Informe Técnico de la Etapa Final del Proyecto

Universidad Nacional de Costa Rica (UNA)

Sede Regional Brunca Campus P.Z

Título del Proyecto: Costa Rican Enchanted Gems

Carrera:

Ingeniería en Sistemas

Profesor:

Rubén Mora Vargas

Estudiante:

Johan Serrano Víctor

Curso:

Programación 1

Fecha: 12 de noviembre 2025

1. Resumen

El presente informe técnico detalla el desarrollo del proyecto final titulado “Costa Rican Enchanted Gems”, un videojuego tipo Match-3 desarrollado en C++ utilizando la librería SFML 2.6.2. El proyecto fue diseñado como una propuesta de aplicación práctica de los conceptos fundamentales de programación, aplicando principios de programación orientada a objetos (POO), encapsulamiento, modularidad, manejo de memoria y diseño estructurado.

El videojuego recrea una experiencia de rompecabezas ambientada en un entorno tropical costarricense, donde el jugador debe combinar gemas del mismo tipo para completar objetivos específicos en diferentes niveles. A lo largo del desarrollo se implementaron sistemas de renderizado gráfico, detección de matches, animaciones, persistencia de datos y administración de recursos, todo bajo una arquitectura modular que facilita la escalabilidad y el mantenimiento del código.

2. Introducción

El proyecto surge como parte de la evaluación final del curso de Programación 1 y su propósito principal fue demostrar el dominio de los pilares fundamentales de la programación estructurada y orientada a objetos. Para ello, se utilizó el lenguaje C++ junto con la librería gráfica SFML (Simple and Fast Multimedia Library), lo que permitió implementar un entorno visual interactivo con sprites, texturas y eventos.

La finalidad del informe es describir de forma técnica cada uno de los componentes del sistema, incluyendo su diseño, estructura, funcionamiento interno y las relaciones entre clases, así como la lógica general del juego.

3. Arquitectura del Proyecto

La arquitectura del proyecto se basa en una estructura modular dividida en componentes principales, ubicados dentro de la carpeta Match3_Game_JohanSerrano/. A continuación, se muestra un esquema simplificado (NO DEFINITIVO) de la organización:

```
Match3_Game_JohanSerrano/
├──── Match3_Game_JohanSerrano/
|     ├──── saves/ (datos persistentes)
│     ├──── Game.cpp/.h
│     ├──── Board.cpp/.h
│     ├──── Player.cpp/.h
│     ├──── Gem.cpp/.h
│     ├──── Level.cpp/.h
│     ├──── LevelManager.cpp/.h
│     ├──── ResourceManager.cpp/.h
│     ├──── UIManager.cpp/.h
│     ├──── UXManager.cpp/.h
│     └──── main.cpp
├──── assets/ (sprites, fuentes, texturas)
└──── external/ (bibliotecas SFML)
```

4. Descripción de las Clases Principales

Game: Clase controladora del ciclo principal del juego, responsable de la inicialización de la ventana, la gestión de escenas y el control de flujo. Incluye métodos como “runGameLoop()”, “startLevel ()”, etc, además de un jugador por partida gestionando la persistencia de los datos del Player mediante el uso de la clase PersistenceManager.

Board: Maneja el tablero principal del juego, las posiciones de las gemas, la detección de combinaciones, la lógica de movimiento y las animaciones. Define el estado del tablero mediante enumeraciones y mantiene una matriz de objetos `Gem`.

Gem: Clase base de las gemas. Contiene propiedades como posición, tipo, sprite y estado. Se utiliza herencia para derivar tipos especiales como “BombGem”, “IceGem” y “NormalGem”. Cada subclase redefine el comportamiento de desaparición o efectos en cadena.

Level y LevelManager: Controlan los objetivos, condiciones de victoria y progresión del jugador. El archivo `levels.txt` define los parámetros de cada nivel, y el “LevelManager” administra la carga, validación y transición entre niveles.

ResourceManager: Gestiona la carga única de texturas, fuentes y sonidos mediante un patrón Singleton, evitando duplicación de recursos y mejorando la eficiencia del programa.

UIManager y UXManager: Controlan los elementos visuales (botones, textos, paneles) y la interacción del usuario, manteniendo la separación entre la lógica del juego y la interfaz gráfica.

5. Integración con SFML

SFML fue la librería gráfica central del proyecto. Se utilizó para crear la ventana de renderizado, cargar texturas, manejar eventos del teclado y mouse, y reproducir sonidos. La integración permitió desarrollar una interfaz fluida y animaciones coherentes con la estética retro del juego.

6. Persistencia de Datos

El proyecto implementa un sistema de persistencia de datos en la carpeta “ saves ”, utilizando archivos de texto para almacenar información del progreso del jugador. La clase “PersistenceManager” maneja las operaciones de lectura y escritura, garantizando que los datos puedan ser recuperados tras cerrar el programa.

7. Flujo Lógico del Juego

El flujo del juego inicia con la carga de recursos mediante “ResourceManager”, seguido de la pantalla principal controlada por “UXManager”. Al iniciar un nivel, la clase “Game” invoca al “Board” para inicializar la disposición de gemas. Durante la partida, los eventos del mouse son capturados, procesados y reflejados en el tablero.

Una vez detectada una combinación válida, el “Board” elimina las gemas correspondientes, actualiza la puntuación y genera nuevas piezas. Este ciclo continúa hasta que se cumple la condición de victoria definida por “LevelManager”.