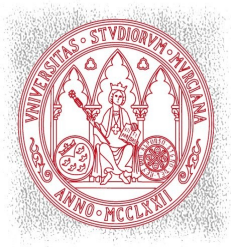


Universidad de Murcia

Facultad de Informática

Departamento de Ingeniería y Tecnología de Computadores

Área de Arquitectura y Tecnología de Computadores

PRÁCTICAS DE REDES

I.I./I.T.I. SISTEMAS /I.T.I. GESTIÓN

Práctica 1. Programación con *Sockets*

FECHA TOPE DE ENTREGA:

PRIMER DÍA DE COMIENZO DEL PERIODO DE EXÁMENES EN CADA CONVOCATORIA

PROFESOR: FÉLIX J. GARCÍA CLEMENTE (FGARCIA@UM.ES)

1.Objetivos

El objetivo de la presente práctica es el aprendizaje del uso de *sockets* para la programación de aplicaciones cliente/servidor, así como el diseño e implementación de los protocolos de comunicaciones subyacentes, a partir de una especificación funcional de alto nivel.

2.Gestión de Recursos Distribuidos

La aplicación tiene como objetivo permitir la comunicación de *hosts* conectados entre sí, mediante el protocolo TCP/IP de Internet, para la gestión de recursos distribuidos de diferente naturaleza.

La aplicación distingue tres tipo de entidades, el servidor de localización de recursos (SLR), el servidor de petición de recursos (SPR) y el cliente del SPR. El SPR y el cliente del SPR interactúan con el SLR de acuerdo con los diagramas de flujo de las figuras 1 y 2, respectivamente. De acuerdo con estos diagramas, el SPR debe ser capaz de registrar un recurso que tenga disponible en el SLR. De esta forma, cuando un cliente cualquiera solicite al SLR dicho recurso, éste deberá proporcionarle la dirección IP y el puerto del SPR en el que el recurso está disponible. A continuación, el cliente se dirigirá al SPR para utilizar el recurso.

3.Servidor de gestión de cuentas de correo

En nuestro caso, el recurso proporcionado será un servidor para la gestión centralizada de cuentas de usuario. Dicho servidor debe proporcionar las funciones básicas para realizar las funciones habituales de gestión de usuarios, es decir, creación de usuarios, cambio de clave, eliminación de usuarios, consulta, etc.

Cada grupo de prácticas tendrá que implementar un SPR que registre un servicio de gestión de cuentas de correo, con el correspondiente cliente, para procesar consultas y realizar actualizaciones con una sintaxis especificada y, en consecuencia, devolver los resultados o modificar (dependiendo del tipo de petición) el(los) fichero(s) de gestión de usuarios.

1. El SPR registra el recurso en el SLR (Figura 1).
2. El cliente envía al SLR una petición de localización del recurso (Figura 2).
3. El cliente envía al SPR una solicitud de conexión.
4. El SPR comprueba la dirección IP del cliente.
   4.1. Si se encuentra entre las direcciones permitidas (listadas en un fichero), devuelve la versión del servidor.
   4.2. Si no se encuentra entre las direcciones permitidas, devuelve un código de error y cierra la conexión.
5. El cliente envía al SPR una solicitud de acceso. Esta solicitud de acceso debe indicar al SPR un nombre de usuario y una contraseña.
6. El SPR comprueba el usuario/contraseña:
   6.1. Si es correcto, devuelve la versión del servidor
   6.2. Si no es correcto, devuelve un código de error y cierra la conexión
7. El cliente envía al SPR una consulta o actualización.
8. El SPR atiende la petición recibida del cliente. Deberá tener en cuenta el acceso/modificación de los datos de forma concurrente. A la hora de atender las peticiones, deberá tener en cuenta los permisos del usuario conectado (el indicado en el paso 5.):

8.1. Si el usuario conectado, tiene permisos de administrador, podrá realizar cualquier operación.

8.2. Si el usuario tiene permisos de operador, solo podrá realizar cambios de contraseña. No podrá realizar operaciones de creación, borrado (ni otras actualizaciones). Podrá realizar consultas sobre cualquier cuenta.

8.3. En cualquier otro caso, el usuario solo puede cambiar su propia contraseña. Tampoco puede realizar operaciones de consulta.

8.4. En caso de que un usuario solicite una operación para la que no tiene acceso, se devolverá un error relativo a este problema.

9. En el caso de una consulta, el SPR envía al cliente los datos solicitados. En el caso de una actualización, el SPR informa al cliente si la operación se ha realizado con éxito o si ha habido algún error. En todos los casos, se indicará también el tiempo transcurrido (**con precisión mínima de microsegundos**) desde que el servidor recibió la petición hasta que terminó de procesarla.

10. Cuando el cliente recibe los datos, mostrará el resultado al usuario y vuelve al paso 6.

Para ilustrar el funcionamiento de la aplicación, se muestra un posible escenario de interacción con el usuario:

```
$ ./cliente …..
Versión del servidor: 1.0
orden?# listado                          (listado de todos los usuarios)
uid|login|nombre|email|clave
1000|juan|Juan Palomo|juanpalomo@um.es|miclave|administrador
1001|pepito|Jose Grillo|pepito.grillo@ditec.um.es|otraclave|operador
1002|luis|Luis González|luisito@ditec.um.es|opensjue|usuario
0.01 segundos[1]

orden?# crea laura laura@um.es ei2wuw operador Laura Martínez
     (creación de usuario)
OK uid: 1003
0.02 segundos

orden?# buscar login=l.*                              (consulta)
1002|luis|Luis González|luisito@ditec.um.es|opensjue|usuario
1003|laura|Laura Martínez|laura@um.es|ei2wuw|operador
0.01 segundos

orden?# cambiaclave juan miclave oloqw8ue          (actualización)
OK
0.01 segundos
```

1 El tiempo necesario para realizar la operación se debe indicar en una unidad de tiempo acorde a la duración normal de las operaciones, es decir, se debe mostrar un tiempo en una unidad de tiempo que sea significativa.

```
orden?# salir
Tiempo de procesamiento total: 0.05 segundos
Número total de operaciones: 4
Número total de actualizaciones: 2
```

4.Funcionalidad de clientes y servidor SPR

4.1.Interfaz mínima de clientes

El fichero binario ejecutable para los clientes se llamará `cliente`:

```
cliente -l SLR -p PSLR -r REC
```

donde:

•`SLR` es la dirección IP del SLR.

•`PSLR` es el puerto del SLR.

•`REC` es el recurso que quiere utilizar el cliente.

•**Las opciones de ejecución del cliente deben poder indicarse en cualquier orden.**

*4.2.*Interfaz mínima del SPR

El fichero binario ejecutable para los clientes se llamará `spr`:

```
spr -q PSPR -b FICHERO -l SLR -p PSLR -r REC
```

donde:

- `PSPR` es el puerto del SPR (el puerto en que va a escuchar peticiones de sus clientes).
- `FICHERO` es un fichero donde se encuentran los usuarios. **La implementación puede hacer uso de otros ficheros auxiliares y/o temporales para almacenar los datos necesarios**.
- `SLR` es la dirección IP del SLR.
- `PSLR` es el puerto por el que el SLR recibe las peticiones de localización de recursos.
- `REC` es el recurso que registra el SPR en el SLR.
- **Las opciones de ejecución del spr deben poder indicarse en cualquier orden.**
- **El SPR debe ser capaz de escuchar peticiones de los clientes en un puerto distinto del 3490.**

*4.3.*Funcionalidad e implementación de los clientes y el SPR

A continuación se resumen los requisitos funcionales y de implementación que deben cumplir los clientes y el SPR:

- La comunicación entre los clientes y el SPR debe estar basada en *sockets stream*.
- El SPR debe tener un **esquema de implementación concurrente, creando un hilo diferente**

para cada cliente. Deberá, por tanto, soportar conexión simultánea de varios clientes independientes, **garantizando en todo momento la integridad de los datos**.

- El código de los clientes y del SPR debe ser portable.
- Tanto los clientes como el SPR deben mostrar un mensaje descriptivo significativo cada vez que envían o reciben un paquete.
- **Los paquetes intercambiados entre cliente y SPR no pueden estar limitados en tamaño, es decir, deben poder ser de un tamaño arbitrariamente grande**.

4.4.Especificación de consultas y modificaciones

De forma deliberada no se especifica ni el formato de los ficheros de usuarios, ni el formato de los paquetes, que deberá ser elegido por los integrantes del grupo. Sí que se especifican las operaciones y el formato de las mismas para el cliente que accede al SPR.

El conjunto de órdenes a implementar, queda a decisión de los integrantes del grupo, pero al menos se debe proporcionar las siguientes:

salir

- Sintaxis: `salir`
- Descripción: La orden salir finaliza la ejecución del cliente y por tanto, la conexión con el SPR. En la respuesta al comando, el servidor informará al cliente del número de operaciones que ha realizado (el total), el número de actualizaciones realizadas (de entre el total de operaciones) y el tiempo total de procesamiento (del servidor) para atender a las operaciones.

conecta

- Sintaxis: `conecta <login> <clave>`
- Descripción: La orden conecta envía la solicitud de acceso al servidor e inicia una **sesión**. El servidor deberá comprobar que la clave indicada es la correcta para dicho usuario. En la respuesta al comando, el servidor informará al cliente, en caso de éxito, del número de versión del servidor. En caso de que la clave no sea correcta deberá devolver un error.
- Restricciones: No se debe haber realizado un **conecta** previamente.

listado

- Sintaxis: listado
- Descripción: Genera un listado con todos los usuarios existentes

crea

- Sintaxis: crea <login> <email> <clave> <permisos> <nombre>
- Descripción: Crea un nuevo usuario. Se debe tomar por nombre todo lo que se incluya a partir de la clave, es decir, el nombre no debe estar limitado a nombre + apellido. Por ejemplo, serían

instrucciones válidas: “crea angel amateo@um.es miclave Angel Mateo” y “crea angel amateo@um.es miclave Angel Luis Mateo”

- Restricciones: No puede existir previamente un usuario con el login deseado. No puede existir previamente un usuario con el email indicado. Se debe haber iniciado una sesión (comando **conecta**) con un usuario con permisos de administrador. Permisos debe ser igual a “administrador”, “operador” o “usuario”, cualquier otro valor debe generar un error.

buscar

- Sintaxis: buscar <campo>=<patron>
- Descripción: Buscar los usuarios que cumplan con la condición indicada. Los patrones a buscar se deben indicar mediante **expresiones regulares**[2].
- Restricciones: campo debe ser un campo válido. En caso de que la búsqueda no proporcione ningún resultado (pero sí que se busque por un campo válido), no se debe generar un error. Se debe haber iniciado una sesión (comando **conecta**) con un usuario con permisos de administrador u operador.

cambiaclave

- Sintaxis: cambiaclave <login> <clave anterior> <nueva clave>
- Descripción: Cambia la clave de un usuario
- Restricciones: La clave anterior proporcionada debe coincidir con la clave que tuviera el usuario. Si se ha iniciado sesión (comando **conecta**) con un usuario sin privilegios, el usuario solo podrá cambiar su propia contraseña. Si se ha iniciado sesión con un usuario con permisos de operador o administrador, se puede cambiar la clave de cualquier usuario.

borra

- Sintaxis: borra <login>
- Descripción: Borra un usuario
- Restricciones: Debe existir un usuario con el login indicado. Se debe haber iniciado una sesión (comando **conecta**) con un usuario con permisos de administrador.

bloquea

- Sintaxis: bloquea
- Descripción: Bloquea el uso del servidor para varias operaciones. El servidor quedará bloqueado y solo atenderá las peticiones de este cliente hasta que no reciba el comando **desbloquea**.

2 La utilización de expresiones regulares para procesar patrones en las consultas puede simplificarse utilizando las funciones descritas en el manual “The GNU C Library Reference Manual”, Sección 10.3. *Regular Expresion Matching*.

- Restricciones: El servidor no se puede encontrar ya dentro un **bloquea** solicitado por otro cliente. En este caso, debe devolver un error. Se debe haber iniciado una sesión (comando **conecta**) con un usuario con permisos de administrador u operador.

desbloquea

- Sintaxis: desbloquea
- Descripción: Desbloquea el servidor tras una orden **bloquea**. A partir de recibir este comando el servidor ya podrá atender las peticiones que estuvieran esperando a que se liberara el bloqueo.
- Restricciones: El servidor debe haber recibido previamente un comando **bloquea** de este mismo cliente. En caso de no ser así, deberá devolver un error indicativo. Se debe haber iniciado una sesión (comando **conecta**) con un usuario con permisos de administrador u operador.

5.Desarrollo

A continuación se enumeran algunos aspectos relevantes relativos al desarrollo de la práctica:

- Tanto el código fuente del cliente como el del SPR deben estar escritos en lenguaje C bajo Linux (Ubuntu). El cliente debe ser compatible con el SLR proporcionado por los profesores de la asignatura. Además, tanto el cliente como el SPR deben ser compatibles y operativos en el laboratorio de prácticas.
- El cliente debe tener un diseño lo más **modular posible**, que ayude a su comprensión y que aborde el problema de forma **clara y concisa**.
- El SLR se encuentra en la dirección IP **155.54.204.49**, puerto **UDP 3490**.

6.Entrega

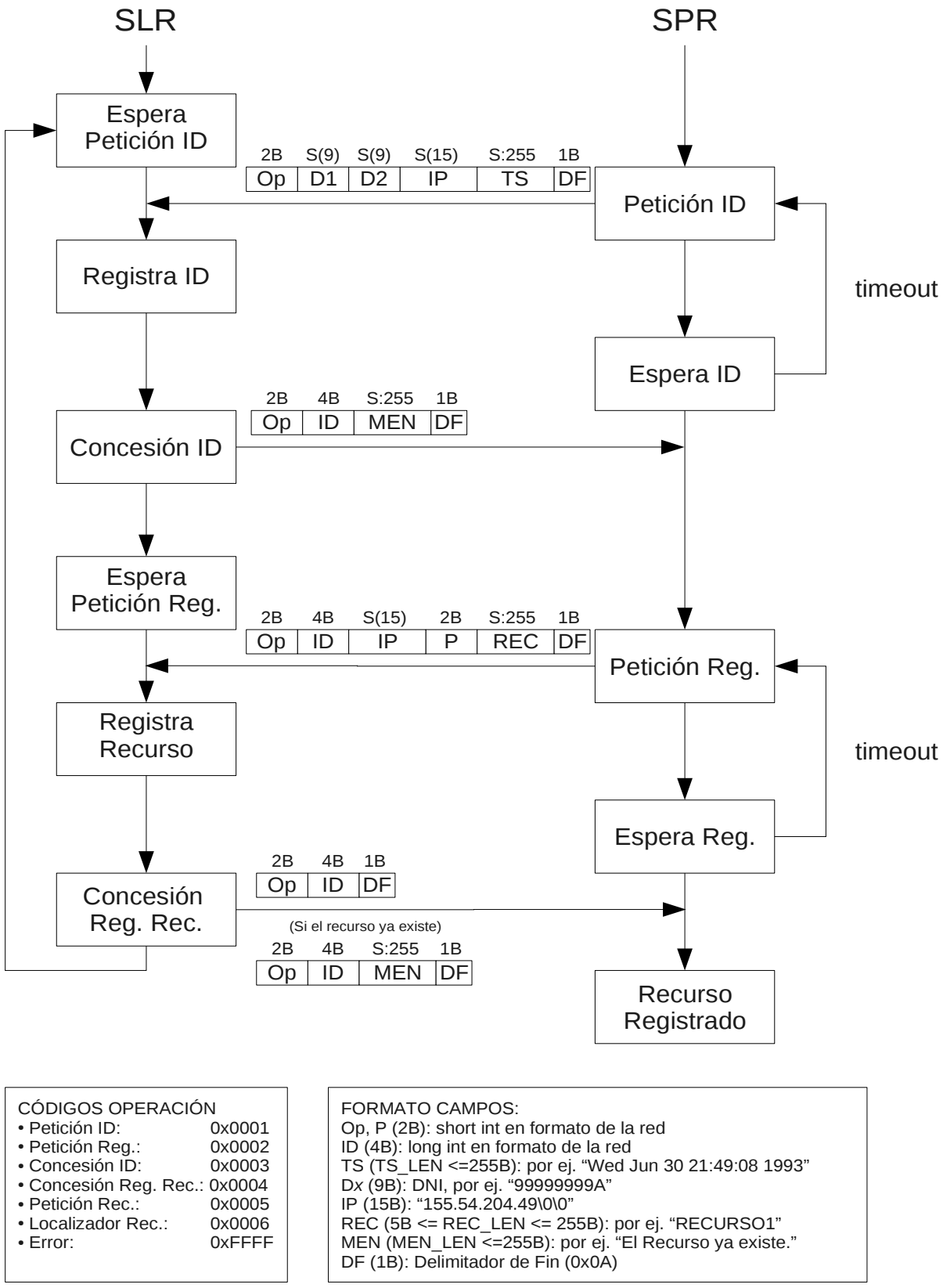
Tanto la memoria como el código fuente se deben envíar a SUMA[3]. La memoria DEBE tener la siguiente estructura:

- Portada que indique claramente la asignatura, la titulación, la práctica y los datos del grupo (nombre y apellidos de los dos miembros del grupo).
- Diagrama de flujo, similar a los de las figuras 1 y 2, que describa la interacción entre el SPR y los clientes, y que incluya el formato **EXACTO** de los paquetes. El diagrama debe reflejar el protocolo implementado de forma **PRECISA**.
- Explicación BREVE, CLARA y CONCISA (no más de un folio por una cara) de los aspectos más relevantes de implementación, así como de las MEJORAS, si las hubiera.
- La memoria **NO** incluirá el listado del código (éste se enviará al profesor a través de SUMA).

3 En la fecha de entrega, el profesor retirará de la zona del alumno el fichero correspondiente

Se debe dejar en SUMA un fichero TAR.GZ con la memoria de la práctica y el código de la práctica. Será un fichero TAR.GZ que incluya el código fuente del cliente, el código fuente del SPR, un `Makefile` sin rutas absolutas, que permita su compilación y montaje, y un fichero de texto `grupo.txt` con los datos de los miembros del grupo. El nombre del fichero DEBE ajustarse al siguiente patrón `REDES-N1-N2.tar.gz`, donde N1 y N2 son los nombres completos de los dos miembros del grupo. Si el grupo está formado por un único componente, N2 se omitirá. El profesor ignorará todos aquellos ficheros que no se ajusten a este patrón.

NOTAS: EL INCUMPLIMIENTO DE LAS NORMAS DE ENTREGA IMPLICA UN SUSPENSO EN PRÁCTICAS. LA COPIA DE PRÁCTICAS SE TRADUCIRÁ EN LA APERTURA DE UN EXPEDIENTE SANCIONADOR A TODOS LOS ALUMNOS PERTENECIENTES A LOS GRUPOS INVOLUCRADOS.



CÓDIGOS OPERACIÓN
- Petición ID: 0x0001
- Petición Reg.: 0x0002
- Concesión ID: 0x0003
- Concesión Reg. Rec.: 0x0004
- Petición Rec.: 0x0005
- Localizador Rec.: 0x0006
- Error: 0xFFFF

FORMATO CAMPOS:
Op, P (2B): short int en formato de la red
ID (4B): long int en formato de la red
TS (TS_LEN <=255B): por ej. “Wed Jun 30 21:49:08 1993”
D*x* (9B): DNI, por ej. “99999999A”
IP (15B): “155.54.204.49\0\0”
REC (5B <= REC_LEN <= 255B): por ej. “RECURSO1”
MEN (MEN_LEN <=255B): por ej. “El Recurso ya existe.”
DF (1B): Delimitador de Fin (0x0A)

Figura 1: Diagrama de flujo para el registro de recursos

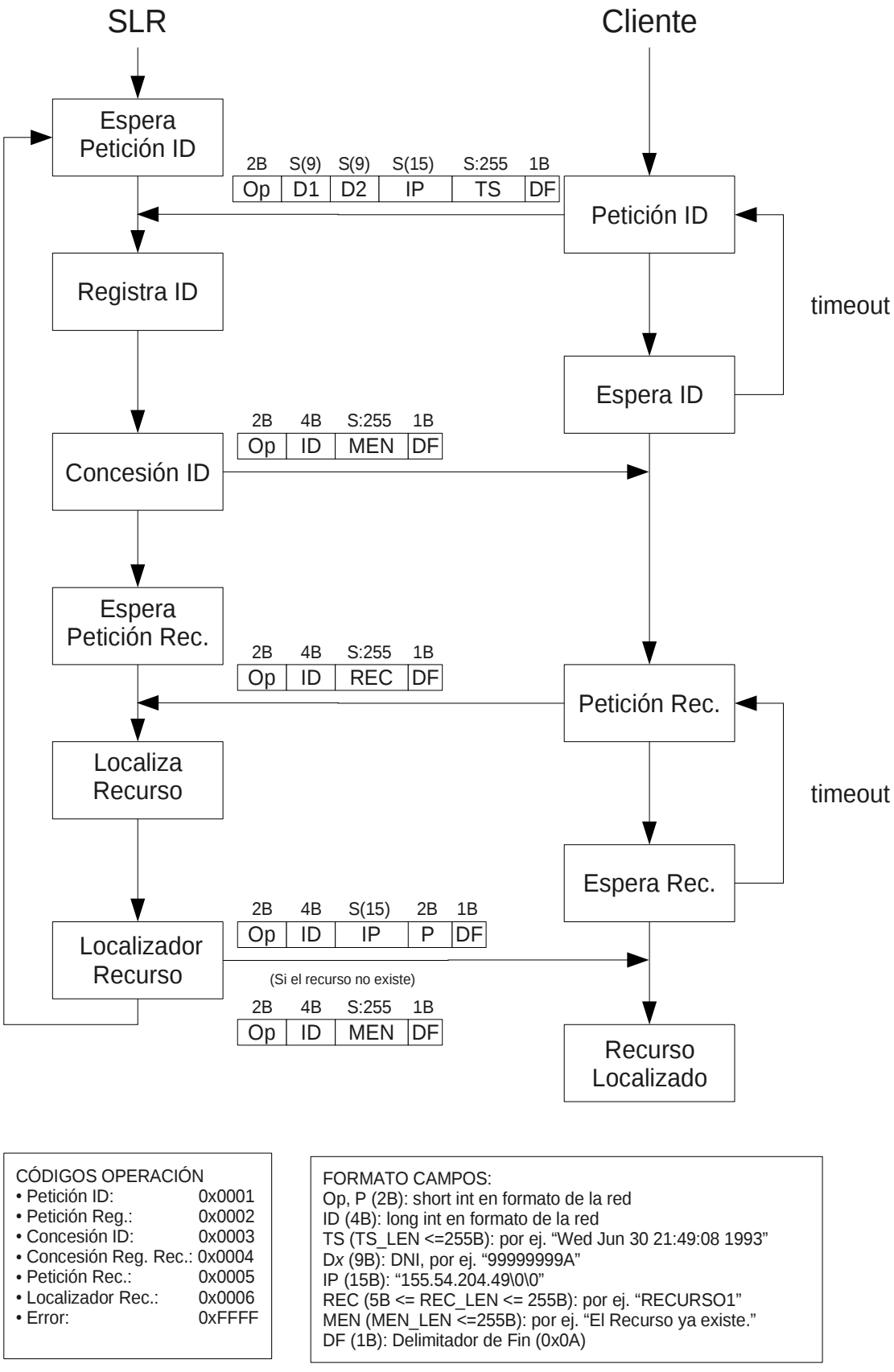


Figura 2: Diagrama de flujo para la localización de recursos