

Universidad de las Ciencias Informáticas

Facultad-5



**Prototipo de videojuego de lucha para repositorio de
videojuegos del centro VERTEX.**

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor:

Yosmani Rafael Claro Merino

Tutores:

Ing. Roberto Elías Pérez Ozete

Ing. Victor Manuel Armas Pis

La Habana, diciembre de 2015

Año del 56 aniversario de la Revolución

DECLARACIÓN DE AUTORÍA

Declaro ser el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmo la presente a los _____ días del mes de diciembre del año 2015.

Autor: Yosmani Rafael Claro Merino

Tutor: Ing. Roberto Elías Pérez Ozete

Tutor: Ing. Victor Manuel Armas Pis

Datos de Contacto

Tutor: Roberto Elías Pérez Ozete

Categoría Científica: Ingeniero

Especialidad: Ingeniero en Ciencias Informáticas

Correo Electrónico: reperez@uci.cu

Categoría docente: -

Años de experiencia: 2

Años de graduado: 2

Tutor: Victor Manuel Armas Pis

Categoría Científica: Ingeniero

Especialidad: Ingeniero en Ciencias Informáticas

Correo Electrónico: vmarmas@uci.cu

Categoría docente: -

Años de experiencia: 2

Años de graduado: 2

Dedicatoria

Con cariño y afecto a mi familia y mi esposa.

Agradecimientos

De una forma u otra siempre es difícil hablar de agradecimientos en un trabajo como este en el que tantas personas están involucradas de cierta manera y siempre por mucho que uno lo no lo desee termina olvidando alguien.

Agradezco a todas esas personas que tuvieron que ver conmigo dentro y fuera del aula dígase estudiantes, profesores y todas las personas involucradas en todo el proceso educativo de mi formación desde la educación primaria hasta mi 5to año en la universidad. Queriendo destacar algunos de ellos los cuales siempre consideraré principales culpables de todo lo que he alcanzado hoy en día por mucho o poco que eso sea: la profesora Magalis que creó la base del estudiante que siempre me consideré, las profes Ivís Ochoa e Idalmis Matos las cuales siempre me transmitieron todo su afecto, mi profe guía durante todo el Politécnico Rubén Rowel Basiel quien me enseñó que no todo es sacar buenas notas y siempre mantuvo una conducta férrea sobre mí.

Dentro de la universidad el profe Dani (diría que es el mejor profe que he tenido por combinar muy bien tres cosas muy difíciles ser amigo, profe y joven), la profe Zoraida quien siempre consideré una excelente amiga pero ella se consideraba mi mamá en la universidad por siempre estar atenta de mí, a la rectora Miriam Nicado por darme la oportunidad de defender mi trabajo de diploma, al decano de la facultad 5 (Rigoberto) por la forma que siempre mantuvo conmigo, a la vicedecana de formación Yadira Ramírez por monitorearme y cerciorarse de que llegara hasta esta instancia y toda la ayuda que me brindó durante el proceso de tesis, a mis tutores Víctor y Roberto sin los cuales no hubiese conseguido terminar esta trabajo, por su amistad por encima de todo, por darme toda la confianza para desarrollar el trabajo de diploma y por las críticas constructivas y el rigor empleado durante todo el proceso, a mi bella esposa Malena Caro Sánchez la cual siempre confió en mí, me impulso y me apoyo incondicionalmente, a mi hermana Yohima Claro Merino por todo su apoyo y cariño, a mi hermano Yadian Claro Merino por el apoyo en varios aspectos y sobre todo a esas personas que sin ellas ni siquiera pudiera haber existido: mi Madre Neyda Isabel Merino Hernández y mi Padre Oreste Claro Vicente por ser las personas que sin importar nada ni nadie siempre me han dado su apoyo incondicional en todas esas decisiones malas y buenas que he tomado, por el cariño y amor inmenso que siempre me han profesado estoy y estaré siempre muy agradecido.

Resumen

El auge de las Tecnologías de la Información y las Comunicaciones trajo consigo cambios en la forma que el hombre interactúa con sus semejantes. Las nuevas tecnologías han logrado llevar al mundo virtual los juegos tradicionales que tanto disfrutaban los niños, dando paso a la era de los videojuegos.

La presente investigación se centra en el proceso de desarrollo de un prototipo funcional de videojuego de lucha que permita la futura implementación de diferentes videojuegos de este género. Se decidió utilizar como metodología de desarrollo de software *Extreme Programming* (XP) y *Huddle*, así como usar como herramienta principal el motor de videojuegos Unity3D, el cual demostró ser una buena alternativa para la creación de soluciones de estas características.

Como resultado principal se obtiene un prototipo funcional de videojuego de lucha el cual posee las características fundamentales de este género, entre las se puede mencionar el uso de diferentes personajes que poseen característica de lucha diferentes, el uso de combinaciones de tecla para realizar diferentes ataques al oponente, así como el uso de técnicas de Inteligencia Artificial para definir el comportamiento de los oponentes en una pelea.

Palabras Clave: prototipo funcional, videojuego, repositorio, Unity3D, proceso de desarrollo

Índice

Dedicatoria	4
Agradecimientos	5
Resumen.....	6
Índice	7
Índice de tablas.....	9
Índice de figuras	12
Introducción.....	13
Capítulo 1: Fundamentación Teórica.....	16
1.1 Introducción	16
1.2 Videojuegos	16
1.3 Géneros de videojuegos.....	16
1.4 Videojuegos de lucha	18
1.4.1 Principales subgéneros.....	19
1.4.2 Análisis de videojuegos de lucha 1 vs 1	20
1.5 Metodología de desarrollo de software	23
1.5.1 Huddle	25
1.5.2 Programación Extrema (Extreme Programming, XP)	26
1.5.3 Selección de la metodología a utilizar	27
1.6 Herramientas utilizadas	28
1.6.1 Gráficos 2D	28
1.6.2 Gráficos 3D	28
1.6.3 Motores de videojuegos	28
1.7 Conclusiones parciales del capítulo	30

Capítulo 2: Solución propuesta	31
2.1 Introducción	31
2.2 Documento de diseño	31
2.3 Fase de exploración	36
2.3.1 Historias de usuario	36
2.3.2 Requisitos no funcionales	42
2.4 Planificación	43
2.4.1 Estimación de esfuerzos por historias de usuario	43
2.5 Iteraciones	44
2.6 Diseño de la solución propuesta	45
2.6.1 Modelo de dominio	45
2.6.2 Tarjetas CRC	48
2.6.3 Componentes	51
2.7 Scripts	54
Capítulo 3: Implementación y prueba	56
3.1 Introducción	56
3.2 Fase de Implementación	56
3.2.1 Iteración 1	56
3.2.2 Iteración 2	58
3.2.3 Iteración 3	60
3.3 Pruebas	62
3.3.1 Pruebas de aceptación	63
3.4 Conclusiones parciales	68
Conclusiones	69
Recomendaciones	70
Referencias bibliográficas	71

Índice de tablas

Tabla 1: Metodologías de desarrollo de software analizadas.	25
Tabla 2: Principales características de Unity3D.	30
Tabla 3: Descripción de las Historias de usuario (HU).	37
Tabla 4: HU Mostrar interfaz de usuario.	37
Tabla 5: HU Seleccionar personaje.	38
Tabla 6: HU Cerrar proceso correctamente.	38
Tabla 7: HU Iniciar combate.	38
Tabla 8: HU Finalizar combate.	39
Tabla 9: HU Activar cambio de escena.	39
Tabla 10: HU Activar botones de acción.	39
Tabla 11: HU Activar animaciones según correspondencia de acción.	40
Tabla 12: HU Ejecutar ataque básico (jugador).	40
Tabla 13: HU Ejecutar combo de golpes (jugador).	41
Tabla 14: HU Ejecutar ataque básico (oponente).	41
Tabla 15: HU Ejecutar combo de golpes (oponente).	41
Tabla 16: HU Añadir sistema de colisiones.	42
Tabla 17: HU Activar efectos de sonido.	42
Tabla 18: HU Activar efectos especiales.	42
Tabla 19: Estimación de esfuerzos por historias de usuario.	44
Tabla 20: Plan de duración de las iteraciones.	45
Tabla 21: Prototipo de una tarjeta CRC.	48
Tabla 22: Tarjeta CRC para Canvas.	48
Tabla 23: Tarjeta CRC para Panel_Menu.	48
Tabla 24: Tarjeta CRC para Panel_Seleccion.	49
Tabla 25: Tarjeta CRC para EvenSystem.	49
Tabla 26: Tarjeta CRC para Canvas.	49
Tabla 27: Tarjeta CRC para Barras.	49
Tabla 28: Tarjeta CRC para Perdido.	49
Tabla 29: Tarjeta CRC para Ganado.	50
Tabla 30: Tarjeta CRC para GameManager.	50

Tabla 31: Tarjeta CRC para Player.....	50
Tabla 32: Tarjeta CRC para IA_Enemy.....	50
Tabla 33: Tarjeta CRC para TriggerDetect.....	50
Tabla 34: Tarjeta CRC para Camera.	51
Tabla 35: Tiempo de implementación de la Iteración 1.	56
Tabla 36: Diseñar e implementar interfaz de usuario del juego.....	57
Tabla 37: Seleccionar personaje.....	57
Tabla 38: Cerrar proceso correctamente.....	57
Tabla 39: Iniciar combate.....	58
Tabla 40: Finalizar combate.....	58
Tabla 41: Tiempo de implementación de la Iteración 2.	58
Tabla 42: Activar cambio de escena.	59
Tabla 43: Activar botones de acción.	59
Tabla 44: Activar animaciones.	59
Tabla 45: Ejecutar ataque básico (jugador).....	60
Tabla 46: Ejecutar combo de golpes (jugador).....	60
Tabla 47: Tiempo de implementación de la Iteración 2.	60
Tabla 48: Ejecutar ataque básico (oponente).....	61
Tabla 49: Ejecutar combo de golpes (oponente).....	61
Tabla 50: Añadir sistema de colisiones.....	61
Tabla 51: Activar efectos de sonido.	62
Tabla 52: Activar efectos especiales.....	62
Tabla 53: Caso de prueba de Aceptación HU1_P1.....	63
Tabla 54: Caso de prueba de Aceptación HU1_P2.....	63
Tabla 55: Caso de prueba de Aceptación HU2_P1.....	64
Tabla 56: Caso de prueba de Aceptación HU3_P1.....	64
Tabla 57: Caso de prueba de Aceptación HU4_P1.....	64
Tabla 58: Caso de prueba de Aceptación HU5_P1.....	65
Tabla 59: Caso de prueba de Aceptación HU6_P1.....	65
Tabla 60: Caso de prueba de Aceptación HU7_P1.....	65
Tabla 61: Caso de prueba de Aceptación HU8_P1.....	66
Tabla 62: Caso de prueba de Aceptación HU9_P1.....	66
Tabla 63: Caso de prueba de Aceptación HU10_P1.....	66
Tabla 64: Caso de prueba de Aceptación HU11_P1.....	67

Tabla 65: Caso de prueba de Aceptación HU12_P1.....	67
Tabla 66: Caso de prueba de Aceptación HU13_P1.....	67
Tabla 67: Caso de prueba de Aceptación HU14_P1.....	68
Tabla 68: Caso de prueba de Aceptación HU15_P1.....	68

Índice de figuras

Figura 1: Mortal Kombat.	19
Figura 2: Power Stone 2.	20
Figura 3: Primer videojuego de lucha: Warrior.	21
Figura 4: Karate Champ.....	21
Figura 5: Street Fighter II.	22
Figura 6: Fases del proceso Huddle.	26
Figura 7: Estados del juego.	32
Figura 8: Menú principal.	33
Figura 9: Selección de personajes.	33
Figura 10: Progreso del juego.	34
Figura 11: Ninja.	35
Figura 12: Bruto.	35
Figura 13: Relación entre el núcleo de Unity y los objetos que componen la escena.....	46
Figura 14: Modelo de dominio de Menú Principal.	46
Figura 15: Modelo de dominio de la escena principal.....	47
Figura 16: Componente Camera.....	51
Figura 17: Componente Audio Source.	52
Figura 18: Componente Rigidbody.....	53
Figura 19: Animator y Animator Controller en acción.	54

Introducción

El juego constituye un elemento básico en la vida de un niño, que además de divertido resulta necesario para su desarrollo. Es el medio natural de autoexpresión y lo usa para recrear lo vivido, exteriorizar sus emociones, sentimientos, frustraciones y fantasías (Fachin, 2009).

El auge de las Tecnologías de la Información y las Comunicaciones trajo consigo cambios en la forma que el hombre interactúa con sus semejantes. Las nuevas tecnologías han logrado llevar al mundo virtual los juegos tradicionales que tanto disfrutaban los niños, dando paso a la era de los videojuegos.

El fenómeno del videojuego se ha convertido en un elemento cotidiano en la actualidad. La implantación de los videojuegos como elementos generalizados de ocio en la sociedad actual ha convertido a este medio en uno de los referentes de transmisión del conocimiento con mayor potencial.

Los videojuegos fomentan la reflexión, la concentración y el razonamiento estratégico. Se relaciona a ellos el desarrollo general de algunos tipos de reflejos y aumentan los niveles de agilidad mental. Estas habilidades podrían contextualizarse dentro de las llamadas competencias espaciales, destrezas de representación espacial implicadas en el procesamiento de palabras, que incluyen la comprensión de la naturaleza secuencial del texto y la conexión entre porciones visibles del texto (Sedeño, 2010).

En nuestro país, a pesar de poseer un desarrollo incipiente en este sector, se están dando pasos en aras de potenciar la creación y comercialización de videojuegos. Un centro pionero en el desarrollo de aplicaciones que se ajustan a esta temática es la Universidad de las Ciencias Informáticas, específicamente el centro de Entornos Interactivos 3D (VERTEX).

Desde sus inicios, en dicho centro se han desarrollado diversos videojuegos entre los que se puede mencionar Energía para Aprender, Rápido y Curioso, Meteorix y Aventura Anatómica. Más recientemente se desarrollaron los videojuegos Chivichana y Superclaria, resultado de la alianza entre el centro y los Estudios de Animación del ICAIC.

Ante las crecientes posibilidades de creación de nuevos videojuegos, surge la idea de crear un repositorio de prototipos funcionales de videojuegos. Dicho repositorio contendrá los elementos bases de cada género de videojuego para a partir de estos realizar su implementación.

De lo anteriormente expresado se define que, los prototipos funcionales que se proponen contendrán los elementos que son comunes a todos los juegos que sobre un género se puedan hacer.

Uno de los prototipos que se necesita implementar es un prototipo de videojuego de lucha debido a que en dicho centro aún no se ha desarrollado un videojuego que aborde esta temática. Entre los elementos que conforman este tipo de videojuego se encuentran los personajes tridimensionales (3D) con diferentes características y que posean diferentes habilidades a la hora del combate. Estos juegos se desarrollan desde cero, debido a que no existen prototipos base que faciliten el proceso de producción.

Es por eso que se plantea como **problema a resolver**: ¿Cómo definir el proceso de desarrollo de videojuegos de lucha?

Se define como **objetivo general** desarrollar un prototipo de videojuego de lucha.

Como **objeto de estudio** se toma el desarrollo de videojuegos y como **campo de acción** el desarrollo de videojuegos de lucha.

Para darle cumplimiento al objetivo planteado se proponen las siguientes **tareas de investigación**:

- Caracterización de los fundamentos teóricos generales que sustentan el objeto de la investigación.
- Análisis del estado del arte, las herramientas y metodologías para la realización de un videojuego de lucha.
- Desarrollo de los artefactos ingenieriles necesarios para el desarrollo de la solución según la metodología seleccionada.
- Implementación de un prototipo de videojuego de lucha.
- Aplicación de pruebas a la solución desarrollada.

Métodos teóricos

Histórico - Lógico: se empleó para la fundamentación y sistematización de los aspectos teóricos contemplados en el desarrollo de la investigación acerca de los videojuegos, las metodologías a utilizar, así como las características que posee el motor de videojuegos Unity 3D.

Analítico - Sintético: durante el proceso de investigación permitió descubrir los elementos esenciales, concepciones y conceptos en torno al objeto de investigación. Entre los conceptos fundamentales definidos está el de videojuego y el de videojuego de lucha.

Métodos Empíricos

Observación: Se empleó como método referencial al observar distintos videojuegos que sirvieron como objeto de análisis y comparación para establecer las características y elementos fundamentales que debía cumplir la propuesta que plantea el autor. El principal ejemplo que se analizó fue el videojuego *Street Fighter II*, el cual sirvió de referencia a la hora de definir las características básicas que debía tener un videojuego de lucha.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En el presente capítulo se presenta un estudio sobre las características fundamentales de los videojuegos, los distintos géneros que existen, haciendo análisis más profundo en los videojuegos de lucha. Además, se hace referencia a las tecnologías, herramientas y metodología a utilizar en el proceso de desarrollo de la solución.

1.2 Videojuegos

Un videojuego o juego de vídeo es un software creado para el entretenimiento en general y basado en la interacción entre una o varias personas y un aparato electrónico que ejecuta dicho videojuego; este dispositivo electrónico puede ser una computadora, una máquina *arcade*, una videoconsola, un dispositivo móvil, entre otros (ProChile, 2012).

Algunos de los conceptos que se pueden encontrar dicen que “un videojuego consiste en un entorno informático que reproduce sobre una pantalla un juego cuyas reglas han sido previamente programadas” (Levis, 1997).

Por otra parte, Marqués lo define como todo tipo de juego electrónico interactivo que oferta una serie de actividades lúdicas (contenido), cuyo punto de apoyo común es el medio que se utiliza (plataforma electrónica), con independencia de su soporte (ROM interno, cartucho, disco magnético u óptico, on-line) y de la plataforma tecnológica que utiliza (máquina de bolsillo, videoconsola conectable al TV, máquina recreativa, microordenador, video interactivo, red telemática, teléfono móvil). (Marqués, 2000)

De manera general podemos plantear que un videojuego es aquel programa informático, normalmente asociado a un hardware específico, que recrea un ejercicio sometido a reglas, donde se deben lograr uno o varios objetivos, y es la interacción de los jugadores mediante la toma de decisiones la que permite que se logren o no los objetivos antes mencionados.

1.3 Géneros de videojuegos

Los géneros de videojuegos son las categorías que se utilizan para clasificar y organizar a los videojuegos por elementos clave de su jugabilidad, tales como la forma de juego, los controles y el objetivo. A diferencia de otros medios como la literatura o el cine, los géneros de los videojuegos no toman en cuenta aspectos relacionados al argumento o la temática situacional de las historias presentadas.

La clasificación por géneros es un sistema que se basa en las coincidencias que existen entre distintos videojuegos por su jugabilidad básica, esta se ha ido formando a medida que fueron surgiendo numerosos juegos con características en común que permitían agruparlos. De esta manera, los videojuegos con una forma de juego parecida constituyen géneros que se pueden dividir nuevamente en subgéneros cada vez más específicos. Un videojuego no necesariamente debe pertenecer a un único género, muchos de estos videojuegos presentan elementos combinados de dos o más géneros distintos, en estos casos se considera al género que predomina para categorizarlo.

Las clasificaciones más aceptadas son (I Gavalda, 2008):

- **Aventura:** Los juegos de aventura son aquéllos en los que se deben realizar diversas acciones a medida que se desarrolla el discurso narrativo hasta alcanzar un objetivo final. También existe un dinamismo en la evolución de la trama.
- **Deportivos:** Son aquellos que simulan juegos de deporte real; entre ellos encontramos: golf, tenis, fútbol, hockey, juegos olímpicos, entre otros.
- **Disparos:** Son aquellos en los que el jugador asume el rol de una persona, vehículo o nave y el objetivo principal es atacar a objetivos diversos disparándoles proyectiles.
- **Educativos:** Son videojuegos que educan al jugador y están destinados mayoritariamente a un público infantil y juvenil. Normalmente están pensados para enseñar algún aspecto específico mediante el uso de mini-juegos diseñados por pedagogos y especialistas.
- **Lucha:** Son todos aquellos que incluyen combates directos entre personajes, usando los puños o armas. El objetivo es siempre atacar al rival hasta lograr vencerlo.
- **Rompecabezas:** Conocidos también como videojuegos de lógica o de inteligencia. Este género incluye a los juegos en donde cada nivel se presenta como una situación problemática al jugador, que este debe resolver siguiendo las reglas impuestas mediante el uso de su razonamiento lógico.
- **Rol:** Popularmente conocidos como **RPG**, este género está basado en los juegos de rol de mesa. Por lo general, el jugador inicia una partida en la que escoge un equipo de personajes, cada uno con sus propias características y habilidades, expresadas en forma de datos estadísticos (salud, magia, ataque, defensa, sabiduría). El juego en general se divide en explorar pueblos, en donde el equipo puede abastecerse y comprar distintos objetos; y en explorar mazmorras o calabozos, que son escenarios laberínticos

repletos de enemigos en donde los personajes combaten para ganar experiencia y dinero y así se van fortaleciendo.

- **Simulación:** Este género agrupa a los juegos que simulan una realidad determinada que puede ser semejante a la vida real o presentar un mundo ficticio en donde el jugador interactúa con el entorno haciendo uso de recursos y bienes.
- **Estrategia:** Son aquellos en donde la forma de juego requiere el uso de un pensamiento táctico y la planificación de acciones para alcanzar la victoria. Estos juegos suelen caracterizarse por dar una gran libertad al jugador para diseñar su camino a seguir.
- **Carreras:** Son aquellos en donde el jugador controla a un personaje o vehículo que compite en una carrera contra otros vehículos a lo largo de una pista.
- **Arcade:** Llamados así porque mantienen los principios de los antiguos juegos de *Arcade*, basados en el entretenimiento inmediato y una jugabilidad sencilla y adictiva. Su presencia está muy extendida en los sistemas móviles. Estos videojuegos presentan controles muy simples, niveles de corta duración y los objetivos son fáciles de comprender al instante. Se caracterizan por tener una gran cantidad de niveles cortos en donde la dificultad va aumentando rápidamente al progresar.
- **Nuevas tendencias:** esta clasificación evoluciona a medida que salen nuevas propuestas, fusionando géneros y creando nuevos tipos de juegos. Un ejemplo de una nueva categoría de juegos lo podemos encontrar en los *web games*.

1.4 Videojuegos de lucha

Son juegos de acción donde el jugador se enfrenta en combates contra otros personajes, controlados por la máquina o por otro jugador humano.

El combate se efectúa mayoritariamente mediante el uso de técnicas especiales, que se activan mediante combinaciones de botones con el mando de dirección, lo que implica normalmente una curva de aprendizaje más costosa de lo normal.

En sus inicios la mayoría de estos juegos se desarrollaban en una perspectiva lateral, aspecto que todavía se conserva a día de hoy, usando las flechas del teclado o *joystick* para desplazarse en dichas direcciones, y las flechas de arriba y abajo para saltar y agacharse respectivamente. El resto de combinaciones, con o sin los botones de disparo, dan paso a una serie de golpes y movimientos que varían de un juego a otro.

Si bien casi todos disponen de diferentes tipos de golpes, es un denominador común de este tipo de juegos que los personajes emulen artes de lucha milenarios o fantásticos que tienen como premisa común que emplean sus características físicas para poder aplicarlos.

1.4.1 Principales subgéneros

Lucha 1 vs 1

Consiste en pelear contra otro luchador, manejado por la máquina o por otro jugador, donde la finalidad es derrotarlo o evitar ser derrotado, aplicando técnicas de lucha cuerpo a cuerpo para atacar y contraatacar.



Figura 1: Mortal Kombat.

Los combates suelen ser regidos por asaltos y estos terminan cuando se vence el tiempo o uno de los luchadores pierde la vida, la cual es señalizada por una barra que va disminuyendo según vaya recibiendo daño el luchador. El término vida puede definirse como un valor numérico que disminuye en dependencia de las acciones que realiza el oponente sobre el personaje con que se juega.

Ejemplos: *Street Fighter II*, *Tekken*, *Samurai Showdown*, *Mortal Kombat*.

Lucha *Free-for-All*

Juego de lucha que el jugador debe enfrentarse a dos o más oponentes y que se desarrolla en escenarios más elaborados, en donde los competidores pueden moverse libremente, y usar los elementos y objetos presentes como parte de su estrategia.



Figura 2: Power Stone 2.

En este tipo de juegos usualmente el ganador es el último que sobrevive a la pelea. Sin embargo, puede haber otros modos de determinar al ganador como su ratio de eliminaciones por tiempo, o puntos de acuerdo al desempeño en el combate, entre otros.

Ejemplos: *Power Stone 2*, *Super Smash*.

1.4.2 Análisis de videojuegos de lucha 1 vs 1

Para conocer los inicios de los juegos de lucha tendremos que remontarnos al año 1979. En este año fue publicado el primer título catalogado como un juego de peleas: *Warrior*. Fue programado por *Tim Skelly*, un desarrollador que trabajaba para la empresa *Cinematronics*, y publicado bajo el sello de *Vectorbeam*. En este juego se enfrentaban cara a cara dos imágenes vectorizadas¹ que intentaban dar la apariencia de ser caballeros con armadura. El objetivo era vencer al rival mediante el uso de espadas. Las limitaciones tecnológicas impidieron que el juego tuviera fondos o algún otro elemento característico de los juegos actuales (MeriStation, 2009).

¹ Imagen digital formada por objetos geométricos independientes (segmentos, polígonos, arcos, etc.), cada uno de ellos definido por distintos atributos matemáticos de forma, de posición, de color, etc.

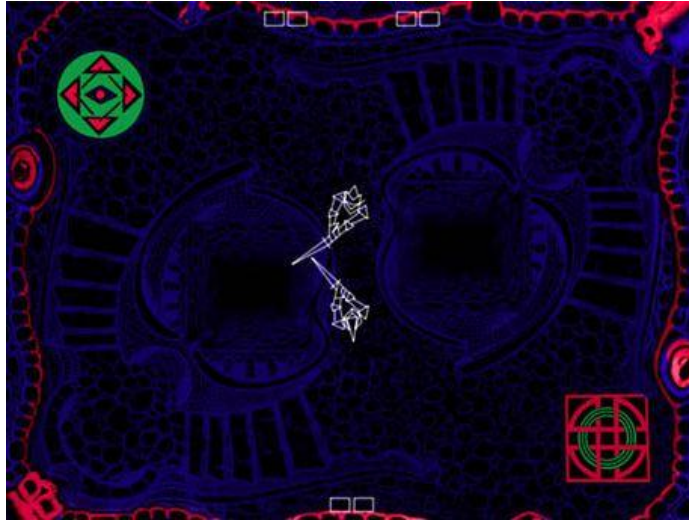


Figura 3: Primer videojuego de lucha: Warrior.

En 1984 la compañía *Technos* junto a *Data East* lanzó el juego *Karate Champ*. Este juego fue pionero en utilizar una perspectiva lateral y dividió las peleas en puntos o *rounds*. También enfrentó a dos personajes manejados por usuarios y propuso combinaciones de palancas para ejecutar ataques, elemento que se convertiría en estándar del género (VicbenGames, 2011).



Figura 4: Karate Champ.

En 1985 se lanza el primer juego que incorporaba una barra de vida: *Shanghai Kid*. En este juego se ejecutan movimientos concretos de ataque o defensa para vencer al rival, además es el primer juego en el que se pueden ejecutar *combos* de golpes (VicbenGames, 2011).

Es entonces, que en 1987 sale a la luz un proyecto conocido como *Street Fighter*. Diseñado por *Takashi Nishiyama* y *Hiroshi Matsumoto* para la compañía japonesa *Capcom*. El juego debutó en los *arcades* de Japón en agosto, colocándose rápidamente como un título fuera de lo normal,

y, en muchos sentidos, si era un concepto lleno de innovaciones: combinaciones de palanca y botones, ataques especiales, personalidad en los enemigos y una modalidad para que cualquier humano pudiera desafiarte para luchar uno a uno en cualquier momento (MeriStation, 2009).

Con el lanzamiento de *Street Fighter*, *Capcom* inauguró una de las franquicias de videojuegos más exitosas, y rentables, de todos los tiempos. Aquel título, el primero de la historia en introducir los ataques especiales y los combos de golpes, marcó el estándar en el sector que seguirían otros juegos como *Fatal Fury* (1991), *Mortal Kombat* (1992) o *Tekken* (1994).

Cuatro años después del estreno de la recreativa original llegó *Street Fighter II*, que supuso el verdadero despegue de la saga. Este juego vendió más de medio millón de unidades en todo el mundo, lo que supuso unos ingresos para *Capcom* cercanos a los 1.500 millones de dólares.

Además, su éxito fue aún más allá y convirtió el videojuego en parte de la cultura popular, una autentica referencia para toda una generación de jóvenes y adolescentes.

Street Fighter II ostenta aún los record *Guinness* de “máquina recreativa más vendida” y “juego de lucha más copiado de la historia”. Además, su éxito inspiró numerosas secuelas, casi una veintena, series de televisión e incluso dos películas de *Hollywood*.

Street Fighter II

Desde 1984 que salió al mercado el primer videojuego de este género varios fueron los títulos que aparecieron y aportaron nuevos elementos técnicos y artísticos. Pero el juego que realmente revolucionó el género fue lanzado en 1991 y se tituló *Street Fighter II: The World Warrior*. Creado por la compañía japonesa *Capcom*, es considerado la obra maestra que sentó las bases de cómo debe ser un juego de lucha en 2D.



Figura 5: *Street Fighter II*.

Cuenta con 8 personajes a elegir, 4 jefes finales y un final diferente por cada personaje. A su vez, poseía un control que utiliza combinaciones de la palanca y 6 botones para ejecutar ataques especiales durante la pelea.

Apareció en los *arcade* en 1991 en Japón, y enseguida para el resto del mundo. Su primera conversión para consolas domésticas fue para *Super Nintendo* (1992), saliendo más tarde versiones en otras plataformas como *Turbografx* y *Sega Mega Drive* en 1993.

En el torneo para un jugador, el jugador se enfrenta contra los otros siete luchadores antes de proceder a combatir contra los oponentes finales, los cuales son cuatro personajes no seleccionables que son controlados por la máquina, conocidos como los "Grandes Maestros".

Street Fighter II fue un hito en la historia de los videojuegos por muchos aspectos:

- Es la primera vez que se podía jugar con todos los personajes involucrados, sean "buenos" o "malos" en personalidad, exceptuando los cuatro jefes finales.
- Cada personaje era distinto en poderes, velocidad, técnicas de combate y cada uno tenía su propia historia. Algo inédito hasta ese momento.
- Gráficos extraordinarios para la época y un número impresionante de voces.

El control de los personajes (movimientos, técnicas) era de una precisión nunca antes vista hasta la fecha en los juegos de lucha.

1.5 Metodología de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de técnicas, procedimientos y herramientas que ayudan a los desarrolladores a realizar un nuevo producto. Indican paso a paso todas las actividades a realizar para lograr el producto informático deseado, señalan además el papel que tendrán las personas que estarán presentes en el desarrollo de las actividades (Inteco, 2009). Estas metodologías pueden clasificarse, a grandes rasgos, en metodologías ágiles y tradicionales.

La tendencia a utilizar metodologías ágiles para videojuegos tomó fuerza en los últimos años por existir varios casos de empresas en esta industria que logran adaptar estas metodologías. (Keith, 2009)

Entre las empresas con casos de éxito documentados se encuentran *Large Animal Games* (Tobey, 2008), *Crytek* (Crytek, 2008) y *DICE* (Nutt, 2008) que utilizan *Scrum*; *Titus Interactive Studios* que utilizó XP (Beck, 2004) y *High Moon Studios* (Keith, 2009) que utiliza ambas.

A pesar de los beneficios que reportan, ninguna de estas adaptaciones está especificada formal y públicamente.

A continuación, se muestra una tabla comparativa donde analizan los elementos de algunas de las metodologías más usadas para el desarrollo de software:

Metodología	Descripción	Valoración
Modelo en Cascada	Proceso de desarrollo de software especializado secuencial en el cual el desarrollo se basa en el modelo Cascada a través de las fases de concepción, iniciación, análisis, diseño, construcción y pruebas. (Flood, 2003)	El producto final se demora más de lo esperado ya que cualquier problema que se presente en una de las etapas se tiene que regresar a una anterior para corregirlo. Requiere hacer muchos cambios a los documentos y regresar a etapas anteriores propicia a que se vuelva un proceso muy desordenado. Sin embargo es uno de los procesos más populares en la industria de los videojuegos.
<i>Rational Unified Process</i>	Es un proceso de desarrollo de software iterativo, es adaptable y entallable para satisfacer las necesidades del equipo del proyecto. (Kruchten, 2004)	Proceso muy completo, pero no está enfocado al desarrollo de videojuegos. Demasiada documentación permite realizar un buen producto con lo que ello conlleva (tiempo, dinero y personas involucradas). Sin embargo esta metodología genera documentación innecesaria para el desarrollo de un videojuego.
<i>Scrum</i>	<i>Scrum</i> es un <i>framework</i> de desarrollo de software iterativo-incremental utilizado en el desarrollo de software ágil. El trabajo está estructurado en ciclos conocidos como <i>sprints</i> .	<i>Scrum</i> facilita la iteración, permite a los equipos entregar características pulidas para probar la calidad del juego a lo largo de su desarrollo y así incorporar la retroalimentación de jugadores.

	<p>Durante cada sprint los equipos toman los requerimientos de una lista ordenada por prioridades conocidas como historias de usuario. Al terminar cada sprint, se tiene una versión potencialmente final del producto. (Scrum Alliance, 2009)</p>	<p><i>Scrum</i> no es solo para programadores, involucra a muchas personas a un solo proyecto. Es útil debido a que los videojuegos hoy en día se vuelven más complejos e involucran a personas multidisciplinarias.</p>
--	--	--

Tabla 1: Metodologías de desarrollo de software analizadas.

Estas metodologías aportan ventajas importantes para el proceso de desarrollo de un videojuego, pero no son las adecuadas para este. La principal razón es que estas metodologías se centran en el desarrollo de aplicaciones de carácter general y no en el proceso creativo que existe detrás del desarrollo de un videojuego. Este tipo de aplicaciones, además de presentar similitudes a programas de ordenador, presentan características muy particulares del mundo de la producción audiovisual que estas metodologías no soportan.

1.5.1 Huddle

Pese a no existir una metodología estándar para el desarrollo de videojuegos, si existen metodologías que específicas para este tipo de proyectos. Tal es el caso de la metodología *Huddle*.

El término *Huddle* hace referencia a la reunión que se realiza en el juego antes de cada jugada en el fútbol americano. La filosofía es que, mediante breves reuniones de planeación a corto plazo, se planea cada “jugada” que se inicie; con esto se da un seguimiento más estrecho al avance del proyecto y es posible hacer correcciones tempranas a posibles desviaciones (Urrutia, 2010).

Huddle es un proceso específico para desarrollo de videojuegos con las siguientes características (Rayo, 2011):

- Metodología ágil.
- Óptimo para equipos multidisciplinarios de 5 a 10 personas.
- Proceso iterativo, incremental y evolutivo.

Todo el proceso se divide en 3 fases (Rayo, 2011):

- Preproducción

- Producción.
- Postmortem.



Figura 6: Fases del proceso Huddle.

1.5.2 Programación Extrema (*Extreme Programming, XP*)

XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico (Letelier, 2006).

Algunas de las principales características de XP son:

- Pocos artefactos.
- Pocos Roles.
- El cliente es parte del equipo de desarrollo.
- Aplicable en grupos pequeños, menos de 10 integrantes y trabajando en el mismo sitio.
- Desarrollo iterativo e incremental.
- Pruebas unitarias continuas, frecuentemente repetidas y automatizadas.
- Programación en parejas.
- Corrección de todos los errores antes de añadir nuevas funcionalidades.
- Refactorización del código.
- Propiedad del código compartida.

- Simplicidad en el código.

El ciclo de vida ideal de XP consiste de seis fases (Beck, 1999):

- Fase I: Exploración
- Fase II: Planificación de la Entrega
- Fase III: Iteraciones
- Fase IV: Producción
- Fase V: Mantenimiento
- Fase VI: Muerte del Proyecto

1.5.3 Selección de la metodología a utilizar

No existe una metodología universal para hacer frente con éxito a cualquier proyecto de desarrollo de software. Toda metodología debe ser adaptada al contexto del proyecto (recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, etc.). En el caso particular de la presente investigación se decide usar las metodologías *Huddle* y XP para obtener una solución de la manera más óptima posible.

Ambas metodologías poseen características similares debido a que son metodologías ágiles, especialmente diseñadas para equipos multidisciplinarios integrados por no más de 10 personas y que son procesos iterativos e incrementales.

XP es una metodología adaptable al software a desarrollar, así como a las condiciones de trabajo. Es flexible cuando los requerimientos sufren cambios a menudo y permite administrarlos de forma óptima. También XP reduce el tiempo entre una idea, su criterio de validación y su implementación.

Huddle permite definir elementos técnicos y artísticos propios de un videojuego mediante el artefacto Documento de diseño, lo cual no es posible si se utiliza solamente una metodología para el desarrollo de software de carácter general. No obstante se considera que no permite describir de manera clara los elementos referentes a los requisitos del juego a desarrollar.

Es por eso que se propone utilizar la metodología XP para el desarrollo ingenieril de la solución y específicamente el artefacto Documento de diseño de la metodología *Huddle* debido a que este expresa formalmente la idea principal y detalles de la propuesta de videojuego.

1.6 Herramientas utilizadas

1.6.1 Gráficos 2D

Adobe Photoshop CS6

Photoshop CS6 es un editor de gráficos rasterizados desarrollado por *Adobe Systems Incorporated*. Usado principalmente para el retoque de fotografías y gráficos 2D.

Algunas de sus principales características son:

- Motor de gráficos *Mercury Graphics Engine*.
- Capas mejoradas.
- Capaz de detectar caras y seleccionar el tono de la piel.
- Edición de vídeo.
- Las herramientas de Recortar han sido rediseñadas.
- Filtro de ángulo ancho para corregir imágenes automáticamente.
- Presenta opciones de auto ajuste.
- Máscaras vectoriales para crear objetos vectoriales.
- Cámara RAW7.

1.6.2 Gráficos 3D

Autodesk 3ds Max

Es un programa de creación de gráficos y animación 3D desarrollado por *Autodesk*. Con su arquitectura basada en *plugins*, es uno de los programas de animación 3D más utilizado, especialmente para la creación de videojuegos, anuncios de televisión, en arquitectura o en películas.

1.6.3 Motores de videojuegos

Un motor de videojuego es un término que hace referencia a una serie de rutinas de programación que permiten el diseño, la creación y la representación de un videojuego.

Este ha de ofrecer al programador un motor de renderizado para gráficos 2D y 3D, un motor que detecte la colisión física de objetos y la respuesta a dicha colisión, sonidos y música, animación, inteligencia artificial, comunicación con la red para juegos multijugador, posibilidad de ejecución en hilos, gestión de memoria o soporte para localización (traducción de los textos y audios del juego según idioma).

A continuación, se analiza el motor Unity 3D, puesto que es la tecnología que se potencia actualmente en el centro VERTEX para el desarrollo de videojuegos.

Unity 3D

Unity es una plataforma de desarrollo flexible y poderosa para crear juegos y experiencias interactivos 3D y 2D multiplataforma. Es un ecosistema completo para todo aquel que busque desarrollar un negocio a partir de la creación de contenido de alta gama y conectarse con sus jugadores y clientes más fieles y entusiastas. (Unity, 2015)

Una de las claves de Unity es que el juego puede ser portado a 21 plataformas diferentes empleando un solo código, lo cual facilita mucho a los desarrolladores crear una versión de su juego para varias plataformas.

Unity 3D integra *MonoDevelop* como entorno de desarrollo ya que es libre y gratuito y este fue diseñado primordialmente para C# y otros lenguajes .NET. Además, este es el entorno de desarrollo incluido por defecto en Unity 3D.

A continuación, alguna de sus principales características:

Parámetros	Unity 3D
Plataformas	PC, X360, Wii, PS3, Linux, Mac, iPhone, iPad, Android, Web.
Licencia	Existen dos tipos de licencias principales: Unity y Unity-Pro. La primera es gratuita (con algunas limitaciones) y está orientada a estudiantes y personas que deseen aprender a trabajar con este motor. La versión Pro no es gratuita, pero tiene características adicionales. Tanto Unity como Unity-Pro incluyen el entorno de desarrollo, tutoriales, ejemplos de proyectos y contenido, el apoyo a través de la comunidad de foros, wiki, y las futuras actualizaciones de la versión.
Entorno de edición	Entorno integrado de desarrollo con jerarquía, edición visual, inspector de propiedades y vista previa del videojuego.
Animación avanzada	El sistema de animación de Unity permite tratar a los

	objetos tanto genéricos como humanoides, en este último caso permite el uso de cada parte del cuerpo por separado para la animación de mismo bajo la configuración IK.
Scripting	Unity soporta tres lenguajes de <i>Scripting</i> : JavaScript, C#, y una versión modificada de Python llamada Boo. Los tres son igual de rápidos e interoperables. Además pueden utilizar librerías .NET con soporte a base de datos, expresiones regulares, XML, acceso a ficheros y funciones de red.
Física	NVIDIA's PhysX.
Iluminación y sombras	Sistema de iluminación altamente optimizado con <i>lightmaps</i> y sombras de tiempo real.
Cinematografía	Permite utilizar las cámaras, las partículas y las cinemáticas en cada escena ejecutándolas en tiempo real, además de tener varios efectos de imágenes que brinda un realismo sensacional a la visualización del juego.
Red	Posee funcionalidades multijugador como chat y puntuaciones en línea.
Audio	Mezcla en tiempo real gráficos en 3D con <i>streaming</i> de audio y vídeo.
Requerimientos mínimos de hardware	Windows: XP SP2; Mac OS X. Tarjeta gráfica con 64 MB. El uso del <i>Occlusion Culling</i> requiere GPU.

Tabla 2: Principales características de Unity3D.

1.7 Conclusiones parciales del capítulo

En este capítulo se abordaron aspectos teóricos y conceptos fundamentales para el desarrollo de la presente investigación. Se expuso el concepto de videojuego de manera general y el de videojuego de lucha y se realizó un estudio del estado del arte de este último. Además, fueron seleccionadas las herramientas y metodología con que se desarrollará el prototipo de videojuego en cuestión.

Capítulo 2: Solución propuesta

2.1 Introducción



El siguiente capítulo está enfocado en la presentación detallada de la solución al problema de investigación. Se realiza una descripción acerca de la estructura y funcionamiento de la propuesta de prototipo elaborado.

2.2 Documento de diseño

El documento de diseño expresa formalmente la idea principal y detalles de la propuesta de prototipo de videojuego. Este documento es revisado con la finalidad de saber si es factible el desarrollo, en caso contrario se modificará el documento de diseño hasta que sea aprobado o en su caso rechazado definitivamente.

Concepto	
Título	Fighter
Estudio/Diseñadores	Yosmani Claro Merino
Género	Lucha - Plataforma - Vista Lateral
Plataforma	PC
Versión	1
Sinopsis de Jugabilidad y Contenido	Se basa en la lucha entre personajes. Estos personajes poseen características propias que los diferencian unos de otros. Le juego se basa en escoger uno de estos personajes y se realizan varias rondas de lucha.
Mecánica	El jugador puede desplazarse lateralmente en el terreno. Dispone de dos tipos de golpes diferentes y un combo de golpes que se activa por combinaciones de teclas.
Tecnología	<ul style="list-style-type: none">• Hardware (Requerimientos mínimos):<ul style="list-style-type: none">-Tarjeta Gráfica con 64 MB.• Software:<ul style="list-style-type: none">-Adobe Photoshop CS6- Autodesk 3DMax- Unity 3D• Lenguaje de programación:<ul style="list-style-type: none">-C#
Público	Apto para mayores de 5 años

HISTORIAL DE VERSIONES	
Versión 1 (04/12/2015)	
VISIÓN GENERAL DEL JUEGO	
“Fighter” surge como necesidad de crear prototipo de juego de lucha que en un futuro pueda convertirse en un juego terminado y listo para su comercialización.	
MECÁNICA DEL JUEGO	
Cámara	3D Vista Lateral
Periféricos	Teclado y Mouse
Controles	Para moverse se usará los Axis de Control (Left y Right) Las teclas de acción son Z y X. La combinación de dichas teclas produce los combos de golpes.
Puntaje	La puntuación está implícita en la cantidad de golpes que hacen que el jugador pueda ganar o perder la lucha.
Guardar/Cargar	-
ESTADOS DEL JUEGO	
<pre> graph LR Start(()) --> MP[Menú Principal] MP --> SDP[Selección De Personajes] SDP --> P[Pelea] P -- "Salir al Menú Principal" --> MP P -- "Pause" --> P </pre>	
<i>Figura 7: Estados del juego.</i>	
INTERFACES	
Menú Principal	
Nombre de la Pantalla	Menú Principal
Descripción de la Pantalla	Es la interfaz inicial del juego, en ella el jugador tiene acceso a : Entrar: para iniciar el juego.

	Salir: Para salir del juego
Imagen	 <p style="text-align: center;"><i>Figura 8: Menú principal.</i></p>
Seleccionar personajes	
Nombre de la Pantalla	Seleccionar personajes
Descripción de la Pantalla	Es la interfaz para seleccionar el personaje con que se jugará. Presenta dos botones para seleccionar el personaje y un botón para comenzar la partida.
Imagen	 <p style="text-align: center;"><i>Figura 9: Selección de personajes.</i></p>
Niveles	
Escenario de batalla	
Título del Nivel	Escenario de batalla
Encuentro	Es el primer escenario que se activa al comenzar la partida.
Descripción	Rejilla en gris en la cual se simula una pelea en un entorno computacional.
Objetivos	Derrotar el oponente.

Progresos	Después que el caracter neutraliza el primer contrincante, aparece un segundo oponente en el mismo escenario.
Ítems	No se utiliza ningún objeto en este nivel.
Personajes	Ninja y Bruto.
Música y Efectos de Sonido	La música utilizada en este nivel recrea un ambiente de lucha. Se utilizan efectos de sonido para los diferentes golpes que efectúan y que reciben los caracteres.
Referencias de BGM y SFX	Música de fondo: BGM01. Efectos de audio: SFX01, SFX02, SFX03, SFX04, SFX05, SFX06.

Progreso del juego

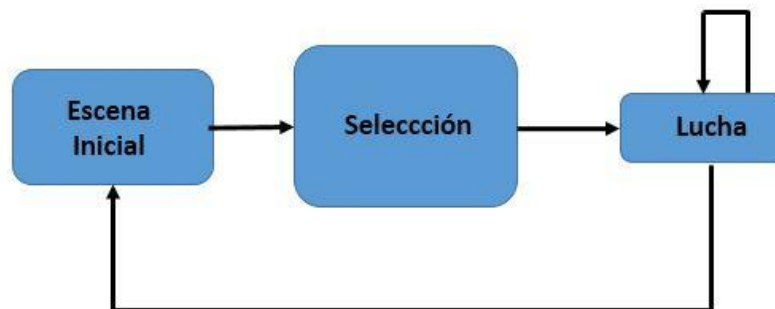

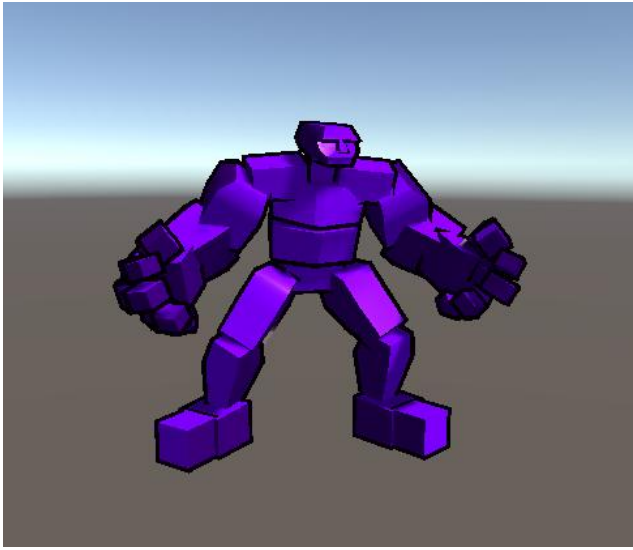


Figura 10: Progreso del juego.

Personajes

Ninja

Nombre del Personaje	Ninja
Descripción	Personaje hábil, que posee un arma a su disposición. Domina las técnicas de artes marciales.
Imagen	

	 <p data-bbox="850 632 1015 659"><i>Figura 11: Ninja.</i></p>
Concepto	-
Encuentro	Personaje seleccionable.
Habilidades	<ul style="list-style-type: none"> -Corte con espada. -Patada alta. -Combo: Danza de espada.
Bruto	
Nombre del Personaje	Bruto
Descripción	Personaje increíblemente fuerte, robusto, alto. No maneja ningún arte marcial ni objeto, solo la fuerza bruta.
Imagen	 <p data-bbox="850 1774 1015 1801"><i>Figura 12: Bruto.</i></p>
Concepto	-

Encuentro	Personaje seleccionable.
Habilidades	-Golpe con 1 sola mano. -Golpe con 2 manos. -Golpe aéreo. -Combo: Derribo.
Música y Sonido	
Música de fondo: <ul style="list-style-type: none"> • BGM01: BattleSong.ogg Efectos de audio: <ul style="list-style-type: none"> • SFX01: atack_hit.ogg • SFX02: atackfx1.ogg • SFX03: atackfx2.ogg • SFX04: atackfx3.ogg • SFX05: ogre1.ogg • SFX06: ogre2.ogg 	
Imágenes de Concepto	
-	
Miembros de Equipo	
Diseñador y desarrollador: Yosmani Rafael Claro Merino	
Detalles de Producción	
Fecha de inicio	28 septiembre de 2015
Fecha de Terminación	4 de diciembre de 2015

2.3 Fase de exploración

Es la fase en la que se define el alcance general del proyecto. El cliente define lo que necesita mediante la redacción de las Historias de Usuarios. Los programadores estiman los tiempos de desarrollo en base a esta información (Letelier, 2006).

Esta fase dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado.

2.3.1 Historias de usuario

Las historias de usuario son la técnica utilizada por XP para especificar los requisitos del software. Se realiza una por cada funcionalidad del sistema, se emplean para hacer

estimaciones de tiempo y para el plan de lanzamientos de iteraciones del producto. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Historias de usuario	
Número:	Nombre:
Programador:	Iteración Asignada: Número de iteración.
Prioridad: Alta, Media o Baja.	Tiempo Estimado: Tiempo en días que se le asignará. (Estimado)
Descripción:	

Tabla 3: Descripción de las Historias de usuario (HU).

A continuación se muestran las historias de usuario presentes en la solución:

Historias de usuario	
Número: 1	Nombre: Mostrar interfaz de usuario.
Programador: Yosmani Rafael Claro	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 0.5
Descripción: Cuando se ejecuta el juego el sistema muestra la interfaz de usuario o menú principal. Se muestra la opción "Entrar" para entrar al menú de selección de caracteres y "Salir" para cerrar el juego.	

Tabla 4: HU Mostrar interfaz de usuario.

Historias de usuario	
Número: 2	Nombre: Seleccionar personaje.
Programador: Yosmani Rafael Claro	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 0.5
Descripción:	

El usuario selecciona el personaje deseado para luchar y la computadora selecciona aleatoriamente el oponente contra el cual se luchará. Una vez seleccionado el personaje comienza el combate.

Tabla 5: HU Seleccionar personaje.

Historias de usuario	
Número: 3	Nombre: Cerrar proceso correctamente.
Programador: Yosmani Rafael Claro	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 0.5
Descripción: El sistema debe permitir cerrar el juego cuando finalice el combate o cuando lo desee el jugador. En el momento que se seleccione la opción "Cerrar" el juego debe dejar de ejecutarse.	

Tabla 6: HU Cerrar proceso correctamente.

Historias de usuario	
Número: 4	Nombre: Iniciar combate.
Programador: Yosmani Rafael Claro	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 0.5
Descripción: Después de seleccionar el personaje con que se jugará se inicia el combate. Para iniciar el combate se construye una escena con el escenario donde se efectuará el combate, se colocan los personajes que se enfrentarán y se activan las teclas que controlan a estos caracteres.	

Tabla 7: HU Iniciar combate.

Historias de usuario	
Número: 5	Nombre: Finalizar combate.
Programador: Yosmani Rafael Claro	Iteración Asignada: 1
Prioridad: Alta	Tiempo Estimado: 0.5

<p>Descripción:</p> <p>El sistema debe permitir finalizar el combate cuando uno de los caracteres gana el combate o cuando el jugador decide cancelar dicho combate. Cuando finaliza el combate se retorna al menú principal.</p>
--

Tabla 8: HU Finalizar combate.

Historias de usuario	
Número: 6	Nombre: Activar cambio de escena.
Programador: Yosmani Rafael Claro	Iteración Asignada: 2
Prioridad: Alta	Tiempo Estimado: 0.5
<p>Descripción:</p> <p>El sistema permitirá activar una escena en dependencia del botón que se haya pulsado.</p> <p>Si se presiona el botón “Comenzar” del menú principal se activa la escena de selección de caracteres.</p> <p>Si se presionan las imágenes de los caracteres en el menú de selección de personajes se activa la escena del combate.</p> <p>Si se gana o se pierde el combate se muestra un menú con el resultado obtenido y pregunta si se desea jugar otra vez o salir del juego.</p>	

Tabla 9: HU Activar cambio de escena.

Historias de usuario	
Número: 7	Nombre: Activar botones de acción.
Programador: Yosmani Rafael Claro	Iteración Asignada: 2
Prioridad: Alta	Tiempo Estimado: 0.5
<p>Descripción:</p> <p>Cuando se selecciona el personaje con que se jugará y se activa la escena del combate, inmediatamente se activan los botones que permiten controlar el carácter. Estos botones son las letras “Z”, “X” y las flechas de adelante y atrás.</p>	

Tabla 10: HU Activar botones de acción.

Historias de usuario

Número: 8	Nombre: Activar animaciones según correspondencia de acción.	
Programador: Yosmani Rafael Claro	Iteración Asignada: 2	
Prioridad: Alta	Tiempo Estimado: Tiempo en días que se le asignará. (Estimado)	
<p>Descripción:</p> <p>Se activará la animación del personaje en dependencia de la acción que el jugador desee ejecutar.</p> <p>Si se pulsa la tecla “Z” o “X” se activará la animación de realizar patada o de ejecutar un golpe con el puño.</p> <p>Si se pulsan las teclas de adelante y atrás se activará la animación de caminar hacia adelante y hacia atrás.</p> <p>Si se pulsa la combinación de teclas Adelante y “X” se activa la animación de combo de golpes.</p>		

Tabla 11: HU Activar animaciones según correspondencia de acción.

Historias de usuario		
Número: 9	Nombre: Ejecutar ataque básico (jugador).	
Programador: Yosmani Rafael Claro	Iteración Asignada: 2	
Prioridad: Alta	Tiempo Estimado: 1	
<p>Descripción:</p> <p>Cuando se presionan las teclas “Z” o “X” el personaje controlado por el jugador realizará un ataque básico al oponente.</p>		

Tabla 12: HU Ejecutar ataque básico (jugador).

Historias de usuario		
Número: 10	Nombre: Ejecutar combo de golpes (jugador).	
Programador: Yosmani Rafael Claro	Iteración Asignada: 2	
Prioridad: Alta	Tiempo Estimado: 1	
<p>Descripción:</p>		

Cuando se presionan la combinación de las teclas de Adelante y "X" el personaje controlado por el jugador realizará un combo de golpes al oponente.

Tabla 13: HU Ejecutar combo de golpes (jugador).

Historias de usuario	
Número: 11	Nombre: Ejecutar ataque básico (oponente).
Programador: Yosmani Rafael Claro	Iteración Asignada: 3
Prioridad: Alta	Tiempo Estimado: 1
Descripción: El oponente ejecuta un ataque básico de manera automática basándose en técnicas de Inteligencia Artificial.	

Tabla 14: HU Ejecutar ataque básico (oponente).

Historias de usuario	
Número: 12	Nombre: Ejecutar combo de golpes (oponente).
Programador: Yosmani Rafael Claro	Iteración Asignada: 3
Prioridad: Alta	Tiempo Estimado: 1
Descripción: El oponente ejecuta un combo de golpes de manera automática basándose en técnicas de Inteligencia Artificial.	

Tabla 15: HU Ejecutar combo de golpes (oponente).

Historias de usuario	
Número: 13	Nombre: Añadir sistema de colisiones.
Programador: Yosmani Rafael Claro	Iteración Asignada: 3
Prioridad: Alta	Tiempo Estimado: 0.5
Descripción: Se le adicionan los componentes "RigidBody", "CapsuleCollider", "BoxCollider" y	

<p>“EsphereCollider” a ambos personajes con el objetivo de que detecten colisiones con los objetos externos.</p>
--

Tabla 16: HU Añadir sistema de colisiones.

Historias de usuario	
Número: 14	Nombre: Activar efectos de sonido.
Programador: Yosmani Rafael Claro	Iteración Asignada: 3
Prioridad: Media	Tiempo Estimado: 0.5
<p>Descripción: Se activan efectos de sonido para cada uno de los tipos de golpes que se pueden efectuar. Además de que en el escenario del combate se activará un sonido de fondo que se ejecutará mientras dure la pelea.</p>	

Tabla 17: HU Activar efectos de sonido.

Historias de usuario	
Número: 15	Nombre: Activar efectos especiales.
Programador: Yosmani Rafael Claro	Iteración Asignada: 3
Prioridad: Media	Tiempo Estimado: 0.5
<p>Descripción: Se activarán los efectos especiales en dependencia del golpe que se ejecute.</p>	

Tabla 18: HU Activar efectos especiales.

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Estos requisitos a menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema. (Sommerville, 2005)

Requerimientos de software

El juego podrá ejecutarse en los siguientes sistemas operativos Windows 7 SP1+, Windows 8, Windows 8.1 y Windows 10.

Requerimientos de software

Se necesitan como requerimientos mínimos:

- Procesador Pentium IV a 2,6 GHz, 512 MB de RAM, 32 bits de profundidad de color y 128 MB de memoria de video integrado o 64 MB de video dedicado, compatible con Open GL o DirectX 3D.

Rendimiento

El juego debe ejecutarse en computadoras de bajas prestaciones sin afectar la calidad visual.

Usabilidad

El juego deberá ser fácil de usar por todo tipo de usuarios.

2.4 Planificación

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses.

2.4.1 Estimación de esfuerzos por historias de usuario

Historia de usuario	Estimación (semanas)
Mostrar interfaz de usuario	0.5
Seleccionar personaje	0.5
Cerrar proceso correctamente	0.5
Iniciar combate	0.5
Finalizar combate	0.5
Activar cambio de escena	0.5
Activar botones de acción	0.5
Activar animaciones según correspondencia de acción	1
Ejecutar ataque básico (jugador)	1
Ejecutar combo de golpes (jugador)	1
Ejecutar ataque básico (oponente)	1
Ejecutar combo de golpes (oponente)	1

Añadir sistema de colisiones	0.5
Activar efectos de sonido	0.5
Activar efectos especiales	0.5

Tabla 19: Estimación de esfuerzos por historias de usuario.

2.5 Iteraciones

Después de haber identificado las historias de usuarios y realizado una previa estimación de esfuerzos se procede a la planificación de la etapa de implementación del sistema. En este plan se especifica la prioridad con que se implementarán las historias de usuarios organizadas por iteraciones así como posibles fechas de liberación. Este paso es de suma importancia en el ciclo de desarrollo de XP debido a que las funcionalidades son planificadas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración.

Iteraciones	Descripción de la iteración	Orden de las HU a implementar	Duración total
1	Se implementan las historias de usuario iniciales, obteniendo al final de la misma una primera versión del producto.	HU1, HU2, HU3, HU4, HU5	2.5 semanas
2	Se realiza la implementación de las HU que presentan un nivel de complejidad medio. Además, se corregirán errores detectados en la implementación de las funcionalidades de la iteración anterior.	HU6, HU7, HU8, HU9, HU10	4 semanas
3	Se implementan las HU de mayor nivel de complejidad. En este caso se encuentran las relacionadas con el comportamiento del	HU11, HU12, HU13, HU14, HU15	3.5 semanas

	<p>oponente, el sistema de colisiones y los efectos especiales y de sonido.</p> <p>Además se corrigen errores de iteraciones anteriores.</p> <p>Al finalizar se obtiene la versión final del producto.</p>		
--	--	--	--

Tabla 20: Plan de duración de las iteraciones.

2.6 Diseño de la solución propuesta

La metodología XP, no requiere la descripción del sistema mediante diagramas de clase utilizando la notación UML, sino que en su lugar se guía por técnicas como las tarjetas CRC (Contenido, Responsabilidad y Colaboración). Esto no implica que no se utilicen dichos diagramas para obtener una mejor visión y comunicación entre el equipo de trabajo, siempre y cuando no posea una alta complejidad y defina información importante.

2.6.1 Modelo de dominio

En Unity 3D casi todos los elementos, visibles o no, se denominan GameObject. Es la clase básica para todos los objetos que se usan en cada una de las escenas del juego. En las escenas estos objetos pueden instanciarse en múltiples ocasiones, lo que se traduciría en múltiples objetos del mismo tipo en dicha escena. Además de que puede establecerse una relación de manera jerárquica entre ellos, donde un GameObject puede formarse por la combinación de otros objetos.

No obstante, los GameObject no hacen nada por sí mismos. Estos necesitan propiedades especiales antes de que puedan volverse un personaje, un ambiente, o un efecto especial.

Estas propiedades especiales las proporcionan los Componentes y los *Scripts*, los cuales se añaden a un GameObject para determinar su comportamiento. Vale aclarar que los Scripts son componentes que el programador implementa para definir comportamientos de estos GameObject.

A continuación se muestra gráficamente como se relacionan estos conceptos:

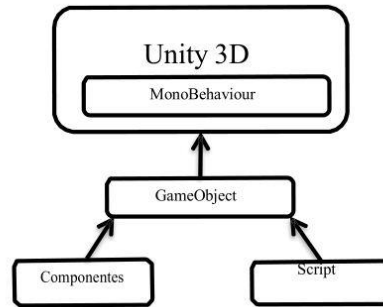


Figura 13: Relación entre el núcleo de Unity y los objetos que componen la escena.

A continuación se muestra el modelo de dominio que permitirá visualizar la relación que entre los diferentes GameObject incluidos en cada escena del juego y los componentes que cada uno tiene asociado

El modelo para la escena Menú Principal se muestra a continuación:

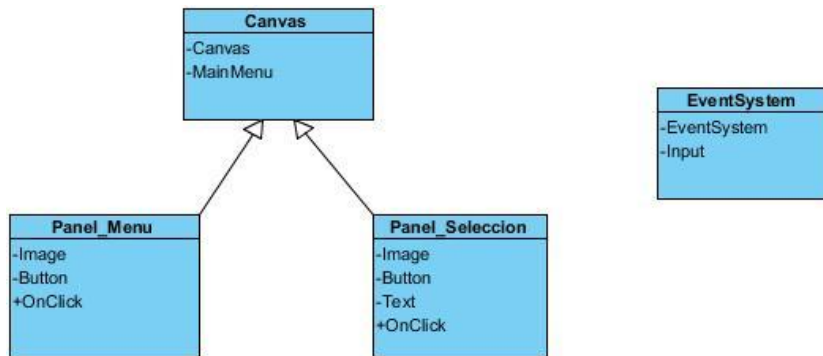


Figura 14: Modelo de dominio de Menú Principal.

Luego, el de la escena principal sería el siguiente:

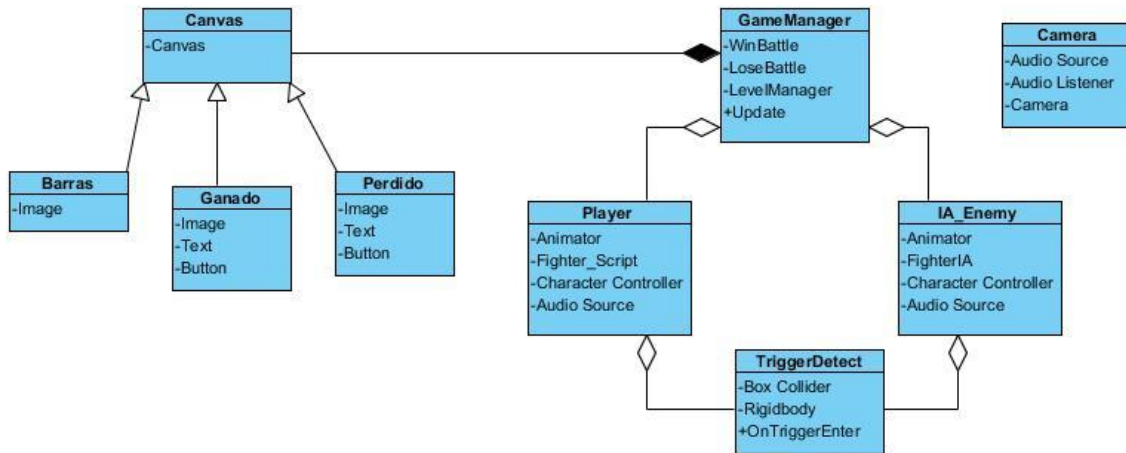


Figura 15: Modelo de dominio de la escena principal.

Detalles de los elementos del diagrama de clases:

Canvas: Contenedor de los componentes de interfaces en 2D.

Panel_Menu: Encargado de manejar todas las conexiones entre las interfaces y los eventos de selección.

Panel_Seleccion: Contiene la interfaz de selección de caracteres y envía la selección al script MainMenu.

EventSystem: Se encarga de manejar todas las entradas de datos que contiene la interfaz. Componente nativo de Unity.

Barras: Contiene la interfaz de las barras de vida de los jugadores.

Perdido: Contiene la interfaz que se muestra cuando se pierde una partida.

Ganado: Contiene la interfaz que se muestra cuando se gana una partida.

GameManager: Se encarga de gestionar toda las entradas y las salidas de la batalla, el daño que se causan los jugadores y decide que interfaz mostrar si se gana o se pierde la partida.

Player: Se encarga de manejar las acciones del caracter.

IA_Energy: Se encarga de manejar la IA del enemigo.

Camera: Se encarga de renderizar los objetos existentes en la escena.

TriggerDetect: Detecta colisiones en los golpes.

2.6.2 Tarjetas CRC

Una tarjeta CRC es el artefacto ingenieril que propone la metodología XP para encontrar los objetos del dominio de la aplicación, sus responsabilidades y cómo colaboran con otros para realizar tareas. En estas tarjetas se registra el nombre de las clases, sus responsabilidades y las otras clases con las que colabora.

Clase	
Responsabilidades	Colaboradores

Tabla 21: Prototipo de una tarjeta CRC.

Clase: Nombre de la clase que se está modelando

Responsabilidades: Es una descripción de alto nivel del propósito de la clase.

Colaboradores: Indica las clases con las que se relaciona para cumplir su responsabilidad.

A continuación, se definen las tarjetas CRC asociadas al Menú Principal:

Canvas	
Responsabilidades: Contiene los componentes de interfaces en 2D. Encargado de mostrar en la pantalla todos los elementos de la interfaz de usuario (UI).	Colaboradores: Panel_Menu Panel_Seleccion

Tabla 22: Tarjeta CRC para Canvas.

Panel_Menu	
Responsabilidades: Encargado de manejar todas las conexiones entre las interfaces y los eventos de selección.	Colaboradores:

Tabla 23: Tarjeta CRC para Panel_Menu.

Panel_Seleccion	
Responsabilidades: Contiene la interfaz de selección de caracteres y envía la selección al script MainMenu.	Colaboradores:

Tabla 24: Tarjeta CRC para Panel_Seleccion.

EventSystem	
<p>Responsabilidades:</p> <p>Se encarga de procesar y manejar todos los eventos en una escena de Unity. Envía eventos a objetos en la aplicación basado en dispositivos de entrada que pueden ser el teclado, el mouse, una pantalla táctil, u otro dispositivo los cuales manejan las entradas de datos que contiene la interfaz.</p>	<p>Colaboradores:</p>

Tabla 25: Tarjeta CRC para EvenSystem.

Tarjetas CRC asociadas a la Escena principal.

Canvas	
<p>Responsabilidades:</p> <p>Contiene los componentes de interfaces en 2D. Encargado de mostrar en la pantalla todos los elementos de la interfaz de usuario (UI).</p>	<p>Colaboradores:</p> <p>Barras Perdido Ganado</p>

Tabla 26: Tarjeta CRC para Canvas.

Barras	
<p>Responsabilidades:</p> <p>Contiene la interfaz de las barras de vida de los jugadores.</p>	<p>Colaboradores:</p>

Tabla 27: Tarjeta CRC para Barras.

Perdido	
<p>Responsabilidades:</p> <p>Contiene la interfaz que se muestra cuando se pierde una partida.</p>	<p>Colaboradores:</p>

Tabla 28: Tarjeta CRC para Perdido.

Ganado

Responsabilidades: Contiene la interfaz que se muestra cuando se gana una partida.	Colaboradores:
--	-----------------------

Tabla 29: Tarjeta CRC para Ganado.

GameManager	
Responsabilidades: Se encarga de gestionar toda las entradas y las salidas de la batalla, el daño que se causan los jugadores y decide que interfaz mostrar si se gana o se pierde la partida.	Colaboradores: Player IA_Enemy

Tabla 30: Tarjeta CRC para GameManager.

Player	
Responsabilidades: Se encarga de manejar las acciones del caracter.	Colaboradores: TriggerDetect

Tabla 31: Tarjeta CRC para Player.

IA_Enemy	
Responsabilidades: Se encarga de manejar la IA del enemigo.	Colaboradores: TriggerDetect

Tabla 32: Tarjeta CRC para IA_Enemy.

TriggerDetect	
Responsabilidades: Encargado de detectar las colisiones en los personajes cuando realizan un golpe al oponente.	Colaboradores:

Tabla 33: Tarjeta CRC para TriggerDetect.

Camera	
Responsabilidades: Se encarga de renderizar los objetos	Colaboradores: TriggerDetect

existentes en la escena. Capturan y muestran el mundo para el jugador.	
--	--

Tabla 34: Tarjeta CRC para Camera.

2.6.3 Componentes

Los componentes son elementos que aportan características a los GameObject. Al agregar componentes, se agregan funcionalidades a los GameObject.

Dependiendo del objeto que se quiera crear, se agregan diferentes combinaciones de componentes al GameObject.

Existen componentes para el trabajo con las mallas; componentes para la funcionalidad de audio y cámara; componentes relacionados con la física (colisionadores y cuerpos rígidos), sistemas de partículas, sistemas de búsqueda de ruta, componentes personalizados de terceros, entre otros. Además de que se pueden crear componentes mediante el uso de los Scripts.

Componentes de visión

- **Camera**

Las cámaras son los dispositivos que capturan y muestran el mundo al jugador. Se puede tener un número ilimitado de cámaras en la escena. Ellas pueden ser establecidas para renderizar en cualquier orden, en cualquier lugar de la pantalla, o en solo ciertas partes de la pantalla.

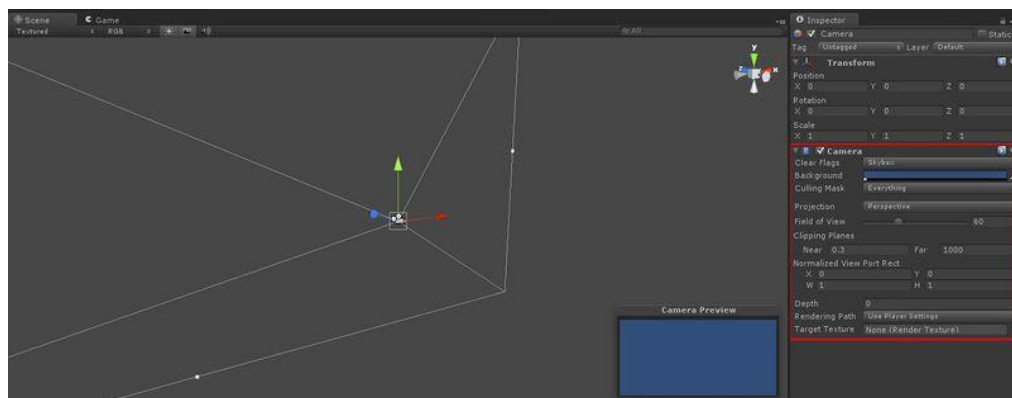


Figura 16: Componente Camera.

Componentes de audio

- **Audio Listener**

Actúa como un dispositivo parecido a un micrófono. Este recibe información desde cualquier fuente de audio que se encuentre en la escena y reproduce el sonido a través de los altavoces del ordenador.

- **Audio Source**

Este componente reproduce un clip de audio en la escena. Si es un clip de audio 3D, la fuente se reproduce en una posición determinada y se atenúa con la distancia.

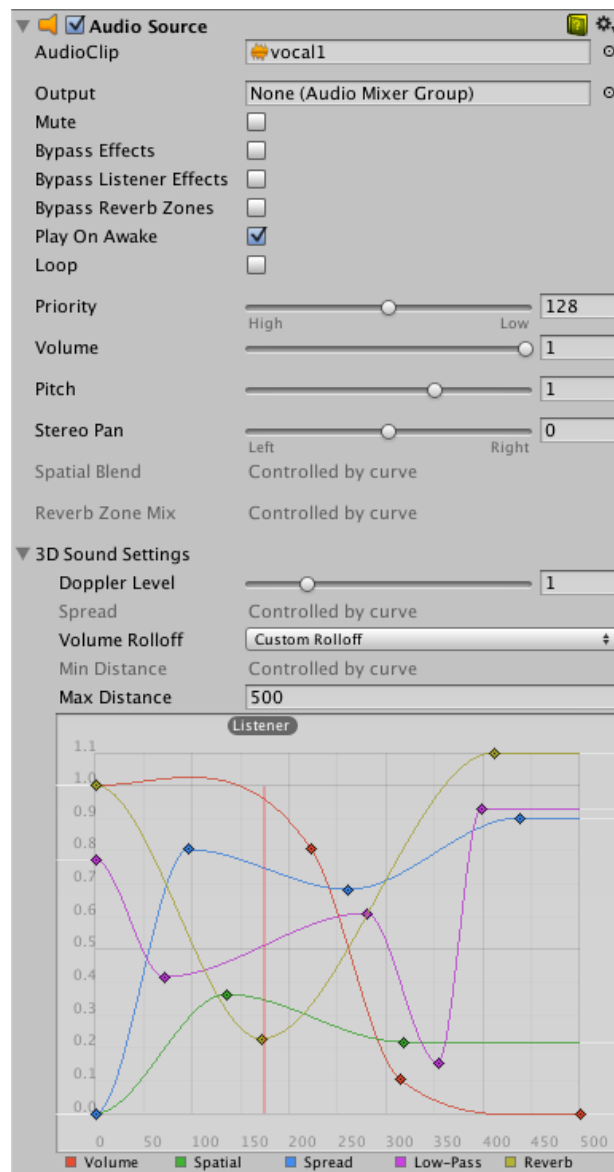


Figura 17: Componente Audio Source.

Componentes de física

- **Box Collider**

Un Box Collider es una primitiva básica de colisión en forma de cubo. Son útiles para cualquier objeto con forma cúbica, como una caja o un cofre.

- **Capsule Collider**

El Capsule Collider se compone de dos medias esferas unidas por un cilindro. Se utiliza en el controlador de caracteres y funciona bien para los polos, o se puede combinar con otros Colliders para formas inusuales.

- **Sphere Collider**

El Sphere Collider es una colisión básica en forma de esfera primitiva. Puede ser redimensionado mediante la propiedad Radius pero no puede ser escalado a lo largo de los tres ejes independientemente. Así como el uso obvio para objetos esféricos como una pelota de tenis, la esfera también funciona bien para piedras que están cayendo y otros objetos que necesitan rodar y caerse.

- **Rigidbody**

El Rigidbody permite a los GameObject actuar bajo el control del motor de física. Este puede recibir fuerza y torque para hacer que los objetos se muevan en una manera realista.

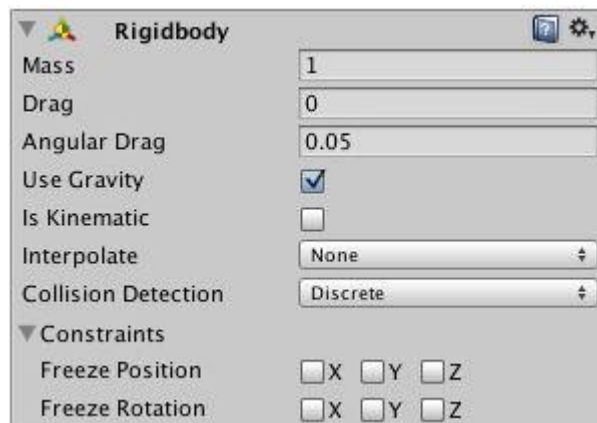


Figura 18: Componente Rigidbody

Componentes de animación

- **Animator**

El componente Animator es usado para asignar una animación a un GameObject en la escena. Este componente requiere una referencia a un Animator Controller el cual define que clip de animación usar, y controla cuándo y cómo hacer la transición entre uno y otro.

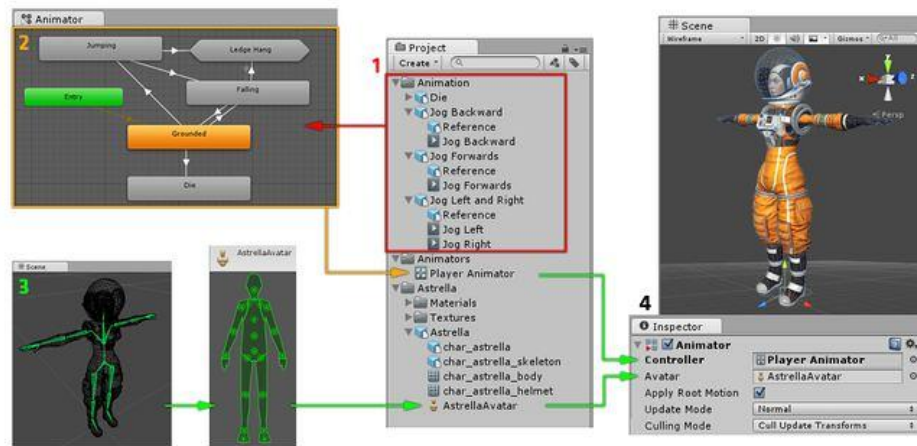


Figura 19: Animator y Animator Controller en acción.

2.7 Scripts

La programación de *scripts* es un ingrediente esencial en todos los juegos. Incluso el juego más simple necesitará *scripts* para responder a entradas del jugador y asegurar que los eventos del juego se ejecutan en el momento adecuado. Además, los *scripts* pueden ser usados para crear efectos gráficos, controlar el comportamiento físico de objetos o incluso implementar un sistema de inteligencia artificial para los personajes del juego (Unity, 2015).

Unity permite la creación de *scripts* en tres lenguajes diferentes: C#, JavaScript y Boo. Cada proyecto puede contener *scripts* escritos en cualquiera de estos lenguajes, aunque es recomendado que se escoja un solo lenguaje para implementar toda la lógica del juego.

A continuación, se detallan los *scripts* creados para realizar este prototipo de videojuego:

FighterIA

Implementa la inteligencia artificial (IA) del contrincante. Su funcionamiento se basa en la distancia a que se encuentra del jugador, en dependencia de esta selecciona una acción a ejecutar.

FighterScript

Maneja el jugador principal. En dependencia de la combinación de teclas que se introduzca, se ejecuta la acción en el personaje.

GameManager

Se encarga de activar el personaje seleccionado y seleccionar aleatoriamente los contrincantes con los que luchará. Maneja la interfaz de usuario y es el que dicta quien gana y quien pierde.

TriggerDetect

Encargado de enviar mensajes al contrario sobre el daño recibido en una acción.

MainMenu

Encargado de manejar la interfaz principal y la selección de personajes. Además de crear un enlace entre los botones de la interfaz y el script del personaje.

2.8 Conclusiones parciales

En este capítulo se logró definir las principales características que el prototipo de videojuego debe poseer. Se expusieron los artefactos generados en las fases que establece metodología de desarrollo de software XP. Se definieron 15 historias de usuario que serán implementadas en 3 iteraciones, detallando sus niveles de complejidad y prioridad. Se obtuvo el diagrama de clases y las tarjetas CRC que permitieron modelar la problemática planteada para la creación del videojuego.

Capítulo 3: Implementación y prueba

3.1 Introducción

La metodología de desarrollo de software XP plantea que la implementación de un software debe realizarse de forma iterativa, obteniendo, al culminar cada iteración, un producto funcional que debe ser probado y mostrado al cliente para incrementar la visión de los desarrolladores.

En el presente capítulo se detallan las iteraciones realizadas durante la etapa de construcción del prototipo de videojuego, además de exponer las tareas de ingeniería generadas para cada historia de usuario que se desarrolló. Se ejecutaron pruebas de aceptación para probar las funcionalidades descritas anteriormente.

3.2 Fase de Implementación

En esta fase se lleva a cabo la implementación de cada una de las historias de usuarios, previamente organizadas por iteraciones. Para la correcta implementación de estas HU se crean tareas de programación o de ingeniería. Estas tareas pueden escribirse en un lenguaje técnico, más familiar a los expertos que las desarrollarán.

3.2.1 Iteración 1

En esta iteración se desarrollan las HU iniciales del proyecto, las cuales serán las que aportarán las características básicas de un videojuego.

Historias de usuario	Tiempo de Implementación	
	Estimación	Real
Mostrar interfaz de usuario	0.5	0.5
Seleccionar personaje	0.5	0.5
Cerrar proceso correctamente	0.5	0.5
Iniciar combate	0.5	0.5
Finalizar combate	0.5	0.5

Tabla 35: Tiempo de implementación de la Iteración 1.

A continuación alguna de las tareas de ingeniería definidas para la primera iteración:

Tarea de Ingeniería	
No.Tarea: 1	No. HU: 1
Nombre: Diseñar e implementar interfaz de usuario del juego.	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5

Fecha de inicio: 28/09/15	Fecha de fin: 30/09/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Tiene como objetivo diseñar e implementar la interfaz que se mostrará al usuario al iniciar el juego.	

Tabla 36: Diseñar e implementar interfaz de usuario del juego.

Tarea de Ingeniería	
No.Tarea: 2	No. HU: 2
Nombre: Seleccionar personaje.	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 30/09/15	Fecha de fin: 02/10/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Se muestran al usuario los personajes con los que puede jugar y este escoge uno de ellos.	

Tabla 37: Seleccionar personaje.

Tarea de Ingeniería	
No.Tarea: 3	No. HU: 3
Nombre: Cerrar proceso correctamente.	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 05/10/15	Fecha de fin: 07/10/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Una vez finalizada la partida se cierra el proceso que ejecuta la aplicación.	

Tabla 38: Cerrar proceso correctamente.

Tarea de Ingeniería	
No.Tarea: 4	No. HU: 4
Nombre: Iniciar combate.	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 07/10/15	Fecha de fin: 09/10/15
Programador responsable: Yosmani Rafael Claro	

Descripción: Una vez seleccionado el personaje con que se jugará, se crea una lista de oponentes escogidos al azar. Posteriormente se activa la escena de combate, se posicionan ambos caracteres en la escena y se comienza el combate.
--

Tabla 39: Iniciar combate.

Tarea de Ingeniería	
No.Tarea: 5	No. HU: 5
Nombre: Finalizar combate.	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 12/10/15	Fecha de fin: 14/10/15
Programador responsable: Yosmani Rafael Claro	
Descripción: El combate termina si uno de los dos caracteres gana el combate o si el usuario termina el combate. Cuando el combate finaliza se destruye el escenario de combate y se regresa al Menú Principal.	

Tabla 40: Finalizar combate.

3.2.2 Iteración 2

En la presente iteración se añaden funcionalidades que permiten a los caracteres ejecutar ataques básicos y combos de golpes, todos ejecutados a partir de una combinación de teclas. Además se implementan los cambios entre las diferentes escenas del juego.

Historias de usuario	Tiempo de Implementación	
	Estimación	Real
Activar cambio de escena	0.5	0.5
Activar botones de acción	0.5	0.5
Activar animaciones según correspondencia de acción	1	1
Ejecutar ataque básico (jugador)	1	1
Ejecutar combo de golpes (jugador)	1	1

Tabla 41: Tiempo de implementación de la Iteración 2.

Para esta iteración se definieron las siguientes tareas:

Tarea de Ingeniería	
No.Tarea: 6	No. HU: 6

Nombre: Activar cambio de escena	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 14/10/15	Fecha de fin: 16/10/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Se implementa la funcionalidad de cambiar de una escena a otra en dependencia de la acción realizada por el usuario.	

Tabla 42: Activar cambio de escena.

Tarea de Ingeniería	
No.Tarea: 7	No. HU: 7
Nombre: Activar botones de acción	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 19/10/15	Fecha de fin: 21/10/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Una vez seleccionado el personaje con que se jugará y se active la opción de Comenzar partida se activarán los botones con los que se controlará dicho personaje.	

Tabla 43: Activar botones de acción.

Tarea de Ingeniería	
No.Tarea: 8	No. HU: 8
Nombre: Activar animaciones	
Tipo de tarea: Desarrollo	Puntos de estimación: 1
Fecha de inicio: 21/10/15	Fecha de fin: 28/10/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Cuando se pulse una de las teclas de acción, se activará la animación correspondiente a dicha acción.	

Tabla 44: Activar animaciones.

Tarea de Ingeniería	
No.Tarea: 9	No. HU: 9
Nombre: Ejecutar ataque básico (jugador)	
Tipo de tarea: Desarrollo	Puntos de estimación: 1
Fecha de inicio: 28/10/15	Fecha de fin: 04/11/15
Programador responsable: Yosmani Rafael Claro	

Descripción: Cuando se pulse las teclas Z o X, el personaje controlado por el jugador realizará un ataque básico a su oponente.

Tabla 45: Ejecutar ataque básico (jugador).

Tarea de Ingeniería	
No.Tarea: 10	No. HU: 10
Nombre: Ejecutar combo de golpes (jugador)	
Tipo de tarea: Desarrollo	Puntos de estimación: 1
Fecha de inicio: 04/11/15	Fecha de fin: 11/11/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Cuando se pulse la combinación de teclas definida, el personaje controlado por el jugador ejecutará un combo de golpes. Este combo causa un daño superior al ataque básico.	

Tabla 46: Ejecutar combo de golpes (jugador).

3.2.3 Iteración 3

Por último se implementa el comportamiento de los oponentes mediante técnicas de Inteligencia Artificial. Además se agregan las colisiones a los personajes, así como los efectos especiales y de sonido.

Historias de usuario	Tiempo de Implementación	
	Estimación	Real
Ejecutar ataque básico (oponente)	1	1
Ejecutar combo de golpes (oponente)	1	1
Añadir sistema de colisiones	0.5	0.5
Activar efectos de sonido	0.5	0.5
Activar efectos especiales	0.5	0.5

Tabla 47: Tiempo de implementación de la Iteración 2.

Las tareas referentes a esta iteración son las siguientes:

Tarea de Ingeniería	
No.Tarea: 11	No. HU: 11
Nombre: Ejecutar ataque básico (oponente)	

Tipo de tarea: Desarrollo	Puntos de estimación: 1
Fecha de inicio: 11/11/15	Fecha de fin: 18/11/15
Programador responsable: Yosmani Rafael Claro	
<p>Descripción:</p> <p>Se implementa la opción de realizar un ataque básico al carácter controlado por el jugador. El caracter toma dicha decisión basado en las reglas de Inteligencia Artificial definidas en el script FighterIA.</p>	

Tabla 48: Ejecutar ataque básico (oponente).

Tarea de Ingeniería	
No.Tarea: 12	No. HU: 12
Nombre: Ejecutar combo de golpes (oponente)	
Tipo de tarea: Desarrollo	Puntos de estimación: 1
Fecha de inicio: 18/11/15	Fecha de fin: 25/11/15
Programador responsable: Yosmani Rafael Claro	
<p>Descripción:</p> <p>Se implementa la opción de realizar un combo de golpes al carácter controlado por el jugador. El caracter toma dicha decisión basado en las reglas de Inteligencia Artificial definidas en el script FighterIA.</p>	

Tabla 49: Ejecutar combo de golpes (oponente).

Tarea de Ingeniería	
No.Tarea: 13	No. HU: 13
Nombre: Añadir sistema de colisiones	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 25/11/15	Fecha de fin: 27/11/15
Programador responsable: Yosmani Rafael Claro	
<p>Descripción:</p> <p>Se les adicionan los componentes "RigidBody", "CapsuleCollider", "BoxCollider" y "SphereCollider" a ambos personajes con el objetivo de que detecten colisiones con los objetos externos.</p>	

Tabla 50: Añadir sistema de colisiones.

Tarea de Ingeniería

No.Tarea: 14	No. HU: 14
Nombre: Activar efectos de sonido	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 30/11/15	Fecha de fin: 02/12/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Reproducir clips de sonido en dependencia de la acción ejecutada.	

Tabla 51: Activar efectos de sonido.

Tarea de Ingeniería	
No.Tarea: 15	No. HU: 15
Nombre: Activar efectos especiales	
Tipo de tarea: Desarrollo	Puntos de estimación: 0.5
Fecha de inicio: 02/12/15	Fecha de fin: 04/12/15
Programador responsable: Yosmani Rafael Claro	
Descripción: Al ejecutar un ataque o un combo de golpes se activará el efecto especial correspondiente.	

Tabla 52: Activar efectos especiales.

3.3 Pruebas

Uno de los pilares de la metodología XP es el uso de las pruebas para comprobar el funcionamiento de los códigos que se vayan implementando. Esto permite aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También permite aumentar la seguridad de evitar efectos colaterales no deseados a la hora de realizar modificaciones y refactorizaciones.

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, encargadas de verificar el código de forma automática y las pruebas de aceptación, destinadas a evaluar si al final de una iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente. (Gutiérrez, 2010)

3.3.1 Pruebas de aceptación

Las pruebas de aceptación, son especificadas por el cliente y se centran en las características y funcionalidades generales del sistema que son visibles y revisables por parte del cliente. Las pruebas de aceptación derivan de las historias de usuario que se han implementado como parte de la liberación del software.

En ellas se especifican la perspectiva del cliente, y los escenarios para probar que la HU ha sido implementada correctamente.

A continuación se muestran las pruebas de aceptación a realizarse en cada iteración:

Caso de Prueba de Aceptación	
Código: HU1_P1	Historia de Usuario: 1
Nombre: Mostrar interfaz de usuario	
Descripción: Se verifica que la interfaz de usuario se muestra correctamente.	
Condiciones de Ejecución: El juego debe estar en ejecución.	
Entradas/Pasos de ejecución: Se revisa la información mostrada y se comprueban que al pulsar los botones se realicen las acciones correctamente.	
Resultado esperado: Se muestra la interfaz de usuario correctamente.	

Tabla 53: Caso de prueba de Aceptación HU1_P1.

Caso de Prueba de Aceptación	
Código: HU1_P2	Historia de Usuario: 1
Nombre: Mostrar interfaz en diferentes resoluciones de pantalla.	
Descripción: Se verifica que la interfaz de usuario se muestre correctamente en diferentes resoluciones de pantalla.	
Condiciones de Ejecución: El juego no debe estar en ejecución.	
Entradas/Pasos de ejecución: Se modifica la resolución de pantalla a los siguientes valores 800x600, 1024x600, 1024x768, 1280x720 y 1280x800. Se ejecuta el juego y se comprueba que la interfaz de usuario no sufre ningún cambio visual.	
Resultado esperado: Se muestra la interfaz de usuario correctamente.	

Tabla 54: Caso de prueba de Aceptación HU1_P2.

Caso de Prueba de Aceptación	
Código: HU2_P1	Historia de Usuario: 2
Nombre: Seleccionar personaje.	
Descripción: Se selecciona un personaje para jugar.	
Condiciones de Ejecución: El juego se encuentra en ejecución.	
Entradas/Pasos de ejecución: El jugador selecciona uno de los personajes para jugar. Luego selecciona la opción de comenzar el combate.	
Resultado esperado: Se selecciona el personaje correctamente.	

Tabla 55: Caso de prueba de Aceptación HU2_P1

Caso de Prueba de Aceptación	
Código: HU3_P1	Historia de Usuario: 3
Nombre: Cerrar proceso correctamente.	
Descripción: Se cierra el juego terminando así la partida.	
Condiciones de Ejecución: El juego se encuentra en ejecución.	
Entradas/Pasos de ejecución: El jugador selecciona la opción Cerrar el juego.	
Resultado esperado: Se cierra correctamente el juego.	

Tabla 56: Caso de prueba de Aceptación HU3_P1.

Caso de Prueba de Aceptación	
Código: HU4_P1	Historia de Usuario: 4
Nombre: Iniciar combate.	
Descripción: Se sitúan los personajes en el escenario de batalla y se inicia el combate.	
Condiciones de Ejecución: El juego se encuentra en ejecución y se seleccionó el personaje con que se desea jugar.	
Entradas/Pasos de ejecución: El jugador selecciona la opción Comenzar partida.	
Resultado esperado: Se carga correctamente el escenario principal conteniendo el personaje seleccionado y el contrincante al que enfrentará y se inicia la partida.	

Tabla 57: Caso de prueba de Aceptación HU4_P1.

Caso de Prueba de Aceptación	
Código: HU5_P1	Historia de Usuario: 5
Nombre: Finalizar combate.	
Descripción: Se finaliza el combate.	
Condiciones de Ejecución: El juego se encuentra en ejecución y existe un combate activo.	
Entradas/Pasos de ejecución: El jugador selecciona la opción Finalizar partida.	
Resultado esperado: Se finaliza la partida y se retorna al Menú Principal.	

Tabla 58: Caso de prueba de Aceptación HU5_P1.

Caso de Prueba de Aceptación	
Código: HU6_P1	Historia de Usuario: 6
Nombre: Activar cambio de escena.	
Descripción: Se oculta la escena del Menú Principal y se activa la escena de combate.	
Condiciones de Ejecución: El juego se encuentra en ejecución.	
Entradas/Pasos de ejecución: Se presiona el botón Comenzar partida para cambiar a la escena del combate.	
Resultado esperado: Se activa la escena de combate correctamente.	

Tabla 59: Caso de prueba de Aceptación HU6_P1.

Caso de Prueba de Aceptación	
Código: HU7_P1	Historia de Usuario: 7
Nombre: Activar botones de acción.	
Descripción: Se activan los botones mediante los cuales se controlará al caracter.	
Condiciones de Ejecución: El juego se encuentra en ejecución.	
Entradas/Pasos de ejecución: Se presiona el botón Comenzar partida y automáticamente se activarán los botones de acción.	
Resultado esperado: Se activan los botones de acción correctamente.	

Tabla 60: Caso de prueba de Aceptación HU7_P1.

Caso de Prueba de Aceptación

Código: HU8_P1	Historia de Usuario: 8
Nombre: Activar animaciones.	
Descripción: Se activan las animaciones del caracter.	
Condiciones de Ejecución: El juego se encuentra en ejecución y se inició el combate.	
Entradas/Pasos de ejecución: Se presiona un botón de acción y automáticamente debe activarse la animación correspondiente a dicha acción.	
Resultado esperado: Se activan las animaciones correctamente.	

Tabla 61: Caso de prueba de Aceptación HU8_P1.

Caso de Prueba de Aceptación	
Código: HU9_P1	Historia de Usuario: 9
Nombre: Ejecutar ataque básico (jugador).	
Descripción: El caracter ejecuta un ataque básico.	
Condiciones de Ejecución: El juego se encuentra en ejecución y se inició el combate.	
Entradas/Pasos de ejecución: Se presiona un botón de acción y el carácter ejecuta un ataque básico.	
Resultado esperado: Se ejecuta el ataque correctamente.	

Tabla 62: Caso de prueba de Aceptación HU9_P1.

Caso de Prueba de Aceptación	
Código: HU10_P1	Historia de Usuario: 10
Nombre: Ejecutar combo de golpes (jugador).	
Descripción: El caracter ejecuta un combo de golpes.	
Condiciones de Ejecución: El juego se encuentra en ejecución y se inició el combate.	
Entradas/Pasos de ejecución: Se presiona un botón de acción y el carácter ejecuta un combo de golpes.	
Resultado esperado: Se ejecuta el combo de golpes correctamente.	

Tabla 63: Caso de prueba de Aceptación HU10_P1.

Caso de Prueba de Aceptación	
Código: HU11_P1	Historia de Usuario: 11

Nombre: Ejecutar ataque básico (oponente).
Descripción: El oponente ejecuta un ataque básico.
Condiciones de Ejecución: El juego se encuentra en ejecución y se inició el combate.
Entradas/Pasos de ejecución: El oponente ejecuta un ataque básico.
Resultado esperado: Se ejecuta el ataque correctamente.

Tabla 64: Caso de prueba de Aceptación HU11_P1.

Caso de Prueba de Aceptación	
Código: HU12_P1	Historia de Usuario: 12
Nombre: Ejecutar combo de golpes (oponente).	
Descripción: El oponente ejecuta un combo de golpes.	
Condiciones de Ejecución: El juego se encuentra en ejecución y se inició el combate.	
Entradas/Pasos de ejecución: El oponente ejecuta un combo de golpes.	
Resultado esperado: Se ejecuta el combo de golpes correctamente.	

Tabla 65: Caso de prueba de Aceptación HU12_P1.

Caso de Prueba de Aceptación	
Código: HU13_P1	Historia de Usuario: 13
Nombre: Añadir sistema de colisiones.	
Descripción: Permite adicionar a un personaje los componentes necesarios para la detección de colisiones.	
Condiciones de Ejecución: Debe haber un personaje seleccionado. El personaje no debe contener componentes para la detección de colisiones.	
Entradas/Pasos de ejecución: El usuario selecciona el personaje. Se adicionan los componentes "Box Collider", "Capsule Collider", "Sphere Collider" y "Rigidbody".	
Resultado esperado: Se añade el sistema de colisiones correctamente.	

Tabla 66: Caso de prueba de Aceptación HU13_P1.

Caso de Prueba de Aceptación	
Código: HU14_P1	Historia de Usuario: 14
Nombre: Activar efectos de sonido.	
Descripción: Permite activar los efectos de sonido en el combate.	
Condiciones de Ejecución: Debe existir un combate en iniciado.	
Entradas/Pasos de ejecución: Se activan las teclas de acción del caracter. Se reproducen los clips de audio referentes a las diferentes acciones ejecutadas.	
Resultado esperado: Se activaron los efectos de sonido correctamente.	

Tabla 67: Caso de prueba de Aceptación HU14_P1.

Caso de Prueba de Aceptación	
Código: HU15_P1	Historia de Usuario: 15
Nombre: Activar efectos especiales.	
Descripción: Permite activar los efectos especiales en el combate.	
Condiciones de Ejecución: Debe existir un combate en iniciado.	
Entradas/Pasos de ejecución: Se activan las teclas de acción del caracter. Se accionan los efectos especiales relacionados con cada acción ejecutada.	
Resultado esperado: Se activaron los efectos especiales correctamente.	

Tabla 68: Caso de prueba de Aceptación HU15_P1.

3.4 Conclusiones parciales

En el presente capítulo se abordaron todos los temas referentes a la implementación de la solución y a las pruebas aplicadas durante la elaboración del mismo. Se desarrollaron las tareas correspondientes para dar solución a las historias de usuario y las pruebas de aceptación que propician al cliente conformidad y seguridad ante el sistema. Con la finalización de este capítulo se da por terminada la propuesta de solución del prototipo de videojuego.

Conclusiones

Tras haber culminado el presente trabajo de diploma se arribó a las siguientes conclusiones:

- El presente trabajo de diploma permite definir paso a paso un proceso de desarrollo de videojuegos de lucha el cual puede ser aplicado al desarrollo de otros videojuegos.
- Se logró la implementación de un prototipo de videojuego de lucha para el repositorio de videojuegos del centro VERTEX el cual posee los elementos base que todo videojuego de este género debe poseer.
- La utilización de las metodologías Huddle y XP es una buena alternativa para el desarrollo de videojuegos debido a Huddle permite definir elementos técnicos y artísticos propios de un videojuego y XP describe un proceso iterativo e incremental, ideal para equipos de desarrollo pequeños, basándose en la realimentación continua entre el cliente y el equipo de desarrollo.
- Se demostró que Unity 3D es una buena alternativa para la creación de videojuegos de tipo lucha debido a las facilidades que ofrece para el trabajo con objetos 3D, a los cuales se les puede modificar su comportamiento mediante el uso de *scripts* y componentes nativos de dicho motor.

Recomendaciones

A continuación, se reflejan algunas recomendaciones que se consideran necesarias para futuras versiones del producto:

- Agregar nuevos personajes con diferentes características y estilos de pelea.
- Mejorar la calidad visual y las animaciones de los personajes utilizados en el prototipo de videojuego.
- Incorporar nuevas reglas de IA para realizar el ataque del oponente.

Referencias bibliográficas

FACHIN, G. G. La importancia del juego en la infancia, 2009. [2015]. Disponible en: <http://crianzapositiva.org/2009/08/la-importancia-del-juego-en-los-ninos/>

SEDEÑO, A. M. Videojuegos como dispositivos culturales: las competencias espaciales en educación. In *Comunicar*. 2010, vol. XVII, no. 34, p. 183-189.

BARINAGA, B. L. *Juego. Historia, teoría y práctica del diseño conceptual de videojuegos*. Edtion ed. Madrid: Alesia Games & Studies, 2010. ISBN 978-84-614-0894-8.

LEVIS, D. Los videojuegos, un fenómeno de masas. Barcelona: Editorial Paidós, 1997.

MARQUÈS, P. Videojuegos: efectos psicológicos. *Revista de Psiquiatría Infantil y Juvenil*, 2000, vol. 2, p. 106-116.

PROCHILE. Estudio de Mercado. Videojuegos en Estados Unidos, 2012. [Disponible en: http://www.prochile.gob.cl/wp-content/blogs.dir/1/files_mf/documento_12_11_12175027.pdf]

I GAVALDÀ, Jordi Duch; I ESTEVE, Josep Jorba; NAVARRO, Heliodoro Tejedor. Introducción a los videojuegos. UOC Universitat Oberta de Catalunya, 2008.

MERISTATION. La historia de los juegos de peleas, 2009. [Disponible en: <http://zonaforo.meristation.com/topic/1264647/>]

VICBENGAMES. Historia de los juegos de lucha, 2011. [2015]. [Disponible en: <http://vicbengames.blogspot.com/2011/10/historia-de-los-juegos-de-lucha.html>]

INTECO. Ingeniería del software: Metodologías y ciclos de vida. 2009

URRUTIA, Gerardo Abraham Morales, et al. Procesos de desarrollo para videojuegos. CULCyT, 2015, no 37.

RAYO, J. C. P. El uso de los videojuegos como motivador del aprendizaje autónomo de las matemáticas en los estudiantes de educación básica en Colombia, 2011.

KEITH, C. Advanced scrum and agile development. In Game Developer Conference, 2009.

TOBEY, B. Introducing scrum at large animal games: a look back at the first year of agile development. 2008. [Disponible en: <http://www.gamasutra.com/view/feature/3677/introducing>]

CRYTEK. Transition to scrum midway through a development cycle: Lessons learned. In Game Developer Conference. 2008.

NUTT, C. Living On The Edge: DICE's Owen O'Brien Speaks. 2008. [Disponible en: <http://www.gamasutra.com/view/feature/3684/living>]

LETELIER, Patricio. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). 2006.

BECK, K. *Extreme Programming Explained: Embrace Change*. Edtion ed.: Addison-Wesley Professional, 2004.

KEITH, C. Advanced scrum and agile development. In Game Developer Conference. 2009.

Flood, K. 2003. Game Unified Process. GameDev. 2003. [Disponible en: <http://www.gamedev.net/reference/articles/article1940.asp>]

KRUCHTEN, P. The rational unified process: an introduction. Addison-Wesley Professional, 2004.

Scrum Alliance. Learn About Scrum. 2009. [Disponible en: <https://www.scrumalliance.org/why-scrum>]

UNITY. ¿Qué es Unity? 2015. [Disponible en: <https://unity3d.com/es/unity>]

BECK, K. "Extreme Programming Explained. Embrace Change", Pearson Education, 1999. Traducido al español como: "Una explicación de la programación extrema. Aceptar el cambio", Addison Wesley, 2000.

SOMMERVILLE, I. *Ingeniería de software*. Edtion ed. Madrid: Pearson Educación, 2005. 712 p. ISBN 84-7829-074-5.

UNITY. Unity - Manual: Scripting. 2015. [2015]. [Disponible en: <http://docs.unity3d.com/es/current/Manual/ScriptingSection.html>]

GUTIÉRREZ, J. J. Pruebas del Sistema en Programación Extrema. 2010, pp. 12. [Disponible en: http://www.lsi.us.es/~javierj/investigacion_ficheros/PSISEXTREMA.pdf]