

PRÁCTICA 1.1: SERVICIO DE TRANSFERENCIA DE ARCHIVOS

Hernández Tapia Luis Enrique.

Escuela Superior de Cómputo
Instituto Politécnico Nacional, México
tapia641@gmail.com

Resumen: En esta práctica se desarrolla una aplicación para seleccionar múltiples archivos y enviarlos a través del protocolo TCP.

Palabras clave: TCP, IP, Sockets de flujo.

1 Introducción

El envío de archivos a través de la red es una característica importante para la gran mayoría de las aplicaciones que hoy día se utilizan (blogs, redes sociales, mensajería instantánea, declaración de impuestos, educación en línea, etc.), sin embargo, no todas las aplicaciones disponibles permiten el envío de archivos de gran tamaño (p.e. El correo electrónico no permite enviar archivos de más de 10 o 20 MB). Esto hace necesario el desarrollo de aplicaciones que permitan transferir archivos sin importar el tamaño de éstos.

2 Desarrollo

A partir de los programas **CArchivo** y **SArchivo** que te serán proporcionados por el profesor deberás realizar los siguientes programas:

Instrucciones:

- El programa **Selección** implementa una caja de diálogo para seleccionar un archivo a través del ratón en el sistema de archivos local. Deberás modificar el código para que te permita seleccionar más de un archivo a la vez y devuelva como salida la lista con los nombres y tamaños de los archivos seleccionados.
- El programa **RecibeArchivo** implementa un servidor de flujo bloqueante que realiza la recepción de un archivo ya predefinido y éste es recibido utilizando flujos orientados a byte. Deberás modificar el código para que en lugar de recibir un archivo, este programa permita recibir desde uno hasta cualquier cantidad de archivos (secuencialmente). Para esto, primero deberá recibir un número que indique el número de archivos que serán recibidos, posteriormente, por cada archivo a ser recibido, primero se recibirá el nombre del archivo, luego su tamaño en bytes y después se recibirá el contenido del mismo.
- Durante la transferencia de los archivos el usuario deberá visualizar el porcentaje de envío en pantalla. El programa **EnviaArchivo** implementa un cliente de flujo bloqueante que envía un archivo ya preestablecido y éste es enviado usando flujos orientados a byte. Deberás modificar el código para que en lugar de enviar un solo archivo, éste sea capaz de enviar uno o más archivos que serán seleccionados por el usuario a través del programa **Selección**.
- Durante la transferencia de los archivos el usuario deberá visualizar el porcentaje de envío en pantalla. El programa **EnviaArchivo** implementa un cliente de flujo bloqueante que envía un archivo ya preestablecido y éste es enviado usando flujos orientados a byte. Deberás modificar el código para que en lugar de enviar un solo archivo, éste sea capaz de enviar uno o más archivos que serán seleccionados por el usuario a través del programa **Selección**.

Cada archivo se enviará de manera individual y el proceso de envío será de la siguiente manera:

Primero se enviará un número indicando la cantidad de archivos que serán transferidos.

Después, de manera iterativa, por cada archivo a ser enviado se mandará previamente el nombre de éste y su tamaño en bytes.

Posteriormente el contenido del archivo.

- Realiza pruebas intentando enviar distintos tipos de archivo (imágenes, texto, ejecutables), así mismo intenta enviar archivos de distintos tamaños (menos de 100KB, más de 100KB y menos de 10MB, más de 10MB y menos de 200MB, más de 200MB y hasta 2GB).

3 Pruebas

Mostrando caja de diálogo:

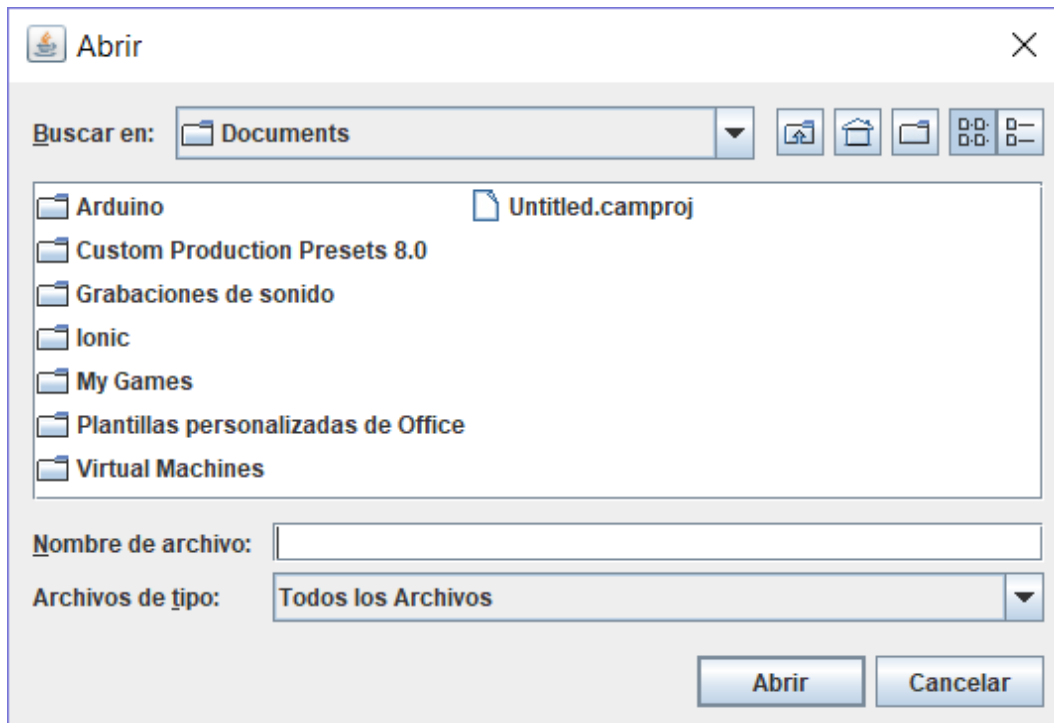
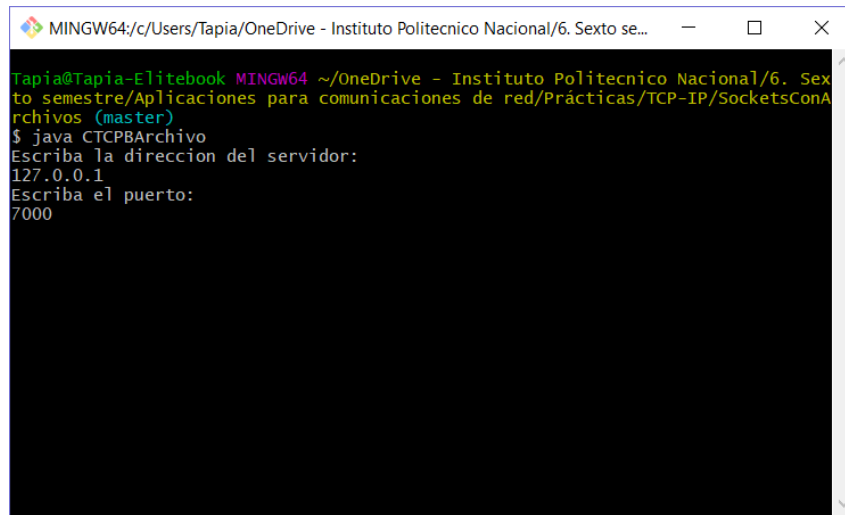


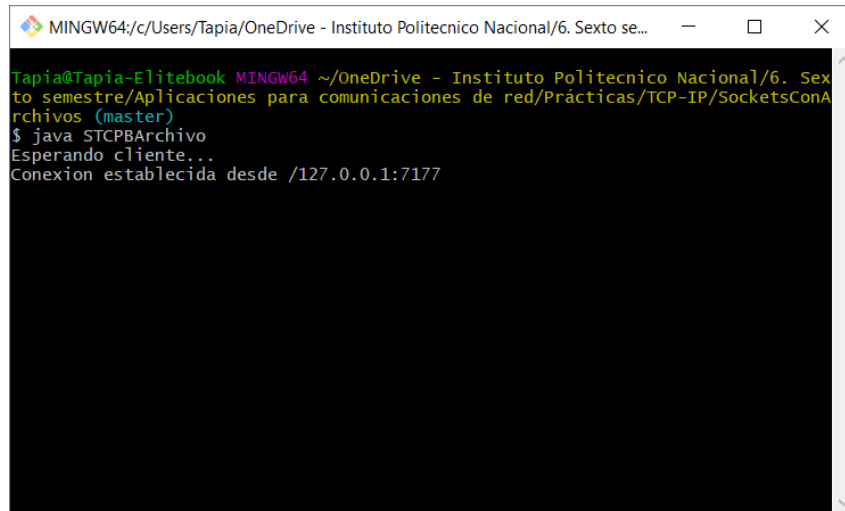
Figure 1: Caja de diálogo para seleccionar archivos.

Probando conexión:

A terminal window titled "MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se..." displays the execution of a Java client application. The user runs the command `$ java CTCPBArchivo`. The program prompts for the server address: `Escriba la direccion del servidor:`, where `127.0.0.1` is entered. It then prompts for the port: `Escriba el puerto:`, where `7000` is entered.

```
MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...
Tapia@Tapia-Elitebook MINGW64 ~/OneDrive - Instituto Politecnico Nacional/6. Sexto semestre/Aplicaciones para comunicaciones de red/Prácticas/TCP-IP/SocketsConArchivos (master)
$ java CTCPBArchivo
Escriba la direccion del servidor:
127.0.0.1
Escriba el puerto:
7000
```

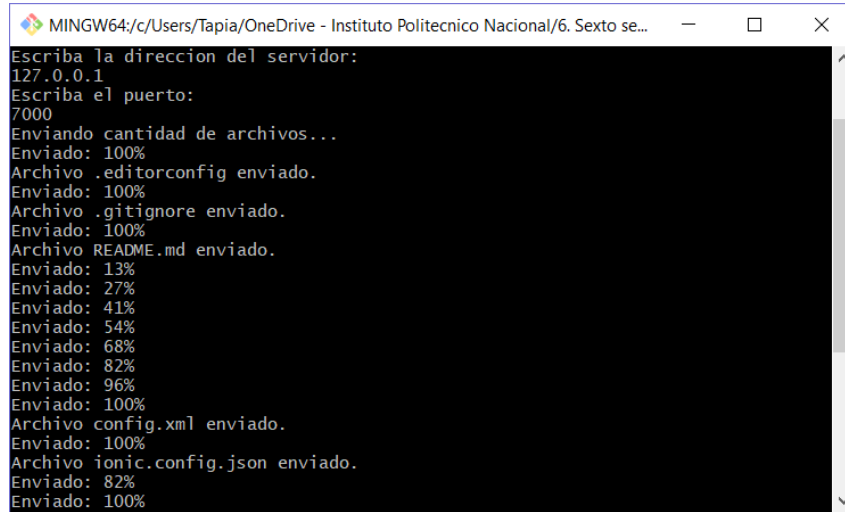
Figure 2: Cliente.

A terminal window titled "MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se..." displays the execution of a Java server application. The user runs the command `$ java STCPBArchivo`. The program outputs `Esperando cliente...` and then `Conexion establecida desde /127.0.0.1:7177`, indicating a successful connection from the client.

```
MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...
Tapia@Tapia-Elitebook MINGW64 ~/OneDrive - Instituto Politecnico Nacional/6. Sexto semestre/Aplicaciones para comunicaciones de red/Prácticas/TCP-IP/SocketsConArchivos (master)
$ java STCPBArchivo
Esperando cliente...
Conexion establecida desde /127.0.0.1:7177
```

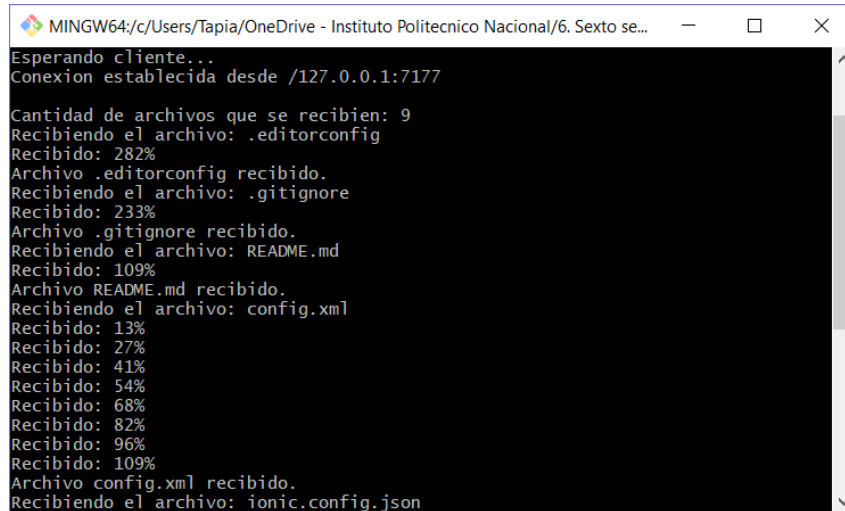
Figure 3: Servidor.

Enviando archivos de a lo más 100KB:



```
MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...
Escriba la direccion del servidor:
127.0.0.1
Escriba el puerto:
7000
Enviando cantidad de archivos...
Enviado: 100%
Archivo .editorconfig enviado.
Enviado: 100%
Archivo .gitignore enviado.
Enviado: 100%
Archivo README.md enviado.
Enviado: 13%
Enviado: 27%
Enviado: 41%
Enviado: 54%
Enviado: 68%
Enviado: 82%
Enviado: 96%
Enviado: 100%
Archivo config.xml enviado.
Enviado: 100%
Archivo ionic.config.json enviado.
Enviado: 82%
Enviado: 100%
```

Figure 4: Cliente.

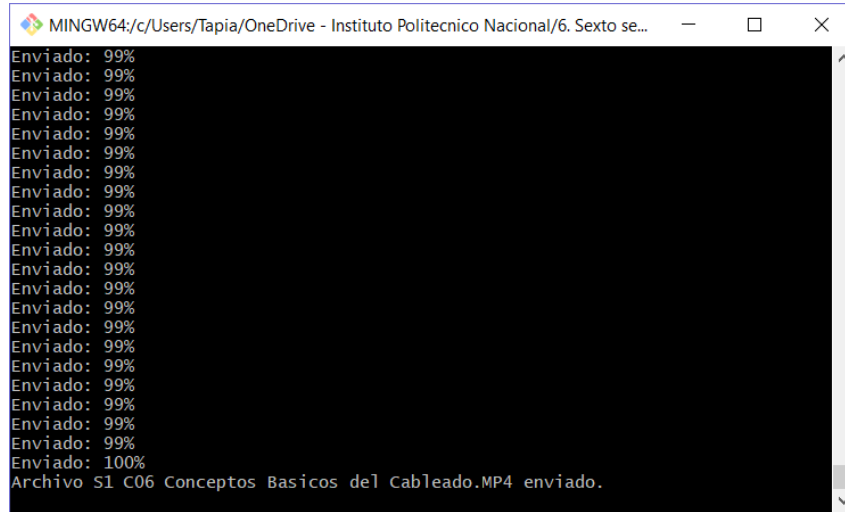


```
MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...
Esperando cliente...
Conexion establecida desde /127.0.0.1:7177

Cantidad de archivos que se reciben: 9
Recibiendo el archivo: .editorconfig
Recibido: 282%
Archivo .editorconfig recibido.
Recibiendo el archivo: .gitignore
Recibido: 233%
Archivo .gitignore recibido.
Recibiendo el archivo: README.md
Recibido: 109%
Archivo README.md recibido.
Recibiendo el archivo: config.xml
Recibido: 13%
Recibido: 27%
Recibido: 41%
Recibido: 54%
Recibido: 68%
Recibido: 82%
Recibido: 96%
Recibido: 109%
Archivo config.xml recibido.
Recibiendo el archivo: ionic.config.json
```

Figure 5: Servidor.

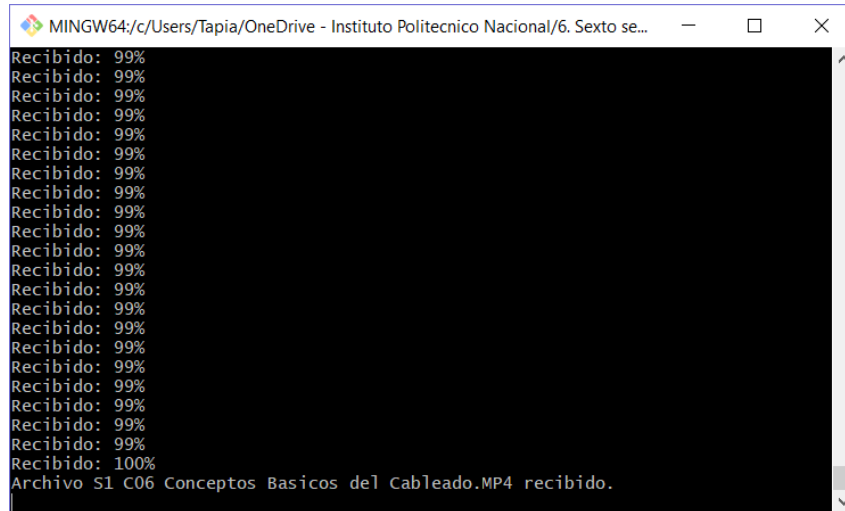
Enviando archivos de a lo más 200MB:



A screenshot of a terminal window titled "MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...". The terminal displays a series of status messages for file uploads. The first 19 lines are "Enviado: 99%", followed by "Enviado: 100%", and the final line is "Archivo S1 C06 Conceptos Basicos del Cableado.MP4 enviado.".

```
MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 99%
Enviado: 100%
Archivo S1 C06 Conceptos Basicos del Cableado.MP4 enviado.
```

Figure 6: Cliente.



A screenshot of a terminal window titled "MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...". The terminal displays a series of status messages for file downloads. The first 19 lines are "Recibido: 99%", followed by "Recibido: 100%", and the final line is "Archivo S1 C06 Conceptos Basicos del Cableado.MP4 recibido.".

```
MINGW64/c/Users/Tapia/OneDrive - Instituto Politecnico Nacional/6. Sexto se...
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 99%
Recibido: 100%
Archivo S1 C06 Conceptos Basicos del Cableado.MP4 recibido.
```

Figure 7: Servidor.

Enviando archivos de a lo más 200MB:

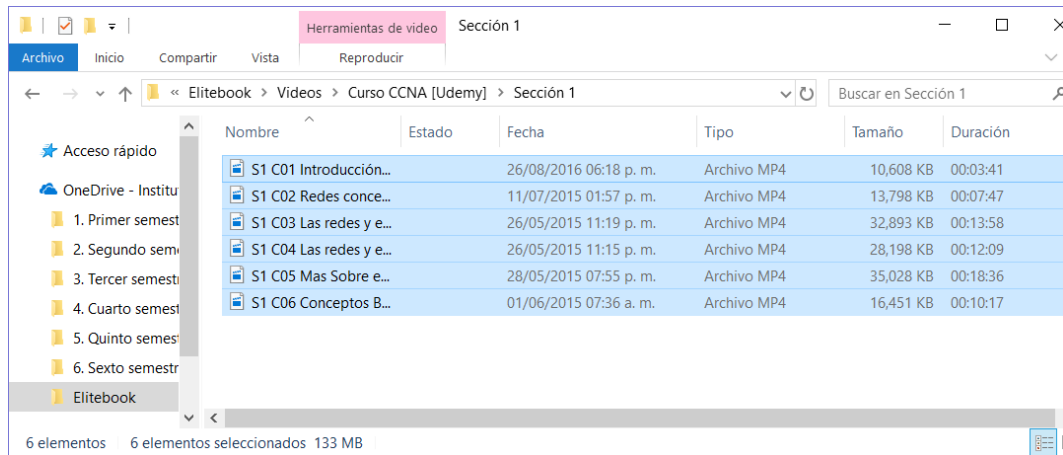


Figure 8: Directorio origen.

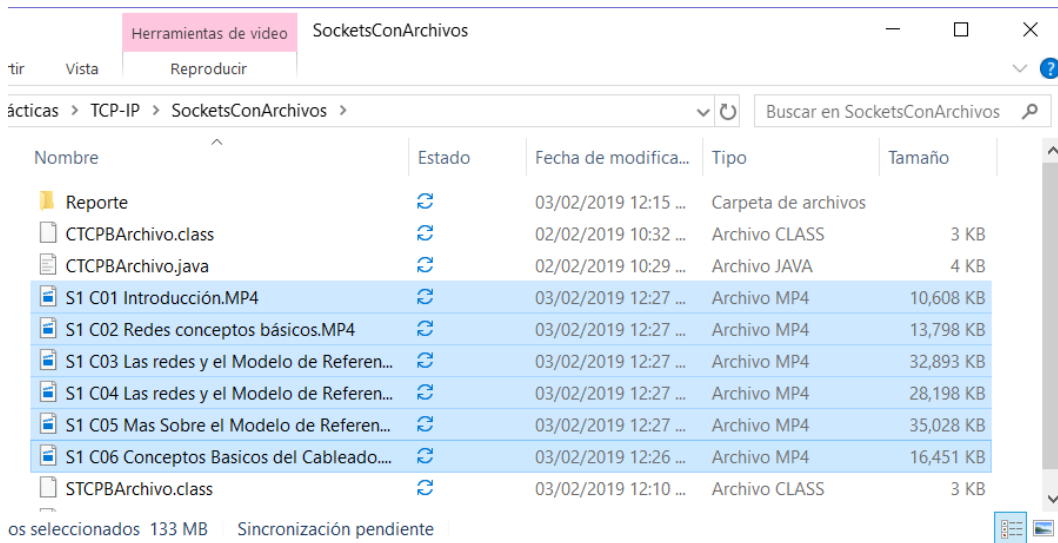


Figure 9: Directorio destino.

Enviando archivos de más 2GB:

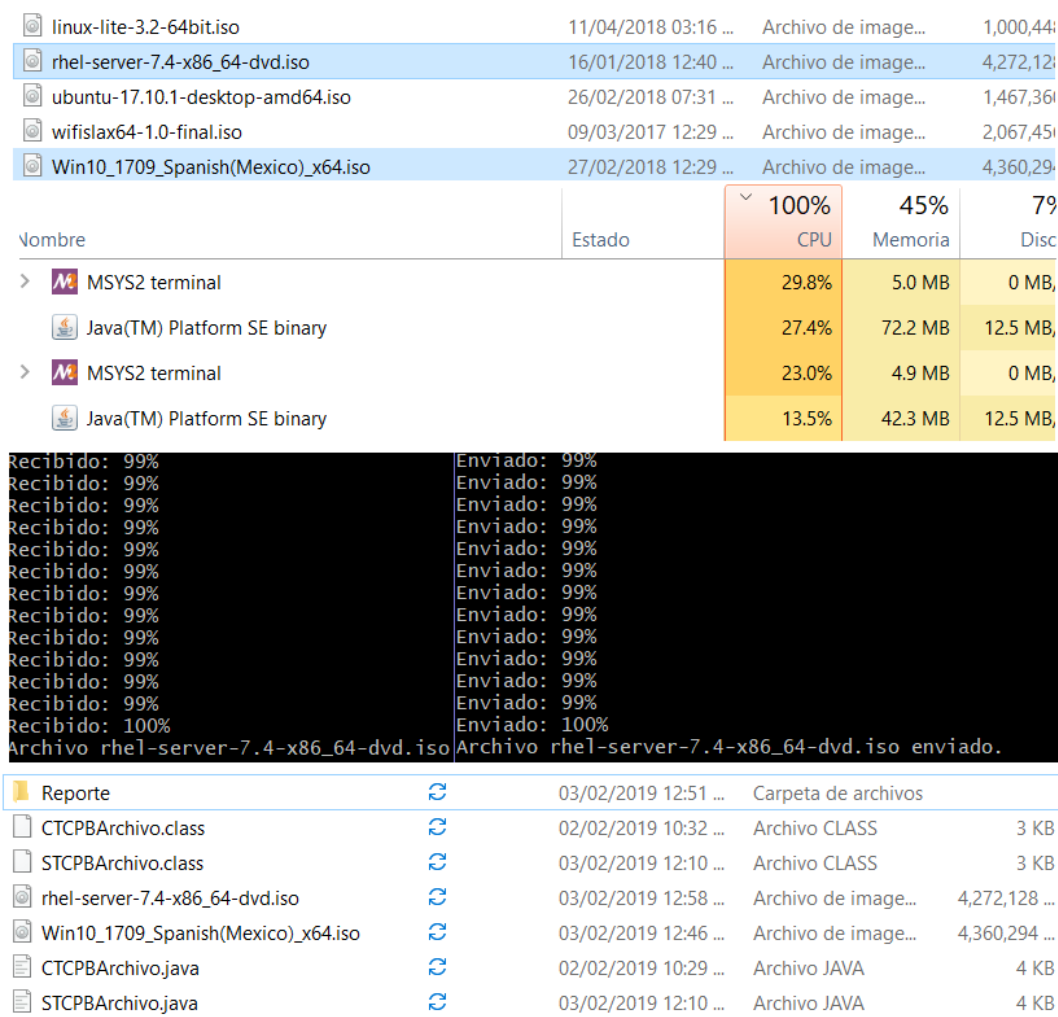


Figure 10: Resumen de la transferencia.

4 Preguntas

1. ¿Qué tipo de archivos se enviaron más rápido?
R: Los de menor cantidad de bytes, ejemplificando algunos como JPG, PNG, MP4.
2. ¿Cuál fue el número máximo de archivos que fue posible enviar a la vez?
R: A la vez 1, pero consecutivamente realicé pruebas con hasta 300 archivos en formato JPG, por lo cual no tuve ningún número máximo de archivos, es decir, pude envíar de forma ilimitada cualquier cantidad de archivos.
3. ¿Cuál fue el tamaño de archivo más grande que se pudo transferir? ¿por qué?
R: Pude envíar archivos de hasta 4GB sin ningún problema, revisando el código, sería aproximadamente el tamaño de un **Long** en Java.
4. Si deseáramos enviar archivos de tamaño muy grande, ¿qué cambios sería necesario hacer con respecto a los tipos de datos usados para medir el tamaño de los archivos, así como para leer bloques de datos del archivo?
R: Cambiar el número de tamaño de los datagramas tanto para el **Cliente** como para el **Servidor**, ya que lo tenemos en 1024.

5 Bibliografía

References

- [1] *Kenneth L. Calvert TCP/IP Sockets in Java: Practical Guide for Programmers. 2nd Edition. Publisher: Morgan Kaufmann*