

programacion de servicios y procesos

Actividad 2 tema 3



carles ricart

# ¿Cuales son las tres categorías de los puertos lógicos de red?

**Puertos bien conocidos**: Puertos del 0 al 1023 son puertos reservados para el sistema operativo y usados por "Protocolos Bien Conocidos" como por ejemplo HTTP (servidor Web), POP3/SMTP (servidor de e-mail), Telnet y FTP.

**Puertos registrados**: Comprendidos entre 1024 y 49151 son denominados "registrados" y pueden ser usados por cualquier aplicación

**Puertos dinamicos o privados**: Comprendidos entre los números 49152 y 65535 son denominados dinámicos o privados, normalmente se asignan en forma dinámica a las aplicaciones de clientes al iniciarse la conexión. Su uso es poco común, son usados en conexiones peer to peer (P2P).

# Explica la necesidad de cierre de los sockets.

Por una parte, siempre es mejor cerrar cualquier canal de comunicación que hayamos dejado de utilizar porque ha cumplido ya con su función, ya que de esta manera, liberamos los recursos que ya no necesitamos.

Por otra parte, es una cuestión de seguridad, ya que existe un riesgo potencial, si dejamos abierto un canal de comunicación que ya no utilizamos, sin control.

Además, cerrando los sockets, terminamos el proceso con orden.

En el caso de los sockets orientados a conexión, además de los sockets propiamente dichos, tendremos que cerrar los *streams* de entrada/salida que se hayan utilizado.

Además, el orden de cierre es fundamental.

En el caso de los sockets orientados a conexión (TCP), el procedimiento sería (considerando que 1º hemos abierto el *stream* de recepción, y luego el de envío):

En el lado del cliente, tendremos el socket y los *streams* utilizados para la comunicación, por lo que el orden de cierre sería:

1. Cierre del *stream* de envío, p. ej.: out.close();

2. Cierre del *stream* de recepción, p. ej.: in.close();

3. Cierre del socket, p. ej.: socket.close();

En lado del servidor, tendremos el socket propio (*ServerSocket*), y el socket utilizado para la comunicación con el cliente (creado mediante el método *accept* del *ServerSocket*), además de los *streams*, por lo que el orden de cierre sería:

1. Cierre del *stream* de envío, p. ej.: out.close();

2. Cierre del *stream* de recepción, p. ej.: in.close();

3. Cierre del socket de comunicación, p. ej.: socketCliente.close();

4. Cierre del socket del servidor, p. ej.: socketServidor.close();

En el caso de los sockets no orientados a conexión (UDP), el procedimiento sería, tanto para el cliente, como para el servidor:

Cierre del DatagramSocket,

socket.close();