TECNOLÓGICO DE MONTERREY, CAMPUS MONTERREY

Proyecto Integrador

Por

Ernesto Andre González Castro, #A01741225

Programación orientada a objetos (Gpo 306)

Luis Alberto Maltos Ortega

Nidia Colmenero Lucio

14 de junio del 2023

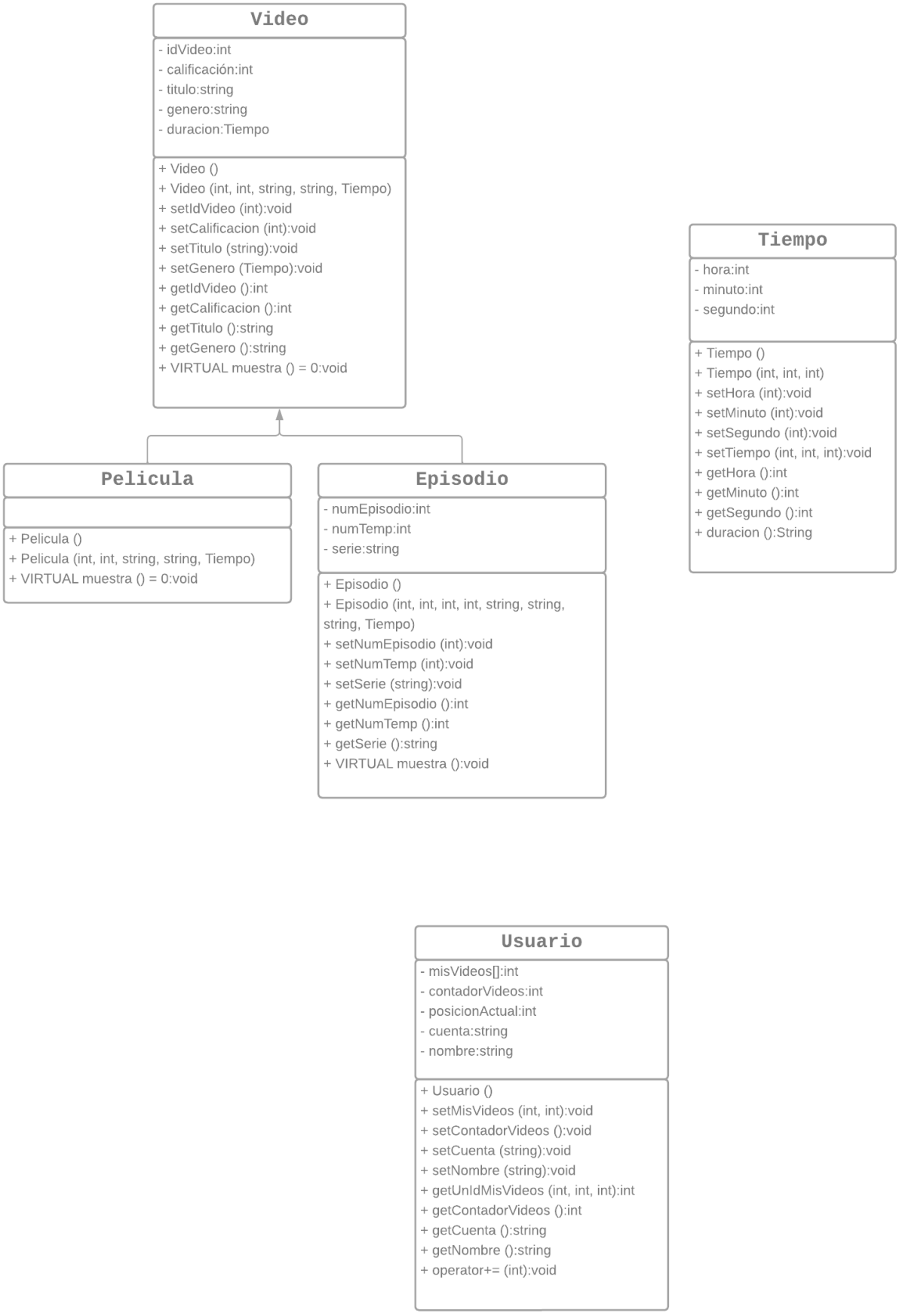
El presente documento tiene como objetivo abordar la situación problema del modelado de un servicio de “streaming” de videos bajo demanda. En este contexto, se plantea el desarrollo de un sistema que permita gestionar y mostrar diferentes tipos de videos, como películas y series, así como también realizar operaciones relacionadas con la calificación y visualización de estos.

Para alcanzar este objetivo, se utilizarán los principios de la Programación Orientada a Objetos, haciendo uso de conceptos fundamentales como la herencia, el polimorfismo y la sobrecarga de operadores. Estos conceptos permitirán diseñar y construir un sistema de clases que facilite la solución de la situación problema, al brindar una estructura organizada y reutilizable.

El sistema de clases propuesto constará de varias entidades principales, entre ellas la clase base "Video", que contendrá atributos generales como el ID, el nombre, la duración y el género. A partir de esta clase base, se derivarán las clases "Película" y "Serie", las cuales agregarán características específicas a cada tipo de video.

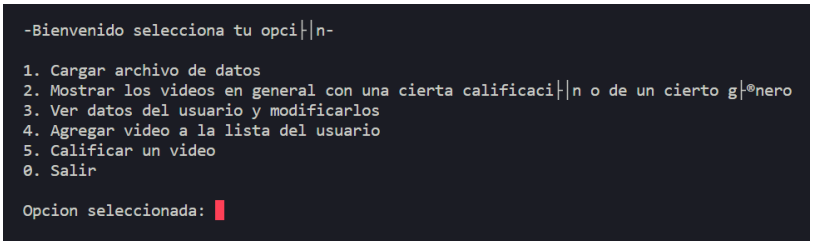
Además, se introducirá la clase "Episodio", que representará los episodios de una serie, y se implementará una interfaz o clase abstracta denominada "Calificable", que permitirá gestionar la calificación de los videos. Estas estructuras y relaciones entre las clases se plasmarán en un diagrama UML, que brindará una representación visual del diseño del sistema.

Una vez diseñado el sistema de clases, se procederá a construir un programa que utilice dichas clases para realizar diferentes tareas, como mostrar videos con sus respectivas calificaciones, mostrar episodios de una serie con sus calificaciones asociadas, y generar reportes según género y calificación de películas. El programa se encargará también de validar todas las entradas proporcionadas por el usuario, garantizando la integridad y consistencia de los datos.

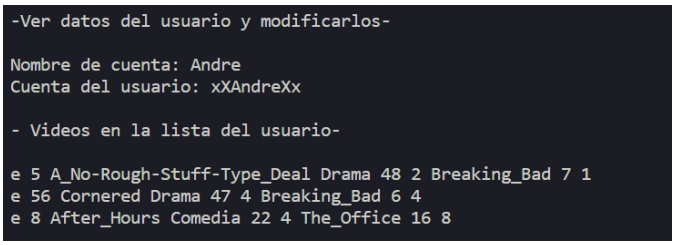

Video
- idVideo:int
- calificación:int
- titulo:string
- genero:string
- duracion:Tiempo
+ Video ()
+ Video (int, int, string, string, Tiempo)
+ setIdVideo (int):void
+ setCalificacion (int):void
+ setTitulo (string):void
+ setGenero (Tiempo):void
+ getIdVideo ():int
+ getCalificacion ():int
+ getTitulo ():string
+ getGenero ():string
+ VIRTUAL muestra () = 0:void
Tiempo
- hora:int
- minuto:int
- segundo:int
+ Tiempo ()
+ Tiempo (int, int, int)
+ setHora (int):void
+ setMinuto (int):void
+ setSegundo (int):void
+ setTiempo (int, int, int):void
+ getHora ():int
+ getMinuto ():int
+ getSegundo ():int
+ duracion ():String
Pelicula
+ Pelicula ()
+ Pelicula (int, int, string, string, Tiempo)
+ VIRTUAL muestra () = 0:void
Episodio
- numEpisodio:int
- numTemp:int
- serie:string
+ Episodio ()
+ Episodio (int, int, int, int, string, string, string, Tiempo)
+ setNumEpisodio (int):void
+ setNumTemp (int):void
+ setSerie (string):void
+ getNumEpisodio ():int
+ getNumTemp ():int
+ getSerie ():string
+ VIRTUAL muestra ():void
Usuario
- misVideos[]:int
- contadorVideos:int
- posicionActual:int
- cuenta:string
- nombre:string
+ Usuario ()
+ setMisVideos (int, int):void
+ setContadorVideos ():void
+ setCuenta (string):void
+ setNombre (string):void
+ getUnIdMisVideos (int, int, int):int
+ getContadorVideos ():int
+ getCuenta ():string
+ getNombre ():string
+ operator+= (int):void

El proceso de ideación para la creación de este diagrama UML se basó en tres populares fundamentales, flexibilidad, optimización y legibilidad. El primer pilar, flexibilidad, se basa en la creación de la clase Tiempo. Esto con el fin de poner adecuar la duración exacta de todo contenido audiovisual y darle exactitud a la plataforma. El siguiente, optimización, esta predefinido por las clases ya existentes en el PDF, con las cuales, se pude lograr acoplar todos los métodos necesarios para la creación del servicio de streaming. El ultimo, pero más importante, legibilidad, el cual permite la comprensión general de las personas para el funcionamiento del "main", esto gracias a al nombramiento exacto de métodos.

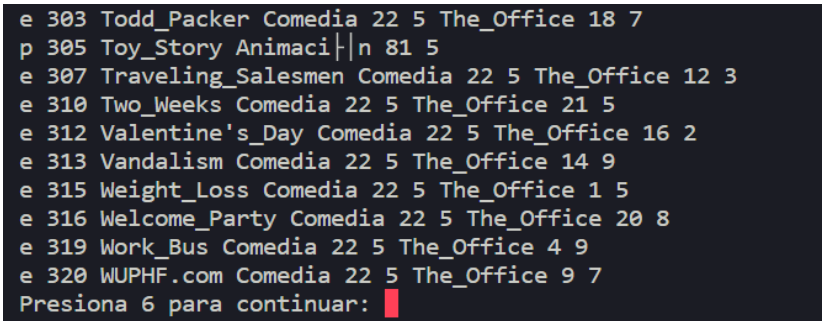
-Ejemplo de ejecución-



- Ver datos del usuario y modificarlos.



- Mostrar videos con cierta calificación.



Argumentación de las partes del proyecto

a) Utilización de clases: Se basa en 3 clases “claves”, “película”, “episodio” y “usuario”. Con una clase base en común de “película” y “episodio”, y una clase extra para flexibilidad la cual es “tiempo”.
b) Conceptos de herencia: Se ahorra la escritura de variables y métodos por parte de “película” y “episodio”. Esto gracias a la clase padre “video”.
c) Modificadores de acceso: Modificadores de acceso: Conforma tanto “getters” como “setters” para acceder a las variables durante el código.
d) Sobrecarga y sobreescritura: Esto se realizó con el método “muestra” el cual se encuentra en las dos clases hijas, pero tiene un desarrollo diferente en la clase “video” y en la clase “película”.
e) Conceptos de polimorfismo: Durante la implementación del método muestra, este cambia dependiendo de la clase que lo mande llamar.
f) Clases abstractas: La clase video cuenta con un método virtual puro, siendo incapaz de instanciarse siguiendo los lineamientos de clases abstractas.
g) Sobrecarga de operadores: En la clase usuario se presenta la sobrecarga de operadores para el añadido de videos a la lista, siendo este el =+.

Argumentación de la solución

Se opto por esta solución porque es la que puede utilizar todos los temas vistos durante el curso. De la misma forma, optimiza de mejor manera posible el código para que sea capaz de realizar tu trabajo en la manera más corta y legible de líneas. Además, involucra un sistema switch lo que hace mucho más fácil todo. En otras palabras, simplemente es un programa en el

que tienes un menú base, luego seleccionas opciones y al final regresa a ese menú base, haciendo muy comprensible todo el funcionamiento interno.

Otro tipo de soluciones involucraba el repetir continuamente líneas de código o el realizar aún más clases para utilizar más métodos. Por mi parte, considero que otro tipo de soluciones hubieran abarcado trabajo innecesario y hubieran consumido más tiempo y recursos para llevarse a cabo.

Identificación de casos

Debido al buen uso de técnicas de programación. Los únicos casos en los que el programa podría llegar a fallar son introducción de valores “string” y “boolean” en los inputs del programa, y en el caso de errores en los archivos. En este caso destaco el caso en el que el archivo TXT no se encuentro, o se encuentre modificado. Igualmente, casos en los que otros archivos principales falten como CPP o .H, o su falta de #include en el “main”. Concluyendo que de la misma forma que el primer caso, si algún archivo importante llegará a ser eliminado o modificado todo el programa llegaría a fallar.

Conclusión

Para finalizar, el proyecto abordado, que consistió en el modelado y desarrollo de un sistema de streaming de videos bajo demanda, ha sido una experiencia sumamente enriquecedora desde el punto de vista de la programación orientada a objetos.

A lo largo del proceso de diseño y desarrollo, se han aplicado conceptos fundamentales como la herencia, el polimorfismo y la sobrecarga de operadores para construir un sistema de clases robusto y modular. Estos conceptos han permitido organizar eficientemente las entidades y los comportamientos relacionados con los videos, fomentando la reutilización de código y la flexibilidad del sistema.

El uso de un diagrama UML ha sido de gran utilidad para visualizar y comprender la estructura del sistema, así como las relaciones existentes entre las diferentes clases. Esto ha sido especialmente beneficioso al implementar la aplicación, asegurando la coherencia con el diseño propuesto y la correcta implementación de cada funcionalidad.

A lo largo del proyecto, se ha podido apreciar la importancia de validar los datos y garantizar la integridad de la información proporcionada por el usuario. La implementación de estas verificaciones ha sido fundamental para asegurar un funcionamiento adecuado y prevenir posibles errores o inconsistencias en el sistema.

El proyecto de modelado y desarrollo del sistema de streaming de videos bajo demanda ha sido una experiencia valiosa que ha permitido aplicar conocimientos y habilidades en programación orientada a objetos. A través de este proyecto, se ha adquirido una mayor confianza en el diseño de sistemas complejos y en la implementación de aplicaciones funcionales y eficientes.

Referencias:

- **Stack Overflow - Where Developers Learn, Share, & Build Careers. (s. f.). Stack Overflow. https://stackoverflow.com/**