

Fundamentos de Redes

Definición e implementación de un protocolo de aplicación

José María Martín Luque
Antonio Coín Castro
Grupo 2

22 de noviembre de 2017

1. Descripción de la aplicación: conversor de imágenes

Vamos a programar una aplicación, siguiendo el paradigma *cliente/servidor*, que sirva como conversor de imágenes de formato JPEG a PNG y viceversa.

En nuestro modelo, el cliente actúa como un usuario que posee una imagen en un formato concreto, y quiere convertirla al otro. El servidor, por su parte, actúa como un conversor, que aporta la funcionalidad necesaria para modificar el formato de la imagen y devolverla al cliente.

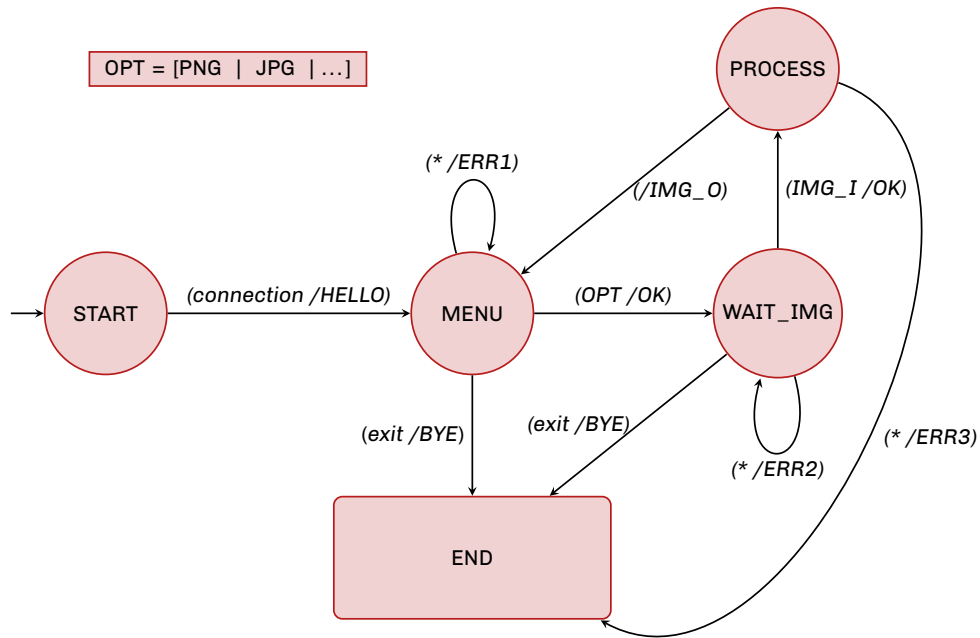
Utilizaremos sockets TCP como medio de comunicación entre el cliente y el servidor, pues al tratarse de una imagen, no podemos permitir que se pierdan datos en el proceso, que lleguen desordenados, etc. Además, el servidor procesará las peticiones de varios clientes de forma **concurrente**.

A grandes rasgos, el funcionamiento es el siguiente: el cliente establece una conexión con el servidor, que le proporciona un menú donde elegir el formato de conversión. Una vez elegido, el cliente envía por el canal TCP la imagen como una cadena de bytes, y el servidor la recibe, la procesa, y la devuelve de nuevo como una cadena de bytes, ya en el formato correcto.

El lenguaje elegido para programar la aplicación es **Go**. Hemos elegido este lenguaje por su facilidad de gestión de servicios en red, así como la simplicidad con la que gestiona la concurrencia (mediante las llamadas *goroutines*). Empleamos una librería de conversión de imágenes para aportar la funcionalidad deseada.

2. Diagrama de estados del servidor

Mostramos ahora un diagrama de los posibles estados en los que puede encontrarse el servidor al establecer conexión con un cliente.



3. Mensajes que intervienen

Los distintos mensajes que intervienen en la comunicación son los siguientes. Los errores se manejan de forma apropiada dentro de los programas del cliente y del servidor.

3.1. Cliente

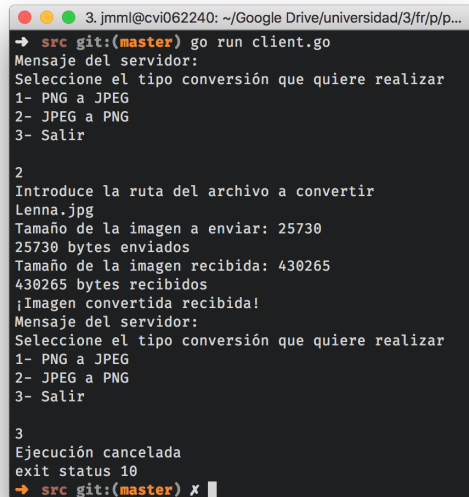
Código	Cuerpo	Descripción
00	OPT + o	Opción de formato a convertir: o
01	IMG_I + img	Datos de la imagen a convertir: img

3.2. Servidor

Código	Cuerpo	Descripción
10	HELLO	Conexión establecida
11	OK	Datos recibidos correctamente
12	ERR1 + str	Opción incorrecta
13	ERR2 + str	Fallo al recibir la imagen
14	ERR3 + str	Fallo al procesar la imagen
15	IMG_O + img	Imagen convertida: img
16	BYE	Finalizar la conexión

4. Evaluación de la aplicación

Por último, mostramos evidencia gráfica de que la aplicación funciona correctamente.

A terminal window titled '3. jmml@cvi062240: ~/Google Drive/universidad/3/fr/p/p...' showing the execution of a client program. The user runs 'go run client.go'. The program prompts for a conversion type, with options: 1- PNG a JPEG, 2- JPEG a PNG, 3- Salir. The user enters '2'. It then prompts for the file path, with 'Lenna.jpg' entered. It shows 'Tamaño de la imagen a enviar: 25730' and '25730 bytes enviados'. Then it shows 'Tamaño de la imagen recibida: 430265' and '430265 bytes recibidos'. A message '¡Imagen convertida recibida!' is displayed. The program then prompts for the conversion type again, with the same options. The user enters '3'. The program outputs 'Ejecución cancelada' and 'exit status 10'. The prompt returns to 'src git:(master) X'.

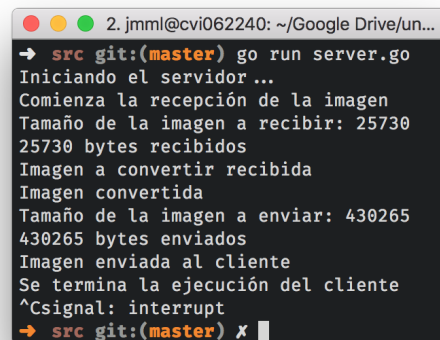
```
3. jmml@cvi062240: ~/Google Drive/universidad/3/fr/p/p...
→ src git:(master) go run client.go
Mensaje del servidor:
Seleccione el tipo conversión que quiere realizar
1- PNG a JPEG
2- JPEG a PNG
3- Salir

2
Introduce la ruta del archivo a convertir
Lenna.jpg
Tamaño de la imagen a enviar: 25730
25730 bytes enviados
Tamaño de la imagen recibida: 430265
430265 bytes recibidos
¡Imagen convertida recibida!
Mensaje del servidor:
Seleccione el tipo conversión que quiere realizar
1- PNG a JPEG
2- JPEG a PNG
3- Salir

3
Ejecución cancelada
exit status 10
→ src git:(master) X
```

Figura 1: Ejemplo de ejecución de un cliente

Como vemos, el cliente selecciona la opción del menú que desee, y a continuación especifica la ruta donde se encuentra la imagen a convertir. La imagen convertida se envía de vuelta al cliente por el canal TCP, y este la guarda en el directorio donde se encontraba la imagen original.

A terminal window titled '2. jmml@cvi062240: ~/Google Drive/un...' showing the execution of a server program. The user runs 'go run server.go'. The program outputs 'Iniciando el servidor...'. It then says 'Comienza la recepción de la imagen'. It shows 'Tamaño de la imagen a recibir: 25730' and '25730 bytes recibidos'. Then it says 'Imagen a convertir recibida'. It then says 'Imagen convertida'. It shows 'Tamaño de la imagen a enviar: 430265' and '430265 bytes enviados'. Then it says 'Imagen enviada al cliente'. It then says 'Se termina la ejecución del cliente'. It outputs '^Csignal: interrupt'. The prompt returns to 'src git:(master) X'.

```
2. jmml@cvi062240: ~/Google Drive/un...
→ src git:(master) go run server.go
Iniciando el servidor...
Comienza la recepción de la imagen
Tamaño de la imagen a recibir: 25730
25730 bytes recibidos
Imagen a convertir recibida
Imagen convertida
Tamaño de la imagen a enviar: 430265
430265 bytes enviados
Imagen enviada al cliente
Se termina la ejecución del cliente
^Csignal: interrupt
→ src git:(master) X
```

Figura 2: Ejemplo de ejecución del servidor

Cuando el servidor detecta la conexión de algún cliente, envía el menú para que este pueda elegir el formato, y a continuación espera la recepción de la imagen. Una vez recibida, la procesa para convertirla y la devuelve, quedando a la espera de dar servicio a más clientes.