Informe Desafío 2

Tomás Restrepo Saldarriaga, Daniel Velásquez Parra.

28 de octubre de 2025

1. Introducción

A continuación se describe el desarrollo del Desafío 2, el objetivo principal consiste en aplicar los conocimientos en programación orientada a objetos mediante la creación de una aplicación funcional en C++ y Qt. El proyecto, llamado UdeATunes, simula una plataforma de música digital que integra usuarios, artistas, álbumes, canciones, colaboradores y publicidad. Inicialmente se plantea una estructura general que nos permita tener un panorama adecuado para dar solución al problema. Esto se realiza a partir de clases, manejo de memoria dinámica y control de interacción mediante menús. El desarrollo permitió integrar los conceptos básicos del paradigma orientado a objetos con un enfoque modular y estructurado.

2. Objetivos

2.1. Objetivo general

Diseñar e implementar una aplicación en C++ con Qt que simule el servicio de streaming musical UdeATunes, permitiendo la gestión estructurada de usuarios (estándar y premium), artistas, álbumes, canciones y mensajes publicitarios. El sistema deberá integrar funcionalidades como la reproducción aleatoria de canciones, la administración de listas de favoritos, la inserción probabilística de anuncios según la categoría del usuario y la medición del consumo de recursos (iteraciones y memoria utilizada). Para ello, se aplicarán principios de programación orientada a objetos —abstracción, encapsulamiento, herencia y sobrecarga— junto con el diseño de estructuras de datos dinámicas propias que garanticen eficiencia, consistencia y escalabilidad en la ejecución del programa.

2.2. Objetivos específicos

- Implementar relaciones entre clases utilizando composición y agregación.
- Aplicar constructores, destructores y operadores de asignación para manejar memoria dinámica.
- Diseñar menús y funciones de interacción con el usuario.
- Cargar y guardar información desde archivos externos de texto.
- Simular la funcionalidad de un reproductor musical básico con control de reproducción y publicidad.

3. Planteamiento del problema

El desafío plantea la necesidad de desarrollar un sistema que simule el funcionamiento de una plataforma de música digital, donde los usuarios puedan registrarse, iniciar sesión, escuchar canciones, y dependiendo de su tipo de cuenta (estándar o premium), acceder a distintas funcionalidades. El sistema debe además gestionar información asociada a artistas, álbumes, canciones y créditos de colaboradores, garantizando la correcta representación de sus relaciones. Por tanto, el reto se centra principalmente en diseñar una arquitectura orientada a objetos que permita modelar esta plataforma de manera estructurada y eficiente, esto manteniendo un control riguroso sobre los recursos de memoria dinámica.

4. Diseño

La plataforma se estructuró pensando en que las diferentes clases se encuentren relacionadas entre sí, además de que permitan representar las entidades y funcionalidades principales mencionadas en el documento. Llevar un orden de esta manera facilita la comunicación entre los distintos módulos del sistema y garantiza que la arquitectura sea consistente. También, definir correctamente las clases y sus relaciones permite reutilizar código que puede ser útil lo cual reduce redundancias y mejora la estructura del proyecto. De esta manera, cada componente cumple un propósito específico dentro del proyecto, con esto, llegamos al cumplimiento de los objetivos funcionales de la plataforma manejando adecuadamente la información.

4.1. Clases principales

- **usuario:** Representa a los usuarios del sistema, con atributos como nombre, ciudad, país, fecha de registro y tipo de cuenta (premium o estándar). Puede almacenar canciones favoritas y seguir a otros usuarios.
- **artista:** Contiene la información de los artistas, incluyendo país, edad, número de seguidores, posición global y los álbumes asociados.
- **album:** Modela un álbum musical con su identificación, sello discográfico, fecha, puntuación y las canciones que lo componen.
- **cancion:** Representa una canción, con su duración, número de reproducciones, rutas de archivo y créditos de los colaboradores.
- **colaborador y creditos:** Permiten registrar a productores, músicos y compositores que participaron en la creación de una canción.
- **publicidad:** Gestiona los mensajes publicitarios que se muestran a los usuarios estándar durante la reproducción.
- **reproductor:** Controla la reproducción, pausa, volumen y anuncios publicitarios.
- **udeaTunes:** Actúa como clase principal del programa, integrando todas las funcionalidades, cargando los datos y gestionando los menús principales de interacción.

5. Implementación

Durante la implementación, se utilizó C++ como lenguaje principal, aplicando los principios de programación orientada a objetos. El proyecto se dividió en archivos de cabecera (`.h`) y de implementación (`.cpp`) para cada clase, facilitando la organización y mantenibilidad del código.

El sistema carga la información de usuarios, artistas, colaboradores y créditos desde archivos de texto externos. La memoria dinámica se gestiona cuidadosamente mediante constructores de copia, operadores de asignación y destructores definidos en cada clase.

La interfaz de usuario se maneja mediante menús de consola que permiten navegar por las diferentes opciones del sistema. Se implementó un reproductor que simula la reproducción de canciones, mostrando anuncios en los usuarios estándar y omitiéndolos en los usuarios premium.

El flujo principal está controlado por la clase *udeaTunes*, que coordina la carga de datos, el inicio de sesión y la interacción con los menús correspondientes.

6. Pruebas y resultados

Se realizaron diversas pruebas para validar el funcionamiento de la plataforma:

- Carga y lectura de los archivos de texto de entrada.
- Registro y almacenamiento de nuevos usuarios.
- Diferenciación de funcionalidades entre usuarios premium y estándar.
- Simulación de reproducción musical con control de volumen y modo aleatorio.
- Verificación de la correcta liberación de memoria dinámica al finalizar el programa.

Las pruebas son un componente fundamental a la hora de implementar un proyecto en el área de programación, puesto que allí debemos ser supremamente cuidadosos y estrictos. Realizar pruebas de manera adecuada y con antelación proyecta fiabilidad sobre quien presenta el proyecto. Además, nos aseguramos de que no se presente ningún tipo de vulnerabilidad y poder corregir a tiempo cualquier error que se pueda presentar. También, hay que ser conscientes de que corregir errores antes de entregar un proyecto en la mayoría de los casos es mucho más económico que hacerlo después.

7. Conclusiones

Al modelar una plataforma que simula un escenario real ponemos en práctica no solo las pautas académicas, sino que también nos acercamos a la realidad, desarrollar un proyecto de este tipo implica plantear la solución de un problema en distintas etapas y cumplir con el objetivo de manera estructurada no solo siendo ágiles sino eficientes.

En general, en este desafío aplicamos de manera extendida los conocimientos previos en el desafío anterior y los conceptos correspondientes a esta unidad, lo cual proporcionó dicha continuidad con los temas del curso, a su vez notamos la importancia de planificar arquitecturas orientadas a objetos antes de la implementación.

8. Referencias

- Notas de clase.