DISEÑO DE SOFTWARE

PRÁCTICA 2

DANIEL YESTE LÓPEZ — 16884276
CHRISTIAN GUERRERO GÓMEZ — 16847191
RODRIGO CABEZAS QUIRÓS — 20100426

29 DE NOVIEMBRE DE 2018

Índice

1. Objetivos

La finalidad de esta práctica es realizar el diseño de la aplicación STUB, SeriesTimeUB, mediante la metodología Test Driven Development. Con esta metodología y partiendo desde el modelo de dominio, se implementan las clases necesarias aplicando los principios SOLID y patrones de diseño.

En esta entrega se implementarán las siguientes funcionalidades:

1. Registro de un nuevo cliente.
2. Login de un nuevo cliente.
3. Visualización del catálogo de series.
4. Suscripción a un episodio.
5. Visualización de un episodio.
6. Valoración de un episodio.
7. Visualización de la lista Watched List (series vistas y acabadas).
8. Visualización de la lista Watch Next (series con episodios no vistos).
9. Visualización de la lista Not Started Yet (series donde ya se han visto todos los episodios estrenados pero que todavía faltan por estrenar).

En esta práctica también se utilizará una capa de recursos, que se encargará de los datos del sistema. Cuando la aplicación se abre, carga y guarda todos los datos que le proporciona la capa de datos.

2. Diagramas

2.1. Historias de usuario:

ID Historia	Alta nuevo cliente	Nº Escenario	1
Rol	Como usuario no registrado.		
Funcionalidad	Quiero darme de alta a STUB.		
Objetivo	Para poder ver series.		
Criterios de aceptación			
Título	T1.1 - Dar alta usuario no registrado		
Contexto	En el caso que un usuario no registrado quiera darse de alta con un nombre y contraseña correctos.		
Evento	Cuando se piden los datos de usuario para dar el alta.		
Resultado	El usuario quedará registrado.		
Título	T1.2 - Dar alta usuario registrado		
Contexto	En el caso que un usuario ya registrado quiera darse de alta otra vez.		
Evento	Cuando se piden los datos de usuario para dar el alta		
Resultado	Se mostrará un error y se volverá al menú principal.		
Título	T1.3 – Nick ya cogido		
Contexto	En el caso que un usuario no registrado quiera darse de alta con un nombre ya usado.		
Evento	Cuando se piden los datos de usuario para dar el alta.		
Resultado	Se mostrará un mensaje de nick ya cogido y se pedirá que se escoja otro.		

ID Historia	Login nuevo cliente	Nº Escenario	2
Rol	Como usuario registrado.		
Funcionalidad	Quiero hacer el login a STUB.		
Objetivo	Para acceder al contenido.		

Criterios de aceptación	
Título	T2.1 y T2.2 - Login de usuario incorrecto
Contexto	En el caso que un usuario no introduzca correctamente el nick o la contraseña.
Evento	Cuando se hace el login.
Resultado	Se mostrará un mensaje y se pedirá que se vuelvan a introducir los datos.

ID Historia	Visualizar catalogo series	Nº Escenario	3
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero acceder al catálogo de series de la aplicación.		
Objetivo	Para ver las series disponibles.		
Criterios de aceptación			
Título	T3.1 – Ver catálogo		
Contexto	En el caso que un usuario escoja ver el catálogo de series.		
Evento	Cuando el usuario seleccione ver el catálogo de series.		
Resultado	Se mostrará un catálogo no vacío.		
Título	T3.2 – Ver catálogo cuando no hay series		
Contexto	En el caso que no haya series en el catálogo.		
Evento	Cuando el usuario seleccione ver el catálogo de series.		
Resultado	Se mostrará un catálogo vacío.		

ID Historia	Suscribirse a episodio	Nº Escenario	4
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero suscribirme a un episodio.		
Objetivo	Para no perderme ningún episodio.		
Criterios de aceptación			
Título	T4.1 – Suscripción a episodio		
Contexto	En el caso que un usuario quiera suscribirse a un episodio en el que no se ha suscrito aún y haya visto el episodio.		
Evento	Cuando se muestre el listado de episodios de una serie.		

Resultado	El usuario quedará suscrito al episodio elegido.
Título	T4.2 – Repetir suscripción a episodio
Contexto	En el caso que un usuario quiera suscribirse a un episodio en el que ya se ha suscrito anteriormente.
Evento	Cuando se muestre el listado de episodios de una serie.
Resultado	No se realizará la suscripción.

ID Historia	Ver episodio	Nº Escenario	5
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero ver un episodio.		
Objetivo	Para entretenerme.		
Criterios de aceptación			
Título	T5.1 – Reproducir episodio		
Contexto	En el caso que un usuario escoja un episodio disponible para verlo.		
Evento	Cuando se muestre el listado de episodios de una serie.		
Resultado	El sistema redirige al usuario al enlace del episodio.		

ID Historia	Valorar episodio	Nº Escenario	6
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero valorar un episodio.		
Objetivo	Para expresar mi opinión sobre el episodio.		
Criterios de aceptación			
Título	T6.1 – Valorar episodio que no existe		
Contexto	En el caso que un usuario quiera hacer la valoración de la emoción de un episodio.		
Evento	Cuando se muestre el listado de episodios de una serie.		
Resultado	No se podrá hacer la valoración.		
Título	T6.2 – Valorar episodio visto		
Contexto	En el caso que un usuario quiera hacer la valoración de la emoción de un episodio.		
Evento	Cuando se muestre el listado de episodios de una serie.		

Resultado	Se añadirá la valoración hecha por este usuario
Título	T6.3 – Valorar un episodio con una puntuación no permitida
Contexto	En el caso que un usuario quiera hacer la valoración de la emoción de un episodio.
Evento	Cuando se muestre el listado de episodios de una serie..
Resultado	No se hará la valoración.

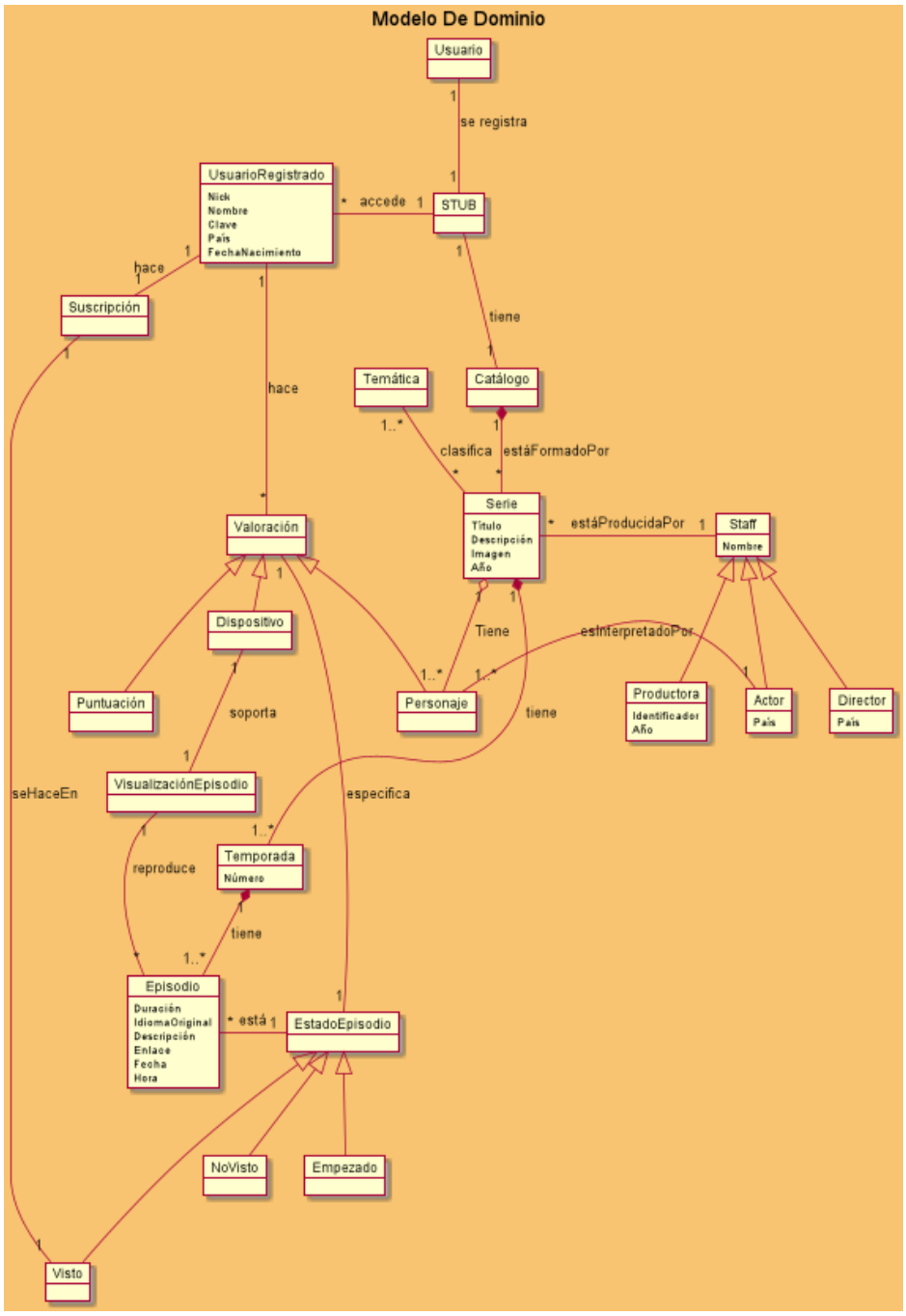
ID Historia	Ver Watched List	Nº Escenario	7
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero ver la lista de series WatchedList.		
Objetivo	Para así saber que series he visto ya.		
Criterios de aceptación			
Título	T7.1 – WatchedList vacía		
Contexto	En el caso de que no haya ninguna serie en la lista.		
Evento	Cuando vayamos a ver la lista.		
Resultado	Aparecerá un mensaje diciendo "Parece que no has visto ninguna serie".		

ID Historia	Ver Watch Next	Nº Escenario	8
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero ver la lista Watch Next.		
Objetivo	Para así saber que series tengo incompletas.		
Criterios de aceptación			
Título	T8.1 – Ningún episodio pendiente		
Contexto	En el caso de que no tenga ningún episodio por ver en las series suscritas.		
Evento	Cuando se vaya a mostrar la lista.		
Resultado	Saldrá un mensaje diciendo "No tienes ningún episodio pendiente".		
Título	T8.2 – Ninguna serie empezada		
Contexto	En caso de que no haya empezado ninguna serie.		

Evento	Cuando se vaya a mostrar la lista.
Resultado	Saldrá un mensaje diciendo "No has comenzado ninguna serie".

ID Historia	Ver Not Started Yet	Nº Escenario	9
Rol	Como usuario registrado y logueado.		
Funcionalidad	Quiero ver la lista NotStartedYet.		
Objetivo	Para ver que series que he visto sacarán nuevo contenido.		
Criterios de aceptación			
Título	T9.1 – No hay series acabadas		
Contexto	En caso de que no haya series acabadas.		
Evento	Cuando se vaya a mostrar la lista.		
Resultado	Se mostrará un mensaje diciendo "No hay series acabadas".		
Título	T9.2 – Episodio para estrenar en las suscripciones		
Contexto	En caso de que haya un episodio nuevo próximamente.		
Evento	Cuando vamos a seleccionar "Ver Not StartedYet List".		
Resultado	Se mostrará un texto anunciando el nuevo episodio.		

2.2. Modelo de dominio:



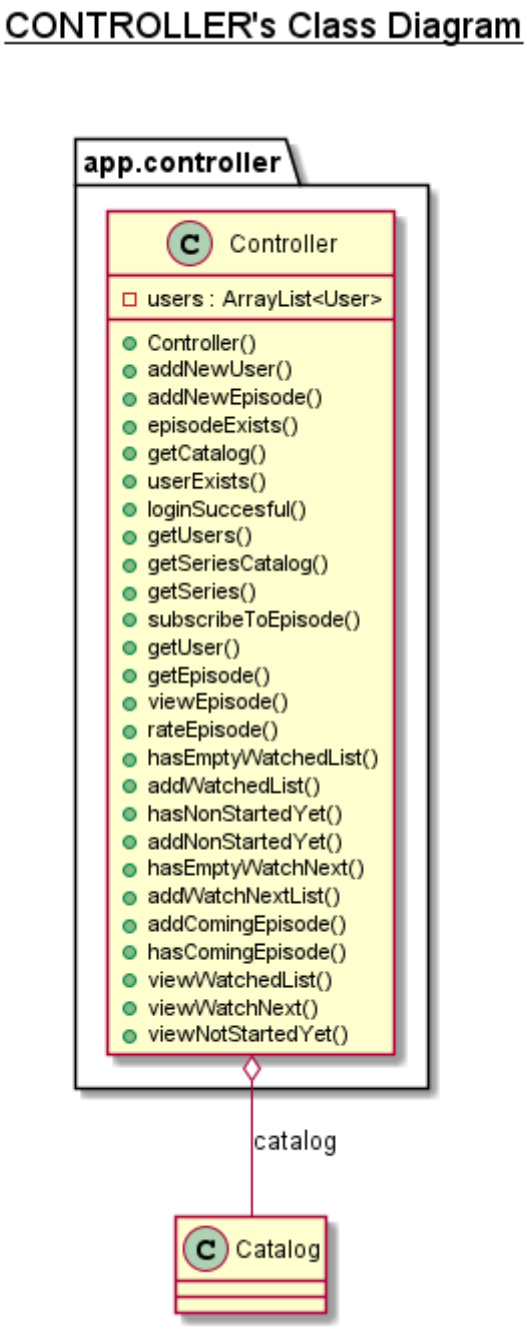
Seguir hipervínculo para ver la imagen más grande.

Hemos modificado ligeramente el modelo de dominio respecto al de la práctica 1 para corregir errores. Por una parte, la clase conceptual “EstadoSerie” ha desaparecido, ya que es extrapolable de “EstadoEpisodio”. Además, hemos añadido dos clases conceptuales nuevas: “VisualizaciónEpisodio” y “Suscripción”, ya que ambas tienen asociaciones interesantes, por lo que son

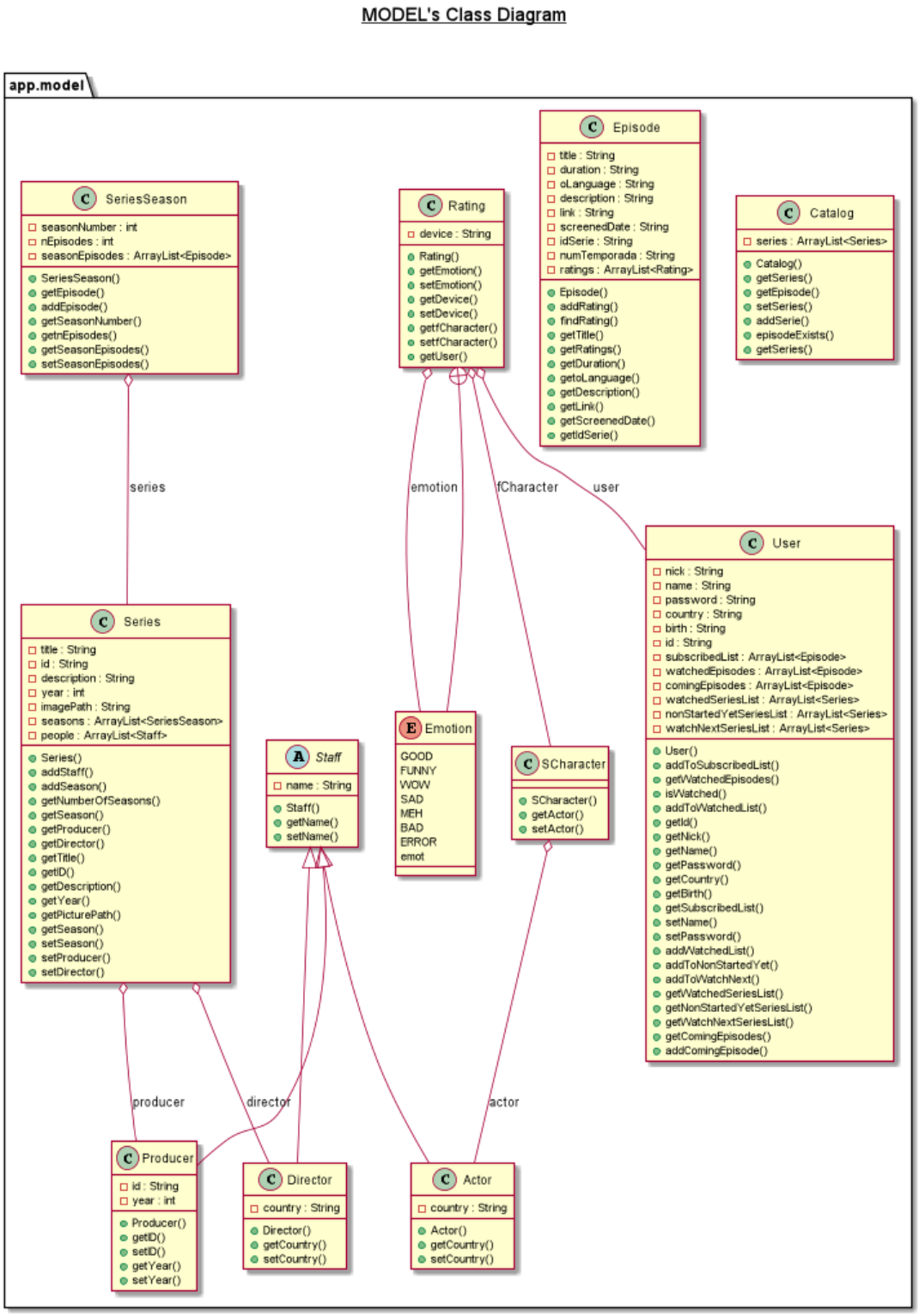
necesarias. La suscripción la hace un usuario a un episodio visto y la visualización de un episodio es soportada por un dispositivo y reproduce un episodio.

2.3. Modelo de clases:

2.3.1. Paquete controlador:

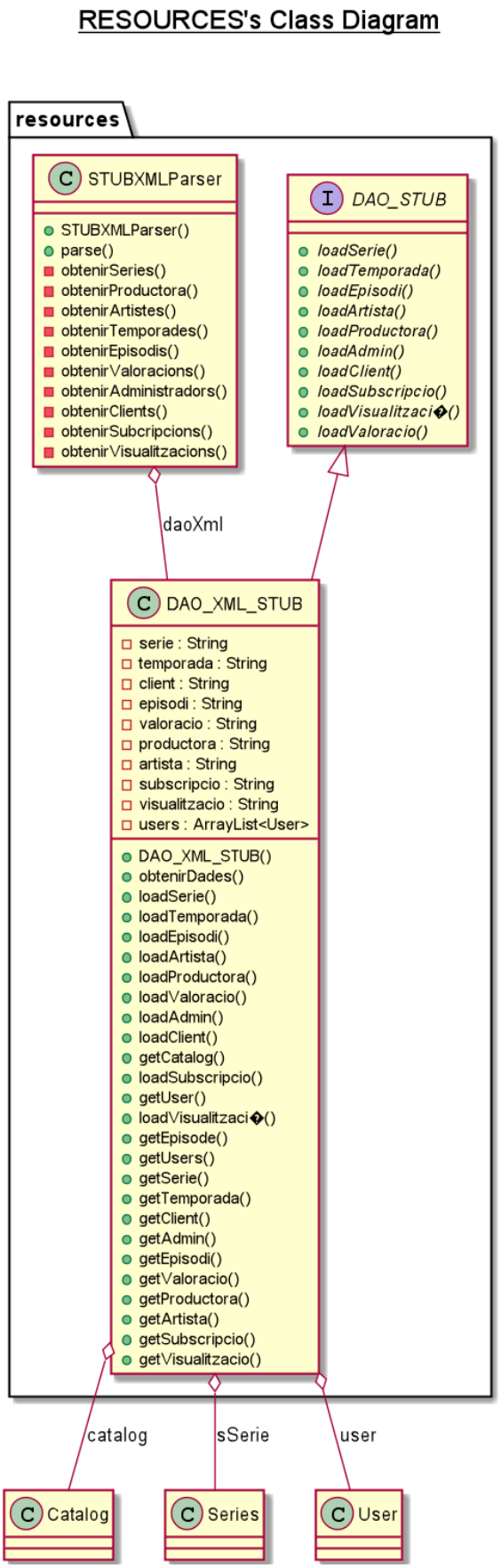


2.3.2. Paquete modelo:



Seguir hipervínculo para ver la imagen más grande.

2.3.3. Paquete recursos:



3. Observaciones

Durante el desarrollo de la práctica hemos tomado diferentes decisiones tanto para la implementación de la aplicación como de los tests.

En el constructor de la clase Rating, hemos usado un switch y un enum para pasar de la puntuación numérica a las diferentes emociones que soporta el sistema. A parte de las emociones que teníamos que implementar, hemos añadido una séptima valoración de emoción que es de error, cuando no se ha utilizado un número en el rango permitido. Aunque haciéndolo de esta manera se viola el principio Open-Closed, eliminarlo supondría complicar la lectura del código y su posterior mantenimiento. Es por esto que hemos considerado que no es necesaria su sustitución.

Para la capa de personalización hemos decidido no implementar *DAO_XML_STUB* como atributo. Como controlador solo lo utiliza durante la creación para cargar los datos, hemos pensado que es completamente innecesario que controlador delegue en *DAO_XML_STUB* para realizar sus funciones. Dicho esto, el constructor de controlador, crea un objeto *DAO_XML_STUB* el cual desecha una vez recupera los usuarios y el catálogo de este.

Otra cosa que hemos tenido en cuenta es que a la hora de hacer las clases, hemos intentado que cada una de ellas tenga una única responsabilidad, para no violar el principio de Single-Responsability de los principios SOLID. Es por ello que han aparecido muchas clases con bajo acoplamiento, aunque algunas irremediablemente están más acopladas que otras.

También hemos observado un problema con el parser, y es que viola el principio Interface Segregation, ya que nos obliga a implementar comportamientos que no nos hacen falta, más aún no nos proporciona elementos que tendríamos que

usar, como es el caso de los personajes de una serie o el dispositivo en el que se reproducen. Para solucionarlo, hemos añadido estos parámetros de forma global para todos los métodos que los necesitaban.

Para el último test, el de ver la lista de NonStartedYet he creado un criterio de aceptación para saber si el usuario al que llamamos con su *id* tiene episodios que saldrán próximamente. Para ello añadimos un episodio que vaya a salir próximamente a una lista que tenemos en usuario y si, coincide con la serie que tenemos lo agregamos en la lista de episodios que van a salir próximamente.

Finalmente en todos los test, hemos implementado un método llamado *init()* con el tag *@Before*. Este se ejecuta siempre antes de cada test y carga de nuevo los datos en controlador. De este modo, en los test podemos modificar los datos de controlador sin tener que preocuparnos de arrastrar errores a los siguientes test.

4. Conclusiones

Cuando se nos propuso esta segunda práctica de Diseño de Software decidimos repartir las tareas que nos recomendaban en la guía de manera equitativa para así poder aligerar la carga del proyecto. Aún así, Rodrigo nos facilitó las clases y la mayoría de los métodos comunes entre ellas para que pudiéramos trabajar con más facilidad bajo un estándar de código. Christian trabajó en el modelo de dominio ya que debíamos cambiar la estructura dado que los requisitos del código y del diseño tenían que ser parejos. Hicimos tres test cada uno, aunque nos ayudáramos para facilitar las tareas de todos.

Durante el trabajo, hemos aprendido como desarrollar software según el estándar del “Test Driven Development”. Para ello hemos tenido que aprender a usar concordion y un poco del lenguaje de programación *httml* para hacerlo funcionar. También hemos tenido que adaptar las necesidades del código a nuestro modelo de dominio.

A la vez que realizábamos el trabajo hemos tenido en cuenta todos los principios de SOLID para poder aplicarlos. Hemos respetado todos los principios excepto el de Open/Closed por una excepción explicada anteriormente. Nos hemos visto obligados debido a la implementación que debíamos hacer.

En resumen, hemos distribuido el trabajo y hemos intentado ceñirnos lo mas posible al guion siendo flexibles cuando el código lo pedía para así poder cumplir igual los requisitos del diseño de software.