PROJECTE IMUB

Javier Hengda Wu Lin
NIUB: 20723043
Joan Josep
NIUB: 20649226

2023 - 2024

INDEX

Contents

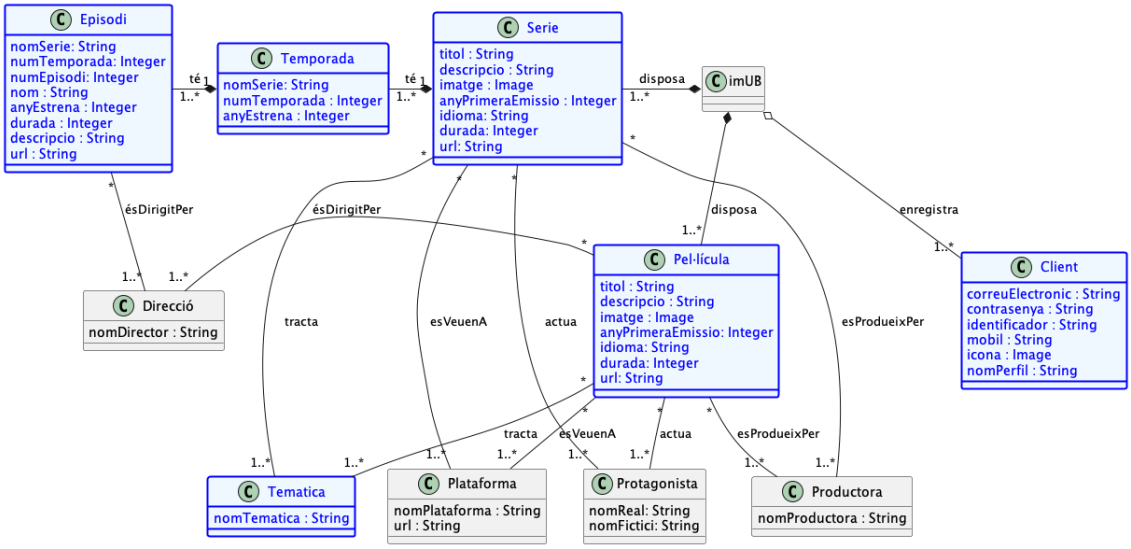
Introducció

En aquesta pràctica s'ha desenvolupat en Java els Minims Productes Variables de l'enunciat de IMUB: Interesting Movies at UB (but not just movies) fins a les extensions opcionals incloent la valoració d'episodis i el Top10 de les Series millor valorades. A continuació es detallarà tota la implementació feta.

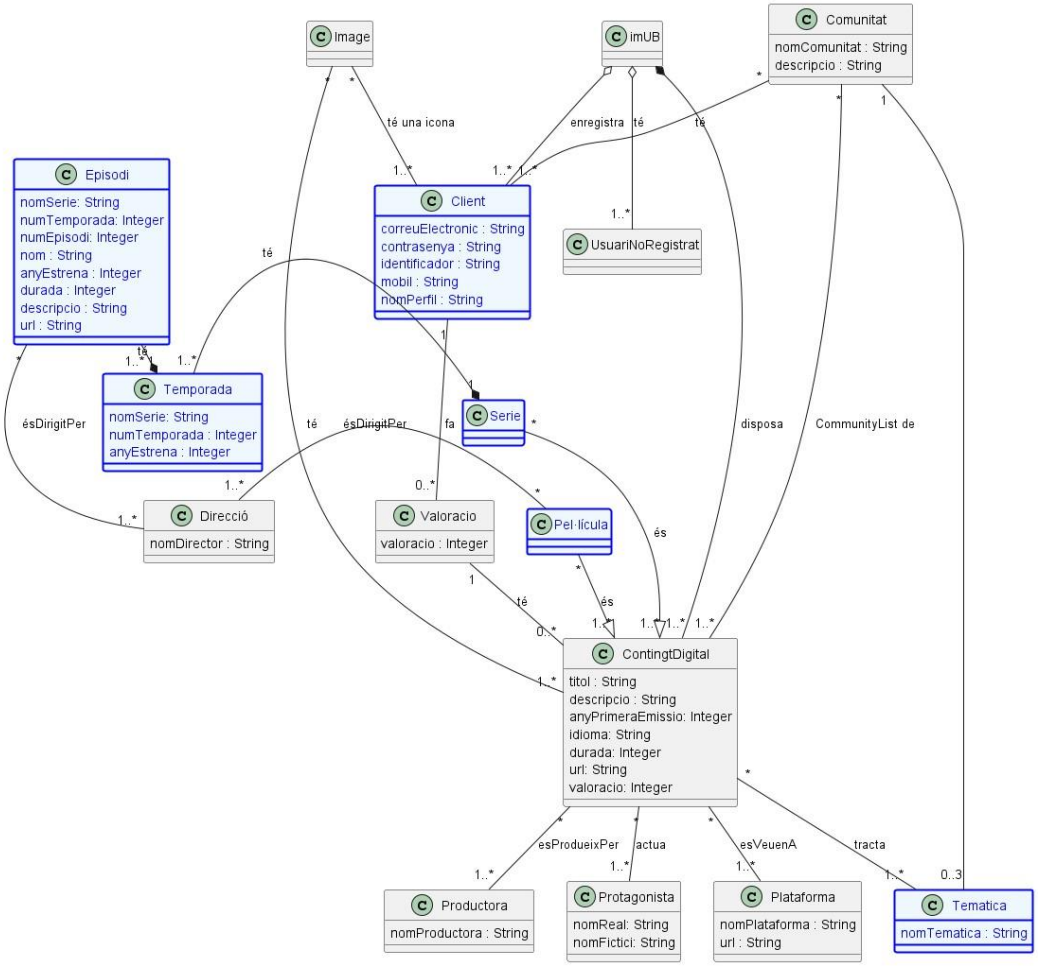
- S'ha connectat la base de dades proporcionada al programa principal.
- S'ha implementat la WishList de les series y les pelicules com una clase de Llista per guardar Contingut Digital.
- S'ha implementat HashMaps de cada tipus de valoració (punts, estrelles, likes) per guardar les valoracions fetes per l'usuari.
- S'ha implementat la funcionalitat de poder valorar les Series y Pelicules del sistema.
- S'ha implementat els diferents tipus de valoracions (estrelles i likes)
- S'ha implementat els diferents tipus de càlcul per poder valorar.
- S'ha implementat la top10 de Series y Pelicules en el menu principal del programa.

Model de Domini

Aquest es el model de domini de la versió de la pràctica 1



Aquest es el model de domini de la versió de la pràctica 4



Millores respecte altres versions

En aquesta pràctica s'ha incorporat noves patrons de disseny que no estaven en les altres pràctiques que es detallarà en el capítol següent, per resumir la part dels patrons de disseny, s'implementat el ús de Strategy i Method Factory per poder fer les valoracions, el Strategy s'ha fet servir per els diferents tipus de càlcul i Method Factory per la creació de les valoracions. També s'ha incorporat HashMaps per guardar les valoracions, en total tres HashMap que correspon al tipus de valoració (punt, estrelles i likes), en els HashMaps es guarda el Contingut Digital directament amb el numTemporada i el numEpisodi, també es guarda el Client que ha valorat i la Valoració creat, amb aquest enfocament s'ha de tenir en compte que les Pelicules emmagatzemades al HashMap contindrà també un numTemporada i un numEpisodi, per defecte s'ha deixat els seus valors a (numTemporada = 1, numEpisodi = 1) per les pelicules.

Sobre les valoracions també s'ha creat una clase enum ValorType per gestionar el tipus de Valoració més fàcilment, s'ha creat dos clases relacionades al funcionament de la valoració, una clase RatingManager.java que realitza les addicions de les valoracions al HashMap y RatingValue.java que retorna les llistes de Serie y Pelicula en el format correcte de la pràctica.

Patrons de Disseny

Implementacions

- Singleton: S'ha implementat aquest patró al Controller per només crear un controlador per aplicació. Encara que no serveix per a molt perquè el controlador només es crea una vegada durant tota la execució.

- Strategy: S'ha implementat per els càlculs de les valoracions per punt mitjançant absolut o promig.

- Method Factory: Per la creació de noves valoracions.

- Observer: Utilitzat per saber quan una Serie o Pelicula es afegida a la WishList o valorat, s'ha creat una clase UpdaterManager.java com una clase "Publisher", una interfície UpdateListener.java com una clase "Subscriber" i una clase "MainListener" que implementa la clase UpdateListener, per poder actualitzar la ventana Main. La "EscenaMain" implementa el MainListener i quan les clases "EscenaValorarObra", "EscenaPelicula" i "EscenaEpisodi" notifiqui al Main, enviarà una solicitud a la "EscanaMain" i actualitzarà amb les noves dades.

Singleton:

o Quin problema soluciona en la pràctica

S'ha implementat al Controller però no soluciona res en la pràctica perquè durant execució només es crea un controlador (opcional: s'utilitza per accedir al "ViewMemory.java" desde la MainEscena.java per donar notificació de precaució al tancar la ventana del MainEscena), al contrari al fer el Controller Singleton pot ser que es vulneri el Single Responsability Principle perquè el Controlador serveix com una classe de control de lògica entre el View i el Model i també controla la creació de instancies.

o El patró genèric del patró

El patró genèric del patró consisteix en els següents passos:

1. Fer el constructor de la classe privat.
2. Crear un atribut static de la classe a implementar el patró.
3. Crear metòde static "getInstance()" per inicialitzar la clase desde un metòde que avaluarà si ja s'ha creat una instancia de la classe o no, si ja s'ha creat retornarà l'instancia, si no, es retornarà una nova instancia.

o La identificació de cada component del patró (Client, Context, etc.)

Client = "AppMain.java"

Singleton = "Controller.java"

Strategy:

o Quin problema soluciona en la pràctica

S'ha implementat per fer el càlcul de les valoracions, soluciona principalment el Open Closed Principle, ja que per afegir un nou càlcul de valoració ara només s'ha de crear una nova clase i acoplar-lo a la clase interfície Strategy que en aquest cas es "ValorarStrategy.java". També es pot considerar solucionar el Single Responsability Principle ja que la clase només realitza una funcionalitat i desacopla de les altres clases el càlcul de les valoracions.

o El patró genèric del patró

El patró genèric del patró consisteix en els següents passos:

1. Crear la classe Context que manté una referencia a una de les estratègies creades, es pot canviar de estratègia amb un metòde "setStrategy()" i realitzar l'operació de la estratègia escollida amb una altra metòde o metòdes.
2. Crear la interfície que conectarà el Context amb les Concrete Strategies.
3. Crear els Concrete Strategies i construir els metòdes d'operació en cada una d'elles.

o La identificació de cada component del patró (Client, Context, etc.)

Context = "ComputeValue.java"

Strategy <interface> = "ValorarStrategy.java"

Concrete Strategies = "ComputeByAbsolut.java" i "ComputeByPromig.java"

FactoryMethod:

o Quin problema soluciona en la pràctica

S'ha implementat per la creació de les valoracions, soluciona principalment el Single Responsability Principle, ja que encapsula la creació dels tipus de valoració i el principi de Open Closed, ja que per crear un nou tipus de valoració ara només es necessari crear una nova classe que representi el nou tipus de valoració.

o El patró genèric del patró

El patró genèric del patró consisteix en els següents passos:

1. Crear la classe Creator que serà l'encargat de crear els diferents tipus de classe.
2. Crear les classes de Concrete Creator que crearà les classes concretes.
3. Crear la interfície Product que conectarà els Creators amb els Productes.
4. Crear els Concrete Product que són les diferents implementacions de un Product.

o La identificació de cada component del patró (Client, Context, etc.)

Creator = "ValorFactory.java"

Concrete Creator = "PointValorType.java", "StarValorType.java", "LikeValorType.java"

Product <interface> = "Valoracio.java"

Concrete Products = "PointValor.java", "LikeValor.java" i "StarValorType"

Observer:

o Quin problema soluciona en la pràctica

S'ha implementat per l'actualització de la EscenaMain, soluciona el problema de refresc de les tables de la WishList i el Top10 de la interfície gràfica.

o El patró genèric del patró

El patró genèric del patró consisteix en els següents passos:

1. Crear la interfície Subscriber que conectarà els Publisher i els Concrete Subscribers.
2. Crear o identificar la classe Publisher que mantindrà una llista dels subscriptors, quan el Publisher canvia d'estat o executa una instrucció d'interès per "Subscriber", informarà als "Subscriber" que estiguin subscrits.
3. Crear els Concrete Subscribers que necesitem saber sobre l'estat de la classe Publisher.

o La identificació de cada component del patró (Client, Context, etc.)

Subscriber = "UpdateListener.java" que lo utilitzarà "EscenaPeliculaDetalls.java", "EscenaEpisodiDetalls.java" i "EscenaValorarObra.java".

Publisher = "UpdaterManager.java"

Concrete Subscribers = "MainListener.java" que lo utilitzarà "EscenaMain.java"

Diagrama de classes

*Ho deixo a la carpeta Doc del projecte

Conclusions

En aquesta pràctica s'ha enfocat en fer la part de les valoracions correctament, s'ha implementat noves patrons de disseny i s'ha implementat el correcte funcionament de la interfície gràfica incloent relacionar la base de dades amb el programa, fer una nova classe "VistaMemory" per saber el estat dels components de la interfície gràfica i implementat la funcionalitat de valoracions i wishlist del projecte.

El projecte s'ha realitzat totalment per Javier Hengda Wu Lin fins aquest part.