INFORME TÉCNICO: PROTOTIPO UDEATUNES

Manuel Felipe Salazar Burgos, Juan Pablo Rivera

Universidad de Antioquia

ABSTRACT

Este informe detalla el diseño y la implementación de UdeATunes, un prototipo en C++ de un servicio de streaming de música. El sistema gestiona una biblioteca musical, autenticación de usuarios, listas de favoritos personalizadas y un módulo de publicidad dinámica. Se enfoca en la aplicación de los principios de la Programación Orientada a Objetos (POO) y la gestión manual de memoria, excluyendo el uso de la STL para las estructuras de datos. Una característica clave es la instrumentación del código para medir métricas de rendimiento, como el consumo de memoria y las iteraciones computacionales, proporcionando un análisis de la eficiencia de la solución.

Index Terms— Programación Orientada a Objetos, C++, streaming de música, gestión de memoria, métricas de rendimiento, autenticación de usuarios, publicidad dinámica.

1. INTRODUCCIÓN

La industria musical ha evolucionado desde los formatos físicos hacia las plataformas de streaming, las cuales ofrecen acceso instantáneo a vastos catálogos de contenido. Este proyecto, **UdeATunes**, tiene como objetivo desarrollar un prototipo funcional en C++ que simula la lógica subyacente de un servicio de streaming musical. La aplicación se enfoca en la gestión de la biblioteca musical, el manejo de usuarios con sus listas de favoritos y la implementación de un sistema de reproducción con funcionalidades básicas, aplicando los principios de la **Programación Orientada a Objetos (POO)**.

Un requisito fundamental del desarrollo es la medición del rendimiento del sistema. Por ello, el programa está instrumentado para calcular y reportar métricas clave como el **consumo de memoria dinámica** y el **número de iteraciones** requeridas para operaciones críticas. Este informe presenta un análisis de la solución implementada, detallando la arquitectura del sistema y el diseño de sus componentes principales.

2. ANÁLISIS DEL PROBLEMA Y SOLUCIÓN PROPUESTA

El desafío principal consiste en construir un sistema de consola que gestione y reproduzca una colección de archivos de audio, operando bajo restricciones técnicas específicas, como la prohibición del uso de la biblioteca estándar de plantillas (STL) para las estructuras de datos. Esta versión introduce mejoras significativas en la gestión de usuarios y la experiencia de uso.

Los aspectos más importantes que se abordaron en la solución son:

- **Autenticación de Usuarios:** Se implementó un sistema de inicio de sesión que verifica las credenciales del usuario (nombre y contraseña) contra un archivo de sistema (`sudo.txt`). Esto permite diferenciar entre usuarios y cargar sus configuraciones personalizadas.
- **Gestión de la Biblioteca Musical:** El sistema mantiene la capacidad de cargar canciones desde un repositorio completo de directorios o desde listas de reproducción personales (`favoritas.txt`).
- **Manejo de Usuarios y Persistencia:** La clase `Usuario` ahora gestiona credenciales y una lista personal de favoritos. El sistema crea un directorio para cada usuario donde se almacena su archivo `favoritas.txt`, garantizando la persistencia de sus datos.
- **Sistema de Reproducción:** La lógica de reproducción incluye un modo secuencial y uno aleatorio, manteniendo un historial de las canciones reproducidas para la funcionalidad de ”pista anterior”.
- **Integración de Publicidad Dinámica:** El sistema integra un módulo de publicidad. Para los usuarios con suscripción estándar, se seleccionan y muestran anuncios de forma ponderada entre canciones.

3. DESCRIPCIÓN DE LAS ENTIDADES (CLASES)

El sistema UdeATunes se articula en torno a cuatro clases principales que interactúan para proporcionar la funcionalidad descrita.

3.1. `Cancion`

Representa una pista de audio individual. Almacena atributos esenciales como el **nombre**, **artista**, **álbum** y la **ruta del archivo**. Es la unidad fundamental del sistema.

3.2. `Album`

Actúa como el motor de reproducción y el contenedor principal de canciones. Aunque su nombre es ”Album”, esta clase es polivalente para gestionar cualquier colección de pistas. Sus responsabilidades incluyen almacenar un array de punteros a objetos `Cancion`, cargar las canciones y controlar la reproducción.

3.3. `Usuario`

Modela a un usuario del sistema. Contiene datos de autenticación y estado de suscripción. Es responsable de gestionar la lista de favoritos, permitiendo agregar o eliminar canciones por su ID y persistiendo los cambios en un archivo `favoritas.txt` dentro de la carpeta del usuario.

3.4. `Publicidad`

Representa un anuncio. Cada objeto tiene un **tipo**, **duración**, **mensaje** y **prioridad**. Incluye un método estático para leer y crear una lista de anuncios desde un archivo de texto, facilitando su gestión.

4. REFERENCES

[1] B. Stroustrup, *The C++ Programming Language*, 4th ed. Addison-Wesley, 2013.

[2] Universidad de Valladolid, ”Ficheros en C++”. [En línea]. Disponible: `https://www2.eii.uva.es/fund_inf/cpp/temas/10_ficheros/ficheros_cpp.html`

[3] GeeksforGeeks, ”Manejo de archivos a través de clases en C++”. [En línea]. Disponible: `https://www.geeksforgeeks.org/file-handling-c-classes/`

[4] GeeksforGeeks, ”Librería Chrono en C++”. [En línea]. Disponible: `https://www.geeksforgeeks.org/chrono-in-c/`