | 19 |
| 4.3 | Lista de tareas realizadas en la iteración 1 | 20 |
| 4.4 | Lista de tareas realizadas en la iteración 2 | 20 |

Índice de tablas

Índice de Listados

Índice de Acrónimos

A lo largo del documento serán utilizadas una serie de abreviaturas con el fin de hacer más cómoda su lectura. Todos los términos están indicados a continuación:

- **TFG:** Trabajo Final de Grado
- **PC:** Personal Computer
- **RTS:** Real Time Strategy
- **RPG:** Rol Play Game
- **IA:** Inteligencia Artificial
- **NPC:** Non-Player Character
- **AoE:** Age of Empires
- **TaB:** They are Billions
- **ECS:** Entity-Component-System
- **GDD:** Game Design Document

1 Introducción

Este proyecto trata sobre el desarrollo de un simulador de batallas para Personal Computer (PC) desarrollado en *Linux*, escrito en C++ y haciendo uso de librerías escritas en C como TinyPTC [add reference](#).

El simulador, [título](#), se trata de una demo del género Real Time Strategy (RTS) compuesta por un escenario a través del cual tendremos que comandar y luchar junto a nuestro ejército con el fin de abatir al ejército rival. El escenario estará compuesto por estructuras y obstáculos que deberemos sortear para alcanzar nuestro objetivo.

A nivel técnico el proyecto cuenta con el desafío de desarrollar una Inteligencia Artificial (IA) grupal para los Non-Player Character (NPC) basada en el uso de técnicas como los *Steering behaviours* y el *Flocking* con el fin de emular un comportamiento coordinado entre las distintas entidades. El objetivo es crear una versión inicial sencilla mediante la cual poder profundizar en los conceptos en los cuales se basan las mencionadas técnicas con el fin de desarrollar una versión más compleja y específica para nuestro proyecto de forma que se adapte de la mejor forma posible a nuestras necesidades.

Por otro lado, como objetivo adicional fuera de la demo como tal es realizar un *port* a *Windows* con el fin de poder mostrar los resultados en ambos sistemas operativos.

2 Marco teórico

2.1. Estudio de mercado

A lo largo de los años los RTS siempre han contado con una gran aceptación entre los usuarios de PC, un claro ejemplo de esto puede ser el juego *'Age of Empire II: The Age of Kings'*¹ el cual se posicionó como uno de los juegos más vendidos del momento superando el millón de copias. Actualmente el juego cuenta con una remasterización ² la cual ha conseguido vender más de 50.000 copias en *'Steam'*. Datos extraídos de su artículo en Wikipedia.

Si miramos en el panorama nacional actual, podemos encontrar el juego *'They Are Billions'* ³ disponible inicialmente para PC y posteriormente lanzado en *'PlayStation 4'* y *'Xbox One'* debido a su éxito. A día de hoy el juego a conseguido vender solamente en *'Steam'* más de 25.000 unidades consiguiendo una valoración muy positiva por parte de los usuarios de la plataforma como podemos ver en la página del juego de Numantian en dicha tienda.

El género puede aparentar ser cosa del pasado pero vistas las cifras podemos concluir que sigue siendo un reclamo para los jugadores.

2.2. Referentes

A la hora de establecer las mecánicas, el estilo artístico y/o otros aparatados del juego es habitual basarse o tener en cuenta lo que han hecho otras entregas anteriores en su momento. Para el desarrollo de este proyecto han sido de gran utilidad una serie de entregas entre las que cabe destacar dos.

¹Desarrollado por 'Ensemble Studios' y lanzado a finales de 1999.

²Co-desarrollado por Forgotten Empires, Tantalus Media y Wicked Witch. Lanzado en noviembre de 2019.

³Desarrollado por Numantian Games y lanzado en junio de 2019.

2.2.1. Age of Empires II: The Age of Kings

En primer lugar encontramos el juego '*Age of Empires II: The Age of Kings*', en este juego tendremos que gestionar una civilización a elegir entre un amplio abanico de facciones con sus propias unidades y edificaciones.

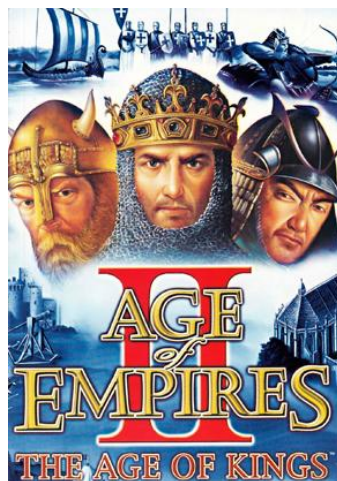


Figura 2.1: Carátula del juego original.

El jugador deberá ser capaz de guiar a las unidades, gestionar las ciudades y conseguir recursos a lo largo del escenario con el fin de derrotar a las demás facciones. Además, se nos presenta una serie de campañas en las cuales manejaremos a grandes personajes de la historia como 'William Wallace' o 'Juana de Arco' entre otros.

El juego hace de referente en varios aspectos entre los que podemos encontrar el apartado visual donde se utiliza una perspectiva isométrica con cámara fija y el uso de una estética clásica propia de las épocas a las que pertenecen las civilizaciones.

Todas las unidades de las que podemos disponer cuentan con una serie de características que las hacen más o menos fuertes en función del objetivo al que atacan, podemos encontrar el ejemplo de las máquinas de asedio las cuales son más fuertes contra estructuras pero más débiles contra infantería, o las unidades a caballo que son fuertes contra arqueros e infantería pero débiles frente a lanceros. Este tipo de mecánicas dotan al juego de una importante componente táctica en el manejo de las unidades que debemos dominar si queremos completar los diferentes niveles de forma satisfactoria 2.2.

2.2.2. The Age of Billions

En segundo lugar podemos encontrar el juego '*They are Billions (TaB)*' el cual reutiliza una serie de características propias del género y como pueden ser la perspectiva



Figura 2.2: Ciudad siendo asediada.

isométrica pero sin dejar de innovar introduciendo nuevas mecánicas y/o formas de plantear la jugabilidad.



Figura 2.3: Imágen promocional del juego.

En '*TaB*' podemos encontrar funciones interesantes como la pausa táctica la cual nos permitirá visualizar detenidamente el estado del mapa sin tener que preocuparnos por no estar atendiendo algunos posibles eventos como ataques a nuestras tropas por parte del enemigo. En juegos anteriores como los '*Age of Empires (AoE)*' es fácil encontrarnos en la situación de tener trabajadores en la ciudad sin hacer tareas un rato y no poder mirar cuales son e ir pensando su ocupación siguiente por estar atrapado en refriegas con otros jugadores, con este tipo de mecánicas estas situaciones se solventan en mayor o menor medida y permiten al jugador tomarse el tiempo que necesite para pensar las acciones que quiere realizar.

Otro aspecto llamativo en el podemos fijarnos es en la aparición de solamente dos

facciones. Por un lado encontramos ‘El Nuevo Imperio’, la facción del jugador, la cual representa una serie de colonias que tendremos que desarrollar a lo largo de los niveles. En el otro lado encontramos las infinitas hordas de zombies que tratarán de exterminar a la raza humana.



Figura 2.4: Ejemplo de unidad mecánica del imperio y estética del juego

A diferencia de en ‘*AoE*’ que todas la facciones poseen las mismas unidades ⁴, en ‘*TaB*’ cada facción dispondrá de unidades únicas con acciones propias. Como podemos ver en la imagen 2.4 las tropas del imperio se basan en el uso de maquinaria y armas de fuego para repeler las hordas enemigas, por otro lado los zombies contarán con diversas características físicas mejoradas conforme sean de mayor “nivel” pudiendo encontrar algunos más rápidos, o más fuertes o más grandes y resistentes al daño. 2.5

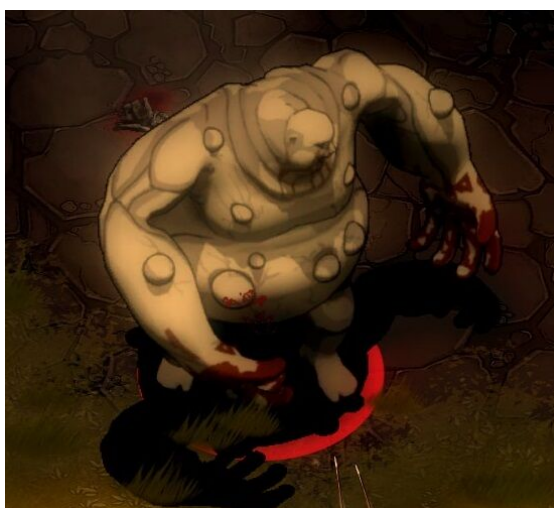


Figura 2.5: Infectado gigante

⁴Algunos valores pueden variar según bonos de facción

Como último punto a destacar de la entrega tenemos la introducción del un modo de juego de supervivencia donde a lo largo del tiempo irán apareciendo oleadas de zombis donde cada vez aparecen más enemigos y más poderosos de forma infinita, haciendo así que la partida termine cuando el jugador deje de aguantar la ofensiva enemiga.

2.3. Técnicas de inteligencia artificial

Como ya se ha mencionado anteriormente en la introducción 1 de este Trabajo Final de Grado (TFG) el grueso del desarrollo y uno de los objetivos más importantes del proyecto recaen en el desarrollo de una IA haciendo uso de las técnicas de '*Flocking*' y '*Steering behaviors*'.

Para ello nos basaremos principalmente en las explicaciones y ejemplos que podemos encontrar en el libro (Millington, 2009, ch. 3) donde de forma extensa y detallada se nos introduce en la teoría relacionada a los algoritmos y la forma en la que estos se estructuran e interactúan entre ellos. Para dar un poco de contexto sobre el tema resumiremos brevemente las ideas que se nos presentan a lo largo del capítulo dedicado a estas técnicas.

Los '*Steering behaviors*' pueden ser entendidos como una serie de algoritmos destinados a guiar la forma en la cual los NPC se desplazan por el escenario y/o interactúan con los distintos elementos que puedan encontrarse en la escena. Siguen una filosofía de crear movimientos complejos a base de una combinación de movimientos y/o acciones simples, un ejemplo común puede ser la acción de perseguir a un objetivo mientras se sortean obstáculos en el proceso.

En este caso no tendríamos una función llamada "*persigue-enemigo-mientras-esquivas()*" y esta encargarse de todo el trabajo, sino que, tendremos el cálculo de la velocidad y dirección necesarias para alcanzar el objetivo, la compropiación para saber si hay algún tipo de obstáculo por el camino y la rectificación de la trayectoria en caso de haberlos cada uno por su lado y es la resultante de todos los pasos la que defina el movimiento final.

Esta forma de estructurar y formar actividades complejas en base a acciones más simples nos permite reutilizar y jugar con los diferentes comportamientos permitiéndonos crear con ellos un amplio espectro de resultados.

En lo referente al '*Flocking*' podemos observar como en esencia es lo mismo que los '*Steering behaviors*' pero añadiendo factores y/o componentes grupales, el origen del modelo lo podemos encontrar en las publicaciones de Reynolds (1986) donde se nos introduce el concepto de "*Boid*" como entidad generica que simula su comportamiento bajo este algoritmo. Además, se nos introducen los tres comportamientos básicos en los que se basa la técnica para generar el movimiento emergente, que son:

La **separación** 2.6 que cada *Boid* mantendrá entre las demás entidades en su vecindad,

con esto evitaremos solapamientos y respetar el espacio y movimiento de las demás entidades.

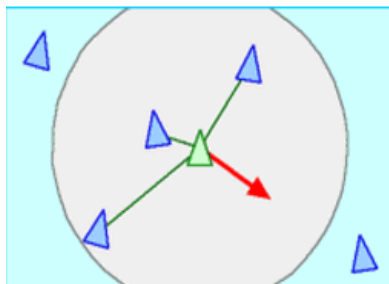


Figura 2.6: Separation behavior

Por otro lado podemos encontrar el **alineamiento** 2.7 de la dirección del movimiento propio con las de las entidades más cercanas, de esta forma conseguimos un movimiento armónico entre los *boids* y produciremos una sensación de coordinación entre ellos.

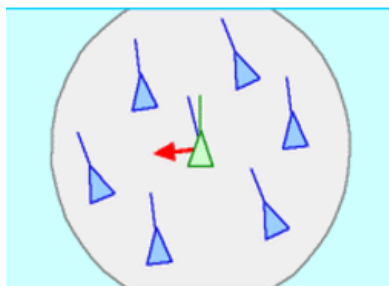


Figura 2.7: Alignment behavior

Por último encontramos la **cohesión** 2.8 la cual se encargará de mantener a las entidades cercanas juntas para crear esa sensación de grupo que buscamos con el algoritmo.

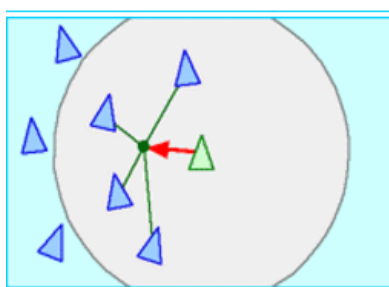


Figura 2.8: Cohesion behavior

Por otro lado, podemos ver en el artículo de ‘Raynolds’ como a lo largo de los años se ha ido modificando y ampliando el algoritmo con el fin de añadir variaciones en el

comportamiento y/o introducir más factores influyentes en la decisión de los *Boids* como puede ser el olor de determinada entidad/es y/o escenario.

Esto sin duda es gracias a la versatilidad que nos proporciona el uso de los '*Steering behaviors*' y jugar con la importancia de las distintas componentes a la hora de hacer la toma de decisiones.

3 Documento de Diseño del Juego (GDD)

3.1. Características

- **Título:** N/A.
- **Plataforma:** PC (Windows y Linux).
- **Género:** Real Time Strategy (RTS).
- **Idioma:** Inglés.
- **Clasificación:** PEGI 7¹

3.2. Descripción general

El juego que se pretende desarrollar consiste en un RTS haciendo hincapie en el apartado de navegación por el mapa y el combate, dejando así la gestión de recursos y el desarrollo de una civilización o facción a un lado debido a las limitaciones de tiempo y personal. A lo largo de los distintos niveles, el jugador deberá hacer frente a distintos desafíos como pueden ser: trasladar todas sus unidades de un punto del mapa a otro mientras se evita la muerte de algún personaje importante y/o eliminar a todas las entidades enemigas del escenario.

3.3. Mecánicas

La finalidad del juego es la de completar los distintos niveles de forma satisfactoria, esto sucederá ya sea cuando eliminemos a todas las unidades enemigas desplegadas a lo largo del nivel o alcancemos el punto indicado. Como herramienta para alcanzar este objetivo dispondremos de un ejercito a nuestro mando el cual deberemos gestionar de forma efectiva para sortear los obstáculos y desafíos propuestos.

El jugador será capaz de seleccionar con el ratón las unidades exactas sobre las que quiere lanzar la acción y seleccionar o deseleccionar todas la unidades a su disposición

¹Web de la asociación <https://pegi.info>

mediante atajos de teclado en cualquier momento. Las acciones disponibles para su uso serán las de desplazar las unidades, hacer que se queden a la espera y realizar un ataque sobre las unidades rivales que se indiquen. Como puede verse en la imagen 3.1

El jugador perderá cuando todas sus unidades mueran.

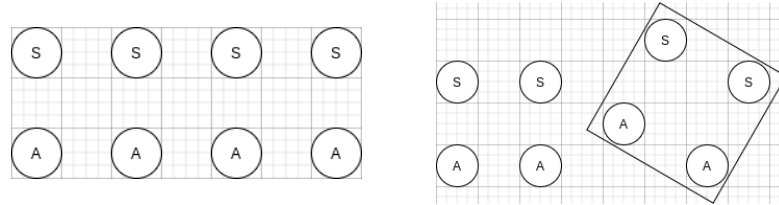


Figura 3.1: MockUp selección.

3.4. Unidades

Entre las unidades podemos encontrar diferentes arquetipos con características propias que nos permitan crear variedad en las posibles soluciones a la hora de superar el nivel.

Los distintos tipos son los siguientes:

- **Soldado:** es la unidad más básica que podemos encontrar en el campo de guerra, esta armado con una espada y posee estadísticas bajas.
- **Arquero:** va equipado con arco y flechas para atacar a distancia a sus rivales, tiene menos resistencia que los soldados por lo que tendremos que protegerlos para asegurar su supervivencia.

3.5. Controles

A la hora de jugar tendremos una serie de teclas asignadas a las acciones que el jugador puede realizar cuando interactúe con el juego. Para enumerarlas dividiremos las acciones en dos grupos, dependiendo de si son para navegar por los menús o si representan acciones durante el *gameplay*.

Las teclas para navegar por los menús son las siguientes:

- **Enter:** mediante esta tecla podremos avanzar por los menús una vez estemos sobre la opción deseada.
- **Retroceso:** mediante esta otra podremos ir hacia atrás por los menús.
- **Escape:** esta nos permitirá salir de la ejecución del programa.

Las asignadas para jugar son las siguientes:

- **Click derecho:** mediante esta tecla del ratón podremos seleccionar la unidad sobre la que queremos trabajar, si mantenemos pulsado y arrastramos por el escenario podremos seleccionar varias unidades cercanas. Si no se selecciona ninguna unidad, se deshará la selección actual.
- **Click izquierdo:** con esta otra podremos seleccionar donde queremos desplazar a las unidades seleccionadas y/o atacar a otras unidades.
- **R:** usaremos esta tecla para deseleccionar todas las unidades desde el teclado.
- **T:** la usaremos para seleccionar a todas las unidades, independientemente de donde se encuentren.

3.6. Pantallas

Al ejecutar el programa la primera pantalla que aparecerá será la de inicio 3.2. Se compone del título del juego, la fecha de lanzamiento, el nombre del desarrollador y un mensaje que nos indica que tenemos que pulsar al tecla *entre* para continuar, esta pantalla se mantendrá hasta que el jugador presione dicha tecla.

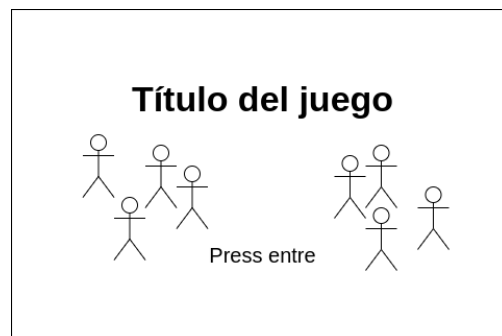


Figura 3.2: MockUp pantalla de inicio.

Una vez pulsado el botón indicado se procederá a cargar el juego mientras se muestra una barra que indica el progreso de cargado 3.3.

A continuación de la carga pasaremos directamente al escenario donde se desarrollará el *gameplay* y nos mantendremos en esta pantalla hasta que termine el juego, ya sea por victoria o derrota del jugador 3.4.

El final de la partida nos trae dos posibles escenarios, la victoria y la derrota. Para cada uno saldrá su respectivo mensaje 3.5 3.6 y pasado un momento se nos mandará automáticamente a la pantalla de inicio 3.2.

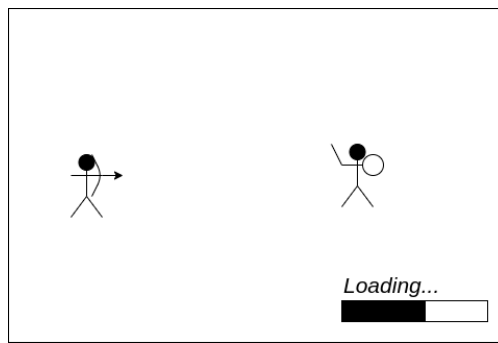


Figura 3.3: MockUp pantalla de carga.

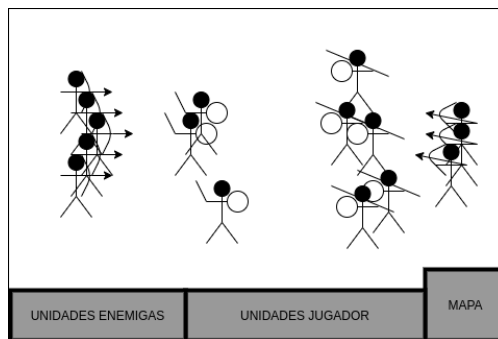


Figura 3.4: MockUp escenario juego.

Como última pantalla con la que el jugador podrá interactuar es la del menú de pausa 3.7 en el cual se le dará la opción de salir o de volver a la partida, mientras esta pantalla este activa la acción en el juego se paralizará hasta que el jugador decida.



Figura 3.5: MockUp victoria.

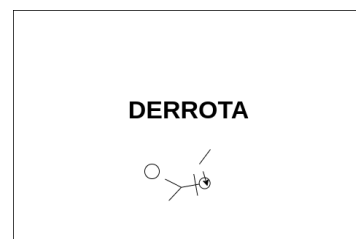


Figura 3.6: MockUp derrota.

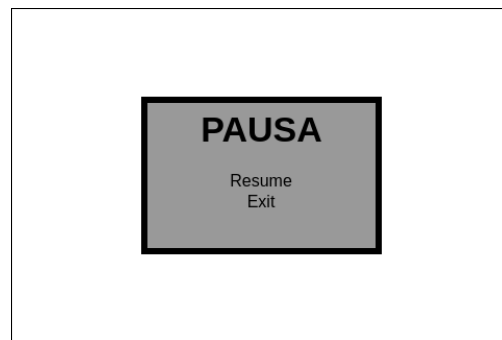


Figura 3.7: MockUp menú de pausa.

3.7. Estados del juego

Una vez mostradas todas las posibles pantallas con las que podrá interactuar el jugador es interesante dibujar un diagrama de flujo que plame sus conexiones y posibilidades con el fin de crear una representación gráfica que sirva como esquema global.

Dicho esquema podemos encontrarlo en la figura 3.8.

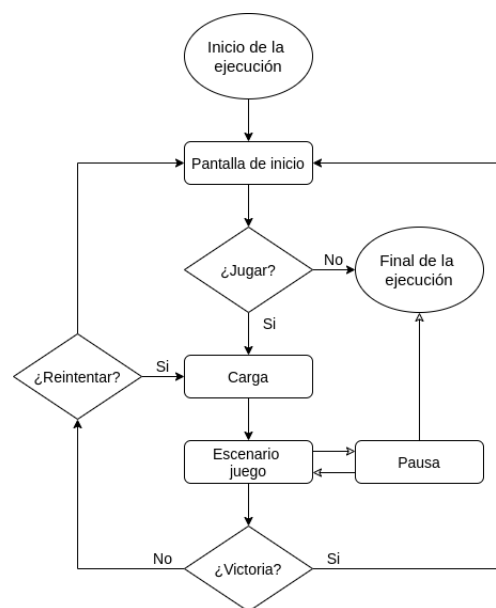


Figura 3.8: MockUp escenario juego.

3.8. Escenarios

A lo largo de los niveles la intención es que nos encontremos con distintos biomas como pueden ser praderas, bosque o desiertos como los que podemos encontrar en AoE II como hemos visto anteriormente en la imagen 2.2 o en las 3.9 y 3.10.



Figura 3.9: Ejemplo escenario con vegetación.



Figura 3.10: Ejemplo escenario desértico.

Además de imitar el estilo artístico el juego se desarrollará empleando una cámara aérea fija manteniendo una perspectiva isométrica que nos permitirá visualizar desde un punto elevado una gran porción del escenario, esto lo haremos con el fin de dar al jugador la posibilidad de conocer lo máximo posible el territorio cercano posible y a la vez le libramos de tener que estar preocuparse de situar la cámara para cada momento. Al mantener una posición fija podemos situar todos los elementos en la escena de forma que no obstaculicen la visión o molesten al jugador.

4 Metodología y fases del producto

4.1. Metodología

La metodología llevada a cabo durante el proyecto es una creación propia basada en características copiadas de metodologías ágiles como el '**Scrum**' y adaptadas a la forma de trabajar propia. La idea es estructurar todo el proyecto en diversas iteraciones y durante estas marcarse objetivos y/o tareas con el fin de añadir funcionalidades completas.

El uso de **iteraciones** para dividir la carga de trabajo y agrupar tareas nos ayudará a conseguir acotar la duración de algunas tareas y marcarnos objetivos a corto/medio plazo. Además de separar por funciones el proyecto, las tareas que conllevan su desarrollo también deberemos analizarlas e intentar reducir su tamaño y carga lo máximo posible, esto permite obtener una sensación de éxito de forma rápida lo cual ayuda a mantener una moral y motivación altas.

Para poner un ejemplo de como elaborar esta separación por subtareas, el objetivo que vamos a usar es “dibujar elementos por pantalla”. Como estamos en un proyecto basado en Entity-Component-System (ECS) rápidamente vemos que vamos a requerir mínimo un componente de dibujado (*RenderCmp*) y un “RenderSystem” el cual se encargará de recoger todos los elementos visuales y hacer las llamadas necesarias para dibujarlos. En caso de no tener una librería de dibujado, otra tarea podría ser el buscar una que sea capaz de realizar el trabajo requerido y/o implementar una propia ¹.

Para ayudar a la planificación del trabajo usaremos la herramienta ‘Trello’ ² la cual nos permitirá crear diversas listas según el ámbito o el estado de desarrollo de las distintas tareas que contendrán, cada tarea se ve representada con una tarjeta la cual puede tener una serie de subtareas asociadas dentro. Esto nos permitirá tener un registro de todas las labores que quedan por hacer en cada iteración y cuales ya están terminadas completamente, además podremos conservar las listas de las tareas realizadas en iteraciones anteriores para futuras revisiones.

Otra herramienta de control que usaremos durante el desarrollo será ‘Toggl’ ³ la cual nos permitirá llevar a cabo un registro de las horas que dedicamos en cada tarea y agrupar tareas en función del campo de estudio o parte del desarrollador al que pertenecen.

¹Esto supondría una serie de tareas en base a la cantidad de funcionalidades que se requieran.

²Página de ‘Trello’: <https://trello.com/es>

³Página de ‘Toggl’: <https://toggl.com>

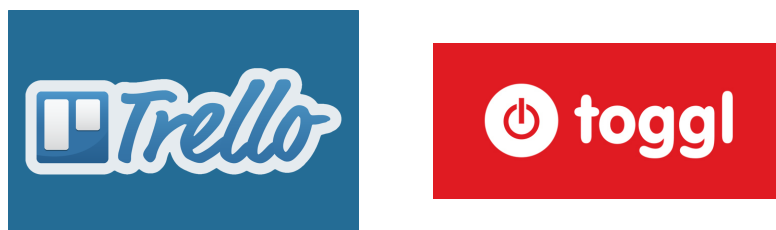


Figura 4.1: Logos de Trello y Toggl.

4.2. Iteraciones

En esta sección se comentarán brevemente los objetivos planteados y las tareas que se han llevado a cabo en cada iteración adjuntando el tablero de ‘Trello’ asociado.

4.2.1. Iteración 0

Esta iteración tenía como proposito terminar de definir la idea para el proyecto, ya que, todavía era un poco difusa la imagen del producto final que se quería desarrollar. Para ir entrando en una dinámica productiva e ir definiendo lo que se quería hacer, comenzamos por iniciar el desarrollo de un prototipo a la vez que preparabamos un poco los materiales relacionados con la memoria, como puede ser la lectura de las directrices y/o la revisión de la plantilla de L^AT_EX 4.2.

Al final de esta iteración el prototipo contaba con una versión básica del bucle principal del juego donde creamos una serie de sistemas, *managers* y entidades además del dibujado y movimiento⁴ de estas.

4.2.2. Iteración 1

A lo largo de esta iteración seguimos añadiendo sistemas al prototipo como puede ser el de *Input* el cual nos permitirá interactuar con el producto, además de herramientas como el mapeado del teclado o el tipo de dato en coma fija en el cual profundizaremos más tarde.

Por otro lado, se comenzó a trabajar en la IA del juego creando los primeros comportamientos y herramientas para cambiar entre ellos. Como veremos más adelante también, aquí caeremos en los primeros errores de concepto a la hora de trabajar con los ‘*Steering behaviors*’.

Por último, finalizamos la introducción en el uso de L^AT_EX, ‘*JabRef*’ y la plantilla comenzando así con la redacción de esta memoria 4.3.

⁴En esta versión nos limitamos a un “Goto” a través de 4 puntos.

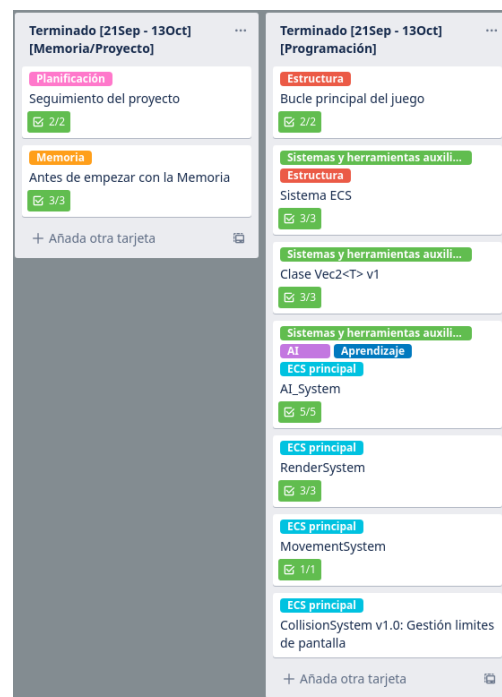


Figura 4.2: Lista de tareas realizadas en la iteración 0

4.2.3. Iteración 2

Durante esta tercera iteración los objetivos eran los de corregir el funcionamiento de los *'Steering behaviors'*, añadir elementos visuales que ayuden al debug de la IA mostrando las componentes del movimiento de las entidades y herramientas para el control del tiempo de ejecución y el tamaño del *'DeltaTime'*.

Como podemos apreciar en la lista 4.4, las tareas de control de la ejecución del programa no fueron realizadas a tiempo, en su lugar se le dió prioridad a mejorar el uso de la coma fija para un código más limpio y crear metodos auxiliares como puede ser el paso de coordenadas continuas a pixeles en pantalla, permitiendo así trabajar a lo largo del programa usando enteros con signo y procesarlos en el sistema de *'Render'*.

En lo referente a la memoria comenzamos la redacción de una versión preliminar del Game Design Document (GDD), Estado del Arte y metodología.



Figura 4.3: Lista de tareas realizadas en la iteración 1

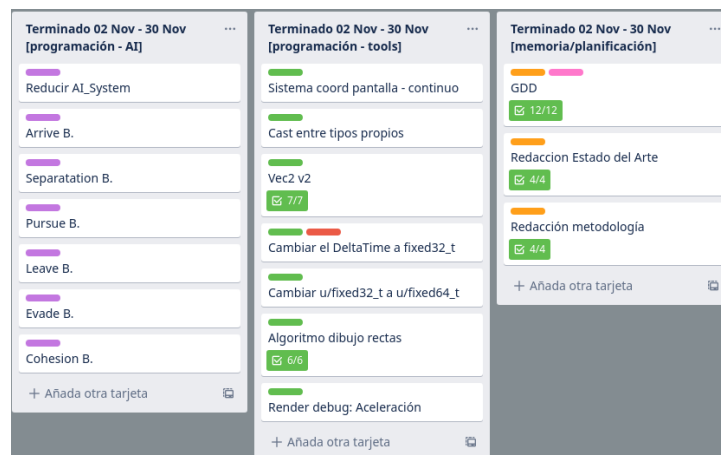


Figura 4.4: Lista de tareas realizadas en la iteración 2

Bibliografía

Millington, I. (2009). *Artificial intelligence for games*. Morgan Kaufmann/Elsevier, Burlington, MA.

Numantian (2019). They are Billions. https://store.steampowered.com/app/644930/They_Are_Billions/.

Reynolds, C. (1986). Boids: Background and update. <http://www.red3d.com/cwr/boids/>.

Wikipedia (2020). Age of Empire II. https://en.wikipedia.org/wiki/Age_of_Empires_II.