

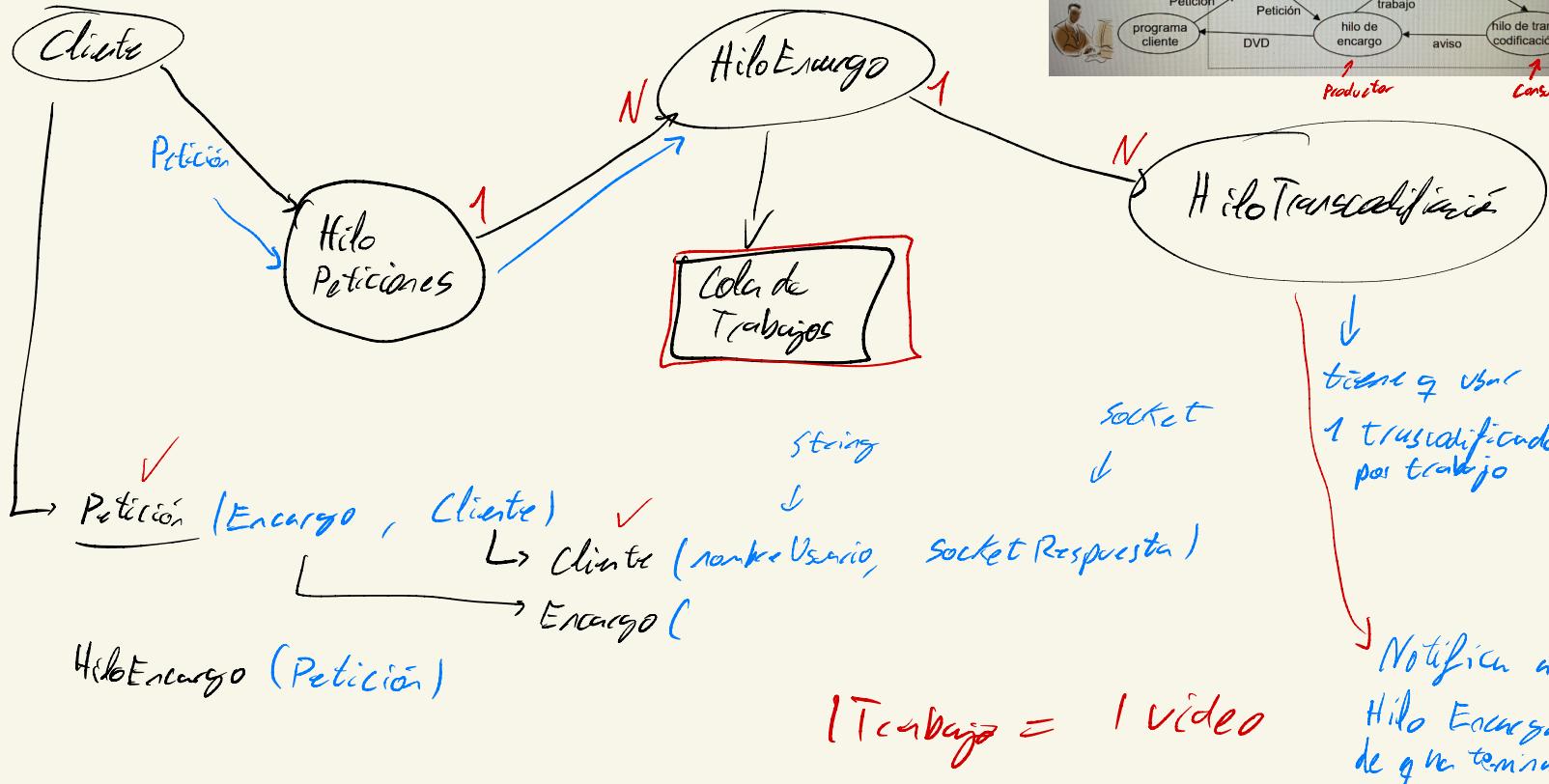

En principal = Hilo Peticiones

ReceptorPeticiones o

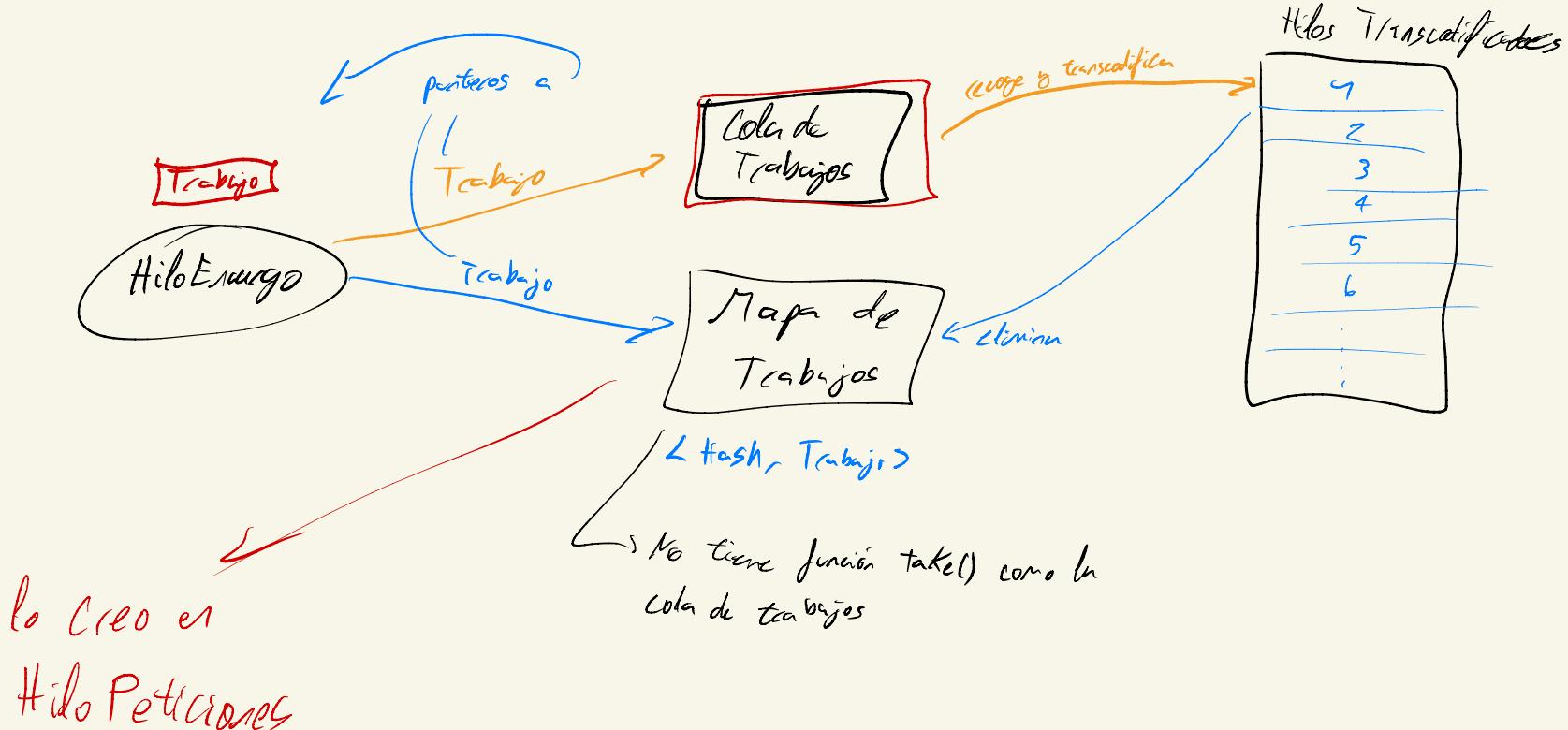
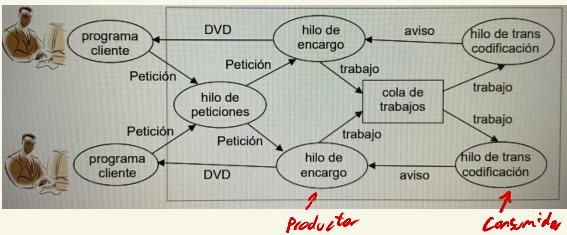
while do

devuelve Petición
T y la
o.recibir Petición ();
paso a
Hilo Encargo

cola de trabajo en puente visualizador



FASE 4



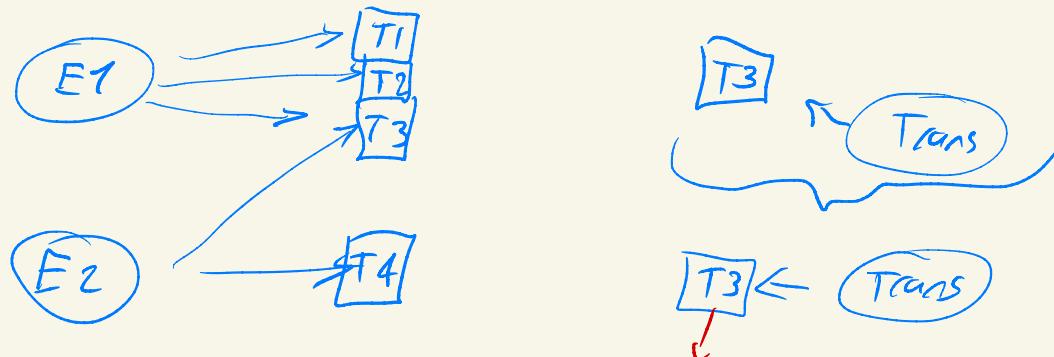
HILOENCARGO

1. El HiloEncargo lo primero que hará es meter el Trabajo en el MAPA de trabajos y comprobar si ese trabajo ya existe, si es así no lo mete en el ARRAY de trabajos
2. Si el trabajo no existe, lo mete en el ARRAY de trabajos y a continuación en el MAPA de trabajos

HIOTRANSCODIFICADOR

1. El HIOTRANSCODIFICADOR lo que hará es sacar vídeos del ARRAY, y tras haberlos transcodificado, lo eliminará del MAPA de trabajos (usando remove())
2. El HIOTRANSCODIFICADOR deberá recorrer el array de semáforos que tiene el trabajo para ir desbloqueando a cada uno de los hilosEncargo

Aviso de los Hilos Transcodificadores a los Hilos Encargo



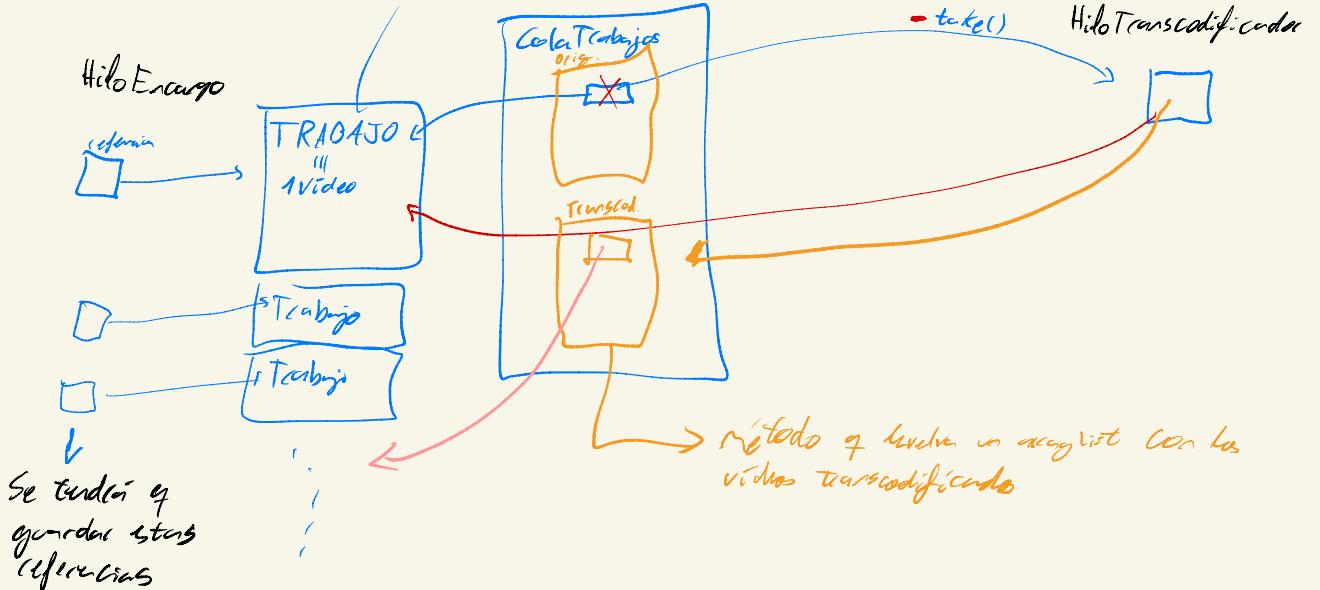
en lugar de tener 1 semáforo de 1 solo largo, tendrá una Lista de Semáforos.

* el método para meter su semáforo en la lista de semáforos del trabajo debe ser Synchronized

{ El hiloEncargo si ve en el MAPA q existe, lo q hará a continuación es meter en ese trabajo su semáforo

El hiloTranscodificador en lugar de hacer un único release(), recorrerá la lista de semáforos haciendo release()

meter al Trabajo en la cola es meter en la cola una referencia al objeto Trabajo

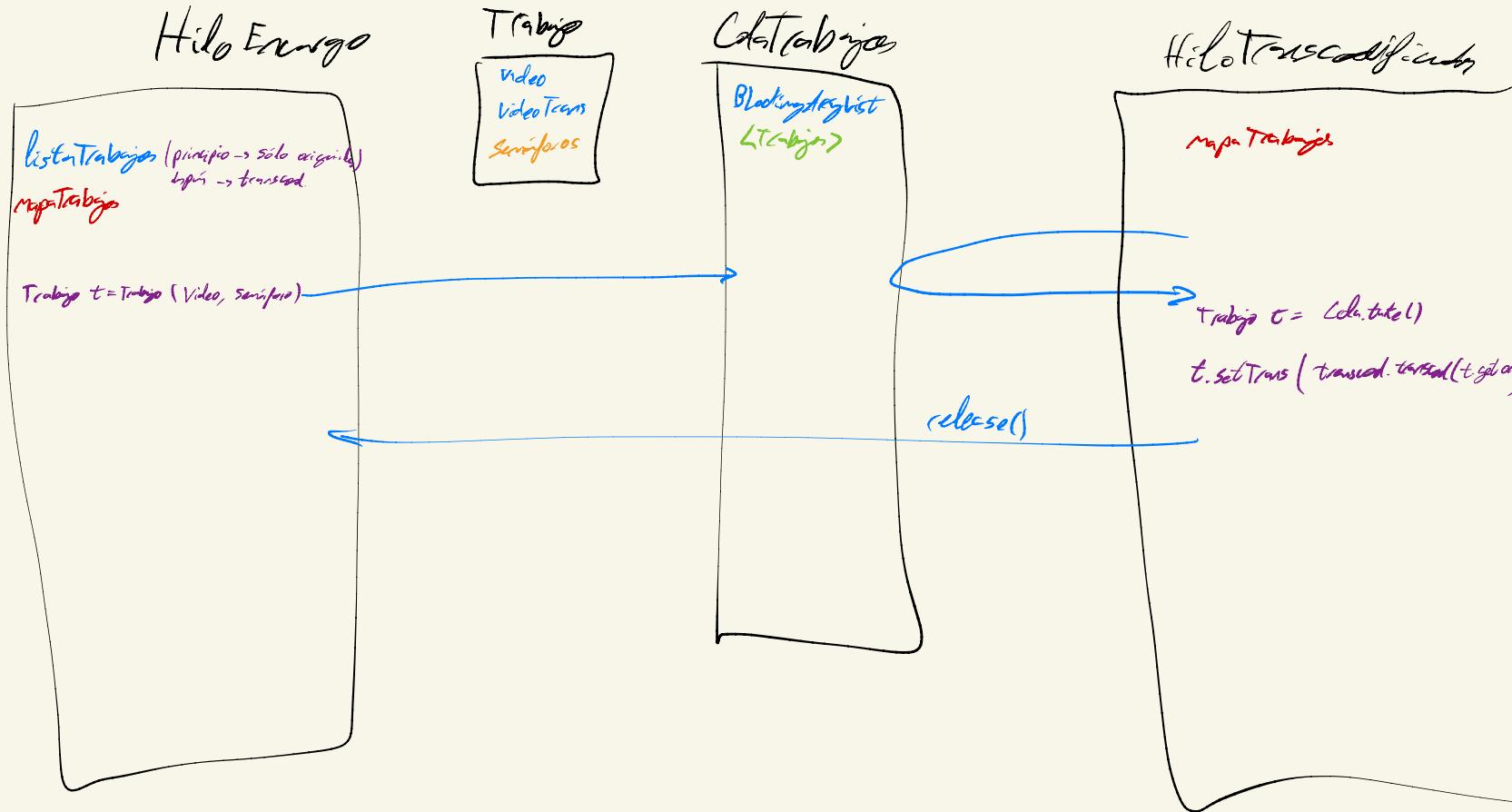


Cola de trabajos → Se puede usar sus métodos desde el main
↳ encolar y desencolar

— Nuevo

Lo que voy a hacer para controlar los HilosTranscodificadores es:

1. Una executorService haciendo una pool donde sólo quepan MAX_TRANSCODIFICADORES que puedan ser ejecutados
2. Hacerlo sin una lista, donde cada HiloEncargo crea sus propios TRANSCODIFICADORES, controlando el número máximo de éstos con un AtomicInteger compartido entre los HilosEncargo



La no entrega del código de la práctica en tiempo y forma conlleva una calificación de cero puntos en la parte de laboratorio del bloque 2. Además, se realizarán análisis de plagio de las entregas y se aplicará la [Normativa de evaluación del aprendizaje en las titulaciones oficiales de Grado y Máster Universitario de la Universidad Politécnica de Madrid](#) que establece, entre otras cosas, que "Ante la comprobación de fraude académico en una prueba de evaluación, se calificará con la puntuación de cero al estudiante o estudiantes implicados en la calificación final de la convocatoria correspondiente a la celebración de la prueba (ordinaria o extraordinaria)."

4 Enunciado

Una empresa del sector audiovisual cuenta con una amplia colección de vídeos de muchos pebibytes. Estos vídeos están almacenados en un servidor central (al que llamaremos en adelante *servidor de vídeos*), con unas ciertas características de codificación, tamaño de imagen, etc. Los trabajadores de la empresa (a los que llamaremos en adelante *usuarios*) pueden solicitar, mediante una cierta herramienta, que el servidor de vídeos produzca un DVD en formato ISO 9660 con un grupo de vídeos procesados de una cierta forma.

Para ofrecer este servicio, se ha diseñado un sistema que sigue el esquema cliente-servidor. En el servidor de vídeos se ejecutará el *programa servidor* (a desarrollar por el alumno), y los *programas clientes* (que se proporcionan ya codificados) se ejecutarán en las estaciones de trabajo de los usuarios. Cuando un usuario necesita que se produzca un determinado DVD, el programa cliente generará una *peticIÓN* que contendrá el encargo a realizar (lista de vídeos que compondrán el DVD + información adicional) y el *cliente* que lo solicita (información de la conexión por la que llega la petición, y que servirá para que el programa servidor envíe el DVD al programa cliente que lo ha solicitado).

El servidor de vídeos es una máquina con varias CPU (en adelante diremos que tiene N CPU) y utiliza un programa multihilo para realizar el procesamiento, con la idea de aprovechar todo el potencial de las N CPU. La arquitectura y funcionamiento del programa servidor es la siguiente:

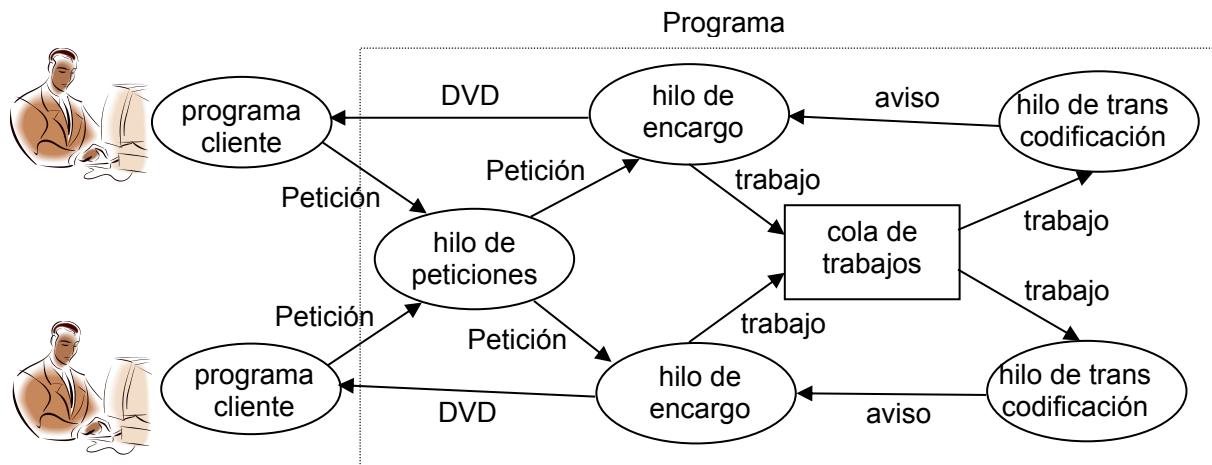
- Hay un hilo (al que llamaremos en adelante ***hilo de peticIONes***), cuya función es recibir las *peticIONes* con los *encargos* de los usuarios. Su funcionamiento esquemático es un bucle sin fin en el que realiza lo siguiente:
 - ~~p1.~~ Recibir una petición con un encargo de un usuario.
 - ~~p2.~~ Crear y lanzar un *hilo de encargo* para procesar el encargo que viene en la petición.
- Hay un tipo de hilo al que llamaremos ***hilo de encargo*** cuya función es coordinar el procesamiento de un encargo. No hay una cantidad predefinida de *hilos de encargo*, sino que son creados por el *hilo de peticIONes* según se necesitan (un nuevo hilo por cada petición recibida). Su funcionamiento esquemático es el siguiente:
 - ~~e1.~~ Analizar los detalles del encargo y generar tantos *trabajos de transcodificación* como vídeos se hayan solicitado en el encargo.
 - ~~e2.~~ Meter todos los trabajos de transcodificación generados en una cola para que sean procesados por los *hilos de transcodificación*. Llamaremos *cola de trabajos* a esta cola.
 - ~~e3.~~ Esperar a que sean completados todos los trabajos de transcodificación que ha generado.
 - ~~e4.~~ Generar el DVD con los vídeos transcodificados y añadir el menú raíz y otras características comunes al DVD.
 - ~~e5.~~ Enviar el DVD resultante al cliente que envió la petición. Después este hilo finaliza su ejecución.
- Hay un número fijo (N-1) de hilos que procesan los trabajos de transcodificación a los que llamaremos ***hilos de transcodificación***. Su funcionamiento esquemático es un bucle sin fin en el que realizan lo siguiente:

Extraer un trabajo de transcodificación de la cola de trabajos.

Procesar el trabajo de transcodificación.

73. Notificar la terminación del trabajo de transcodificación al *hilo de encargo* que lo solicitó.

La siguiente figura muestra el diagrama que relaciona los diferentes tipos de hilos de los que se compone la aplicación, y los objetos que intercambian entre ellos.



Cabe esperar que, en ocasiones, varios encargos concurrentes hagan referencia al mismo vídeo. En ese caso, y con el fin de mejorar el rendimiento del sistema, el servidor no debería transcodificar el mismo video varias veces, sino transcodificarlo una sola vez y aprovecharlo en todos los encargos abiertos en ese momento. Para ello, se propone que, cuando hay un trabajo de transcodificación pendiente para un determinado vídeo, si llega una petición con un encargo que incluye ese vídeo, no se encole otro trabajo de transcodificación para ese mismo vídeo. Eso sí, cuando termine el trabajo de transcodificación del vídeo, todos los hilos de encargo que estuvieran esperando que terminara el trabajo deberán ser notificados, y todos ellos deberán poder recoger el vídeo para componer el DVD y enviarlo a su cliente.

5 Fases de realización de la práctica

Para alcanzar plenamente los objetivos finales de la práctica, se deben seguir las siguientes fases en su realización. De esta forma se irán añadiendo nuevas funcionalidades de manera progresiva y se podrá ir comprobando su evolución.

5.1 Fase 1

Cree la clase **HiloEncargo** que será la que implemente los hilos de encargo. De momento, estos hilos se limitarán a escribir por pantalla los datos del encargo que viene en la petición (nº de videos del encargo y nombre del usuario que lo ha solicitado), esperar 30 segundos, escribir un mensaje de despedida y terminar su ejecución.

Desarrolle un programa principal que implemente el **hilo de peticiones** (bucle con los pasos p1 y p2). Observe que no es necesario crear un hilo nuevo para el hilo de peticiones, sino que el hilo de peticiones puede ser el hilo principal del programa. No obstante, si lo prefiere, también puede hacerlo en un hilo nuevo.

Utilice los **programas clientes** que se proporcionan para enviar varias peticiones al servidor, y compruebe que van apareciendo hilos de encargo nuevos para cada uno de ellos, mostrando por pantalla el contenido de la petición.

5.2 Fase 2

Modifique el programa para que cree la **cola de trabajos**. La cola de trabajos deberá implementar la interfaz `ssoo.videos.Cola`. Además, deberá permitir encolar y desencolar trabajos de forma concurrente.

Cree la clase **HiloTranscodificador** que implementará los hilos de transcodificación. Tenga en cuenta que estos hilos necesitarán extraer trabajos de la cola de trabajos. De momento, estos hilos se limitarán, en su bucle principal, a extraer trabajos de la cola de trabajos (paso t1) y a procesarlos (paso t2).

Modifique el programa para que cree $N-1$ hilos de transcodificación antes de meterse en el bucle que recibe peticiones de los clientes y crea los hilos de encargo correspondientes. Recuerde que los hilos de transcodificación necesitarán extraer trabajos de la cola de trabajos que se ha creado en el programa principal (o en su defecto en el hilo de peticiones).

Modifique el programa para que, al crear los hilos de encargo, tenga en cuenta que estos hilos necesitarán acceder a la cola de trabajos y que necesitarán enviar el DVD al cliente que lo solicitó. Recuerde que, si ha optado por un hilo de peticiones específico, este hilo será el encargado de crear los hilos de encargo.

Modifique la clase **HiloEncargo** para que los hilos de encargo que se creen generen los trabajos de transcodificación correspondientes al encargo que estén procesando y los metan en la cola de trabajos (pasos e1, e2).

Utilice los clientes de prueba que se proporcionan para enviar varias peticiones de encargos al servidor y compruebe que van apareciendo hilos de encargo nuevos para cada una de ellas.

5.3 Fase 3

Modifique la clase **HiloTranscodificador** para que, al terminar la transcodificación del vídeo, el hilo de transcodificación ponga el vídeo transcodificado a disposición del hilo de encargo que lo solicitó y le notifique que éste ha terminado (paso t3).

Modifique la clase **HiloEncargo** para que los hilos de encargo esperen a que finalicen todos los trabajos de transcodificación que han generado y, después, obtengan los vídeos transcodificados y realicen los pasos finales de generación y envío del DVD (pasos e3, e4, e5).

Si no lo ha hecho antes, registre la cola de trabajos en el panel visualizador, tal como se indica al final del Apartado 6 de este enunciado.

5.4 Fase 4

Modifique las clases necesarias para conseguir que, si un hilo de encargo necesita un vídeo para el que ya hay un trabajo de transcodificación encolado, no encole uno nuevo, sino que anote que él también lo necesita. Una vez terminado el trabajo de transcodificación, habrá que informar de ello a todos los hilos de encargo que lo solicitaron.

Note que, en esta fase, ya no es suficiente con que la cola permita encolar y desencolar trabajos, sino que, por ejemplo, debe permitir buscar si hay algún trabajo para un determinado vídeo y modificar dicho trabajo.

6 Material que se entrega

Se proporcionan, en [Moodle](#), las clases de Java necesarias para desarrollar la práctica (fichero `p3videos.jar`) y la documentación relativa a las mismas (fichero `javadoc-videos.zip`). Consulte también la referencia [cómo integrar un paquete JAR en un proyecto de Eclipse](#).

Lea atentamente la documentación correspondiente a estas clases, puesto que en este apartado se da únicamente una descripción somera de sus funcionalidades.

- `ssoo.videos.servidor.ReceptorPeticiones`: permite recibir peticiones enviadas desde los clientes. Lo usará el hilo de peticiones.
- `ssoo.videos.servidor.Petition`: representa a una petición, compuesta por un Encargo y el Cliente que lo ha efectuado.
- `ssoo.videos.servidor.Cliente`: representa al cliente que ha enviado una petición. Se utiliza para enviar un DVD al cliente que lo solicitó.
- `ssoo.videos.Encargo`: representa a un encargo. Contiene, entre otras cosas, la lista de vídeos que compondrá el DVD solicitado.
- `ssoo.videos.Video`: representa a un vídeo. El contenido del vídeo es ficticio, aunque su nombre lo identifica únicamente.
- `ssoo.videos.Transcodificador`: permite transcodificar vídeos. El procesamiento que hace es ficticio. Cada hilo de transcodificación usará un Transcodificador para procesar los vídeos.
- `ssoo.videos.Dvd`: representa a un DVD, con un título, un menú raíz, y una serie de pistas de vídeo. El contenido y procesamiento del DVD son ficticios.
- `ssoo.videos.MenuRaiz`: representa al menú raíz de un DVD. El contenido es ficticio.
- `ssoo.videos.Cola`: interfaz que debe implementar la cola de trabajos para que el panel visualizador pueda mostrar cuántos trabajos hay encolados en cada momento.

Además, se entregan dos clientes de pruebas:

- `ssoo.videos.clientes.ClienteInteractivo`: permite la introducción interactiva de los datos del nombre del usuario y el encargo que realiza. Una vez que disponga de estos datos, creará una petición, la enviará al servidor y esperará a recibir el DVD.
- `ssoo.videos.clientes.ClienteAutomatizado`: genera automáticamente una serie de encargos a partir de los argumentos indicados en la línea de órdenes, o mediante los datos solicitados interactivamente por el programa (e introducidos por el usuario), envía al servidor las peticiones correspondientes y espera la recepción de los DVD.

Finalmente, se entrega una clase que servirá para monitorizar el funcionamiento del sistema:

- `ssoo.videos.PanelVisualizador`: permite mostrar gráficamente alguna información acerca del estado de ciertos elementos del servidor de vídeos, con el objetivo de proporcionar una comprensión mejor de cómo una aplicación multihilo puede realizar múltiples tareas concurrentemente. Esta clase es usada por los objetos de tipo `ssoo.videos.Transcodificador` y `ssoo.videos.servidor.Cliente` para visualizar el progreso de las operaciones que realiza el servidor. Para que el panel visualizador pueda mostrar el número de trabajos encolados, se le deberá proporcionar la cola de trabajos (que implemente la interfaz `ssoo.videos.Cola`) por medio del método `PanelVisualizador.getInstancia().registrarColaPeticiones()`.

7 Resolución de situaciones de concurrencia

Se deben analizar y dar una solución a todas las situaciones de concurrencia que se plantean en esta práctica. Algunas de estas situaciones se comentan a continuación. Sin embargo, dependiendo de cómo se aborde la realización de la práctica, podrían aparecer más. Las soluciones que se adopten para resolver estos problemas deben satisfacer el objetivo de maximizar el nivel de concurrencia.

- Los hilos de encargo deben esperar (de manera inactiva) a que todos los vídeos que necesita para generar el DVD correspondiente al encargo que está procesando hayan sido transcodificados.
- Podría ocurrir que, si un encargo necesita pocos vídeos y estos son de corta duración, la transcodificación de estos vídeos concluya antes de que el hilo de encargo se ponga a esperar a que terminen todos sus trabajos asociados. En ese caso, el hilo de encargo tendría que seguir adelante sin quedarse bloqueado.
- Cuando varios encargos necesitan el mismo vídeo, habría que detectarlo para que el vídeo se transcodifique una sola vez.

- Cuando varios hilos de encargo necesitan el mismo vídeo, el vídeo debe estar disponible para todos los hilos que lo necesiten hasta que hayan compuesto y enviado el DVD correspondiente. Tenga en cuenta que, en Java, un objeto existe mientras haya alguna referencia a dicho objeto.
- Los trabajos de transcodificación no deben perderse ni duplicarse por mala gestión de la cola de trabajos.

8 Referencias

Disponible en la plataforma moodle de la asignatura:



[1]. [Información de interés sobre Java](#)



[2]. [Cómo usar git con Eclipse](#)



[3]. [Cómo integrar un JAR en un proyecto de Eclipse](#)

En internet:



[4]. [TortoiseGit](#)

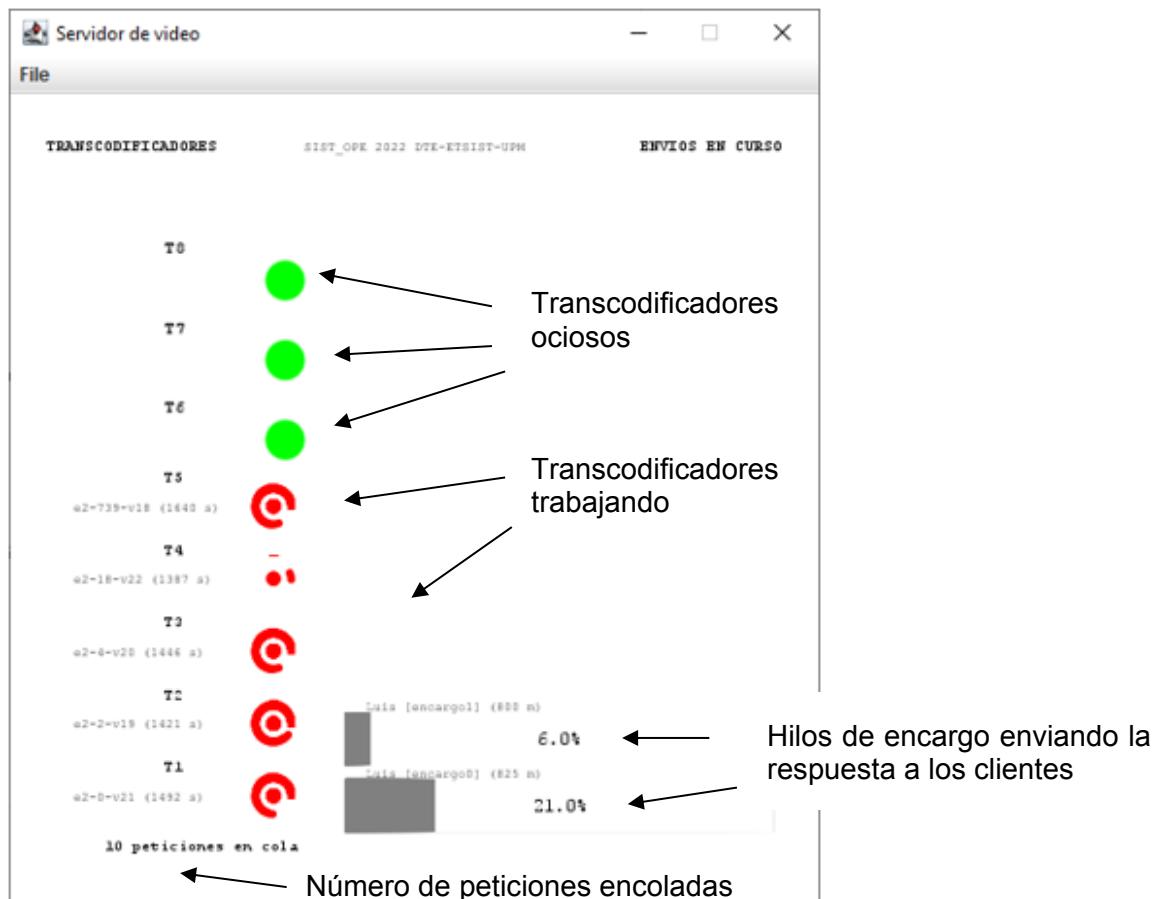


[5]. [Git for windows](#)

Anexo I. Ejemplo de ventana del panel visualizador:

En la ventana del panel visualizador se muestra información acerca del estado del servidor de videos. En concreto, se muestra el estado de los hilos transcodificadores, el estado de los hilos de encargo y el número de peticiones pendientes (trabajos de transcodificación que aún no han sido iniciados).

El panel visualizador representa el estado de hasta 10 hilos de transcodificación. Un círculo de color verde indica que el hilo de transcodificación correspondiente está ocioso. Cuando se pone de color rojo, indica cómo va progresando la transcodificación. También informa del envío del DVD al cliente mediante una barra de progreso. La siguiente imagen muestra un ejemplo:



Anexo II. Ejecución de los clientes

Debe ejecutarlos desde un intérprete de órdenes (en Windows puede usar la aplicación Windows PowerShell o la aplicación Símbolo de sistema):

Cliente interactivo: ejecute la orden:

```
java -cp p3videos.jar sssoo.videos.clientes.ClienteInteractivo
```

Ejemplo de ejecución donde el usuario se llama Luis, el título del DVD es “Mi primer DVD”, está compuesto por 5 vídeos, la duración media de estos vídeos es de 1.500 segundos y la desviación máxima es de 100 segundos:

```
I:\SSOO\Practica3>java -cp p3videos.jar sssoo.videos.clientes.ClienteInteractivo
Introduzca un nombre de usuario: Luis
Título: Mi primer DVD
Número de videos que tendrá el encargo: 5
Duración media de los videos (seg): 1500
Desviación máxima de la duración de los videos (seg): 100
DVD recibido:
RECIBIDO DVD
    Título: Mi primer DVD
    Duración: 124 min
        Vídeo 1: video0 (1522 s)
        Vídeo 2: video1 (1518 s)
        Vídeo 3: video2 (1354 s)
        Vídeo 4: video3 (1454 s)
        Vídeo 5: video4 (1597 s)
I:\SSOO\Practica3>
```

Cliente automatizado: ejecute la orden:

```
java -cp p3videos.jar sssoo.videos.clientes.ClienteAutomatizado
```

Mostrará un texto explicativo de cómo puede ejecutarse este programa mediante argumentos en la línea de órdenes. Si no se le han pasado argumentos, los pedirá por teclado. Es más cómodo poner esos datos como argumentos y así poder repetir las pruebas fácilmente.

Anexo II. Ejecución de los clientes

Debe ejecutarlos desde un intérprete de órdenes (en Windows puede usar la aplicación Windows PowerShell o la aplicación Símbolo de sistema):

Cliente interactivo: ejecute la orden:

```
java -cp p3videos.jar sssoo.videos.clientes.ClienteInteractivo
```

Ejemplo de ejecución donde el usuario se llama Luis, el título del DVD es “Mi primer DVD”, está compuesto por 5 vídeos, la duración media de estos vídeos es de 1.500 segundos y la desviación máxima es de 100 segundos:

```
I:\SSOO\Practica3>java -cp p3videos.jar sssoo.videos.clientes.ClienteInteractivo
Introduzca un nombre de usuario: Luis
Título: Mi primer DVD
Número de videos que tendrá el encargo: 5
Duración media de los videos (seg): 1500
Desviación máxima de la duración de los videos (seg): 100
DVD recibido:
RECIBIDO DVD
    Título: Mi primer DVD
    Duración: 124 min
        Vídeo 1: video0 (1522 s)
        Vídeo 2: video1 (1518 s)
        Vídeo 3: video2 (1354 s)
        Vídeo 4: video3 (1454 s)
        Vídeo 5: video4 (1597 s)
I:\SSOO\Practica3>
```

Cliente automatizado: ejecute la orden:

```
java -cp p3videos.jar sssoo.videos.clientes.ClienteAutomatizado
```

Mostrará un texto explicativo de cómo puede ejecutarse este programa mediante argumentos en la línea de órdenes. Si no se le han pasado argumentos, los pedirá por teclado. Es más cómodo poner esos datos como argumentos y así poder repetir las pruebas fácilmente.