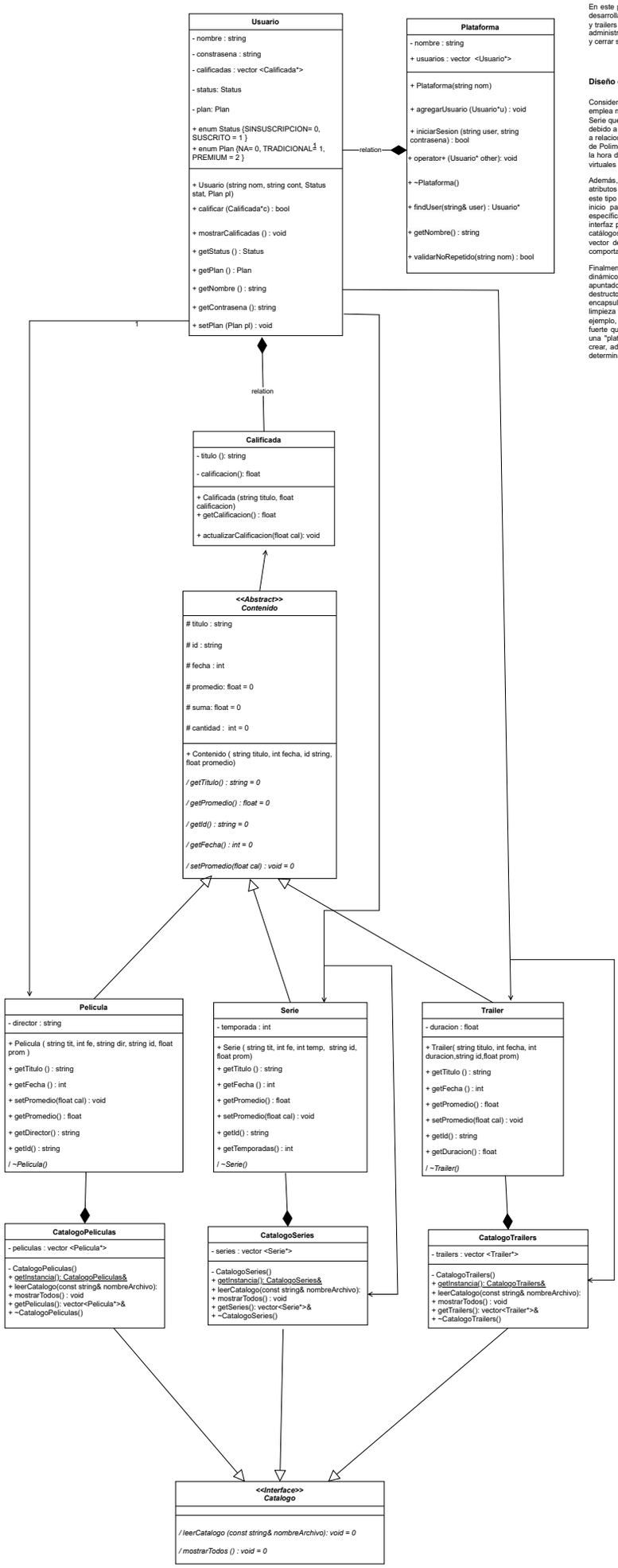
Proyecto integrador

Hecho por: Azul Paniagua Bucio (A01424351) y Alejandra Galván Bojórquez (A01424388)



Planteamiento del problema

En este proyecto, decidimos enfocarnos en la plataforma de streaming de Netflix para la cual desarrollamos un sistema de calificaciones de diversos contenidos tales como películas, series y trailers siguiendo un diseño orientado a objetos. Adicionalmente, la plataforma es posible de administrar usuarios, es decir, crear cuentas, hacer inicio de sesión, renovar el plan del usuario y cerrar sesión.

Diseño orientado a objetos

Consideramos que el diagrama de clases cumple con un diseño orientado objetos, pues emplea mecanismos de Herencia como, por ejemplo, en el caso de la clase Película, Trailer y Serie que heredan de la clase Contenido. La razón por la que se manejó de esta manera fue debido a que identificamos atributos y métodos en común entre estas clases, lo que nos lleva a relacionarlas por medio de una herencia. Asimismo, nuestro diseño se apoya del concepto de Polimorfismo al incluir métodos que posteriormente pueden ser definidos o sobrescritos a la hora de implementarlos según su contexto, por ejemplo, esto sucede con ciertos métodos virtuales tales como getId o mostrarTodos.

Además, decidimos que la clase Contenido fuera abstracta porque nos permite declarar atributos y métodos específicos que después pueden ser sobrescritos y sobre todo porque este tipo de clases funcionan como clase base para otras clases, es decir, puede ser un buen inicio para definir atributos y funcionalidades en común que se derivan a clases más específicas como lo son Película, Serie y Trailer. Por otra parte, decidimos implementar un interfaz para la clase Catalogo porque a diferencia de la clase Contenido, en este caso los catálogos no cuentan con atributos en común, es decir, cada catalogo cuenta con su propio vector de series, películas o trailers según sea el caso, pero sí cuenta con métodos o comportamientos comunes que se pueden implementar en clases más específicas.

Finalmente, el uso de destructores fue necesario para manejar la liberación de recursos dinámicos asignados al objeto, pues durante el desarrollo se utilizaron vectores de apuntadores para el contenido y los usuarios de la plataforma, así mismo el uso de destructores promueve otro concepto importante del diseño orientado a objetos que es la encapsulamiento al ocultar la implementación de la liberación de dichos recursos y la limpieza final al usuario de la clase. También, se presentaron relaciones de composición, por ejemplo, entre la clase Plataforma y Usuario, ya que consideramos que hay una asociación fuerte que se puede traducir en palabras más sencillas como que el "usuario" depende de una "plataforma" para existir, así como también es la plataforma quien tiene el control de crear, administrar, acceder, etc. a los usuarios, es decir, es la plataforma quien contiene y determina el ciclo de vida del usuario.