**DESARROLLO DE UN VIDEOJUEGO HACIENDO USO DEL FRAMEWORK “UNITY ETH UQ”**

**INFORME FINAL**

Presentado como requisito para obtener el título de Ingeniero de sistemas y computación, por:

Dival Mauricio Hoyos Castro

Julian David Serna Echeverri

Y supervisado por:

Christian Andrés Candela Uribe

Armenia, Septiembre 2016

A Dios, nuestros padres,

profesores, amigos y compañeros que

siempre nos brindaron su apoyo incondicional.

**Agradecimientos**

Queremos expresar nuestros agradecimientos a:

* Dios que nos ha dado la dedicación para poder cumplir con nuestras metas y sueños.
* A nuestras familias que siempre nos han brindado su cariño y apoyo incondicional.
* A nuestros compañeros que a lo largo de la carrera se han vuelto nuestros amigos con los cuales hemos vivido grandes momentos.
* A nuestro director, el ingeniero Christian Andrés Candela Uribe quien nos orientó y asesoro de una forma incondicional durante todo el desarrollo del proyecto.
* A nuestros asesores Luis Eduardo Sepúlveda Rodríguez y Julián Esteban Gutiérrez Posada, quienes nos ayudaron en la consecución de nuestro título.
* Gracias a todos los profesores que participaron en nuestro desarrollo profesional durante la carrera.

**Resumen**

El grupo de investigación en redes, Información y Distribución – GRID, de la universidad del Quindío, identificó la necesidad de realizar una validación externa al entregable principal del proyecto “*Framework de desarrollo de aplicaciones y juegos para sistemas IOS y Android en Unity con sistema de analíticas de uso y marketing*” cuyo nombre clave es “*Unity ETH UQ*”, el cual se desarrolló en conjunto con la empresa quindiana de desarrollo de videojuegos Ethereal GF y avalado por Colciencias, con el objetivo de identificar las características de los módulos *util*, *components*, *audio* y *analytics* de dicho framework, desarrollar un videojuego para probar su funcionamiento y finalmente obtener un reporte para mejorar la calidad del proyecto. Según Pressman (2010) el software se prueba para descubrir errores que se cometieron de manera inadvertida conforme se diseñó y construyó, además Jústiz Nuñez et al. (2014) concluyen que cuando el equipo de pruebas está integrado en su mayoría por especialistas externos al equipo de desarrollo se logra una mayor objetividad en la comprobación de los productos ya que se disminuye la posibilidad de omitir la prueba de alguna funcionalidad. En este sentido, se implementa el proceso descrito buscando probar los cinco módulos mencionados anteriormente por un equipo de trabajo externo a la organización que los implementó, a través de la realización del trabajo de grado: *Desarrollo de un videojuego haciendo uso del framework “Unity ETH UQ”.*

**Abstract**

The research group in networks, information and distribution – GRID, at Quindío University, identified the need of making an external validation to the main deliverable of project “*Framework de desarrollo de aplicaciones y juegos para sistemas IOS y Android en Unity con sistema de analíticas de uso y marketing*” whose code name is “*Unity ETH UQ*”, which was developed altogether with quindian videogame developer company Ethereal GF and endorsed by Colciencias, with the objective of identify the characteristics of modules *util*, *components*, *audio* and *analytics*, develop a videogame for testing its performance and finally get a report to improve the quality of the project. According Pressman (2010) software is tested for discovering mistakes that were made inadvertently during design and construction, also Jústiz Nuñez et al. (2014) concluded that when the testing team is composed of external experts greater objectivity in products proof is achieved because possibility of omit a functional test is lower. In this way, described process is implemented looking for test the five modules mentioned before by an external team to the organization that implemented them, through the realization of the thesis entitled: *Desarrollo de un videojuego haciendo uso del framework “Unity ETH UQ”.*

**CONTENIDO**

[1. INTRODUCCION 1](#_Toc460927321)

[2. CONTEXTO 2](#_Toc460927322)

[2.1 Planteamiento del problema 2](#_Toc460927323)

[2.2 Objetivos 2](#_Toc460927324)

[2.2.1 General 2](#_Toc460927325)

[2.2.2 ESPECÍFICOS 2](#_Toc460927326)

[2.3 Alcance y delimitación 2](#_Toc460927327)

[2.4 Presupuesto 3](#_Toc460927328)

[2.5 Marco teórico 5](#_Toc460927329)

[2.5.1 Videojuegos 5](#_Toc460927330)

[2.5.1.1 Desarrollo de videojuegos 6](#_Toc460927331)

[2.5.1.2 Motor de videojuegos 7](#_Toc460927332)

[2.5.1.3 Validación 7](#_Toc460927333)

[2.6 Cronograma 10](#_Toc460927334)

[3. DESARROLLO 14](#_Toc460927335)

[3.1 Fase 1: Análisis de las características, requerimientos y funcionalidades del framework 14](#_Toc460927336)

[3.1.1 Entrevistas 14](#_Toc460927337)

[3.1.1.1 Einer Zapata 15](#_Toc460927338)

[3.1.1.2 Andrés Herrera 15](#_Toc460927339)

[3.1.1.3 Andrés Carvajal 15](#_Toc460927340)

[3.1.2 Análisis de los módulos 15](#_Toc460927341)

[3.1.3 Identificación de las funcionalidades 16](#_Toc460927342)

[3.1.4 Requerimientos 16](#_Toc460927343)

[3.2 Fase 2: Diseño y construcción 17](#_Toc460927344)

[3.2.1 Selección de la metodología de desarrollo 17](#_Toc460927345)

[3.2.1.1 Establecer las guías para el análisis de decisiones 17](#_Toc460927346)

[3.2.1.2 Establecer los criterios de evaluación 17](#_Toc460927347)

[3.2.1.3 Métricas de los criterios 18](#_Toc460927348)

[3.2.1.4 Identificar las soluciones alternativas 18](#_Toc460927349)

[3.2.1.5 Evaluar las soluciones alternativas 21](#_Toc460927350)

[3.2.1.6 Seleccionar la solución 22](#_Toc460927351)

[3.2.2 Concepto 22](#_Toc460927352)

[3.2.2.1 Definir los aspectos de juego 22](#_Toc460927353)

[3.2.2.2 Definir los aspectos técnicos 29](#_Toc460927354)

[3.2.3 Planificación 30](#_Toc460927355)

[3.2.3.1 Planificación administrativa 30](#_Toc460927356)

[3.2.3.2 Especificación del videojuego 32](#_Toc460927357)

[3.2.4 Elaboración 34](#_Toc460927358)

[3.2.4.1 Iteración 1 35](#_Toc460927359)

[3.2.4.2 Iteración 2 37](#_Toc460927360)

[3.2.4.3 Iteración 3 40](#_Toc460927361)

[3.2.4.4 Iteración 4 42](#_Toc460927362)

[3.2.4.5 Iteración 5 44](#_Toc460927363)

[3.2.4.6 Iteración 6 47](#_Toc460927364)

[3.3 Fase 3: Pruebas 49](#_Toc460927365)

[3.3.1 Iteración 1 de pruebas 50](#_Toc460927366)

[3.3.1.1 Definir medio de distribución 50](#_Toc460927367)

[3.3.1.2 Definir verificadores beta 50](#_Toc460927368)

[3.3.1.3 Definir como se reportan los errores 50](#_Toc460927369)

[3.3.1.4 Identificación de incidencias 50](#_Toc460927370)

[3.3.1.5 Seguimiento de la iteración 51](#_Toc460927371)

[3.3.1.6 Cierre 51](#_Toc460927372)

[3.3.2 Iteración 2 de pruebas 51](#_Toc460927373)

[3.3.2.1 Definir medio de distribución 52](#_Toc460927374)

[3.3.2.2 Definir verificadores beta 52](#_Toc460927375)

[3.3.2.3 Definir como se reportan los errores 52](#_Toc460927376)

[3.3.2.4 Identificación de incidencias 52](#_Toc460927377)

[3.3.2.5 Seguimiento de la iteración 53](#_Toc460927378)

[3.3.2.6 Cierre 53](#_Toc460927379)

[3.3.3 Iteración 3 de pruebas 53](#_Toc460927380)

[3.3.3.1 Definir medio de distribución 54](#_Toc460927381)

[3.3.3.2 Definir verificadores beta 54](#_Toc460927382)

[3.3.3.3 Definir como se reportan los errores 54](#_Toc460927383)

[3.3.3.4 Identificación de incidencias 54](#_Toc460927384)

[3.3.3.5 Seguimiento de la iteración 55](#_Toc460927385)

[3.3.3.6 Cierre 55](#_Toc460927386)

[3.4 Fase 4: Validación 55](#_Toc460927387)

[3.4.1 Módulo audio 56](#_Toc460927388)

[3.4.1.1 Reproducir audio 56](#_Toc460927389)

[3.4.1.2 Reproducir un efecto 56](#_Toc460927390)

[3.4.1.3 Reproducir un efecto de forma cíclica 56](#_Toc460927391)

[3.4.1.4 Modificar el volumen de la música 57](#_Toc460927392)

[3.4.1.5 Modificar el volumen de los efectos 57](#_Toc460927393)

[3.4.1.6 Observaciones del módulo audio 57](#_Toc460927394)

[3.4.2 Módulo util 57](#_Toc460927395)

[3.4.2.1 Manejo de timer 58](#_Toc460927396)

[3.4.2.2 Definir si la plataforma en la que se ejecuta el aplicativo es móvil 58](#_Toc460927397)

[3.4.2.3 Modificar la visibilidad de un Game Object 58](#_Toc460927398)

[3.4.2.4 Obtener un Game Object hijo a partir de su nombre y su padre 58](#_Toc460927399)

[3.4.2.5 Observaciones del módulo util 59](#_Toc460927400)

[3.4.3 Módulo components 59](#_Toc460927401)

[3.4.3.1 Soporte de idiomas a imagen 59](#_Toc460927402)

[3.4.3.2 Soporte de idiomas a texto 59](#_Toc460927403)

[3.4.3.3 Reproducción de audio a un toggle 59](#_Toc460927404)

[3.4.3.4 Soporte de idiomas a botón 60](#_Toc460927405)

[3.4.3.5 Reproducción de audio a un botón 60](#_Toc460927406)

[3.4.3.6 Modificar el valor de progreso a una barra de progreso 60](#_Toc460927407)

[3.4.3.7 Animar la barra de progreso cuando se modifica su valor 61](#_Toc460927408)

[3.4.3.8 Cargar una escena 61](#_Toc460927409)

[3.4.3.9 Al cargar una escena crear un panel sobre el canvas y mostrar un cargador 61](#_Toc460927410)

[3.4.3.10 Obtener el lenguaje del sistema 61](#_Toc460927411)

[3.4.3.11 Observaciones del módulo components 61](#_Toc460927412)

[3.4.4 Módulo appSystem 63](#_Toc460927413)

[3.4.4.1 Inicializar el sistema de analíticas 63](#_Toc460927414)

[3.4.4.2 Registrar un evento en el sistema de analíticas 63](#_Toc460927415)

[3.4.4.3 Registrar una pantalla en el sistema de analíticas 64](#_Toc460927416)

[3.4.4.4 Obtener y modificar variables registradas en el sistema 64](#_Toc460927417)

[3.4.4.5 Observaciones del módulo AppsSystem 64](#_Toc460927418)

[4. RESULTADOS 64](#_Toc460927419)

[4.1 Identificar características y requerimientos del framework “Unity ETH UQ” 64](#_Toc460927420)

[4.1.1 Identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ” 64](#_Toc460927421)

[4.2 Realizar el diseño de un videojuego utilizando características del framework “Unity ETH UQ” 65](#_Toc460927422)

[4.2.1 GDD Risky jungle 65](#_Toc460927423)

[4.3 Desarrollar el videojuego utilizando el diseño realizado 65](#_Toc460927424)

[4.3.1 Selección de la metodología de desarrollo 65](#_Toc460927425)

[4.3.2 Definición de roles y responsabilidades 65](#_Toc460927426)

[4.3.3 Definición de herramientas técnicas 65](#_Toc460927427)

[4.3.4 Especificación de características 65](#_Toc460927428)

[4.3.5 Plantilla de desarrollo de iteración 65](#_Toc460927429)

[4.3.6 Documento de desarrollo de iteración 1 66](#_Toc460927430)

[4.3.7 Documento de desarrollo de iteración 2 66](#_Toc460927431)

[4.3.8 Documento de desarrollo de iteración 3 66](#_Toc460927432)

[4.3.9 Documento de desarrollo de iteración 4 66](#_Toc460927433)

[4.3.10 Documento de desarrollo de iteración 5 66](#_Toc460927434)

[4.3.11 Documento de desarrollo de iteración 6 66](#_Toc460927435)

[4.3.12 Plantilla de desarrollo de iteración de pruebas 66](#_Toc460927436)

[4.3.13 Documento de desarrollo de iteración de pruebas 1 67](#_Toc460927437)

[4.3.14 Documento de desarrollo de iteración de pruebas 2 67](#_Toc460927438)

[4.3.15 Documento de desarrollo de iteración de pruebas 1 67](#_Toc460927439)

[4.3.16 Proyecto Unity “Risky Jungle” 67](#_Toc460927440)

[4.3.17 APK del juego 67](#_Toc460927441)

[4.4 Evaluar la funcionalidad de los componentes del framework mediante el juego desarrollado 67](#_Toc460927442)

[4.4.1 Informe de hallazgos y validación del framework 67](#_Toc460927443)

[5. CONCLUSIONES Y TRABAJO FUTURO 68](#_Toc460927444)

[6. BIBLIOGRAFIA Y REFRENCIAS 68](#_Toc460927445)

**Listado de ilustraciones**

[Ilustración 1. Bosquejo del HUD. 23](#_Toc460922379)

[Ilustración 2. Bosquejo del menú en juego. 24](#_Toc460922380)

[Ilustración 3. Bosquejo del menú principal cuando existe una partida iniciada. 24](#_Toc460922381)

[Ilustración 4.Bosquejo del menú principal cuando no existe una partida iniciada. 25](#_Toc460922382)

[Ilustración 5. Bosquejo de la pantalla de configuración. 25](#_Toc460922383)

[Ilustración 6.Bosquejo de la pantalla de game over 26](#_Toc460922384)

[Ilustración 7. Boceto del personaje principal. 27](#_Toc460922385)

[Ilustración 8.Boceto del oso. 27](#_Toc460922386)

[Ilustración 9. Boceto del cocodrilo. 27](#_Toc460922387)

[Ilustración 10.Boceto de la serpiente. 28](#_Toc460922388)

[Ilustración 11. Boceto del tigre. 28](#_Toc460922389)

[Ilustración 12. Boceto de la cerbatana. 29](#_Toc460922390)

[Ilustración 13.Boceto de la lanza. 29](#_Toc460922391)

**Listado de tablas**

[Tabla 1. Funcionalidades del framework 16](#_Toc460922419)

[Tabla 2. Peso de los criterios de selección de la metodología de desarrollo 18](#_Toc460922420)

[Tabla 3. Posibles valores de cada criterio de selección 18](#_Toc460922421)

[Tabla 4. Comparación entre las posibles soluciones 22](#_Toc460922422)

[*Tabla 5. Evaluación de las posibles soluciones* 22](#_Toc460922423)

[Tabla 6. Asignación de roles al equipo de trabajo 32](#_Toc460922424)

[Tabla 7. Especificación, priorización y asignación de características del videojuego 34](#_Toc460922425)

[Tabla 8. Características seleccionadas para la iteración 1 35](#_Toc460922426)

[Tabla 9. Descomposición de las características de la primera iteración en tareas 37](#_Toc460922427)

[Tabla 10. Características seleccionadas para la iteración 2 38](#_Toc460922428)

[Tabla 11. Descomposición de las características de la segunda iteración en tareas 39](#_Toc460922429)

[Tabla 12. Características seleccionadas para la iteración 3 40](#_Toc460922430)

[Tabla 13. Descomposición de las características de la tercera iteración en tareas 41](#_Toc460922431)

[Tabla 14. Características seleccionadas para la iteración 4 42](#_Toc460922432)

[Tabla 15. Descomposición de las características de la cuarta iteración en tareas 43](#_Toc460922433)

[Tabla 16. Características seleccionadas para la iteración 5 45](#_Toc460922434)

[Tabla 17. Descomposición de las características de la quinta iteración en tareas 46](#_Toc460922435)

[Tabla 18. Características seleccionadas para la iteración 6 47](#_Toc460922436)

[Tabla 19. Descomposición de las características de la quinta iteración en tareas 48](#_Toc460922437)

[Tabla 20. Verificadores beta de la primera iteración 50](#_Toc460922438)

[Tabla 21.Errores identificados y priorizados de la iteración 1 de la fase de pruebas 51](#_Toc460922439)

[Tabla 22. Seguimiento a los errores identificados en la iteración 1 de la fase de pruebas 51](#_Toc460922440)

[Tabla 23. Errores identificados y priorizados de la iteración 2 de la fase de pruebas 53](#_Toc460922441)

[Tabla 24. Seguimiento a los errores identificados en la iteración 2 de la fase de pruebas 53](#_Toc460922442)

[Tabla 25. Errores identificados y priorizados de la iteración 3 de la fase de pruebas 55](#_Toc460922443)

[Tabla 26. Seguimiento a los errores identificados en la iteración 3 de la fase de pruebas 55](#_Toc460922444)

[Tabla 27. Funcionalidades implementadas del módulo audio 56](#_Toc460922445)

[Tabla 28. Funcionalidades implementadas del módulo util 58](#_Toc460922446)

[Tabla 29. Funcionalidades implementadas del módulo util 59](#_Toc460922447)

[Tabla 30. Funcionalidades implementadas del módulo appsSystem 63](#_Toc460922448)

# INTRODUCCION

Según Myers, Sandler, & Badgett (2011), en un mundo ideal, se buscaría probar cada permutación posible de un programa, pero en la mayoría de los casos esto no es posible, y de serlo tomaría mucho tiempo y consumiría muchos recursos. Además el encargado de las pruebas necesita una actitud apropiada para lograr probar una aplicación con éxito, es por esto que hacen énfasis en que las pruebas de software son una tarea técnica, pero que además envuelve importantes consideraciones económicas y de psicología humana.

Polo Usaola (2006) define que el estándar ISO/IEC 12207 (ISO/IEC 1995) identifica tres grupos de procesos en el ciclo de vida del software, los cuales son:

* Procesos principales, grupo en el que incluye los procesos de Adquisición, Suministro, Desarrollo, Operación y Mantenimiento.
* Procesos de la organización, en donde se encuentran los procesos de Gestión, Mejora, Infraestructura y Formación.
* Procesos de soporte o auxiliares, en donde están los procesos de Documentación, Gestión de la Configuración, Auditoría, Resolución de Problemas, Revisión Conjunta, Aseguramiento de la Calidad, Verificación, Validación.

Pero no define un proceso de pruebas como tal, sino que aconseja, durante la ejecución de los procesos principales o de la organización, utilizar los procesos de soporte. Entre éstos se encuentran los procesos de Validación y de Verificación:

* El proceso de Validación tiene como objetivo determinar si los requisitos y el sistema final cumplen los objetivos para los que se construyó el producto, respondiendo así a la pregunta ¿el producto es correcto?
* El proceso de Verificación intenta determinar si los productos software de una actividad se ajustan a los requisitos o a las condiciones impuestas en actividades anteriores. De este modo, la pregunta a la que responde este proceso es ¿se está construyendo el producto correctamente?

En el presente documento se muestra la realización de una validación a los componentes útil, components, audio y analytics del proyecto de investigación “Framework de desarrollo de aplicaciones y juegos para sistemas IOS y Android en Unity con sistema de analíticas de uso y marketing”, cuyo nombre clave es “Unity ETH UQ”, mediante la realización de un videojuego de tipo *side-scroller* en dos dimensiones (2D) para el sistema operativo Android. A continuación se da una descripción de los capítulos que componen este documento.

# CONTEXTO

## Planteamiento del problema

El Grupo de Investigación en Redes, Información y Distribución – GRID, de la universidad del Quindío, realizó en conjunto con la empresa de desarrollo de videojuegos Ethereal GF un proyecto de investigación avalado por Colciencias llamado “*Framework de desarrollo de aplicaciones y juegos para sistemas IOS y Android en Unity con sistema de analíticas de uso y marketing*” cuyo nombre clave es “Unity ETH UQ”. Este framework ha sido incluido en todos los proyectos que ha realizado Ethereal GF, durante los cuales ha estado sometido a cambios y mejoras que lo han llevado a convertirse en una solución adaptada para la empresa, es por esto que surge la necesidad de realizar una validación externa para analizar el estado actual del framework.

## Objetivos

### General

Desarrollar un videojuego haciendo uso del framework “Unity ETH UQ” elaborado por ETHEREAL GF con el fin de evaluar sus principales funcionalidades.

### ESPECÍFICOS

* Identificar características y requerimientos del framework “Unity ETH UQ”.
* Realizar el diseño de un videojuego utilizando características del framework “Unity ETH UQ”.
* Desarrollar el videojuego utilizando el diseño realizado.
* Evaluar la funcionalidad de los componentes del framework mediante el juego desarrollado.

## Alcance y delimitación

El alcance de este proyecto está delimitado por la elaboración de un videojuego para la plataforma Android, que permita evaluar los componentes útil, display, audio y analytics del proyecto “Framework de desarrollo de aplicaciones y juegos para sistemas IOS y Android en Unity con sistema de analíticas de uso y marketing” desarrollado por la empresa ETHEREAL GF y cuyo nombre clave es “Unity ETH UQ”.

Durante el desarrollo del proyecto se elaboraron los siguientes entregables:

* Documento de anteproyecto.
* WBS
* Documento detallando los requerimientos, funcionalidades, ventajas y desventajas que posee el framework.
* Documento de selección de metodología de desarrollo (Análisis DAR).
* Documento GDD.
* Documento de especificación de herramientas.
* Bocetos, diseños y modelado de niveles, objetos y personajes.
* Documento de planificación de iteraciones durante el desarrollo del juego.
* Documento de selección de la metodología de pruebas.
* Documento de planificación de pruebas.
* Documento de reporte de incidencias en la fase de pruebas.
* Documento de cierre del desarrollo del videojuego.
* Informe de la validación de los componentes del framework.
* Informe final del trabajo de grado.
* Informes mensuales durante la ejecución del proyecto.
* Aplicativo del videojuego.

En lo referente a la temática de juego, aunque es una especificación secundaria para el contexto del presente documento, la cual estará detallada en el GDD, cabe destacar que será un juego de entretenimiento, mas no de un juego serio (*serious game*), de tipo *side scroller* del género acción-aventura *survival*, el cual contará con un mínimo de 3 niveles, entre ellos un nivel introductorio para que el usuario se familiarice con los controles, el estilo de juego y los personajes, los cuales serán, un personaje principal, con el cuál se deberán superar diferentes obstáculos, y al menos tres tipos de rivales que deberán ser derrotados.

## Presupuesto

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Rubros** | **Estudiantes** | | **Universidad del Quindío** | | **Total** |
|  | **Efectivo** | **Recurrente** | **Efectivo** | **Recurrente** |  |
| 1. **Personal** |  |  |  |  |  |
| Trabajo director: $49523 la hora, dedicación 4 horas semana, durante 6 meses |  |  |  | $7.131.312 | $7.131.312 |
| Trabajo Asesor 1: $49523 la hora, dedicación 4 horas semana, durante 6 meses |  |  |  | $7.131.312 | $7.131.312 |
| Trabajo Asesor 2: $49523 la hora, dedicación 4 horas semana, durante 6 meses |  |  |  | $7.131.312 | $7.131.312 |
| Trabajo Estudiante 1: $5300 la hora, dedicación 20 horas semana, durante 6 meses. |  | $2.544.000 |  |  | $2.544.000 |
| Trabajo Estudiante 2: $5300 la hora, dedicación 20 horas semana, durante 6 meses |  | $2.544.000 |  |  | $2.544.000 |
| 1. **Viajes** |  |  |  |  |  |
| Transporte reuniones del equipo: $10.000 por viaje, 3 viajes por semana, durante 6 meses |  | $720.000 |  |  | $720.000 |
| Transporte a las asesorías y revisiones: $10.000 por viaje, 1 viaje por semana, durante 6 meses |  | $240.000 |  |  | $240.000 |
| 1. **Bibliografía** |  |  |  |  |  |
| No aplica |  |  |  |  |  |
| 1. **Servicios Técnicos** |  |  |  |  |  |
| No aplica |  |  |  |  |  |
| 1. **Equipos** |  |  |  |  |  |
| Equipo 1: Samsung ultrabook 530U4B Serie 5 |  | $1’400.000 |  |  | $1’400.000 |
| Equipo 2: Asus X450C |  | $1’960.000 |  |  | $1’960.000 |
| 1. **Difusión** |  |  |  |  |  |
| No aplica |  |  |  |  |  |
| 1. **Propiedad Intelectual** |  |  |  |  |  |
| No aplica |  |  |  |  |  |
| 1. **Software** |  |  |  |  |  |
| No aplica |  |  |  |  |  |
| 1. **Materiales e insumos** |  |  |  |  |  |
| Fotocopias | $50.000 |  |  |  | $50.000 |
| Impresiones | $50.000 |  |  |  | $50.000 |
|  |  |  |  |  |  |
| **TOTALES** | $100.000 | $9.408.000 | $0 | $21.393.936 | $30.901.936 |

## Marco teórico

### Videojuegos

“Un videojuego es concebido como un medio de entretenimiento, en donde se incluye a uno o varios usuarios, llamados Players o simplemente Jugadores, los cuales mantienen una interacción constante con varias interfaces, como pueden ser los joysticks o controles, teclado, mouse, entre otros, y un dispositivo de video” (Arce, 2011). Dicho medio de entretenimiento suele tener una trama, la cual puede ser de diferentes tipos (fantasía, acción, lógica, deporte, miedo, etc.), que acompañada de diversos efectos multimedia atrae al jugador recibiendo su tiempo y dedicación para resolver problemas, eliminar obstáculos y superar metas con el objetivo de llegar al final del juego y obtener algún tipo de recompensa (Arce, 2011).

Según Benito García (2006) , el mercado de los videojuegos es uno de los sectores de la industria de la comunicación y entretenimiento que más factura, inclusive superando al cine en varios países. Lo cual se debe a su evolución, que ha llegado a combinar elementos de narración, música, animación y deporte donde el código es como una partitura musical la cual es tocada por una computadora y los juegos se vuelven a veces tan competitivos que se juegan como deporte (Morales Urrutia, Nava López, Fernández Martínez, & Rey Corral, 2010).

La competencia en las eras subsecuentes, la postura de la sociedad y el lugar en el que se encuentra la industria actualmente, implica equipos de desarrollo de 25 o más miembros para lograr publicar un solo videojuego comercial y los proyectos pueden durar más de 3 años y un presupuesto de millones de dólares. Un videojuego no sólo es un producto artístico, debe de pasar por varias fases desde que es concebido hasta que es olvidado, es decir, que como todo software, tiene un ciclo de vida (Morales Urrutia et al., 2010), el cual debe ser una secuencia de pasos organizados para llegar finalmente a un producto. Para garantizar que dicho producto esté construido de la mejor manera y que el proceso de su creación, independientemente de aspectos como el presupuesto o el tamaño del equipo de desarrollo, sea optimo han surgido metodologías de desarrollo y motores de videojuegos (frameworks), cuyo principal objetivo es democratizar tanto los procesos de desarrollo como el código.

#### Desarrollo de videojuegos

Genéricamente la ingeniería de software establece las siguientes fases, las cuales son independientes del área de aplicación y tamaño o complejidad del proyecto:

* La fase de definición se centra sobre el qué. Es decir, durante la definición, el que desarrolla el software intenta identificar qué información ha de ser procesada, qué función y rendimiento se desea, qué comportamiento del sistema, qué interfaces van a ser establecidas, qué restricciones de diseño existen, y qué criterios de validación se necesitan para definir un sistema correcto. Por tanto, han de identificarse los requisitos clave del sistema y del software. Aunque los métodos aplicados durante la fase de definición variarán dependiendo del paradigma de ingeniería del software (o combinación de paradigmas) que se aplique, de alguna manera tendrán lugar tres tareas principales: ingeniería de sistemas o de información, planificación del proyecto del software y análisis de los requisitos (Arce, 2011).
* La fase de **desarrollo** se centra en el ***cómo***. Es decir, durante el desarrollo un ingeniero del software intenta definir cómo han de diseñarse las estructuras de datos, cómo ha de implementarse la función dentro de una arquitectura de software, cómo han de implementarse los detalles procedimentales, cómo han de caracterizarse interfaces, cómo ha de traducirse el diseño en un lenguaje de programación (o lenguaje no procedimental) y cómo ha de realizarse la prueba. Los métodos aplicados durante la fase de desarrollo variarán, aunque las tres tareas específicas técnicas deberían ocurrir siempre: diseño del software, generación de código y prueba del software (Arce, 2011).
* La fase de **mantenimiento** se centra en el cambio que va asociado a la corrección de errores, a las adaptaciones requeridas a medida que evoluciona el entorno del software y a cambios debidos a las mejoras producidas por los requisitos cambiantes del cliente (Arce, 2011).

El proceso de desarrollo que se adopte depende del software que se esté construyendo, un proceso puede ser apropiado para crear software de un sistema de aviación, mientras que un proceso diferente por completo puede ser adecuado para la creación de un sitio *web*. El proceso define un marco de trabajo para un conjunto de áreas clave de proceso que se deben establecer para la entrega efectiva de la tecnología de la ingeniería del software. Las áreas claves del proceso forman la base del control de gestión de proyectos del software y establecen el contexto en el que se aplican los métodos técnicos, se obtienen productos del trabajo (modelos, documentos, datos, informes, formularios, etc.), se establecen hitos, se asegura la calidad y el cambio se gestiona adecuadamente (Arce, 2011). Debido a las fases genéricas y a las condiciones propias de cada proyecto han surgido metodologías de desarrollo, las cuales actualmente se dividen en dos grandes grupos, las robustas como *Rational Unified Process* (RUP), *Microsoft Solutions Framework* (MSF) o Métrica 3 y las ágiles como *Extreme Programming* (XP), *SCRUM* o *Lean Development* (LP).

Desarrollar un videojuego implica conocer de varias disciplinas y poseer varias habilidades, pasando por la creatividad y recorriendo varias ciencias formales y sociales, cuyo proceso va mucho más allá de un típico desarrollo de software. Así es como los videojuegos unifican el arte, la ciencia y la tecnología (Arce, 2011). Es por esto que las metodologías de desarrollo deben ser adaptadas específicamente al desarrollo de videojuegos, pero según Morales Urrutia et al. no existen procesos específicos para el desarrollo de videojuegos que sean públicos; posiblemente existan algunos procesos cerrados en el sentido de que su información, modelos, plantillas y herramientas no están disponibles al público en general y por lo tanto puede considerarse que existe la necesidad de modelos de procesos que puedan ser utilizados por la industria en general y que en su momento contribuyan al desarrollo de este tipo de aplicaciones.

Morales Urrutia et al. sostienen que la industria de los videojuegos se aferró muchos años a utilizar la metodología cascada y aún muchas compañías siguen creando productos de esta manera. Por lo cual proponen una metodología, la cual llamaron *Huddle* y cuyas fases se definen en la ilustración 1,que es una adaptación de las metodologías cascada y SCRUM, además la compañía de videojuegos uruguaya *Gemserk* liberó una adaptación de SCRUM la cual denominaron SUM. Por lo tanto se puede concluir que las metodologías de desarrollo para videojuegos son escasas, pero las pocas adaptaciones existentes brindan una guía muy sólida para equipos desarrolladores que quieran incursionar en la industria.

#### Motor de videojuegos

Un Motor de Videojuegos (en inglés Game Engine) es una aplicación de software que ofrece todas las herramientas necesarias para el diseño y desarrollo completo de un videojuego, disponiendo de un motor de renderizado para gráficos 2D y 3D, detector de colisiones, sonidos, scripting, animación, inteligencia artificial, redes, streaming, administración de memoria y mucho más (Arce, 2011). Existe actualmente una amplia gama de motores de videojuegos, con diferentes tipos de licencias y orientados a cumplir distintos tipos de propósitos. Se puede encontrar motores comerciales y gratuitos, con metodologías 2D o 3D, inclusive que brindan soluciones de juegos a variadas plataformas (Windows, Linux, android, etc.) (Arce, 2011).

Según Mocholí (2014), Unity 3D es una de las plataformas para desarrollar videojuegos más completas que existen debido a que permite la creación de juegos para múltiples plataformas a partir de un único desarrollo, incluyendo [el desarrollo de juegos](https://www.yeeply.com/) para consola (PlayStation, Xbox y Wii), escritorio (Linux, PC y Mac), navegador, móviles y tabletas (iOS, Android, Windows Phone y BlackBerry). Lo cual es una de las razones que llevó a Ethereal GF a utilizar este engine para el desarrollo de sus videojuegos y además a construir un framework que facilitara el desarrollo de ciertos componentes. Dicho framework consiste en un grupo de librerías, cada una con funciones específicas (componentes, audio, video, analíticas, entre otras), que se incluyen en el proyecto. Su utilización no solo tiene como fin la optimización del tiempo de desarrollo, sino garantizar la utilización de código de calidad.

#### Validación

La validación de cualquier proyecto es crucial para asegurar la calidad del producto y brindar la mejor experiencia posible al cliente o público objetivo. Como anteriormente se mencionó, la metodología cascada fue ampliamente utilizada en el desarrollo de videojuegos, y aún se usa en algunos proyectos, en dicha metodología la validación es un proceso que inicia junto con el desarrollo, pero la concentración de esfuerzos en esta es mínima y solo al final del desarrollo la validación toma un rol principal, lo cual en ocasiones conlleva a encontrar errores cuya magnitud fue creciendo a lo largo del desarrollo. Es por esto que las metodologías agiles, al ser iterativas incrementales en su mayoría, involucran la validación en cada iteración, encontrando así errores tempranamente y evitando que su magnitud aumente a lo largo del desarrollo. Quedando así la decisión de como efectuar la validación del proyecto en manos de sus planeadores.

Para Polo Usaola (2006) la fase de pruebas es una de las más costosas del ciclo de vida software, deben realizarse pruebas en sentido estricto a todos los artefactos generados durante la construcción de un producto, pero, obviamente se aplican diferentes técnicas de prueba a cada tipo de producto software. Y según Pressman (2010) el software se prueba para descubrir errores que se cometieron de manera inadvertida conforme se diseñó y construyó, además Jústiz Nuñez et al. (2014) concluyen que cuando el equipo de pruebas está integrado en su mayoría por especialistas externos al equipo de desarrollo se logra una mayor objetividad en la comprobación de los productos ya que se disminuye la posibilidad de omitir la prueba de alguna funcionalidad.

## Cronograma

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ACTIVIDAD | 2016 | | | | | | | | | | | | | | | | | | | | | | | |  |
| Mes 1 | | | | Mes 2 | | | | Mes 3 | | | | Mes 4 | | | | Mes 5 | | | | Mes 6 | | | | Ocupación en Horas |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Entrevistas con miembros de Ethereal GF | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 15,5 |
| Identificación de características del *framework* | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 62 |
| Selección de metodología de desarrollo |  | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 20 |
| Elaboración de GDD (*Game Document Design*) |  |  | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 47,5 |
| Elaboración del documento de definición del equipo de trabajo |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 3 |
| Elaboración del documento de selección de herramientas técnicas |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 5 |
| Elaboración bocetos |  |  |  | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 20 |
| Elaboración diseños |  |  |  |  | **X** | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 85 |
| Elaboración de modelado |  |  |  |  |  |  | **X** | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 83 |
| Realización de animaciones |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 34 |
| Planificación de iteraciones |  |  |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 3,5 |
| Iteración 1 de desarrollo |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  | 52,5 |
| Iteración 2 de desarrollo |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  | 52,5 |
| Iteración 3 de desarrollo |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** | **X** |  |  |  |  |  |  |  |  |  |  | 52,5 |
| Iteración 4 de desarrollo |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  |  |  |  |  |  | 52,5 |
| Iteración 5 de desarrollo |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  |  |  |  |  | 52,5 |
| Iteración 6 de desarrollo |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** | **X** |  |  |  |  |  |  | 52,5 |
| Elaboración de documento de cierre de desarrollo |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  | 10 |
| Planificación de pruebas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  | 4 |
| Iteración 1 de pruebas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  |  | 46 |
| Iteración 2 de pruebas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  |  | 46 |
| Iteración 3 de pruebas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |  | 46 |
| Cierre fase de pruebas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** |  |  |  | 6 |
| Elaboración informe de validación del *framework* |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** | **X** |  | 50 |
| Informe final del trabajo de grado | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | X | 40 |

# DESARROLLO

El presente documento es el informe final de todo el proceso de validación del framework “Unity ETH UQ” mediante la realización de un videojuego llamado *Risky jungle,* se divide en cuatro fases que son detalladas a continuación: En la fase 1, fase de análisis, se identificaron los requerimientos del framework y las características de los módulos a evaluar, para lo cual se realizaron una serie de entrevistas con algunos miembros de Ethereal GF y se definieron las funcionalidades a ser implementadas en el proyecto para cumplir el objetivo de validación. En la fase 2, diseño y construcción, fueron definidos y documentados aspectos claves del juego como el gameplay, historia, género, entre otros, posteriormente se realizaron los diseños necesarios para finalmente realizar la construcción del videojuego incluyendo el framework en su código fuente mediante una metodología de desarrollo que fue seleccionada teniendo en cuenta las necesidades del equipo de trabajo. En la fase 3, pruebas, se sometió el aplicativo a pruebas mediante un proceso iterativo, iniciando como testers los desarrolladores, posteriormente miembros del grupo de investigación en redes información y distribución – GRID de la universidad del Quindío y finalmente un grupo de usuarios totalmente ajenos al proyecto. Finalmente la fase 4, validación, se realizó paralelamente a las fase 2 y 3, la cual consistió en implementar las funcionalidades del framework identificadas en la fase 1 para probar su funcionamiento en el aplicativo, esta fase culminó con la realización de un informe detallando los hallazgos.

## Fase 1: Análisis de las características, requerimientos y funcionalidades del framework

Esta fase es el hincapié para las demás debido a que en esta es cuando se conoció el funcionamiento del framework “Unity ETH UQ” y se definieron explícitamente las funcionalidades a ser probadas, además permitió la primera interacción del equipo de desarrollo con el código fuente, lo cual posibilitó estructurar el videojuego de forma que se pudieran abarcar dichas funcionalidades en el desarrollo.

Para el desarrollo de esta fase se realizaron entrevistas a algunos de los miembros de Ethereal GF con el fin de conocer el objetivo, requisitos y características del framework, además se analizó el código de cada una de las clases de los módulos *util, display, audio, analytics, components* y *appsystem* para analizar como interactuaban entre ellas y definir sus funcionalidades. Lo cual quedó plasmado en un documento, que se encuentra adjunto con el presente informe ([Ver Identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ”](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Identificación%20de%20características,%20requerimientos%20y%20funcionalidades%20del%20framework)), del cual a continuación se detalla su estructura.

### Entrevistas

La sección de entrevistas recopila los aspectos más relevantes de las reuniones con algunos de los miembros de Ethereal GF y lo que cada uno resalta como los aspectos más importantes, los cuales son presentados a continuación.

#### Einer Zapata

El desarrollador de Ethereal GF resalta que el framework tiene como principales aspectos:

* Agilizar el desarrollo de funcionalidades y componentes complicados.
* Garantizar la utilización de código de calidad.
* Permitir el uso de analíticas de uso y marketing.
* Sobrecargar métodos propios de Unity 3D para optimizar su función.

#### Andrés Herrera

El desarrollador de Ethereal GF resalta que el framework tiene como principales aspectos:

* Agilizar el proceso de aprendizaje de Unity 3D.
* Mejoramiento del tiempo en el cual se cumplen las tareas.
* Las funciones del framework facilitan el desarrollo.

Posteriormente a la entrevista Andrés Herrera informó al equipo de trabajo que Ethereal GF desarrolló la segunda versión del módulo de analíticas, con lo cual el componente *analytics* quedaba obsoleto y le daba lugar al nuevo componente *appsystem*.

#### Andrés Carvajal

El administrador de proyectos de Ethereal GF resalta que el framework tiene como principales aspectos:

* Realizar funcionalidades de manera más rápida.
* Fácil integración de componentes que pueden enriquecer el juego.
* Fácil solución de errores.
* Buena curva de aprendizaje.

Posteriormente a la entrevista, Andrés Carvajal informó al equipo de trabajo que debido a que Unity 3D a partir de su versión 5.0 incluyo soporte a elemento UI Ethereal GF reemplazo el módulo *display* por el módulo *components*.

### Análisis de los módulos

En esta sección se analizó el código de cada una de las clases que componen los módulos con el objetivo de entender cómo interactúan entre ellas y poder identificar fácilmente las funcionalidades de cada módulo. Dicho análisis se llevó a cabo utilizando una tabla para cada clase la cual contiene las columnas método, funcionalidad, métodos llamados y retorno, donde se plasmaban el nombre del método, lo que este hace, que otros métodos utiliza y su tipo de retorno en caso de tenerlo, respectivamente. Para conocer a detalle el contenido mencionado se debe observar la sección “Framework” del documento Identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ” ([Ver Identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ”](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Identificación%20de%20características,%20requerimientos%20y%20funcionalidades%20del%20framework)),.

### Identificación de las funcionalidades

En esta sección se indican explícitamente cada una de las funcionalidades del framework, discriminadas por cada módulo. Para su realización se tomó como base el análisis de la sección anterior y se definieron las funcionalidades de la Tabla 1.

|  |  |
| --- | --- |
| **Módulo** | **Funcionalidad** |
| Audio | Reproducir un audio |
| Reproducir un efecto |
| Reproducir un efecto de forma cíclica |
| Modificar el volumen de la música |
| Modificar el volumen de los efectos |
| Util | Verificar la existencia de un archivo |
| Manejo de Timer |
| Definir si la plataforma en que se ejecuta la aplicación es móvil |
| Modificar la visibilidad de un game object |
| Obtener un game object por su nombre |
| Obtener un game object hijo a partir de su nombre y su padre |
| Components | Soporte de idiomas a imagen |
| Soporte de idiomas a texto |
| Reproducción de audio a un toggle |
| Soporte de idiomas a botón |
| Reproducción de audio a un botón |
| Modificar el valor de progreso a una barra de progreso |
| Animar la barra de progreso cuando se modifica su valor |
| Cargar una escena |
| Al cargar una escena crear un panel sobre el canvas y mostrar un cargador |
| Obtener el lenguaje del sistema |
| AppSystem | Inicializar el sistema de analíticas |
| Registrar un evento en el sistema de analíticas |
| Registrar una pantalla en el sistema de analíticas |
| Obtener variables registradas en el sistema de analíticas |

Tabla 1. Funcionalidades del framework

### Requerimientos

Esta sección reúne los requerimientos técnicos necesarios para el correcto funcionamiento del framework, los cuales fueron identificados mediante el análisis del código fuente y la información brindada por el administrador de proyectos de Ethereal GF.

Ya que la versión del framework que Ethereal GF brindó fue construida sobre la versión 5.1.1f1 de Unity 3D, para evitar posibles problemas de compatibilidad de algunas funciones al importar el framework, se recomendó que el desarrollo del videojuego fuera sobre la misma versión. Los demás aspectos técnicos para la realización del videojuego dependieron de la compatibilidad de Unity 3D y la disposición del equipo de trabajo.

## Fase 2: Diseño y construcción

En esta fase se diseñó y construyó el videojuego utilizando el framework “Unity ETH UQ”, para lo cual antes de iniciar se debió seleccionar una metodología de desarrollo teniendo en cuenta ciertos criterios definidos por el equipo de trabajo, dicha selección se encuentra en el documento *Selección de la metodología de desarrollo* ([Ver Selección de la metodología de desarrollo](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Selección%20de%20la%20metodología%20de%20desarrollo.docx))que está adjunto con este documento. Posteriormente siguiendo el marco de trabajo que propone la metodología SUM, que fue la seleccionada, se procedió a ejecutar las tres primeras fases, las cuales son:

* **Concepto:** en la cual se definen los aspectos técnicos y elementos de juego.
* **Planificación:** en donde se define la manera en que trabarán los roles y personas involucradas en el proyecto, además de planificar las principales características que se implementan en esta fase.
* **Elaboración:** donde mediante un proceso iterativo e incremental se construyó el videojuego, lo cual implicó la realización de una planeación y seguimiento de cada iteración, las cuales fueron seis.

### Selección de la metodología de desarrollo

Este artefacto tiene como función formalizar el proceso de selección de la metodología de desarrollo que se usó para el diseño, construcción y validación del videojuego, dicho proceso se lleva a cabo en seis pasos, los cuales son:

#### Establecer las guías para el análisis de decisiones

Se analiza el contexto sobre el cual se va a hacer la selección, el cual es el proceso de un proyecto software, la importancia de las metodologías de desarrollo y se exponen las dos grandes ramas que son las metodologías robustas y las agiles junto con sus principales características.

#### Establecer los criterios de evaluación

Teniendo en cuenta las necesidades del equipo de trabajo en esta sección se exponen explícitamente los criterios sobre los cuales se van a evaluar las posibles soluciones. Dichos criterios fueron:

* **Tamaño del equipo de trabajo:** Dado que el proyecto es un trabajo de grado, está limitado a un grupo reducido de integrantes, específicamente dos, por lo cual dos de las condiciones ideales para la metodología son que no contenga demasiados roles y que el proceso de documentación no sea muy extenso.
* **Involucrar al cliente en el proceso:** Debido a que un trabajo de grado está bajo constante revisión, donde el director y principal revisor adquiere el rol de cliente, la metodología debe garantizar su constante interacción en el proceso.
* **Magnitud del proyecto:** El proyecto tiene como fin poner a prueba ciertos componentes del framework desarrollado por ethereal game factory en un entorno de desarrollo externo, es por esto que su dimensión es reducida.
* **Entregas:** Las constantes revisiones del proyecto abren la necesidad de mostrar avances sólidos cíclicamente y la mejor manera de hacerlo es mostrarlos sobre el aplicativo funcional, permitiendo de esta manera un crecimiento y evaluación secuencial.

#### Métricas de los criterios

En este paso del proceso de selección de la metodología de desarrollo se definieron las métricas de cada uno de los criterios seleccionados, para lo cual se decidió que la suma de los cuatro sería diez y el equipo de trabajo estableció los valores que se muestran en la Tabla2.

|  |  |
| --- | --- |
| **Criterio** | **Peso** |
| Tamaño del equipo de trabajo | 3 |
| Involucrar al cliente en el proceso | 2 |
| Magnitud del proyecto | 3 |
| Entregas | 2 |

Tabla 2. Peso de los criterios de selección de la metodología de desarrollo

Además se definió que cada uno de estos criterios podría tomar tres posibles valores, los cuales se muestran en la tabla 3.

|  |  |
| --- | --- |
| **Valor** | **Descripción** |
| Cero | Para indicar que la solución no cumple con las necesidades del equipo en ese criterio. |
| La mitad del peso del criterio | Para indicar que la solución no es óptima para el criterio pero puede adaptarse. |
| La totalidad del peso del criterio | Para indicar que la solución es óptima en dicho criterio. |

Tabla 3. Posibles valores de cada criterio de selección

#### Identificar las soluciones alternativas

En este paso del proceso de selección se plasman las alternativas más viables después de un proceso de documentación que fue llevado a cabo por el equipo de trabajo, donde se encontró que en la actualidad las metodologías para desarrollo de software son muy diversas y hay una cantidad considerable de estas, a la hora de ser implantadas para el desarrollo de videojuegos estas deben ser sometidas a ciertos cambios y es allí donde la diversidad escasea. Además según Morales Urrutia et al. (2010) son escasos los procesos específicos para el desarrollo de videojuegos que sean públicos y los que existen, en su mayoría son procesos cerrados en sentido de que su información, modelos, plantillas y herramientas no están disponibles al público en general. A continuación se exponen las tres opciones tomadas.

**Metodología cascada**

Esta metodología se caracteriza por tener unas fases muy claras, las cuales son totalmente secuenciales de tal manera que el inicio de una etapa está ligada a la culminación de la anterior, pero antes de dar por culminada una etapa se debe realizar una evaluación para determinar su aceptación o corrección. Un aspecto relevante de esta metodología es que se enfoca bastante en la documentación para dejar claro desde un inicio todos los aspectos del proyecto sin dejar opción a cambios durante las demás etapas.

Para el desarrollo de videojuegos esta metodología ha sido levemente modificada ya que a diferencia de los productos software tradicionales los videojuegos incluyen componentes gráficos y sonoros, los cuales son de vital importancia. Las etapas de esta metodología para desarrollo de videojuegos según Chaparro García (2015) son:

* **Especificación del juego:** Se documenta la especificación del videojuego desde la perspectiva del usuario.
* **Biblia del arte:** Los productores y directores artísticos especifican las herramientas, conceptos, historia, guion y diseño.
* **Especificación técnica:** Se describen las herramientas para realizar los diagramas, interacciones y código.
* **Construcción:** Se lleva a cabo la elaboración del videojuego.
* **Aseguramiento de calidad:** Se lleva a cabo la verificación de lo realizado con lo estipulado en los documentos de planeación y diseño.
* **Pruebas:** Se realiza el testing de las primeras versiones del videojuego para una posterior corrección de errores. Generalmente esta etapa se subdivide en pruebas internas con el equipo de desarrollo, posteriormente pruebas alfa y beta.
* **Liberación:** Esta etapa es cuando se libera la versión estable del videojuego al público.

**Metodología SUM**

Esta metodología cuenta con dos publicaciones muy importantes que la describen en su totalidad, las cuales son la publicación “*Una metodología para desarrollo de videojuegos”* elaborada por Nicolás Acerenza, Ariel Coppes, Gustavo Mesa, Alejandro Viera Eduardo Fernández, Tomás Laurenzo y Diego Vallespir de la facultad de ingeniería de la universidad de la república, Uruguay y la especificación de la metodología utilizando Eclipse EPF publicada por la empresa de videojuegos Gemserk.

Esta metodología para videojuegos tiene como objetivo primordial el desarrollar videojuegos de calidad en tiempo y costo garantizando la mejora continua para mejorar su eficacia y eficiencia. Además de administrar eficientemente tanto los recursos como los riesgos optimizando así la productividad del equipo de trabajo.

SUM es una metodología ágil que está basada en SCRUM permitiendo flexibilidad en la elaboración del proyecto, ideal para equipos de desarrollo pequeños conformados de dos a siete integrantes, proyectos cortos de duración inferior a un año, equipos multidisciplinarios y además permite una alta interacción con el cliente. Las etapas de esta metodología son:

* **Concepto:** Se definen aspectos de negocio como los objetivos del proyecto y a qué audiencia se apunta, elementos sobre el producto a desarrollar como la historia y los personajes y técnicos como el entorno de desarrollo a utilizar para la elaboración de dicho producto.
* **Planificación:** En esta fase se planifican el resto de fases del proyecto y se especifican las características a implementar del videojuego.
* **Elaboración:** Se realiza un trabajo de forma iterativa e incremental para presentar versiones funcionales hasta culminar en una versión estable.
* **Beta:** Su objetivo es evaluar y ajustar de forma iterativa el videojuego eliminando la mayor cantidad de errores detectados.
* **Cierre:** Esta es la etapa final, en la cual se pone el videojuego a disposición del público.
* **Gestión de riesgos:** Esta etapa mantiene vigente durante todo el ciclo de vida del proyecto y su función es administrar los riesgos y su solución.

**Metodología HUDDLE**

Esta metodología fue propuesta por Gerardo Abraham Morales Urrutia, Claudia Esther Nava López, Luis Felipe Fernández Martínez, y Mirsha Aarón Rey en su publicación Procesos de desarrollo para videojuegos en el 2010.

Se llama Huddle a la reunión que se realiza en el juego antes de cada jugada en el futbol americano; la filosofía es que mediante breves reuniones de planeación a corto plazo, se planee cada “jugada” que se inicie; con el fin de dar un seguimiento más estrecho al avance del proyecto y poder hacer correcciones tempranas. Es un proceso específico para desarrollo de videojuegos óptimo para equipos multidisciplinarios de 5 a 10 personas, ágil, iterativo, incremental y evolutivo, sin embargo puede usarse en equipos de menos de 5 elementos. Las etapas de esta metodología son:

* **Preproducción:** Tiene como objetivo migrar la idea del diseñador al Feature Log y al Sprint Plan; estos documentos dan la pauta a la planeación y producción del videojuego.
* **Producción:** Esta etapa se apoya en las herramientas de Scrum como son los Daily Meetings, los Sprints y Sprint Reviews, y en artefactos como el Sprint Backlog y Burn-down Charts y como su nombre lo indica, en esta etapa se realiza la producción del videojuego.
* **Postmortem:** Para realizar esta fase, el equipo debe realizar la última actividad llamada End-game Huddle en la cual se analizarán los aspectos positivos y negativos del proyecto. Del End-game Huddle saldrán sugerencias que deberán ser analizadas y filtradas con la intención de generar un reporte que incluya todas aquellas propuestas de mejora al proceso con el objetivo de que sean incorporadas en el próximo proyecto.

#### Evaluar las soluciones alternativas

El objetivo de este paso en el proceso de selección de la metodología de desarrollo es someter las alternativas seleccionadas a una evaluación bajo los criterios previamente definidos. Para hacer más comprensible y funcional este proceso se realizó la Tabla 4, la cual establece una comparación entre las alternativas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Tamaño del equipo de trabajo** | **Involucrar al cliente** | **Magnitud del proyecto** | **Entregas** |
| **Cascada** | Es flexible al tamaño del equipo, pero su alto nivel de documentación consume una cantidad de tiempo considerable. | El cliente es involucrado en las etapas iniciales y su interacción se ve reducida hasta que se tiene un producto funcional. | Es flexible al tamaño del proyecto pero debido a su alto nivel de documentación se hace ideal para proyectos de gran escala. | Tiene etapas y entregables estructurados, pero es difícil percibir propuestas de valor hasta una entrega funcional y estable. |
| **SUM** | Ideal para equipo de dos a siete personas. | Es una metodología ágil, por lo cual el cliente es un rol vital para las propuestas de valor del proyecto. | Es una metodología ideal para proyectos cortos, de duración menor a un año, por ende su magnitud no es muy grande. | Al ser una metodología ágil, esta plantea sprints en los cuales se le agrega propuesta de valor a un producto funcional. |
| **Huddle** | Óptimo para equipos de 5 a 10 personas, sin embargo, puede utilizarse en equipos de menos de 5 elementos. | Conserva la interacción con el cliente al igual que en SCRUM ya que está basado en este. | Es flexible al tamaño del proyecto, pero se adapta mejor a proyectos no muy largos. | Sus bases agiles hacen que sea una metodología iterativa incremental y en cada huddle se agrega una propuesta de valor. |

Tabla 4. Comparación entre las posibles soluciones

Basándose en la Tabla 4, el equipo de trabajo asignó las calificaciones a cada metodología, las cuales se muestran en la Tabla 5.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Tamaño del equipo de trabajo** | **Involucrar al cliente** | **Magnitud del proyecto** | **Entregas** | **Total** |
| **Cascada** | 1,5 | 0 | 0 | 0 | 1,5 |
| **SUM** | 3 | 2 | 3 | 2 | 10 |
| **Huddle** | 1,5 | 2 | 3 | 2 | 8,5 |

*Tabla 5. Evaluación de las posibles soluciones*

#### Seleccionar la solución

Finalmente se plasmó la decisión final teniendo en cuenta las calificaciones y la adaptación a los criterios más relevantes para el proyecto. Se llegó a la conclusión de que SUM y Huddle son las metodologías más convenientes, pero SUM presenta características mejor adaptadas a grupos y proyectos pequeños.

### Concepto

El primer paso en el proceso que establece la metodología SUM es definir los aspectos de negocio, técnicos y elementos de juego sobre el producto a desarrollar. Los aspectos de negocio a decidir involucran la audiencia la que se apunta y los posibles modelos de negocio. Los elementos del juego a determinar son las principales características, la historia, los personajes, la ambientación, el gameplay, entre otros. Las decisiones técnicas involucran la elección de las herramientas y las tecnologías a utilizar. Debido a la naturaleza del proyecto, el cual es un proyecto académico de investigación, se omitieron los aspectos de negocio.

Para definir los aspectos anteriormente nombrados el equipo de trabajo realizó dos documentos, los cuales se detallan a continuación.

#### Definir los aspectos de juego

Para definir los aspectos de juego el equipo de trabajo realizó un Game Document Design – GDD, en el cual se plasmaron las ideas que sentaron las bases para el producto final y se encuentra adjunto con el presente documento ([Ver GDD Rysky Jungle](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\GDD%20Risky%20jungle.docx)). A continuación se encuentra la estructura de dicho documento.

**Resumen**

En esta sección se plasmó el preámbulo a la historia del videojuego, se describió a grandes rasgos el tipo de juego y se complementó con un mensaje emotivo que incita a jugarlo.

**Plataforma objetivo**

Se ratificó nuevamente que el videojuego solo será para el sistema operativo Android debido a que el framework está enfocado a los sistemas operativos IOS y Android, pero el equipo de trabajo no cuenta con las herramientas necesarias para soportar IOS.

**Estilo visual**

Se definió que el estilo visual del juego presentaría una modalidad grafica oscura, en escala de grises, simple y minimalista con el objetivo de brindar un ambiente misterioso e inquietante.

**Estilo de audio**

Teniendo en cuenta el entorno y la temática definidos, se deicidio que la música fuera sombría y minimalista para generar suspenso en el jugador, pero sin llegar al punto de ser opresiva. Además se definió que los personajes no tendrían voces y los efectos serían acordes a la situación y personajes involucrados.

**Comienzo de juego e introducción**

En esta sección se definió como sería la interacción inicial del usuario con el juego, donde encontrará un menú principal y apenas inicie una partida se mostrarán las cinemáticas y podrá jugar un nivel tutorial.

**HUD y menús en juego**

En esta sección se definieron las opciones e información que el usuario tendría disponible durante una partida sin salir al menú principal (Ilustración 1). La pantalla en la esquina inferior derecha cuenta con dos botones, uno para que el personaje principal salte y otro para que ataque, en la parte superior izquierda dos botones con los que se escogerá el arma a usar, uno para la lanza y otro para la cerbatana y en la parte superior derecha un botón de pausa, la distribución de dichos componentes se muestra en la Ilustración 1, el botón de pausa habilitaría un menú con las opciones de reanudar la partida, reiniciarla o ir al menú principal, como se muestra en la Ilustración 2.

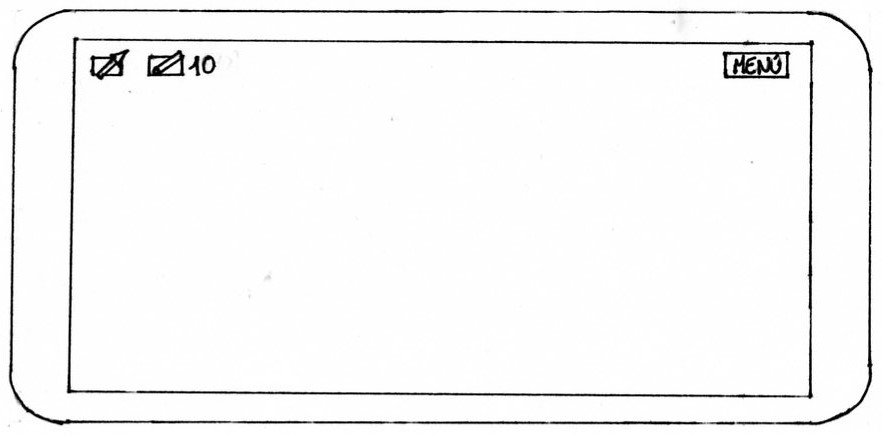


Ilustración 1. Bosquejo del HUD.

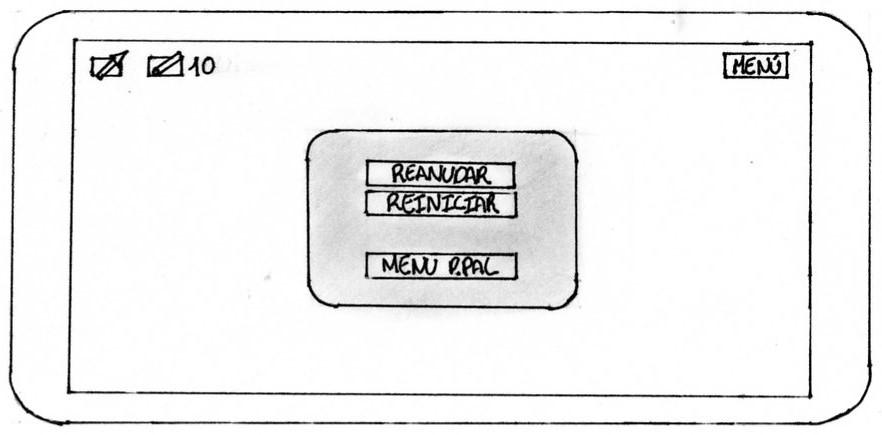


Ilustración 2. Bosquejo del menú en juego.

**Menú principal**

El siguiente paso fue definir el menú principal, para el cual el equipo de trabajo decidió que como seria para un dispositivo móvil, este debería ser sencillo e intuitivo y brindaría las opciones de iniciar una nueva partida, continuar una partida iniciada, configurar el sonido e idioma y mostrar la información del juego, la distribución de dichas opciones en la pantalla se muestra en la Ilustración 3. Cuando el jugador no tiene una partida inicial el menú no presentará la opción de continuar, como se muestra en la Ilustración 4.



Ilustración 3. Bosquejo del menú principal cuando existe una partida iniciada.



Ilustración 4.Bosquejo del menú principal cuando no existe una partida iniciada.

**Ventana de configuración**

La ventana de configuración consta de modificar el volumen de la música, de los efectos y cambiar entre idioma español e inglés, como se muestra en la Ilustración 5, en dicho menú el framework brindó funcionalidades útiles.

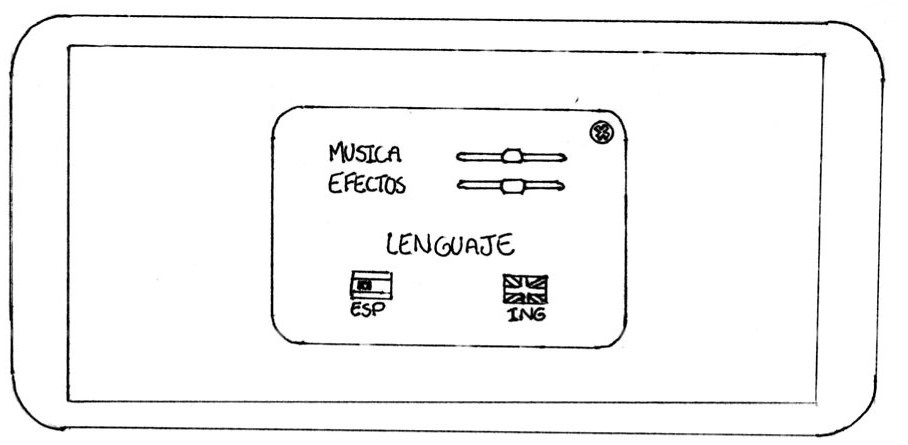


Ilustración 5. Bosquejo de la pantalla de configuración.

**Ventana de game over**

Cuando el personaje principal muere, ya sea atacado por uno de los rivales o por caer en un obstáculo la partida termina y aparece una ventana indicando el game over, cuyo boceto se muestra en la Ilustración 6, y brinda las opciones de reintentar o ir al menú principal.

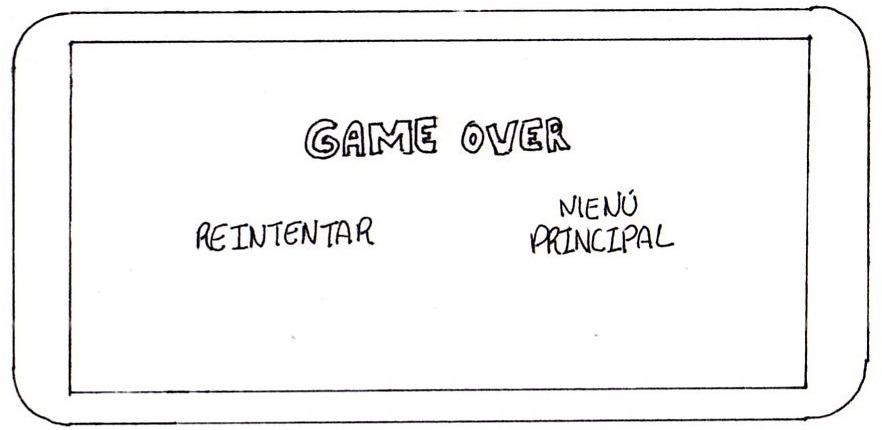


Ilustración 6.Bosquejo de la pantalla de game over

**Selección de niveles**

Los niveles son secuenciales, por lo cual el usuario solo puede escoger iniciar una nueva partida desde el nivel tutorial o continuar en el nivel que terminó su última partida.

**Gameplay**

El modo de juego está determinado por una campaña de un solo jugador, el cual estará enmarcado en una vista lateral (Side scroller). En el transcurso de los niveles el jugador encontrará cuatro tipos de rivales (Serpientes, cocodrilos, osos y tigres), los cuales deberá abatir de diferentes formas para poder continuar su aventura y estar cada vez más cerca de resolver el enigma (Survival).

**Mecánicas y controles**

Para el desplazamiento por el escenario, la selección de arma y sobrevivir, el equipo de trabajo definió los siguientes controles:

* Si el jugador presiona, sostiene y desliza el dedo hacia la derecha o izquierda, el personaje se desplazará hacia esa dirección
* Si el jugador presiona, sostiene y desliza el dedo hacia abajo, el personaje se agachará.
* En la parte superior izquierda habrán dos botones, el primero de izquierda a derecha es para equipar la lanza como arma para luchar y el segundo es para equipar la cerbatana.
* En la parte inferior derecha habrán dos botones, el primero de izquierda a derecha es para atacar con el arma que tenga seleccionada, y el segundo para saltar.

**Criterios de éxito**

Teniendo en cuenta que el juego trata de supervivencia, el equipo de trabajo definió que no habrá check points y la única manera de que el jugador tiene para ganar en un nivel es superar todos los desafíos y haber vencido todos los animales y peligros que se encuentran en el recorrido.

**Personajes**

El equipo de trabajó definió los personajes, los cuales son: el indígena, que es el personaje principal y su boceto se muestra en la Ilustración 7 y los rivales que son oso, cocodrilo, serpiente y tigre, cuyos bocetos se muestran en las Ilustraciones 8, 9, 10 y 11 respectivamente.

* **Personaje principal**

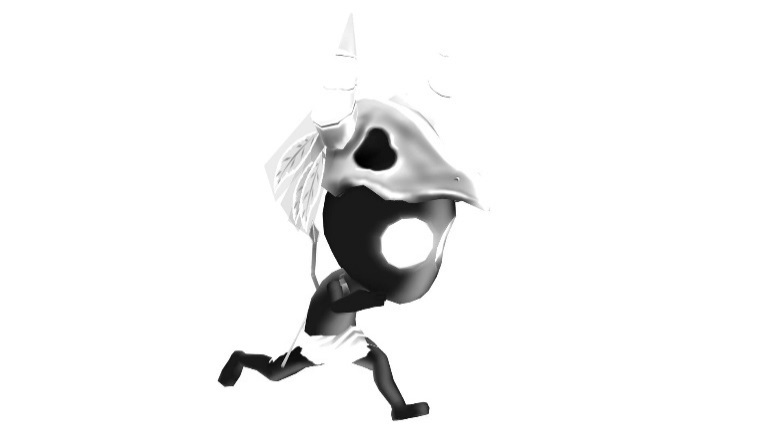


Ilustración 7. Boceto del personaje principal.

* **Oso**



Ilustración 8.Boceto del oso.

* **Cocodrilo**

**

Ilustración 9. Boceto del cocodrilo.

* **Serpiente**



Ilustración 10.Boceto de la serpiente.

* **Tigre**

**

Ilustración 11. Boceto del tigre.

* **Animaciones del personaje principal**
  + Idle
  + Ataque con cerbatana
  + Ataque con lanza
  + Animación de agachado
  + Animación de deslizamiento
  + Animación disparo con cerbatana agachado
  + Animación de salto
  + Animación de muerte
* Animaciones de animales
  + Animación Idle
  + Animación de muerte
  + Animación de ataque

**Armas**

Se definieron las armas para derrotar los rivales estratégicamente distribuidos, las cuales fueron la lanza, cuyo boceto se muestra en la Ilustración 12, y la cerbatana, cuyo boceto se muestra en la Ilustración 13.



Ilustración 12. Boceto de la cerbatana.



Ilustración 13.Boceto de la lanza.

**Audio**

En esta sección se definió la música del juego y los diversos efectos.

**Cinemáticas**

Se definieron las cinemáticas para el videojuego, las cuales son dos. La primera previa al primer nivel, explicando cómo inicia la aventura y la segunda al final del último nivel resolviendo el misterio y generando intriga en el jugador.

#### Definir los aspectos técnicos

Para definir los aspectos técnicos del proyecto el equipo de trabajo creó un documento donde se muestra una descripción general de cada una de las herramientas y su versión específica a usar en el proyecto, dicho artefacto se encuentra adjunto con el presente informe ([Ver Documento de herramientas técnicas](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Definición%20de%20herramientas%20técnicas.docx)). A continuación se detallan brevemente dichas herramientas.

**Unity 3D**

Esta herramienta fue usada en su versión 5.1.1f1 debido a que el framework “Unity ETH UQ” fue desarrollado sobre esta versión y usar una diferente podría implicar que algunas funcionalidades no sean compatibles y se usó como motor para la construcción del videojuego. La descripción que se plasmó en el documento fue basada en un análisis que desarrolló el equipo de investigación en el grupo de excelencia de Redes de Neuronas Artificiales y Sistemas Adaptativos (RNASA) de la Universidad de Coruña.

**Microsoft Word**

Se seleccionó esta herramienta en su versión 2013 debido a su popularidad y que su licencia está incluida en los computadores del equipo de desarrollo y se usó para elaborar los documentos e informes necesarios para el proyecto. La descripción brindada se basa en un informe creado por Ruiz Severiche (2012).

**Sublime text**

Esta herramienta se utilizó para el análisis del código del framework para poder identificar sus funcionalidades ya que su interfaz de color oscuro y la riqueza de coloreado de la sintaxis centra la atención en la interacción de métodos, variables y clases. Se seleccionó la versión 2.0.2 debido a que era la última versión estable disponible en la página oficial al momento de realizar el documento de herramientas técnicas, ya que la versión 3 aún se encontraba en fase beta.

**Blender**

Esta herramienta se utilizó para el modelado, diseño y animación de los personajes y texturas del videojuego ya que el equipo de desarrollo cuenta con experiencia en su uso y su licencia es gratuita. La versión que se utilizó fue la 2.77 ya que era la última versión estable disponible en la página oficial al momento de realizar el documento de selección de herramientas técnicas.

**Github**

Esta herramienta se usó como controlador de versiones del proyecto y plataforma de desarrollo colaborativo debido a que el equipo de trabajo cuenta con experiencia en su uso. La versión que se usó fue la 3.2.0.0 la cual era la última versión disponible en la página oficial cuando se elaboró el documento de herramientas técnicas.

### Planificación

El segundo paso en el proceso que establece la metodología SUM tiene dos objetivos principales, planificar el desarrollo y especificar las características a implementar en el videojuego. Dichos objetivos se logran realizando una planificación administrativa y especificando las características del videojuego respectivamente, los cuales se detallan a continuación.

#### Planificación administrativa

Para administrar el proyecto es necesario definir roles y responsabilidades, es por esto que el equipo de trabajo elaboró un documento llamado *Definición de roles y responsabilidades* el cual se encuentra adjunto con el presente informe ([Ver Definición de roles y responsabilidades](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Definición%20de%20roles%20y%20responsabilidades.docx)). Teniendo en cuenta el número de integrantes el equipo de trabajo y sus conocimientos se definieron los roles de cliente, diseñador de juego, programador y artista gráfico. A continuación se detalla la estructura del documento.

**Introducción**

En la introducción se plasmó el objetivo del documento el cual es la división y asignación de roles dentro del equipo de trabajo de la manera especificada en la metodología SUM y publicada en <http://www.gemserk.com/sum/> por la empresa de desarrollo de videojuegos uruguaya Gemserk.

**Roles**

En esta sección del documento de definición de herramientas técnicas se describieron las definiciones y responsabilidades de cada rol definidas en la metodología de desarrollo SUM.

**Cliente**

Este rol se encarga de especificar y mantener la visión del proyecto y se encarga de:

* Definir y valida el concepto del juego, aprueba los planes de proyecto y determina los hitos.
* Priorizar las características y tareas que dan más valor al videojuego en cada momento.
  + Evaluar el cumplimiento de las tareas y el producto obtenido al finalizar cada iteración, y participa de la evaluación del proyecto.
* Priorizar los errores a corregir buscando la mejor calidad posible de acuerdo a sus intereses.
* Valida las versiones del producto.

Se decidió que el cliente, dadas sus actividades, debería ser el director del trabajo de grado.

**Diseñador de juego**

Este rol se encarga de diseñar el gameplay, historia, ambientación, personajes, niveles y todos los elementos que hacen a la experiencia del jugador. Un diseñador de juego debe:

* Ser creativo y original.
* Entender bien el mercado y el público de los videojuegos.
* Tener habilidades para resolver problemas.
* Tener habilidades para narrar historias.
* Ser buen comunicador.
  + Entender las capacidades y beneficios de las distintas plataformas, tecnologías y técnicas de software.
* Tener habilidades básicas de dibujo y diseño 3d.
* Adaptarse rápido al cambio.
* Trabajar bien en equipo.
* Trabajar bien bajo presión.
* Tomar de buena manera las críticas.
* Estar actualizado con los desarrollos y las tendencias del mercado de videojuegos.

**Programador**

El programador tiene como principal responsabilidad implementar el software que compone al juego. Además deberá realizar el diseño de software necesario para poder realizar el desarrollo y posteriormente verificarlo. Este rol se destaca durante el proyecto por:

* Tener habilidades para programar.
* Debe saber cómo funciona el sistema o aplicación en una prueba.
* Debe estar familiarizado con la tarea de Verificación.
* Disfrutar de los videojuegos.
* Tener habilidades para la solución de problemas.
* Buena comunicación.
* Trabajar bajo presión.

**Artista gráfico**

Las habilidades necesarias para un artista varían según los requerimientos del juego en particular. De cualquier forma requieren conocimientos sobre las últimas herramientas gráficas, creatividad, talento y técnica. Debido a que en el equipo de trabajo ningún miembro cuenta con los conocimientos y talento necesario se decidió incorporar un nuevo miembro que si contará con las condiciones ideales para esta labor. Este rol se destaca durante el proyecto por:

* Poseer talento creativo, originalidad y fuerte sentido visual
* Tener sólidos conocimientos de informática

**Asignación de roles**

Teniendo en cuenta los roles anteriormente descritos y sus responsabilidades se asignaron los roles de la manera que se muestra en la Tabla 6.

|  |  |
| --- | --- |
| **Nombre** | **Rol** |
| Christian Andrés Candela Uribe | Cliente |
| Dival Mauricio Hoyos Castro | Diseñador de Juego  Programador |
| Julian David Serna Echeverri | Diseñador de Juego  Programador |
| Luis Marulanda Ramírez | Artista Gráfico |

Tabla 6. Asignación de roles al equipo de trabajo

#### Especificación del videojuego

Para especificar, priorizar y asignar las características que componen el videojuego, el equipo de trabajo tomó la plantilla Excel brindada por la especificación de la metodología elaborada por Gemserk. Las características identificadas se muestran en la Tabla 7, en donde se encuentra que la actividad con mayor prioridad es el diseño del personaje principal debido a que este es el punto de partida para realizar la estructuración del videojuego.

La priorización se llevó a cabo mediante una reunión del equipo de trabajo en donde cada integrante proponía un valor de 1 a 10 para cada actividad y el valor definitivo se elegía por consenso. El hecho de que una actividad tuviera más prioridad que otra no necesariamente significaba que fuera a ser implementada primero, ya que el orden dependía en gran parte de los objetivos de cada iteración de desarrollo. El archivo de Excel completo se encuentra adjunto con el presente documento ([Ver Especificación de características](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Especificación%20de%20características.xlsx)).

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **Característica** | **Responsable** |
| 9 | Diseño personaje principal | Artista Gráfico |
| 5 | Diseño cocodrilo | Artista Gráfico |
| 5 | Diseño serpiente | Artista Gráfico |
| 4 | Diseño oso | Artista Gráfico |
| 4 | Diseño objetos | Artista Gráfico |
| 7 | Escenario tutorial | Diseñador |
| 6 | Configuración animaciones personaje principal | Programador |
| 5 | Configuración animaciones cocodrilo | Programador |
| 5 | Configuración animaciones serpiente | Programador |
| 5 | Configuración animaciones oso | Programador |
| 4 | Configuración animaciones tigre | Programador |
| 5 | Configuración animaciones objetos | Programador |
| 3 | Primer escenario | Diseñador |
| 2 | Segundo escenario | Diseñador |
| 2 | Tercer escenario | Diseñador |
| 6 | Físicas personaje principal | Programador |
| 3 | Físicas cocodrilo | Programador |
| 3 | Físicas oso | Programador |
| 3 | Físicas tigre | Programador |
| 3 | Físicas serpiente | Programador |
| 3 | Físicas objetos | Programador |
| 7 | Configuración cámara | Programador |
| 4 | Texturas de las ventanas | Artista Gráfico |
| 4 | Menú principal | Programador |
| 3 | Menú de configuración | Programador |
| 2 | UI de créditos | Programador |
| 2 | Cargador de escenas | Programador |
| 2 | Menú en juego | Programador |
| 7 | Importación del framework | Programador |
| 3 | Conexión de escenas | Programador |
| 5 | Interacción personaje-serpiente | Programador |
| 5 | Interacción personaje-cocodrilo | Programador |
| 5 | Interacción personaje-oso | Programador |
| 5 | Interacción personaje-tigre | Programador |
| 5 | Interacción personaje-objetos | Programador |
| 4 | Cambio de arma | Programador |
| 3 | Elaborar HUD | Programador |
| 2 | Configurar soporte de idiomas | Programador |
| 2 | Configurar el sonido | Programador |

Tabla 7. Especificación, priorización y asignación de características del videojuego

### Elaboración

El tercer paso en el proceso que establece la metodología SUM es implementar el videojuego, para lo cual se trabaja de forma iterativa incremental para lograr una versión ejecutable del juego al finalizar cada iteración (Gemserk, 2008). Según Gemserk, en la especificación de la metodología utilizando Eclipse EPF, con esta forma de trabajo se puede evaluar el avance del proyecto, lo cual permite realizar cambios a tiempo y tomar decisiones para cumplir con los plazos planificados. Además, la experiencia adquirida permite mejorar la forma de trabajo en cada iteración y aumentar la productividad.

El equipo de trabajo definió que cada iteración tendría una duración de una semana, y para su culminación, el equipo de trabajo debería realizar cuatro actividades, las cuales son:

* Planificar la iteración
* Desarrollar las características
* Realizar seguimiento a la iteración
* Cerrar la iteración

De estas cuatro actividades el equipo debe realizar primeramente la planificación, en la cual se debe:

* **Definir objetivos:** Los objetivos describen que se pretenden lograr al finalizar la iteración y sirven también como guía para la toma de decisiones.
* **Seleccionar características:** Basándose en los objetivos de la iteración, la prioridad de las características plasmadas en el documento de *especificación de características* adjunto con el presente informe ([Ver Especificación de características](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Especificación%20de%20características.xlsx)) y la duración de esta se seleccionan las que serán implementadas en la presente iteración.
* **Refinar características:** Las características del videojuego planificadas para la presente iteración se deben descomponer en tareas de menor complejidad con el objetivo de medir más eficientemente el progreso y llevar un control más riguroso de esta.

Posteriormente se procede a desarrollar y verificar las características planificadas en la actividad anterior a través de la ejecución de las tareas que la componen. Al ejecutar una tarea se puede identificar la necesidad de ejecutar nuevas tareas, en dicho caso, estas deben ser ingresadas como nuevas tareas de la iteración.

Paralelamente a la actividad de desarrollo se realiza el seguimiento de la iteración, cuyo objetivo es mantener la visión y el control de la iteración con base a los objetivos planteados. Para poder realizar este seguimiento cada miembro del equipo debe retroalimentar el estado actual de sus labores para determinar la existencia de problemas o desvíos y en caso de existir se registran y se identifican las posibles soluciones entre los involucrados.

Finalmente se realiza la actividad de cierre, en la cual se evalúa el estado del videojuego y lo ocurrido en la iteración teniendo en cuenta el cumplimiento de las tareas de cada característica y los objetivos. Se debe prestar atención a los problemas y dificultades que expresen los miembros del equipo de trabajo con el objetivo de evitar que se repitan.

Para las actividades de planeación, seguimiento y cierre, se creó una plantilla, la cual se llamó *Plantilla de planificación de iteración* y se encuentra adjunta con el presente informe ([Ver Plantilla de desarrollo de iteración](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Plantilla%20de%20desarrollo%20de%20iteración.docx)).

#### Iteración 1

En la primera iteración se realizaron las texturas del background del escenario y del terreno, así como el modelado y las animaciones del personaje principal y los tres rivales. El documento completo de la planificación, seguimiento y cierre de la iteración se llamó *Documento de desarrollo de iteración 1* y se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración 1](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%201.docx)).

**Objetivos**

Se definieron los siguientes objetivos:

* Elaborar las texturas del background del escenario.
* Elaborar las texturas del terreno.
* Diseñar y animar el personaje principal.
* Diseñar y animar los tres rivales.
* Montaje inicial del background en la herramienta de desarrollo.

**Selección de características**

De acuerdo a los objetivos y al documento de especificación de características, las seleccionadas para esta iteración fueron las plasmadas en la Tabla 8 donde se destaca el diseño de los personajes y el montaje del menú principal.

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **ID** | **Característica** |
| 9 | 1 | Diseño personaje principal |
| 5 | 2 | Diseño cocodrilo |
| 5 | 3 | Diseño serpiente |
| 4 | 4 | Diseño oso |
| 4 | 5 | Diseño objetos |
| 7 | 6 | Escenario tutorial |

Tabla 8. Características seleccionadas para la iteración 1

**Refinación de características**

Las características seleccionadas fueron descompuestas en tareas, de las cuales la mayoría fueron definidas en la planificación de la iteración, pero durante el desarrollo los programadores detectaron la necesidad de incluir nuevas tareas. La Tabla 9 contiene la totalidad de tareas realizadas durante la iteración.

|  |  |
| --- | --- |
| **Características** | **Tareas** |
| Diseño personaje principal | Creación del esqueleto |
| Creación de las texturas |
| Creación casco |
| Creación vestimenta |
| Creación textura del casco |
| Creación textura de la vestimenta |
| Creación cerbatana |
| Creación textura de la cerbatana |
| Creación lanza |
| Creación textura de la lanza |
| Realizar animación idle |
| Realizar animación agachado |
| Realizar animación ataque con cerbatana |
| Realizar animación ataque con lanza |
| Realizar animación deslizar |
| Realizar animación disparo agachado |
| Realizar animación salto |
| Realizar animación muerte |
| Diseño cocodrilo | Creación esqueleto |
| Creación de las texturas |
| Realizar animación idle |
| Realizar animación ataque |
| Realizar animación muerte |
| Diseño serpiente | Creación esqueleto |
| Creación de las texturas |
| Realizar animación idle |
| Realizar animación ataque |
| Realizar animación muerte |
| Diseño oso | Creación esqueleto |
| Creación de las texturas |
| Realizar animación idle |
| Realizar animación ataque |
| Realizar animación muerte |
| Realizar animación ataque secundario |
| Diseño objetos | Creación texturas del agua |
| Creación texturas de la balsa |
| Creación texturas de la cascada |
| Creación del diseño de la roca rodando |
| Creación textura de la roca rodando |
| Realizar animación de la roca rodando |
| Escenario tutorial | Organizar las texturas del background para el nivel tutorial |
| Crear los prefabs para el terreno |

Tabla 9. Descomposición de las características de la primera iteración en tareas

**Seguimiento de la iteración**

En el transcurso de la iteración se presentaron dos situaciones relevantes, las cuales fueron:

* Desconocimiento de la herramienta por parte de los desarrolladores, lo cual afecto el ritmo de trabajo.
* Inicialmente las texturas del background estaban muy oscuras y solucionarlo le llevo bastante tiempo a los desarrolladores.

**Cierre y evaluación**

A pesar que se presentaron situaciones adversas, en esta iteración se llevaron a cabo la totalidad de las tareas y se cumplieron los objetivos dentro de la semana prevista, por ende esta iteración se cerró con una ejecución del 100% de las tareas.

**Lecciones aprendidas**

Como principal lección aprendida se obtuvo que:

* Los programadores se familiarizaron mejor con la herramienta Unity 3D, lo cual les brindara mayor fluidez en el desarrollo.

**Mejoras al proceso**

Como mejora al proceso de creación del videojuego se obtuvo:

* Al surgir una tarea que los programadores no conozcan su proceso de implementación, se debe recurrir a varias fuentes para así conocer diversas formas de realizar dicha tarea y poder generar una solución adaptada a las necesidades del proyecto.

#### Iteración 2

En la segunda iteración se realizaron las configuraciones de las animaciones del personaje principal y los tres rivales. Con el objetivo de dejar los prefabs listos para su uso en diferentes partes del juego. El documento completo de la planificación, seguimiento y cierre de la iteración se llamó *Documento de desarrollo de iteración 2* y se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración 2](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%202.docx)).

**Objetivos**

Se definieron los siguientes objetivos:

* Elaborar el prefab del personaje principal con las animaciones y collider.
* Realizar el controlador de las animaciones del personaje principal.
* Elaborar el prefab del oso con las animaciones y collider.
* Realizar el controlador de las animaciones del oso.
* Elaborar el prefab del cocodrilo con las animaciones y collider.
* Realizar el controlador de las animaciones del cocodrilo.
* Elaborar el prefab de la serpiente con las animaciones y collider.
* Realizar el controlador de las animaciones de la serpiente.

**Selección de características**

De acuerdo a los objetivos y al documento de especificación de características, las seleccionadas para esta iteración fueron las plasmadas en la Tabla 10, donde se destaca la configuración de las animaciones de los personajes.

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **ID** | **Característica** |
| 6 | 7 | Configuración animaciones personaje principal |
| 5 | 8 | Configuración animaciones cocodrilo |
| 5 | 9 | Configuración animaciones serpiente |
| 5 | 10 | Configuración animaciones oso |
| 4 | 11 | Configuración animaciones tigre |

Tabla 10. Características seleccionadas para la iteración 2

**Refinación de características**

Las características seleccionadas fueron descompuestas en tareas, de las cuales la mayoría fueron definidas en la planificación de la iteración, pero durante el desarrollo los programadores detectaron la necesidad de incluir nuevas tareas. La Tabla 11 contiene la totalidad de tareas realizadas durante la iteración.

|  |  |
| --- | --- |
| **Características** | **Tareas** |
| Configuración animaciones personaje principal | Incluir el modelo del personaje principal en Unity 3D y configurar su prefab. |
| Implementar collider. |
| Incorporar las animaciones al prefab. |
| Agregar las decisiones al controlador para la administración de las animaciones. |
| Configurar animaciones cocodrilo | Incluir el modelo del cocodrilo en Unity 3D y configurar su prefab. |
| Implementar collider. |
| Incorporar las animaciones al prefab. |
| Agregar las decisiones al controlador para la administración de las animaciones. |
| Configuración animaciones serpiente | Incluir el modelo de la serpiente en Unity 3D y configurar su prefab. |
| Implementar collider. |
| Incorporar las animaciones al prefab. |
| Agregar las decisiones al controlador para la administración de las animaciones. |
| Configuración animaciones oso | Incluir el modelo del oso en Unity 3D y configurar su prefab. |
| Implementar collider. |
| Incorporar las animaciones al prefab. |
| Agregar las decisiones al controlador para la administración de las animaciones. |
| Configuración animaciones tigre | Incluir el modelo del tigre en Unity 3D y configurar su prefab. |
| Implementar collider. |
| Incorporar las animaciones al prefab. |
| Agregar las decisiones al controlador para la administración de las animaciones. |

Tabla 11. Descomposición de las características de la segunda iteración en tareas

**Seguimiento de la iteración**

En el transcurso de la iteración se presentaron las siguientes situaciones relevantes:

* Al configurar las animaciones en la máquina de estados de Unity 3D se presentaron problemas en las transiciones, es por esto que se tomó la decisión de efectuar las animaciones directamente en el controlador de cada personaje.
* La malla no era del mismo tamaño que el modelo, lo cual tomo bastante tiempo al equipo de desarrollo reparar.
* El tiempo previsto para la iteración concluyó sin terminar todas las tareas, por lo cual se dio una prórroga de una semana teniendo en cuenta el estado actual de las tareas de la iteración y los problemas que han surgido, con el objetivo de que los desarrolladores realizaran retroalimentaciones con el diseñador y se documentaran para la solución de errores.

**Cierre y evaluación**

Al final de la prórroga se habían implementado todas las características definidas y los objetivos se cumplieron, por ende esta iteración a pesar de no cumplirse en el tiempo establecido se cierra con un 100% de las tareas realizadas.

**Lecciones aprendidas**

Como lecciones aprendidas se obtuvo que:

* Al implementar los modelos en Unity 3D se debe verificar que sus huesos y malla estén correctamente ajustados.
* Para escalar un modelo es recomendable usar la propiedad scale factor del modelo y no el transform de las instancias.

**Mejoras al proceso**

Como mejora al proceso de creación del videojuego se obtuvo:

* El uso de prefabs hace que la configuración de un game object pueda ser reutilizada.

#### Iteración 3

En la tercera iteración se realizó la configuración de la cámara, la cual consistió en definir su resolución para que se adaptará a los dispositivos y realizara el seguimiento adecuado al personaje principal, además, se configuraron las físicas del personaje y os rivales con respecto al escenario, para en posteriores iteraciones poder validar la interacción entre objetos. El documento completo de la planificación, seguimiento y cierre de la iteración se llamó *Documento de desarrollo de iteración 3* y se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%203.docx)).

**Objetivos**

Se definieron los siguientes objetivos:

* Configurar que la cámara le haga seguimiento al personaje.
* Configurar las físicas del personaje principal para que los movimientos sean realistas.
* Configurar las físicas del prefab de la serpiente.
* Configurar las físicas del prefab del cocodrilo.
* Configurar las físicas del prefab del oso.
* Configurar las físicas del prefab del tigre.
* Configurar las físicas de los objetos.

**Selección de características**

De acuerdo a los objetivos y al documento de especificación de características, las seleccionadas para esta iteración fueron las plasmadas en la Tabla 12, donde se destaca la configuración de las físicas que afectan los personajes.

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **ID** | **Característica** |
| 7 | 22 | Configuración cámara |
| 6 | 16 | Físicas personaje principal |
| 3 | 17 | Físicas cocodrilo |
| 3 | 18 | Físicas oso |
| 3 | 19 | Físicas tigre |
| 3 | 20 | Físicas serpiente |
| 3 | 21 | Físicas objetos |

Tabla 12. Características seleccionadas para la iteración 3

**Refinación de características**

Las características seleccionadas fueron descompuestas en tareas, de las cuales la mayoría fueron definidas en la planificación de la iteración, pero durante el desarrollo los programadores detectaron la necesidad de incluir nuevas tareas. La Tabla 13 contiene la totalidad de tareas realizadas durante la iteración.

|  |  |
| --- | --- |
| **Características** | **Tareas** |
| Configuración cámara | Configurar la resolución de la cámara |
| Configurar el scroll parallax |
| Hacer que la cámara siga al personaje de izquierda a derecha mas no en sentido contrario |
| Físicas personaje principal | Configurar las físicas del personaje para que los movimientos y acciones del personaje sean realistas y de acorde a los obstáculos |
| Físicas cocodrilo | Configurar las físicas del prefab del cocodrilo para que sus movimientos y acciones sean realistas y de acorde a los obstáculos |
| Físicas oso | Configurar las físicas del prefab del oso para que sus movimientos y acciones sean realistas y de acorde a los obstáculos |
| Físicas tigre | Configurar las físicas del prefab del tigre para que sus movimientos y acciones sean realistas y de acorde a los obstáculos |
| Físicas serpiente | Configurar las físicas del prefab de la serpiente para que sus movimientos y acciones sean realistas y de acorde a los obstáculos |
| Físicas objetos | Configurar las físicas de lo prefabs de los diversos objetos para que sus movimientos y acciones sean realistas y de acorde a al entorno |

Tabla 13. Descomposición de las características de la tercera iteración en tareas

**Seguimiento de la iteración**

En el transcurso de la iteración se presentó la siguiente situación relevante:

* El scroll parallax volvía a su posición inicial una vez culminaba un movimiento afectando la perspectiva del entorno y su solución llevó un tiempo considerable.

**Cierre y evaluación**

A pesar de la situación adversa mencionada anteriormente, las tareas fueron realizadas en su totalidad y a tiempo, por ende esta iteración se cierra con una ejecución del 100%.

**Lecciones aprendidas**

Como principal lección aprendida se obtuvo que:

* El uso de transiciones de imágenes es muy eficaz para animar componentes como el agua.

**Mejoras al proceso**

Como mejora al proceso de creación del videojuego se obtuvo:

* El uso de un background estático, al cual se le modifica la variable offset consume menos recursos que instanciar varios elementos en toda la escena.

#### Iteración 4

En la cuarta iteración se realizó el código necesario para validar las interacciones entre el personaje, los rivales y los objetos, con el objetivo de en futuras iteraciones usar todos los elementos y prefabs creados para finalizar el primer nivel y sea más sencilla la implementación de los niveles restantes. El documento completo de la planificación, seguimiento y cierre de la iteración se llamó *Documento de desarrollo de iteración 4* y se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración 4](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%204.docx)).

**Objetivos**

Se definieron los siguientes objetivos:

* Codificar y validar la interacción entre el personaje y el rival tipo serpiente.
* Codificar y validar la interacción entre el personaje y el rival tipo cocodrilo.
* Codificar y validar la interacción entre el personaje y el rival tipo oso.
* Codificar y validar la interacción entre el personaje y el rival tipo tigre.
* Codificar y validar la interacción entre el personaje y el rival y los objetos de la escena.
* Habilitar y validar la interacción del cambio de arma.

**Selección de características**

De acuerdo a los objetivos y al documento de especificación de características, las seleccionadas para esta iteración fueron las plasmadas en la Tabla 14, donde se destaca la configuración de las interacciones de los personajes.

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **ID** | **Característica** |
| 5 | 31 | Interacción personaje - serpiente |
| 5 | 32 | Interacción personaje - cocodrilo |
| 5 | 33 | Interacción personaje - oso |
| 5 | 34 | Interacción personaje - tigre |
| 5 | 35 | Interacción personaje - objetos |
| 4 | 36 | Cambio de arma |

Tabla 14. Características seleccionadas para la iteración 4

**Refinación de características**

Las características seleccionadas fueron descompuestas en tareas, de las cuales la mayoría fueron definidas en la planificación de la iteración, pero durante el desarrollo los programadores detectaron la necesidad de incluir nuevas tareas. La Tabla 15 contiene la totalidad de tareas realizadas durante la iteración.

|  |  |
| --- | --- |
| **Características** | **Tareas** |
| Interacción personaje - serpiente | Generar el código necesario para implementar las interacciones por parte del personaje |
| Generar el código necesario para implementar las interacciones por parte de la serpiente |
| Interacción personaje - cocodrilo | Generar el código necesario para implementar las interacciones por parte del personaje |
| Generar el código necesario para implementar las interacciones por parte del cocodrilo |
| Interacción personaje - oso | Generar el código necesario para implementar las interacciones por parte del personaje |
| Generar el código necesario para implementar las interacciones por parte del oso |
| Interacción personaje - tigre | Generar el código necesario para implementar las interacciones por parte del personaje |
| Generar el código necesario para implementar las interacciones por parte del tigre |
| Interacción personaje - objetos | Generar el código necesario para implementar las interacciones por parte del personaje |
| Generar el código necesario para implementar las interacciones por parte de cada objeto |
| Cambio de arma | Habilitar el cambio de arma |
| Validar la interacción del personaje con cada arma |

Tabla 15. Descomposición de las características de la cuarta iteración en tareas

**Seguimiento de la iteración**

En el transcurso de la iteración se presentó la siguiente situación relevante:

* La longitud de la lanza del personaje principal era demasiado corta para realizar una interacción adecuada.

**Cierre y evaluación**

En esta iteración se llevaron a cabo la totalidad de las tareas y se cumplieron los objetivos sin mayor percance, por ende esta iteración se cierra con una ejecución del 100% de tareas.

**Lecciones aprendidas**

Como principal lección aprendida se obtuvo que:

* A medida que el proyecto avanza los desarrolladores se documentan acerca de Unity 3D enriqueciendo sus conocimientos y experiencia.

**Mejoras al proceso**

Como mejora al proceso de creación del videojuego se obtuvo:

* El uso de diversos colliders y triggers en un mismo game object facilita la validación de acciones y eventos.

#### Iteración 5

En la quinta iteración se realizó la importación del framework “Unity ETH UQ” y se realizaron el menú principal, menú en juego, HUD, menú de game over, además, se configuró el soporte de idiomas y los controles del personaje, todo lo anterior utilizando dicho framework. El documento completo de la planificación, seguimiento y cierre de la iteración se llamó *Documento de desarrollo de iteración 5* y se encuentra adjunto con el presente informe ([Ver Documento del desarrollo de iteración 5](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%205.docx)).

**Objetivos**

Se definieron los siguientes objetivos:

* Importar el framework “Unity ETH UQ” en el videojuego.
* Crear las texturas para los backgrounds de los menús y sus componentes.
* Crear y poner en funcionamiento el menú principal.
* Crear y poner en funcionamiento el HUD.
* Crear y poner en funcionamiento el menú en juego.
* Poner en funcionamiento el soporte de idiomas.
* Crear y poner en funcionamiento la ventana de créditos.
* Crear y poner en funcionamiento el menú de configuraciones.

**Selección de características**

De acuerdo a los objetivos y al documento de especificación de características, las seleccionadas para esta iteración fueron las plasmadas en la Tabla 16, donde se destaca la elaboración de los diferentes menús.

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **ID** | **Característica** |
| 4 | 23 | Texturas de las ventanas |
| 7 | 29 | Importación del framework |
| 4 | 24 | Menú principal |
| 3 | 25 | Menú de configuración |
| 2 | 26 | UI de créditos |
| 2 | 28 | Menú en juego |
| 3 | 37 | Elaborar HUD |
| 2 | 38 | Configurar soporte de idiomas |

Tabla 16. Características seleccionadas para la iteración 5

**Refinación de características**

Las características seleccionadas fueron descompuestas en tareas, de las cuales la mayoría fueron definidas en la planificación de la iteración, pero durante el desarrollo los programadores detectaron la necesidad de incluir nuevas tareas. La Tabla 17 contiene la totalidad de tareas realizadas durante la iteración.

|  |  |
| --- | --- |
| **Características** | **Tareas** |
| Texturas de las ventanas | Crear las texturas del background del menú principal |
| Crear las texturas en español de los botones del menú principal |
| Crear las texturas en inglés de los botones del menú principal |
| Crear la textura en español del background del menú de configuraciones |
| Crear la textura en inglés del background del menú de configuraciones |
| Crear las texturas de los botones del menú de configuraciones |
| Crear la textura del background para los demás menús |
| Crear las texturas para los botones del HUD |
| Crear las texturas en español para el menú en juego |
| Crear las texturas en inglés para el menú en juego |
| Crear las texturas en español para indicar el game over |
| Crear las texturas en inglés para indicar el game over |
| Importación del framework | Realizar la importación del framework “Unity ETH UQ” en el proyecto |
| Menú principal | Elaborar la escena del menú principal |
| Configurar las dimensiones de la cámara |
| Crear y configurar los botones |
| Menú de configuración | Elaborar la ventana modal del menú de configuración |
| Agregar y configurar los botones toggle para seleccionar el idioma |
| Agregar y configurar los slider para configurar el volumen de la música y efectos |
| UI de créditos | Elaborar la ventana modal para los créditos |
| Agregar el texto de los créditos |
| Menú en juego | Elaborar la ventana modal del menú en juego |
| Crear y configurar los botones |
| Elaborar HUD | Crear y configurar los botones que componen el HUD |
| Configurar soporte de idiomas | Agregar soporte de idiomas a las texturas de background que sea necesario |
| Crear el archivo XML para que funcione como diccionario de idiomas |
| Agregar soporte de idiomas a los textos |

Tabla 17. Descomposición de las características de la quinta iteración en tareas

**Seguimiento de la iteración**

En el transcurso de la iteración se presentaron las siguientes situaciones relevantes:

* Los elementos de tipo EthImage al tener soporte de idiomas no conservaban su tamaño inicial y entorpecían la correcta distribución de la escena, para ello los programadores encontraron que la última línea del método setImageLang reconfiguraba su tamaño ignorando las dimensiones del game object padre, cuya solución fue deshabilitar dicha línea de código.
* Para configurar el soporte de idiomas del componente texto de los botones se presentaba un error. Los programadores encontraron que cuando se hacía la petición al diccionario de datos se estaba haciendo uso de una variable privada que no referenciaba al componente texto, se cambió dicha variable privada por una publica que ya estaba configurada en la clase y si hacía referencia al elemento texto.

**Cierre y evaluación**

A pesar de las situaciones adversas anteriormente nombradas se llevaron a cabo todas las tareas y se cumplieron los objetivos, por lo cual esta iteración se cierra con una ejecución del 100%.

**Lecciones aprendidas**

Como principal lección aprendida se obtuvo que:

* Los programadores se familiarizaron con el funcionamiento del framework, lo cual es crucial para la validación de sus funcionalidades.

**Mejoras al proceso**

Como mejora al proceso de creación del videojuego se obtuvo:

* Los programadores enriquecen su conocimiento sobre el framework leyendo la documentación de las clases y el análisis realizado en la primera fase del proyecto.

#### Iteración 6

En la sexta y última iteración se crearon los tres niveles restantes, cuya puesta en funcionamiento no tuvo mayor complejidad ya que los prefabs estaban configurados, por ende solo se debieron configurar aspectos específicos de cada nivel. Además se realizó la conexión entre escenas y se implementó un cargador que se muestra entre escenas y finalmente se configuró el audio, quedando así el videojuego listo para iniciar la fase de pruebas. El documento completo de la planificación, seguimiento y cierre de la iteración se llamó *Documento de desarrollo de iteración 6* y se encuentra adjunto con el presente informe ([Ver Documento del desarrollo de iteración 6](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%206.docx)).

**Objetivos**

Se definieron los siguientes objetivos:

* Crear el nivel 1.
* Crear el nivel 2.
* Crear el nivel 3.
* Configurar las conexiones entre escenas.
* Crear y configurar el cargador para mostrar entre escenas.

**Selección de características**

De acuerdo a los objetivos y al documento de especificación de características, las seleccionadas para esta iteración fueron las plasmadas en la Tabla 18, donde se destaca la creación de los escenarios restantes.

|  |  |  |
| --- | --- | --- |
| **Prioridad** | **ID** | **Característica** |
| 3 | 13 | Primer escenario |
| 2 | 14 | Segundo escenario |
| 2 | 15 | Tercer escenario |
| 2 | 27 | Cargador de escenas |
| 3 | 30 | Conexión de escenas |

Tabla 18. Características seleccionadas para la iteración 6

**Refinación de características**

Las características seleccionadas fueron descompuestas en tareas, de las cuales la mayoría fueron definidas en la planificación de la iteración, pero durante el desarrollo los programadores detectaron la necesidad de incluir nuevas tareas. La Tabla 19 contiene la totalidad de tareas realizadas durante la iteración.

|  |  |
| --- | --- |
| **Características** | **Tareas** |
| Primer escenario | Instanciar y posicionar la cámara |
| Instanciar y posicionar los terrenos del nivel |
| Instanciar y posicionar los obstáculos de agua |
| Instanciar y posicionar los rivales |
| Configurar los aspectos propios del nivel |
| Verificar el funcionamiento del nivel |
| Segundo escenario | Instanciar y posicionar la cámara |
| Instanciar y posicionar los terrenos del nivel |
| Instanciar y posicionar los obstáculos de agua |
| Instanciar y posicionar los rivales |
| Configurar los aspectos propios del nivel |
| Verificar el funcionamiento del nivel |
| Tercer escenario | Instanciar y posicionar la cámara |
| Instanciar y posicionar los terrenos del nivel |
| Instanciar y posicionar los obstáculos de agua |
| Instanciar y posicionar los rivales |
| Configurar los aspectos propios del nivel |
| Verificar el funcionamiento del nivel |
| Cargador de escenas | Crear las texturas para el cargador en español |
| Crear las texturas para el cargador en inglés |
| Crear el prefab del cargador |
| Conexión de escenas | Generar el código y las validaciones para realizar la conexión entre las escenas |
| Implementar el cargador en cada conexión |

Tabla 19. Descomposición de las características de la quinta iteración en tareas

**Seguimiento de la iteración**

En el transcurso de la iteración se presentó la siguiente situación relevante:

* Al instanciar todos los elementos de los nuevos niveles, se detectó que el HUD no funcionaba, lo cual era debido a que no se estaba instanciando el sistema de eventos en los niveles nuevos.

**Cierre y evaluación**

A pesar de las situaciones adversas anteriormente nombradas se llevaron a cabo todas las tareas y se cumplieron los objetivos, por lo cual esta iteración se cierra con una ejecución del 100%.

**Lecciones aprendidas**

Como principal lección aprendida se obtuvo que:

* A pesar de que interpretar el framework para ponerlo en funcionamiento tomo un tiempo considerable a los programadores, la reutilización de dichas funciones para diversas situaciones ahorra tiempo ya que evita que los programadores tengan que documentarse y evita le reiteración de errores.

**Mejoras al proceso**

Como mejora al proceso de creación del videojuego se obtuvo:

* Dada la relativa facilidad con la que se crearon los niveles, el equipo ratificó que el uso de prefabs y configuraciones al iniciar el desarrollo ahorra bastante tiempo y recursos.

## Fase 3: Pruebas

Esta fase tiene como objetivos evaluar y ajustar distintos aspectos de jugabilidad del videojuego, además de eliminar la mayor cantidad de errores detectados (Gemserk, 2008). Según la metodología de desarrollo SUM en esta fase se  trabaja en forma iterativa liberando distintas versiones del videojuego para verificar su jugabilidad. En cada ciclo primero se planifica y distribuye la versión beta para ser verificada, paralelamente, se envían reportes con los errores o evaluaciones realizadas. Estos reportes son analizados para ver la necesidad de realizar ajustes al videojuego. Dicho proceso fue realizado en tres iteraciones, donde el aspecto más relevante es que los verificadores de cada iteración eran diferentes, con el fin de detectar la mayor cantidad de errores posible.

Para llevar a cabo esta fase, el equipo de trabajo creó una plantilla, la cual se llamó *Plantilla* *de desarrollo de iteración de pruebas,* donde se llevaría registro de la planeación, seguimiento y cierre de cada iteración, de tal manera que una vez planificada la iteración los verificadores beta realizaban la prueba de jugabilidad y reportaban errores y sugerencias para que posteriormente el equipo de trabajo definiera y priorizara los cambios y mejoras que debían ser realizados y finalmente, una vez realizados dichos cambios se procediera a cerrar la iteración y si había una siguiente continuar con ella. La plantilla de desarrollo de las iteraciones de prueba se encuentra adjunta con el presente informe ([Ver Plantilla de desarrollo de iteración de pruebas](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Plantilla%20de%20desarrollo%20de%20iteración%20de%20pruebas.docx))

En cuanto a los verificadores beta, estos cambiaron cada iteración, con el fin de que cada iteración fuera objetiva, para la primera fueron los programadores, los cuales reportaron los errores y sugerencias en un documento Excel llamado *Reporte de pruebas programadores,* el cual se encuentra adjunto con el presente informe ([Ver Reporte de pruebas programadores](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Reporte%20de%20pruebas%20programadores.xlsx)). Para la segunda iteración los verificadores beta fueron cuatro integrantes del Grupo de Investigación en Redes, Información y Distribución – GRID de la universidad del Quindío, quienes reportaron los errores y sugerencias mediante un formulario creado por los programadores en google, al cual se puede acceder mediante la URL <https://goo.gl/forms/c9MIVwuONw1yzAW53> y cuyo consolidado de respuestas se encuentra adjunto con el presente informe ([Ver Consolidado de respuestas iteración 2](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Consolidado%20de%20respuestas%20iteración%202.xlsx)). Finalmente para la última iteración de pruebas los verificadores beta fueron personas totalmente ajenas al proyecto quienes registraron sus sugerencias y errores detectados mediante el mismo mecanismo que se utilizó en la iteración anterior y cuyo consolidado de respuestas se encuentra adjunto con el presente informe ([Ver Consolidado de respuestas iteración 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Consolidado%20de%20respuestas%20iteración%203.xlsx)).

### Iteración 1 de pruebas

Esta iteración tuvo como objetivo principal refinar el videojuego y su proceso se detalla a grandes rasgos a continuación. El documento completo de planeación, identificación de errores, seguimiento y cierre de esta iteración se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración de pruebas 1](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%20de%20pruebas%201.docx)).

#### Definir medio de distribución

El medio por el cual se distribuyó el videojuego a los verificadores beta fue por medio de correo electrónico a cada uno, donde se les brindaba acceso al archivo apk que debían instalar en sus dispositivos para poder probar el videojuego.

#### Definir verificadores beta

Dado que esta fue la primera iteración de la fase de pruebas y el objetivo era refinar el videojuego, los verificadores beta fueron los programadores, cuyos nombres y roles en el proyecto se muestran en la Tabla 20.

|  |  |
| --- | --- |
| **Verificador beta** | **Rol(es) en el proyecto** |
| Dival Mauricio Hoyos Castro | Programador  Diseñador de juego |
| Julian David Serna Echeverri | Programador  Diseñador de juego |

Tabla 20. Verificadores beta de la primera iteración

#### Definir como se reportan los errores

Se les brindó acceso a los verificadores beta a una hoja de cálculo en google drive para reportar los errores y sugerencias, donde debieron detallar en que consistía el error y definir los pasos para reproducirlo. Dicha hoja de cálculo fue descargada como archivo Excel para el posterior análisis y se encuentra adjunto con el presente informe ([Ver Reporte de pruebas programadores](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Reporte%20de%20pruebas%20programadores.xlsx)).

#### Identificación de incidencias

El equipo de trabajo analizó el reporte de los verificadores e identificó y priorizó los errores plasmados en la Tabla 21.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Error a solucionar** | **Descripción** | **Prioridad** |
| 1 | Cuando el personaje salta y cae muy al borde del terreno se queda pegado. | Una parte del collider del personaje queda sobre el terreno, pero no el verificador de suelo que se encuentra en la mitad de dicho collider, es por esto que el personaje queda sobre el terreno pero no puede avanzar. | 9 |
| 2 | El menú de configuraciones se sobrepone al botón de configuraciones y se hace muy difícil cerrar dicho menú. | El menú de configuraciones cubre el botón en algunos dispositivos, lo cual hace imposible cerrar dicha ventana. | 8 |
| 3 | Los saltos no siempre son iguales. | Cuando el personaje salta, en ocasiones lo hace con más fuerza que en otras. | 9 |
| 4 | El personaje muere pero no se hunde en el agua. | Al tocar el agua, el personaje muere, pero queda flotando. | 7 |
| 5 | En el primer nivel en la sección de pasar dos cocodrilos consecutivamente es casi imposible. | Los cocodrilos se despiertan muy rápido y el primero no da tiempo de dormir al segundo. | 7 |

Tabla 21.Errores identificados y priorizados de la iteración 1 de la fase de pruebas

#### Seguimiento de la iteración

De acuerdo a los errores plasmados en la Tabla 21, y en orden de prioridad el equipo de trabajo solucionó dichos errores. En la tabla 22 se muestra si cada error fue solucionado exitosamente y se da una descripción de su solución.

|  |  |  |
| --- | --- | --- |
| **ID** | **Solución identificada** | **Solucionado** |
| 1 | El verificador de suelo está en el centro del collider del personaje y genera el error, la solución es la implementación de dos verificadores de suelo, uno en el extremo derecho y otro al izquierdo del collider. | Si |
| 3 | El salto se realiza generando una fuerza en el eje Y, alguna de las variables de la fuerza cambian y por ende su resultado es diferente, la solución es cambiar dicha fuerza por una velocidad en el mismo eje. | Si |
| 2 | Debido a que el juego será ejecutado en diversos dispositivos la mejor solución es crear un botón de cierre en el panel del menú de configuraciones. | Si |
| 4 | La solución es agregar una validación para que cuando el personaje esté sobre el agua lo afecte la gravedad pero a menor proporción. | Si |
| 5 | La solución es aumentar el tiempo del temporizador que genera el evento de despertar en los cocodrilos. | Si |

Tabla 22. Seguimiento a los errores identificados en la iteración 1 de la fase de pruebas

#### Cierre

Esta iteración se realizó con el objetivo de refinar el videojuego, pero al ser sus propios desarrolladores los verificadores, hay una alta probabilidad de omitir errores, por lo cual el número de errores reportados en las siguientes iteraciones fue mayor. Los errores identificados fueron corregidos en su totalidad, quedando así el proyecto listo para la siguiente iteración de pruebas.

### Iteración 2 de pruebas

Esta iteración tuvo como objetivo someter el videojuego a una primera interacción con personas diferentes al equipo de trabajo, pero que a la vez fueran conocedoras del proyecto y tuvieran conocimientos y experiencia en el desarrollo de videojuegos. El documento completo de planeación, identificación de errores, seguimiento y cierre de esta iteración se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración de pruebas 2](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%20de%20pruebas%202.docx)).

#### Definir medio de distribución

El medio por el cual se distribuyó el videojuego a los verificadores beta fue por medio de correo electrónico a cada uno, donde se les brindaba acceso al archivo apk que debían instalar en sus dispositivos y al formulario para reportar los hallazgos.

#### Definir verificadores beta

Debido a que el presente proyecto fue concebido como un trabajo de grado para validar las funcionalidades de los módulos audio, components, útil y appSystem del framework “Unity ETH UQ”, el cual es un proyecto de investigación llevado a cabo por el Grupo de Investigación en Redes, Información y distribución – GRID de la universidad del Quindío en colaboración con la empresa de desarrollo de videojuegos Ethereal GF, es por esto que los integrantes del GRID conocen el presente proyecto y dos de sus integrantes que también fueron parte del equipo de trabajo de Ethereal GF serán los verificadores beta, los cuales fueron Andrés Herrera y Einer Zapata.

#### Definir como se reportan los errores

Los verificadores beta tuvieron acceso a un formulario alojado en google drive, el cual puede ser accedido en <https://goo.gl/forms/gyLh21dar1CU9Vmc2>,donde debieron responder algunas preguntas concretas y tuvieron espacio para reportar las sugerencias y errores encontrados. El formulario contiene la siguiente estructura:

* Inicialmente se brinda una introducción y se informa al verificador el objetivo del formulario.
* Se pide que el verificador registre su nombre y apellido.
* Se pide que seleccione como considera la historia del videojuego entre muy bueno, bueno, regular, malo y muy malo.
* Se pide que seleccione como considera los gráficos y personajes del videojuego entre muy buenos, buenos, regulares, malos y muy malos.
* Se pide que seleccione como considera la música y efectos de sonido del videojuego entre muy buenos, buenos, regulares, malos y muy malos.
* Se pide que seleccione un valor entre 0 y 10 en el cual considera que tan entretenido es el videojuego, siendo 0 nada entretenido y 10 muy entretenido.
* Se pide que describa cada uno de los errores encontrados y los pasos para generarlo.
* Finalmente se pide que el verificador detalle las sugerencias que tenga sobre el videojuego.

Una vez todos los verificadores hicieron su reporte, se descargó el consolidado de respuestas como una hoja de cálculo de Excel para el análisis de los reportes, el cual se encuentra adjunto con el presente informe ([Ver Consolidado de respuestas iteración 2](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Consolidado%20de%20respuestas%20iteración%202.xlsx)).

#### Identificación de incidencias

El equipo de trabajo analizó el reporte de los verificadores e identificó y priorizó los errores plasmados en la Tabla 23.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Error a solucionar** | **Descripción** | **Prioridad** |
| 6 | El fondo se mueve de manera extraña. | Cuando el personaje inicia el movimiento el fondo se mueve muy rápido y sin detenerse. | 8 |
| 7 | La música de fondo debe ser diferente. | La música de fondo no es acorde al contexto. | 5 |
| 8 | En ocasiones el dardo no toca la serpiente pero aun así la mata. | Se debe ajustar la validación de colisiones del dardo. | 6 |
| 9 | En ocasiones al atacar con la lanza el rival muere, pero el personaje no realiza la animación de ataque. | Se debe validar el ataque con lanza. | 6 |
| 10 | Cuando un rival mata al personaje el sonido de efecto queda durante toda la sesión de juego. | Se debe verificar la validación en que los efectos paran. | 5 |

Tabla 23. Errores identificados y priorizados de la iteración 2 de la fase de pruebas

#### Seguimiento de la iteración

De acuerdo a los errores plasmados en la Tabla 23, y en orden de prioridad el equipo de trabajo solucionó dichos errores. En la tabla 24 se muestra si cada error fue solucionado exitosamente y se da una descripción de su solución.

|  |  |  |
| --- | --- | --- |
| **ID** | **Solución identificada** | **Solucionado** |
| 6 | Verificar las validaciones de scroll parallax para que su movimiento sea acorde con el movimiento de la cámara y el personaje. | Si |
| 8 | Se debe ajustar la malla que detecta las colisiones de la serpiente. | Si |
| 9 | Validar que la animación sea a tiempo con la acción de ataque. | Si |
| 7 | Se debe implementar una canción de fondo ambiental. | Si |
| 10 | Validar las condiciones para que cuando el personaje muera los efectos paren. | Si |

Tabla 24. Seguimiento a los errores identificados en la iteración 2 de la fase de pruebas

#### Cierre

Esta iteración se realizó con el objetivo de que verificadores ajenos al equipo de trabajo, pero con experiencia en el desarrollo de videojuegos y conocimiento del proyecto realizaran las pruebas para encontrar la mayor cantidad de errores, a los cuales se les realizaron las respectivas correcciones, quedando así el juego listo para la tercera iteración de pruebas siendo sometido a un público más amplio.

### Iteración 3 de pruebas

Esta iteración tuvo como objetivo someter al videojuego a un público más en comparación con las dos iteraciones anteriores, con el fin de que al haber más verificadores fuera menor la probabilidad de pasar un error por alto. El documento completo de planeación, identificación de errores, seguimiento y cierre de esta iteración se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración de pruebas 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Consolidado%20de%20respuestas%20iteración%203.xlsx)).

#### Definir medio de distribución

El medio por el cual se distribuyó el videojuego a los verificadores beta fue por medio de correo electrónico a cada uno, donde se les brindaba acceso al archivo apk que debían instalar en sus dispositivos y al formulario para reportar los hallazgos.

#### Definir verificadores beta

Dado que esta fue la tercera y última iteración de la fase de pruebas y su objetivo era someter el videojuego a un grupo amplio de verificadores, el equipo de trabajo seleccionó personas de su entorno que estuvieran dispuestas a realizar la prueba, para conocer los verificadores se debe observar el documento de planeación, identificación de errores, seguimiento y cierre de esta iteración que se encuentra adjunto con el presente informe ([Ver Documento de desarrollo de iteración de pruebas 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%20de%20pruebas%203.docx)).

#### Definir como se reportan los errores

Con el propósito de agilizar y hacerle a los verificadores voluntarios más fácil el proceso, se utilizó el formulario creado para la iteración anterior, del cual una vez registrados los reportes de todos los verificadores se generó el consolidado de respuestas como una hoja de cálculo de Excel para el análisis de los reportes, el cual se encuentra adjunto con el presente informe ([Ver Consolidado de respuestas iteración 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Consolidado%20de%20respuestas%20iteración%203.xlsx)).

#### Identificación de incidencias

El equipo de trabajo analizó el archivo de reportes ([Ver Consolidado de respuestas iteración 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Consolidado%20de%20respuestas%20iteración%203.xlsx)) de la iteración tres una vez todos los verificadores realizaron el reporte y se definieron y priorizaron las correcciones plasmadas en la Tabla 25.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Error a solucionar** | **Descripción** | **Prioridad** |
| 11 | Si el personaje se devuelve y cae a un precipicio el juego se pega | Cuando el personaje se devuelve y la cámara no lo sigue, si hay un precipicio y este cae, el juego se pega. | 8 |
| 12 | En la sección donde el personaje se desliza en el tercer nivel se puede saltar repetidas veces sin tocar el suelo | En la sección donde el personaje resbala en el tercer nivel se pueden realizar múltiples saltos y el personaje desaparece. | 8 |
| 13 | Cuando el personaje principal es asesinado el efecto de sonido del animal que infinito. | Al ser asesinado el personaje el sonido del animal queda sonando repetitivamente durante todo el juego hasta que se cierre. | 7 |
| 14 | En algunos bordes donde hay agua, si el personaje se acerca (aun estando en tierra) se muere. | El agua sobrepone el terreno y el personaje muere (ahogado). | 7 |
| 15 | Cuando el personaje salta y no alcanza a llegar al siguiente terreno, si queda pegado al muro y se puede volver a saltar. | Si el personaje queda pegado al muro, mas no encima del terreno se puede saltar cosa que no debería poderse hacer ya que la lógica es que caiga al precipicio. | 5 |

Tabla 25. Errores identificados y priorizados de la iteración 3 de la fase de pruebas

#### Seguimiento de la iteración

De acuerdo a los errores plasmados en la Tabla 25, y en orden de prioridad el equipo de trabajo solucionó dichos errores. En la tabla 26 se muestra si cada error fue solucionado exitosamente y se da una descripción de su solución.

|  |  |  |
| --- | --- | --- |
| **ID** | **Solución identificada** | **Solucionado** |
| 11 | Aumentar el rango del collider que detecta cuando el personaje ha caído. | Si |
| 12 | Validar el salto único en los triggers de la sección resbaladiza. | Si |
| 13 | Verificar las validaciones de muerte del personaje. | Si |
| 14 | Acomodar la posición del agua con respecto al terreno. | Si |
| 15 | Agregar un material sin fricción para los precipicios. | Si |

Tabla 26. Seguimiento a los errores identificados en la iteración 3 de la fase de pruebas

#### Cierre

Esta iteración se realizó con el objetivo de exponer el videojuego a un grupo más amplio de verificadores, con lo cual se logró el reporte de errores que habían pasado desapercibidos en las iteraciones anteriores. Los errores identificados fueron corregidos en su totalidad por ende se cerró esta iteración con una corrección del 100% de incidencias.

## Fase 4: Validación

La elaboración de esta fase se llevó a cabo paralelamente con las fases dos y tres debido a que en estas fases se utilizaron y refinaron las funcionalidades del framework identificadas en la fase uno del proyecto. Su objetivo es validar el funcionamiento de las funcionalidades de los módulos a evaluar, reportar incidencias, describir el proceso de implementación y realizar conclusiones que permitan conocer el estado actual del framework “Unity ETH UQ”. Para esto el equipo de trabajo realizó un informe, el cual se encuentra adjunto con el presente documento ([Ver Informe de hallazgos y validación](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Informe%20de%20hallazgos%20y%20validación%20del%20framework.docx)), en el cual se describen detalladamente los siguientes aspectos:

* Las funcionalidades de cada módulo.
* El proceso de implementación en el proyecto de cada funcionalidad.
* Conclusión del funcionamiento de cada funcionalidad de acuerdo a los resultados del proceso de implementación.
* Observaciones del funcionamiento de cada módulo.

Dicho informe describe que inicialmente el presente proyecto de grado estaba delimitado por la elaboración de un videojuego para la plataforma Android, que permitiera evaluar los componentes *util*, *display*, *audio* y *analythics* del proyecto “Framework de desarrollo de aplicaciones y juegos para sistemas IOS y Android en Unity con sistema de analíticas de uso y marketing” desarrollado por la empresa ETHEREAL GF y cuyo nombre clave es “Unity ETH UQ”. Pero en el trascurso de la etapa inicial del proyecto, la cual consistió en identificar las características, requerimientos y funcionalidades de dicho framework, Andrés Carvajal, administrador de proyecto de Ethereal GF informó al equipo de trabajo que debido a que la versión 5 de Unity 3D soporta elementos de interfaz de usuario el framework fue modificado y el módulo *display* fue reemplazado por el módulo *components*. Así mismo Andrés Herrera, desarrollador de Ethereal GF quien es el encargado de la mejora continua del módulo de analíticas informó al equipo de trabajo que dicho módulo había sido actualizado a la versión 2, lo cual implicaba que la administración de analíticas ya no se realizara en el módulo *analytics* sino en el módulo *appSystem*. Quedando así el presente proyecto delimitado por la elaboración de un videojuego para la plataforma Android, que permitiera evaluar los componentes *util*, *components*, *audio* y *appSystem* del framework “Unity ETH UQ”. De acuerdo a las funcionalidades de los anteriores módulos, identificadas en la primera fase del proyecto, las cuales se listaron en la Tabla 1, se procedió al desarrollo del informe como se describe a continuación.

### Módulo audio

Como se indica en la Tabla 27, la totalidad de las funcionalidades de este módulo fueron implementadas en el videojuego exitosamente. El proceso de implementación de cada funcionalidad fue detallado en el informe de validación ([Ver Informe de hallazgos y validación](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Informe%20de%20hallazgos%20y%20validación%20del%20framework.docx)).

|  |  |
| --- | --- |
| **Funcionalidad** | **Implementación exitosa** |
| Reproducir un audio | Si |
| Reproducir un efecto | Si |
| Reproducir un efecto de forma cíclica | Si |
| Modificar el volumen de la música | Si |
| Modificar el volumen de los efectos | si |

Tabla 27. Funcionalidades implementadas del módulo audio

#### Reproducir audio

Su implementación se realizó de tal manera que al iniciar el menú principal se reproduce una canción, la cual se repite mientras el juego esté abierto actuando como música de fondo. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que ejecuta un sonido de fondo mientras la aplicación está en ejecución.

#### Reproducir un efecto

Su implementación se realizó de tal manera que cuando el personaje principal realiza un ataque con lanza se reproduce un efecto de sonido y también cuando el cocodrilo ataca y mata al personaje principal. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que reproduce el efecto de sonido una única vez cada que se realiza el ataque con lanza o el cocodrilo ataca al personaje.

#### Reproducir un efecto de forma cíclica

Su implementación se llevó a cabo en diferentes secciones del videojuego, las cuales fueron:

* Cuando el personaje está cerca de una serpiente se reproduce el audio de la serpiente atacando cíclicamente hasta que esté la mate o se aleje.
* Cuando el personaje está cerca de un oso se reproduce el audio de la serpiente atacando cíclicamente hasta que esté lo mate o se aleje.
* Cuando el personaje está cerca de un tigre se reproduce el audio de la serpiente atacando cíclicamente hasta que esté lo mate o se aleje.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que reproduce el efecto de sonido cíclicamente hasta que el personaje se aleja del rival o le mata.

#### Modificar el volumen de la música

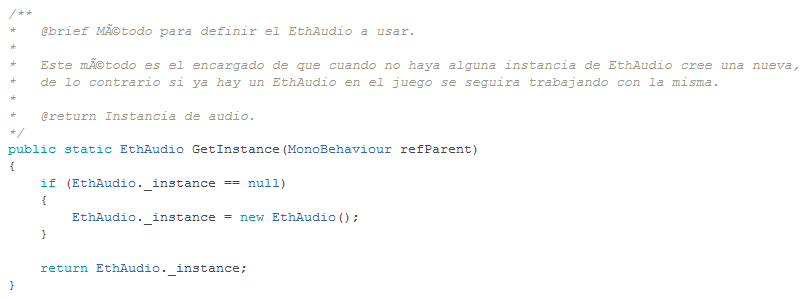
Su implementación se llevó a cabo en el menú de configuraciones, de tal manera que cuando el usuario desplace la barra que indica el volumen de la música se haga uso de la funcionalidad del framework y el volumen aumente o disminuya según el desplazamiento del usuario. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al desplazar el scroll el volumen de la música de fondo se modifica.

#### Modificar el volumen de los efectos

Su implementación se llevó a cabo en el menú de configuraciones, de tal manera que cuando el usuario desplace la barra que indica el volumen de los efectos se haga uso de la funcionalidad del framework y el volumen aumente o disminuya según el desplazamiento del usuario. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al desplazar el scroll el volumen de los efectos se modifica.

#### Observaciones del módulo audio

Para poder ejecutar los métodos que realizan las funcionalidades del módulo es necesario obtener la instancia actual de la clase *“EthAudio”,* para lo cual se debe utilizar el método *“GetInstance”,* el cualrecibe un parámetro que nunca es usado, por ende se sugiere la eliminación de este parámetro en el código fuente del framework. En la Ilustración 14 se muestra el método que debería ser modificado.



*Ilustración 14. Método GetInstance de la clase EthAudio que utiliza un parámetro innecesario*

### Módulo util

Como se indica en la Tabla 28, la totalidad de las funcionalidades de este módulo fueron implementadas en el videojuego exitosamente. El proceso de implementación de cada funcionalidad fue detallado en el informe de validación ([Ver Informe de hallazgos y validación](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Informe%20de%20hallazgos%20y%20validación%20del%20framework.docx)).

|  |  |
| --- | --- |
| **Funcionalidad** | **Implementación exitosa** |
| Manejo de Timer | Si |
| Definir si la plataforma en que se ejecuta la aplicación es móvil | Si |
| Modificar la visibilidad de un game object | Si |
| Obtener un game object hijo a partir de su nombre y su padre | Si |

Tabla 28. Funcionalidades implementadas del módulo util

#### Manejo de timer

Los EthTimer tienen como funcionalidad iniciar una espera y posteriormente ejecutar un método, los cuales fueron utilizados en las siguientes secciones del videojuego:

* Después de iniciar la animación de ataque con cerbatana esperar medio segundo y generar el dardo.
* Un segundo después de iniciar un ataque la variable auxiliar que define si el personaje está realizando un ataque cambie.
* Un segundo y medio después del personaje morir muestre la pantalla de game over.
* Hacer que el cocodrilo despierte después de cinco segundos de haber sido aturdido.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al crearse el EthTimer en las diferentes secciones después del tiempo especificado se ejecuta el método ingresado.

#### Definir si la plataforma en la que se ejecuta el aplicativo es móvil

Su implementación se llevó a cabo de tal forma que si la plataforma sobre la que se ejecuta la aplicación no es móvil los controles de movimiento y ataque no se muestren en pantalla debido a que se usarán las teclas. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que los controles de movimiento y ataque no se muestran cuando la aplicación no se ejecuta en un dispositivo móvil.

#### Modificar la visibilidad de un Game Object

Su implementación se llevó a cabo en el botón continuar del menú principal, de tal manera que si el usuario no tiene una partida iniciada en un nivel superior al tutorial este botón este oculto. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que cuando el usuario no tiene una partida iniciada en un nivel superior al tutorial el botón continuar no se muestra en el menú principal.

#### Obtener un Game Object hijo a partir de su nombre y su padre

Esta funcionalidad se utilizó en diferentes secciones del videojuego las cuales fueron:

* Para que cuando un rival muera, este obtenga su collider y lo cambie a trigger para que el personaje pueda continuar su camino.
* Para obtener el botón continuar, la ventana de configuración y el cargador para modificar su visibilidad al iniciar el menú.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que se inicializan los elementos y se pudieron cambiar sus propiedades de ahí en adelante.

#### Observaciones del módulo util

La utilización de las funcionalidades de este módulo no tuvo ningún problema, sin embargo se aconseja que para la clase *“EthUtils”* concuerde el nombre del archivo con el de la clase en el código, ya que el archivo se llama “EthUtils” y la clase en el código se llame “Eth”.

### Módulo components

Como se indica en la Tabla 29, la totalidad de las funcionalidades de este módulo no fueron implementadas en el videojuego, las causas que llevaron a no implementar algunas funcionalidades se detallan a continuación. El proceso de implementación de cada funcionalidad fue detallado en el informe de validación ([Ver Informe de hallazgos y validación](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Informe%20de%20hallazgos%20y%20validación%20del%20framework.docx)).

|  |  |
| --- | --- |
| **Funcionalidad** | **Implementación exitosa** |
| Soporte de idiomas a imagen | Si |
| Soporte de idiomas a texto | Si |
| Reproducción de audio a un toggle | Si |
| Soporte de idiomas a botón | Si |
| Reproducción de audio a un botón | Si |
| Modificar el valor de progreso a una barra de progreso | No |
| Animar la barra de progreso cuando se modifica su valor | No |
| Cargar una escena | Si |
| Al cargar una escena crear un panel sobre el canvas y mostrar un cargador | Si |
| Obtener el lenguaje del sistema | Si |

Tabla 29. Funcionalidades implementadas del módulo util

#### Soporte de idiomas a imagen

Su implementación se llevó a cabo en diferentes secciones del videojuego, las cuales fueron:

* Imagen de fondo de los botones de nuevo juego y continuar del menú principal.
* Imagen de fondo del menú de configuración.
* Imagen de fondo de los botones del menú en juego.
* Imagen de fondo del menú de game over.
* Imagen de fondo de los botones del menú de game over.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que la imagen de fondo cambia de acuerdo al idioma seleccionado.

#### Soporte de idiomas a texto

Su implementación se llevó a cabo en cada uno de los consejos que se dan al usuario en el nivel tutorial. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que el texto del consejo cambia de acuerdo al consejo a mostrar.

#### Reproducción de audio a un toggle

Su implementación se llevó a cabo en el menú de configuraciones en la selección de idioma, donde al dar clic en alguno de los botones toggle se reproduce un efecto de sonido. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al dar clic sobre el botón toggle se reproduce el efecto asignado.

#### Soporte de idiomas a botón

La implementación de esta funcionalidad se llevó a cabo en diferentes secciones del videojuego, las cuales fueron:

* Los botones de salto y ataque.
* El botón de pausa.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que el texto del botón cambia de acuerdo al idioma seleccionado.

#### Reproducción de audio a un botón

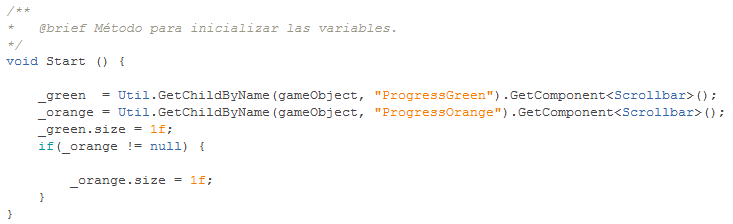
La implementación de esta funcionalidad se llevó a cabo en diferentes secciones del videojuego, las cuales fueron:

* Todos los botones del menú principal.
* Todos los botones del menú de configuraciones.
* Todos los botones del menú en juego.
* Todos los botones del menú de game over.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al hacer clic en el botón se reproduce el efecto especificado.

#### Modificar el valor de progreso a una barra de progreso

La implementación del componente *“EthUIProgressbar”* debido a que el script en el método *“Start”* en las líneas 35 y 36, mostradas en la Ilustración 15, hacen uso de dos Game Objects propios del último juego donde Ethereal GF implementó el framework.



*Ilustración 15. Líneas del script “EthUIProgressbar” que son propias de otro juego*

Al usar Game Objects propios de otro juego, este script se convierte en una solución adaptada y no una parte funcional del framework. Por ende se recomienda eliminar el script o corregir el código para dar soporte al componente progressbar.

#### Animar la barra de progreso cuando se modifica su valor

Debido a que el script *“EthUIProgressbar”* utiliza Game Objects de otro juego y no se puede implementar el componente, esta funcionalidad tampoco se puede implementar. Al esta funcionalidad ser derivada del componente “EthUIScrollbar” se recomienda modificar el código fuente para arreglar su funcionalidad o eliminar el componente y todos sus derivados.

#### Cargar una escena

La implementación de esta funcionalidad se llevó a cabo en diferentes secciones del videojuego, las cuales fueron:

* Para cargar el nivel tutorial al iniciar una nueva partida.
* Al superar un nivel, para cargar el siguiente.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al cumplir con las condiciones establecidas se realiza el cambio de escena.

#### Al cargar una escena crear un panel sobre el canvas y mostrar un cargador

La implementación de esta funcionalidad se llevó a cabo en la carga de las escenas más pesadas, las cuales fueron:

* Transición del menú principal al nivel tutorial.
* Transición del nivel tutorial al primer nivel.
* Transición del primer nivel al segundo nivel.
* Transición del segundo nivel al tercer nivel.

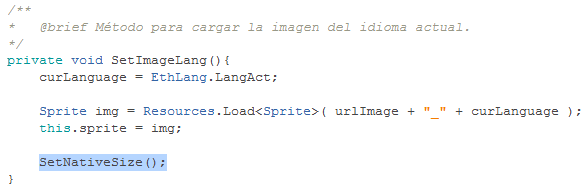
Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que mientras se carga la escena se muestra el cargador.

#### Obtener el lenguaje del sistema

La implementación de esta funcionalidad se llevó a cabo al iniciar el menú principal del videojuego, donde se valida si la variable que contiene el lenguaje ya tiene uno asignado, de lo contrario se toma el idioma del Sistema. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al iniciar el juego se configura un idioma.

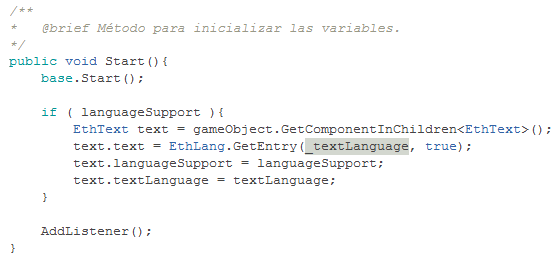
#### Observaciones del módulo components

Al configurar el soporte de idiomas a una imagen, esta se renderiza y no conserva el tamaño asignado al Game Object que la contiene, el equipo de trabajo encontró que la clase *“EthImage”* en el método *“SetImageLang”* en su última línea ejecuta el método *“etNativeSize”, el cual se muestra en la Ilustración 16,* lo cual generaba esta situación. Se recomienda modificar el método para evitar esta situación.



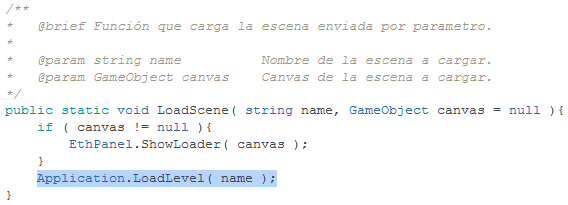
*Ilustración 16. Código que genera que la imagen con soporte de idiomas no conserve su tamaño*

Al configurar el soporte de idiomas de un componente *“EthUIButton”* el texto no cambia dependiendo del idioma, lo cual es debido a que en el método *“Start”* se utiliza la variable privada *“\_textLanguage”* para obtener los datos del diccionario, como se muestra en la Ilustración 17, cuando se debería usar la variable publica *“textLanguage”*.



*Ilustración 17. Variable que no permite el soporte de idiomas para el botón*

El método “LoadScene” del script “SceneHandler” hace uso de la instrucción “Application.LoadLevel” de unity, como se muestra en la Ilustración 18, la cual tiene una sobrecarga y puede recibir el nombre o el índice de la escena a cargar. Sería útil que el framework también tuviera dicha sobrecarga para permitir ingresar el índice de la escena.



*Ilustración 18. Utilización de la instrucción “Application.LoadLevel” de unity en el framework*

### Módulo appSystem

Como se indica en la Tabla 30, la totalidad de las funcionalidades de este módulo fueron implementadas en el videojuego exitosamente. El proceso de implementación de cada funcionalidad fue detallado en el informe de validación ([Ver Informe de hallazgos y validación](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Informe%20de%20hallazgos%20y%20validación%20del%20framework.docx)).

|  |  |
| --- | --- |
| **Funcionalidad** | **Implementación exitosa** |
| Inicializar el sistema de analíticas | Si |
| Registrar un evento en el sistema de analíticas | Si |
| Registrar una pantalla en el sistema de analíticas | Si |
| Obtener y modificar variables registradas en el sistema | Si |

Tabla 30. Funcionalidades implementadas del módulo appsSystem

#### Inicializar el sistema de analíticas

La implementación de esta funcionalidad se realizó de tal manera que cuando se ejecuta el juego, antes de mostrar el menú principal se inicie el sistema de analíticas, es por esto que antes de mostrar el menú principal se muestra un cargador. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al iniciar el juego se registra una nueva descarga o una nueva sesión en caso de ya estar registrada la descarga en la plataforma de analíticas de Ethereal GF.

#### Registrar un evento en el sistema de analíticas

La implementación de esta funcionalidad se realizó en diferentes secciones del juego con la finalidad de reportar diferentes eventos al sistema, las cuales fueron:

* Registrar el evento de iniciar un nuevo juego.
* Registrar cuando se abre el menú de configuraciones.
* Registrar cuando se inicia un nivel.
* Registrar cuando se completa un nivel.

Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al ejecutarse el método donde está incluida la instrucción “Log”, se registra el evento en la plataforma de analíticas de Ethereal GF.

#### Registrar una pantalla en el sistema de analíticas

La implementación de esta funcionalidad se realizó de tal manera que cuando se abra el menú de configuraciones o de pausa en un nivel se registra en el sistema de analíticas. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al ejecutarse el método donde está incluida la instrucción “LogScreen”, se registra la pantalla en la plataforma de analíticas de Ethereal GF.

#### Obtener y modificar variables registradas en el sistema

La implementación de esta funcionalidad se realizó de tal manera que cuando se inicia el juego, se obtiene la variable que indica el nivel actual del jugador y cuando se supera un nivel se modifica el valor de la variable. Al implementar esta funcionalidad se obtuvieron los resultados esperados ya que al obtener el valor de una variable este quedaba guardado en el juego, así mismo cuando en el juego se modifica una variable del sistema esta quedaba guardada en la plataforma de Ethereal GF.

#### Observaciones del módulo AppsSystem

A pesar de la documentación de las clases de este módulo, la interpretación de las funcionalidades es compleja, por lo cual se requirió el asesoramiento de Andrés Herrera. Además en la plataforma solo se actualizan los datos cada 24 horas, lo que hace que no se pueda verificar al 100% si los registros se están realizando de forma correcta.

# RESULTADOS

A continuación se describen y referencian los resultados obtenidos en cada uno de los objetivos específicos planteados.

## Identificar características y requerimientos del framework “Unity ETH UQ”

### Identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ”

Este resultado es un documento en el cual se plasman entrevistas realizadas a algunos de los miembros de Ethereal GF para identificar los requerimientos y potencialidades del framework “Unity ETH UQ”, posteriormente se encuentra un análisis de los módulos, sus clases y la interacción de métodos para luego identificar explícitamente las funcionalidades de cada módulo y finalmente se plasman los requerimientos del framework. Dicho documento se encuentra adjunto con el presente informe y su nombre es *identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ”* ([Ver identificación de características, requerimientos y funcionalidades del framework “Unity ETH UQ”](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Identificación%20de%20características,%20requerimientos%20y%20funcionalidades%20del%20framework)).

## Realizar el diseño de un videojuego utilizando características del framework “Unity ETH UQ”

### GDD Risky jungle

Este resultado es un documento en el cual se plasman todos los aspectos del diseño del videojuego con el objetivo de fijar un curso para empezar el desarrollo. Dicho documento se encuentra adjunto con el presente informe y su nombre es GDD Risky jungle ([Ver GDD Risky jungle](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\GDD%20Risky%20jungle.docx)).

## Desarrollar el videojuego utilizando el diseño realizado

### Selección de la metodología de desarrollo

Este resultado es un documento en el cual se formalizó la toma de la decisión de seleccionar la metodología de desarrollo utilizando la técnica DAR establecida por CMMI, donde se seleccionó la metodología SUM para desarrollo de videojuegos. Dicho documento se encuentra adjunto con el presente informe y su nombre es Selección de la metodología de desarrollo ([Ver Selección de la metodología de desarrollo](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Selección%20de%20la%20metodología%20de%20desarrollo.docx)).

### Definición de roles y responsabilidades

Este resultado es un documento en el cual se especificaron los roles y responsabilidades de cada miembro del equipo de trabajo durante el desarrollo del proyecto. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Definición de roles y responsabilidades* ([Ver Definición de roles y responsabilidades](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Definición%20de%20roles%20y%20responsabilidades.docx)).

### Definición de herramientas técnicas

Este resultado es un documento en el cual se especificaron las herramientas y su versión que iban a ser utilizadas en el transcurso del presente trabajo de grado. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Definición de herramientas técnicas* ([Ver Definición de herramientas técnicas](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Definición%20de%20herramientas%20técnicas.docx)).

### Especificación de características

Este resultado es un documento Excel que contiene las características del videojuego a ser implementadas. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Especificación de características* ([Ver Especificación de características](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Especificación%20de%20características.xlsx)).

### Plantilla de desarrollo de iteración

Este resultado es un documento que creó el equipo de trabajo con el objetivo de estandarizar los documentos de desarrollo de las iteraciones. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Plantilla de desarrollo de iteración* ([Ver Plantilla de desarrollo de iteración](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Plantilla%20de%20desarrollo%20de%20iteración.docx)).

### Documento de desarrollo de iteración 1

Este resultado es el documento basado en la plantilla de desarrollo de iteración y contiene la planeación, seguimiento y cierre de la primera iteración en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración 1* ([Ver Documento de desarrollo de iteración 1](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%201.docx)).

### Documento de desarrollo de iteración 2

Este resultado es el documento basado en la plantilla de desarrollo de iteración y contiene la planeación, seguimiento y cierre de la segunda iteración en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración 2* ([Ver Documento de desarrollo de iteración 2](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%202.docx)).

### Documento de desarrollo de iteración 3

Este resultado es el documento basado en la plantilla de desarrollo de iteración y contiene la planeación, seguimiento y cierre de la tercera iteración en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración 3* ([Ver Documento de desarrollo de iteración 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%203.docx)).

### Documento de desarrollo de iteración 4

Este resultado es el documento basado en la plantilla de desarrollo de iteración y contiene la planeación, seguimiento y cierre de la cuarta iteración en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración 4* ([Ver Documento de desarrollo de iteración 4](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%204.docx)).

### Documento de desarrollo de iteración 5

Este resultado es el documento basado en la plantilla de desarrollo de iteración y contiene la planeación, seguimiento y cierre de la quinta iteración en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración 5* ([Ver Documento de desarrollo de iteración 5](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%205.docx)).

### Documento de desarrollo de iteración 6

Este resultado es el documento basado en la plantilla de desarrollo de iteración y contiene la planeación, seguimiento y cierre de la sexta y última iteración en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración 6* ([Ver Documento de desarrollo de iteración 6](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%206.docx)).

### Plantilla de desarrollo de iteración de pruebas

Este resultado es un documento que creó el equipo de trabajo con el objetivo de estandarizar los documentos de desarrollo de las iteraciones de pruebas. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Plantilla de desarrollo de iteración de pruebas* ([Ver Plantilla de desarrollo de iteración de pruebas](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Plantilla%20de%20desarrollo%20de%20iteración%20de%20pruebas.docx)).

### Documento de desarrollo de iteración de pruebas 1

Este resultado es el documento basado en la plantilla de desarrollo de iteración de pruebas y contiene la planeación, seguimiento y cierre de la primera iteración de pruebas en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración de pruebas 1* ([Ver Documento de desarrollo de iteración de pruebas 1](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%20de%20pruebas%201.docx)).

### Documento de desarrollo de iteración de pruebas 2

Este resultado es el documento basado en la plantilla de desarrollo de iteración de pruebas y contiene la planeación, seguimiento y cierre de la segunda iteración de pruebas en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración de pruebas 2* ([Ver Documento de desarrollo de iteración de pruebas 2](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%20de%20pruebas%202.docx)).

### Documento de desarrollo de iteración de pruebas 1

Este resultado es el documento basado en la plantilla de desarrollo de iteración de pruebas y contiene la planeación, seguimiento y cierre de la tercera y última iteración de pruebas en el desarrollo del videojuego. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Documento de desarrollo de iteración de pruebas 3* ([Ver Documento de desarrollo de iteración de pruebas 3](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Documento%20de%20desarrollo%20de%20iteración%20de%20pruebas%203.docx)).

### Proyecto Unity “Risky Jungle”

Este resultado es la estructura de directorios y archivos necesarios para importar el proyecto en la herramienta Unity 3D. En este se pueden encontrar los diseños de todos los personajes y objetos, los scripts, el framework “Unity ETH UQ”, los sonidos que se utilizan en el juego y demás archivos que en su totalidad hacen el juego. Dicho proyecto se encuentra en un directorio llamado *Risky Jungle.*

### APK del juego

Este archivo es el resultado final de compilar, configurar y exportar el proyecto, el cual se puede instalar en dispositivos con sistema operativo Android para poder disfrutar de la experiencia de jugar Risky Jungle. Dicho archivo, con extención apk, se encuentra adjunto con el presente informe y se llama *Risky\_Jungle.apk.*

## Evaluar la funcionalidad de los componentes del framework mediante el juego desarrollado

### Informe de hallazgos y validación del framework

Este resultado es un informe en el cual se reporta dónde y cómo se implementó cada funcionalidad del framework y se brinda una conclusión de su funcionamiento, además a cada módulo se le hacen observaciones que identificó el equipo de trabajo en el transcurso del desarrollo del presente proyecto. Dicho documento se encuentra adjunto con el presente informe y su nombre es *Informe de hallazgos y validación del framework* ([Ver Informe de hallazgos y validación del framework](file:///C:\Users\Dival\Dropbox\Universidad\Trabajo%20de%20Grado\Informe%20final\Entrega%20CD\Documentos\Informe%20y%20manuales\Anexos\Informe%20de%20hallazgos%20y%20validación%20del%20framework.docx)).

# CONCLUSIONES Y TRABAJO FUTURO

* El framework "Unity ETH UQ" agiliza el proceso de soporte de idiomas y reproducción de audio en los componentes UI, lo cual disminuye significativamente el tiempo de desarrollo en proyectos que requieran dichas funcionalidades.
* Se verificó que la utilización de las analíticas del framework "Unity ETH UQ" permiten la recolección, análisis y predicción de datos, añadiendo un valor agregado a la aplicación.
* Se identificaron y documentaron las funcionalidades de los módulos audio, components, appSystem y util, lo cual facilita su utilización.
* Se generó un informe de validación de las funcionalidades de los módulos tatata, lo cual permite apreciar el estado actual del framework, dando la opción a Ethereal GF y al Grupo de Investigación en Redes, Información y Distribución - GRID realizar un proceso de mejora a su proyecto de investigación.
* Se documentó la forma en que se implementaron cada una de las funcionalidades, lo cual sirve como ejemplo para nuevos desarrolladores que utilicen el framework.

Como trabajo futuro se sugiere:

* Realizar la validación de los módulos del framework que no fueron utilizados en el presente proyecto para permitir un seguimiento más completo del framework.
* Realizar un proceso de mejora continua al módulo de analíticas del framework “Unity ETH UQ” enfocado a los desarrolladores de videojuegos para brindarles un mejor aprovechamiento de los datos de uso y marketing de sus aplicaciones.

# BIBLIOGRAFIA Y REFRENCIAS

Acerenza, N., Coppes, A., Mesa, G., Viera, A., Fernández, E., Laurenzo, T., & Vallespir, D. (2009). Una metodología para desarrollo de videojuegos. Retrieved June 1, 2016, from https://www.fing.edu.uy/sites/default/files/biblio/22811/asse\_2009\_16.pdf

Arce, L. J. (2011). Desarrollo de videojuegos, 1 – 164. Retrieved from http://bibliotecadigital.uda.edu.ar/objetos\_digitales/35/tesis-981-el.pdf

Benito García, J. M. (2006). El mercado del videojuego: Unas cifras. *Revista ICONO14. Revista Científica de Comunicación Y Tecnologías Emergentes*, *4*(1), 36–47. Retrieved from http://www.icono14.net/ojs/index.php/icono14/article/view/396

Chaparro García, B. (2015). *Ingeniería de software aplicada al desarollo de videojuegos*.

Equipo de investigación en el grupo de excelencia de Redes de Neuronas Artificiales y Sistemas Adaptativos (RNASA) de la Universidad de Coruña. (n.d.). Intro Unity. Retrieved from http://sabia.tic.udc.es/gc/Contenidos adicionales/trabajos/ProgramacionVideoJuegos/Unity3D/introunity.html

Gemserk. (2008). SUM. Retrieved February 15, 2016, from http://www.gemserk.com/sum/

Jústiz Nuñez, D., Gómez Suárez, D., & Delgado Dapena, M. (2014). Proceso de pruebas para productos de software en un laboratorio de calidad Testing process for software products at a quality.

Mocholí, A. (2014). Desarrollo de juegos con Unity 3D. Retrieved May 17, 2016, from https://www.yeeply.com/blog/desarrollo-de-juegos-con-unity-3d/

Morales Urrutia, G. A., Nava López, C. E., Fernández Martínez, L. F., & Rey Corral, M. A. (2010). Procesos de desarrollo para videojuegos. *CULCyT: Cultura Científica Y Tecnológica*, (36), 25–39. Retrieved from http://dialnet.unirioja.es/servlet/articulo?codigo=3238114&info=resumen&idioma=SPA

Myers, G. J., Sandler, C., & Badgett, T. (2011). *The Art of Software Testing*. Retrieved from https://books.google.es/books?hl=es&lr=&id=GjyEFPkMCwcC&oi=fnd&pg=PT5&dq=software+tetsing&ots=AgxNI3jW6n&sig=SjrY9jBpldZWqob8v4YHsCVOHXo#v=onepage&q=software tetsing&f=false

Polo Usaola, M. (2006). Pruebas del Software, *1*, 45.

Pressman, R. S. (2010). Ingeniería de software un enfoque práctico. Retrieved May 20, 2016, from http://187.216.127.158:8383/greenstone3/sites/localsite/collect/ciencia1/index/assoc/HASH015f/ceb375c1.dir/33040073.pdf

Ruiz Severiche, L. (2012). Que es microsoft word. Retrieved from http://es.slideshare.net/lele\_ruiz12/que-es-microsoft-word