Universitat Autònoma
de Barcelona

Guitar Invaders. Juego para Xbox 360

Memòria del projecte

d'Enginyeria Tècnica en

Informàtica de Gestió

realitzat per

Jordi Martín Sánchez

i dirigit per

Xavier Verge Mestre

Escola Universitària d'Informàtica

Sabadell, *Setembre* de *2010*

El sotasignat, ***Xavier Verge Mestre***,

professor de l'Escola d'Enginyeria de la UAB,

CERTIFICA:

Que el treball al que correspon la present memòria
ha estat realitzat sota la seva direcció
per en ***Jordi Martín Sánchez***

I per a que consti firma la present.
Sabadell, ***Setembre*** de ***2010***

Signat: ***Xavier Verge Mestre***

Resumen

El presente proyecto consiste en la creación de un videojuego sencillo para la consola de videojuegos Xbox 360.

Se trata de un “Shoot ‘em up” o juego de naves, en el que deberemos ir esquivando y matando a nuestros enemigos.

Todo esto ha sido posible gracias a las librerías XNA que permiten crear código en C# que la consola Xbox 360 puede leer.

Así pues, disponemos de una pantalla de presentación, un menú principal, una pantalla con tres oleadas distintas de enemigos y un jefe final y un selector de idiomas y otro de dificultades. Todo esto creado en un entorno que mezcla el espacio y la música para darle una estética única y una identidad propia.

Además este proyecto en todo momento ha tenido la intención de innovar en varios aspectos y es por ello que se realiza en dicha plataforma y todo está preparado para ser utilizado mediante una guitarra compatible con el sistema.

A lo largo de este documento se explicará cómo se ha constituido este proyecto y que herramientas han permitido que se lleve a cabo.

Contenido

1 Introducción 1

1.1 Descripción 1

1.2 Objetivos 1

1.3 Estado del arte 2

1.4 Motivaciones personales 3

1.5 Estructura de la memoria 4

2 Análisis y planificación 5

2.1 Introducción 5

2.2 Requisitos funcionales 5

2.2.1 Esenciales 5

2.2.2 Opcionales 6

2.2.3 Extras 7

2.3 Requisitos no funcionales 7

2.4 Método de desarrollo 8

2.5 Herramientas de desarrollo 9

2.5.1 Xbox 360 9

2.5.2 C# 9

2.5.3 XNA Game Studio 3.0 9

2.5.4 Microsoft Visual Studio 2008 9

2.6 Riesgos 10

2.7 Planificación temporal 12

3 Marco Teórico 15

3.1 XNA 15

3.2 XACT 15

3.3 Xbox 360 16

3.4 C# 16

3.5 Microsoft Visual Studio 2008 16

4 Diseño 18

4.1 Diagrama de Casos de Uso 18

4.2 Especificación de Casos de Uso 18

4.3 Arquitectura del Sistema 23
4.3.1 Clases de la aplicación 27
4.4 Interfaz de usuario 35
5 Codificación y pruebas 43
5.1 Estilo de codificación 43
5.2 Pruebas 44
5.3 Errores importantes 45
5.3.1 Cambio de requisitos 45
5.3.2 Ejemplo de error encontrado por un Betatester. 45
5.3.3 Error estético en la detección de colisiones 46
6 Conclusiones 48
6.1 Cumplimiento de objetivos 48
6.2 Desviaciones sobre la planificación 49
6.3 Líneas de ampliación 52
6.4 Consideraciones personales 53
7 Bibliografía 55
8 Anexo I: Manual de usuario 56

Tabla de Figuras

Figura 1 Casos de Uso. 18
Figura 2 Program Class 23
Figura 3 Object Class 24
Figura 4 GameComponent Class 25
Figura 5 Game Class 26
Figura 6 Enemy Class 27
Figura 7 Final Boss Class 28
Figura 8 Shoot Class 29
Figura 9 Spaceship Class 29
Figura 10 Menu Class 30
Figura 11 Death Class 32
Figura 12 Pausa Class 33
Figura 13 Guitar Invaders Class 34
Figura 14: Pantalla de inicio 35
Figura 15: Menú principal 36
Figura 16 Primera oleada 37

Figura 17 Primera oleada difícil 38
Figura 18 Menú Pausa 38
Figura 19 Segunda oleada 39
Figura 20 Tercera oleada 40
Figura 21 Enemigo final 40
Figura 22 Protagonista 41
Figura 23 Nave enemiga 42
Figura 24 Animación enemigo final 42
Figura 25 Guitarra 56

1 Introducción

1.1 Descripción

Guitar Invaders es un videojuego para la consola Xbox360. Desde un principio se quería que fuese un proyecto con posibilidades de comercialización aunque no lo fuera de forma inmediata. Para alcanzar esta meta se partió con una serie de ideas primitivas en que el proyecto debía estar constituido.

La primera idea en que se sustenta este proyecto es en la innovación. Guitar invaders tiene tres características que lo hacen innovador:

- Es un juego amateur de naves para la consola de videojuegos Xbox 360.

- Utiliza como controlador principal una guitarra compatible con el sistema.

- Está traducido a tres idiomas inglés español y catalán.

En segundo lugar, desde el inicio, también se ha tenido en cuenta la voluntad de que fuese un proyecto que siente bases y explore herramientas que otro proyectista pueda necesitar, no tanto para su continuación sino para emprender un proyecto similar y por tanto en este proyecto existe mucho interés en poder mostrar el potencial que nos otorgan las librerías de XNA que a la práctica y, en contra de lo que se suele pensar, son asequibles y no requieren de grandes desembolsos iniciales, ni tampoco de sistemas complejos para ser utilizadas, aunque obviamente requieren un esfuerzo.

El tercer concepto se basa en mostrar el lado creativo de la programación. La programación es algo que puede llegar a ser muy creativa si uno se esfuerza lo suficiente y veremos como una idea se puede plasmar en una aplicación tal y como nosotros la teníamos en mente.

1.2 Objetivos

A diferencia de otros proyectos donde existe una problemática definida que hay que resolver y esto constituye el objetivo del proyecto, en este caso los objetivos se entremezclan la necesidad de realizar un proyecto final de carrera para poder terminar los estudios, el deseo personal de trabajar en un campo que me apasiona y la posibilidad de profundizar en un aspecto concreto de la programación que de otra manera seria difícil poder realizar.

Por lo tanto, como conjunción de todo lo anterior hemos decidido fijar el objetivo de este proyecto en **la creación de un videojuego perfectamente operativo en XBOX360 del nivel de los juegos "amateur" que actualmente existen para esta plataforma**.

Para poder crear este videojuego sin perder de vista los conceptos originales que teníamos en mente, finalmente se decidió por idear un videojuego de naves espaciales controladas por una guitarra compatible con el sistema y todo esto enmarcarlo en una estética que ligase el espacio y la música.

Además este videojuego pretende ser el eje de un futuro proyecto más ambicioso, con más recursos, con más personal, con mucho más tiempo y por ello ha de ser divertido, pues el objetivo de cualquier videojuego ha de ser siempre el divertir y hacer pasar un rato agradable al jugador.

No es objetivo de este proyecto establecer un modelo de negocio que permita comercializarlo, más que nada por cuestiones de tiempo, aunque si es el camino que se desea tomar como ampliación posterior.

1.3 Estado del arte

La situación actual en proyectos de videojuegos "amateurs" es muy difícil de medir pues existen gran cantidad de juegos que no están realizados por grandes compañías pero la mayoría de estos juegos están realizados para PC y cuando nos centramos un poco a mirar más de cerca los videojuegos para consolas nos damos cuenta que realmente es un mundo aún por explorar.

De todas formas no todo el peso de estos videojuegos se ha centrado en los ordenadores puesto que cada vez existen más y más juegos para dispositivos móviles.

Este mercado ha crecido de forma exponencial y parece no conocer límites. Los dispositivos móviles cada vez tienen mejores prestaciones y su capacidad de almacenamiento y cálculo aumentan drásticamente en cada nueva versión lo que se deriva en la posibilidad de hacer juegos cada vez más complejos para este tipo de terminales. Este mercado basado sobre todo en la telefonía móvil es el más amplio en la actualidad y parte con ventaja respecto a la creación de juegos amateurs en cuanto a videoconsolas se refiere.

En este sentido Microsoft ha sido una gran pionera dándoles herramientas a los usuarios para poder dar rienda suelta a su imaginación y poder crear sus propios videojuegos para Xbox 360.

El modelo de negocio es bastante sencillo. Un programador cualquiera utiliza las herramientas de XNA para crear su videojuego, luego se lo envía a Microsoft y estos lo

hacen accesible a todo el mundo a cambio de un 20-30% del total de su valor que se estima normalmente entre los 5$

Este hecho ha provocado que en los últimos meses haya emergido un mercado cada vez mayor de aficionados y pequeñas empresas que intentan hacerse un hueco en tan atractivo pastel y lo cierto es que las previsiones son muy optimistas al respecto y se cree que este nuevo modelo de negocio con una clientela potencial de 20M de personas puede convertirse en un referente en cuanto a interacción de las empresas con los usuarios.

El problema de todo esto es que es algo muy novedoso y que evoluciona a una velocidad tan grande que aún es pronto para sacar conclusiones de la situación actual y la futura aunque se augura que el mercado siga un camino parecido al de los juegos realizados por aficionados en los móviles.

1.4 Motivaciones personales

Este proyecto tiene una gran carga emocional en mi porque se trata de un sueño que siempre quise alcanzar.

Mi vida ha girado en torno a los videojuegos desde que tengo conocimiento y una de mis metas en la vida siempre ha sido conseguir programar mi propio videojuego. Esta idea está tan enraizada dentro de mí que incluso fue determinante a la hora de escoger la carrera en la que quería basar mi futuro. Sabía que este era el camino que debía seguir si quería alcanzar este hito y después de muchos años de estudios por fin he podido cumplir ese objetivo.

Además como muestra de esa gran vocación de cara a los videojuegos puedo decir que hace más de cinco años que soy colaborador de un programa de radio conocido como "El reino champiñón" *http://www.elreino.net/web/principal.php* de la emisora Nova Onda 101.9 FM Albacete dónde se habla de la actualidad de los videojuegos tanto en territorio español como a nivel mundial y que me permite poder asistir a eventos de presentación oficiales de videojuegos de todas las compañías actuales y todas las plataformas.

A todo esto hemos de añadirle que una vez terminados todos mis estudios me gustaría poder trabajar en este sector y quizás este proyecto pueda convertirse en una llave que me abra algunas puertas al respecto. Si el resultado final es el esperado quizás podría utilizar este proyecto como una primera carta de presentación.

Alejándonos un poco de esta motivación principal debida a la atracción por todo lo relacionado con los videojuegos también es necesario comentar que este proyecto se ha convertido en un reto para mí.

Se trata un proyecto innovador, en un entorno desconocido, con un lenguaje de programación que no hemos trabajado directamente durante los estudios y que además está centrado en un tema que no está sobreexplotado y del cual no existe una documentación demasiado extensa. Todo ello conlleva un nivel elevado de riesgo donde no exista seguridad alguna de que este proyecto pueda finalizarse con éxito ni de que disponga de los medios necesarios para poder superar todos los retos que me encuentre por el camino, lo cual le da un toque de emoción al conjunto. Esto, que en principio podría parecer algo demasiado arriesgado, se traduce, a su vez, en una forma de poder autoevaluar mi capacidad de aprendizaje, de abstracción, de adaptación, etc. En cierto sentido este proyecto considero que es una prueba de fuego para poder averiguar todas las capacidades que he adquirido a lo largo de estos años de carrera.

Para terminar y no menos importante, hay otro tema que también ha llamado mi atención y es la capacidad que tienen estos juegos en XNA de convertirse en un modelo de negocio y la gran aceptación que están teniendo. Hacer un juego que sea accesible a 20 millones de clientes potenciales y que además la gran mayoría de éstos sean precisamente de la segmentación del target que estás buscando, tiene una probabilidad muy alta de éxito empresarial, aunque, como ya he avanzado, esto queda para después de finalizado.

1.5 Estructura de la memoria

A continuación se explicará el contenido y estructura del resto de la memoria.

En primer lugar hablaremos del análisis y la planificación de todo el proyecto donde comentaremos los requisitos que ha de cumplir el proyecto, los riegos que conlleva, la metodología de trabajo que se va a utilizar y su planificación temporal.

En el siguiente apartado centraremos el proyecto en un marco teórico y explicaremos un poco al detalle las herramientas utilizadas para su creación.

El cuarto capítulo de esta memoria se centra en el diseño de toda la aplicación y sus interfaces.

Para el quinto capítulo tenemos preparada la metodología seguida en la corrección y detección de posibles errores.

El sexto y último apartado trata de finalizar todo el conjunto explicando las conclusiones obtenidas una vez finalizado el proyecto y las ampliaciones que se le pueden realizar.

Para terminar se adjuntarán un anexo con el manual de usuario.

2 Análisis y planificación

2.1 Introducción

Este proyecto se ha desarrollado siguiendo un método iterativo la cuál cosa hace complicado listar todos los requisitos pues estos se cambiaban y se modificaban a cada iteración.

Los siguientes apartados tratan estos requisitos desde el punto de vista retrospectivo, es decir, los requisitos tal y como se ven una vez pasadas todas las iteraciones y teniendo en cuenta el orden y la prioridad que se les había otorgado en cada momento.

2.2 Requisitos funcionales

En este apartado hablaremos de los requisitos funcionales que debe cumplir este proyecto. Los hemos dividido en tres categorías según su nivel de importancia.

Al primer grupo lo vamos a denominar esenciales pues son los que deben cumplirse para considerar que el proyecto ha sido un éxito. Los siguientes los llamaremos opcionales pues estos son importantes pero no estrictamente necesarios y por último tenemos los extras que son ideas a tener en cuenta pero que solo se realizarán si el proyecto avanza a buen ritmo y además se contempla una mejora sustancial al incluirlos.

2.2.1 Esenciales

- **Interpretar los movimientos de la guitarra**
  El software ha de ser capaz de funcionar íntegramente con una guitarra compatible con la consola Xbox 360 y por lo tanto ha de poder controlar y utilizar los datos que esta envíe al sistema.

- **Creación de la nave**
  En el juego se ha de crear y mostrar una nave que sea capaz de moverse con la guitarra y también ha de poder hacer las acciones de disparar proyectiles de diferentes colores y destruirse caso que sea alcanzada por un enemigo.

- **Creación de los enemigos**
  Los enemigos deben ser capaces de moverse por la pantalla y matar a la nave en caso de colisionar con ella y también han de morirse al ser impactados por proyectiles de su mismo color. Además estos enemigos deben de poder aparecer tanto de forma predeterminada como aleatoria para darle una mayor gracia al juego.

- **Mostrar la puntuación**
  Mientras se juega la puntuación ha de poder ser visible.

- **Empezar y acabar partida**
  Se deben crear unos eventos que muestren al jugador cuando ha ganado o perdido la partida.

- **Incluir efectos de sonido**
  Es muy importante que durante la ejecución podamos escuchar una música de fondo y varios efectos de sonido al darse ciertas circunstancias.

2.2.2 Opcionales

- **Hacer un menú con un selector de idiomas**
  Este requisito trata de crear un pequeño menú dónde poder escoger el idioma que deseamos entre inglés, castellano y catalán. Este requisito se considera opcional porque para lograr nuestro objetivo de innovación debido al ser un juego de consola en catalán se puede hacer directamente en catalán pero de cara al futuro esto podría recortar enormemente el público al que estaría destinado el videojuego.

- **Crear un enemigo final**
  El objetivo principal de este proyecto es hacer una pantalla jugable y esta no necesariamente debe tener un enemigo final aunque de cara al público el juego gana muchos puntos si se pone un enemigo final que cueste un poco de destruir y que nos suponga un pequeño reto.

- **Añadir pausa**
  Hoy en día cualquier juego comercial dispone de algún mecanismo para parar el tiempo en caso de que el jugador deba ausentarse unos momentos. Por eso

este requisito que consiste en dar esta opción de detener el tiempo al jugador es una cosa importante aunque no es esencial.

2.2.3 Extras

- **Establecer tres dificultades distintas**
  Consistiría en crear una opción del menú que permitiera seleccionar la dificultad en la que queremos jugar y además hacer que la pantalla de juego reflejase de forma fiel estas diferencias entre las dificultades posibles. Este requisito conseguiría aumentar de forma considerable la capacidad del juego de gustar a personas con diferentes habilidades.

- **Crear todos los objetos gráficos**
  Aunque el objetivo de este proyecto no es el tener que dibujar ni crear efectos visuales y por tanto se podría optar por buscarlos en alguna librería libre, si le daría un toque mucho más personalizado y robusto a todo el juego si se creasen todas las imágenes, logos, animaciones etc.

- **Introducir un modo cooperativo**
  Otra ampliación importante que se podría realizar se trata de crear un modo de juego para dos jugadores simultáneos que lucharan conjuntamente con dos naves distintas para ganar la partida.

2.3 Requisitos no funcionales

- **Facilitar el uso de la guitarra**
  El juego debe ser realizado de tal manera que sea coherente con los controles de la guitarra y fácilmente entendible por el usuario final.

- **Se debe codificar según las especificaciones de la consola**
  El juego debe de ser capaz de jugarse en una Xbox 360

- **Debe ser divertido**
  Este requisito es difícilmente mesurable pero un videojuego debe tener como máxima prioridad el ser capaz de divertir y entretener es por ello que resulta indispensable añadirlo a la lista.

2.4 Método de desarrollo

Este proyecto se ha desarrollado siguiendo las pautas de un modelo iterativo.

En este caso se ha hecho encuentros semanales entre el desarrollador y el director de proyecto pero además de esto también se ha contado con diferentes sesiones dónde un grupo de personas han sido capaces de probar el sistema y dar su opinión durante todas las etapas del desarrollo de la aplicación.

El encuentro semanal con el director de proyecto ha servido para poder analizar todo el trabajo realizado a lo largo de una semana y comprobar que se cumplían los requisitos en los plazos estimados. De esta forma el proyecto se ha podido controlar minuciosamente y se ha podido reaccionar rápidamente ante los problemas o cambios de requerimientos que han ido surgiendo conforme la aplicación iba creciendo.

El grupo de personas que han podido probar la aplicación durante su desarrollo, a las cuales nos referiremos como Betatesters a partir de este momento, han tenido un impacto muy grande sobre el proyecto pues han aportado ideas y perspectivas que quizás no hubiesen sido tomadas en cuenta y también han servido de pauta para saber si el proyecto marchaba por buen camino y cumplía sus objetivos.

En realidad esta práctica de utilizar Betatesters pasó de ser una simple anécdota a convertirse en un motor principal del proyecto. Las aportaciones realizadas por estas personas tomaron una importancia trascendental en el desarrollo de todas las facetas.

Por una parte han servido para realizar pruebas y encontrar fallos de todo tipo en la aplicación. Por otra parte han sido fundamentales en la detección, el control y la gestión de riesgos de todo el proyecto y además también han sido claves en el refinamiento de los requerimientos.

Las aportaciones por estas personas han sido de valor incalculable para lograr los objetivos propuestos.

En cuanto al hecho de que se haya estimado oportuno utilizar este método de desarrollo se debe en gran parte al conocimiento de que este proyecto tiene riesgos muy altos y que trae consigo una dificultad muy elevada y por tanto para asegurar un exitoso desenlace debía controlarse muy bien su evolución durante todas sus etapas.

2.5 Herramientas de desarrollo

En este apartado se pretende explicar el motivo por el cual se han escogido estas herramientas para el desarrollo del proyecto y no otras. En este caso la mayor decisión fue la plataforma en la que íbamos a trabajar puesto que eso ha delimitado mucho las herramientas que podíamos utilizar y por tanto en la mayoría de casos no hemos realizado nosotros la selección sino que nos ha venido impuesta.

2.5.1 Xbox 360

Xbox 360 es la consola de videojuegos de Microsoft que soporta nuestro videojuego. Esta consola tiene unas grandes prestaciones que nos ayudaran a que el juego sea estable y además los requisitos para hacerlo funcionar son mínimos lo cual le da una gran ventaja respecto al resto de opciones. Solamente necesitamos una consola Xbox 360 con una licencia de XNA y una conexión a Xbox Live que es el servicio online de Microsoft.

2.5.2 C#

C# es el lenguaje de programación que se ha utilizado para desarrollar todo el videojuego. El motivo principal por el que se ha utilizado este lenguaje es porque las librerías XNA están en este lenguaje y es el que comprende la consola Xbox 360.

2.5.3 XNA Game Studio 3.0

XNA se ha escogido porque es la herramienta principal que hace que todo este proyecto sea posible. Se trata de un conjunto de librerías en C# que se añaden al Visual Studio para poder crear videojuegos. Esta herramienta puede utilizarse para crear aplicaciones en Windows, en Zune y también en Xbox 360.

2.5.4 Microsoft Visual Studio 2008

Microsoft Visual Studio 2008 es la plataforma donde se ha desarrollado toda la aplicación en C#. Se ha escogido este y no otro pues es el mejor para trabajar con dicho lenguaje y además es el único que admite la inclusión de las librerías XNA Game Studio que son esenciales para poder comunicarnos con la consola.

2.6 Riesgos

Este proyecto destaca, entre otras cosas, por la cantidad de riesgos que tiene principalmente debidos a su carácter innovador y el desconocimiento por parte del programador de la inmensa mayoría de aspectos que deben ser desarrollados.

La explicación se debe a que el lenguaje es desconocido, el soporte en el que se trabaja se desconoce, nunca se ha realizado nada muy parecido en los estudios y se requiere un alto nivel de aprendizaje durante el desarrollo para poder superar los obstáculos tanto de programación como de diseño conceptual y gráfico.

- **Planificación temporal optimista**
  Es muy difícil prever cuanto vamos a tardar en realizar cada una de las tareas pues no hay manera alguna de tener un punto de referencia que se asemeje a lo propuesto en este proyecto. Además es de esperar que surjan problemas durante su realización y el proyecto tarde más en ser completado de lo esperado. En este caso el proyecto podría verse afectado a recortes que perjudicarían la calidad final.

- **Imposibilidad de realizar alguna tarea**
  Un proyecto de este tipo a nivel profesional se realiza con cientos de personas en diferentes áreas de especialización y se tarda años en realizarse. En este caso se pretende hacer un proyecto más acotado pero siguiendo los estándares de la industria para parecer lo más profesional posible y eso nos induce a que muy probablemente topemos con problemas de programación, de diseño o de planteo que estén más allá de nuestras posibilidades.
  Este riesgo es muy crítico y difícil de superar, la única manera de poder superarlo es llevar un control muy estricto sobre el avance del proyecto y ser capaces de reaccionar y encontrar alternativas ante las imposibilidades que nos vayan surgiendo.

- **Fallo de Hardware en la Xbox 360**
  La consola de Microsoft es muy potente y tiene muchas ventajas pero también es conocida por estropearse con facilidad. Por lo visto tiene un pequeño problema de fábrica en relación a su refrigeración y esto provoca que en algunos casos la consola se sobrecaliente y deje de funcionar. Para solucionar esta situación podríamos utilizar la garantía de 3 años que tienen estas máquinas para este problema pero eso significaría tener que enviarla a reparar a Alemania lo cual demoraría en gran medida el avance del proyecto.

- **Equipo de desarrollo demasiado reducido**
  En este proyecto se espera que una sola persona sea capaz de crear un proyecto que tiene vertientes de carácter muy diferentes. Muchas de estas vertientes tienen precedencias entre sí haciendo que su realización en un cierto orden sea estrictamente necesario y esto nos lleva al problema raíz que se plantea aquí y es que una persona puede tener muy buenas cualidades en un tipo de actividad productiva pero no necesariamente debe ser un experto en el resto. Si se diera este suceso bastante probable, el proyecto se vería resentido en cuanto a su estimación temporal y la calidad de algunos aspectos podría no satisfacer los objetivos planteados.

- **Herramientas de desarrollo inadecuadas**
  Es complicado saber que herramientas vas a necesitar para realizar un proyecto que no se asemeja a nada que hayas realizado antes y esto podría ocurrirnos en este caso. Esto repercutiría principalmente en los costes planteados y además podría suponer un descenso en la calidad del producto final. De todas maneras este riesgo se ha tenido muy presente desde el inicio del proyecto y para evitar que pueda ocurrir se ha decidido estudiar muy bien todo el proceso productivo que vamos a llevar a cabo y de esta manera asegurarnos que nuestro planteamiento inicial sea correcto y tenga en cuenta todas las posibilidades.

- **Calidad del producto final insuficiente**
  Este proyecto es, probablemente, la base de lo que un día acabará vendiéndose en el mercado y por tanto la satisfacción de un producto acabado y de una buena calidad es algo primordial que debemos conseguir. Si la calidad fuese insuficiente las ventas y por tanto nuestra estimación de costes/beneficios podrían verse muy afectadas y por ello es un apartado que siempre debemos tener presente. Para suplir este posible problema se han dedicado muchas horas a que personas ajenas totalmente al proyecto puedan probarlo y dar su opinión para conseguir mejorar y hacer el producto más atractivo.

- **Incumplimiento de alguna ley**
  Uno de los riesgos más importantes cuanto tratamos con software es el incumplimiento de alguna ley pues existen infinidad de leyes que tratan multitud de aspectos sobre la creación de software, el tratado de datos y otras actividades relacionadas pero este proyecto tal y como está planteado intenta desde sus inicios no caer en ningún punto que pueda verse afectado por la ley. Es debido a esto que se ha decidido realizar el apartado gráfico desde cero para no caer en problemas de propiedad intelectual y en cuanto a la música se ha

optado por buscarla en lugares donde tienen licencia libre. De todas formas de cara a la futura aplicación que se tiene pensado hacer una vez termine este proyecto, uno de los máximos objetivos que habrá será realizar nuestra propia música para poder darle un toque más personal al juego y no aprovechar el trabajo de otras personas que han tenido la amabilidad de dejar sus creaciones libres para la comunidad.

2.7 Planificación temporal

Este proyecto en un principio fue planteado como un proyecto que seguía un método tradicional o método de cascada donde las fases estaban muy bien definidas y se seguía un orden preestablecido de trabajo.

A nivel teórico parecía que todo podía encajar a la perfección pero debido a la naturaleza de este proyecto fácilmente se llegó a la conclusión de que ese modelo no se adecuaba a las necesidades.

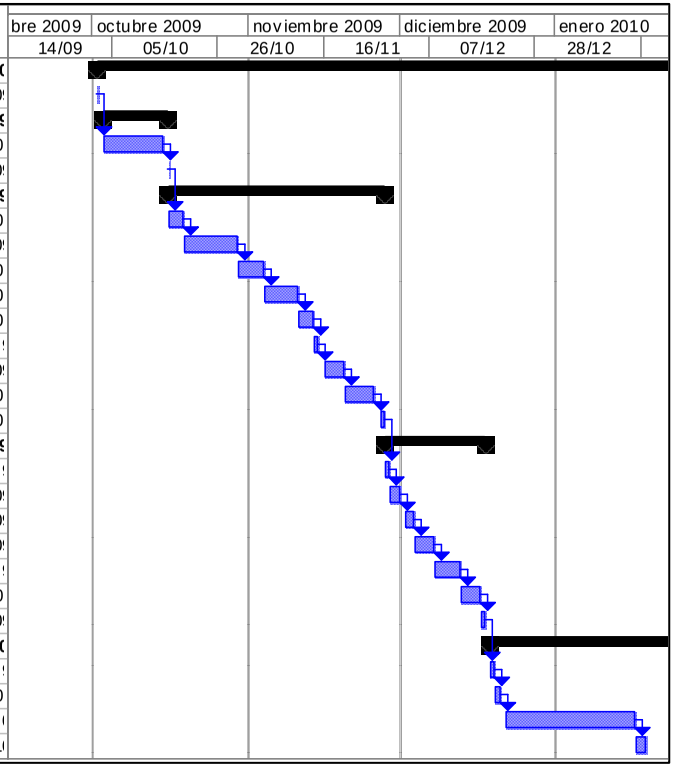
Por este motivo se volvió a plantear todo el proyecto y se enfocó utilizando un modelo iterativo.

El esquema que veremos a continuación pertenece a una visión global de este modelo iterativo y sus tiempos previstos antes de realizarlos. Esto ha decidido hacerse así porque se marcaron unos objetivos desde el comienzo y en base a su dificultad y a los conocimientos que se tenía en cada iteración se decidía que era lo próximo que debía añadirse al proyecto y en cada iteración se revisaba y mejoraba todo lo anterior.

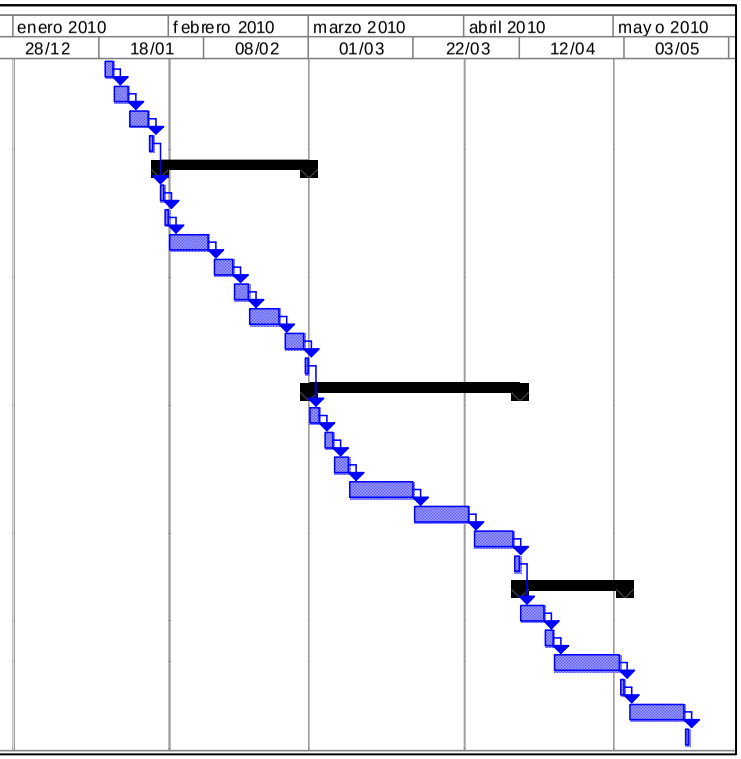
El proyecto inicialmente se planteó con inicio el día 1 de Octubre de 2009 y finalización el viernes 14 de Mayo de 2010. Pero como explicaremos más adelante esta previsión finalmente no se pudo cumplir y se tuvo que reprogramar gran parte del proyecto.

A continuación tenemos el diagrama de Gantt que representa todo nuestro proyecto dividido en las 6 iteraciones más importantes que se han dado.

Id	Nombre de tarea	Duración	Comienzo	Fin
1	**Guitar Invaders**	**161 días**	**jue 01/10/09**	**sáb 15/05/10**
2	Inicio del proyecto: asignación y matriculación	2 horas	jue 01/10/09	jue 01/10/09
3	**Planificación**	**9 días**	**vie 02/10/09**	**jue 15/10/09**
4	Estudio de viabilidad	30 horas	vie 02/10/09	mié 14/10/09
5	Aprobación del Estudio de Viabilidad (Punto d	1 hora	jue 15/10/09	jue 15/10/09
6	**Primera Iteración**	**31 días**	**jue 15/10/09**	**vie 27/11/09**
7	Estudio de los requisitos	8 horas	jue 15/10/09	dom 18/10/09
8	Diseño del sistema	24 horas	dom 18/10/09	jue 29/10/09
9	Creación y control de la nave	12 horas	jue 29/10/09	mar 03/11/09
10	Creación de la primera oleada de enemigos	16 horas	mar 03/11/09	mar 10/11/09
11	Implementar los disparos	6 horas	mar 10/11/09	vie 13/11/09
12	Detectar colisiones	4 horas	vie 13/11/09	sáb 14/11/09
13	Diseño gráfico	10 horas	dom 15/11/09	jue 19/11/09
14	Test y pruebas	15 horas	jue 19/11/09	mié 25/11/09
15	Entrevista con el director de proyecto	3 horas	jue 26/11/09	vie 27/11/09
16	**Segunda Iteración**	**14 días**	**vie 27/11/09**	**jue 17/12/09**
17	Revisión de los requisitos	3 horas	vie 27/11/09	sáb 28/11/09
18	Cambios de diseño	6 horas	sáb 28/11/09	lun 30/11/09
19	Hacer aleatorios los primeros enemigos	5 horas	mar 01/12/09	jue 03/12/09
20	Crear la segunda oleada de enemigos	10 horas	jue 03/12/09	lun 07/12/09
21	Controlar los estados del juego	10 horas	lun 07/12/09	sáb 12/12/09
22	Test y pruebas	10 horas	sáb 12/12/09	mié 16/12/09
23	Entrevista con el director de proyecto	3 horas	mié 16/12/09	jue 17/12/09
24	**Tercera Iteración**	**29 días**	**vie 18/12/09**	**jue 28/01/10**
25	Revisión de los requisitos	3 horas	vie 18/12/09	sáb 19/12/09
26	Cambios de diseño	5 horas	sáb 19/12/09	dom 20/12/09
27	Crear el Menú principal	60 horas	lun 21/12/09	sáb 16/01/10
28	Mejorar el sistema de colisiones	5 horas	sáb 16/01/10	lun 18/01/10



Id	Nombre de tarea	Duración	Comienzo	Fin
29	Ajustar las oleadas de enemigos	4 horas	lun 18/01/1	mié 20/01/1
30	Añadir una tercera oleada de enemigos	6 horas	mié 20/01/1	sáb 23/01/1
31	Test y pruebas	10 horas	sáb 23/01/1	mié 27/01/1
32	Entrevista con el director de proyecto	3 horas	mié 27/01/1	jue 28/01/1
33	**Cuarta Iteración**	**20 días**	**vie 29/01/1**	**dom 28/02/1**
34	Revision de los requisitos	3 horas	vie 29/01/1	sáb 30/01/1
35	Cambios de diseño	3 horas	sáb 30/01/1	dom 31/01/1
36	Incluir las opciones de cambiar el idioma	20 horas	dom 31/01/1	lun 08/02/1
37	Insertar efectos de sonido	10 horas	mar 09/02/1	sáb 13/02/1
38	Rediseñar el sistema de control del tiempo	7 horas	sáb 13/02/1	mar 16/02/1
39	Añadir la pausa en el juego	15 horas	mar 16/02/1	lun 22/02/1
40	Test y pruebas	10 horas	mar 23/02/1	sáb 27/02/1
41	Entrevista con el director de proyecto	3 horas	sáb 27/02/1	dom 28/02/1
42	**Quinta Iteración**	**30 días**	**dom 28/02/1**	**dom 11/04/1**
43	Revisión de los requisitos	5 horas	dom 28/02/1	mar 02/03/1
44	Cambios de diseño	5 horas	mié 03/03/1	vie 05/03/1
45	Crear el enemigo final	8 horas	vie 05/03/1	lun 08/03/1
46	Incluir 3 dificultades distintas	30 horas	lun 08/03/1	dom 21/03/1
47	Mejorar el apartado gráfico general	25 horas	dom 21/03/1	jue 01/04/1
48	Test y pruebas	20 horas	vie 02/04/1	sáb 10/04/1
49	Entrevista con el director de proyecto	3 horas	sáb 10/04/1	dom 11/04/1
50	**Sexta iteración**	**15 días**	**dom 11/04/1**	**dom 02/05/1**
51	Ajustar las dificultades	10 horas	dom 11/04/1	vie 16/04/1
52	Añadir animaciones	5 horas	vie 16/04/1	dom 18/04/1
53	Test y pruebas	30 horas	dom 18/04/1	sáb 01/05/1
54	Entrevista con el director de proyecto	4 horas	sáb 01/05/1	dom 02/05/1
55	Generación de documentos	25 horas	lun 03/05/1	vie 14/05/1
56	Entrega del proyecto	2 horas	vie 14/05/1	sáb 15/05/1



3 Marco Teórico

En este apartado se pretende dar unas nociones sobre las herramientas empleadas y situar un poco el proyecto en su contexto.

3.1 XNA

XNA son un conjunto de librerías que nos permiten crear videojuegos de tal manera que nuestra videoconsola sea capaz de ejecutar.

El concepto básico que debemos saber sobre XNA es que nos crea unas bases sobre las que vamos a poder trabajar, es decir, estas librerías nos permiten generar código que la consola es capaz de interpretar y controlar con ella los dispositivos conectados, la tarjeta gráfica, la tarjeta de sonido etc. Básicamente nos dan una arquitectura que hace de intérprete entre nuestro código en C# y el hardware de la consola.

3.2 XACT

XACT es una herramienta que está incluida dentro de las herramientas de XNA pero que tiene valor por sí misma.

Se trata de un editor de música que es capaz de crear proyectos con diferentes archivos musicales para luego poder ser tratados como canciones o efectos.

Además tiene características que permiten modificar el volumen de cada canción, la capacidad de incluirles un bucle o la probabilidad de que suenen.

Este programa está muy bien integrado con el resto del XNA y permite que se gestione todo el sonido de una forma muy lógica y cómoda dando una gran libertad en todo momento y permitiendo que se pueda controlar en todo momento con exactitud.

De todas formas XACT es un módulo independiente y no necesariamente debe utilizarse para trabajar en proyectos de XNA. Tal y como está diseñado y creado se podría utilizar en otros entornos sin problemas pero en nuestro caso se ha hecho servir para poder gestionar todos los sonidos de nuestra aplicación.

3.3 Xbox 360

Xbox 360 es la última videoconsola de Microsoft. Se trata de la consola de séptima generación que actualmente posee la mayor red de jugadores online y que ha sido un gran éxito tanto en ventas como en catálogo de juegos que posee.

Esta consola lanzada en Europa el 2 de diciembre de 2005 tiene una CPU de tres núcleos de 3.2 GHz y una GPU de 500 MHz

Esta máquina dispone de gran variedad de periféricos y de tipos de control distintos y aún le queda un gran recorrido en este sentido con la llegada de Kinect en los próximos meses.

Además esta es la única consola que permite, de forma legal, crear tus propios videojuegos sin necesidad de tener un equipo especial de desarrollo.

3.4 C#

C Sharp es un lenguaje de programación que apareció en 2001 de la mano de Microsoft.

Se trata de un lenguaje de programación orientado a objetos que está basado en C++ pero incluye ideas de java y de otros lenguajes.

Una de las características más importantes que tiene es que es multiplataforma y es capaz incluso de ser interpretado por una Xbox 360.

3.5 Microsoft Visual Studio 2008

Microsoft Visual Studio 2008 es un entorno de desarrollo integrado que se utiliza en entornos de Windows.

Con este entorno podemos compilar código de diferentes lenguajes entre ellos el C # y tiene algunas características que lo hacen muy potente, y a su vez, muy cómodo para trabajar.

Una de estas características es que cuando se produce un error en tiempo de ejecución y la aplicación se detiene, el programa te indica que línea ha sido la causante del error y también que error exactamente se ha producido.

Esto junto con otras buenas prácticas hacen que la programación sea muy cómoda y se eviten errores comunes. Una de estas prácticas es tan sencilla como que al seleccionar una llave o paréntesis el programa te marca cuál es la llave o paréntesis que cierra la actual.

De esta forma es muy cómodo y sencillo ver si hemos delimitado bien nuestro código y poder saber con exactitud cuando se producen muchas condiciones, que parte pertenece a cada una de ellas.

Además la característica más importante es que permite integrar el XNA y una vez tienes los dos integrados tienes la opción de compilar utilizando el ordenador y que a través de una red doméstica se envíe el resultado directamente la Xbox 360 y se ejecute únicamente allí.

4 Diseño

En esta sección vamos a describir las funcionalidades del software desde un punto de vista orientado al usuario. De esta forma será posible hacerse una idea de las capacidades del software de una forma bastante sencilla. Además se debe destacar que se van a utilizar casos generales para explicar mejor su funcionamiento.

4.1 Diagrama de Casos de Uso

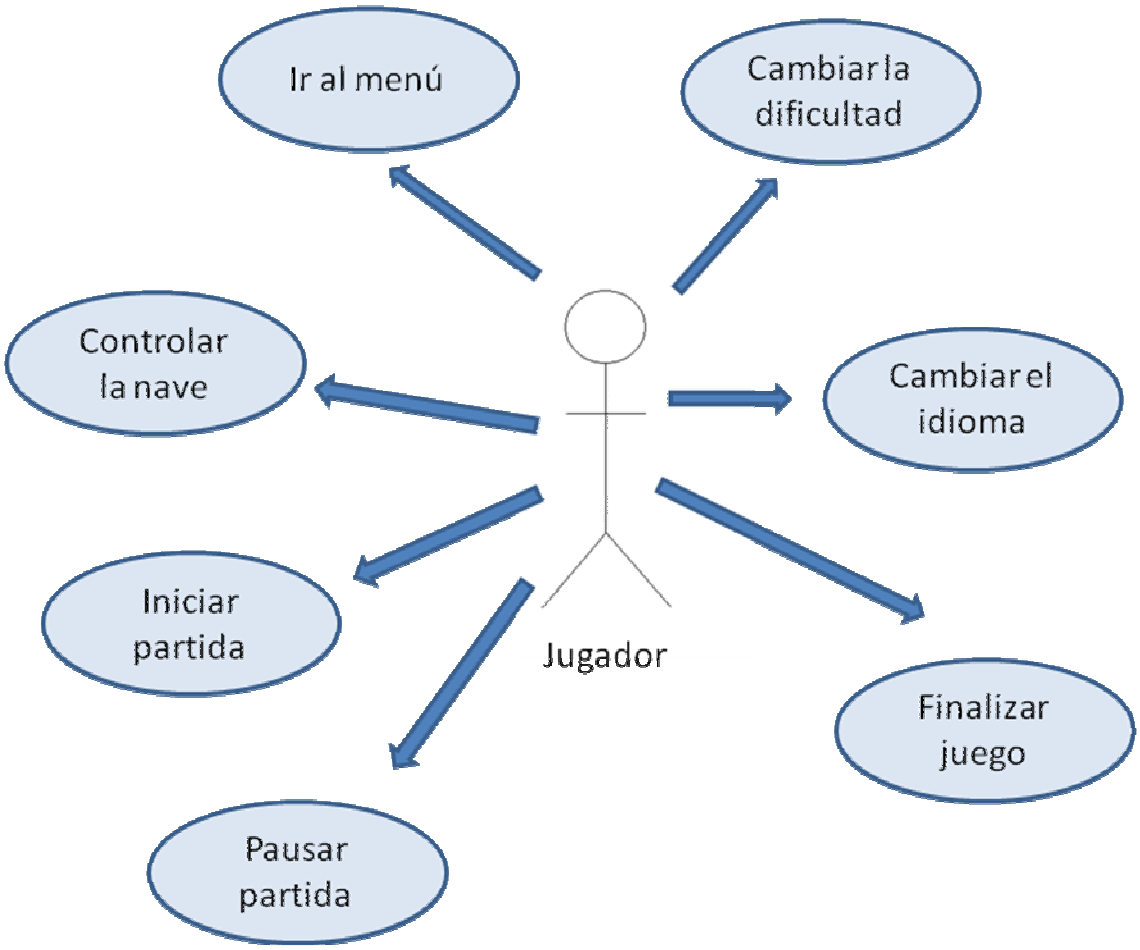


Figura 1 Casos de Uso.

4.2 Especificación de Casos de Uso

En este apartado se va a explicar al detalle los pasos que va a seguir nuestro sistema para realizar cada uno de las acciones posibles que tiene el usuario. La metodología a seguir en este apartado para cada uno de los Casos de Uso será:

- Nombre del caso

- Descripción general del caso: En este apartado se dará una breve explicación sobre el caso de uso.
- Precondiciones: En este apartado se explicarán las condiciones que deben cumplirse para poderse dar este caso de uso.
- Flujo de eventos principal: En este apartado se detallará que serie de pasos se dan durante el caso de uso.
- Flujo de eventos alternativos: Aquí se detallará que otras alternativas pueden ocurrir.
- Postcondiciones: En caso de existir alguna condición que deba cumplirse de forma obligatoria al terminar el caso de uso, esta será indicada en este espacio.
- Notas: Si es necesario dar algún tipo de explicación extra para que se entienda mejor el funcionamiento de el caso de uso se explicará aquí.

Nombre del Caso de Uso	Ir al menú
Descripción	El jugador solicita ir al menú principal del juego.
Precondiciones	El juego debe estar en pausa o terminado para hacer esta solicitud
Flujo principal	1: El jugador solicita ir al menú principal
	2: El sistema detiene el juego y reinicializa todos los elementos de la pantalla
	3:El sistema cambia del módulo jugando al de menú principal
	4:El sistema muestra por pantalla el menú principal con su correspondiente música
Flujos alternativos	No existen flujos alternativos en este caso de uso.
Postcondiciones	Todos los elementos de la pantalla deben haber sido reinicializados y estar listos para su nuevo uso.
Notas	La dificultad y el idioma no se ven alterados y por tanto mantiene los valores que tenían durante la partida.

Nombre del Caso de Uso	Cambiar la dificultad
Descripción	El jugador solicita cambiar la dificultad del juego.
Precondiciones	El juego debe estar en el menú principal.
Flujo principal	1: El jugador solicita ver las opciones
	2: El sistema muestra el apartado de opciones del menú.

	3: El jugador solicita escoger la dificultad.
	4: El sistema muestra las dificultades posibles.
	5: El jugador escoge la dificultad que desea.
	6: El sistema cambia la dificultad actual por la escogida por el usuario.
	7: El sistema muestra el menú principal y reproduce un efecto de sonido para mostrar que ha efectuado el cambio.
Flujos alternativos	No existen flujos alternativos.
Postcondiciones	La dificultad escogida por el jugador se debe mantener en todo momento hasta que se solicite un nuevo cambio.
Notas	Al reiniciar el sistema la dificultad volverá a su parámetro normal, es decir, la dificultad pasará a ser fácil.

Nombre del Caso de Uso	Cambiar el idioma
Descripción	El jugador solicita cambiar el idioma del juego.
Precondiciones	El juego debe estar en el menú principal.
Flujo principal	1: El jugador solicita ver las opciones
	2: El sistema muestra el apartado de opciones del menú.
	3: El jugador solicita escoger el idioma.
	4: El sistema muestra los idiomas disponibles.
	5: El jugador escoge el idioma que desea.
	6: El sistema cambia el idioma actual por el escogido por el usuario.
	7: El sistema reproduce un efecto de sonido y muestra el menú principal en el idioma seleccionado.
Flujos alternativos	No existen flujos alternativos.
Postcondiciones	El idioma escogido por el jugador se debe mantener en todo momento hasta que se solicite un nuevo cambio.
Notas	Al reiniciar el sistema el idioma volverá a su parámetro normal, es decir, el idioma pasará a ser inglés.

Nombre del Caso de Uso	Finalizar el juego
Descripción	El jugador solicita cerrar el juego y volver al menú principal de la consola.
Precondiciones	El juego debe estar en el menú principal o pausado
Flujo principal	1: El jugador solicita finalizar el videojuego.
	2: El sistema libera todos los recursos y devuelve al jugador al menú principal de la consola.

Flujos alternativos	1: El jugador pulsa el botón Back durante la pantalla
	2: El sistema libera todos los recursos y devuelve al jugador al menú principal de la consola.
Postcondiciones	La consola debe tener el control total de los inputs del jugador.
Notas	La opción de salir del juego pulsando Back podría no haber sido incluida pero se ha creado porque dentro de los juegos amateurs para Xbox 360 se considera un estándar tener esa opción.

Nombre del Caso de Uso	Pausar partida
Descripción	El jugador solicita detener el tiempo en la partida.
Precondiciones	El juego debe estar en modo jugando
Flujo principal	1: El jugador pulsa el botón START
	2: El sistema detiene el tiempo de la aplicación evitando que ningún objeto de la pantalla se pueda actualizar ni se realice ningún tipo de cálculo.
	3: El sistema detiene la música actual y los elementos en pantalla y los sustituye por los elementos del módulo de pausa.
	4: El sistema le quita el control sobre la nave al jugador y utiliza los inputs para el menú de pausa.
Flujos alternativos	No hay flujos alternativos.
Postcondiciones	El juego debe de estar en modo pausa y ningún elemento del modo jugando debe estar activo.
Notas	Cuando el juego está en modo pausa se puede seleccionar la opción de continuar jugando para volver al estado anterior y deshacer todos los cambios realizados por el sistema. En este proceso el jugador no pierde ningún tipo de información y todo vuelve a ser exactamente como era en el instante de iniciar la pausa.

Nombre del Caso de Uso	Iniciar partida
Descripción	El jugador pide iniciar una nueva partida.
Precondiciones	El juego debe estar en el menú principal o en pausa.
Flujo principal	1: El jugador solicita empezar una nueva partida.
	2: El sistema comprueba el idioma y dificultad seleccionados.

	3: El sistema muestra la primera pantalla desde su inicio, activa la música de la pantalla e inicia el temporizador.
	4: El sistema utiliza el tiempo del temporizador junto con el nivel de dificultad para determinar la cantidad de enemigos que deben aparecer en pantalla y sus características siguiendo unos parámetros aleatorios acotados por unos baremos.
	5: El sistema permite al jugador controlar la nave.
Flujos alternativos	No existen flujos alternativos
Postcondiciones	El jugador debe ser capaz de controlar la nave y de solicitar una pausa si lo desea. Por su parte el sistema debe ser capaz de actualizar y mostrar en pantalla todos los enemigos que se irán creando durante la ejecución y también ser capaz de detectar y reaccionar ante las diferentes colisiones que se van a ir produciendo.
Notas	Iniciar partida es el caso de uso que da paso a las características que tienen una complejidad más alta por parte del sistema.

Nombre del Caso de Uso	Controlar la nave
Descripción	El jugador realiza alguna acción mientras tiene el control de la nave.
Precondiciones	El juego debe estar en modo jugando.
Flujo principal	1: El jugador mueve la palanca de la guitarra.
	2: El sistema comprueba si la dirección de la palanca es superior o inferior a la posición normal.
	3: El sistema modifica la posición relativa al eje Y de la pantalla de la nave teniendo en cuenta la dirección en que se ha movido la palanca.
	4: El sistema dibuja la nave en la nueva posición y tiene en cuenta esta localización para calcular colisiones.
Flujos alternativos	1: El jugador presiona alguno de los cinco botones de colores que tiene la guitarra.
	2: El sistema comprueba que ese botón anteriormente no estuviera pulsado.
	3: En caso de no estar pulsado anteriormente, el sistema crea un nuevo proyectil desde la nave con velocidad constante hacia la derecha que tendrá el mismo color que el botón pulsado.
	4: El sistema tendrá en cuenta ese proyectil para comprobar si colisiona con algún otro

	elemento.
	5: El sistema cambia el estado del botón de pulsado a levantado cuando detecte que ya no está siendo pulsado.
Postcondiciones	La reacción del sistema ante las peticiones del jugador debe ser rápida y suave.
Notas	La nave dispone de unos límites que no le permiten salirse de un rango específico. El límite inferior coincide con el borde de la pantalla y está en una posición que puede ser atacada por los enemigos pero el límite superior que está también en el borde de la pantalla es un lugar seguro en el que los enemigos no le podrán alcanzar. Esto se ha hecho adrede para poder realizar la defensa de este proyecto y enseñar las diferentes dificultades sin miedo a morir varias veces debido a que la dificultad crece bastante.

4.3 Arquitectura del Sistema

A continuación se muestran unos esquemas que relacionan las clases que hemos creado en este proyecto con las clases de XNA del cual derivan para poder ser compatibles con el Hardware de la Xbox 360

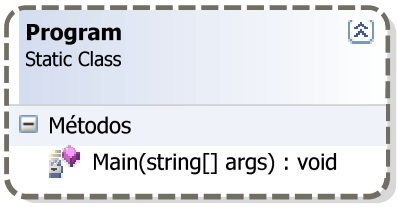


Figura 2 Program Class

Esta clase abstracta es la primera clase que se ejecuta en la aplicación. Simplemente lo que hace esta clase es inicializar la clase Game que será nuestra clase principal durante toda la ejecución.

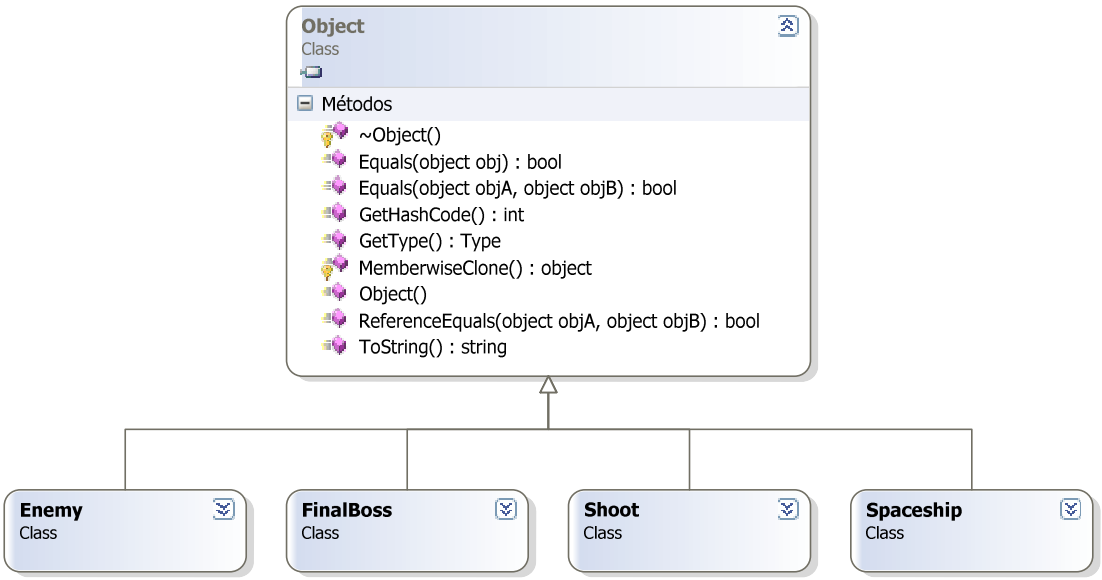


Figura 3 Object Class

En la Figura 3 vemos que existen cuatro clases que heredan directamente de la clase Object de XNA. Esto se debe a que es la clase principal con la que trabaja todo el sistema y es necesaria para poder crear cualquier tipo de objeto que después vayamos a utilizar.

Figura 4 GameComponent Class

En esta nueva figura vemos que existen tres clases que heredan de Drawable Game Component y estas son las clases que crean componentes del juego y que además se utilizan para controlar cuando debemos detener el flujo del tiempo durante la aplicación.

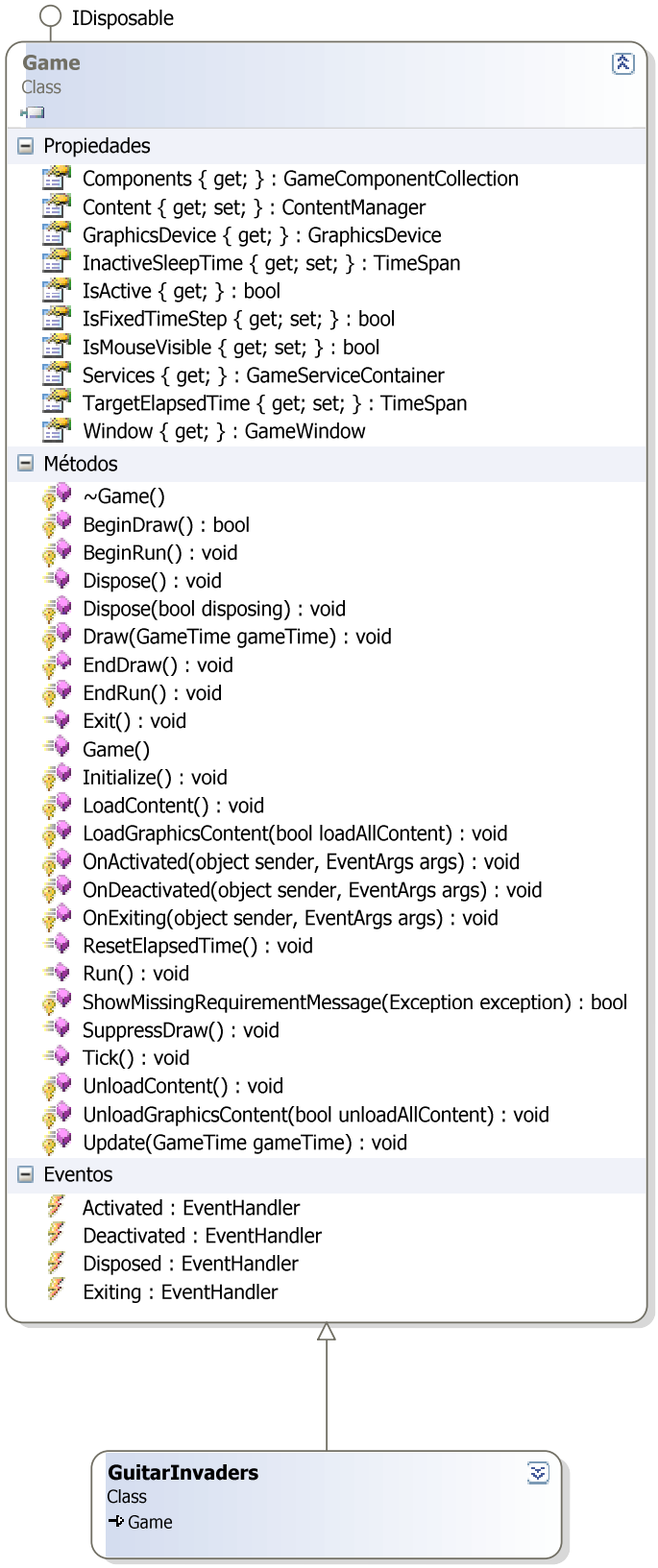


Figura 5 Game Class

La Figura 5 Game Class es la más importante de todas las que tenemos puesto que aquí se puede contemplar que Guitar Invaders, que es nuestra clase principal en el juego, hereda una gran cantidad de funcionalidades que son las que hacen realmente que pueda existir una comunicación entre nuestro código fuente y el Hardware de la consola.

4.3.1 Clases de la aplicación

En este apartado veremos más de cerca cada una de las clases que participan en este proyecto y qué papel juegan.

4.3.1.1 Enemy

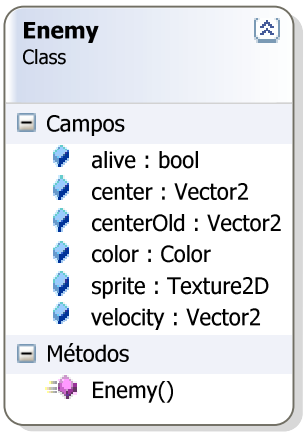


Figura 6 Enemy Class

La clase Enemy es la clase encargada de crear los objetos Enemy, es decir, los enemigos que vemos en pantalla.

Esta clase tiene definidos una serie de campos que sirven para controlar si está vivo o no, en qué posición de la pantalla se encuentra, de qué color es, que textura debemos utilizar para representarlo y la velocidad que tiene.

Es una clase bastante sencilla pero modificando estos campos podemos hacer que los enemigos se comporten de la forma deseada.

4.3.1.2 Final Boss

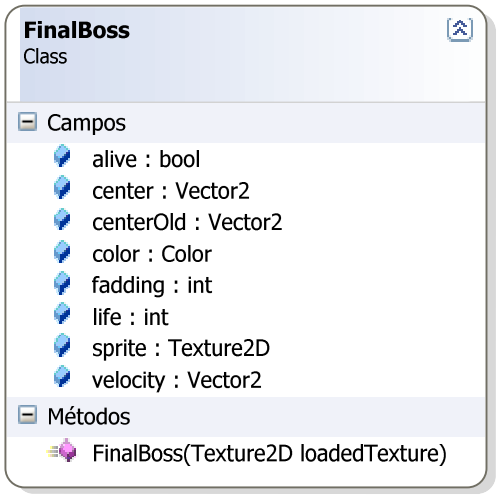


Figura 7 Final Boss Class

FinalBoss es la clase que permite la creación del enemigo final. Como se puede comprobar el concepto es muy similar al que había en Enemy aunque en esta clase tenemos un par de detalles más que son importantes.

Por un lado life nos indicará la cantidad de puntos de vida que tiene el enemigo, puesto que este no se morirá de un solo disparo.

Y por el otro tenemos el campo fadding que se utilizará como indicador para que el sistema sepa cuando tiene que actualizar la imagen mostrada representando al enemigo final, generando de esta forma una animación mediante código.

Puntualizar que en este caso para crear el enemigo se le pasa por parámetro la textura con las diferentes imágenes que formaran parte de la animación.

4.3.1.3 Shoot

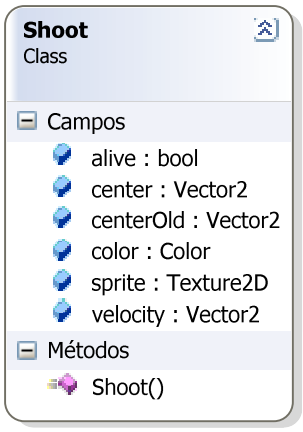


Figura 8 Shoot Class

La clase Shoot es la clase que se utiliza para crear los proyectiles que disparará tanto nuestra nave protagonista como la nave del enemigo final.

Esta clase dispone de las mismas características que la clase Enemy

4.3.1.4 Spaceship

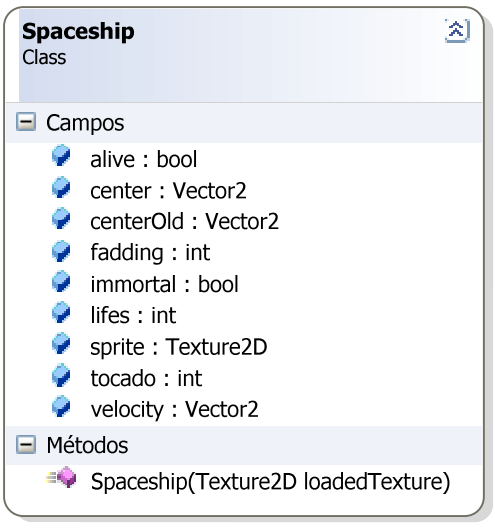


Figura 9 Spaceship Class

La clase Spaceship que vemos en esta figura es la que se encarga de crear nuestra nave protagonista.

Esta clase sigue patrones similares a las anteriores pero tiene algunos detalles extras.

Utiliza el fadding y el lifes para realizar una pequeña animación de transparencias al ser impactadas y controlar la cantidad de vidas de la misma forma que hacia FinalBoss.

Tenemos un nuevo Campo denominado immortal que hace que la nave no pueda recibir daño mientras esté activo. Este campo se utiliza al recibir daño y nos permite hacer que durante unos segundos la nave sea invulnerable.

Tocado solamente es un indicador que se utiliza para agregar un efecto de sonido al ser impactados.

4.3.1.5 Menu

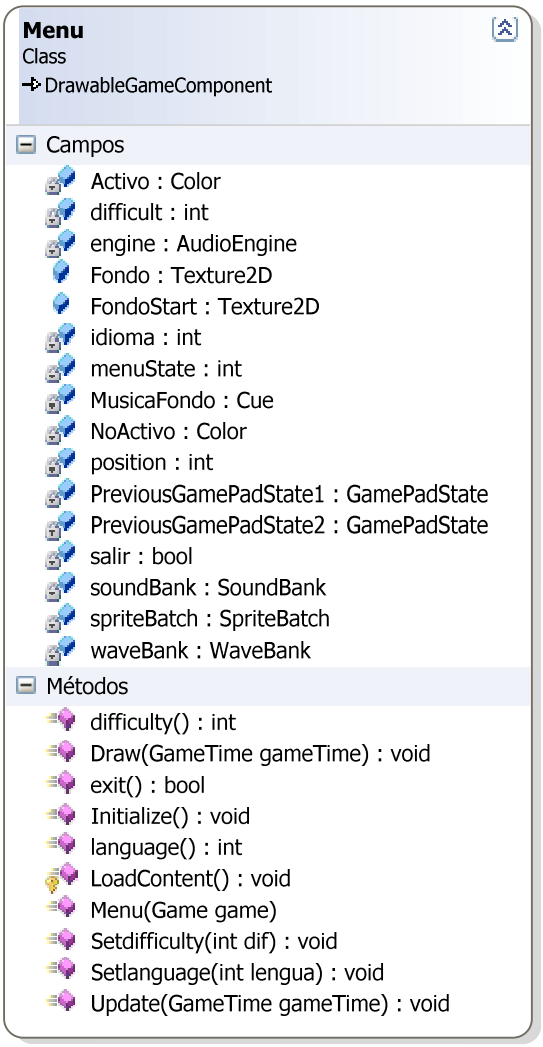


Figura 10 Menu Class

La Figura 10 muestra la clase Menu que ya dispone de una complejidad bastante más elevada que las anteriores.

Se trata de la clase que controla todo el menú principal y por ello es una de las clases más extensas de la aplicación.

En este caso podemos ver que existen una variedad de Campos bastante más grande aunque sus nombres y sus tipos de datos son bastante explicativos en cuanto a su utilidad se refiere.

Centrándonos un poco más en los métodos se puede ver que Menú dispone de un par de métodos para gestionar la dificultad y la lengua de la aplicación. También dispone de un método para salir y el resto de métodos son un poco más complejos.

El primer de estos métodos que nos encontramos es el Draw. Este método se encarga de mostrar en pantalla todo lo que se le indique en su interior cada cierto intervalo de tiempo.

El segundo es Initialize. Este método se encarga de inicializar todo lo que sea necesario.

LoadContent es el tercero que nos encontramos y este sirve para cargar en memoria todo lo que vayamos a utilizar durante la ejecución.

En cuanto al cuarto y último, se trata del método Update que es el encargado de transmitir órdenes y actualizar parámetros cada cierto intervalo de tiempo. Este método es el que controla la ejecución principal tanto en esta clase como en el resto de las que quedan por ver.

4.3.1.6 Death

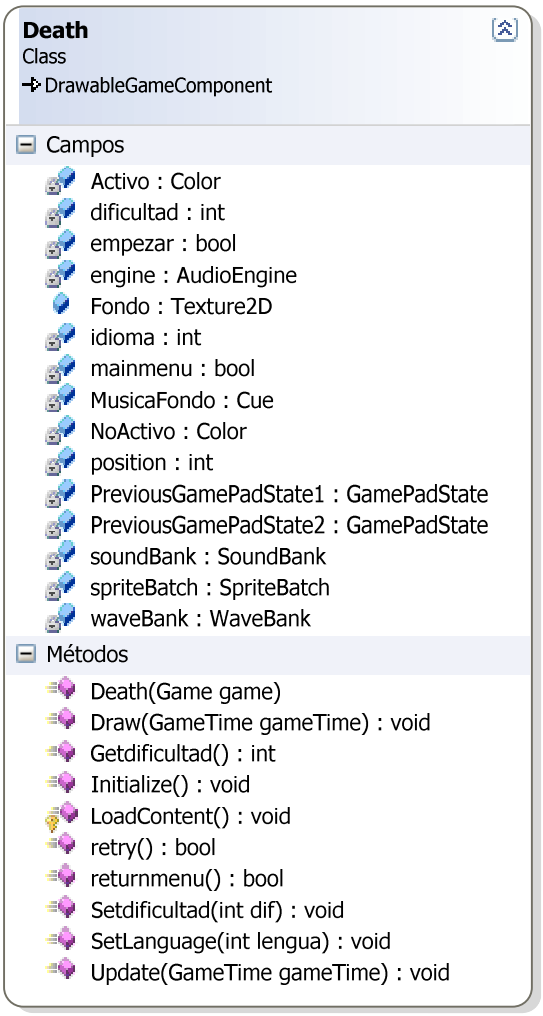


Figura 11 Death Class

La clase Death tiene una gran similitud en cuanto a estructura se refiere con la clase Menu que hemos visto anteriormente.

La diferencia principal entre ambas es que la clase Death se ejecuta cuando nuestra nave pierde todas sus vidas y entonces nos sale un pequeño selector del estilo de los que hay en el menú para decidir si queremos reiniciar el nivel o si preferimos enlazar con el menú principal.

4.3.1.7 Pausa

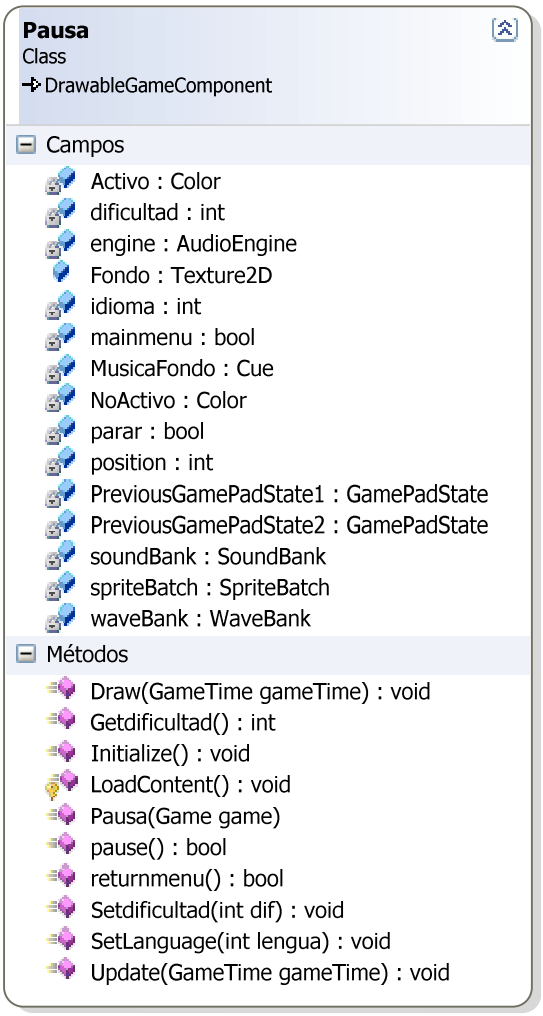


Figura 12 Pausa Class

La clase Pausa que vemos en la Figura 12 tiene una estructura muy similar a las dos anteriores pero con la diferencia de que es la única que mantiene todos los datos de la partida actual inalterados y por tanto es la única capaz reanudar el juego sin provocar ningún conflicto.

Es por ello que aunque las estructuras de estas clases puedan ser similares sus funcionamientos a la práctica no se parecen tanto.

4.3.1.8 Game

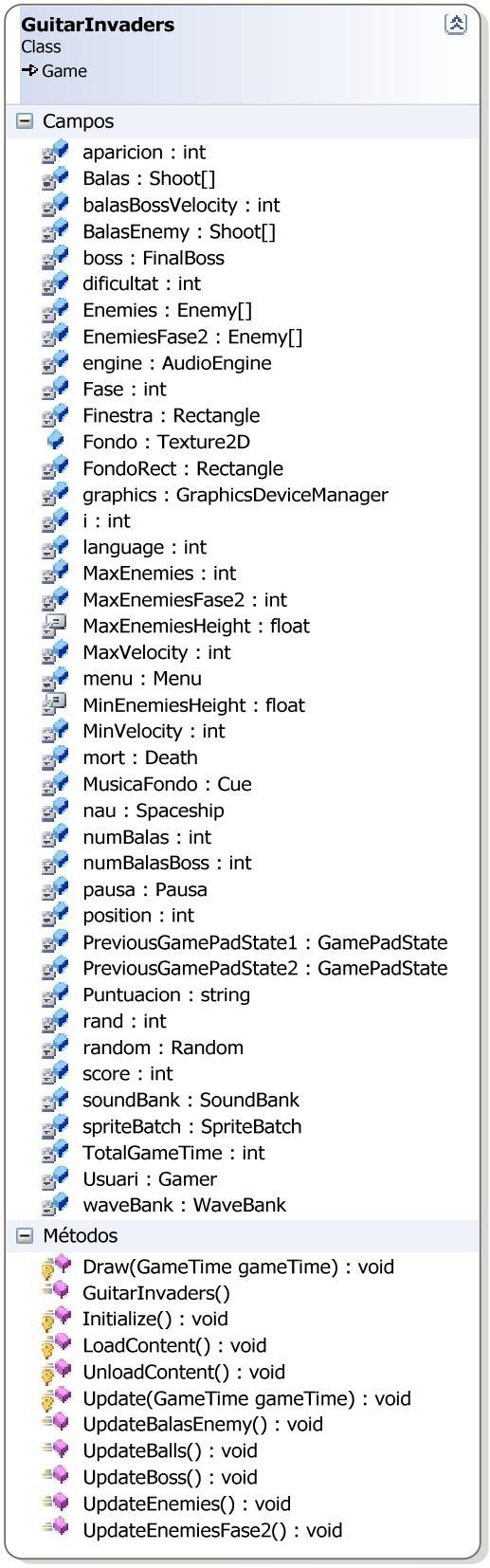


Figura 13 Guitar Invaders Class

Sin duda alguna esta clase es el corazón de todo el videojuego.

En estos métodos se crean y destruyen todos los tipos de objetos que tenemos y también aquí interactúan entre ellos.

La clase Game está pensada como si fuera la columna vertebral de todo el sistema y es dónde se generan todos los eventos que no forman parte de ningún menú.

Como se puede observar en la Figura 13 existen una gran cantidad de métodos que llevan la palabra Update que son los que se utilizan para modificar y actualizar todos los objetos a medida que avanza el tiempo.

También es en estos métodos donde se calculan las posibles colisiones y se utilizan la mayoría de los efectos de sonido.

4.4 Interfaz de usuario

Este apartado se va a centrar en el entorno gráfico y sonoro del juego.

Es importante saber que todo el apartado visual ha sido diseñado y creado específicamente para este videojuego. Eso quiere decir que los diseños intentan cumplir el objetivo de mezclar en algún sentido el espacio y la música que son los dos temas centrales del proyecto en cuanto a creatividad artística se refiere.

Figura 14: Pantalla de inicio

Esta es la primera imagen que veremos en el videojuego. Se trata de la pantalla de inicio que nos da paso al menú y en esta pantalla podemos ver claramente la idea que

se ha comentado anteriormente sobre este intento de fundir los elementos musicales con el espacio para darle una identidad única y facilmente reconocible a nuestro videojuego.

A simple vista lo mas destacable es la llave de sol que sustituye a la letra " G " en nuestro título y el uso de una "i" que interviene en ambas lineas de texto a la misma vez. Además, este cartel junto con las naves espaciales con motivos musicales, constituyen el equilibrio que intentaremos buscar a lo largo del juego y por ello todo esto en conjunto forma el logotipo de Guitar Invaders.

Como último detalle destacar que la imagen de fondo es una imagen real del planeta Marte que hasido tratada para darle un tono rojizo con degradado.

Figura 15: Menú principal

La Figura 15 corresponde al menú principal de la aplicación.

Este menú es vertical para poder interactuar fácilmente con la guitarra y dar una mayor comodidad al usuario. Además se trata de un menú sencillo que utiliza el color negro para indicar la opción que tenemos ahora mismo seleccionada y el blanco para el resto.

En cuanto al logotipo y al fondo de pantalla, se mantienen respecto a la pantalla anterior para darle una continuidad a la estética del juego

Como nota final añadir que esta toma ha sido realizada después de cambiar el idioma del inglés que viene por defecto al catalán.

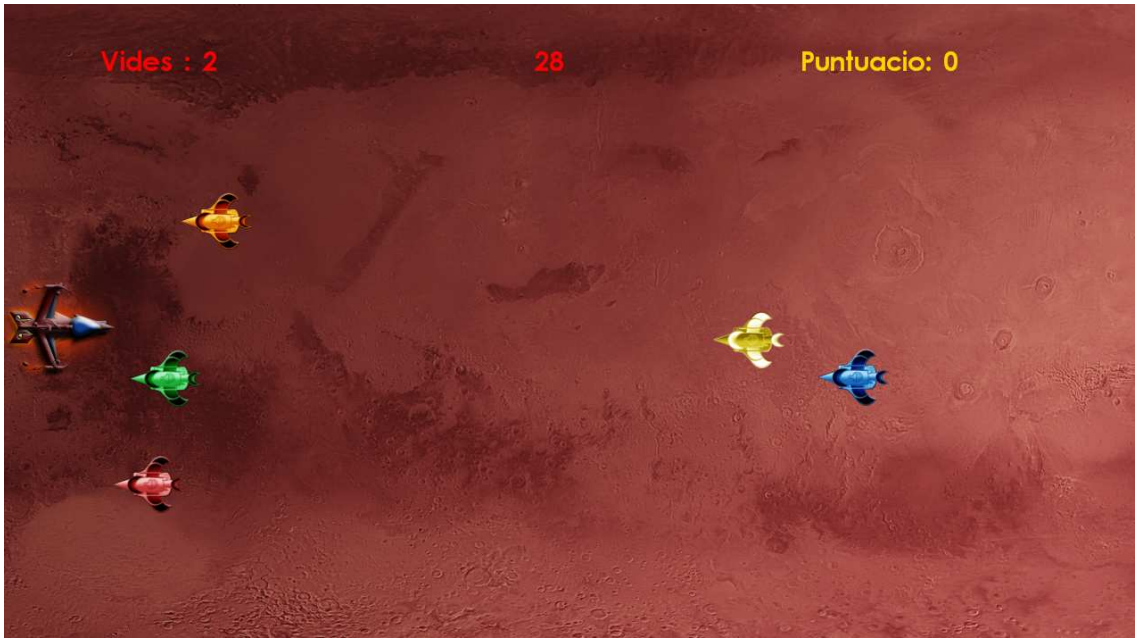


Figura 16 Primera oleada

Esta tercera imagen está extraída de la pantalla de juego. En ella podemos ver perfectamente que tenemos nuestra nave situada a la izquierda de la pantalla y los enemigos se nos irán acercando desde el extremo derecho.

Esta toma es del principio de la pantalla y nos muestra que existen enemigos de cinco colores distintos a los cuales habrá que matar con su color correspondiente o esquivarlos. Además en la parte superior podemos ver que hay unas indicaciones que nos marcan la cantidad de vidas que nos quedan, el tiempo transcurrido y la puntuación que tenemos en este momento.

Cabe destacar que esta primera oleada está programada de tal manera para que sea totalmente aleatoria y por tanto cada vez que juguemos la partida será distinta.

Figura 17 Primera oleada difícil

Esta nueva imagen corresponde también a la primera oleada pero nos sirve para explicar un poco el tema de las dificultades que nos vamos a encontrar. Existen características que se mantienen entre dificultades como por ejemplo que el color de los enemigos sea aleatorio y que en este caso hagan trayectorias horizontales pero como resalta en la imagen, la cantidad de naves enemigas ha aumentado debido a un aumento del nivel de dificultad y aunque en esta imagen no se puede apreciar, la velocidad de dichas naves también se ha incrementado.

Figura 18 Menú Pausa

Esto que se muestra aquí es el menú de pausa. A este menú se accede apretando el botón START durante cualquier momento de la partida y sigue la misma estética que el

resto de menús del juego aunque la tonalidad del fondo se ha variado para indicar que este menú es distinto al resto, pues es el único menú que permite continuar la partida exactamente en el mismo lugar donde estaba.

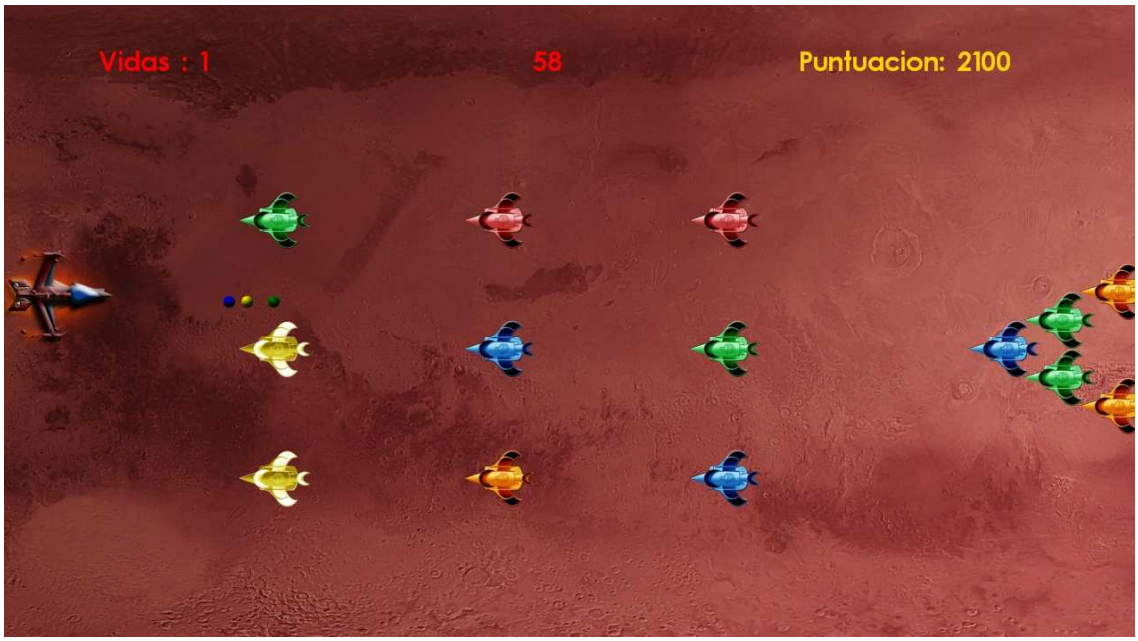


Figura 19 Segunda oleada

Esta nueva imagen corresponde a la segunda oleada de enemigos que nos aparecerán durante la pantalla. La diferencia principal respecto a la oleada anterior es que estos enemigos han sido programados para comportarse de una manera perfectamente definida y además algunas de estas naves enemigas tienen trayectorias oblicuas.

Otro aspecto a destacar de esta imagen son los proyectiles que se ven de distintos colores que precisamente son los disparos de nuestra nave. Estos proyectiles tienen esta forma en honor a la redonda que indica la duración de una nota en música.

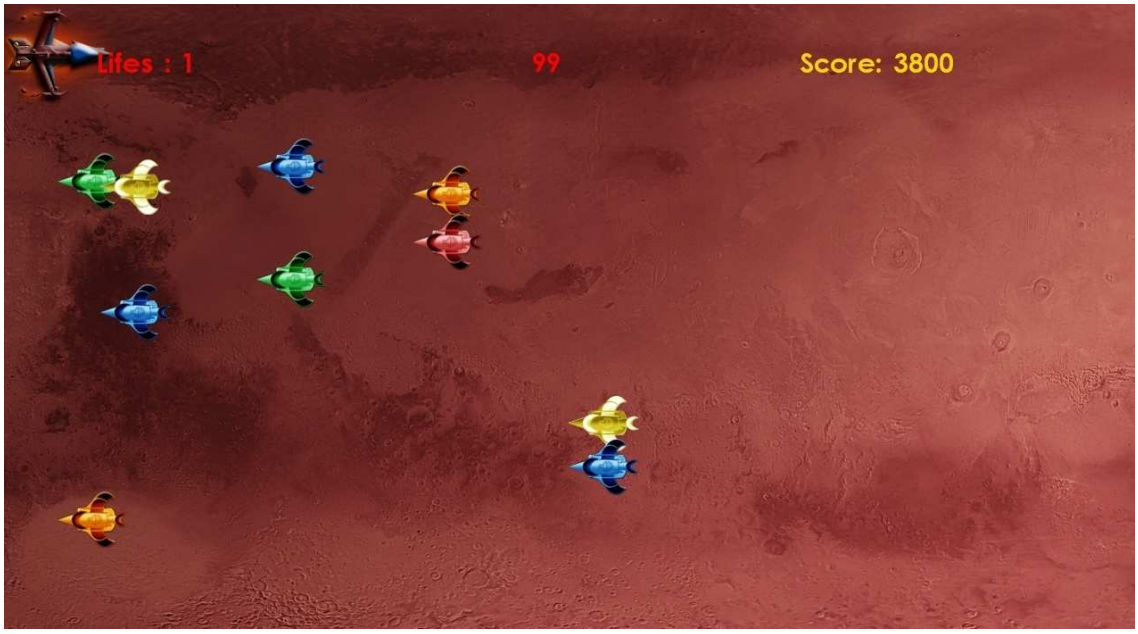


Figura 20 Tercera oleada

Esta imagen corresponde al final de la tercera oleada que es la más difícil pero destaca sobretodo por la localización de nuestra nave. Se ha habilitado la posibilidad de "escondernos" de las naves enemigas en el borde superior de la pantalla. Si este juego quisiéramos venderlo deberíamos eliminar esta opción y hacer que la nave no pudiese alcanzar una altura que le permitiese esquivar a todos los enemigos, pero en este caso se ha optado por permitir esta opción para poder mostrar todos los niveles de dificultad sin miedo a que nuestra nave sea destruida una y otra vez sin poder avanzar y no poder mostrar todo el nivel.

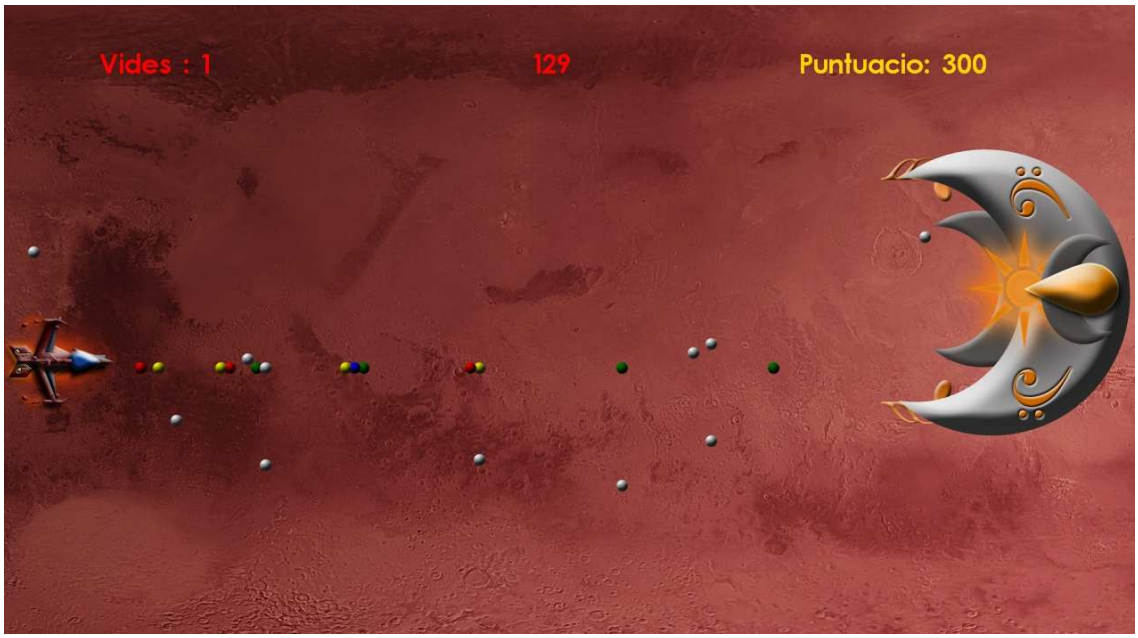


Figura 21 Enemigo final

Finalmente llegamos al enemigo final de nuestra pantalla y en este caso de nuestro juego también. Se trata de una nave gigante que nos disaparara proyectiles grises que nosotros deberemos de ir esquivando. Además esta nave tiene una animación que hace que cambie de color cada cierto intervalo de tiempo haciendo que sea mucho más difícil de matar porque deberemos de impactarle justo en el centro con el color correspondiente en el momento de la colisión.

Una vez enseñado el juego a nivel general en cuanto a su apartado gráfico se merece vamos a resaltar algunos pequeños detalles.

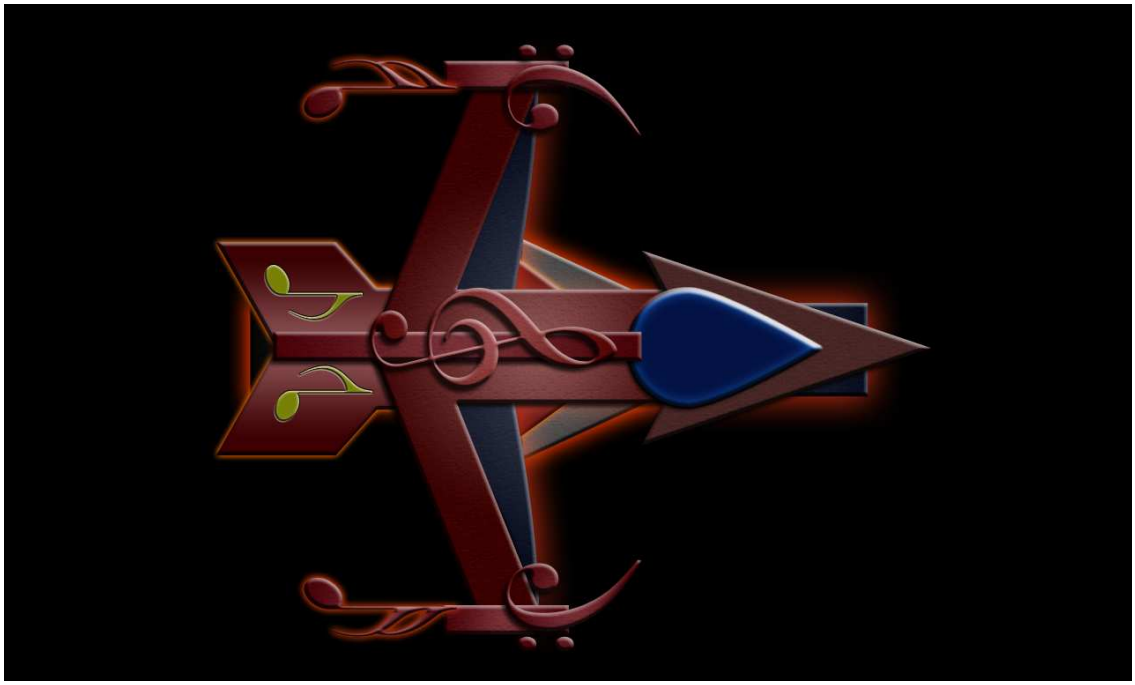


Figura 22 Protagonista

Este es el diseño de la nave protagonista visto más de cerca. Como se puede apreciar en la imagen, la nave ha sido creada íntegramente utilizando formas comunes como flechas o rectángulos y además los detalles se han realizado utilizando notas y simbolos musicales como la llave de sol, las corcheras etc.

Esta nave además dispone de una animación cuando es impactada que reproduce un ruido de alarma y hace un flash de color cambiando el fondo de la pantalla de golpe y haciendo que la nave aparezca y desaparezca durante unos breves instantes.

Figura 23 Nave enemiga

Las naves enemigas quizás a simple vista no parezca que han sido creadas de la misma forma pero tal y como se ha resaltado en esta imagen podemos ver que realmente también han sido creadas utilizando los mismos tipos de formas y también elementos musicales.

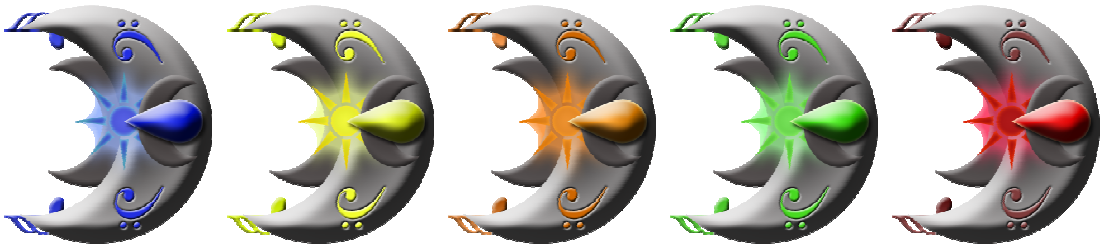


Figura 24 Animación enemigo final

Para acabar este apartado de los elementos visuales que se muestran durante el juego se ha incluido esta última imagen que muestra la animación del enemigo final el cuál tiene los mismos colores que diponemos en la guitarra para acertarle. Esta nave de tamaño mayor que el resto, ha sido realizada exactamente de la misma forma y dispone de elementos musicales igual que las otras.

En cuanto a los sonidos utilizados durante todo el juego, han sido seleccionados de librerías de licencia libre y se ha incluido una canción distinta para cada menú. También existen efectos de sonido que se activan al disparar nuestras armas, al impactar a un enemigo o al ser impactados.

5 Codificación y pruebas

Este capítulo está centrado principalmente en mostrar el tipo de codificación que se ha seguido durante el desarrollo y también para explicar que clases de pruebas se han realizado.

5.1 Estilo de codificación

Al realizar este proyecto ha sido necesario adaptarse al método de trabajo de Microsoft y su XNA. Esto significa que existen un grupo de clases que controlan partes fundamentales del videojuego como podría ser el gestor de gráficos o de sonidos, que están implementados de una manera y no se pueden modificar porque sino la consola sería incapaz de reconocer lo que le pedimos que haga.

Este hecho provoca que tengamos que adaptar nuestras ideas y nuestros diseños a una serie de clases, métodos y variables que son fundamentales para realizar una buena comunicación con la Xbox 360.

El método que se ha seguido a la hora de codificar el juego ha sido el de crear objetos, clases y complementos que interactúan entre ellos a través de nuestra clase Game.cs que es la clase que gestiona todos los recursos de la aplicación.

Durante la creación del proyecto en ningún momento se ha especificado un estándar que debía de ser seguido estrictamente pero si se ha intentado, en la mesura de lo posible, hacer uso de buenas prácticas.

Estas buenas prácticas consisten en utilizar nombres de variables y clases que se puedan identificar con la función que desempeñan. También se ha tenido en cuenta separar métodos complejos como los que tienen que actualizar los valores de los objetos o realizar cálculos entre ellos.

Si contamos los diferentes archivos que forman nuestra aplicación veremos que prácticamente se alcanzan las 2500 líneas de código y tal cantidad puede dificultar un poco la comprensión del mismo. Es por ello que se ha intentado en todo momento mantener los elementos con características similares lo más agrupados posibles para que sea más fácil de comprender la ejecución.

De todas formas existe un grado de complejidad en el sistema sobre todo cuando mezclamos a los componentes con el resto del código que hace que se requiera un nivel alto de comprensión para poder entender que eventos se están ejecutando en cada momento y que cálculos realiza la consola por tal de determinar que debe ejecutarse.

5.2 Pruebas

En este apartado se describirán que pruebas se han ido realizando a lo largo del proyecto por tal de asegurarnos de su correcto funcionamiento y evitar posibles fallos.

El primer tipo de prueba que se ha realizado continuamente durante todo el proyecto es la prueba de unidad que se basa simplemente en comprobar cada pequeña nueva porción de código y mirar si esta hacía exactamente lo que se esperaba de ella.

Este tipo de prueba ha sido muy importante para encontrar fallos en variables y en bucles principalmente y tiene la gran ventaja de trabajar en espacios muy acotados donde encontrar y corregir un error es algo muy sencillo.

El siguiente tipo de prueba, las prueba de integración, se trata de comprobar módulos enteros y probar todas sus posibilidades para ver que interactuaban con el resto de módulos de forma normal y no se producía ningún tipo de anomalía.

En esta aplicación los módulos están conectados entre sí mediante el tiempo y eso hace que para realizar este tipo de pruebas fuese un factor determinante a tener en cuenta y que inducía a error muy frecuentemente.

El siguiente grupo de pruebas que se realizaban eran las pruebas de validación, es decir, se ejecutaba la aplicación para ver si el resultado mostrado por pantalla satisfacía lo que se esperaba de él.

De esta manera se han corregido gran cantidad de carteles, colores, secuencias de enemigos etc.

Otro tipo de prueba que se ha realizado y que ha sido muy determinante se trata de los Betatesters. Durante la realización de este proyecto y cada cierto tiempo se ha buscado un grupo de personas para que probasen el videojuego e hicieran dos cosas. Por un lado criticar constructivamente lo que veían, por tal de tenerlo en cuenta para la siguiente iteración del proyecto, y por otro lado también intentar encontrar la manera de bloquear el sistema.

Estas pruebas han sido muy importantes de cara a la satisfacción final del videojuego pues aunque en algunos casos solo han ayudado a promover pequeñas mejoras en otros se han dado ideas que han ofrecido visiones antes no contempladas de gran importancia.

Algunos errores que se han corregido mediante este método son por ejemplo multiplicar la puntuación para que sea más gratificante, cambiar los colores del menú a tonos blancos y negros por su mejor lectura, graduar de forma lógica los niveles de dificultad y sobretodo un error que era capaz de bloquear el sistema y que será tratado en el siguiente apartado.

5.3 Errores importantes

Durante la creación de este proyecto han surgido miles de errores de todo tipo pero siempre hay algunos que por su difícil detección o su complicada solución sobresalen por encima de otros.

A continuación se listan algunos de estos errores y se explica que se ha hecho para solucionarlos.

5.3.1 Cambio de requisitos

En un determinado momento del proyecto se decidió incluir la función de pausar la partida pues era algo que los Betatesters siempre comentaban como una función básica que echaban en falta en el juego.

Una vez aceptado que se iba a realizar el cambio se descubrió la imposibilidad de realizarlo. Esto era debido a que se utilizaba el tiempo que nos ofrecía el XNA para realizar todos los eventos pero la consola daba un mensaje de error diciendo que no se disponía de los permisos necesarios para la acción que se quería realizar.

Después de varios días se encontró el verdadero motivo por el cual el sistema no nos daba acceso a modificar nuestra variable del tiempo. Al parecer todo el videojuego estaba siendo controlado utilizando el reloj interno de la consola como referencia para los eventos y evidentemente la consola no permitía que se parase dicho reloj.

Para supera este problema se ha ideado una manera de poder trabajar en tiempo real utilizando el reloj de la consola pero sin la necesidad de tener que pararlo.

Este problema que podría no parecer algo tan serio, en su momento creó un serio cambio de requisitos que hizo rehacer el planteamiento del sistema del tiempo entero y por tanto aunque se pudo reutilizar gran parte del código, para lograr este nuevo objetivo fue necesario volver a empezar y establecer unas bases distintas.

5.3.2 Ejemplo de error encontrado por un Betatester.

Durante una de las sesiones de demostración ante personas ajenas al proyecto, mientras se comentaba las posibles mejoras, el juego se paró y la consola emitió un mensaje que decía textualmente “Se ha producido un error muy grave, por favor póngase en contacto con el distribuidor de este juego para más información “

En ese momento se suponía que la versión debía ser estable y no debía producirse errores, pero la cuestión es que algo había fallado. Además la prueba no estaba siendo monitorizada desde el ordenador con lo cual no tenía forma alguna de saber que era lo que había fallado.

Tras unos minutos hablando con el usuario que había bloqueado el sistema decidimos repetir sus pautas para ver si el error se repetía.

Resulta que si jugabas en el nivel de dificultad fácil, morías antes de llegar al enemigo final y le dabas a reiniciar partida, cuando llegabas a la fase con el enemigo final el juego se bloqueaba.

Resulta curioso que hasta ese momento ninguno de los anteriores usuarios que habían testeado el sistema había perdido todas las vidas antes de llegar al enemigo final en el nivel de dificultad fácil y por eso nunca se había dado ese error con anterioridad.

Una vez encontrado el momento en que sucedía el error sólo fue cuestión de tiempo el averiguar que los proyectiles del enemigo final en dificultad fácil no se iniciaban correctamente si no llegabas a luchar contra él antes de reiniciar el nivel por primera vez.

5.3.3 Error estético en la detección de colisiones

Este error es uno de los que se podían detectar a simple vista y también ha sido uno de los que ha sido más difícil de corregir.

Lo que ocurría básicamente es que el sistema detectaba colisiones entre los proyectiles del protagonista y las naves enemigas pero visualmente estas se morían mucho antes de ser alcanzadas.

Para detectar dichas colisiones se generaba un rectángulo de la mida de la imagen del proyectil y otro del a mida de la imagen de la nave enemiga y se comprobaba si existía algún tipo de intersección entre ellas.

El problema venía de que los rectángulos eran mayores que las naves que había dibujadas dentro y por tanto visualmente daba muy mala impresión.

Después de muchos intentos jugando con las midas de los rectángulos se decidió cambiar la forma de realizar dichas detecciones. En vez de hacer rectángulos y mirar si existía alguna intersección entre ellos se optó por realizar una comprobación pixel a pixel y mirar si en ambas imágenes había un pixel que no fuera transparente a la vez, la cual cosa significaría que existe una colisión.

Esta solución era mejor que la propuesta inicialmente pero el resultado seguía sin ser satisfactorio. Se había conseguido que en muchos casos las colisiones pareciesen más reales pero las naves tienen brillos que hacen que el contorno de la nave no sea estrictamente transparente y ello provocaba falsos positivos y además se exigía al sistema realizar una cantidad de cálculos descomunales para cada proyectil y cada nave y eso hacía que el rendimiento general fuera muy precario.

Como esta solución tampoco se consideraba aceptable se volvió a la idea de base pero añadiendo modificaciones. En vez de utilizar un rectángulo para delimitar la nave se ha pasado a utilizar dos, uno para el eje principal y otro para las alas. Además de esto se ha tenido en cuenta solamente la estructura de dichos elementos y por tanto los rectángulos no sobrepasan las naves en ningún punto.

Esta solución es la que hay implementada pues visualmente las colisiones se ven realistas y además tampoco se perjudica el rendimiento de la aplicación.

6 Conclusiones

6.1 Cumplimiento de objetivos

Al principio de este proyecto se marcaron unos objetivos difíciles de lograr y algunos de ellos incluso se consideraban fuera del alcance del programador debido a su alta dificultad y al gran desconocimiento de cómo podía plantearse y afrontarse todo lo que se había propuesto.

Sin embardo, debido a que este proyecto se presentó como un reto en todos los ámbitos y también como una meta personal, el resultado ha sido muy satisfactorio.

Por un lado tenemos que los objetivos personales del proyectista se han visto completados y además han superado las expectativas que había al inicio del proyecto.

Por el otro tenemos todos los objetivos que se han marcado desde el inicio que finalmente han podido cumplirse.

Además de esto, como el proyecto finalmente se ha constituido utilizando un método iterativo, se ha podido trabajar con nuevos objetivos a cada iteración.

Esto quiere decir que no se han completado únicamente los objetivos esenciales para considerar el proyecto un éxito sino que además otros objetivos que se consideraban secundarios o extras, también se han podido alcanzar.

A grandes rasgos podríamos decir que el objetivo que indicaría con certeza si el proyecto había sido un éxito o si se había fracasado era el de crear una pantalla jugable, es decir, ser capaces de crear una nave que pudiese ser controlada mediante una guitarra y que pudiese matar unos cuantos enemigos.

A este objetivo principal cumplido también hemos de añadirle la creación del menú, la posibilidad de seleccionar el idioma que deseemos, la inclusión de una pausa en la partida a petición del jugador, la creación de un enemigo final de pantalla y la posibilidad de poder jugar a todo esto en tres dificultades distintas.

Es por este motivo que el grado de satisfacción con este proyecto es muy alto ya que se han superado lo que se esperaba de si en un inicio.

Además no podemos olvidarnos que otra de las metas alcanzadas ha sido la creación y personalización de todos los elementos que vemos en pantalla puesto que en un principio se pretendía utilizar imágenes de licencia libre.

6.2 Desviaciones sobre la planificación

Probablemente el punto más negativo que ha tenido este proyecto ha estado relacionado con la previsión temporal. Parte de esta problemática se debe a que por motivos personales y académicos durante una serie de meses fue imposible dedicarle el tiempo necesario a este proyecto y además todo esto se junto con otros problemas de carácter más personal que hicieron que se produjera un retraso bastante importante.

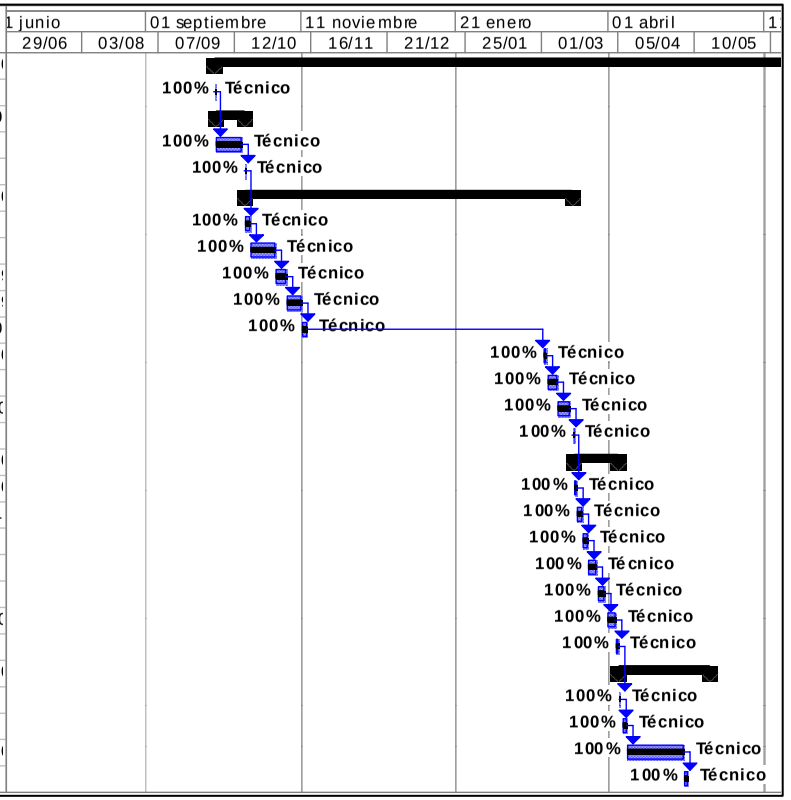
Por otro lado también se ha dado el caso de que como se preveía en los riesgos del proyecto la dificultad y la mala planificación podían causar demoras en el proyecto y aunque se ha intentado evitar este punto en cuanto se ha podido, finalmente ha sucedido y así lo demuestra el diagrama de Gantt que hay en la próxima página.

A destacar está un retraso de 3 meses y comentar que aunque el resultado final en horas ha sido bastante aproximado al que se indicó inicialmente el reparto ha sido un tanto distinto.

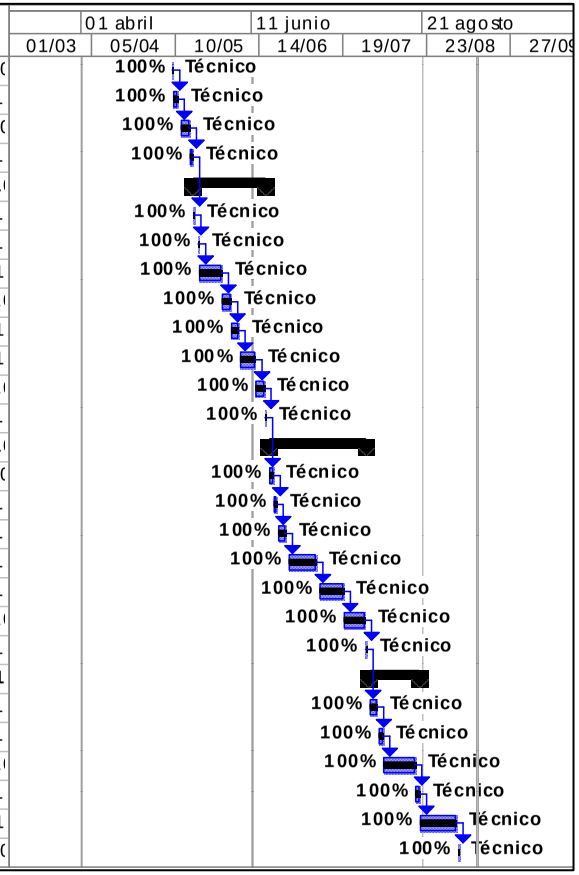
Por una parte las pruebas de testeo cada vez han tenido más importancia y han implicado más horas y por otra el inicio del proyecto ha sido mucho más difícil que el final debido a los conocimientos adquiridos durante el camino.

A continuación el Gantt del proyecto finalizado a día 4 de septiembre de 2010.

Id	Nombre de tarea	Duración	Comienzo	Fin
1	**Guitar Invaders**	**241 días**	**jue 01/10/0**	**sáb 04/09/1**
2	Inicio del proyecto: asignación y matriculaci	2 horas	jue 01/10/0	jue 01/10/0
3	**Planificación**	**9 días**	**vie 02/10/0**	**jue 15/10/0**
4	Estudio de viabilidad	30 horas	vie 02/10/0	mié 14/10/0
5	Aprobación del Estudio de Viabilidad (F	1 hora	jue 15/10/0	jue 15/10/0
6	**Primera Iteración**	**106 días**	**jue 15/10/0**	**dom 14/03/1**
7	Estudio de los requisitos	8 horas	jue 15/10/0	dom 18/10/0
8	Diseño del sistema	24 horas	dom 18/10/0	jue 29/10/0
9	Creación y control de la nave	12 horas	jue 29/10/0	mar 03/11/0
10	Creación de la primera oleada de enem	16 horas	mar 03/11/0	mar 10/11/0
11	Implementar los disparos	6 horas	mar 10/11/0	vie 13/11/0
12	Detectar colisiones	4 horas	lun 01/03/1	mar 02/03/1
13	Diseño gráfico	10 horas	mié 03/03/1	dom 07/03/1
14	Test y pruebas	15 horas	dom 07/03/1	sáb 13/03/1
15	Entrevista con el director de proyecto	3 horas	dom 14/03/1	dom 14/03/1
16	**Segunda Iteración**	**14 días**	**lun 15/03/1**	**dom 04/04/1**
17	Revisión de los requisitos	3 horas	lun 15/03/1	mar 16/03/1
18	Cambios de diseño	6 horas	mar 16/03/1	vie 19/03/1
19	Hacer aleatorios los primeros enemigos	5 horas	vie 19/03/1	dom 21/03/1
20	Crear la segunda oleada de enemigos	10 horas	dom 21/03/1	jue 25/03/1
21	Controlar los estados del juego	10 horas	vie 26/03/1	lun 29/03/1
22	Test y pruebas	10 horas	mar 30/03/1	sáb 03/04/1
23	Entrevista con el director de proyecto	3 horas	sáb 03/04/1	dom 04/04/1
24	**Tercera Iteración**	**30 días**	**dom 04/04/1**	**dom 16/05/1**
25	Revisión de los requisitos	3 horas	dom 04/04/1	lun 05/04/1
26	Cambios de diseño	5 horas	mar 06/04/1	jue 08/04/1
27	Crear el Menú principal	60 horas	jue 08/04/1	mar 04/05/1
28	Mejorar el sistema de colisiones	5 horas	mar 04/05/1	jue 06/05/1



Id	Nombre de tarea	Duración	Comienzo	Fin
29	Ajustar las oleadas de enemigos	4 horas	vie 07/05/1	sáb 08/05/1(
30	Añadir una tercera oleada de enemigos	6 horas	sáb 08/05/1(	lun 10/05/1
31	Test y pruebas	10 horas	mar 11/05/1	sáb 15/05/1(
32	Entrevista con el director de proyecto	3 horas	sáb 15/05/1(	dom 16/05/1
33	**Cuarta Iteración**	**23 días**	**dom 16/05/1**	**mié 16/06/1**
34	Revision de los requisitos	3 horas	dom 16/05/1	lun 17/05/1
35	Cambios de diseño	3 horas	mar 18/05/1	mié 19/05/1
36	Incluir las opciones de cambiar el idiom	20 horas	mié 19/05/1	vie 28/05/1
37	Insertar efectos de sonido	10 horas	vie 28/05/1	mar 01/06/1
38	Rediseñar el sistema de control del tiem	7 horas	mar 01/06/1	vie 04/06/1
39	Añadir la pausa en el juego	15 horas	sáb 05/06/1(	vie 11/06/1
40	Test y pruebas	10 horas	vie 11/06/1	mar 15/06/1
41	Entrevista con el director de proyecto	3 horas	mar 15/06/1	mié 16/06/1
42	**Quinta Iteración**	**29 días**	**jue 17/06/1**	**mié 28/07/1**
43	Revisión de los requisitos	5 horas	jue 17/06/1	sáb 19/06/1(
44	Cambios de diseño	5 horas	sáb 19/06/1(	dom 20/06/1
45	Crear el enemigo final	8 horas	lun 21/06/1	jue 24/06/1
46	Incluir 3 dificultades distintas	30 horas	vie 25/06/1	mié 07/07/1
47	Mejorar el apartado gráfico general	25 horas	jue 08/07/1	dom 18/07/1
48	Test y pruebas	20 horas	dom 18/07/1	mar 27/07/1
49	Entrevista con el director de proyecto	3 horas	mar 27/07/1	mié 28/07/1
50	**Sexta iteración**	**15 días**	**jue 29/07/1**	**jue 19/08/1**
51	Ajustar las dificultades	10 horas	jue 29/07/1	dom 01/08/1
52	Añadir animaciones	5 horas	lun 02/08/1	mié 04/08/1
53	Test y pruebas	30 horas	mié 04/08/1	mar 17/08/1
54	Entrevista con el director de proyecto	4 horas	mar 17/08/1	jue 19/08/1
55	Generación de documentos	35 horas	jue 19/08/1	vie 03/09/1
56	Entrega del proyecto	2 horas	sáb 04/09/1(	sáb 04/09/1(



6.3 Líneas de ampliación

Una vez alcanzado este punto tenemos que darnos cuenta de que el proyecto ha cumplido su propósito, pero que este proyecto se termina y empieza uno de mayor envergadura que utilizará este proyecto como base. Lo que ahora conocemos como Guitar Invaders pasará a ser solamente una pequeña etapa muy importante de lo que tiene que llegar a convertirse.

En este apartado no voy a hablar de posibles futuras mejoras sino de mejoras que se van a implementar pues, durante su creación, he tenido tiempo para poder ver el proyecto con la suficiente perspectiva como para saber que es un proyecto finalizado pero que también es el inicio de un proyecto de mayor alcance que durará mucho más tiempo pero que, a diferencia del actual, este nuevo proyecto no lo voy a realizar solo.

Esto es debido a que durante la realización de mi proyecto he despertado el interés de mucha gente que está dispuesta a trabajar tan duramente como yo para que logremos hacer una aplicación totalmente comercial y podamos venderla a través del servicio de internet de Xbox: Xbox Live.

En esta nueva aventura que dará comienzo al terminar mis estudios actuales se unirán a mí dos programadores, un diseñador, un mánager de grupos de música y varias personas que ayudaran a la traducción en diferentes idiomas como el euskera.

Con este grupo mucho mayor y mucho tiempo por delante se pretende entre otras cosas:

- Crear 40 niveles distintos con sus enemigos característicos y sus jefes.
- Añadirle funcionalidades extra a la nave utilizando los sensores de movimiento de la guitarra.
- Crear un modo de dos jugadores en la misma consola y vía online.
- Hacer un registro de máximas puntuaciones a nivel mundial.
- Traducir el juego al máximo de idiomas posibles.
- Ofrecer un sistema de recompensas para hazañas destacables mediante la personalización de la nave y sus armas.
- Crear objetos que ofrezcan mejoras durante la partida.
- Mejorar la calidad gráfica de todo el juego en general.
- Crear una música expresamente para este videojuego.
- Introducir animaciones, historia y varios personajes que lleven la trama.

- Implementar un sistema de guardado y carga de partidas.

Además llegado el momento de su publicación se tiene la intención de hacer publicidad en algunos medios especializados para dar a conocer este juego

6.4 Consideraciones personales

Durante varios años he cursado unos estudios que me han dado muchas herramientas y muchísimos conocimientos pero necesitaba de alguna manera poder comprobar mi potencial y poder poner en práctica todo aquello que había aprendido. Por este motivo decidí que el proyecto final de carrera iba a ser la prueba que determinaría lo mucho o poco que había aprendido durante la carrera.

Sinceramente no puedo estar más contento y orgulloso por el trabajo realizado.

He pretendido en todo momento convertir este proyecto en un objetivo personal más que una obligación y me he marcado en todo momento metas que me parecían inalcanzables pero he visto que gracias a la capacidad que he conseguido durante la carrera de aprender y afrontar retos, he sido capaz de ir superando uno a uno todos los objetivos que me marcaba e incluso me he sentido capaz de poner retos cada vez más difíciles.

Personalmente encuentro que este proyecto es el resultado no solamente de los cientos de horas invertidos en él, sino de todo el esfuerzo que he realizado durante la carrera y me siento muy satisfecho con el resultado.

Ahora mismo se que si le pongo empeño y muchas ganas a algo soy capaz de lograrlo y esta confianza y moral que he conseguido no la tenía cuando empecé el proyecto.

De todas formas es cierto que si ahora empezase el mismo proyecto de nuevo haría las cosas de forma distinta porque he aprendido muchas cosas y mis conocimientos actuales sobre la materia son muy superiores a los que tenía al inicio.

No obstante me di cuenta con bastante presteza de que este efecto se estaba dando en mí, es decir, a cada minuto que pasaba tenía un mayor conocimiento de la materia y conocía formas mejores y más eficaces de afrontar las situaciones que me encontraba y es por ello que decidí utilizar un método iterativo para poder aplicar estos conocimientos sobre el proyecto.

También me gustaría comentar que la inclusión de Betatesters en el desarrollo de la aplicación era algo que en un principio no se contemplaba y surgió de forma anecdótica puesto que algunas personas se mostraron curiosas ante mi proyecto final de carrera. No obstante creo que su implicación en este proyecto ha sido uno de los puntos claves para su éxito. Tener la oportunidad de que tu trabajo sea valorado por

otras personas y puedan darte consejos e ideas es algo que tiene una fuerza muy importante y que antes desconocía.

Por último añadir que este ha sido con diferencia el trabajo más duro y difícil que he realizado hasta la fecha y no todo ha salido como se esperaba en un inicio, prueba de ello es la planificación temporal que no se cumplió, pero ha sido el más gratificante y el que me hace sentir más orgulloso de todos. Han sido muchísimas horas de sacrificio y de dedicación pero tengo la satisfacción de haber comprobado que he sabido aprovechar estos años de estudios.

7 Bibliografía

e-academy. (s.f.). *MSDN*. Recuperado el 5 de Septiembre de 2010, de http://msdn30.e-academy.com/elms/Storefront/Home.aspx?campus=eunid_inform_sabadell

Microsoft. (2007). *XNA*. Recuperado el 3 de Septiembre de 2010, de http://www.xna.com/

Microsoft. (2010). *XNA Creators club online*. Recuperado el 3 de Septiembre de 2010, de http://creators.xna.com/es-ES

Nitschke, B. (2007). *Professional XNA Game Programming For Xbox 360 and Windows.* Indianapolis: Wiley Publishing.

8 Anexo I: Manual de usuario

En este anexo se pretende explicar qué pasos se deben seguir para poder tener una experiencia satisfactoria con Guitar Invaders.

Lo primero que necesitamos es una Xbox 360 que disponga del juego Guitar Invaders en su interior. Además de esto como se trata de un software basado en XNA Microsoft realizará una comprobación para ver si tenemos la licencia al día y por ello necesitamos tener esta licencia y disponer de una cuenta de Xbox Live para poder conectar la consola a internet y que se realice dicha comprobación.

Una vez dentro debemos dirigirnos a nuestra biblioteca de juegos y en juegos independientes aparecerá Guitar Invaders. Acto seguido se recomienda conectar la guitarra que vayamos a utilizar.

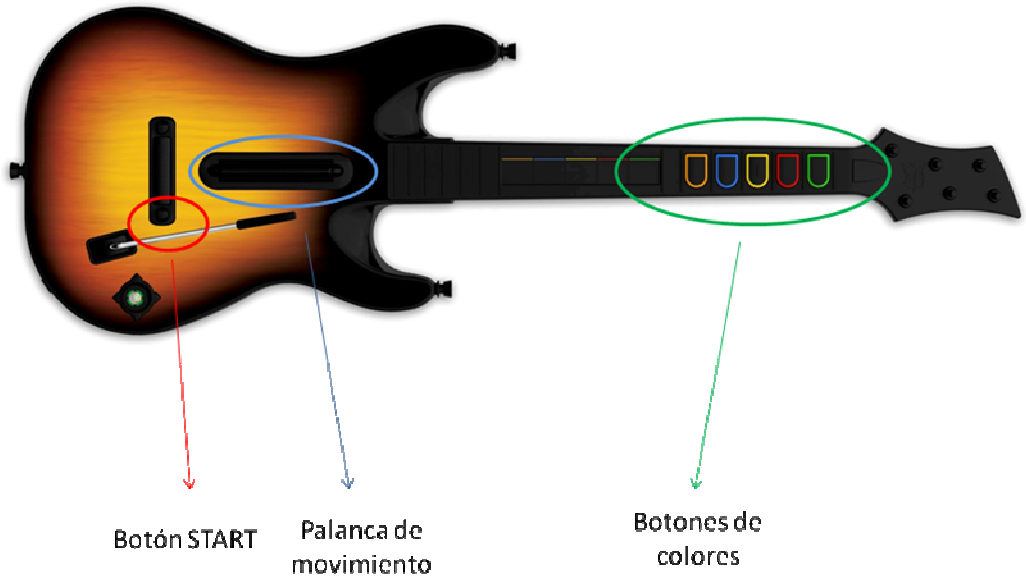


Figura 25 Guitarra

La imagen anterior nos muestra como es una guitarra de este tipo y se han señalado las partes importantes por tal de poder hacer servir Guitar Invaders con facilidad.

El Botón START es un botón que se utiliza al principio del juego en la pantalla de presentación y también durante el juego cuando deseemos poner la pausa.

La palanca de movimiento se mueve de forma vertical y nos permite subir y bajar tanto en las opciones del menú como con la nave dentro del videojuego.

En cuanto a los botones de colores, siguiendo el estándar de la mayoría de juegos, el botón verde será el que nos servirá para aceptar cualquier selección del menú.

Una vez dentro del juego cada uno de los botones de colores servirá para disparar un proyectil de su mismo color que tendremos que utilizar para matar los enemigos de los distintos colores, es decir, cada enemigo debe ser atacado con un proyectil de su mismo color para poder ser destruido.

En cuanto a la dificultad se recomienda empezar en modo fácil para aprender, porque aun así el juego tampoco es excesivamente fácil para evitar que pueda resultar aburrido. Una vez se domine ese nivel de dificultad ya se puede pasar a probar los otros dos que supondrán un reto bastante difícil que superar.

Jordi Martín Sánchez
Sabadell, Septiembre de 2010