Cómo citar este artículo:
N. A. Uribe-Ordoñez, P. C. Ramírez-Prada, D. C. Tello-Beltrán, "Experiencias en la creación de un videojuego artístico como ejercicio interdisciplinario en un proyecto de investigación" Revista Ingeniería, Investigación y Desarrollo, vol. 19 (1), pp. 5-16, junio. 2019. .

EXPERIENCIAS EN LA CREACIÓN DE UN VIDEOJUEGO ARTÍSTICO COMO EJERCICIO INTERDISCIPLINARIO EN UN PROYECTO DE INVESTIGACIÓN

Experiences in the creation of an artistic video game as an interdisciplinary exercise in a research project

Nitae Andrés Uribe Ordoñez[1], Paulo César Ramírez Prada[1], Daniel Camilo Tello Beltrán[2].
[1]Universidad Autónoma de Bucaramanga – UNAB, Programa de Ingeniería de Sistemas.
E-mail: nuribe4@unab4edu.co, pramirez206@unab.edu.co,
[2]Universidad Autónoma de Bucaramanga – UNAB, Programa de Artes Audiovisuales.
E-mail: dtello@unab.edu.co

(Recibido Septiembre de 2018 y aceptado Febrero 08 de 2019)

Resumen

El siguiente documento tiene como propósito compartir los resultados de un primer ejercicio interdisciplinario entre los programas de Ingeniería de Sistemas y Artes Audiovisuales de la Universidad Autónoma de Bucaramanga tomando como temática integradora el diseño y desarrollo de un videojuego artístico en el marco de un proyecto de investigación en sentido estricto. Se exponen los detalles metodológicos llevados a cabo durante la ejecución del proyecto, las experiencias del trabajo entre artes e ingeniería y la descripción de los productos resultantes con el objetivo de que los mismos sirvan como bases para futuros trabajos en el área.

Palabras clave: *Programación, Videojuego, Motores de Juego, Desarrollo de Software, Diseño de Nivel.*

Abstract

The following paper aims to share the results of a first interdisciplinary exercise developed between the programs of Systems Engineering and Audiovisual Arts of the Universidad Autónoma de Bucaramanga taking as an integrative topic the development and design of an artistic videogame in the context of a research project in the strict sense. The research sets out the methodological details carried out during the execution of the project, the experiences of work between the arts and engineering and the description of the resultant products to possibility all of them to serve as bases for future projects in this area.

Key words: *Programming, Videogames, GameEngine, Software Development, Level Design.*

1. INTRODUCCIÓN

La industria de los videojuegos es un sector en constante crecimiento, actualmente ocupa el primer lugar como la industria que más ingresos genera sobre la industria del cine y la música. El desarrollo de videojuegos, sin embargo, a diferencia de sus pares en el sector del entretenimiento requiere de un proceso creativo que incluye a las ciencias de la computación, la ingeniería del software y el pensamiento algorítmico como bases para su ejecución, pues sin estas bases no sería posible la creación de un videojuego. Desde la Ingeniería podemos observar los videojuegos como un desarrollo de software tradicional con puntos en común con las artes audiovisuales donde se busca explorar el arte y la ingeniería como modelo creativo en el desarrollo de métodos y técnicas para la creación de videojuegos

que produzcan experiencias significativas en los usuarios, razón por la cual se ha propuesto, en el marco de la convocatoria bienal de investigación 2015-2016 de la Universidad Autónoma de Bucaramanga, el desarrollo de un proyecto de investigación titulado: "Creación de un Videojuego Artístico (CREVA)" para realizar una investigación formal en el proceso de creación de videojuegos como parte de un trabajo interdisciplinar de arte e ingeniería, conformando un equipo de investigadores de los programas de Ingeniería de Sistemas y Artes Audiovisuales de la Universidad Autónoma de Bucaramanga.

2. METODOLOGÍA

Para el desarrollo del proyecto se conformó un equipo interdisciplinario, específicamente en artes e ingeniería, cada parte aportó las habilidades y conocimientos requeridos para la construcción del proyecto, así también, se propuso una metodología que permitiera la integración de las habilidades y conocimientos del equipo interdisciplinario definiendo para tal fin 3 fases que se enlistan a continuación:

Fase 1 Experimentación y Prototipado:
En esta primera fase se trabajó en explorar mecánicas innovadoras de juego mediante el desarrollo de 3 prototipos funcionales, para ello se utilizaron elementos de la metodología Rapid Game Development [1], la cual busca reducir los tiempos en procesos de desarrollo de juegos, factor determinante debido a que los procesos creativos suelen tomar más tiempo del estimado.

En esta fase se dio prioridad al resultado sobre el proceso, el código de los prototipos será en su mayoría descartado y las imágenes recreadas, por lo que el enfoque será cumplir con los requerimientos a tiempo, sin tener en cuenta escalabilidad o refactorización. En esta etapa la calidad del producto depende, principalmente, de la habilidad de los desarrolladores. A nivel artístico, estos prototipos exploraron técnicas como: stopmotion, pixelacion, cut out, animación 2D, entre otras técnicas. También se exploraron pinceles o texturas que generarán diferentes sensaciones, para crear atmosferas diversas y ensayar si son compatibles con el motor de render seleccionado para los prototipos. Para cada prototipo se realizaron 5 pasos en su definición:

1. Lluvia de ideas, para definir aspectos generales del juego.
2. Definición de la mecánica del juego, para definir limitaciones y alcances de los motores gráficos usados.
3. Delimitación del alcance del prototipo, para delimitar según el tiempo disponible el alcance del prototipo.
4. Creación del estilo artístico, definición de los estilos artísticos al prototipo software a ser desarrollado.
5. Programación del prototipo, construcción y compilación del prototipo software definitivo.

Posteriormente, cuando los 3 prototipos fueron terminados, se socializaron con el equipo del proyecto, así mismo, se realizaron grupos focales para evaluar los prototipos y definir de esta manera la mecánica final y el estilo artístico del juego final según la evaluación hecha a cada aspecto de los prototipos.

Fase 2 Diseño y Desarrollo del Videojuego:
En esta fase se aplicaron los resultados obtenidos en los prototipos, tomando como referentes los mejores exponentes en cuanto a interactividad juego-usuario, de esta manera, se desarrollará un video juego en concreto, y con un motor de render específico.

El producto final será desarrollado con Agil Game Development [2] utilizando elemento de scrum, aprovechando la metodología con que se desarrollaron los prototipos, pero realizando varios ciclos o springs, se realizaran varias veces los pasos hasta terminar todas las mecánicas del juego, esto permite tener al final de cada ciclo un prototipo jugable del producto final:
1. Inicio del Spring
2. Lluvia de ideas
3. Definición de la mecánica del juego
4. Delimitación del alcance de la mecánica
5. Creación del estilo artístico

6. Programación de la mecánica
7. Regreso al inicio del Spring (paso 1)

Fase 3 Promoción y Divulgación:
Esta fase consistió en divulgar el videojuego una vez terminado, a fin de participar en eventos y festivales, para y, de esta manera, obtener registros cuantificables sobre los resultados que pudiese generar el producto y comprobar la efectividad del mismo como producto software multimedia con fines lúdicos. Como consecencia, se buscará comunicar los resultados obtenidos en medios regionales y nacionales. El presente documento expondrá esta experiencia desde el punto de vista del desarrollo de software principalmente en las fases 1 y 2 de la metodología propuesta para el proyecto.

3. GAME ENGINES.

Un motor de videojuegos consiste en una serie de programas, utilidades y rutinas de programación que facilitan el diseño, creación y representación de videojuegos en plataformas idóneas para su implementación. Las funcionalidades básicas de un motor de videojuegos son las de proveer un entorno de desarrollo específico para videojuegos que contemple funcionalidades básicas como son: renderizado en 2D y 3D, motor de físicas (colisiones, gravedad, etc), incorporación de sonido, scripting, animación, programación de inteligencia artificial, integración con protocolos de comunicación (redes, streaming), administración de memoria y escenografía gráfica. El desarrollo de un videojuego puede variar radicalmente con el uso de un motor gráfico, pues facilita la reutilización y adaptación de componentes presentes en el mismo motor a fin de representar videojuegos.

Podemos encontrar gran variedad de motores gráficos, existen motores open-source, como Ogre3D, cuyo uso es libre y se mantienen gracias a los aportes de una comunidad extensa de desarrolladores que aportan mejorar y depurar errores al motor. Igualmente, encontramos la oferta de motores propietarios que requieren del pago de una licencia o un acuerdo de beneficios por los productos desarrolladores, estos motores generalmente son ofrecidos por grandes estudios de videojuegos como Blam! (Bungie), Source (Valve) y CryEngine (Crytek). En el desarrollo de nuestro videojuego hemos optado por probar algunos de los motores más populares para su desarollo, los motores seleccionados fueron: GameMaker Studio (YoYo Games), Unreal Engine 4 (Epic Games) y Unity3D 5 (Unity Technologies).

3.1. Gamemaker Studio

GameMaker Studio es un motor de videojuegos orientado a programadores sin experiencia previa en videojuegos, provee un lenguaje de programación interpretado y un kit de desarrollo el cual permite desarrollar videojuegos de manera fácil y rápida [3].

Fue concebida en 1990 por el profesor Mark Overmars como una herramienta que permitiría a sus estudiantes aprender los conceptos básicos de la animación, con el tiempo este proyecto ha crecido como una herramienta de desarrollo de videojuegos principalmente en 2D. La licencia de uso es propietaria, sin embargo, cuenta con una versión gratuita la cual permite crear videojuegos para PC con ciertos límites.

La versión completa cuenta con utilidades como: compiladores especiales para publicar los juegos desarrollados a dispositivos móviles, y consolas de videojuegos. Así también, permite crear videojuegos sin necesidad de aprender lenguajes de programación como C++, cuenta con una serie de “bloques” que permiten programar los algoritmos necesarios para los juegos, no obstante, para usuarios experimentados ofrece su propio lenguaje llamado GML o Game Maker Language, lenguaje inspirado en Delphi y C++ que funciona por medio de scripts.

A pesar de ser una herramienta pensada para usuarios noveles, la calidad de los desarrollos resultantes se ajusta más a los conocimientos

del usuario. Cabe resaltar que los usuarios más experimentados optan por motores más elaborados, entre otras cosas que permitan la incorporación del 3D en los videojuegos.

3.2. Unreal Engine

Unreal Engine es un motor de videojuegos desarrollado por Epic Games en 1998, principalmente usado en videojuegos Shooter en primera persona, algunos desarrollos destacados del motor son: Unreal Tournament, Turok, Gears of War, Bioshock, Mass Effect, entre otros. Es un motor TOP en la industria, utiliza el lenguaje de programación C++ y es compatible con OpenGL así como con DirectX 11 y 12. Los desarrollos realizados en Unreal Engine son compatibles con plataformas como PC, dispositivos moviles y consolas de videojuegos. Es un motor gráfico orientado al 3D y ofrece herramientas no solo para programadores sino también para diseñadores y artistas [4].

El motor fue liberado de forma gratuita el 2 de marzo de 2015, siendo un motor de propietarios hasta la fecha, actualmente, la versión 4 del motor presenta un modelo de libre uso bajo un convenio con Epic Games donde se estipula que si los desarrollos son comercializados la compañía obtendrá el 5% de las ganancias del producto cuando el mismo supere los primeros 3.000 dólares.

3.3. Unity 3D

Unity3D es un motor de videojuegos desarrollado en 2004 por David Helgason, Nicholas Francis y Joachim Ante, desarrolladores que pusieron a disposición del público una herramienta de fácil acceso para la época, para cualquiera que quisiera desarrollar videojuegos. Desde entonces este ha sido un motor preferido por desarrolladores independientes. En la actualidad, su versión 5, unity ofrece dos versiones de licencia: Unity profesional la cual es propietaria y ofrece herramientas avanzadas para la creación de videojuegos y Unity Personal la cual es una versión gratuita pensada en pequeños desarrollos. Es un motor destacado en la industria, utiliza los lenguajes de programación C# y Javascript y al igual que Unreal es compatible con OpenGL así como DirectX 11 y 12, los desarrollos realizados con Unity son fácilmente exportables a plataformas como PC, moviles y consolas [5].

El aporte más destacado de Unity es el de democratizar el desarrollo de videojuegos, un área muy exclusiva y de difícil incorporación, al brindar las bases de lo que hoy en día es una comunidad de desarrolladores, capacitaciones, paquetes, prototipos y software construido por la comunidad. Producto de este trabajo, en 2010 se lanza la Asset Store, un portal web de recursos para el motor gráfico donde desarrolladores de todo el mundo podrán compartir y vender modelos, texturas, materiales, sistemas, música, efectos, tutoriales, extensiones y proyectos propios. Algunos desarrollos destacados en Unity3D son: Hearthstone: Heroes of Warcraft, Pillars of Eternity y Pokemon GO.

4. EXPERIMENTACIÓN Y PROTOTIPADO.

En el proceso de experimentación y desarrollo de prototipos se optó por probar los motores gráficos seleccionados a fin de conocer sus funcionalidades técnicas, igualmente, se seleccionó géneros de juegos que facilitarán el desarrollo del videojuego final de acuerdo al tiempo disponible para el mismo. Géneros como Rol, Shooter o Estrategia, fueron descartados pues su desarrollo es mucho más complejo. En el desarrollo de los prototipos se decidió no probar el motor Unity3D dada la experiencia previa del equipo de trabajo utilizándolo. Por esta razón, se dispusó producir los prototipos en Game Maker Studio y Unreal Engine, motor recientemente liberado para su uso de forma gratuita al momento de iniciar el proyecto.

4.1. Caffeinic Raptor

Game Engine: Unreal Engine 4
Género: Endless Runner [6]
Plataforma: PC

1. Descripcion del Juego:
Prototipo desarrollado utilizando assets de Infinity Blade y un personaje principal de creación propia modelado y animado en Maya. La jugabilidad se centra en esquivar obstáculos en un circuito infinito en el que se encuentran diferentes tipos de obstáculos e items. El objetivo principal del jugador es conseguir el mayor puntaje posible y como objetivo secundario correr la mayor distancia posible

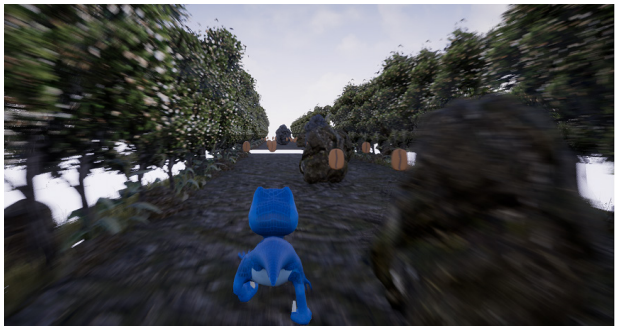


Figura 1. Personaje principal Caffeinic Raptor

-Personaje Principal: El protagonista del juego es un velociraptor azul adicto al café, corre por el escenario en búsqueda de granos de café, pero al comerlos la cafeína lo lleva a correr más rápido
-Item Grano de Café: Los granos de café llevan el puntaje del jugador, para tomarlos debe colisionar el velociraptor con ellos, esto, además de sumar al puntaje aumenta la velocidad del avatar.
-Obstáculo Piedra Pequeña: Esta piedra se destruye cuando el avatar dinosaurio choca con ella, reduciendo la velocidad de la carrera.
-Obstáculo Piedra Grande: Esta piedra de gran tamaño es indestructible y el jugador pierde si llega a colisionar con ella.
-Obstáculo Hueco: El jugador debe saltar sobre los huecos para evitar morir.

Controles: Flecha Izquierda: Mover el personaje hacia la izquierda, Flecha Izquierda (en una esquina): Girar 90 grados a la izquierda, Flecha Derecha: Mover el personaje hacia la Derecha, Flecha Derecha (en una esquina): Girar 90 grados a la Derecha, Espaciadora: Saltar. Instrucciones: 1. Evitar colisionar con las piedras grandes, 2. Evitar colisionar con las paredes, 3. Evitar caer por los huecos, 4. Recolectar la mayor cantidad de granos de café, 5. Colisionar con las piedras pequeñas para disminuir la velocidad, 6. Lograr sobrevivir la mayor cantidad de tiempo PC posible.

2.Dinámica de Juego:
Las dinámicas del juego están dadas por reglas sencillas que se codifican mediante Blueprints, este es un lenguaje propio de Unreal Engine que permite programar usando flujos de trabajo en lugar de código tradicional. Para ello Unreal Engine cuenta con un editor de flujos de trabajo (ver imagen) donde por medio de componentes de relaciones y nodos se realiza la lógica de programación para los objetos del juego.

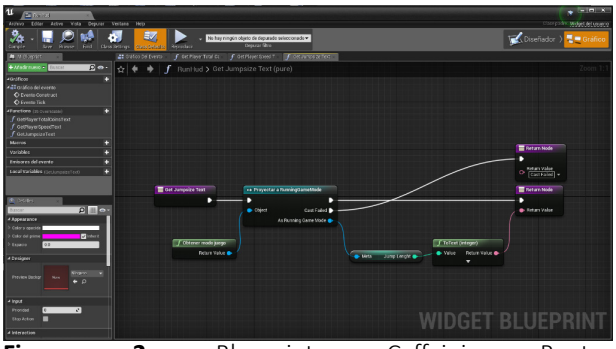


Figura 2. Blueprints Caffeinic Raptor

Las reglas que se definen para cafeinic raptos son:
1. Evitar colisionar con las piedras grandes.
2. Evitar colisionar con las paredes.
3. Evitar caer por los huecos.
4. Recolectar la mayor cantidad de granos de café.
5. Colisionar con las piedras pequeñas para disminuir la velocidad.
6. Lograr sobrevivir la mayor cantidad de tiempo posible.

Evento: Colisión con Pared
Este evento permite controlar si el objeto (self) que en este caso es el jugador (caffeinic raptor) ha colisionado con el componente pared, si esta condición se cumple el objeto (self) es destruido.

Evento: Destrucción de objeto
Este evento permite destruir un objeto referenciado

previamente, sea un grano de café cuando el jugador lo recoge o el jugador mismo al chocar con una pared o roca.

Evento: Recoger granos de Café
Este evento permite que el jugador recoja granos de café, al hacerlo se incrementa la velocidad del jugador y el contador de granos obtenidos por el jugador.

4.2. Soul Judgment

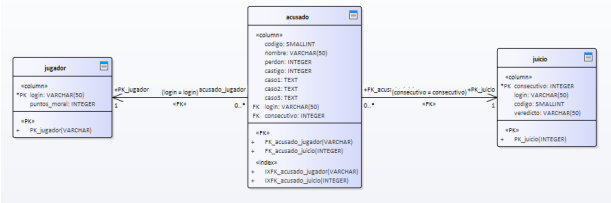
Game Engine: Unreal Engine 4
Género: Trivia [7]
Plataforma: Android

Este prototipo implementa servicios web para la comunicación entre el videojuego y una base de datos relacional alojada en un servidor en la nube de Microsoft Azure. La jugabilidad se centra en resolver diferentes juicios morales que se presentan al jugador de manera aleatoria, el juego guarda el veredicto del jugador sobre si el acusado es castigado (infierno) o salvado (cielo) y asigna un avatar que representa la moralidad de sus respuestas una vez ha evaluado a todos los acusados.

Figura 3. Pantalla de inicio y Arquitectura Cliente-Servidor de Soul Judgment

Definición de la Arquitectura:

1. Capa de Datos: El videojuego móvil accede los datos del juego como son juicios y acusados, asi como a los resultados de los veredictos en sesiones anteriores mediante la implementacion de servicios web de tipo REST en formato JSON. Los datos del juego están alojados en un servidor de bases de datos MySql y para ello se definió una esquema relacional descrito a continuacion:



• Acusado: contiene tres casos o acciones sobre el acusado, las cuales el jugador debe tener en cuenta al momento de realizar su veredicto, estos casos pueden ser acciones buenas, malas o neutrales las cuales el jugador debera realizar un juicio de valor a fin de emitir un veredicto. Esta contempla dos contadores los cuales iran sumando el número de veces que el acusado es castigado o perdonado para efectos estadísticos.
• Juicio: contiene datos de los veredictos realizados por los jugadores a cada uno de los acusados por cada una de las sesiones de cada jugador, además, esta informacion sirve para entregar una estadística sobre la incidencia de los juicios.
• Jugador: contiene datos del login del jugador y los puntos totales dentro de una sesión de juego. Permite posteriormente identificar los jugadores y sus estadísticas de juego.

2. Servicios Web: Los datos son accesibles por el videojuego desde un servidor remoto por medio de servicios web de tipo REST, esta implementación se ha realizado usando Java EE 7 y se encuentran alojados en un servidor de aplicaciones en la nube de Microsoft Azure, mediante la implementacion de un servidor Apache Tomcat 8.5, los servicios web usados son:
• Listar Acusados: Consulta la informacion de los acusados

con sus correspondientes estadísticas para ser presentadas en la app movil.
• Guardar Veredictos: Crear registro de juicio para los acusados de la sesión actual del jugador una vez se ha terminado la partida.
• Listas Estadísticas: Consulta de las estadísticas finales de sesiones anteriores una vez el jugador ha terminado su partida.

3. Interfaz Gráfica: La interfaz gráfica del juego esta desarrollada en Unreal Engine y corre sobre smarthphone o tablet con Android 2.2 o superior, esta aplicación movil se conecta al servidor web mediante servicios tipo REST.

Figura 5. Interfaces de Juego de Soul Judgment

Las instrucciones de uso del juego son:
• El juego comienza solicitando el nombre del jugador quien ejercerá como juez en el trascurso del juego. Este nombre servirá para comparar los resultados del usuario al final de la partida.
• El juego presenta cada una de las almas en pena mostrando una descripción de los actos positivos o negativos que realizaron en vida, el usuario podrá enviarlos al Cielo o al Infierno con el uso de dos botones en la zona inferior de la pantalla.
• Al finalizar cada caso el juego presentará las estadísticas de otras personas que ya han juzgado al personaje, mostrando el total de veces que el alma ha sido juzgada, y el porcentaje de cuantas veces han sido enviadas al Cielo o al Infierno.
• Al finalizar todos los casos, el juego termina mostrando al jugador una imagen alusiva a los juicios emitidos, según el índice de almas enviadas al Cielo o Inferno y según la similitud a los juicios realizados por otros usuarios.

4.3. Flying Fish Double Death

Game Engine: Game Maker Studio
Género: Endless Runner [6]
Plataforma: PC

Flying Fish Double Death es un prototipo desarrollado con gráficos en 2D animados cuadro a cuadro usando sprite sheets. Está basado en un fragmento de la serie The Hunt de la BBC, que se puede ver en https://www.youtube.com/watch?v=szuchBiLrEM, la jugabilidad se centra en recorrer el escenario esquivando peligros en el mar y en el aire, el jugador controla un pez volador y gana puntos según el tiempo de supervivencia.

Flying fish hunt - The Hunt: Episode 4 preview - BBC One

Figura 6. Flying Fish Hunt - BBC

Los peligros que encuentra en el mar son minas explosivas y un pez grande que busca comerse al pez volador, para evitar esto, el pez volador debe esconderse detrás de las minas para que el pez grande se choque con estas. Adicionalmente, el pez volador puede salir del agua y planear sobre la superficie, donde el pez grande no puede seguirlo.

Después de un tiempo en esta carrera, aparece un ave, que ataca al pez volador cuando este está fuera del agua, completando la sentencia de muerte para el pez volador, el jugador debe esforzarse por sobrevivir la mayor cantidad de tiempo posible, sin embargo, su muerte es inevitable.

Figura 7. Flying Fish Double Death - Gameplay

Controles: Flecha Arriba (en el agua): Nadar hacia arriba. Flecha Arriba Sostenida (en el aire): Planear. Instrucciones: 1. Evitar colisionar con las minas. 2. Evitar ser comido por el pez y por el ave. 3. Lograr sobrevivir la mayor cantidad de tiempo posible.

1. Diseño de Personajes: Los personajes del juego son animaciones cuadro a cuadro realizadas sobre el motor gráfico Game Maker Studio, para efectos de este videojuego se usaron fuentes de uso libre de bancos de Sprites de comunicades Open Source.

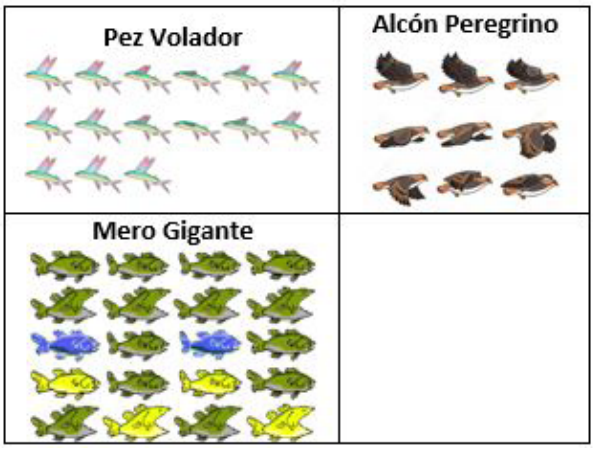


Figura 8. Sprites usados en Flying Fish Double Death

2. Dinámicas de Juego:

• Pez Volador: Es el personaje principal jugable, las dinámicas y controles están definidas en el código fuente del juego. El personaje puede moverse en todas las direcciones mientras esta en el agua, fuera de ella podrá planear hasta volver a ingresar al agua manteniendo la fecha arriba sostenida. A continuación, se presenta un fragmento del código del pez volador. El Pez Volador es el personaje controlado por el jugador, sus controles estan definidos así: Las flechas direccionales controlan la dirección del pez en el agua con una aceleración progresiva a medida que se mantiene presionada la tecla direccional. Si se acerca al borde del agua con suficiente velocidad el pez saltará fuera del agua y podrá planear por un tiempo limitado para escapar de los peligros del agua.

• Minas: Las minas ralentizaran el avance del pez volador y el pez mero gigante, si el pez volador toca 3 minas de manera consecutiva con un intervalo inferior a 15 segundos, el pez volador muere y termina la partida. Las minas se generan de manera aleatoria a medida que el pez avanza en el agua. A continuación, se presenta un fragmento del código del pez volador.

• Pez Mero Gigante: El pez mero gigante seguirá al pez volador mientras este se encuentre en el agua, su movimiento es de ½ la velocidad del pez volador, sin embargo, realiza ataques de carga contra el pez con una velocidad de x3 en línea recta, si el mero gigante alcanza al pez volador, este último es devorado y termina la partida. Cuando el pez volador se encuentra fuera del agua el mero gigante deja de perseguirlo y se retira al final de la pantalla. El pez mero se representa en una maquina de estados con 4 posibles estados:

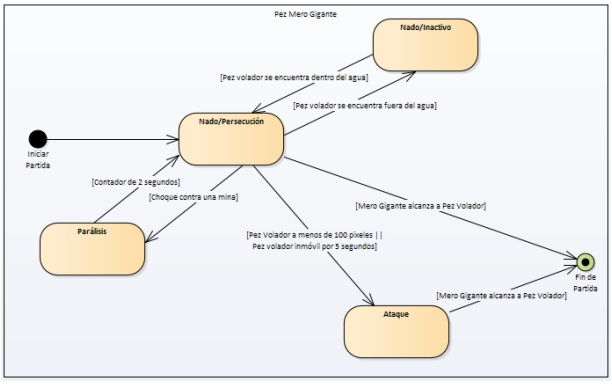


Figura 9. Diagrama de Estados – Pez Mero Gigante

-Nado/Inactivo: cuando el pez volador se encuentra fuera del agua, el pez regresa a la parte final del escenario donde permanecerá inactivo hasta que el pez volador reingrese al agua.
-Nado/Persecución: cuando el pez volador se encuentre dentro del agua, el pez comenzará a seguirlo en línea recta hacia la posición del pez volador, la velocidad del pez mero es ½ la velocidad del pez volador.
-Ataque: cuando el pez volador se encuentre dentro del agua y a un rango de menos de 100 pixeles de distancia o permanezca quieto por 5 segundos, el pez realizará un ataque de embestida hacia el pez volador.
-Parálisis: Cuando el pez mero gigante choque contra una mina este pausará su movimiento por 2 segundos.

• Alcón Peregrino: Aparece en el aire cuando el pez volador complete mas de 10 segundos de planear, el Alcón se mueve con una velocidad de x2 buscando al pez volador, si este es alcanzado es devorado y termina la partida. Una vez que el pez volador ingresa al agua, el Alcón se retira al cabo de 15 segundos. El Alcón no puede ingresar al agua ni perseguir al pez cuando este se encuentre dentro del agua. El Alcón se representa en una máquina con 3 posibles estados, así:

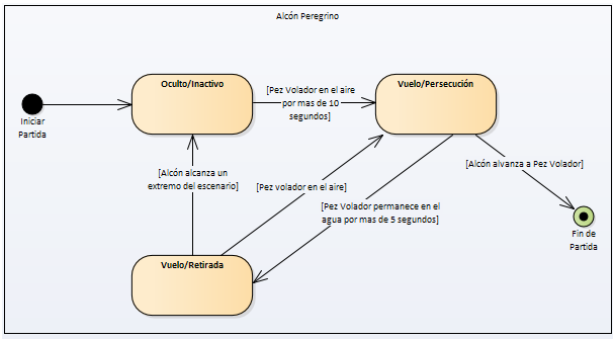


Figura 10. Diagrama de Estados – Alcón Peregrino

-Oculto/Inactivo: cuando el pez volador se encuentra dentro del agua, el Alcón permanece oculto a la vista del jugador.
-Vuelo/Persecución: cuando el pez volador se encuentre fuera del agua por mas de 10 segundos, el Alcón peregrino aparecerá aleatoriamente desde alguno de los bordes del escenario y comenzará a perseguir al pez volador.
-Vuelo/Retirada: cuando el pez volador ingrese al agua y al cabo de 5 segundos, el Alcón comenzara a retirarse hacia una de las esquinas del escenario más cercanas para desaparecer completamente.

4.4. Resultados Experimentación

Terminados los tres prototipos se procedió a analizar y seleccionar los mejores aspectos de cada uno para ser usados en el diseño y desarrollo del videojuego final:

Caffeinic Raptor: Este prototipo se destacó por presentar un personaje principal en 3D de elaboración propia, lo cual causó en el jugador mayor recordación, inclusive, por encima de los demás elementos presentes en el juego, se tomó entonces la decisión de usar elementos en 3D para el desarrollo del videojuego final.

Soul Judgment: Este prototipo se destaca por ser el único de los tres en exportarse a una plataforma móvil, lo cual permite su uso en este tipo de dispositivos con una amplia presencia en el mercado. Así también, fue significativa su interoperabilidad con un servidor web para guardar los progresos de los jugadores en distintos dispositivos. Otro punto a favor, fue la comparación de resultados como elemento rescatable para el proyecto final.

Flying Fish Double Death: Este prototipo se destacó por presentar una dinámica de juego que permitía contemplar todos los elementos en la pantalla del jugador, además de desarrollarse a un ritmo acelerado de toma rápida de decisiones. En este sentido, se tomó la decisión de usar para el videojuego final, una vista de cámara que permitirá ver la mayor cantidad de elementos del juego, lo que significó una vista en tercera persona y una cámara en ángulo cenital. De igual manera, se tuvo en cuenta el ritmo rápido del juego para ser usado en el producto final.

La siguiente tabla resume los aspectos destacados y cómo fueron usados en el videojuego final para dar inicio al diseño.

Tabla 1. Resultados experimentación

Prototipo	Aspectos Destacados	Aspectos Seleccionados
Caffeinic Raptor	Entorno y personajes en 3D de elaboración propia.	Todos
Soul Judgment	Plataforma móvil. Progreso del jugador entre dispositivos.	Se tomó únicamente el uso de plataformas móviles. Por cuestiones de tiempo se omite el guardar progreso entre dispositivos.
Flying Fish Double Death	Jugabilidad y ritmo de juego. Vista amplia de los elementos de juego.	Todos, se contempló el uso de una vista en tercera persona con cámara en ángulo cenital.

Fuente: autores

5. DISEÑO Y DESARROLLO DEL VIDEOJUEGO FINAL: MAGIC LECHONA.

Para el desarrollo del videojuego resultante de los prototipos se asignó 1 año de trabajo. Una vez concluido el proceso de los prototipos, se tomó la decisión de no continuar con ninguno de ellos, esto modificó la metodología planteada inicialmente. Esta decisión fue motivada con base en las siguientes 3 situaciones:

• Los prototipos se desarrollaron utilizando tecnologías nuevas para los investigadores, para el desarrollo final se decidió utilizar Unity 3D, que no fue incluido en los prototipos precisamente por la experticia con el mismo, la decisión de usar un Game Engine nuevo, abrió la puerta para diseñar un nuevo juego orientado a la fase final y descartar los prototipos.
• Al proyecto se vincularon 3 estudiantes de pregrado, de las carreras de Ingeniería de Sistemas, Artes Audiovisuales y Música. Diseñar un nuevo juego les permitió participar activamente en el desarrollo y sentir que el videojuego también era parte de su creación.
• El grupo de investigadores no estaba, en su totalidad, satisfecho con los resultados de los prototipos, por lo que no se encontraban altamentemotivados a continuarlos. Un nuevo juego despertó de nuevo el espíritu creador del equipo. Se procedió entonces a diseñar un nuevo juego y a estructurar su desarrollo desde la producción, enfrentando la difícil tarea de estimar el alcance de un videojuego que requirierá aproximadamente un año de desarrollo.

5.1. Diseño De Magic Lechona

El nuevo juego diseñado se llamó Magic Lechona, inspirado en el famoso "cerdo portter" de Los Simpson, es un juego del género Arcade Puzzle [8], donde el jugador debe lanzar huevos mágicos en el nivel para que estos se rompan, al interior de los huevos aparecen frutas las cuales se recolectan utilizando partes de un cerdo.

Figura 11. Tablero de juego Magic Lechona

El juego consta de 12 niveles, durante los cuales el jugador va recolectando frutas mágicas que puede usar para alimentar sus cerdos y multarlos con el fin de crear la lechona mágica más rara posible.

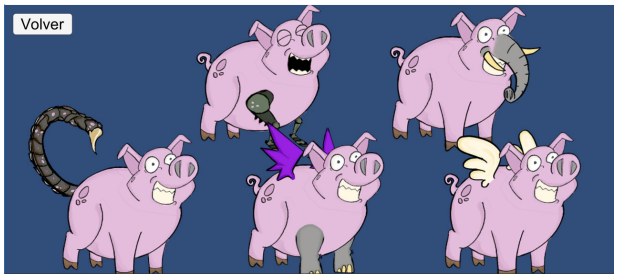


Figura 12. Mutaciones de la lechona mágica

5.2. Desarrollo De Magic Lechona

Para el desarrollo de Magic Lechona se logró una alianza con el estudio de desarrollo de videojuegos Fryos [9], que nació como spin off del semillero de investigación en videojuegos de la Universidad Autónoma de Bucaramanga, el estudio se unió a las labores de diseño, programación y arte, potenciando el equipo de trabajo y aportando nuevos puntos de vista para el juego. Como se mencionó anteriormente, el game engine seleccionado para Magic Lechona fue Unity 3D, debido a la experticia tanto del equipo investigador como de Fryos. El desarrollo presentó nuevos retos a nivel de producción, arte y programación.

En cuanto a producción, el principal reto fue lograr un workflow adecuado con un equipo de trabajo interdisciplinario, con niveles de experiencia muy diferentes, el cual involucró a estudiantes desde segundo a noveno semestre, investigadores académicos y desarrolladores de juegos profesionales. Para coordinar estos esfuerzos se utilizó la plataforma Trello y se hizo seguimiento constante a través de reuniones con miembros del equipo.

El principal aprendizaje desde el arte fue la falta de experiencia en desarrollo de videojuegos, ya que el modelador y animador principal del juego tuvo dificultades al momento de lograr los estándares para videojuegos, esto a pesar de ser un experto en cortometrajes y docente universitario de animación, confirmando que son dos especialidades distintas del arte audiovisual. La experiencia del estudio Fryos fue fundamental para agilizar el proceso artístico.

La experiencia de los programadores fue otro reto del equipo de trabajo, con miembros que contaba con un año de experiencia hasta miembros con 10 años, motivar a los programadores jóvenes y concienciar a los expertos fue la mejor estrategia para fomentar un ambiente de paciencia y aprendizaje.

El desarrollo de Magic Lechona se logró de manera satisfactoria. El resultado palpable, es un juego listo para ser publicado en las Play Store. Se presume que se ofrecerá gratis, pero no se descarta una posible monetización del mismo.

CONCLUSIONES

El desarrollo de videojuegos requiere el uso de metodologías de trabajo mucho más flexibles a las usadas en los procesos de desarrollo de software, al entrelazar el trabajo artístico con el de ingeniería se fomenta el uso de estrategias más creativas en la gestión del trabajo en equipo.

El uso de prototipos funcionales permitió evaluar mecánicas de juego y su impacto en el jugador de una forma más práctica y eficaz que con la utilización de instrumentos como encuestas o sondeos.

La fase de experimentación permitió al equipo adquirir nuevos conocimientos técnicos en motores gráficos novedosos, evaluar de una manera más cercana sus ventajas y desventajas a la hora de ser usado en un proyecto como el estipulado, con tiempos y recursos limitados. Dedicar los tiempos necesarios de evaluación, se convierte en una herramienta vital para salvar un proyecto o terminar en desastre.

Para desarrollar exitosamente un videojuego, se recomienda contar con personal con basta experiencia en el equipo de trabajo, preferiblemente, un líder experimentado en programación, arte y producción.

Tener un documento completo y detallado es fundamental para lograr un flujo de trabajo continuo durante el desarrollo.

La alianza universidad - empresa representa una relación de beneficio mutuo con excelentes resultados, siempre y cuando ambas partes estén sinceramente comprometidas con el proyecto.

La naturaleza artística de los videojuegos hace más complejo el proceso pre-producción en

comparación al desarrollo de software tradicional.

Los procesos de testing permiten tanto validar la temática y mecánicas de juego, como encontrar errores en el mismo.

REFERENCIAS

[1] R. Scott, Level Up!: The Guide to Great Video Game Design, John Wiley & Sons Inc, 2014.

[2] K. Clinton, Agile Game Development with Scrum, Addison-Wesley Educational Publishers Inc, 2010.

[3] YoYo Games Ltd., «GameMaker | YoYo Games,» 2017. [En línea]. Available: https://www.yoyogames.com. [Último acceso: 22 Mayo 2017].

[4] Epic Games, Inc., «What is Unreal Engine 4,» 2017. [En línea]. Available: https://www.unrealengine.com. [Último acceso: 22 Mayo 2017].

[5] Unity Technologies, «Unity - Game Engine,» 2017. [En línea]. Available: https://unity3d.com/es. [Último acceso: 22 Mayo 2017].

[6] TVTropes, «Endless Running Game - TV Tropes,» [En línea]. Available: http://tvtropes.org/pmwiki/pmwiki.php/Main/EndlessRunningGame. [Último acceso: 23 Junio 2017].

[7] Harper Collins Publishers Limited, «Definition of 'trivia game',» [En línea]. Available: https://www.collinsdictionary.com/dictionary/english/trivia-game. [Último acceso: 22 Junio 2017].

[8] Wikimedia Foundation, Inc, «Puzzle video game,» 17 Junio 2017. [En línea]. Available: https://en.wikipedia.org/wiki/Puzzle_video_game. [Último acceso: 22 Junio 2017].

[9] Fryos Studios, «FRYOS STUDIOS - Desarrolladores,» Fryos Studios, 2017. [En línea]. Available: http://fryosstudios.com. [Último acceso: 22 Junio 2017].

[10] Oracle, «Java Platform, Enterprise Edition: The Java EE 7,» 2014. [En línea]. Available: https://docs.oracle.com/javaee/7/tutorial/. [Último acceso: 22 Mayo 2017].

[11] Oracle, «Working with Jersey,» 2012. [En línea]. Available: https://jersey.github.io. [Último acceso: 22 Mayo 2017].

[12] BBC One, «Flying fish hunt- The Hunt: Episode 4 preview- BBC One,» BBC One, 17 Noviembre 2017. [En línea]. Available: https://www.youtube.com/watch?v=szuchBiLrEM. [Último acceso: 22 Junio 2017].