Comunicación en Red

Hoy en día es muy común que las aplicaciones hagan uso de información, recursos u otras aplicaciones que se encuentran en lugares **remotos**. Para poder hacer uso de ellas, se recurre a **redes de comunicación**, como Internet. Por tanto, se hace necesario conocer la forma de implementar sistemas que permitan dicha comunicación.

Fundamentos

Existen diversas formas o **arquitecturas** para implementar la comunicación entre varios sistemas. La más utilizada es sin lugar a dudas la Cliente / Servidor, en la que una máquina normalmente muy potente centraliza el trabajo de otras tantas máquinas menos capaces. La idea principal detrás de esta arquitectura es que un **Servidor** ofrezca una serie de servicios a sus **Clientes**, que serán los consumidores de dichos servicios.

Para que exista una comunicación entre dos sistemas informáticos deben existir los siguientes elementos:

- Un **emisor** y un **receptor**.
- Un **mensaje**.
- Un **canal** de comunicación.
- Un lenguaje común (**reglas**) para poder comunicarse.

Dentro de estas reglas se tienen que considerar cosas tan variadas como:

- Mecanismo para **identificar** de forma unívoca a cada interlocutor, sobre todo si va a haber más de uno.
- Un sistema para asegurar la **integridad** de los mensajes.
- Un sistema para garantizar que el **orden** de los mensajes es el correcto.
- Un sistema para **confirmar** la recepción del mensaje, **notificar errores**, etc.

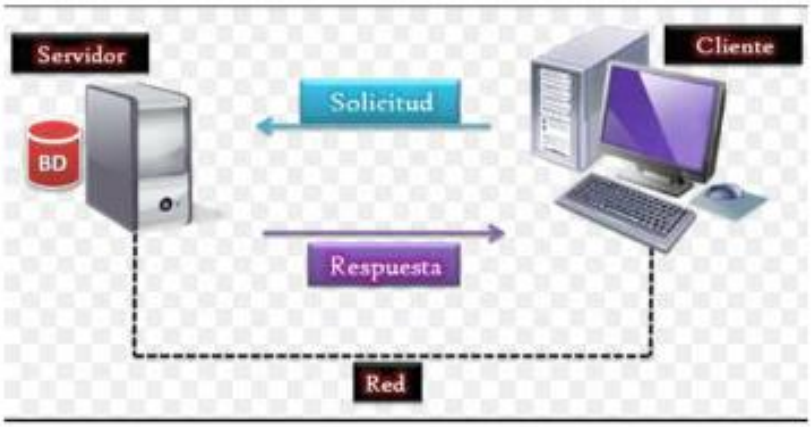
En una palabra, lo que necesitamos es un **Protocolo**. Definimos protocolo como el conjunto de reglas que permiten la comunicación entre ambos interlocutores: qué enviamos por la red, qué respuestas posibles dará el servidor, qué haremos si hay errores…

El conjunto de protocolos sobre el que se construye Internet se llama **Familia de Protocolos de Internet**. Garantizan todo tipo de servicios, desde el acceso a páginas web como ftp, correo electrónico, etc. Son todos **protocolos abiertos**, lo que significa que no pertenecen a ninguna empresa ni organización. Los tres protocolos base de la comunicación en red son **IP**, **TCP** y **UDP**; que además soportan otros protocolos de más alto nivel como **HTTP** o **FTP**.

En Java, para establecer una comunicación sencilla de bajo nivel entre dos sistemas se usa una clase **socket**. Permite establecer una comunicación bidireccional entre dos puntos.

Arquitectura Cliente / Servidor

En esta arquitectura, existe un proceso central (el **Servidor**) que ofrece servicios a uno o más procesos **Clientes**. En proceso Servidor debe estar alojado en una máquina accesible en la red, y debe de ser conocida por los Clientes. Cuando un Cliente requiere de sus servicios, se conecta con el Servidor, iniciando el proceso de comunicación.



Un ejemplo de esta arquitectura es cuando colocamos en un Servidor Web (Tomcat, por ejemplo) una serie de páginas HTML. Cuando un Navegador (Cliente) accede al Servidor, éste envía de vuelta al Cliente las páginas web solicitadas. Por último, el Navegador se dedica a 'montar' el Html, css, imágenes, etc. y nos muestra el resultado por pantalla.

En general, son los **Clientes**, quienes llevan a cabo todo el proceso de interacción con el usuario, quienes inician los procesos de comunicación con los Servidores y realizan las operaciones adecuadas para solicitarles los servicios. Finalmente, presentan al usuario los resultados de la ejecución. La mayor parte del código de un Cliente se encuentra dedicado a la visualización de la información al usuario, y a verificar que no haya errores a la hora de enviar solicitudes.

Los **Servidores** usualmente son pasivos, se limitan a escuchar solicitudes de conexión de los Clientes e implementan los servicios que son capaces de proporcionar, devolviendo los resultados de su ejecución. Generalmente no disponen de interfaz gráfica pero sí van acompañados de programas que facilitan su gestión, administración y control de rendimiento.

Protocolos

La comunicación entre dos sistemas es un proceso complejo en el que intervienen muchos elementos desde diferentes niveles de abstracción. A niveles más bajos, debe de existir un medio físico (cable) que mantenga unidos ambos equipos. A niveles más altos, ciertos programas permiten identificar un equipo y enviar/recibir información con la red.

El modelo de interconexión de sistemas abiertos conocido como **OSI** es el estándar que define esos noveles de abstracción. Define un total de siete, y es la referencia – de libro – que se toma cada vez que se desea construir un sistema de comunicación en red nuevo. El modelo OSI nos dice qué hay que hacer en cada capa, pero no nos dice cómo. Con el tiempo, a partir del modelo OSI se creó el modelo **TCP/IP**, que está mucho más extendido por su amplia implantación comercial.

El modelo **TCP/IP** Tiene cuatro capas o niveles de abstracción:

1. **Capa de aplicación**: en este nivel se encuentran las aplicaciones disponibles para los usuarios. Contempla los protocolos FTP, SMTP, Telnet, SSH, **HTTP**, POP3, DHCP, DNS...

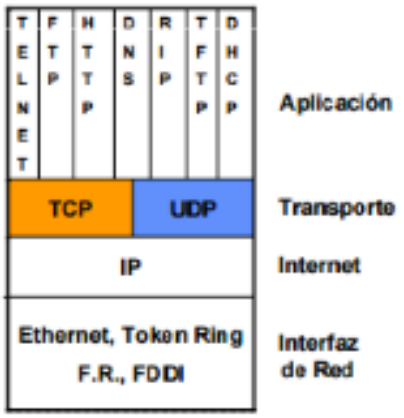
2. **Capa de transporte**: suministra a las aplicaciones servicio de comunicaciones extremo a extremo utilizando 2 tipos de protocolos: **TCP** (Transmission Control Protocol) y **UDP** (User Datagram Protocol).

3. **Capa de red**: tiene como propósito seleccionar la mejor ruta para enviar paquetes por la red. El protocolo principal que funciona en esta capa es **IP** (Internet Protocol).

4. **Capa de enlace**: es la interfaz con la red real. Recibe los datagramas de la capa de red y los transmite al hardware de la red. Ejemplos: Ethernet, Token Ring, FDDI...

TCP

Es un Protocolo basado en la conexión, garantiza que los datos enviados desde un extremo de la conexión llegan al otro extremo, en el mismo orden que fueron enviados y de forma correcta. Se emplea TCP cuando se hace necesario garantizar la integridad de la información transmitida. TCP es usado por protocolos como FTP. Al estar basado en conexiones, se crean canales de comunicación estables entre nodos de la red.



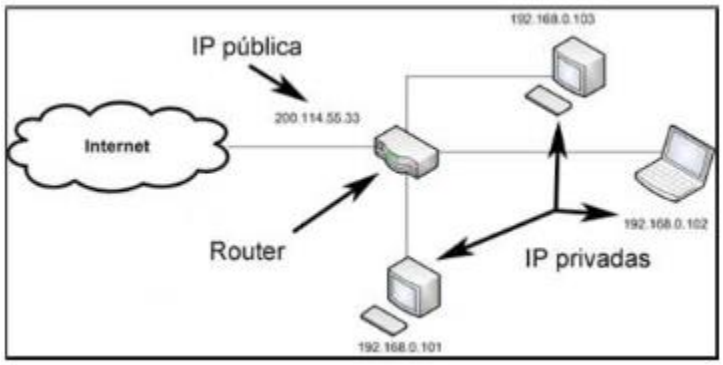
UDP

No está basado en la conexión, por lo que no se crean canales estables de comunicación entre nodos. Los paquetes de datos (datagramas) se envían de forma independientes, y muchas veces alcanzan su destino por diferentes caminos. El orden de entrega de los paquetes por lo tanto no es el esperado, y no se garantiza tampoco la recepción de todos los paquetes enviados. Esta falta de garantías tiene sus ventajas, dado que permite una comunicación mucho más rápida a cambio de aceptar la pérdida de información. Es un protocolo muy usado para transmitir voz o vídeo en tiempo real.

IP

Tanto TCP como UDP hacen uso de IP para el envío de información entre nodos de red. IP no está orientado a la conexión y no garantiza el envío de los paquetes transmitidos. Permite identificar los equipos dentro de una red mediante las **Direcciones IP**. Actualmente existen dos versiones de este protocolo: **IPv4** e **IPv6**. Debido al tamaño de las direcciones de IPv4 (32 bits) se permite identificar a más de cuatro mil millones de dispositivos; algo que con el tiempo se demostró insuficiente. IPv6 (128bits) permite unos dieciséis trillones de direcciones diferentes.

Para saber cuál es la dirección IP de nuestro ordenador en la red se utiliza **ipconfig** (Windows) o **ifconfig** (Ubuntu).



Puertos

Gracias a la dirección IP es posible enviar un paquete de datos hacia una máquina destino y efectuar la entrega correctamente. Ahora bien, en dicha máquina puede haber una gran cantidad de procesos efectuando tareas y esperando recibir información. Para determinar cuál de los procesos es el destinatario de los datos, se utilizan los **puertos**.

Cada proceso que lleva a cabo tareas de comunicaciones en una máquina tiene asociado un número de puerto para identificarlo. En un entero positivo de 16 bits, por lo tanto, en el rango de 0 a 65535. Cuando un **Cliente** se debe conectar a un proceso **Servidor** debe conocer su puerto. Tradicionalmente, cuando las aplicaciones en red eran pocas, se acordó mantener un rango de números de puertos reservados y fijos. Hoy en día aún se mantiene la costumbre:

Servidor	Puerto
SMTP	25
POP3	110
HTTP	80
FTP (Datos)	20
FTP (Control)	21