Tecnológico de Monterrey

Campus Santa Fe

# Situación Problema:

# Proyecto Integrador

TC1030

(Grupo 201)

Prof. Jesús Leopoldo Llano García

**Pablo Banzo Prida**

A01782031

Fecha de entrega: 8 de mayo de 2022

# Índice de contenido

# Introducción

Los servicios de streaming de video similares a Netflix® y Disney+® han estado en auge durante los últimos años. Un futuro proveedor de este tipo de servicios solicita apoyo para generar una primera versión del producto. Las especificaciones requeridas son:

Se quiere trabajar con dos tipos de videos: películas y series. Todo video tiene un ID, un nombre, una duración y un género (drama, acción, misterio).

Las series tienen episodios y cada episodio tiene un título y temporada a la que pertenece.

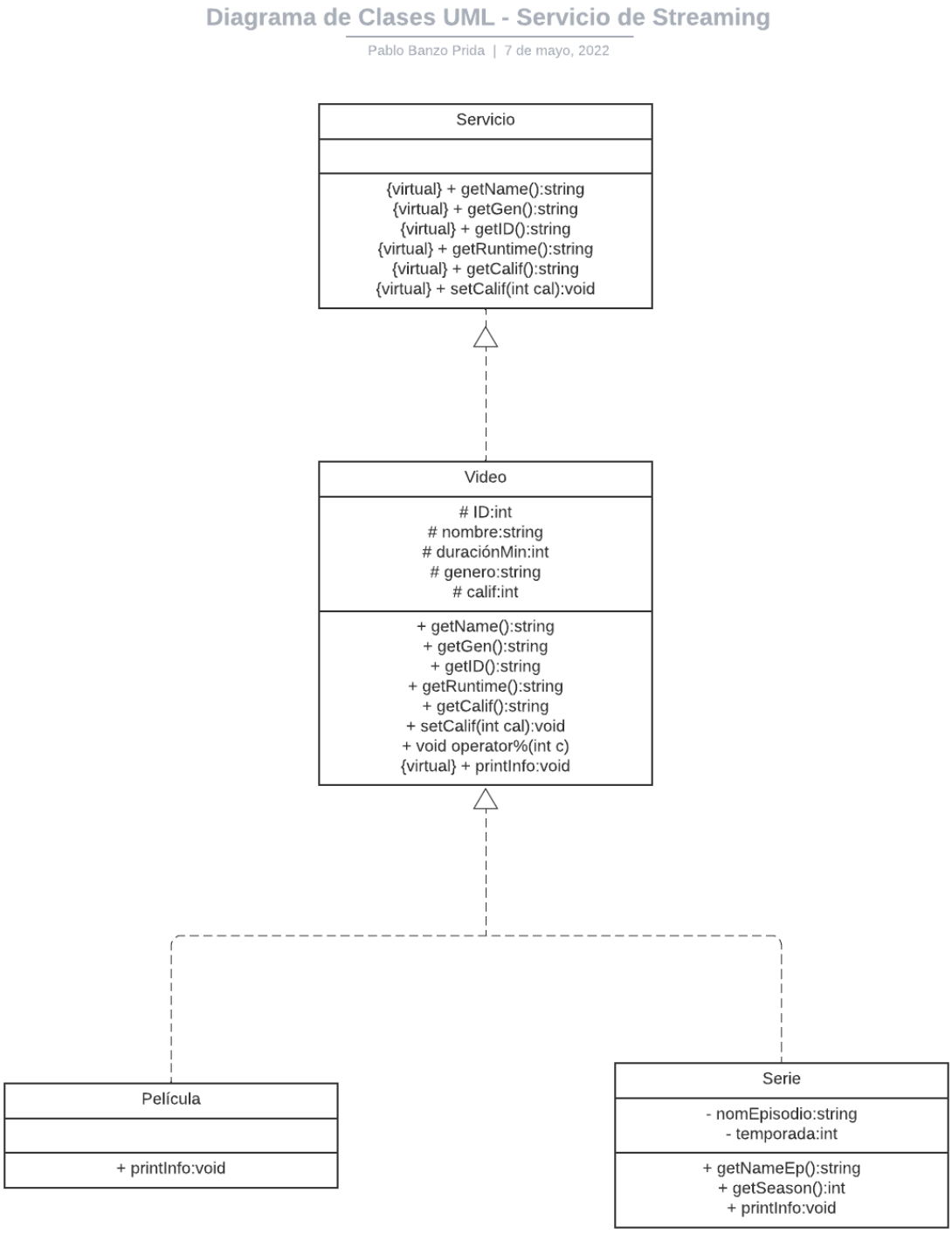
Nos interesa conocer la calificación promedio que ha recibido cada uno de los videos. Esta calificación está en escala de 1 a 5 donde 5 es la mejor calificación.

El sistema debe ser capaz de:
- Mostrar los videos en general con sus calificaciones
- Mostrar los episodios de una determinada serie con sus calificaciones
- Mostrar las películas con sus calificaciones

Se pide generar el diagrama de clases para la solución al problema, utilizando POO, herencia, polimorfismo y sobrecarga de operadores. Asimismo, se solicita crear una aplicación que tome información sobre los diferentes tipos de video y genere los siguientes reportes:

- Películas de un cierto género,
- Series de un cierto género,
- Películas con su calificación.

# Diagrama de clases UML



### Argumentación del diseño: ¿por qué este sistema de clases?

El sistema de clases empleado permite el acceso a las instancias de las clases a partir de ciclos "for" dentro del main(). Cabe agregar, que la jerarquía de herencias fue implementada de esta manera por dos razones principales:

1. El programa se reduce a dos tipos de objeto correspondientes a los tipos de videos existentes en el servicio.
2. Se evita la creación de instancias de tipos no existentes en el servicio mediante la abstracción de las clases, sin perder la simplificación del código que permite la herencia.

Se optó por generar una interfaz previa a la clase Video para definir su funcionalidad previamente sin dejar puerta a ambigüedades. Por otro lado, recordando que la película y la serie comparten la gran mayoría de sus atributos (salvo temporada y nombre de episodio), fue lógico que ambas fueran clases heredadas de una clase padre en común.

Cabe agregar que se decidió no crear una clase "Serie" intermedia entre "Video" y "Episodio" por simplicidad, dado que no se vio necesario que cada episodio tuviera un objeto interno para contener únicamente dos atributos (y métodos) adicionales.

## Ejemplo de ejecución

Menú principal

```
Eliga una de las siguientes opciones:
1. Cargar archivo de datos
2. Mostrar los videos en general con una cierta calificación o de un cierto género
3. Mostrar los episodios de una determinada serie con una calificacion determinada
4. Mostrar las películas con cierta calificacion
5. Calificar un video
0. Salir
```

Submenú opción 2.

```
Eliga una de las siguientes opciones:
1. Mostrar videos en general con una cierta calificación
2. Mostrar videos en general de un cierto género
0. Menú anterior
```

```
Ha elegido:
1. Mostrar videos en general con una cierta calificación
----------------------------
Ingrese una calificación del 1 al 5
5
La película Yo No Fui tiene calificación 5
El episodio Piloto 2 de la temporada 5 de la serie Drop Dead Diva tiene calificación 5
```

```
Ha elegido:
2. Mostrar videos en general de un cierto género
----------------------------
Ingrese un género
Drama
La película Mi Boda es de género Drama
La serie Downton es de género Drama
```

Opción 3.

```
Ha elegido:
3. Mostrar los episodios de una determinada serie con una calificacion determinada
-------------------------
Ingrese una serie
Downton
Ingrese una calificación del 1 al 5
5
El episodio Piloto de la serie Downton tiene calificación 5
```

Opción 4.

```
Ha elegido:
4. Mostrar las películas con cierta calificacion
----------------------------
Ingrese una calificación del 1 al 5
5
La película Yo No Fui tiene calificación 5
```

Submenú Opción 5.

```
Eliga una de las siguientes opciones:
1. Calificar una película
2. Calificar un episodio de una serie
0. Menú anterior
```

```
Ha elegido calificar una película
Eliga el título a calificar
Mi Boda
Ingrese una calificación del 1 al 5
3
La calificacion de la película Mi Boda ahora es 3
```

```
Ha elegido calificar un episodio de una serie
Eliga el nombre de la serie
Drop Dead Diva
Eliga el episodio a calificar
Piloto 2
Ingrese una calificación del 1 al 5
1
La calificacion del episodio Piloto 2 ahora es 1
```

Opción 0:

```
Ha elegido salir, gracias por usar nuestro servicio.
pridapablo@Bletzs-MacBook-Air src %
```

Opción Inválida:

```
Opción inválida... Hasta luego
pridapablo@Bletzs-MacBook-Air src %
```

## Relación Proyecto - Criterios

a) (10 pts) Se identifican de manera correcta las clases a utilizar
Como se muestra en el diagrama UML presentado con anterioridad, las clases necesarias para la construcción del sistema fueron 4, 2 abstractas y 2 concretas. Las clases que permiten el funcionamiento del programa son la clase Película y la clase Serie, dado que son los objetos solicitados por el futuro proveedor del servicio de streaming.

b) (12 pts) Se emplea de manera correcta el concepto de Herencia
La interfaz "Servicio" hereda sus métodos a la clase "Video" y esta segunda clase hace herencia simple de sus miembros tanto para la clase "Película" como para la clase "Serie". Esta estructura de herencia fue la más lógica y sencilla para dar solución al problema planteado, evitando así la reescritura de miembros mediante la herencia.

c) (10 pts) Se emplea de manera correcta los modificadores de acceso
Los atributos de la clase Video son protegidos para que únicamente sea posible acceder a ellos mediante instancias de las clases derivadas de esta. Por otro lado, los setters y los getters son públicos por convención para permitir el acceso a sus atributos correspondientes de forma más segura. En suma, se puede decir que se emplearon los modificadores de acceso de manera correcta para asegurar que únicamente se modifiquen los atributos correctos de los objetos sin afectar a otros miembros de la clase.

d) (12 pts) Se emplea de manera correcta la sobrecarga y sobreescritura de métodos.
Los métodos printInfo() de los Videos están sobrecargados para las clases Película y Serie para imprimir adecuadamente la información de los videos según sean de clase Película o de clase Serie. Esto fue necesario dado que la clase Serie contiene más atributos que la

clase Película, por lo que se tuvo que ampliar la funcionalidad. La clase película tiene este método declarado como virtual para evitar que se generen instancias de este tipo.

e) (12 pts) Se emplea de manera correcta el concepto de Polimorfismo
El comportamiento explicado en el inciso anterior es Polimórfico dado que cambia su comportamiento según la clase de la cual se llame, a pesar de que el método tenga el mismo nombre. Se optó por este comportamiento polimórfico para evitar la creación de métodos con comportamientos equivalentes con diferentes nombres, simplificando así la implementación del menú.

f) (12 pts) Se emplea de manera correcta el concepto de Clases Abstractas
Existen dos clases abstractas en el programa: la interfaz Servicio y la clase Video. La primera es una clase abstracta pura, mientras que la segunda solamente contiene un método virtual. Esta estructura de clases evita la creación de objetos de tipos no funcionales para el programa (mediante la definición abstracta de estas clases).

g) (12 pts) Se sobrecarga al menos un operador en el conjunto de clases propuestas
Se sobrecargó el operador "%" en la clase Video para llamar al setter de calificación de manera más sencilla. Cabe mencionar que este operador se definió a partir del setter encapsulado, para permitir la llamada al método mediante ambas firmas.

h) (opcional) Se utiliza de manera correcta el uso de excepciones tanto predefinidas como definidas por el programador
Se optó por no implementar el uso de excepciones, sin embargo, se comprueba la gran mayoría de las entradas por teclado para asegurar un comportamiento adecuado del sistema.

## Casos Extremos

Existen ciertos casos que provocarían que el sistema dejara de funcionar o bien, que no funcione de la forma adecuada. La principal área de oportunidad es la validación de las calificaciones recibidas por teclado, dado que el input puede ser erróneo y provocar funcionamientos defectuosos. Un ejemplo concreto sería que, en la opción 5, cuando el sistema solicite una calificación del 1 al 5, el usuario ingrese un promedio de 10, almacenando datos incorrectos en los atributos de esa instancia. Un segundo ejemplo sería que en la misma opción se ingrese una calificación promedio que no sea un número entero, provocando así un ciclo infinito del menú.

## Conclusión personal

Se puede concluir que el ejercicio de crear la primera versión del servicio mediante el paradigma orientado a objetos de la programación me ayudó mucho a identificar como implementar soluciones a problemas concretos con apoyo de clases, herencia, polimorfismo y sobrecarga de operadores. El reto principal fue implementar el menú, dado que la creación del sistema de clases fue bastante lógico tras identificar las necesidades del futuro proveedor de este servicio.

## Referencias

No se consultaron recursos externos a la clase.