Servicios Multimedia
Grado de Ingeniería en Tecnología de Telecomunicación – 4º curso

Trabajo Laboratorio: Sistemas Multimedia- Streaming Media - (Real Time Streaming Protocol)

En los tres próximos laboratorios vamos a implementar un cliente y un servidor de vídeo streaming , que se comunican mediante el protocolo Real-Time Streaming Protocol (RTSP). El servidor envía datos utilizando el protocolo Real-Time Transport Protocol (RTP).

En este tipo de servicio, el cliente utiliza el protocolo Real-Time Streaming Protocol (RTSM) para controlar sus acciones sobre el servidor, y el servidor utiliza el protocolo real-time protocol (RTP) para empaquetar el video y transportarlo sobre UDP.

En este laboratorio vamos a analizar el protocolo RTSP en el lado cliente. Para ello, a partir del ejemplo Cliente/Servidor que se propone, vamos a completar las funciones que son llamadas cuando pulsamos los botones de la interface que ofrece el cliente.

Implementaremos en esta sesión, las acciones correspondientes al control “PAUSE”, y al control “TEARDOWN”.

Materiales: Mod_05_A_Model.zip, y Mod_05_A_Sources.zip.

El fichero Mod_05_A_Model.zip contiene las clases del cliente y del servidor, así como las correspondientes al protocolo RTP, al reproductor de vídeo y la película que queremos transmitir.

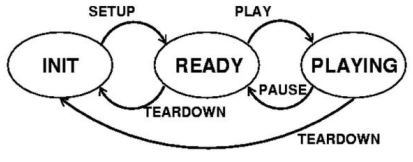
El fichero Mod_05_A_Sources.zip contiene los ficheros Client_V1.java y RTPpacket.java, que tendremos que completar (Cliente_V1.java) para el correcto funcionamiento en la comunicación entre el cliente y el servidor.

Para conocer el comportamiento de este servicio, en primer lugar, ejecutaremos desde la línea de comandos del servidor y del cliente, los siguientes comandos:

Consola 1: C:\java Server 2000

Consola 2: C:\java Client

Sabiendo que el modelo de estados del cliente responde al siguiente diagrama de estados:



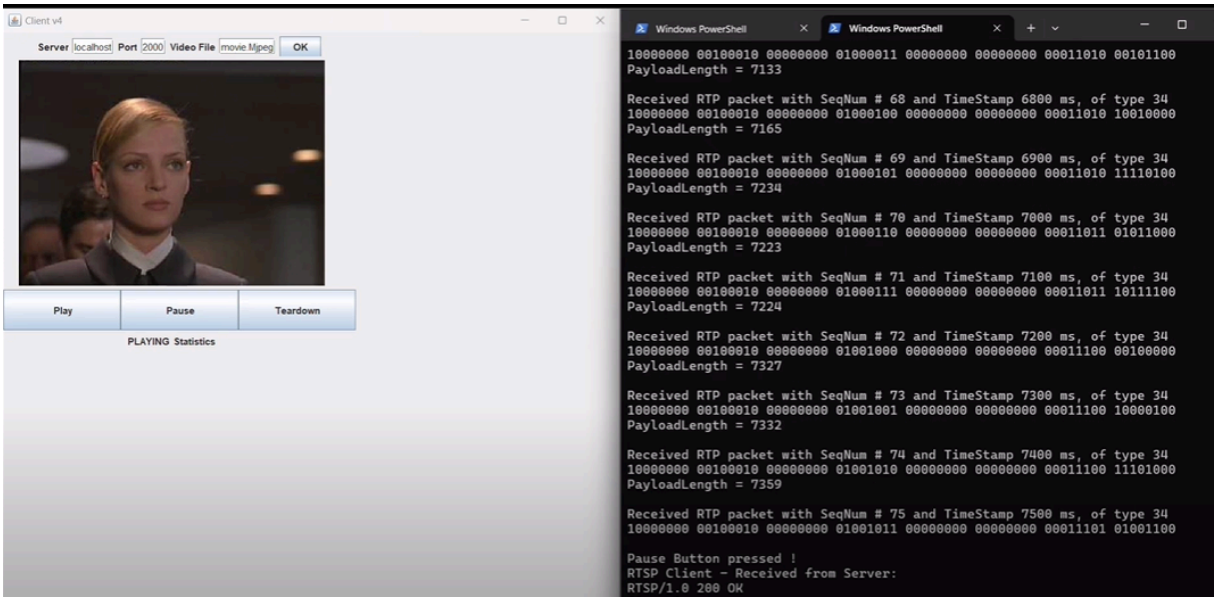
Se pide, sobre el código Client_V1.java:

1. Identificar las diferentes partes en que consiste este proceso cliente.
2. Implementar el código correspondiente a la acción de pulsar sobre la tecla PAUSE, comprobando su correcto funcionamiento. (Conocemos la acción PLAY)

3. Implementar el código correspondiente a la acción de pulsar sobre la tecla TEARDOWN, comprobando su correcto funcionamiento.

Tarea 1: Identificar las diferentes partes en que consiste este proceso cliente.

Screenshot 1:



Breve descripción:

En este apartado verificamos que el cliente pueda conectarse al servidor usando RTSP. Primero, revisamos el archivo Client_V1.java, donde el cliente obtiene el puerto y la dirección del servidor desde los campos de texto en la interfaz gráfica.

Desde la GUI, validamos que los campos estén correctos: el servidor debe ser localhost si todo está en la misma máquina, o una IP válida, y el puerto debe ser 2000. Luego ejecutamos el servidor con java Server 2000 y confirmamos que este escucha correctamente.

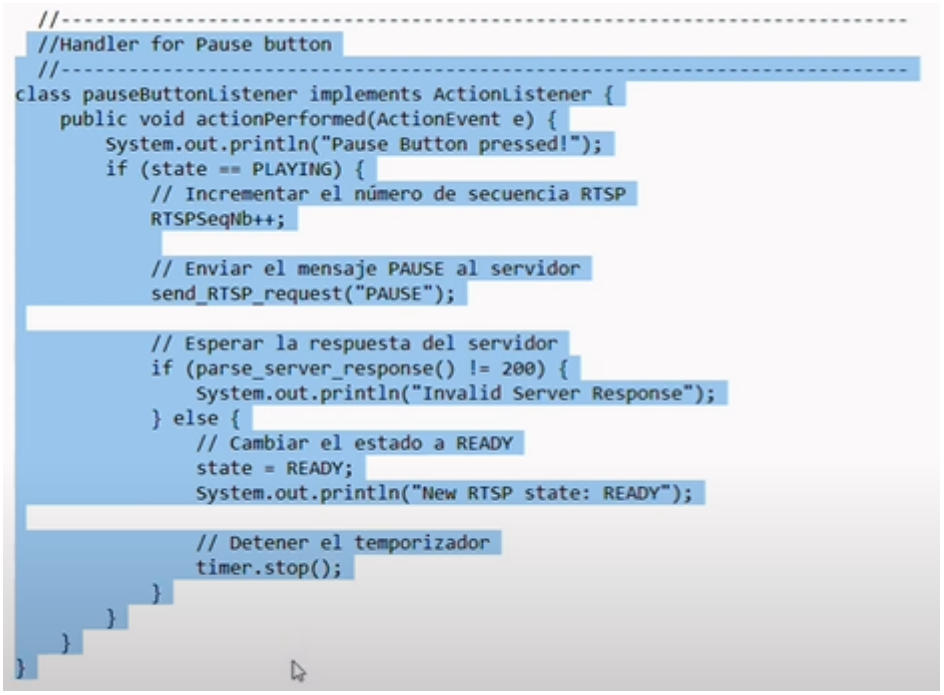
Con el servidor activo, iniciamos el cliente, ingresamos los valores en la GUI, y presionamos Connect. Esto establece la conexión, que confirmamos en la consola con los mensajes de depuración.

Finalmente, al presionar Play, verificamos que el cliente cambie al estado PLAYING y que el servidor comience a enviar paquetes RTP. Si algo falla, revisamos errores comunes como direcciones incorrectas, puertos bloqueados o campos vacíos.

Este proceso garantiza que el cliente y el servidor se comuniquen correctamente mediante RTSP y RTP.

Tarea 2: Implementar el código correspondiente a la acción de pulsar sobre la tecla PAUSE, comprobando su correcto funcionamiento. (Conocemos la acción PLAY)

Screenshot 2:



```
//---------------------------------------------------------------------------
//Handler for Pause button
//---------------------------------------------------------------------------
class pauseButtonListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        System.out.println("Pause Button pressed!");
        if (state == PLAYING) {
            // Incrementar el número de secuencia RTSP
            RTSPSeqNb++;

            // Enviar el mensaje PAUSE al servidor
            send_RTSP_request("PAUSE");

            // Esperar la respuesta del servidor
            if (parse_server_response() != 200) {
                System.out.println("Invalid Server Response");
            } else {
                // Cambiar el estado a READY
                state = READY;
                System.out.println("New RTSP state: READY");

                // Detener el temporizador
                timer.stop();
            }
        }
    }
}
```

Breve descripción:

En este apartado, implementamos la funcionalidad del botón Pause. El objetivo es que al presionarlo, el cliente cambie del estado PLAYING al estado READY. Esto se logra enviando un mensaje PAUSE al servidor, deteniendo la transmisión de video.

Después de enviar el mensaje, verificamos la respuesta del servidor. Si la respuesta es correcta, el cliente cambia su estado a READY y detiene el temporizador que recibe los paquetes RTP.

Tarea 3: Implementar el código correspondiente a la acción de pulsar sobre la tecla TEARDOWN, comprobando su correcto funcionamiento.

Screenshot 3:

```
  //-------------------------------------------------------------------------
  //Handler for Teardown button
  //-------------------------------------------------------------------------
class tearButtonListener implements ActionListener {
    public void actionPerformed(ActionEvent e) {
        System.out.println("Teardown Button pressed!");

        // Incrementar el número de secuencia RTSP
        RTSPSeqNb++;

        // Enviar el mensaje TEARDOWN al servidor
        send_RTSP_request("TEARDOWN");

        // Esperar la respuesta del servidor
        if (parse_server_response() != 200) {
            System.out.println("Invalid Server Response");
        } else {
            // Cerrar los sockets y salir del programa
            try {
                RTSPsocket.close();
                RTPsocket.close();
            } catch (IOException ex) {
                System.out.println("Error closing sockets: " + ex.getMessage());
            }

            System.out.println("Session closed.");
            System.exit(0);
        }
    }
}
```

Breve descripción:

En este apartado, implementamos la funcionalidad del botón Teardown. Este botón finaliza la sesión RTSP y cierra la conexión entre el cliente y el servidor.

Cuando el usuario presiona 'Teardown', el cliente envía un mensaje al servidor para finalizar la transmisión. Una vez recibido el mensaje, el cliente cierra los sockets RTSP y RTP, y termina el programa.
