Dimension Defender: Desarrollo de un videojuego en 3D

Francisco Molero Hoyos

Resumen—Este proyecto consiste en el desarrollo de un videojuego 3D para poder experimentar los diferentes procesos a seguir durante el desarrollo como pueden ser el diseño, la programación, la creación de modelos, etc... Por lo tanto se ha hecho uso de la menor cantidad posible de material de terceros excpetuando la música que por las limitaciones de tiempo no se ha creado una propia. Para la realización de este proyecto se han usado esencialmente 2 herramientas, Unity como motor gráfico y Blender para la creación de modelos. El estilo de juego escogido para este proyecto ha sido el de un "Tower Defense" en el cual van apareciendo una serie de enemigos durante la partida que intentarán llegar a un punto especifico del mapa mientras el usuario coloca diferentes torres de manera estratégica para evitar que esto ocurra y ganar la partida. Como funcionalidad añadida el juego consta de un editor de partidas donde el usuario puede personalizar tanto el mapa, modificando la disposición de sus elementos como las torres y enemigos que aparecen en él.

Palabras clave—Videojuego, juego, 3D, editor, estratègia, inteligéncia artificial, IA, mapa, resisténcia, Unity, Blender, torre, defensa, ordenador, PC.

Abstract—This Project consists on the creation of a 3D videogame in order to experiment the different processes to be followed during the development, such as design, programming, modeling, etc… Therefor it's been used the smallest possible amount of third party material, except music which due to time limitations couldn't be created. Two tools have been used for the development of this project: Unity as a graphic engine and Blender for modeling. The chosen game style for this project was "Tower Defense", in which a number of enemies show up during the game, trying to reach to a specific point on the map while the user is placing the towers in a strategic way in order to prevent this from happening and win. As an added functionality, the game features a games editor where the user can customize the map, changing the location of its elements, as well as the towers and enemies which will appear.

Index Terms—Videogamme, game, 3D, editor, strategy, artifial inteligence, AI, map, resistence, Unity, Blender, tower, defense, computer, PC.

1 Introducción

ESTE proyecto ha sido motivado por la passión por los videojuegos y las ganas de aprender el proceso a seguir para lograr desarrollar un videojuego independiente partiendo de una idea.

Este Proyecto consiste en el desarrollo de un videojuego de ordenador del tipo "Tower Defense" en 3D, en el que el objetivo del jugador es evitar que los enemigos generados durante la partida lleguen a un determinado lugar en el mapa, para ello el jugador deberá construir torres en puntos estratégicos que atacaran a los enemigos que se encuentren dentro de su alcance hasta acabar con todos.

En este juego el usuario puede editar sus propias partidas decidiendo que tipo de enemigos quiere que aparezcan y en qué momento, que torres quiere usar y como quiere que sea el mapa en el que se vaya a jugar la partida, pudiendo cambiar los diferentes parámetros de torres y enemigos para hacer la partida totalmente personalizada.

Para la realización de este proyecto se ha usado el motor gráfico Unity 3D con lenguaje javascript y Blender para el modelado de los objetos.

Como objetivos del proyecto nos proponemos:

- Familiarizarse con herramientas gratuitas de desarrollo de videojuegos para poder desarrollar todos los aspectos de un videojuego sin ningún tipo de coste y con el resultado esperado.
- Diseñar los diferentes conceptos del juego como el gameplay, la transición de pantallas, la disposición de los diferentes elementos en pantalla, la vista y control de la camára, etc… para poder garantizar una buena experiencia de juego de cara al usuario.
- Diseñar aspectos graficos del juego como pueden ser las diferentes partes de las torres, los enemigos, las partículas usadas en los ataques de las torres, interfaz del usuario y otros elementos que aparezcan durante el juego.
- Hacer una IA, tanto para los enemigos para que sean capaces de moverse por el mapa según la situación encontrando el punto al que quieren llegar de la forma mas rápida posible, como para las torres para que puedan tener presentes que enemigos están dentro de su rango de alcance y atacarles si es necesario.
- Crear un editor con el que el usuario pueda persona-

lizar las torres, los enemigos, el mapa y diferentes parametros de la partida para que pueda crear sus propios retos.
- Desarrollar el juego pensando en el futuro, estructurando el codigo y los diferentes elementos usados para el desarrollo del juego de manera que se puedan hacer futuras ampliaciones de manera sencilla como añadir nuevas torres o enemigos, diferentes mapas o incluso un cambio gráfico total.
- Que el resultado final sea jugable para lograr que el usuario pueda disfrutar del juego de forma intuitiva y sin fallos

2 ESTADO DEL ARTE

Actualmente la industria del videojuego se encuentra en una etapa de gran explotación donde se da cabida desde grandes empresas con equipos de un gran numero de personas hasta equipos pequeños de personas indepen dientes. Es por ello que hoy en dia se pueden encontrar muchos videojuegos de todos los tipos y la competencia es muy dura, también en el mercadod de los videojuegos independientes, los cuales gozan cada vez de mayor importancia en la industria.

En lo referente a los "Tower Denfense", es un género que lleva un gran recorrido en la industria ya que debido a sus características jugables y sencillez lo hacían una gran opción en la época en que los gráficos en 3D aún no habían llegado a la industria. En la actualidad éste es un género menos usado, siendo grandes referentes como Plantas VS Zombies de PopCap Games [11] y Defense Grid de Hidden Entertaiment Path [12] proyectos discretos comparados con juegos de otros generos.

Ningun "Tower Defense" en 3D de relevancia en la industria tiene un editor de partidas para compartilas con amigos como el propuesto en este proyecto.

A la hora de aprender a usar las herramientas existen un gran número de libros de ayuda muy interesantes y actuales que explican en detalle como inciarse en un proyecto de este estilo como pueden ser "Desarrollo de Videojuegos: Un enfoque práctico" [1], donde se hace un repaso a todos los aspectos necesarios a la hora de crear un videojuegos explicando desde historia hasta código. Para aprender específicamente Unity [2] [3] y Blender [4] también se han hecho uso de varios libros aunque sin duda alguna la práctica es lo más importante, por eso el material que más me ha ayudado ha sido realizar cursos [5] o buscar por internet ejemplos concretos i blogs [6].

- *E-mail de contacto: fran.molero.hoyos@gmail.com*
- *Mención realizada: Computación*
- *Trebajo tutorizado por: David Vázquez Bermúdez (CVC)*
- *Curso 2015/16*

3 METODOLOGIA

La metodología usada para el desarrollo de este proyecto es SUM, una adaptación de la metodología ágil y los roles de SCRUM que tiene como objetivos desarrollar videojuegos de calidad con limitación tanto de tiempo como de costo y que permite la mejora continua del proceso incrementando la eficacia y eficiencia de este.

Con SUM los ciclos de vida del proyecto tienen flexibilidad para combinarse con otras metodologías de desarrollo para poder adaptar el proyecto. Está pensado para trabajar en proyectos de corta duración, lo que la hace una gran opción para éste.

3.2 Fases de SUM

Como puede verse en la Figura 1 la metodología SUM esta dividida en 5 fases que se explican a continuación.

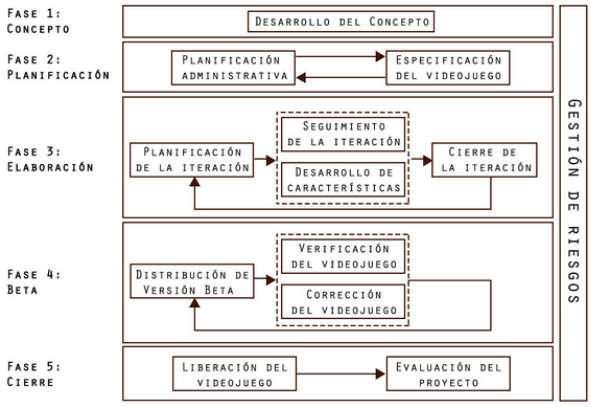


Figura 1. Metodologia SUM

Concepto
El objetivo de esta fase es el de definir conceptos del videojuego como serían los elementos del juego (tipo de gameplay, historia, características …), herramientas a usar, roles, modelo de negocio, etc.. Una vez validadas todas las partes conceptuales por todas las partes involucradas se puede dar por finalizada esta fase.

Planificación
El objetivo de esta fase es el de planificar las siguientes fases del proyecto, para ello es importante realizar un cronograma con las diferentes tareas, sus integrantes presupuesto, desentendencias y priorizar tareas. De esta manera se obtiene una planificación flexible.

Elaboración
El objetivo de esta fase es el de implementar el videojuego, para ello se trabaja de forma iterativa, haciendo que cada entregable o tarea se pueda obtener un ejecutable del juego en las que se pueda evaluar si se han cumplido los objetivos para poder pasar a otro tarea o retocar la misma.

Beta
El objetivo de esta fase es de le evaluar objetivos y ajustar los distintos aspectos del videojuego para balancearlo, para ello se puede distribuir una versión del ejecutable para que lo prueben el numero indicado de personas dependiendo del tamaño del proyecto y poder corregir así los errores o aspectos señalados por usuarios.

Cierre
El objetivo de esta fase es el de entregar la versión final del videojuego al cliente y evaluar el desarrollo, para la evaluación se estudian los problemas y soluciones durante el desarrollo.

Gestión de riesgos
El objetivo de esa fase es el de minimizar la ocurrencia y el impacto de los errores, para ello se deben listar los posibles riesgos junto a su probabilidad y el impacto que puedan tener.

4 Herramientas

Unity 3D – Version 5.3.2f1
Unity [7] [8] es un motor de videojuegos multiplataforma creado por Unity Technologies y lanzado en mayo de 2005. El motor de Unity usa Direct3D para Windows y OpenGL para Mac, Linux, Android… y actualmente es compatible con el software más usado en este tipo de proyectos como puede ser 3ds Max, Maya, Softimage, Photoshop, Blender y un largo etc.

Blender
Blender [9] es un programa de software libre lanzado en 1995 que es muy usado para modelado, iluminación, animación y renderizado de gráficos 3D multiplataforma. Aunque también posee un motor basado en python para la creación de videojuegos su uso en el ámbito profesional no és prácticamente nulo debido a las grandes carencias frente a otros motores como pueden ser Unity o Unreal Engine.

Javascript
Javascript [10] es un lenguaje de programación interpretado, dialecto del stádar ECMAScript. És un lenguaje definido como orientado a objetos muy dinámico usado sobretodo para la programación web del lado de cliente.

5 Planificación

Este proyecto ha sido planificado para realizarse en 300 horas, que es la duración establecida por la asignatura, y a realizar durante 5 meses teniendo en cuenta todas las etapas del SUM y la documentación. Para la realización de la planificación se recogieron todas las tareas/objetivos dando un color diferente a las tareas según del tipo que sea y ordenándolas en un diagrama de Gantt para poder hacer un seguimiento correcto (ver Apéndice A1). Debido a que durante el proyecto podiann surgir varios imprevistos, las fechas tienen bastante holgura, de esta manera nos aseguravamos que el proyecto pudiera ser acabado y en todo momento se avanzaran tareas.

Exceptuando la fase inicial, donde no se podia hacer más que el diseño del videojuego, y la parte final, donde las fechas de informes son preestablecidas, se ha intentado solapar tareas distintas de desarrollo, modelaje y elaboración de informes que no sean dependientes para poder alternar entre diferentes tipos de tareas y lograr así mayor fluidez y aprovechamiento del tiempo.

6 Desarrollo

6.1 Diseño

El diseño de este videojuego se ha dividido en los 2 aspectos en los que se basa el juego, el primero seria el gameplay del juego en si y como se comporta el sistema durante la partida, y el segundo seria la edición de partidas y navegación por los menús del juego.

6.1.1 Diseño de gameplay

El usuario en todo momento tiene una vista en primera persona de la partida por lo que no vera en ningún momento una figura representativa de su personaje en el juego ya que este no existe y simplemente controla una cámara situada a una altura superior a la de las torres que puede desplazarse libremente en horizontal hasta los limites del mapa pero no en vertical ya que el mapa no dispone de diferentes alturas.

El sistema de funcionamiento del juego es simple, al empezar la partida se genera un mapa instanciando los objetos estáticos como las paredes, spawn de los enemigos, objetivos de los enemigos y plataforma de creación de torres. Una vez generado el mapa empieza la partida con una cuenta atrás de 300 segundos que indica al usuario el tiempo límite de generación de enemigos y así tener un mayor control de la partida. El usuario dispondrá de 3 vidas y el juego acabara cuando el usuario pierda todas sus vidas o bien acabe con todos los enemigos.

Durante la partida van apareciendo diferentes enemigos que intentarán llegar al objetivo que se encuentre más cercano quitándole si llega una vida al jugador. Para evitar que esto ocurra y poder ganar la partida el usuario pude colocar torres en puntos estratégicos del mapa según le convenga ya sea para atacar a los enemigos, realentizarlos o ganar puntos que le permitirán poner mas torres.

6.1.2 Diseño de editor y menús

Al iniciar el juego el usuario se encuentra un menú donde puede ir directamente a jugar una de las diferentes partidas ya creadas a modo de tutorial para aprender el funcionamiento del juego o ir a editar su propia partida (ver Figura 2)

Durante el juego siempre habrá una única partida personalizada, en todo momento el usuario puede acceder a ella y modificar los diferentes aspectos para jugarla o guardarla.

El usuario puede modificar 4 grandes aspectos para tener una partida personalizada, el primero es el tipo de enemigos que aparecerán en la partida y sus parámetros como pueden la velocidad a la que se desplaza o la resistencia que tiene o los colores. El segundo aspecto a modificar son las torres, pudiendo escoger el coste de crearlas, el ataque que realizan e incluso su aspecto, cambiando el color y las piezas que la forman. El tercer aspecto a personalizar es el de crear el mapa a su gusto escogiendo los puntos de salida de los enemigos y los diferentes objetivos, por donde pueden para y por donde no los enemigos, y en que lugares se pueden crear torres.Y por último se pueden personalizar aspectos generales de la partida, como pueden ser el dinero total del que dispone el usuario para crear torres, cuantos enemigos quiere que salgan en la partida, de que tipo y en que momento.

Todas estas opciones de personalización hacen que las posibilidades a la hora de crear partidas sean infinitas.

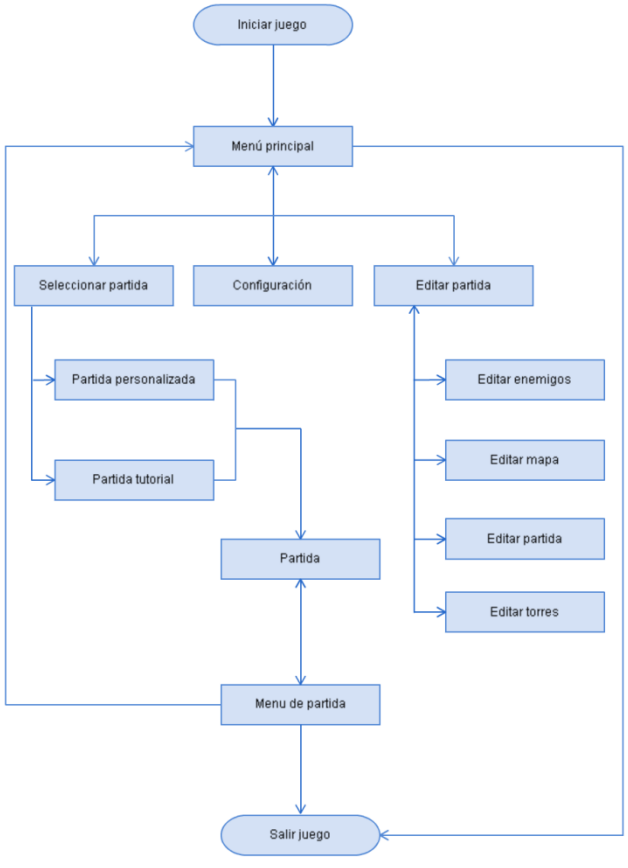


Figura 2. Navegación de pantallas

6.1.3 Diseño de tutorial

Una parte importante a la hora de diseñar un juego es lograr una buena experiéncia para el usuario, para lograr que des del primero momento el usuario entienda la mecanica del juego, aprenda a jugar y vea las diferentes posibilidades que da el juego en cuanto a torres, enemigos y mapas se ha creado una serie de pantallas a modo de tutorial.

El tutorial está compuesto de un total de 12 pantallas divididas en 4 fases distintas (ver Figura 3), en la primera fase el usuario descubrirá las 3 torres que generan directamente daño a los enemigos, la base del juego. En la segunda fase el jugador descubrirá las torres de apoyo, que en lugar de dañar al enemigo sus ataques afectan a otros aspectos de la partida como ganar dinero para poder crear mas torres o realizar a los enemigos. En la tercera fase los enemigos pueden salir de diferentes puntos y el jugador aprenderá la importancia de colocar las torres en un lugar u otro. En la última fase los enemigos pueden salir des de las 4 puntas del mapa y el jugador pondrá en práctica lo aprendido en las anteriores fases para lograr ganar la partida.

Aunque los tutoriales son realmente fáciles el usuario podrá experimentar cierta dificultad al ir avanzando por ellos aumentando el número de enemigos, aumentando las posibles torres a colocar y donde colocarlas o reduciendo el tiempo de reacción.

Figura 3. Selección de niveles

6.2 Diseño grafico

Uno de los objetivos a la hora de desarrollar este juego era implicarse en la mayor parte del desarrollo posible para poder experimentar el trabajo que realizan todos los miembros en un equipo, por ese motivo se ha optado por hacer todo el apartado gráfico des de 0. Al no tener ninguna experiencia prévia en diseño gráfico o modelado 3D se ha optado por realizar unos diseños sencillos con modelos de pocos polígonos (hechos en Blender) que favorecen a la fluidez del juego pudiendo centrar la carga de trabajo en efectos como en las particulas de ataque de las torres.

6.2.1 Diseño General

A la hora de decidir el estilo del apartado gráfico, teniendo en cuenta las limitaciones del desarrollo se optó por generar unos diseños y mapas inspirados en la película TRON [13] (ver Apéndice A2), donde los escenarios son grandes y vacios con el suelo cuadriculado de tono oscuro sin elementos decorativos y los diferentes personajes, objetos o enemigos tienen colores llamativos resaltando sobre todo lo demás de manera que simula los juegos arcade de los 80s. Este diseño se adapta perfectamente al juego ya que permite que se vea bien siendo diseños simples y permite al usuario localizar rápidamente a los enemigos, torres y puntos importantes del mapa gracias al contraste de estos con el escenario.

Para lograr el estilo deseado en los diferentes modelos de forma sencilla y poder cambiarles el color para poder personalizarlos se ha optado por realizar a mano todas las texturas de los diferentes objetos únicamente en color blanco resiguiendo los bordes o vértices en negro, de esta manera al aplicarles una máscara de cualquier color directamente en Unity obtenemos los colores deseados siempre con el borde en negro. Una gran ventaja de esta técnica es la de poder añadir más colores a los objetos del juego en el futuro de forma rápida y sencilla.

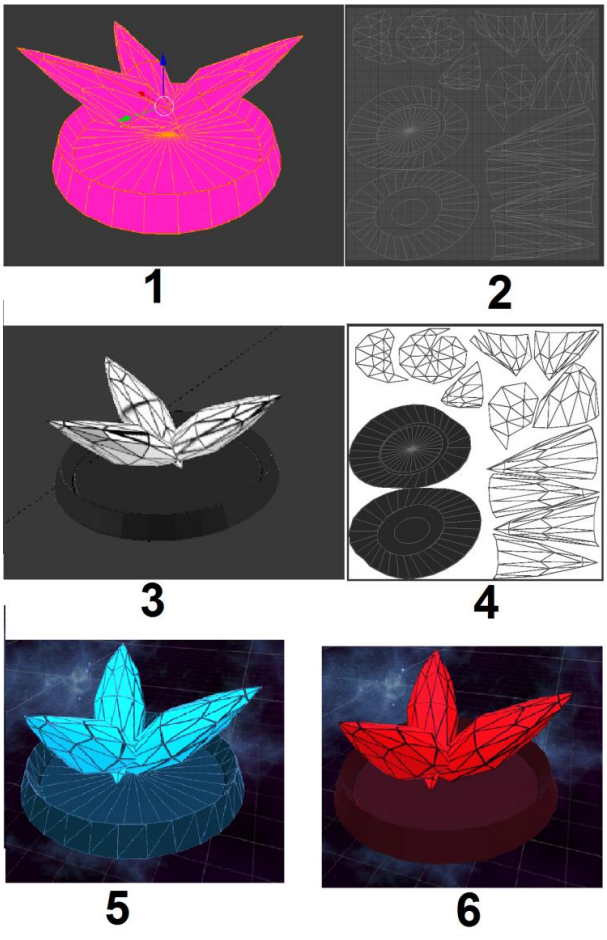


Figura 4. Edición de un modelo

Para la explicación de la creación de las texturas de los modelos nos centraremos en la Figura 4. En la primera imagen podemos ver el modelo creado en Blender, a partir del modelo podemos definir que vértices pueden separarse para poder generar posteriormente el mapa UV que nos hará una representación de la figura en 2D como se puede ver el en punto 2. A partir de este mapa UV, y para llegar al estado de la imagen 3 se ha creado una imagen en blanco y se han ido pitando de gris oscuro la partes que se querrá que cojan menos color y los vértices en negro para dar la sensación de cel shadig en tocos los puntos deseados. Una vez generada la textura en blanco solo hace falta aplicarle una capa de color en Unity para ver los resultados de las imágenes 5 y 6 que serán los colores finales del juego.

6.2.2 Diseño de torres y enemigos

En la versión en que se encuentra el videojuego a la hora de realizar el artículo se encuentran disponibles 5 torres (ver Figura 5). Cada torre esta divida en 3 partes totalmente intercambiables entre ellas y con 6 colores para elegir para hacerlas totalmente personalizadas.

La parte superior de la torre es la que determina que ataque realiza y es la única que afecta al comportamiento del juego siendo las otras 2 solo decorativas.

Figura 5. Diseño de las torres

El diseño de los enemigos el mismo estilo que el de las torres (ver Figura 6), cada uno esta compuesto de tres partes que el usuario pude ir cambiando de color a la hora de personalizarlos aunque diferencia de las torres las diferentes partes no se pueden intercambiar entre vehículos debido a las diferencias de formas que existen entre ellos. Como elemento decorativo cada vehiculo tendrá una o varias llamas para simular el impulso ya que ninguno de estos vehículos toca el suelo.

Figura 6. Diseño de los enemigos

6.2.3 Diseño del HUD

Para el HUD ("Heads-Up Display) se ha usado una distribución típica en este estilo de juegos teniendo los elementos en las partes superior e inferior de la pantalla para dejar la mayor visualización posible de lo que ocurre en el juego y sin perdida de información gracias a unos elementos coloridos y vistosos sin fondo que destacan sobre la partida (ver Figura 7).

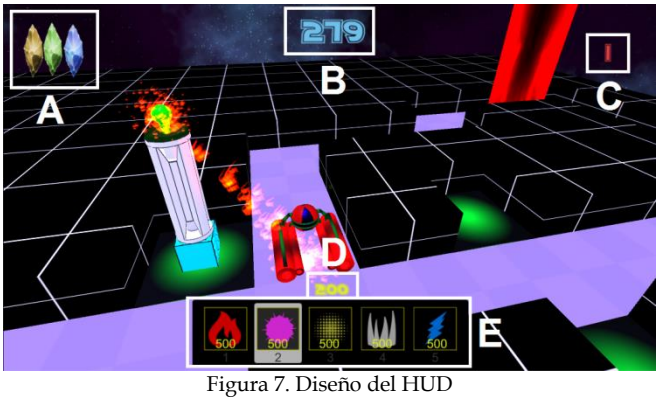


Figura 7. Diseño del HUD

Indicador de vidas (A):
Al inicio de la partida el usuario tendrá un máximo de 3 vidas, a medida que los enemigos lleguen a su objetivo el número de vidas disminuirá y se mostraran menos cristales.

Indicador de tiempo (B):
Empezará siempre a 300 y disminuirá cada segundo, al llegar a 0 el usuario sabrá que ya no se pueden generar más enemigos y la partida acabara al ser destruidos los que haya en ese momento en el terreno.

Indicador de enemigos restantes (C): Empezara con un valor entre 1 y 20 según el número de enemigos que vayan a haber en la partida e irá disminuyendo a medida que se eliminen enemigos, de esta manera el usuario puede hacerse una idea de lo que queda para acabar la partida.

Indicador de presupuesto (D): Indica los puntos que tiene el usuario actualmente y que permitirá crear torres.

Panel de torres (E): Este panel está dividido en 5 partes, una para cada torreta disponible, el icono dentro de cada recuadro muestra el tipo de torre, el numero en amarillo el coste de construir esa torre, el número de debajo de la torre indica el número del teclado a apretar para seleccionar esa torre y el fondo blanco indica la torre seleccionada.

Para la navegación de los menús se ha seguido un estilo en común para todos los elementos dejando los botones de navegación entre escenas (como el de volver hacia atrás) en la esquina inferior derecha, y con el mismo apartado visual siendo todos los botones en amarillo y negro con forma de flecha y el fondo en diferentes tonos de azul para visualizar cuando se pasa por encima igual que en los diferentes elementos que van aparecion, teniendo siempre un acabado de fondo azul sobre elementos amarillos y letras en blanco.

6.2.4 Diseño de partículas

Para crear los ataques de las torres y lograr efectos en pantalla que puedan simular las bolas de energía o el fuego se ha optado por hacer uso del sistema de partículas de Unity. Para crear efectos como el que se puede ver en la Figura 8 Unity tiene un generador de partículas el cual permite definir un gran número de parámetros como pueden ser el tipo de dispersión (en forma de cono, esfera, etc…), el tiempo de vida de las partículas, sus transformaciones a lo largo del tiempo de vida como el color o el tamaño, etc… Al aplicarles a estas partículas una la textura y color deseados obtenemos los diferentes elementos que se usaran durante el juego como el fuego de los enemigos para impulsarse o los diferentes ataques de las torres.

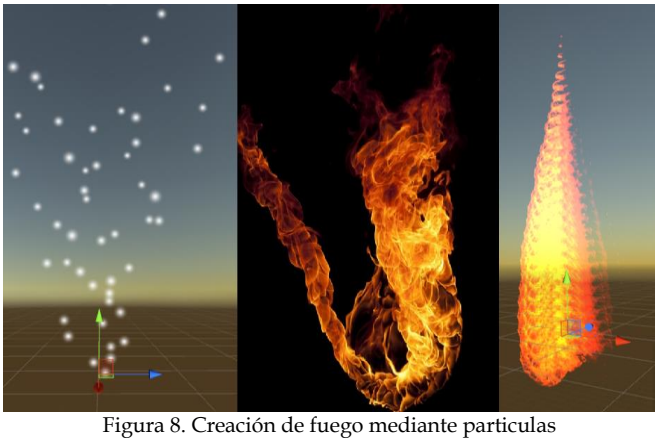


Figura 8. Creación de fuego mediante particulas

6.3 Programación

Este videojuego se ha desarrollado íntegramente con el motor Unity bajo el lenguaje javascript.

El videojuego esta formado por cuatro grandes scripts. El primero se encarga de la navegación de los menús y gestionar como el usuario va personalizando la partida, guardando la información importante en variables globales para poder modificar su partida incluso cambiando de escenas. El segundo script se inicia al comenzar la partida, recibe las variables necesarias para construir el mapa y gestionar la partida y se encargara del control de salida de enemigos, modificar los elementos de la partida según avance y responder a las acciones del usuario. El tercercer y cuarto gran script son los encargados de controlar la actuación de los enemigos y las torres, se explicará su funcionamiento en detalle más adelante.

Acompañando a estos grandes scrips se encuentran muchos otros con pequeñas funcionalidades como controlar la cámara, crear texturas en movimiento, los ataques creados por las torres, las diferentes clases, etc…

IA Enemigos

Cada enemigo tiene un script que genera todas las acciones que realiza la malla que visualiza el usuario para determinar su comportamiento y se comunicará con el script que controla la partida intercambiando la información necesria para llevar a cabo la partida.

Al generarse el enemigo realiza un mapeado de la zona para buscar el camino más corto (explicado más adelante con la Figura 9), una vez sabe el camino a seguir se desplaza por el hasta que llega al final, donde se autodestruye al llegar al colisionar con la meta, o bien hasta que su vida baja hasta 0. Antes de ser destruido el enemigo este hace una llamada al script de la partida para gestionar las variables necesarias.

Para encontrar el camino más corto cada enemigo realiza un mapeado del escenario en una matriz a partir de su posición (casilla roja de la Fig. 1), colocando el punto de partida a 0, los posibles puntos de llegada a -1, las casillas donde no puede pasar a -2, y todas las posibles a 999 (al ser una tablero de 20x20 el número máximo que podría tener una casilla seria de 400). Una vez realizado este paso, a partir de la casilla con el 0 se miran las casillas colindantes verticales y horizontales, si tienen un número mayor se cambian por el siguiente superior al de la casilla que se está mirando y se repite este paso de manera recursiva hasta rellenar la tabla y guardando los diferentes objetivos encontrados. Para acabar se mira el objetivo más cercano (el que tiene un número más bajo) y se van guardando las diferentes casillas con un número inmediatamente menor hasta llegar al 0 para generar el camino del enemigo (Fig. 6).

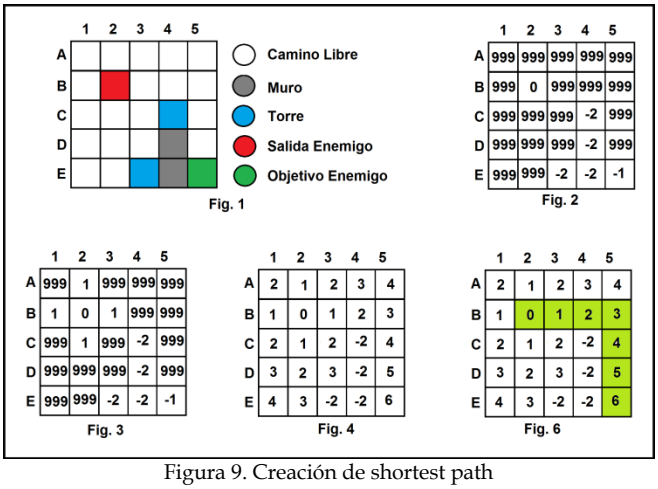


Figura 9. Creación de shortest path

IA Torres

Al generarse una torre la parte superior es la que define el ataque y por lo tanto será esta la que contenga el collider (componente transparente que detecta cuando un objeto entra o está dentro de un área determinada) y el script que realice el ataque.

Una vez generada la torre esta espera a que el collider detecte a un enemigo, mientras el enemigo se encuentre dentro del área la torre generará ataques de una potencia u otra según los parámetros establecidos por la partida. Cada ataque es totalmente diferente pudiendo quemar al enemigo con llamaradas, disparar una bola de energía, desplegar una atmósfera que ralentice al enemigo, etc... Por lo tanto habrá 5 scripts diferentes, uno para cada ataque. Este script se encargara de realizar la correspondiente animación, destruir el generador de partículas al acabar y comunicarse con el enemigo al que ataca para indicarle la vida o velocidad que le baja.

Control HUD y parámetros

El HUD ("Heads-Up Display) es la interfaz que ve en todo momento el usuario durante la partida para mostrarle los parámetros necesarios para jugar como pueden ser las vidas que le quedan, los enemigos restantes, el tiempo para finalizar la partida, etc…

El control del HUD se encuentra en el mismo script de control de partida de esta manera al modificarse una variable que necesita el HUD durante la partida se ve reflejado automáticamente en este.

Editor de partida

Para que el usuario pueda crear sus propias partidas puede moverse por cuatro tipos de pantallas que le permiten modificar diferentes aspectos del juego. Para una mayor sencillez a la hora de que el usuario edite la partida tanto el coste y la potencia de las torres como la fuerza y velocidad de los enemigos se medirán en porcentajes evitando así valores desorbitados y nulos.

Para la creación del mapa el usuario dispone de un una matriz de 20x20 donde crear los diferentes caminos por donde quiere que pasen los enemigos, los posibles puntos de salida y los de llegada. Para que el mismo mapa no sea exavtamente igual cada vez que se juegue cada enemigo elige aleatoriamente des de cual de los puntos de salida quiere partir y cua es el objetivo más cercano. También se pueden escoger en cuantos lugares se pueden poner las torres y donde están. Se puede ver un ejemplo n la Figura 10.

Figura 10. Editor de mapa

Como elementos personalizables globales en la partida el usuario puede determinar el número de enemigos de cada tipo que se crearán y en que segundo de la partida lo harán pero no se puede cambiar el tiempo que dura la partida ya que una partida demasiada corta o larga no favorecería la experiéncia de juego.

7 Futuras implementaciones

Debido a las diferentes limitaciones, sobre todo de tiempo, muchas de las ideas pensadas en un princpio a la hora de desarrollar el juego han sido descartadas para esta versión, algunas poco influyentes y de fácil implementación en un futuro y otras que que requerirían redefinir el videojuego en todos sus aspectos.

Algunas de las implementaciones fáciles podrían ser por elemplo aumentar el número de enemigos y torres a usar, con ataques nuevos tanto de daño directo como de apoyo para dar más variedad al usuario. Otro aspecto a mejorar sería la elección de colores, en esta versión del juego siempre se puede escoger entre 6 colores, pero gracias a la estructura del juego sería fácil no solo aumentar el numero de colores sino también crear una paleta RGB para que el usuario pudiera escoger al gusto el color. El otro aspecto interesante a añadir a este mismo proyecto seria el de poder crear una lista de amigos para poder compartir tus partidas directamente con ellos y crear una tabla de puntuaciones para saber quien ha sido capaz de superarlas, esta implementación aunque sería más costosa que las ateriores también seria posible.

Las ideas iniciales que se plantearon para el gameplay y que no podrían implementarse debido al gran cambio que supondría serian básicamente 2. La primera es la de añadir un modo multijugador, en la que 2 equipos se enfrentan y pueden escoger entre generar torres que le defienda o crear enemigos que ataquen a su oponente, el jugador que lograra ganar más puntos destruyendo enemigos del oponente o consiguiendo que los enemigos generados por el mismo gana la partida. La otra idea sería la de cambiar completamente la estructura de los mediante cambios de nivel y, en lugar de jugar sobre un único plano crear mapas en forma de cubo donde los enemigos podrían teletransportarse al lado opuesto mediante casillas especiales, lo que dificultaría ganar las partidas y daría al juego una mayor posibilidad de plantear estratégias.

8 Conclusiones

A la hora de escribir este articulo y dando el proyecto como finalizado se puede hacer una revisión de los objetivos y afirmar que se han ido cumpliendo. Se ha obtenido un juego muy básico pero completamente completamente jugable y estable en el que el usuario puede editar sus propias partidas y obtener así una experiencia de juego enriquecedora.

Aunque antes de hacer el proyecto se ha dejado un tiempo para el aprendizaje de las herramientas a medida que se iba desarrollando se han aprendido mejores técnicas para mejorar mucho el apartado gráfico que no se han podido llegar a aplicar debido a las limitaciones de tiempo, pero en el apartado de diseño y programación del software los cambios que realizaría si comenzase hoy el proyecto serían mínimos, lo cual indica que el proyecto ha sido planteado de una manera eficiente. El proyecto es totalmente escalable de manera que añadir nuevos elementos como enemigos, torres, pantallas o funcionalidades no debería suponer un gran cambio en el funcionamiento interno del software por lo que el desarrollo del juego puede seguir adelante para ir mejorándolo incluso en el apartado gráfico.

A continuación en las Figuras 11 y 12 se pueden ver un par de capturas del juego en medio de una partida donde se puede apreciar el estado final del juego en mitad de una partida, con enemigos dirigiéndose hacia la meta y las diferentes torres intentando paralos.

Figura 11. Captura de juego

Figura 12. Captura de juego

Agradecimientos

Quisiera mostrar mi agradecimiento a David Vázquez Bremúdez, tutor de este trabajo de fin de grado, por dejarme total libertad a la hora de planificar, organitzar y desarrollar este proyecto y hacer el seguimiento del proyecto, ayudando siempre que ha sido necesario para poder lograr los objetivos.

Bibliografia

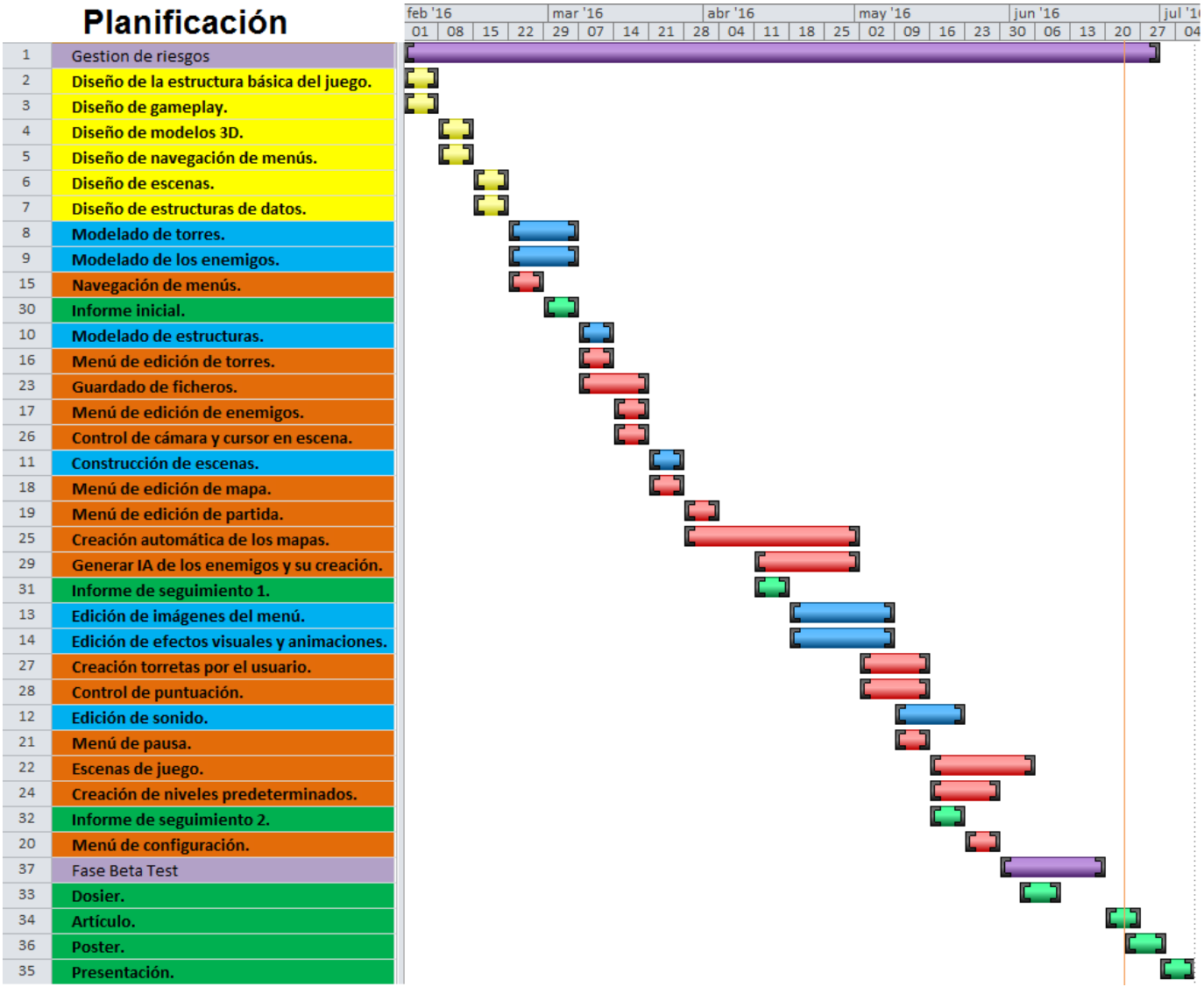
[1] D Vallejo, C González, D Villa, F Jurado, F Moya, “Desarrollo de Videojuegos. Un enfoque práctico”, Ciudad Real: EdLibrix, 2014

[2] Matt Smith, Chico Queiroz, “Unity 5.X Cookbook”, Packt Pub-

lishers 2015
[3] Sue Blackman, “Beginning 3D Game Development with Unity”, Apress 2013
[4] Romain Caudron, Pierre-Armand, “Blender 3D by Example”, Packt Publishers 2015
[5] Universidad Politècnica de València, “Introducción al desarrollo de videojuegos con Unity”, MOOC 2016
[6] Ben Pearson, “Complete Unity Developer”, Blog 2014
[7] Unity Manual, http://docs.unity3d.com/Manual/index.html, Unity Technologies 2016
[8] Unity Scripting API, http://docs.unity3d.com/ScriptReference/index.html, Unity Technologies 2016
[9] Blender Manual, https://www.blender.org/manual, Blender Foundation 2015
[10] Javascript Manual, http://www.w3schools.com/js, Refsnes Data
[11] Pants VS Zombies, PopCap Games 2009
[12] Defense Grid, Hidden Path Entertainment 2008
[13] TRON, Steven Lisberger, 1982

APÉNDICE

A1. DIAGRAMA DE GANTT Y PLANIFICACIÓN



A2. IMÁGENES DE LA PELICULA TRON