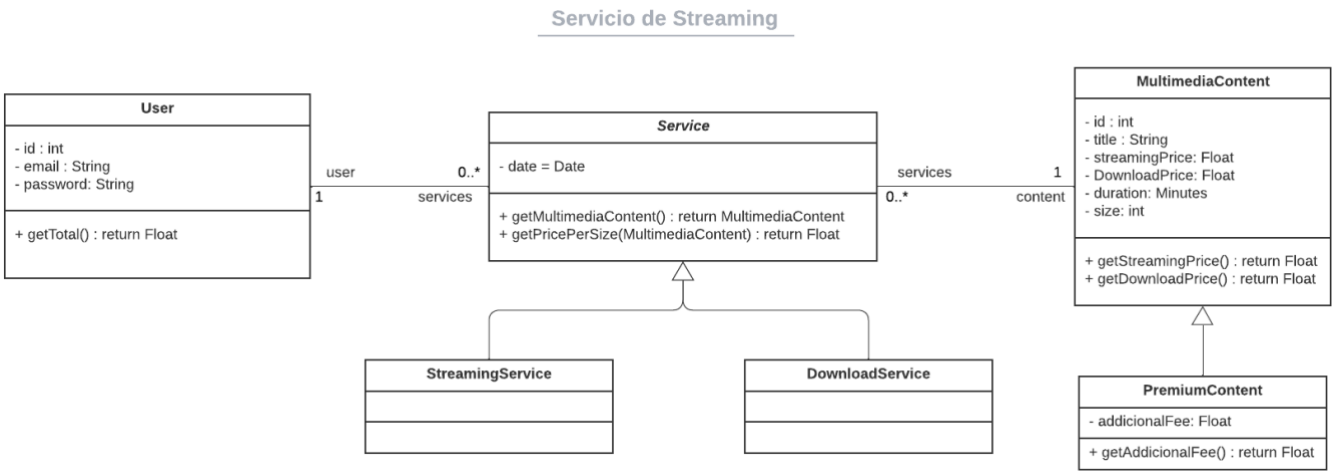
Principios de diseño

Práctica 1: Servicio de Streaming

Nos han contratado para desarrollar un software para gestionar el streaming de películas online. Disponemos de un análisis previo que podemos utilizar como punto de partida, pero que habrá que corregir y mejorar:



Como podemos ver en el diagrama de partida, los usuarios tienen una operación que devuelve el importe total pagado por todos los servicios que han solicitado. Los usuarios pueden contratar el servicio que deseen de cualquier contenido, escogiendo si lo quieren vía Streaming o vía Download.

Ejemplo: Un usuario quiere tener la película de Avatar 2 en sus dispositivos, ya que se va de viaje y necesita ver el contenido en local. Con lo cual, contratará un servicio Download del contenido multimedia Avatar 2. Además, como la película es muy reciente, está catalogada como un contenido premium, así pues, además de su coste de descarga tendrá un cargo adicional por premium.

A continuación, se muestra la implementación del método del usuario registrado que devuelve el total a pagar por todos los servicios que ha contratado.

```
public class User {

	private List<Service> services;

	public float getTotal() {

		float total = 0.0;
		foreach (Service s in services) {

			MultimediaContent mc = s.getMultimediaContent();

			if(typeof(s)==StreamingService) {
				total += mc.getStreamingPrice();
			}else if(typeof(s)==DownloadService){
				total += mc.getDownloadPrice();
			}

			if (typeof(mc)==PremiumContent) {
				total += mc.getAdditionalFee();
			}
		 }
		return total;
	 }
}
```

Ejercicio 1

Revisando el diagrama UML y el código del método getTotal() nos preocupa que su diseño sea un poco frágil y esté violando algunos principios de diseño. Nos planteamos las siguientes preguntas:

¿Este diseño satisface los siguientes principios de diseño? Justifica tu respuesta.

a) Ley de Demeter

b) Abierto-Cerrado

c) Sustitución de Liskov

d) Responsabilidad Única

Ejercicio 2

Propón una solución a los errores que has encontrado en el primer ejercicio. Realiza los cambios que consideres oportuno tanto en el UML como en el código. Puedes generar código de otras clases si lo ves necesario.