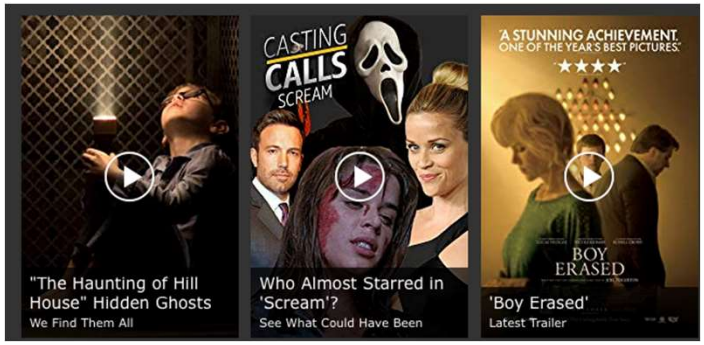
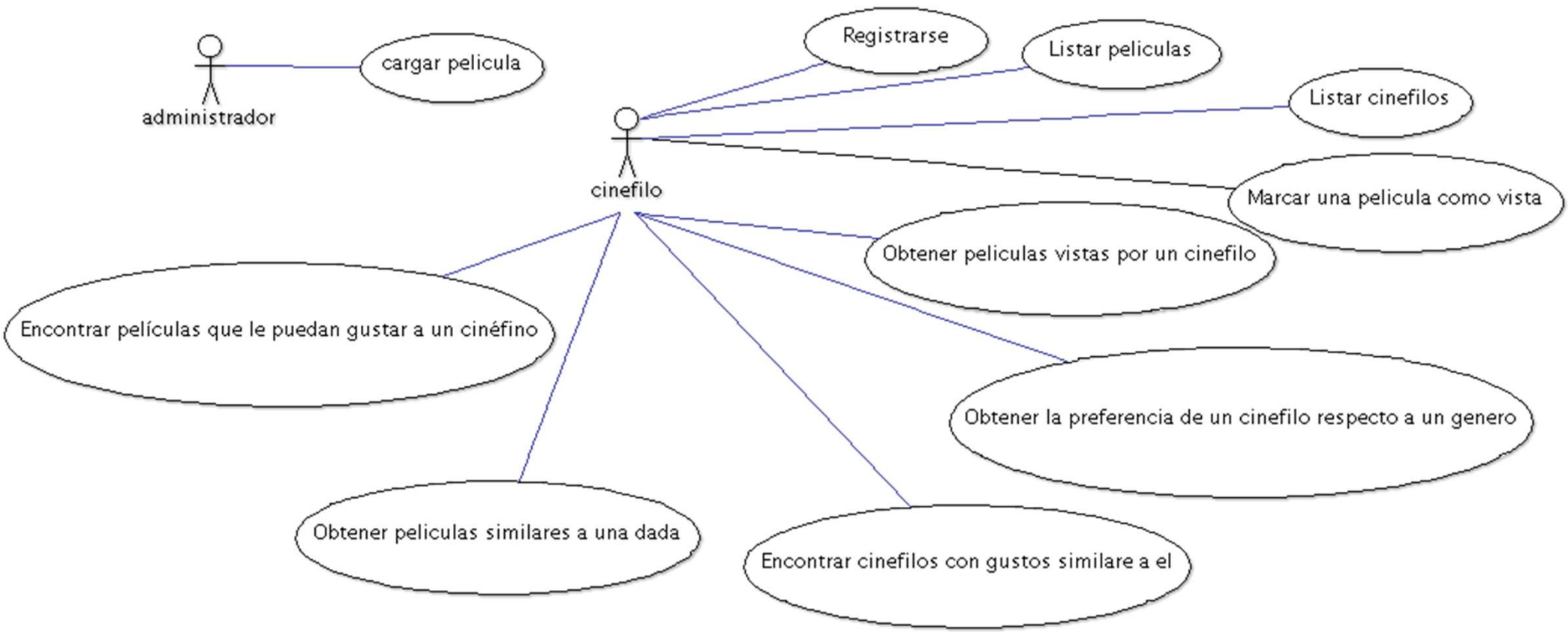


Cinéfiloos

Un sitio para encontrar películas que nos puedan gustar, y gente con gustos como los nuestros

Cinéfilo: El **cinéfilo**, en términos generales, es una persona que tiene un gusto especial por el cine.

Cinefiloos es un sitio para llevar registro de las películas que uno ha visto, encontrar a cinéfilos con gustos parecidos, y obtener sugerencias. El sistema se apoya en IMDB, un sitio en el que ya están registradas la mayoría de las películas. Eso hace innecesario cargar toda la información. A continuación se describen los casos de uso en formato breve.


administrador
cargar pelicula
cinefilo
Registrarse
Listar peliculas
Listar cinefilos
Marcar una pelicula como vista
Obtener peliculas vistas por un cinefilo
Encontrar películas que le puedan gustar a un cinéfino
Obtener la preferencia de un cinefilo respecto a un genero
Obtener peliculas similares a una dada
Encontrar cinefilos con gustos similare a el

Cargar película: Se ingresa título, la URL de la película en IMDB, y la URL de la imagen de portada de la película (también tomada de IMDB). Adicionalmente indica el "perfil de género" de la película. Cinefiloos (el sistema) registra la película.

El perfil de género indica en una escala de 0 a 9, cuánto de cada género tiene una película. Los géneros que se consideran son:

horror, action, romance, suspense, comedy, y sci-fi.

Si no se indica perfil de género, tomará 0 para todos los géneros (sin género definido).

Registrar un cinéfilos: el cinéfilo ingresa su nombre completo, y su email. El sistema registra al cinéfilo y lo retorna.

Obtener películas: El sistema retorna la lista de películas.

Obtener cinéfilos: El sistema retorna la lista de cinéfilos.

Marcar una película como vista: Se indica un cinéfilo y una película. El sistema registra la película, como vista por el cinéfilo.

Obtener películas vistas por un cinéfilo: Dado un cinéfilo, el sistema retorna la lista de películas vistas por el cinéfilo.

Encontrar películas parecidas: Retornar todas las películas que son parecidas a una que se indique. Retorna todas las películas cuyo "índice de similitud" a la película indicada es menor a 6 (menor índice implica, más parecidas). La lista no está ordenada. La película indicada también está en la lista.

	horror	action	romance	suspense	comedy	sci-fi
Película 1	9	0	0	5	0	0
Película 2	0	0	0	9	0	7

Índice de similitud (mejor significa más parecidos):

- por cada género calculo la diferencia entre las dos
- sumo los valores absolutos

índice: abs(9-0)+0+0+abs(5-9)+0+abs(0-7) = 20

Obtener la preferencia de genero de cine de un cinéfilo: Dado un cinéfilo retorna su preferencia de genero de cine (que tiene un número de 0 a 9 por cada género, al igual que las películas).

La preferencia de género de cine de un cinéfilo se calcula a partir de las películas que vio (asumimos que le gustaron).

Si no vió ninguna, se toma 4.5 para todos los géneros (ni muy muy, ni tan tan). Luego, toma el promedio entre esa base y todas las películas que vea.

	horror	action	romance	suspense	comedy	sci-fi
Base	4.5	4.5	4.5	4.5	4.5	4.5
Película 1	9	0	0	5	0	0
Película 2	6	0	0	9	0	2
Promedio	6.16	1.5	1.5	6.1	1.5	2.1

Encontrar cinéfilos con gustos similares: Dado un cinéfilo, encontrar todos aquellos que tienen preferencias de género de cine parecidas. Para calcular la similitud entre dos cinéfilos, se toman las preferencias de género de cine de cada uno.

Retorna todos los cinéfilos cuyo "índice de similitud" al indicado es menor a 5 (menor índice significa más parecido). La lista no está ordenada. El cinéfilo está en la lista también.

El cálculo es idéntico al de similitud entre películas

Encontrar películas que pueden gustar, no vistas: Dado un cinéfilo, retorna todas las películas cuyo "índice de similitud" a la preferencia del cinéfilo es menor a 7, y que el cinéfilo no vio. La lista no está ordenada.

El cálculo es idéntico al de similitud entre películas.

Cargar película: Se ingresa título, la URL de la película en IMDB, y la URL de la imagen de portada de la película (también tomada de IMDB).

Registrar un cinéfilos: el cinéfilo ingresa su nombre completo, y su email. El sistema registra al cinéfilo y lo retorna.

Obtener películas: El sistema retorna la lista de películas.

Obtener cinéfilos: El sistema retorna la lista de cinéfilos.

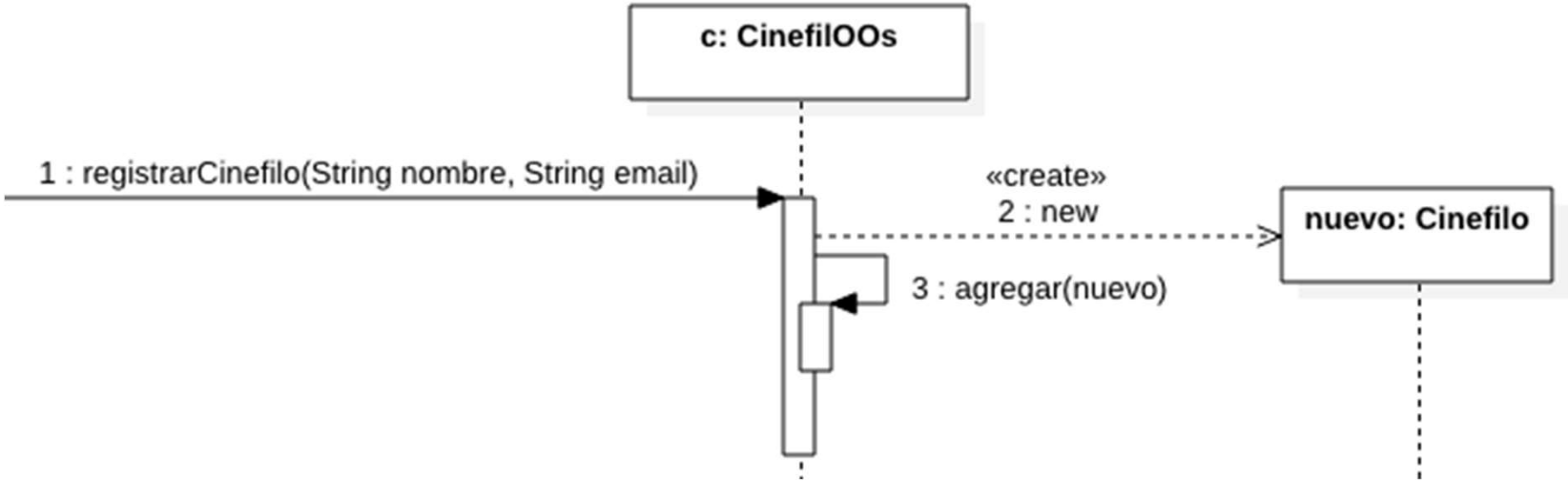
Marcar una película como vista: Se indica un cinéfilo y una película. El sistema registra la película, como vista por el cinéfilo.

Obtener películas vistas por un cinéfilo: Dado un cinéfilo, el sistema retorna la lista de películas vistas por el cinéfilo.

Registrar un cinéfilos: el cinéfilo ingresa su nombre completo, y su email. El sistema registra al cinéfilo y lo retorna.

Qué hay que hacer

- Crear e inicializar el cinéfilo
- Agregar el cinéfilo a la colección
- Devolver el cinéfilo



Registrar un cinéfilos: el cinéfilo ingresa su nombre completo, y su email. El sistema registra al cinéfilo y lo retorna.

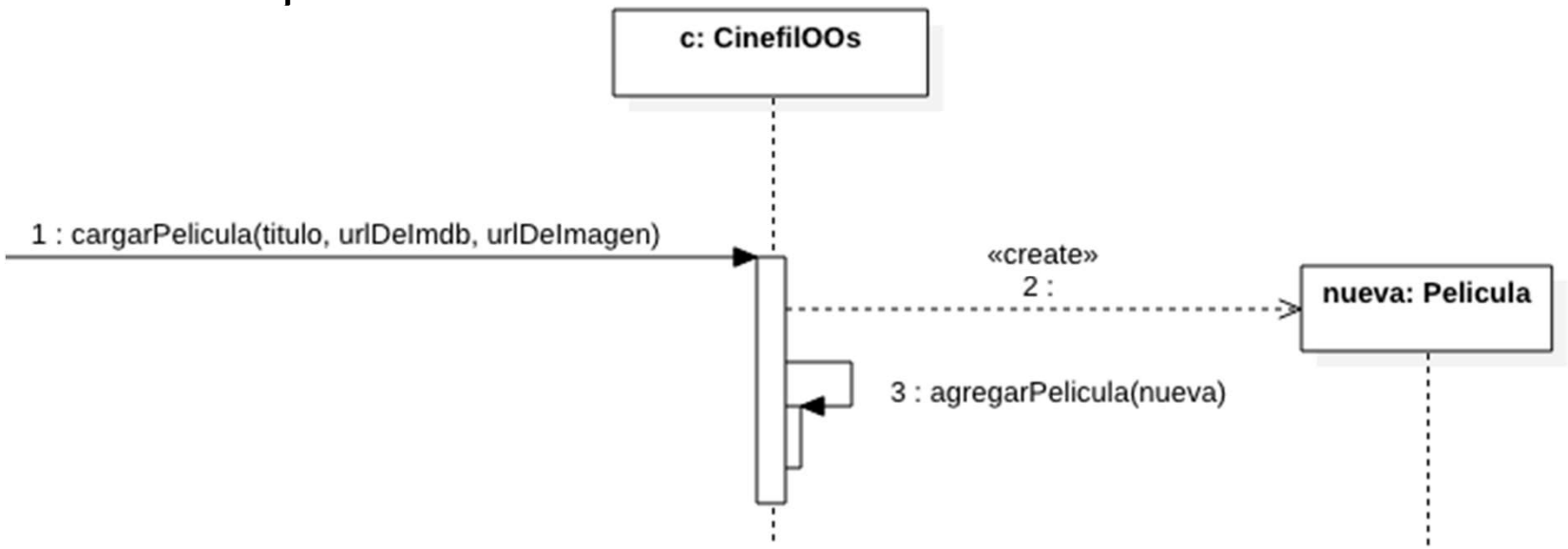
```
public class CinefilOOs
{
    private List<Cinefilo> cinefilos;
    private List<Pelicula> peliculas;

    public CinefilOOs()
    {
        cinefilos = new ArrayList();
        peliculas = new ArrayList();
    }

    public Pelicula registrarPelicula(String titulo, String urlImdb, String urlImage) {
       Pelicula nuevaPelicula = new Pelicula(titulo, urlImdb, urlImage);
       peliculas.add(nuevaPelicula);
       return nuevaPelicula;
    }
```

Cargar película: Se ingresa título, la URL de la película en IMDB, y la URL de la imagen de portada de la película (también tomada de IMDB). Cinefiloos (el sistema) registra la película.

- Crear e inicializar la película
- Agregar la película a la colección
- Devolver la película



Cargar película: Se ingresa título, la URL de la película en IMDB, y la URL de la imagen de portada de la película (también tomada de IMDB). Cinefiloos (el sistema) registra la película.

```
public Cinefilo registrarCinefilo(String nombre, String email) {
    Cinefilo nuevoCinefilo = new Cinefilo(nombre, email);
    cinefilos.add(nuevoCinefilo);
    return nuevoCinefilo;
}
```

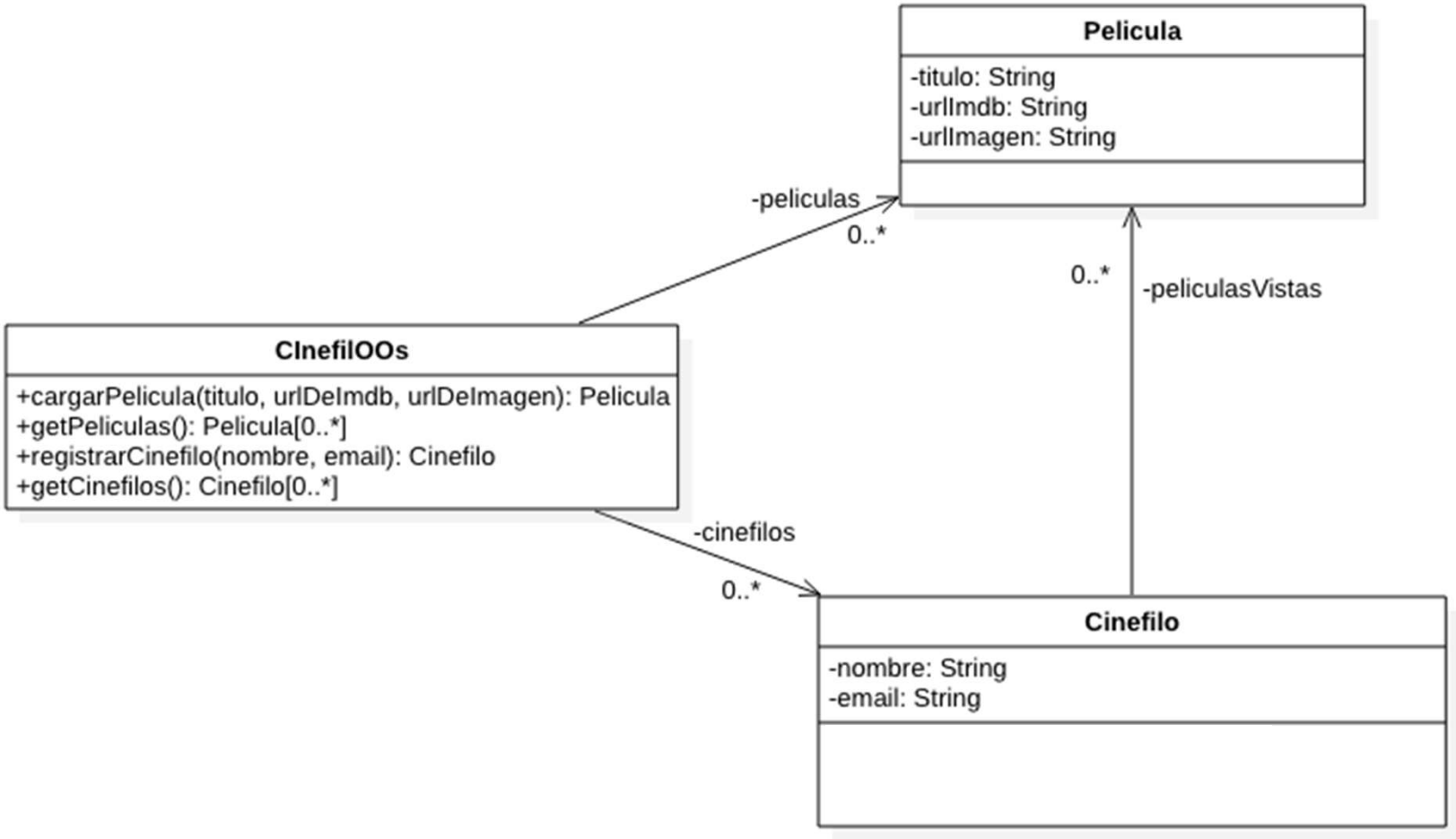
Obtener películas: El sistema retorna la lista de películas.

```
public List<Pelicula> getPeliculas() {
    return peliculas;
}
```

Obtener cinéfilos: El sistema retorna la lista de cinéfilos.

```
public List<Cinefilo> getCinefilos() {
    return cinefilos;
}
```

Cópienlo así lo tenemos a mano



Marcar una película como vista: Se indica un cinéfilo y una película. El sistema registra la película, como vista por el cinéfilo.

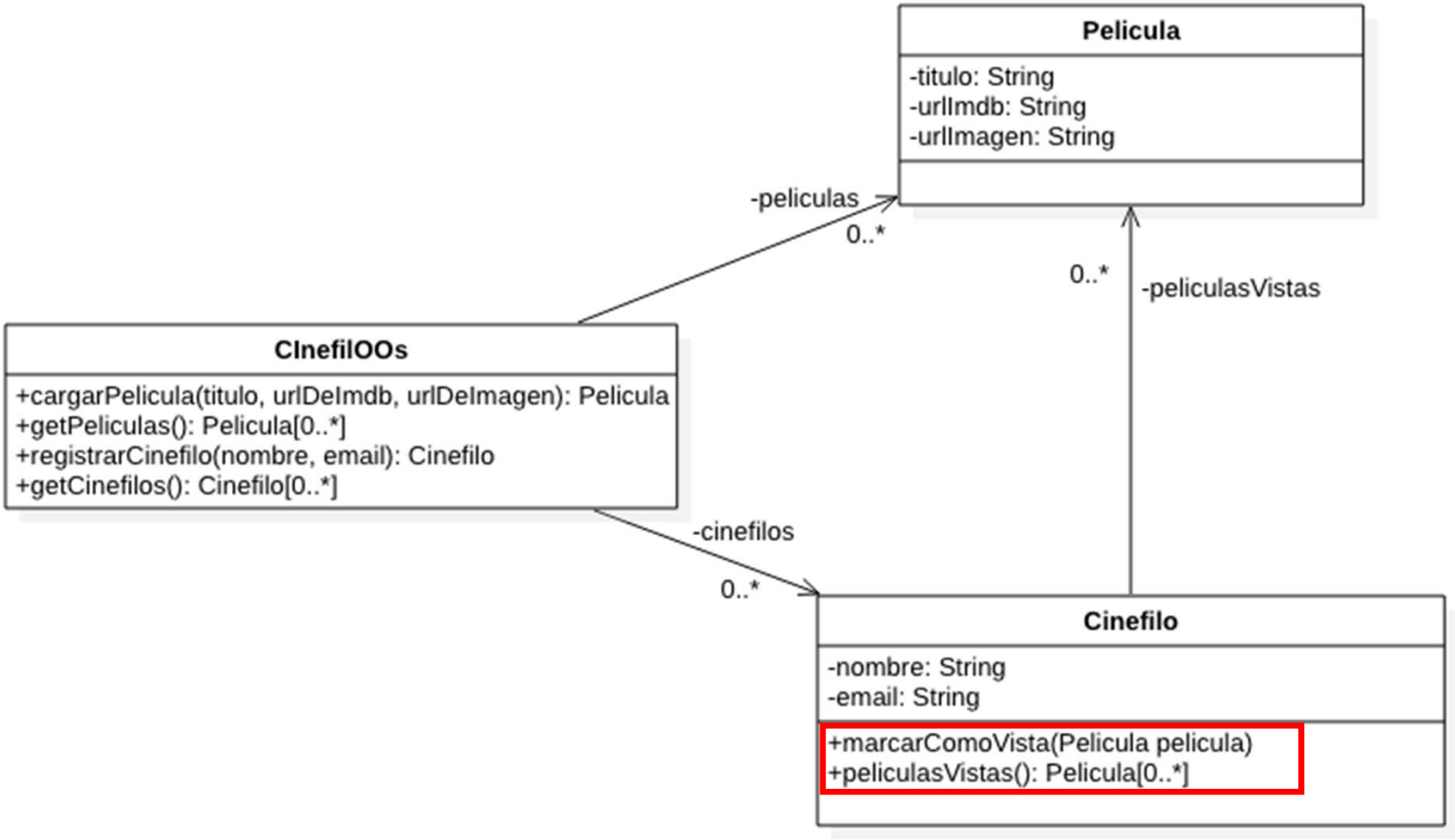
¿Obtener películas vistas por un cinéfilo?: Dado un cinéfilo, el sistema retorna la lista de películas vistas por el cinéfilo.

Tenemos el cinéfilo y la película (los objetos), ¿necesitamos involucrar a CinefilOOs?

Marcar una película como vista: Se indica un cinéfilo y una película. El sistema registra la película, como vista por el cinéfilo.

¿Obtener películas vistas por un cinéfilo?: Dado un cinéfilo, el sistema retorna la lista de películas vistas por el cinéfilo.

Tenemos el cinéfilo y la película (los objetos), ¿necesitamos involucrar a CinefilOOs?



Cópienlo así lo tenemos a mano

Marcar una película como vista: Se indica un cinéfilo y una película. El sistema registra la película, como vista por el cinéfilo.

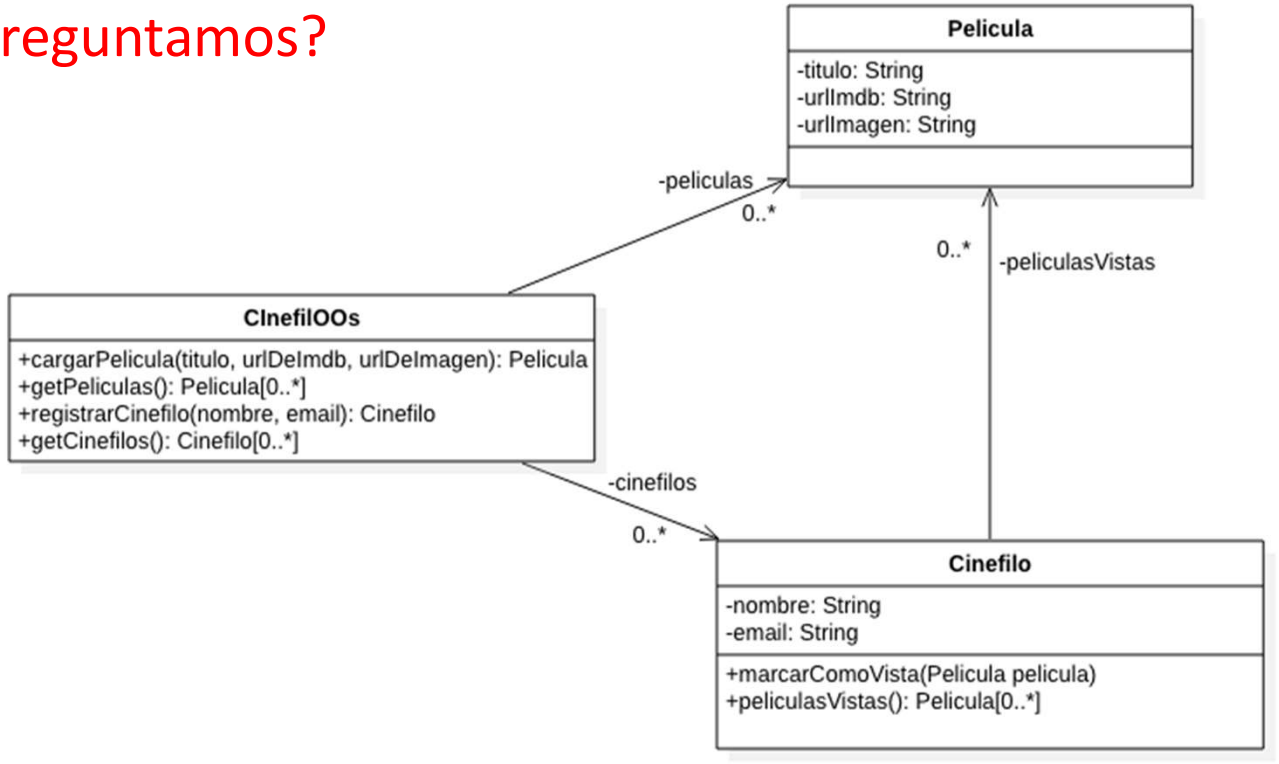
```
public void marcarComoVista(Pelicula pelicula) {
    peliculasVistas.add(pelicula);
}
```

¿Obtener películas vistas por un cinéfilo?: Dado un cinéfilo, el sistema retorna la lista de películas vistas por el cinéfilo.

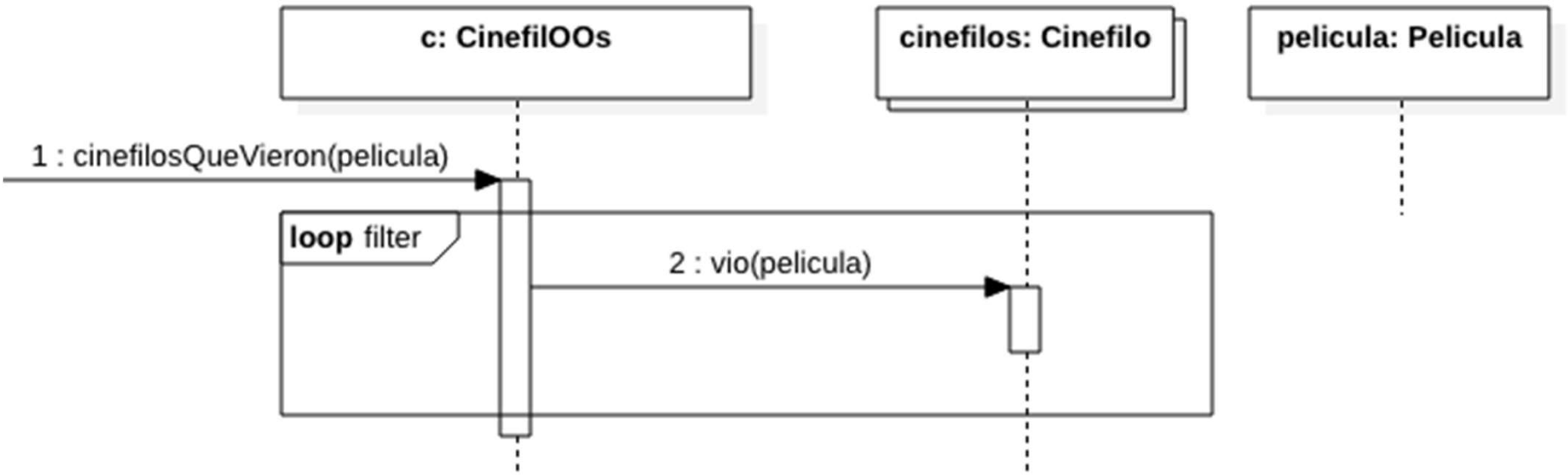
```
public List<Pelicula> getPeliculasVistas() {
    return peliculasVistas;
}
```

Obtener cinéfilos que vieron una película: Dada una película, el sistema retorna la lista de cinéfilos que la vieron**.**

¿A quién le preguntamos?

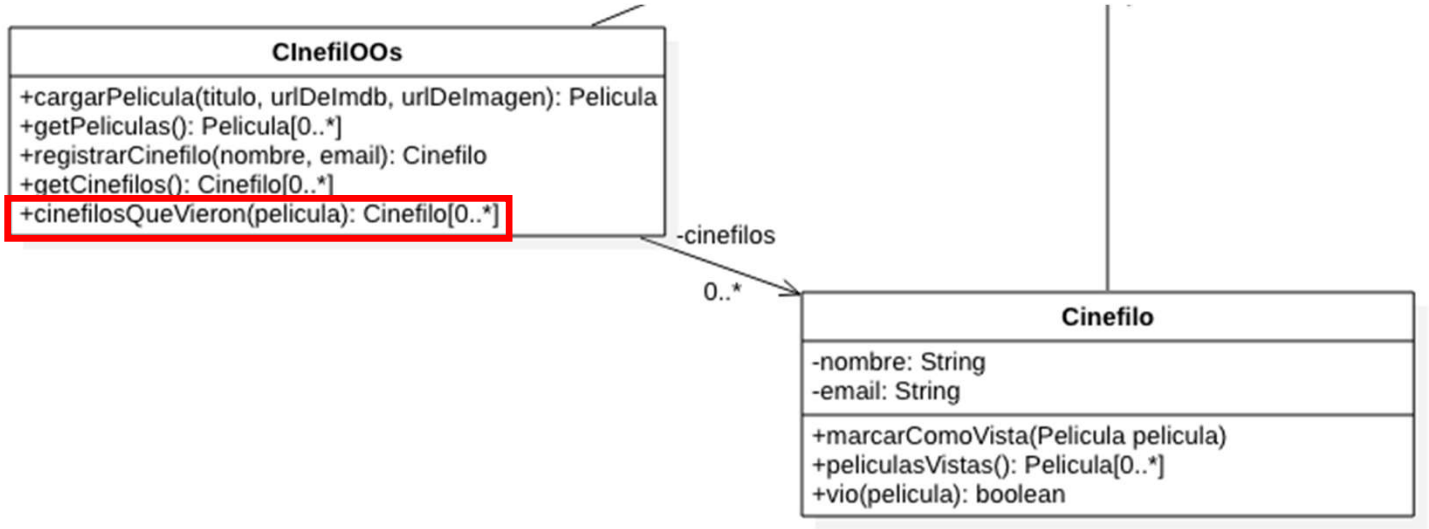


Obtener cinéfilos que vieron una película: Dada una película, el sistema retorna la lista de cinéfilos que la vieron**.**



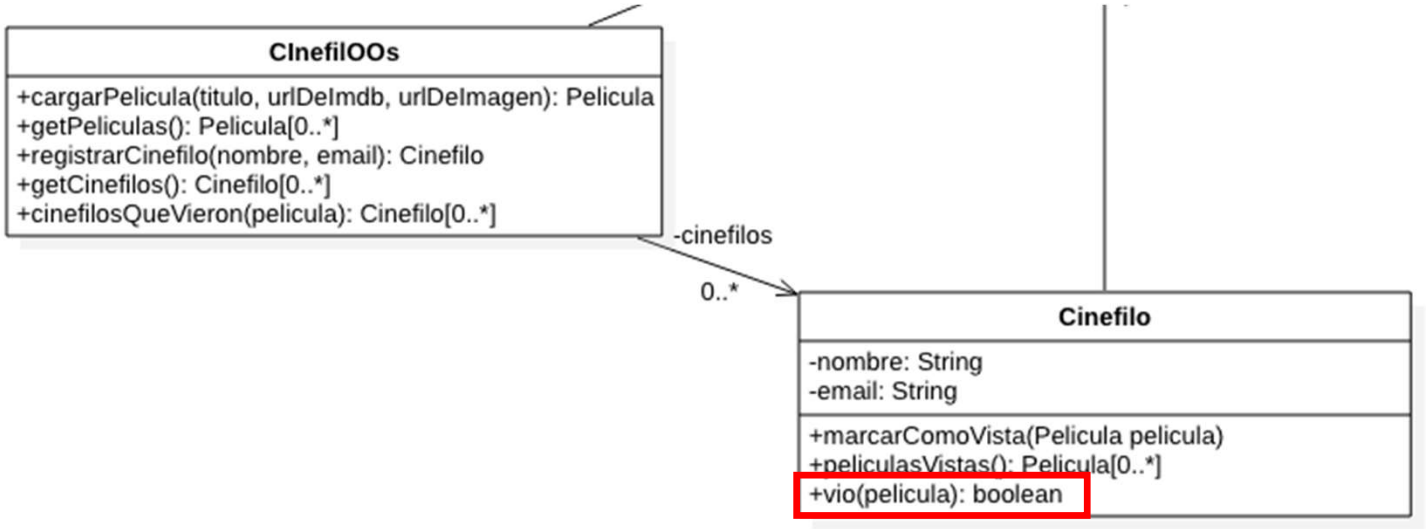
Obtener cinéfilos que vieron una película: Dada una película, el sistema retorna la lista de cinéfilos que la vieron**.**

```java
public List<Cinefilo> getCinefilosQueVieron(Pelicula pelicula) {
    return cinefilos.stream().filter(cinefilo -> cinefilo.vio(pelicula))
        .collect(Collectors.toList());
}
```



Obtener cinéfilos que vieron una película: Dada una película, el sistema retorna la lista de cinéfilos que la vieron**.**

```
public boolean vio(Pelicula pelicula) {
    return peliculasVistas.contains(pelicula);
}
```



Cargar película: Se ingresa título, la URL de la película en IMDB, y la URL de la imagen de portada de la película (también tomada de IMDB). Adicionalmente indica el "perfil de género" de la película. CinefilOOs (el sistema) registra la película.

El perfil de género indica en una escala de 0 a 9, cuánto de cada género tiene una película. Los géneros que se consideran son: horror, action, romance, suspense, comedy, y sci-fi.

Si no se indica perfil de género, tomará 0 para todos los generos (sin género definido).

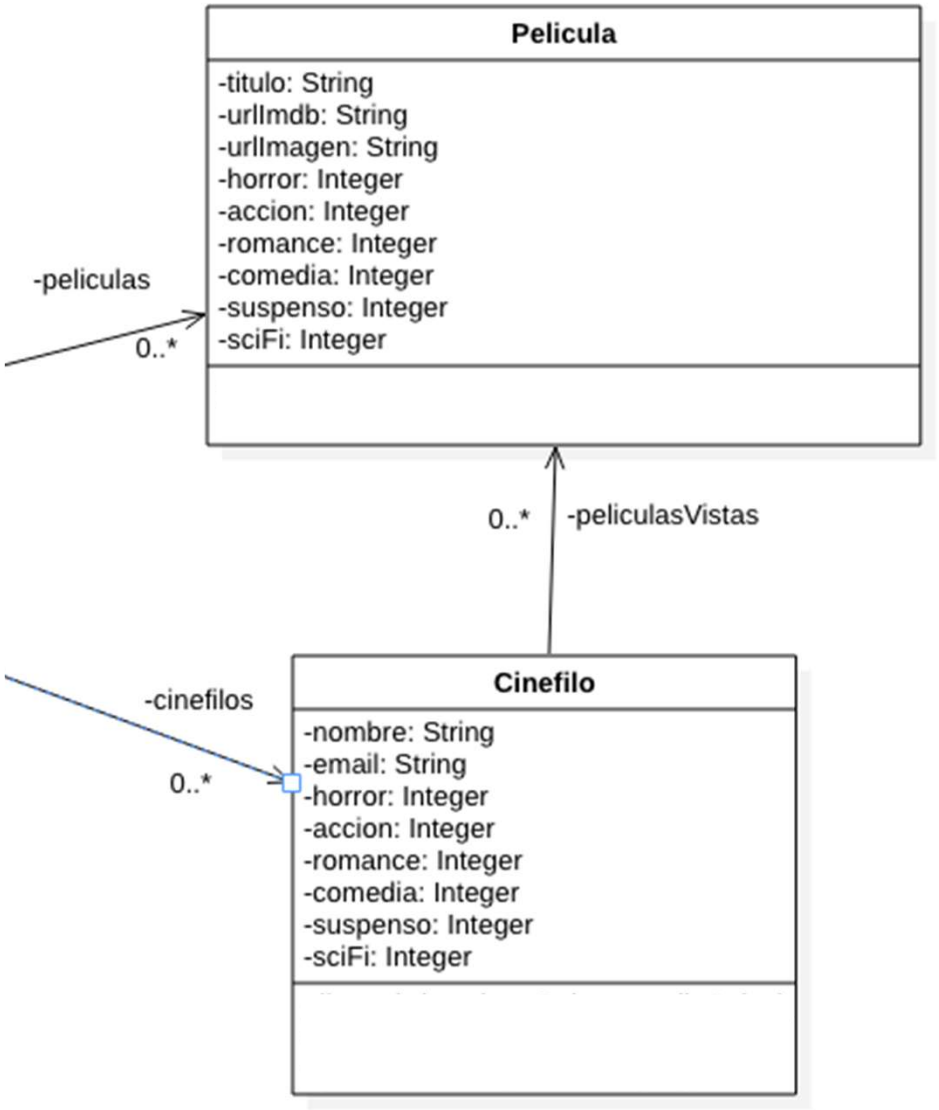
horror	action	romance	suspense	comedy	sci-fi
9	0	0	5	0	0

Obtener la preferencia de genero de cine de un cinéfilo: Dado un cinéfilo retorna su preferencia de genero de cine (que tiene un número de 0 a 9 por cada género, al igual que las películas).

La preferencia de género de cine de un cinéfilo se calcula a partir de las películas que vio (asumimos que le gustaron).

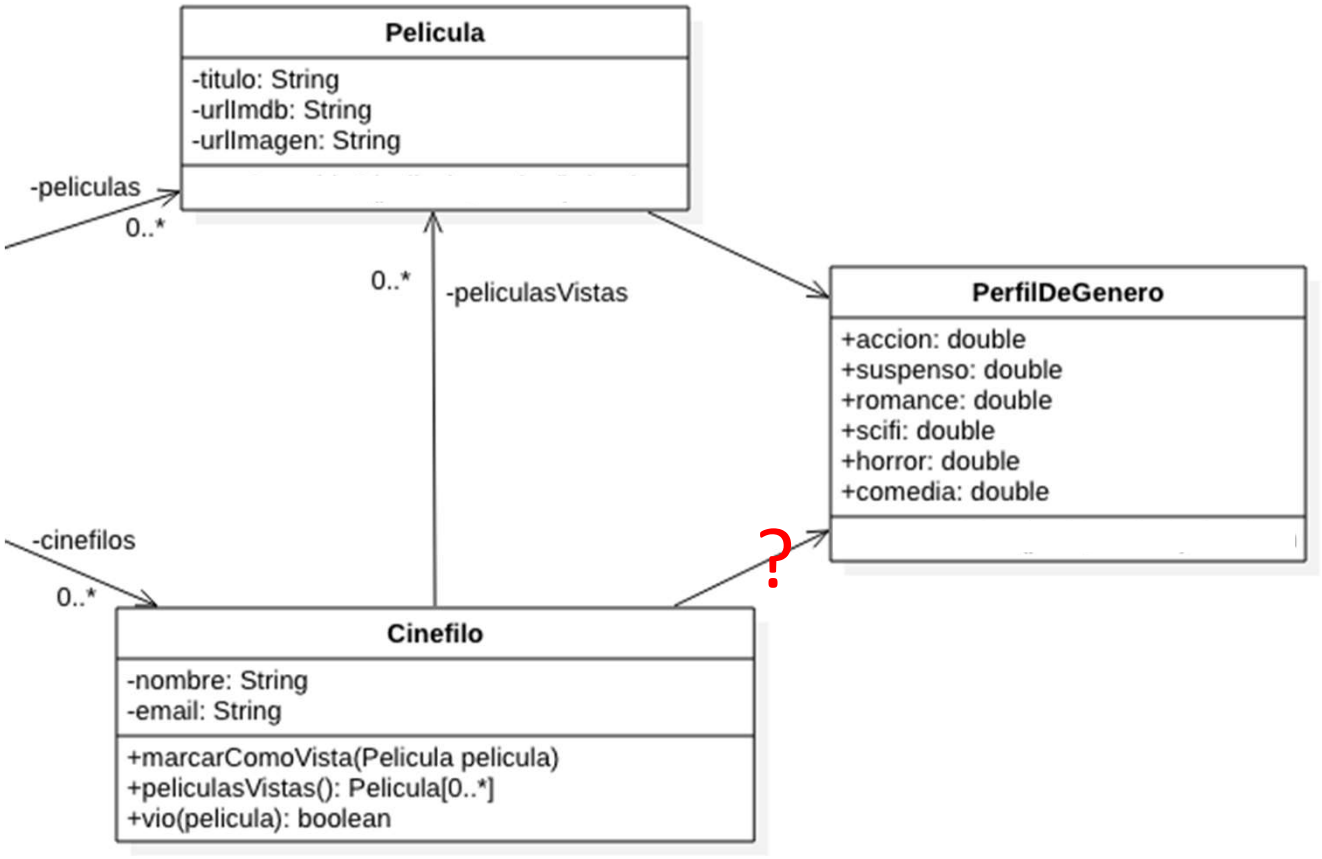
Si no vió ninguna, se toma 4.5 para todos los géneros (ni muy muy, ni tan tan). Luego, toma el promedio entre esa base y todas las películas que vea.

	horror	action	romance	suspense	comedy	sci-fi
Base	4.5	4.5	4.5	4.5	4.5	4.5
Película 1	9	0	0	5	0	0
Película 2	6	0	0	9	0	2
Promedio	6.16	1.5	1.5	6.1	1.5	2.1



- Primer intento de extender el modelo para incluir el perfil de genero de la película, y las preferencias de genero de cine del cinéfilo

- ¿Qué le ven de malo?

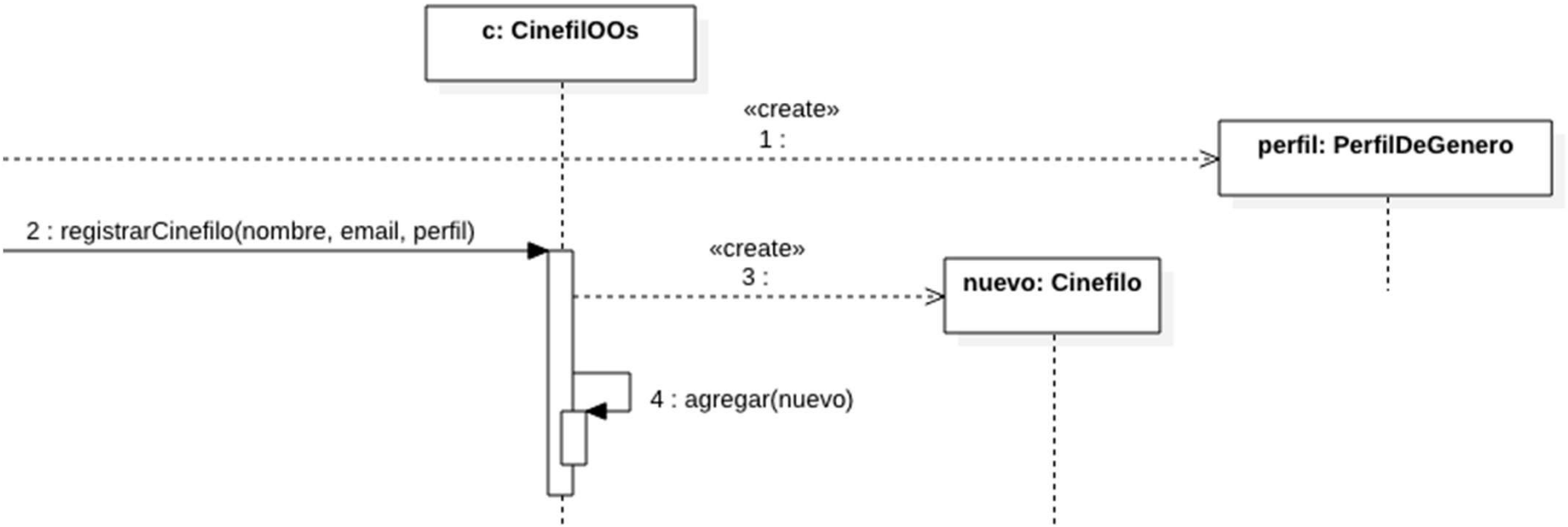


Leo, y releo la especificación para ver si aparece algún concepto del dominio que pueda modelar lo que veo en común (o lo que creo que quita cohesión a mis objetos)

Aplico composición para extraer los aspectos en común a un nuevo objeto

Ahora tengo un objeto más para delegar trabajo

Cargar película: Se ingresa título, la URL de la película en IMDB, y la URL de la imagen de portada de la película (también tomada de IMDB). Adicionalmente indica el "perfil de género" de la película. Cinefiloos (el sistema) registra la película.



Encontrar películas parecidas: Retornar todas las películas que son parecidas a una que se indique. Retorna todas las películas cuyo "índice de similitud" a la película indicada es menor a 6 (menor índice implica, más parecidas). La lista no está ordenada. La película indicada también está en la lista.

	horror	action	romance	suspense	comedy	sci-fi
Película 1	9	0	0	5	0	0
Película 2	0	0	0	9	0	7

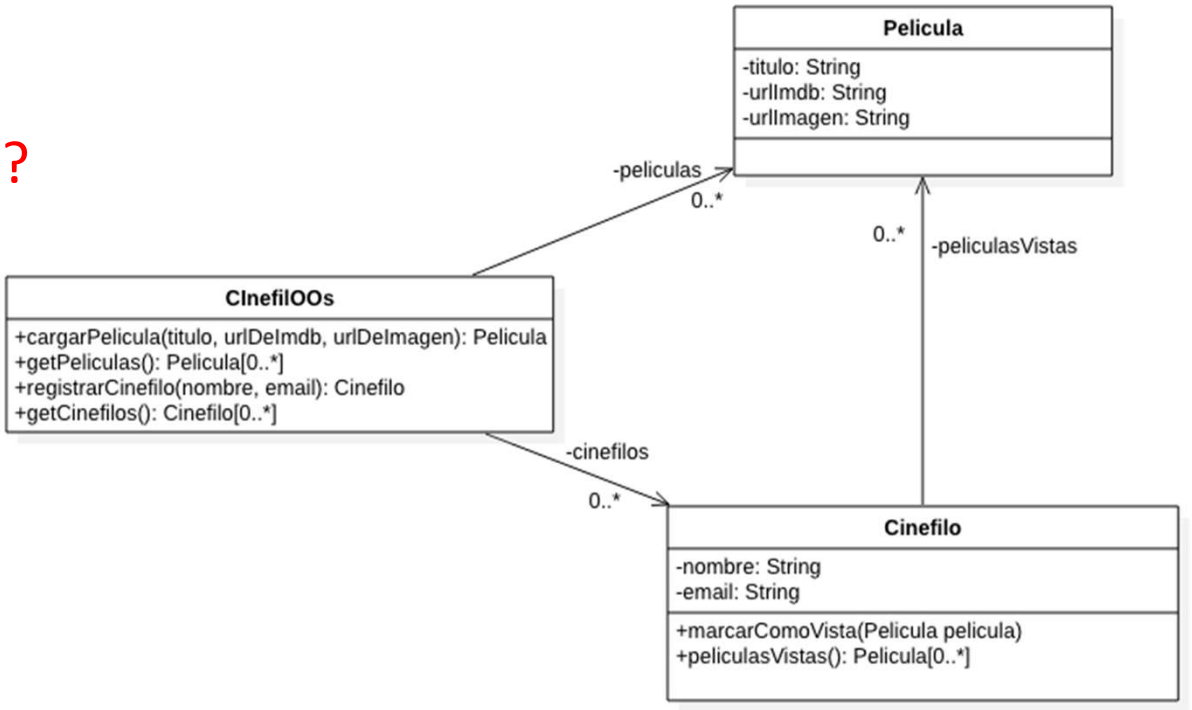
Índice de similitud (mejor significa más parecidos):

- por cada género calculo la diferencia entre las dos
- sumo los valores absolutos

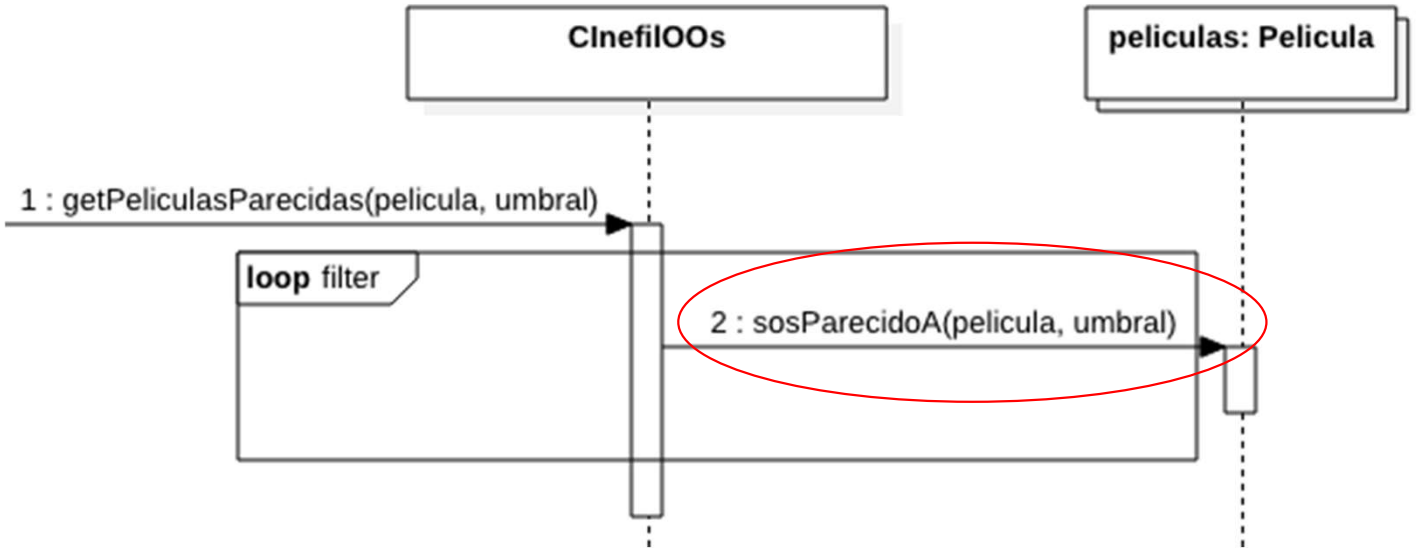
índice: abs(9-0)+0+0+abs(5-9)+0+abs(0-7) = 20

Encontrar películas parecidas: Retornar todas las películas que son parecidas a una que se indique. Retorna todas las películas cuyo "índice de similitud" a la película indicada es menor a 6 (menor índice implica, más parecidas). La lista no está ordenada. La película indicada también está en la lista.

¿A quién le preguntamos?

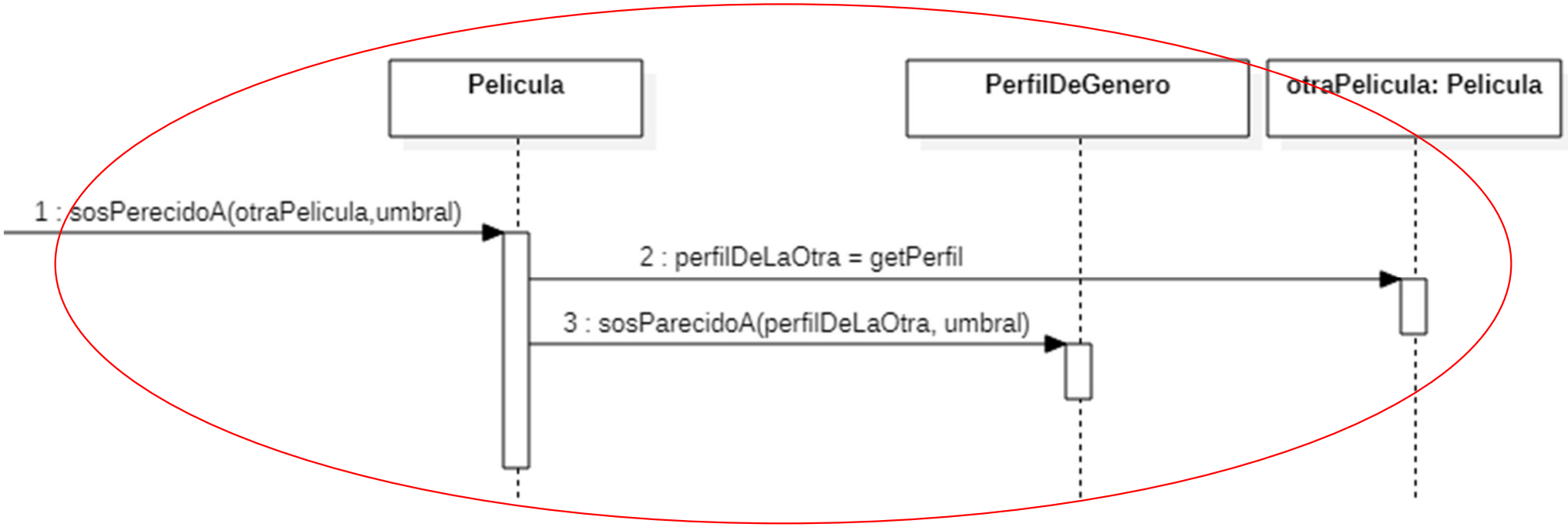


Paso 1: Delegar en las películas el cálculo del índice de similitude (ellas son quienes tienen la información - expertas)




```
public List<Pelicula> getPeliculasParecidas(Pelicula pelicula, double umbral) {
    return peliculas.stream().filter(peli -> peli.sosParecidoA(pelicula, umbral))
        .collect(Collectors.toList());
}
```

Paso 2: Delegar el cálculo de similitud entre perfiles a ellos



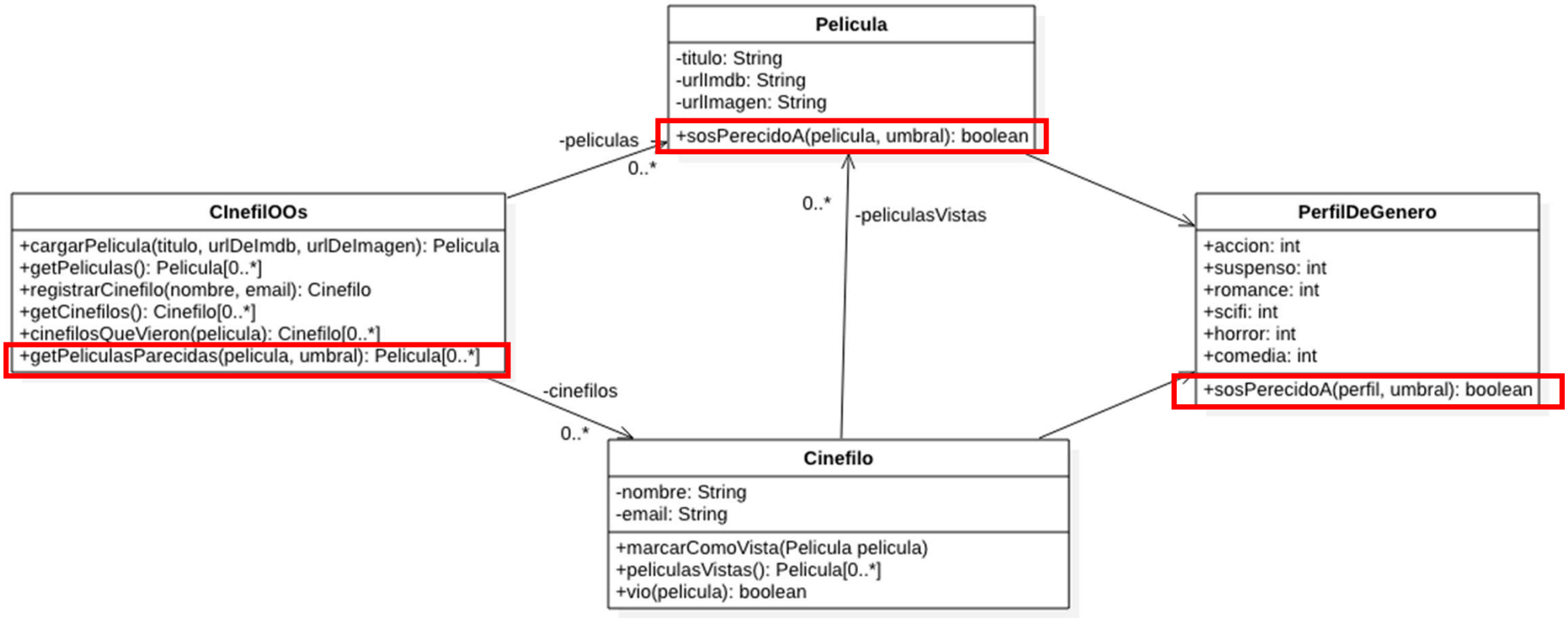

```
public boolean sosParecidoA(Pelicula pelicula, double umbral) {
    return pelicula.getPerfil().sosParecidoA(this.getPerfil(), umbral);
}
```

```java
public class PerfilDeGenero
{
    private double accion, suspenso, romance, scifi, horror, comedia;
```

```java
public boolean sosParecidoA(PerfilDeGenero otroPerfil, double umbral) {
    return this.getSimilitud(otroPerfil) <= umbral;
}

private double getSimilitud(PerfilDeGenero otroPerfil) {
    return Math.abs(accion - otroPerfil.getAccion()) +
        Math.abs(suspenso - otroPerfil.getSuspenso()) +
        Math.abs(romance - otroPerfil.getRomance()) +
        Math.abs(scifi - otroPerfil.getScifi()) +
        Math.abs(horror - otroPerfil.getHorror()) +
        Math.abs(comedia - otroPerfil.getComedia());
}
```

El comportamiento se distribuye – delegación …



Obtener la preferencia de genero de cine de un cinéfilo: Dado un cinéfilo retorna su preferencia de genero de cine (que tiene un número de 0 a 9 por cada género, al igual que las películas).

La preferencia de género de cine de un cinéfilo se calcula a partir de las películas que vio (asumimos que le gustaron).

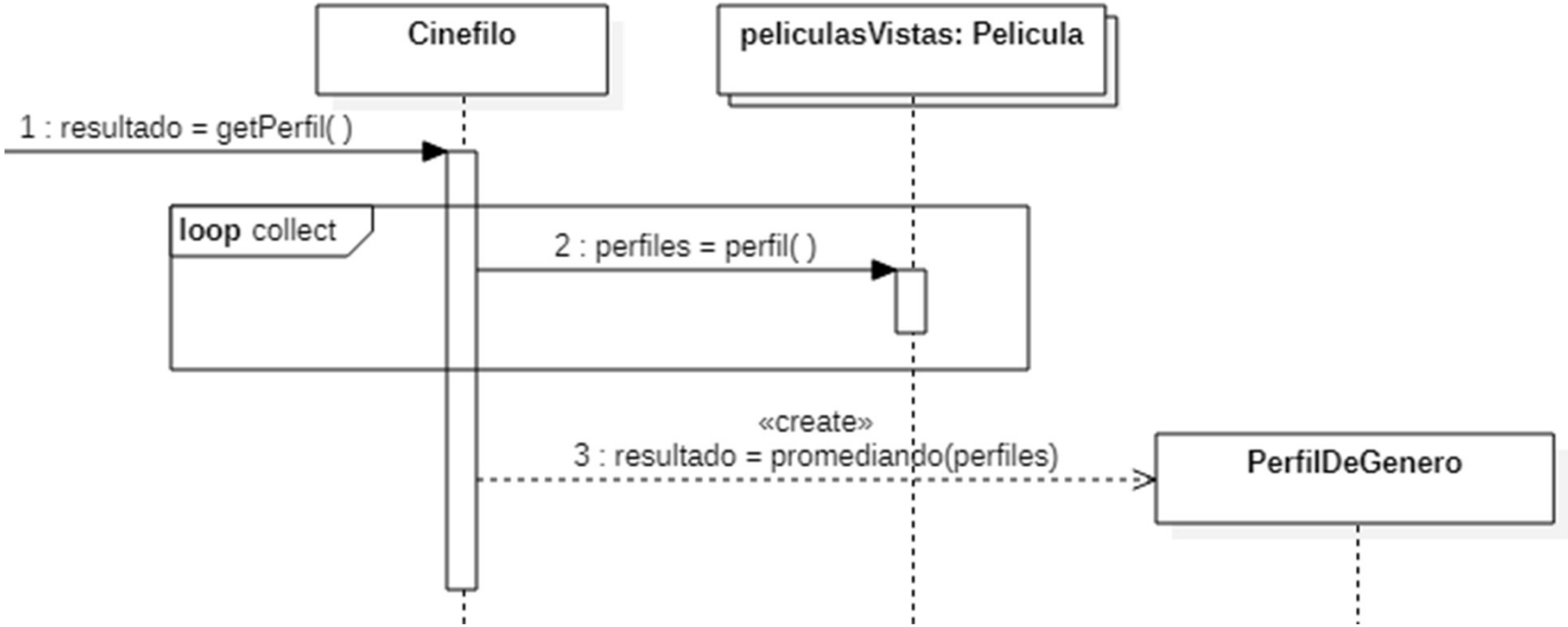
Si no vió ninguna, se toma 4.5 para todos los géneros (ni muy muy, ni tan tan). Luego, toma el promedio entre esa base y todas las películas que vea.

	horror	action	romance	suspense	comedy	sci-fi
Base	4.5	4.5	4.5	4.5	4.5	4.5
Película 1	9	0	0	5	0	0
Película 2	6	0	0	9	0	2
Promedio	6.16	1.5	1.5	6.1	1.5	2.1

getPerfil() en cinéfilo

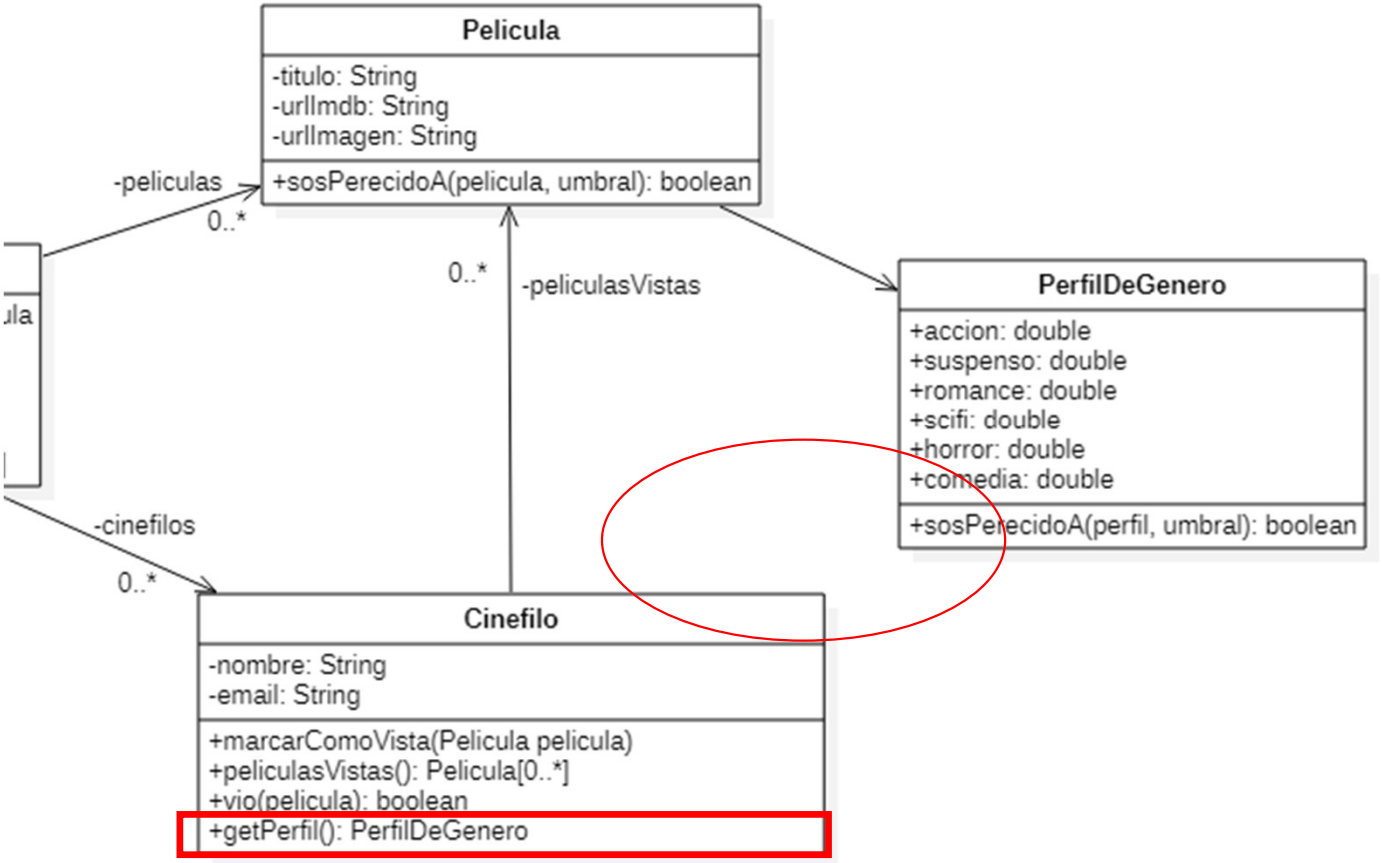
- Hay que calcular el promedio entre los perfiles de genero de las películas que vió y uno base
- Habría que crear una instancia nueva de PerfilDeGenero que sea promedio de todos esos
- Tenemos: el servicio, el cinéfilo, películas que vió, los perfiles de esas películas.

Paso 1: Calcular la preferencia de Genero



¿tenemos que guardarlo en una variable de instancia?

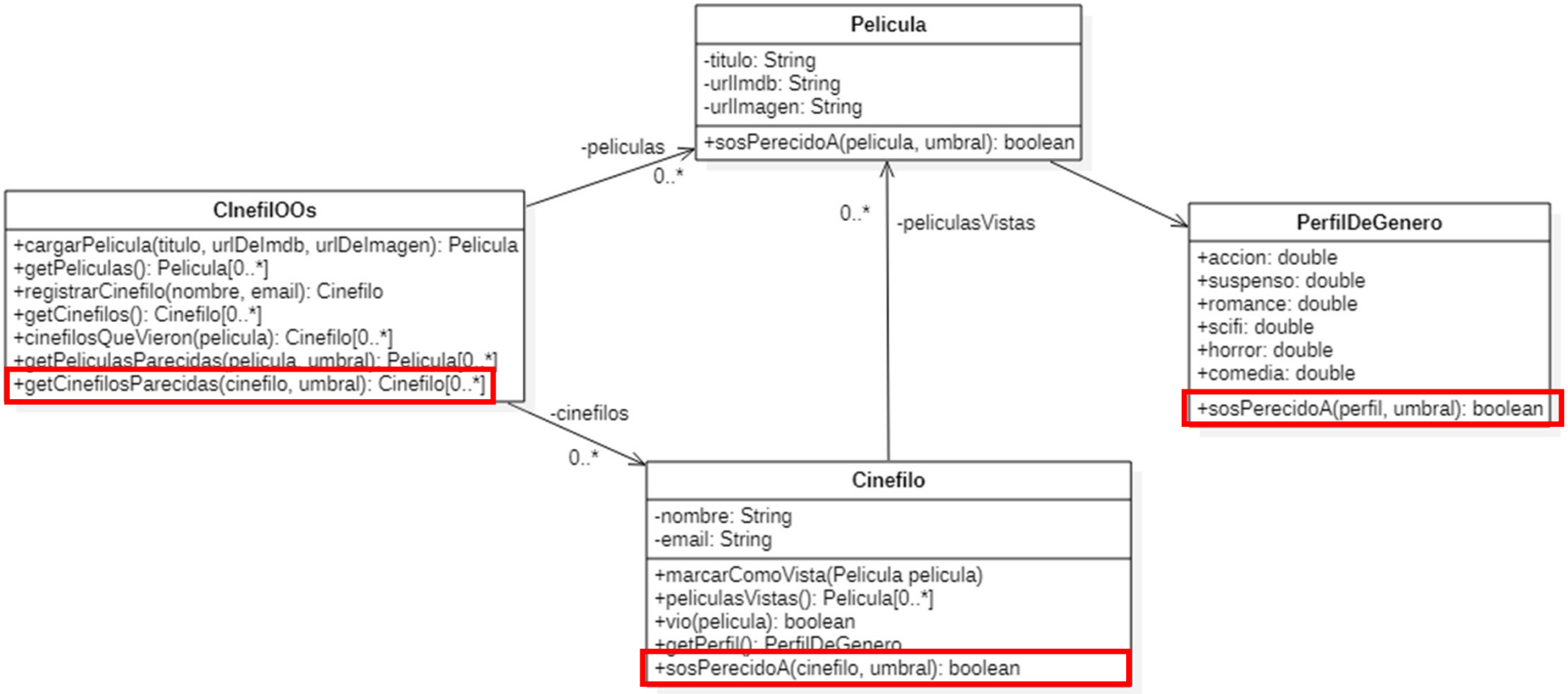
El comportamiento se distribuye – delegación …



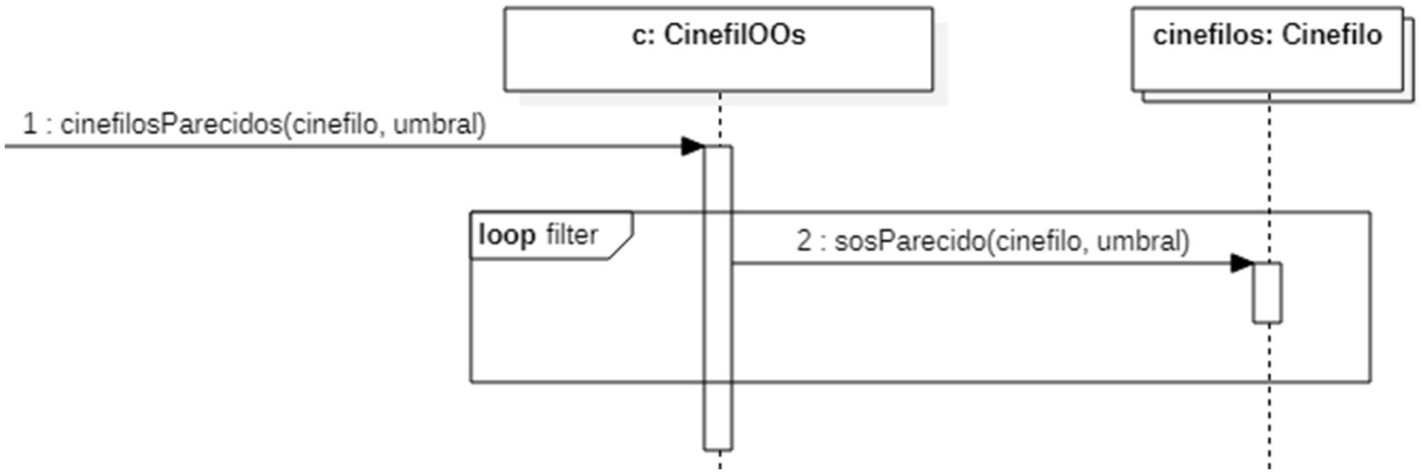
Encontrar cinéfilos con gustos similares: Dado un cinéfilo, encontrar todos aquellos que tienen preferencias de género parecidas. Para calcular la similitud entre dos cinéfilos, se toman las preferencias de género de cine de cada uno.

Retorna todos los cinéfilos cuyo "índice de similitud" al indicado es menor a 6 (menor índice significa más parecido). La lista no está ordenada. El cinéfilo está en la lista también.

El cálculo es idéntico al de similitud entre películas


Pelicula
-titulo: String
-urlImdb: String
-urlImagen: String
+sosPerecidoA(pelicula, umbral): boolean
-peliculas
0..*
CInefilOOs
+cargarPelicula(titulo, urlDeImdb, urlDeImagen): Pelicula
+getPeliculas(): Pelicula[0..*]
+registrarCinefilo(nombre, email): Cinefilo
+getCinefilos(): Cinefilo[0..*]
+cinefilosQueVieron(pelicula): Cinefilo[0..*]
+getPeliculasParecidas(pelicula, umbral): Pelicula[0..*]
+getCinefilosParecidas(cinefilo, umbral): Cinefilo[0..*]
0..*
-peliculasVistas
PerfilDeGenero
+accion: double
+suspenso: double
+romance: double
+scifi: double
+horror: double
+comedia: double
+sosPerecidoA(perfil, umbral): boolean
-cinefilos
0..*
Cinefilo
-nombre: String
-email: String
+marcarComoVista(Pelicula pelicula)
+peliculasVistas(): Pelicula[0..*]
+vio(pelicula): boolean
+getPerfil(): PerfilDeGenero
+sosPerecidoA(cinefilo, umbral): boolean

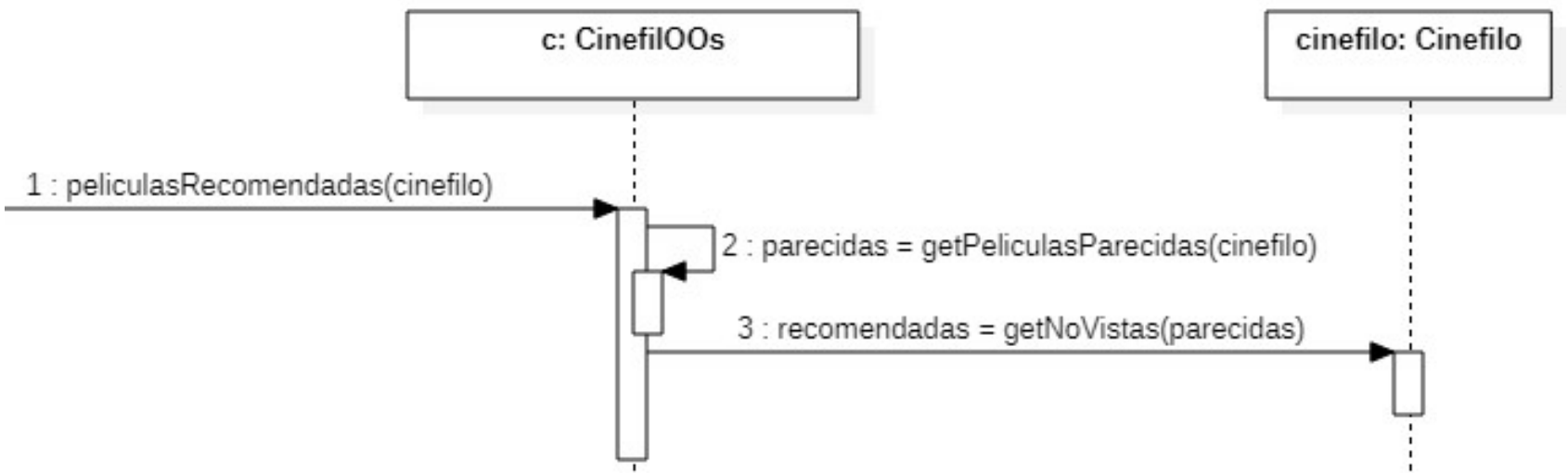
```java
public List<Cinefilo> getCinefilosParecidos(Cinefilo cinefilo, double umbral) {
    return cinefilos.stream().filter(cf -> cf.sosParecidoA(cinefilo, umbral))
        .collect(Collectors.toList());
}
```




```java
public boolean sosParecidoA(Cinefilo otroCinefilo, double umbral) {
    return otroCinefilo.getPerfil().sosParecidoA(this.getPerfil(), umbral);
}
```

Encontrar películas que pueden gustar, no vistas: Dado un cinéfilo, retorna todas las películas cuyo "índice de similitud" a la preferencia del cinéfilo es menor a 5, y que el cinéfilo no vio. La lista no está ordenada.

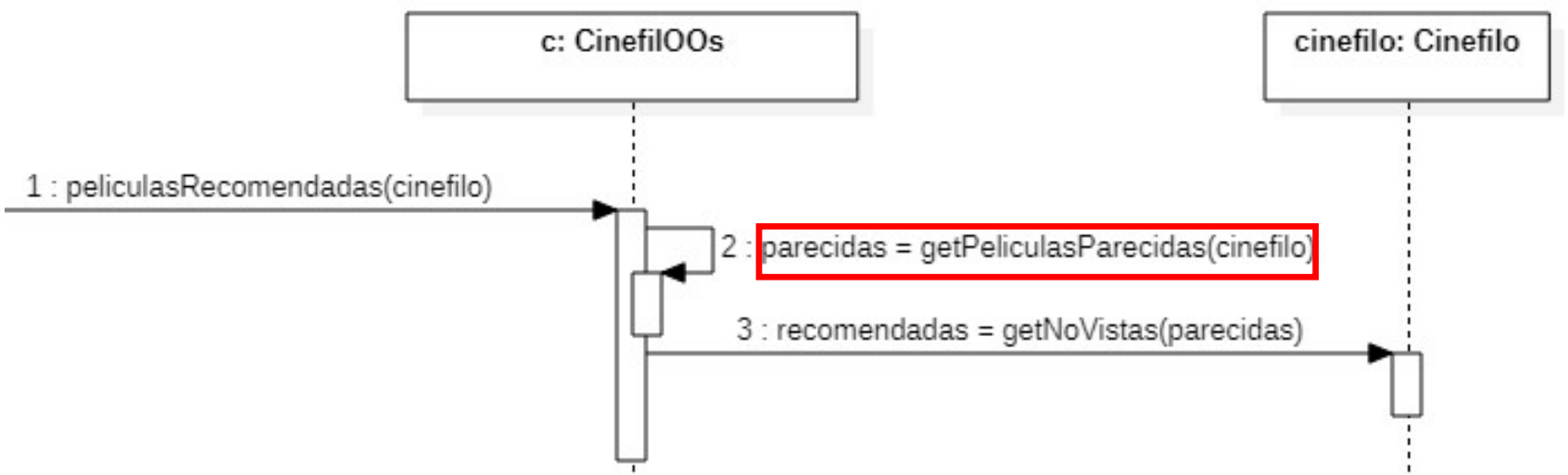
El cálculo es idéntico al de similitud entre películas.



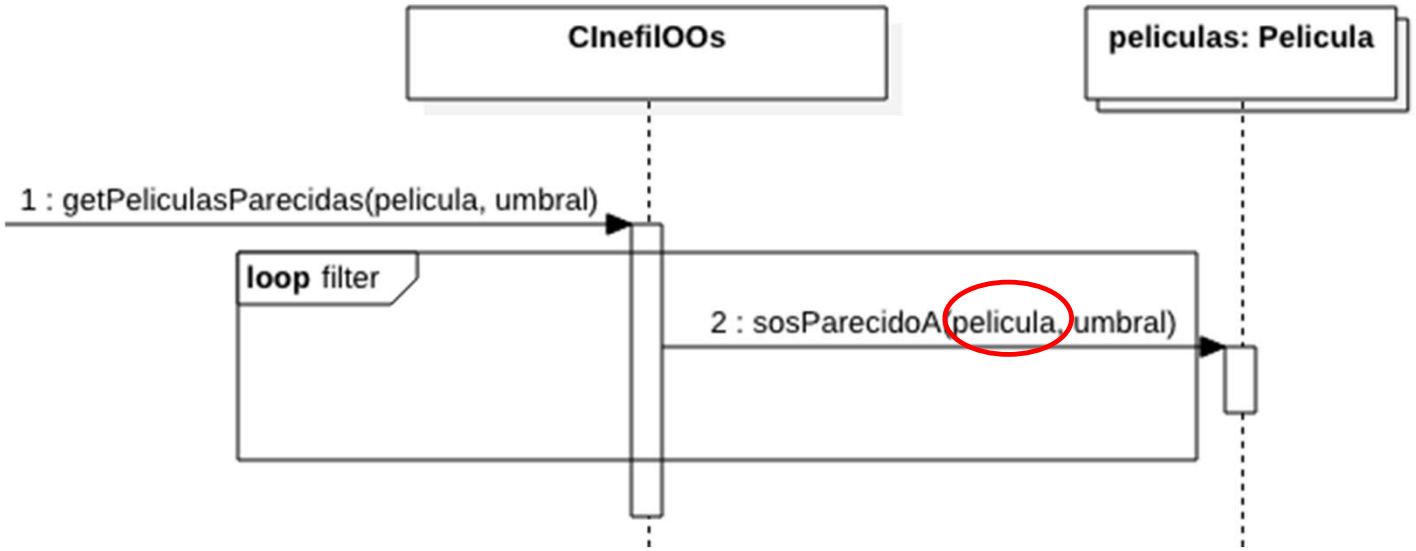
Encontrar películas que pueden gustar, no vistas: Dado un cinéfilo, retorna todas las películas cuyo "índice de similitud" a la preferencia del cinéfilo es menor a 5, y que el cinéfilo no vio. La lista no está ordenada.

El cálculo es idéntico al de similitud entre películas.

¡Necesitamos ver si una película es parecida a un cinéfilo!

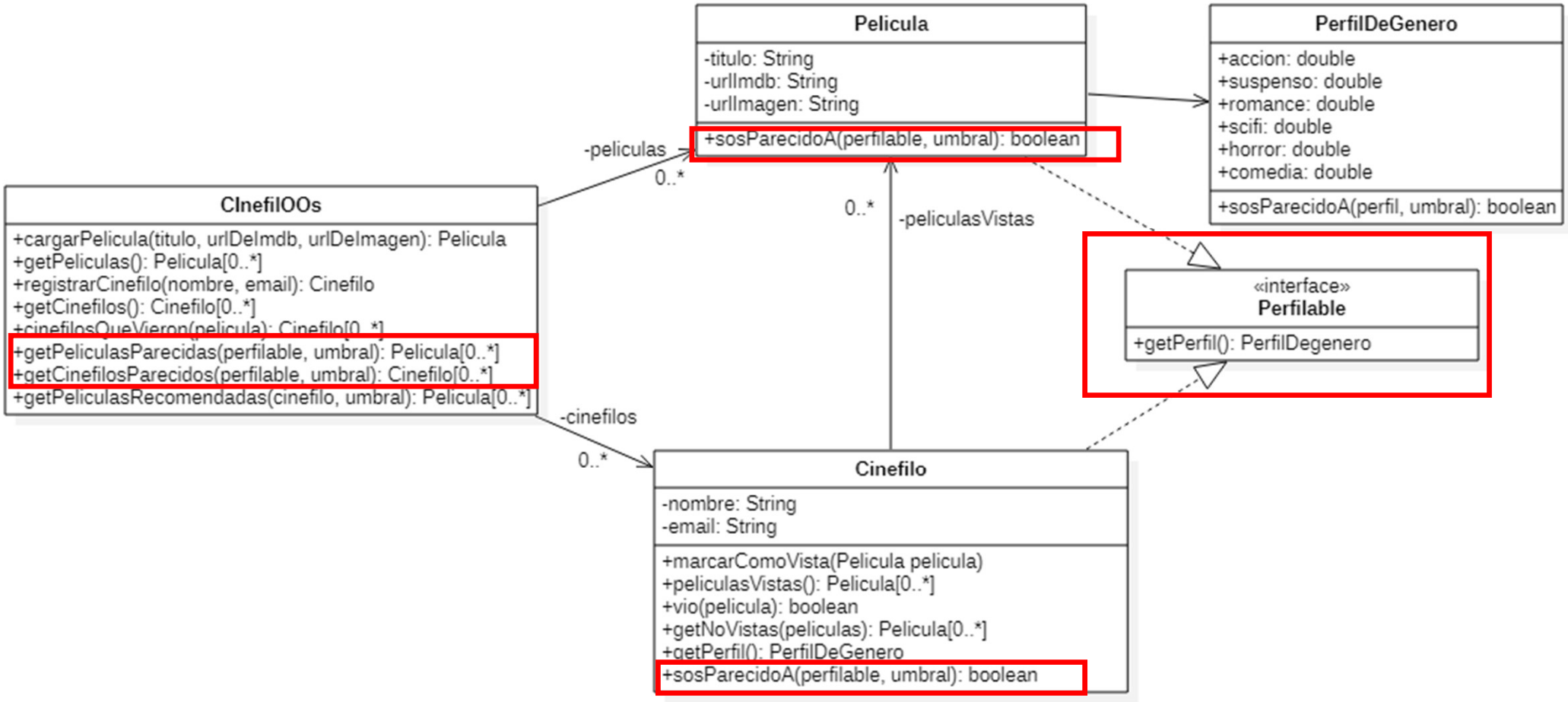


¡Necesitamos ver si una película es parecida a un cinéfilo!

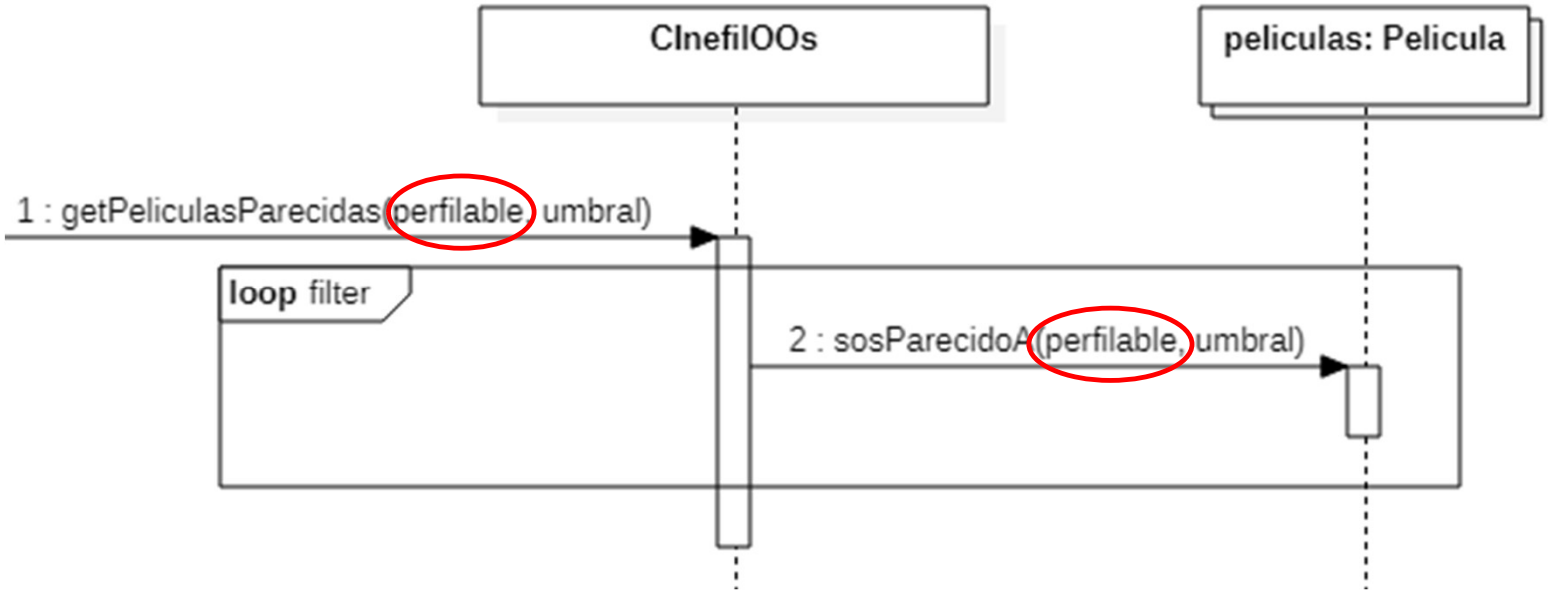



```
public boolean sosParecidoA(Pelicula pelicula, double umbral) {
    return pelicula.getPerfil().sosParecidoA(this.getPerfil(), umbral);
}
```

La interfaz Perfilable captura la ida de tener un perfil de género

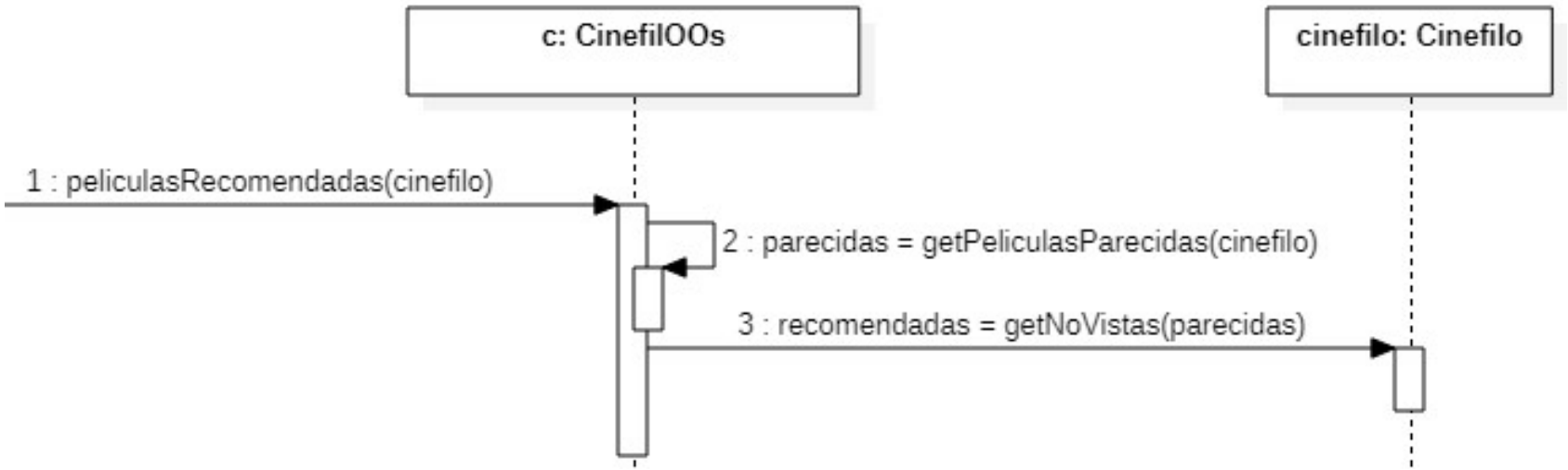


La interfaz Perfilable captura la ida de tener un perfil de género



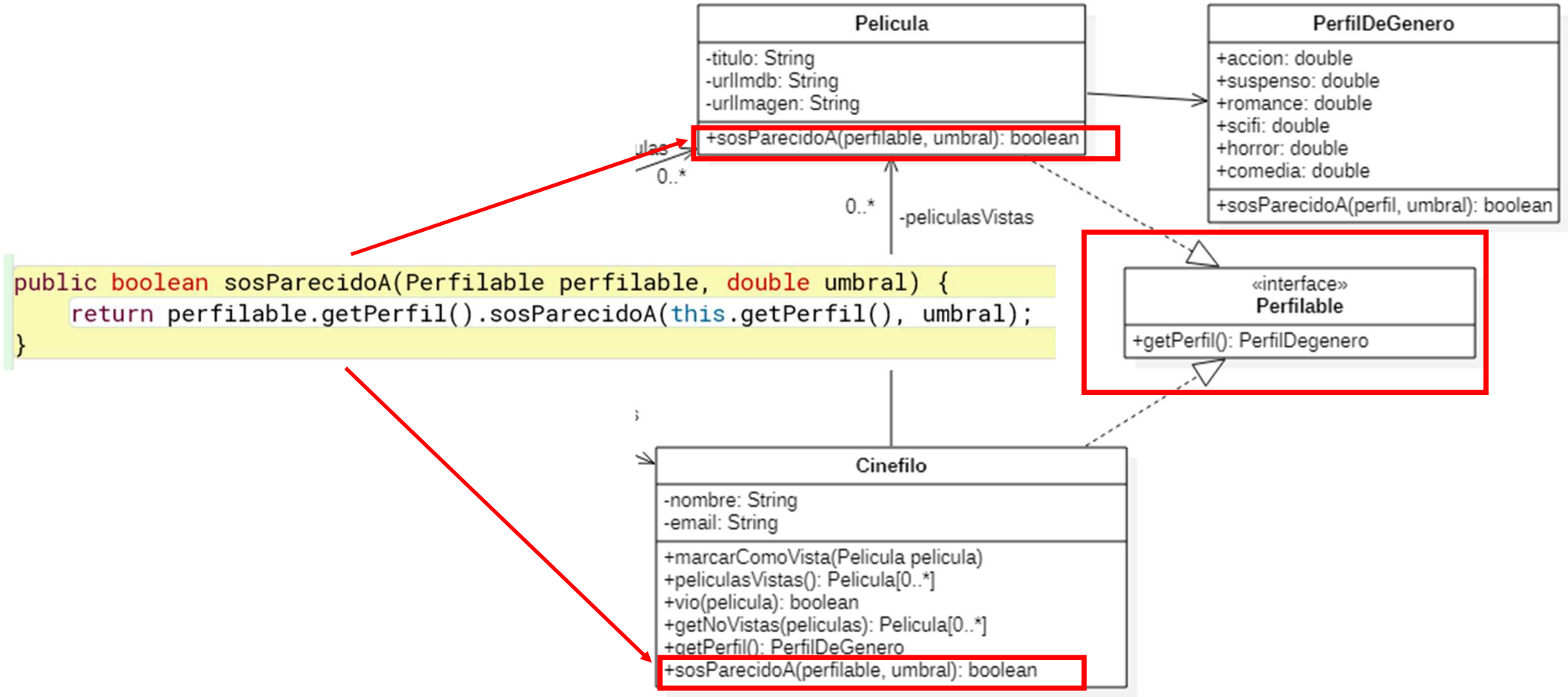

```
public boolean sosParecidoA(Perfilable perfilable, double umbral) {
    return perfilable.getPerfil().sosParecidoA(this.getPerfil(), umbral);
}
```

```java
public List<Pelicula> getPeliculasRecomendadas(Cinefilo cinefilo, double umbral) {
    List<Pelicula> peliculasParecidas = this.getPeliculasParecidas(cinefilo, umbral);
    return cinefilo.getNoVistas(peliculasParecidas);
}
```

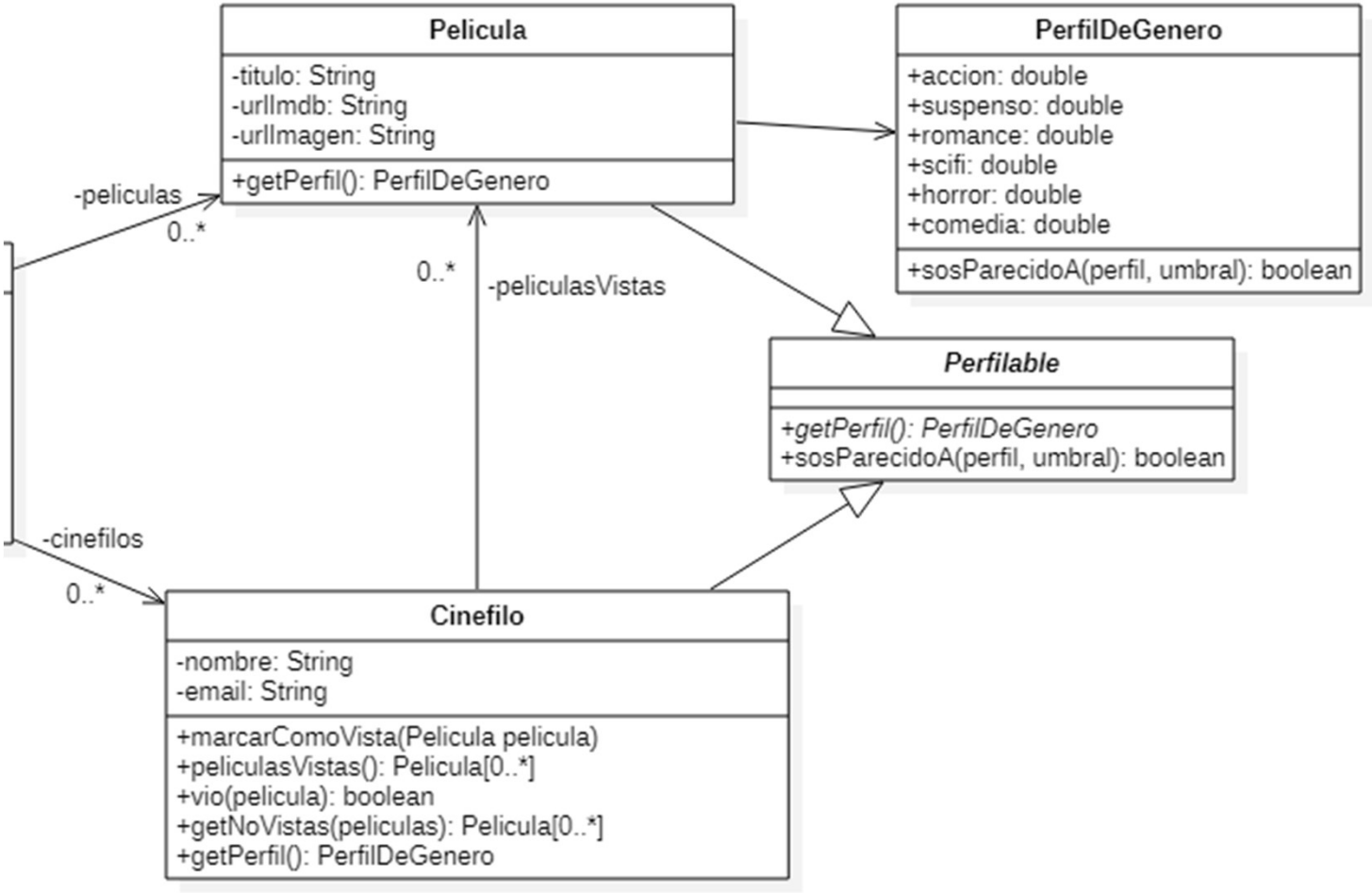



```java
public List<Pelicula> getNoVistas(List<Pelicula> peliculas) {
    return peliculas.stream().filter(peli -> !this.vio(peli))
        .collect(Collectors.toList());
}
```

¿Perfilable es Interfaz o clase abstracta?



¿Perfilable es Interfaz o clase abstracta?



Algunas conclusiones

- Nuestras clases salen del lenguaje de los requerimientos (lenguaje del dominio)
- No todos los objetos aparecen en nuestra primera versión del modelo
- El diseño evoluciona a medida que agregamos funcionalidad y entendemos el dominio
- “delegar” es el criterio principal para un buen diseño
  - Nuestro objeto CinefilOOs (el servicio/sistema) quedó con bastante comportamiento porque nuestros requerimientos tenían que ver mucho con buscar en las colecciones (no siempre es el caso)
  - No todo pasa por ese objeto CinéfilOOs
- En un par de clases vamos a aprender heurísticas que nos apoyan en el proceso de diseño