

# SFT (SECURE FILE TRANSFER)

Aplicación para transferencia de archivos

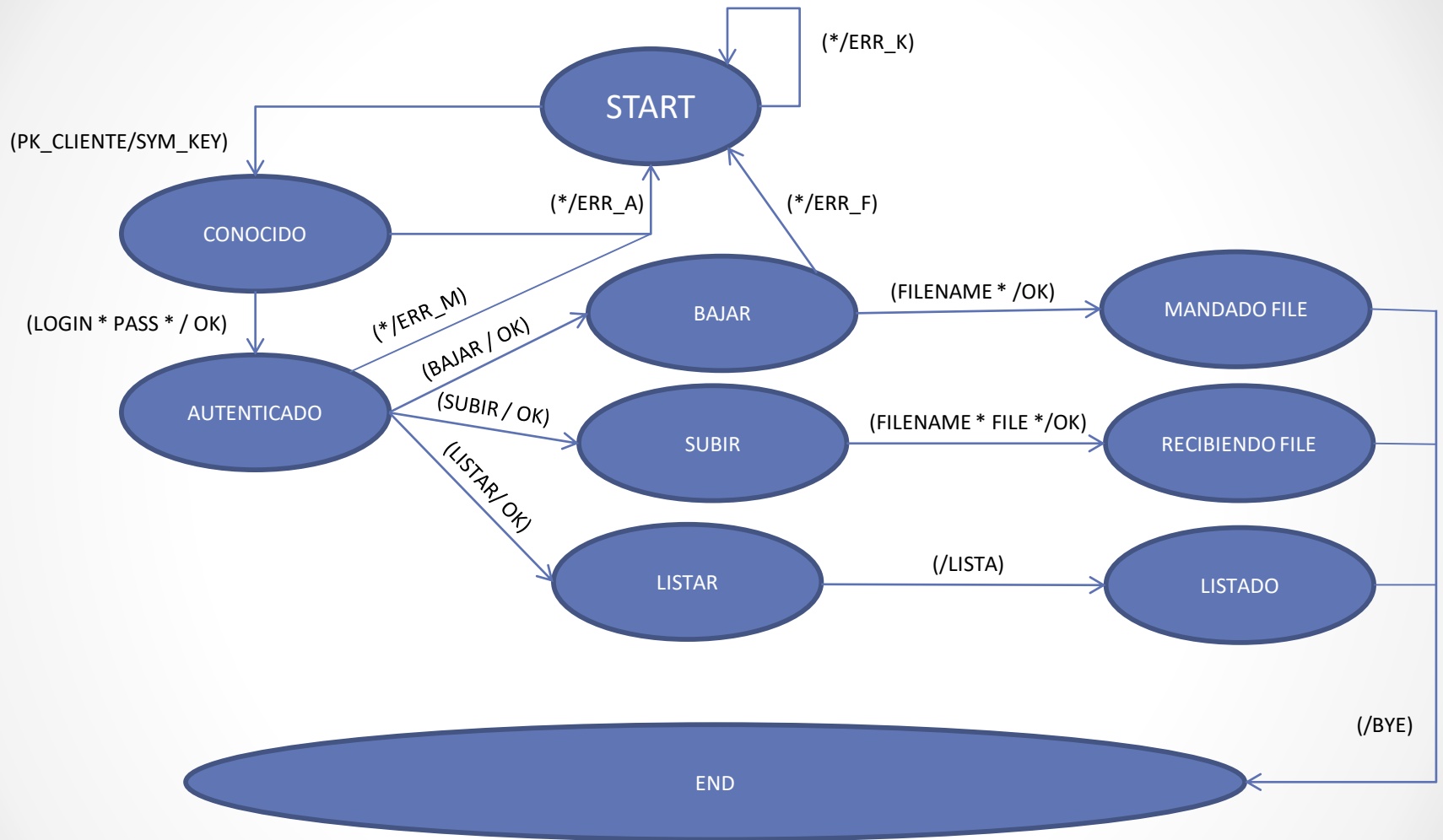
Guillermo Gómez Trenado

Adrián Peláez Vegas

José Antonio Ruiz Millán

# Descripción

- La aplicación permite la transferencia, listado y almacenaje de archivos en un modelo cliente/servidor. Toda la comunicación está encriptada emulando el protocolo TLS (RSA/AES). Asimismo las contraseñas de los usuarios se encuentran hasheadas con MD5.
- Toda la comunicación transcurre sobre TCP ya que necesitamos la garantía de recibir todos los paquetes de forma íntegra y ordenada.
- La aplicación replica el modelo cliente/servidor con clientes concurrentes implementando el paralelismo sobre hebras en Java.
- Cada cliente tiene la opción de alojar, descargar o listar los ficheros asociados a su cuenta de usuario o a la cuenta común.



# Tablas Cliente

Código	Cuerpo	Descripción
1000	PK_CLIENTE	Este mensaje contiene la clave pública (RSA) del cliente encriptada con la clave pública del servidor (esta clave está incluida en la aplicación del cliente para garantizar la identidad).
1001	LOGIN + user + PASS + password	Este mensaje encriptado (AES) contiene la información para autenticarse con el servidor.
2000	BAJAR	Este mensaje (AES) selecciona el modo de trabajo <i>bajar</i> .
2001	SUBIR	Este mensaje (AES) selecciona el modo de trabajo <i>subir</i> .
2002	LISTAR	Este mensaje (AES) selecciona el modo de trabajo <i>listar</i> .
3000	FILENAME + nombre_archivo	Este mensaje (AES) declara el archivo a descargar
3001	FILENAME +nombre_archivo + FILE + binario	Este mensaje (AES) declara el archivo a guardar junto al contenido del mismo.

# Tablas Servidor

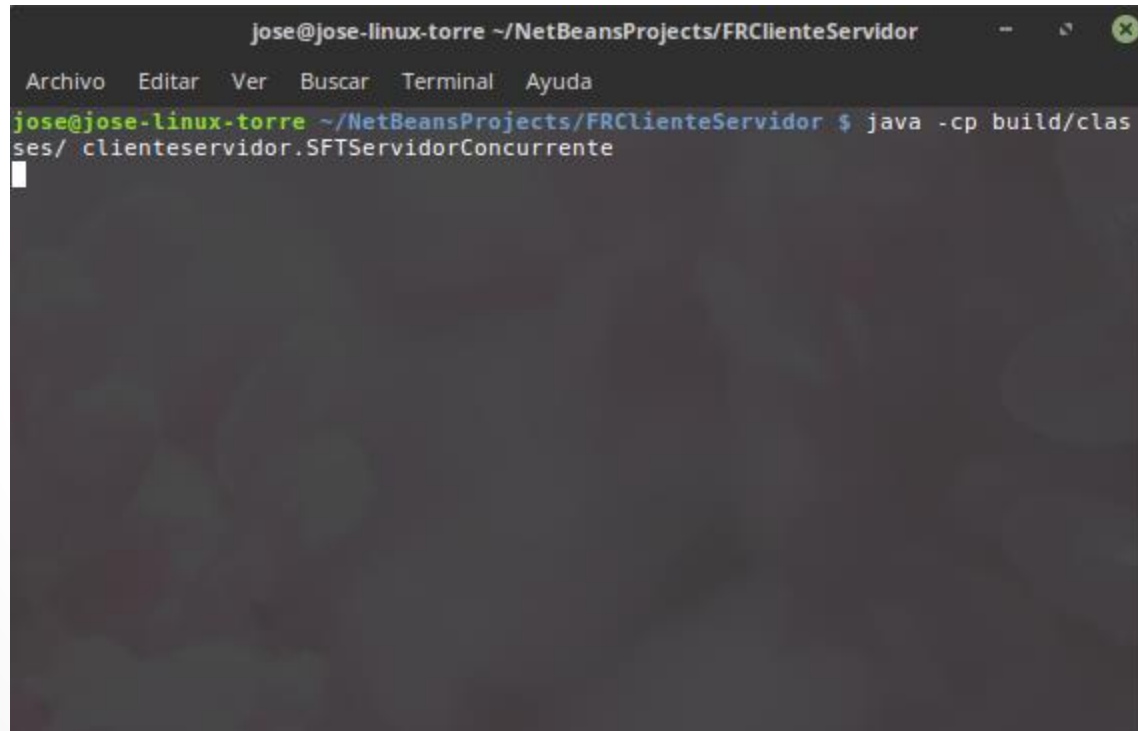
Código	Cuerpo	Descripción
4000	SYM_KEY	Este mensaje encriptado con la clave pública del cliente (RSA) contiene la clave simétrica (AES) sobre la que se define toda comunicación posterior entre el servidor y el cliente.
200	OK	Comunica que el paso ha sido exitoso.
300	ERR_K	Este mensaje comunica que o bien la clave pública del servidor con la que se ha encriptado el mensaje no es correcta o bien, este mensaje no contiene una clave pública del cliente válida.
301	ERR_A	Este mensaje comunica que la clave o usuario no son correctos.
302	ERR_M	Este mensaje comunica que el modo elegido no es un modo válido.
303	ERR_F	Este mensaje comunica que el nombre de archivo asociado al usuario no existe en el sistema.
500	BYE	Cierra la conexión.

# Explicación

- Antes de comenzar con lo práctico, vamos a explicar brevemente las opciones de nuestra aplicación (CLIENTE).
  - Empezando por las distintas opciones, podemos tanto subir como bajar o listar archivos, usando **-up** , **-d** o **-l** respectivamente. En caso de no especificar, se usa **-l**.
  - **-f FICHERO** → para indicar el fichero a subir o bajar.
  - **-t IP** → Dirección IP del servidor, en caso de no indicarla, se usa localhost.
  - **-u USUARIO** → Para indicar el usuario. Si no se especifica se usa el usuario común.
  - **-p PASSWORD** → Para indicar la contraseña del usuario especificado por **-u**.

# Funcionamiento

- Lo primero que vamos a hacer es arrancar el servidor para poder atender a las peticiones.

A screenshot of a terminal window. The title bar at the top reads 'jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor'. Below the title bar is a menu bar with the options 'Archivo', 'Editar', 'Ver', 'Buscar', 'Terminal', and 'Ayuda'. The terminal content shows a prompt 'jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor \$' followed by the command 'java -cp build/ clases/ clienteservidor.SFTServidorConcurrente'. A white cursor is positioned at the end of the command line.

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clases/
clienteservidor.SFTServidorConcurrente
```

# Funcionamiento

- Ahora arrancamos el cliente, en este primer paso vamos a subir un archivo.

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clases/ cliente servidor.SFTClienteTCP -up -u jose -p jose -f /home/jose/pelicula.mkv
```

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clases/ cliente servidor.SFTClienteTCP -up -u jose -p jose -f /home/jose/pelicula.mkv
v
Uploaded: 285,16MB, 88,78MB/s ( 20,14% )
```



# Funcionamiento

- Una vez haya terminado el proceso, nos avisará tanto al cliente como al servidor para llevar un registro de lo que se realiza en el mismo.

## SERVIDOR:

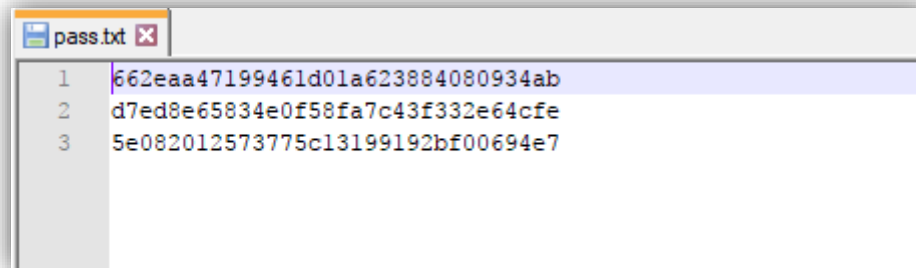
```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clas
ses/ clienteservidor.SFTServidorConcurrente
^Cjose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/cl
ses/ clienteservidor.SFTServidorConcurrente
jose ha subido: Ficheros/jose/pelicula.mkv
```

## CLIENTE:

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clas
ses/ clienteservidor.SFTClienteTCP -up -u jose -p jose -f /home/jose/pelicula.mk
/
Fichero subido correctamente.
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $
```

# Funcionamiento

- Hemos podido percibir que el usuario escribe su contraseña normalmente pero cabe destacar que el servidor las compara y almacena cifradas (MD5) para su seguridad.



```
pass.txt
1 662eaa47199461d01a623884080934ab
2 d7ed8e65834e0f58fa7c43f332e64cfe
3 5e082012573775c13199192bf00694e7
```

# Funcionamiento

- Ahora que ya tenemos un fichero subido, vamos a hacer uso de la opción `-l` para comprobarlo. (La opción `-f ...` no es necesaria y el programa la omite)

SERVIDOR:

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clases/ clienteservidor.SFTServidorConcurrente
^Cjose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clases/ clienteservidor.SFTServidorConcurrente
jose ha subido: Ficheros/jose/pelicula.mkv
Usuario jose a listado sus archivos.
█
```

CLIENTE:

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/clases/ clienteservidor.SFTClienteTCP -l -u jose -p jose -f /home/jose/pelicula.mkv
[pelicula.mkv]
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ █
```

# Funcionamiento

- El siguiente paso será bajar este mismo archivo desde el servidor al cliente. (Con `-f ..` Indicamos donde queremos que se guarde el archivo).

SERVIDOR:

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTServidorConcurrente
^Cjose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTServidorConcurrente
jose ha subido: Ficheros/jose/pelicula.mkv
Usuario jose a listado sus archivos.
jose ha bajado: Ficheros/jose/pelicula.mkv
█
```

CLIENTE:

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTClienteTCP -d -u jose -p jose -f /home/jose/Escritorio/pelicula.mkv
Downloaded: 298,83MB, 81,38MB/s ( 21,11% )
```

# Usuario común

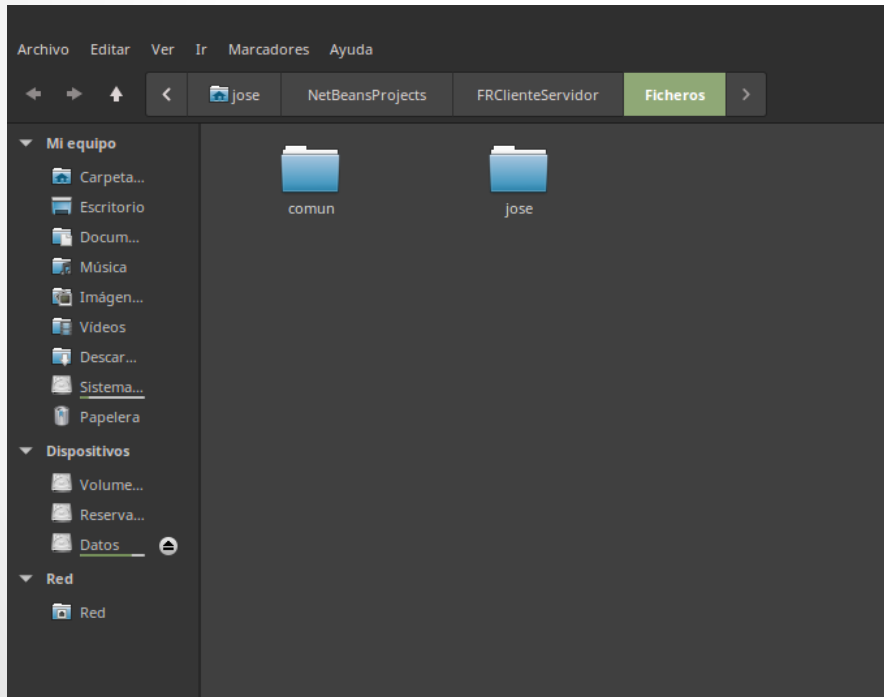
- Hemos dotado a la aplicación de un usuario común que permitirá compartir ficheros de cualquiera con cualquiera, simplemente no tenemos que indicar usuario.

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTServidorConcurrente
^Cjose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTServidorConcurrente
jose ha subido: Ficheros/jose/pelicula.mkv
Usuario jose a listado sus archivos.
jose ha bajado: Ficheros/jose/pelicula.mkv
comun ha subido: Ficheros/comun/pelicula.mkv
□
```

```
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor
Archivo Editar Ver Buscar Terminal Ayuda
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTClienteTCP -d -u jose -p jose -f /home/jose/Escritorio/pelicula.mkv
Fichero bajado correctamente.
jose@jose-linux-torre ~/NetBeansProjects/FRClienteServidor $ java -cp build/classes/ clienteservidor.SFTClienteTCP -up -f /home/jose/Escritorio/pelicula.mkv
Uploaded: 730,47MB, 84,92MB/s ( 51,60% )
```

# Resultados

- Veamos ahora que efectivamente se han creado los directorios oportunos y los ficheros.

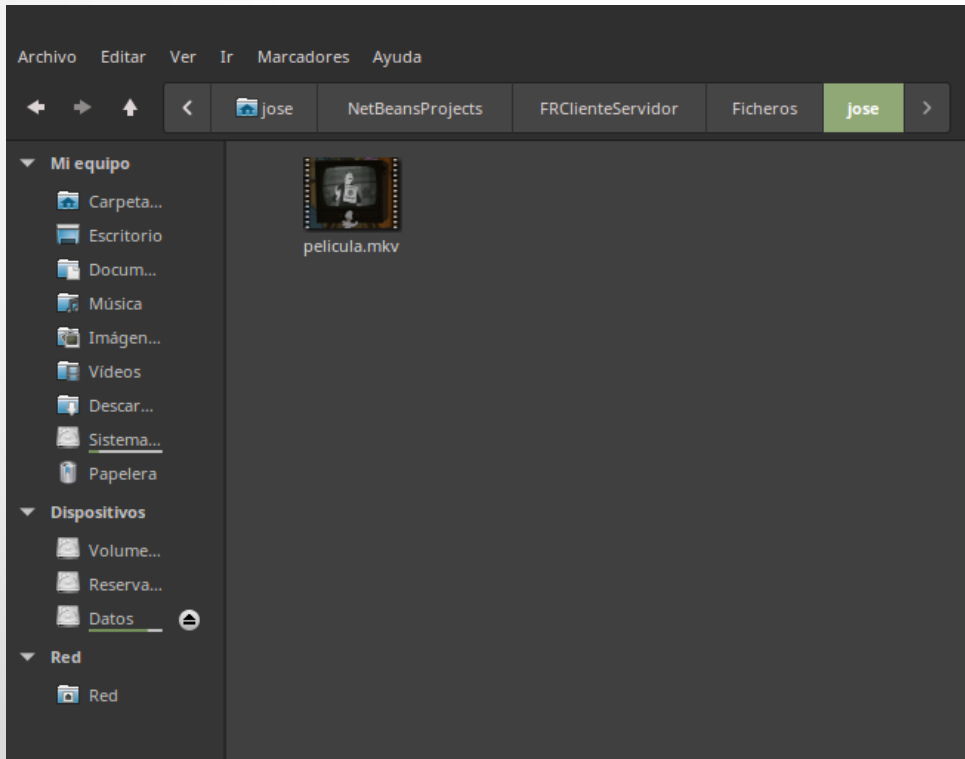


Esta es la parte del servidor, donde efectivamente tenemos creados los directorios tanto de común como de Jose que son los usuarios que han subido información.

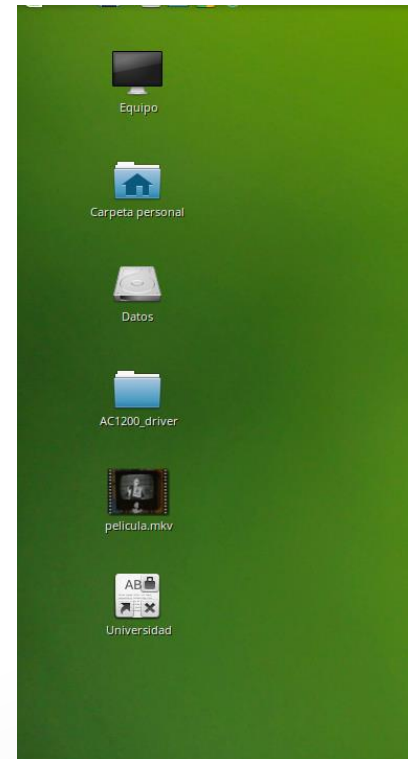
# Resultados

- Veamos ahora que efectivamente se han creado los directorios oportunos y los ficheros.

SERVIDOR:



CLIENTE:



# Final

- Como hemos podido comprobar, la aplicación funciona a la perfección y podemos tener nuestros ficheros almacenados en un servidor y acceder a ellos cuando nos convenga.

