Universidad Nacional de San Agustín de Arequipa
Facultad de Ingeniería de Producción y Servicios
Escuela Profesional de Ingeniería de Sistemas

Curso: Tecnología de Objetos
Docente: Ing. Richart Smith Escobedo Quispe

Memoria Descriptiva

Videojuego Educativo de Carreras en Unity

"Rápidos y Curiosos"

Integrantes del equipo:

- Choquehuanca Zapana Hernan Andy (Rama: hz07s/workspace)
- Larico Rodriguez Bryan Fernando (Rama: BryanLarico/workspace)
- Portugal Portugal Eduardo Sebastian (Rama: Eduardo-P/workspace)
- Mamani Anahua Victor Narciso (Rama: victorma18/workspace)

Arequipa, 7 de diciembre de 2024

Índice

1. Introducción

La presente memoria descriptiva documenta el diseño, desarrollo e implementación del videojuego educativo de carreras *"Rápidos y Curiosos"*, creado en el motor Unity como proyecto del curso Tecnología de Objetos. El sistema combina mecánicas de conducción en tiempo real con un componente de preguntas y respuestas que introduce contenidos educativos durante la experiencia de juego.

El jugador recorre un circuito y, en puntos clave de la pista, se activan preguntas con varias alternativas representadas por portales interactivos. Al atravesar el portal que corresponde a la respuesta correcta, el vehículo recibe un impulso temporal de velocidad; en caso de error, se aplica una penalización que reduce el desempeño por algunos segundos, reforzando así la retroalimentación inmediata y la atención sostenida del usuario.

1.1. Contexto y motivación

Los videojuegos educativos o *serious games* se han consolidado como herramientas de apoyo al proceso de enseñanza–aprendizaje, al integrar dinámicas lúdicas con objetivos formativos concretos. En este contexto, *"Rápidos y Curiosos"* busca aprovechar la familiaridad de los estudiantes con los juegos de carreras para reforzar contenidos académicos mediante la toma de decisiones rápidas.

El proyecto se plantea además como un ejercicio integral de aplicación de programación orientada a objetos, uso de patrones de diseño y trabajo colaborativo con control de versiones, alineado con los lineamientos del curso y la rúbrica de evaluación.

1.2. Problema y oportunidad

En entornos educativos tradicionales resulta difícil mantener la atención constante del estudiante durante actividades teóricas o evaluaciones repetitivas. Asimismo, muchos recursos digitales de evaluación carecen de elementos de motivación extrínseca más allá de la calificación numérica.

El videojuego propone una alternativa en la que el rendimiento académico (responder correctamente) se vincula de forma directa con una ventaja observable en el juego (mayor velocidad y mejor tiempo de carrera), generando una relación inmediata entre aprendizaje y desempeño lúdico.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar un videojuego educativo de carreras en Unity que integre un sistema de preguntas y respuestas a través de portales interactivos, promoviendo el aprendizaje mediante la toma de decisiones rápidas y la retroalimentación inmediata.

1.3.2. Objetivos específicos

- Implementar una mecánica básica de conducción (aceleración, frenado y giro) adecuada para un circuito de carreras educativo.
- Diseñar e integrar un sistema de preguntas con múltiples opciones representadas por portales en la pista.

- Asociar respuestas correctas e incorrectas con efectos de impulso y penalización de velocidad sobre el vehículo.
- Incorporar al menos tres patrones de diseño en la arquitectura del software (por ejemplo: Singleton, State y Observer).
- Proporcionar un modo de juego para uno o varios jugadores locales, con selección de vehículos y visualización de estadísticas.

1.4. Alcance y limitaciones

El alcance de la versión actual incluye:

- Una pista jugable con zonas de activación de preguntas.
- Sistema de preguntas de opción múltiple integrado con portales en el circuito.
- Modos de juego para un jugador y multijugador local en un mismo equipo.
- Selección de vehículos con diferentes atributos de manejo.
- Interfaz gráfica de usuario orientada a escritorio (Windows).

Entre las principales limitaciones se encuentran:

- El sistema de audio (música y efectos de sonido) no se completó según el cronograma inicial y se considera trabajo futuro.
- La gestión de bancos de preguntas se realiza dentro del proyecto (código o recursos de Unity) y aún no se expone un panel avanzado para docentes.
- El juego está pensado para PC con teclado o mando; no se contempla por ahora una versión móvil.

2. Marco teórico y tecnológico

2.1. Videojuegos educativos y toma de decisiones

El videojuego se enmarca dentro de los *juegos serios*, donde la dimensión lúdica se utiliza como medio para lograr objetivos educativos. En este caso, la mecánica de elegir portales bajo presión de tiempo refuerza habilidades de concentración, memoria y toma de decisiones, además de los contenidos específicos que se evalúan.

2.2. Motor Unity y programación orientada a objetos

Unity se utilizó como plataforma principal de desarrollo, empleando C# como lenguaje de programación orientado a objetos. Esta elección permite estructurar el código en clases y componentes reutilizables (controladores de vehículo, gestores de preguntas, controladores de escena, etc.), facilitando la extensión del proyecto en iteraciones futuras.

2.3. Patrones de diseño

Con el fin de mejorar la mantenibilidad y escalabilidad del sistema, se aplicaron diversos patrones de diseño en los componentes principales:

- **Singleton**: para gestores globales como el administrador de juego y el gestor de preguntas, garantizando una única instancia accesible desde distintas escenas.
- **State**: para modelar los estados del vehículo (normal, impulsado, penalizado), separando la lógica de comportamiento según el estado actual.
- **Observer**: para notificar a la interfaz de usuario y a los portales cuando se genera una nueva pregunta o se registra una respuesta.

3. Análisis y diseño del sistema

3.1. Requisitos funcionales

A nivel general, el sistema cumple con los siguientes requisitos funcionales principales:

- RF01: Permitir la conducción de un vehículo a través de un circuito de carreras.
- RF02: Mostrar preguntas educativas durante la carrera con varias opciones de respuesta.
- RF03: Asociar cada opción con un portal interactivo en la pista.
- RF04: Aplicar impulso de velocidad al vehículo cuando la respuesta es correcta.
- RF05: Aplicar penalización de velocidad cuando la respuesta es incorrecta.
- RF06: Ofrecer un menú principal con selección de modo de juego y de vehículo.
- RF07: Mostrar información relevante en pantalla (velocidad, vuelta, mensajes de acierto/error).

3.2. Requisitos no funcionales

Entre los requisitos no funcionales más relevantes se incluyen:

- RNF01: Interfaz intuitiva que permita aprender a jugar en pocos minutos.
- RNF02: Rendimiento adecuado para mantener una tasa de fotogramas estable en equipos de gama media.
- RNF03: Código organizado en clases y scripts coherentes, aplicando patrones de diseño y buenas prácticas.
- RNF04: Uso de herramientas de software libre y licencias adecuadas para los recursos empleados.

3.3. Arquitectura general

La arquitectura se organiza alrededor de escenas y prefabs de Unity, complementados con scripts C# que encapsulan la lógica de juego. A nivel alto, se distinguen los siguientes componentes:

- **Gestor de Juego (GameManager)**: coordina el flujo general (carga de escenas, inicio y fin de carrera).
- **Controlador de Vehículo**: maneja la entrada del usuario y traduce las acciones en movimiento del auto.

- **Gestor de Preguntas**: administra el banco de preguntas, selecciona ítems y evalúa respuestas.
- **Portales de Respuesta**: representan visualmente cada opción y aplican los efectos correspondientes.
- **UI Manager**: muestra textos, preguntas, opciones y mensajes de retroalimentación.

3.4. Escenas y estructura básica

El proyecto se compone de varias escenas principales:

- **Pantalla de inicio**: portada del juego y acceso al menú principal.
- **Menú principal**: selección de modo de juego, vehículos y opciones.
- **Escena de carrera**: circuito de juego con vehículos, portales y HUD.
- **Pantalla de resultados**: resumen de tiempo, aciertos y desempeño.

4. Implementación y funcionalidades

4.1. Mecánica de conducción

La conducción se basa en la lectura de entradas de teclado o mando (acelerar, frenar, girar) aplicadas sobre un controlador de física del vehículo. Se ajustan parámetros como aceleración máxima, fuerza de frenado y sensibilidad de giro para ofrecer una experiencia de manejo estable y apta para usuarios novatos.

4.2. Sistema de preguntas y portales

El sistema de preguntas utiliza una colección de ítems donde cada elemento incluye enunciado, alternativas y respuesta correcta. Al activarse una zona de pregunta, se muestra el enunciado en la interfaz y se habilitan portales asociados a cada alternativa; el vehículo atraviesa uno de ellos y la respuesta se evalúa de forma inmediata, modificando el estado del vehículo a "impulsado" o "penalizado".

4.3. Selección de vehículos y estadísticas

La pantalla de selección de vehículos permite al jugador comparar varios autos con atributos como velocidad máxima, aceleración, capacidad de giro y frenado. Estos atributos se reflejan directamente en el comportamiento del controlador del vehículo dentro de la pista, permitiendo que el usuario elija el estilo de manejo que mejor se adapta a su estrategia.

4.4. Manejo de errores y buenas prácticas

En el desarrollo se consideró el manejo básico de errores y validaciones, por ejemplo:

- Verificar que exista una pregunta válida antes de mostrarla.
- Evitar estados inconsistentes del vehículo (como aplicar impulso y penalización simultáneos).

- Registrar mensajes en la consola de Unity cuando se detectan configuraciones incompletas en la escena.

5. Proceso de instalación y ejecución

5.1. Instalación para desarrolladores

Para trabajar con el proyecto desde Unity:

1. Clonar el repositorio desde GitHub:
   - `https://github.com/rescobedoq/ryc`
2. Abrir la carpeta del proyecto en la versión recomendada de Unity (por ejemplo, una versión LTS 2022 compatible).
3. Permitir que Unity importe escenas, scripts y *assets* necesarios.
4. Verificar que la escena principal de la carrera esté configurada en *Build Settings* como escena de inicio.
5. Ejecutar el juego en modo *Play* para comprobar conducción, sistema de preguntas y HUD.

5.2. Instalación para usuarios finales

Para ejecutar únicamente el videojuego:

1. Descargar la versión compilada del juego (ejecutable y carpeta de datos) proporcionada por el equipo desarrollador.
2. Descomprimir el paquete en una carpeta local con permisos de lectura y escritura.
3. Ejecutar el archivo principal (por ejemplo, `RyC.exe`) desde el sistema operativo Windows.

6. Pruebas y validación

6.1. Estrategia de pruebas

Se realizaron principalmente pruebas manuales de:

- **Jugabilidad**: respuesta del vehículo a las entradas del usuario, dificultad del circuito y sensación de control.
- **Sistema de preguntas**: correcta aparición de preguntas, asociación con portales y aplicación de impulsos o penalizaciones.
- **Interfaz**: visibilidad de textos, mensajes de feedback y consistencia entre escenas.

6.2. Funcionalidades verificadas

Durante las pruebas se verificó que:

- El jugador puede iniciar una partida desde el menú principal, seleccionando vehículo y modo de juego.
- Las preguntas se muestran en los puntos previstos de la pista y los portales responden correctamente a la elección del jugador.
- El HUD refleja información relevante (velocidad, mensajes de acierto/error, etc.).

6.3. Trabajo futuro

Entre las mejoras propuestas para versiones posteriores se encuentran:

- Integrar música de fondo y efectos de sonido coherentes con la temática del juego.
- Ampliar el número de circuitos y bancos de preguntas.
- Incorporar un panel de administración para que los docentes gestionen preguntas desde una interfaz gráfica.
- Añadir tablas de puntuaciones y métricas de aprendizaje para seguimiento del progreso de los estudiantes.

7. Conclusiones

El videojuego educativo de carreras *"Rápidos y Curiosos"* demuestra que es posible integrar mecánicas de conducción arcade con un sistema de evaluación formativa basado en preguntas y respuestas de manera coherente y motivadora. La relación directa entre el acierto en las respuestas y la ventaja en la carrera genera un vínculo claro entre aprendizaje y rendimiento dentro del juego.

Desde el punto de vista técnico, el uso de Unity, C# y patrones de diseño como Singleton, State y Observer permitió construir una arquitectura modular y extensible, adecuada para incorporar nuevas funcionalidades en futuras iteraciones del proyecto. El trabajo colaborativo mediante GitHub y la documentación asociada (manual de usuario y memoria descriptiva) completan el ciclo de desarrollo esperado en el curso.