MEDAC
Instituto Oficial de Formación Profesional

PROGRAMACIÓN DE SERVICIOS Y PROCESOS
TÉCNICO EN DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Introducción a la comunicación entre aplicaciones

07

ÍNDICE

© MEDAC 978-84-18864-98-8
Reservados todos los derechos. Queda rigurosamente prohibida, sin la autorización escrita de los titulares del copyright, bajo las sanciones establecidas en las leyes, la reproducción, transmisión y distribución total o parcial de esta obra por cualquier medio o procedimiento, incluidos la reprografía y el tratamiento informático.

OBJETIVOS

Conocer el modelo OSI, con el que aprenderemos:

- ***El protocolo o modelo TCP/IP***
- ***El protocolo TCP***
- ***El protocolo UDP***

Comprender el concepto de puerto

Conocer varias bibliotecas para el uso de comunicaciones

/ 1. Introducción y contextualización práctica

En esta unidad, aprenderemos sobre la comunicación entre diferentes aplicaciones por medio de protocolos de red.

Veremos qué es el modelo OSI y el protocolo de comunicaciones TCP/IP, así como sus usos y cómo se dividen en diferentes capas.

También estudiaremos los dos protocolos de comunicación de red que se utilizan para cualquier tipo de comunicación entre varios dispositivos a través de internet, los protocolos TCP (que no se deben confundir con el modelo TCP/IPC) y UDP. Veremos qué son y cuáles son sus diferencias y similitudes.

Seguidamente, vamos a conocer el concepto de puerto, qué es y cuál es su papel fundamental en las comunicaciones.

Por último, veremos una serie de bibliotecas que nos ofrece Java para poder realizar las comunicaciones en red.

Todos son nuevos conceptos. Hemos salido de los hilos para adentrarnos en las comunicaciones. No obstante, aunque sean conceptos distintos, ya verás cómo los relacionaremos con la programación, los servicios y los procesos.

Escucha el siguiente audio, en el que planteamos el caso práctico que iremos resolviendo a lo largo de esta unidad.

Fig. 1. Comunicaciones.

Audio Intro. "Comunicación entre aplicaciones"

https://bit.ly/2CobfQa

/ 2. El modelo OSI

Después del surgimiento de Internet, sobre el año 1980, las redes comenzaron a crecer de forma exponencial, ya que muchas empresas lo consideraron una gran oportunidad para expandir sus negocios. Como las redes estaban empezando a surgir, cada una de estas contaban con sus propias especificaciones para funcionar, haciendo que la comunicación entre todas las redes fuese una tarea prácticamente imposible.

Debido a esto, la Organización Internacional para la Estandarización (ISO) empezó a estudiar la forma de arreglar este problema, investigando cómo podían hacer que las redes se comunicaran entre sí. Todo esto llevó a una serie de reglas, que se estandarizarían bajo el nombre de modelo OSI, las cuales dictaban cómo debían funcionar las redes y cómo debían ser sus comunicaciones, haciendo que la interconexión de ellas fuera un paso trivial, ya que todas trabajaban de la misma forma.

Seguro que ya conoces, de otras asignaturas, que el modelo OSI está formado por siete capas que dividen todo el funcionamiento y los estados por los que deben pasar los datos para viajar de una red a otra.

Las capas del modelo OSI son:

- **Capa física:** Esta capa será la encargada de manejar la topología de la red y las conexiones del ordenador, por ejemplo. Podemos decir que gestiona todo el hardware necesario.
- **Capa de enlace de datos:** Será la encargada de realizar el direccionamiento físico de los datos que se envíen. Detectará errores, controlará el flujo y hará que los paquetes lleguen ordenados.
- **Capa de red:** Esta capa enrutará las redes. Su objetivo principal es hacer que los datos lleguen desde su origen a su destino.
- **Capa de transporte:** Como su nombre indica, llevará a cabo el transporte de los datos.
- **Capa de sesión:** Su cometido pasa por mantener la conexión entre dos equipos, reanudándola en caso de interrupción.
- **Capa de presentación:** Será la responsable de la presentación de la información. Aquí se tratan temas como semántica y sintaxis de los paquetes que se transmiten.
- **Capa de aplicación:** Encargada de acceder a los servicios que proveen las demás capas. Aquí es donde se definen los protocolos que usaremos.

Audio 1. "No confundir OSI con TCP/IP"
https://bit.ly/3kEpMbN

/ 3. El modelo TCP/IP

Cualquier red está separada en capas, conocidas también como layers, en inglés. La comunicación entre dichas capas estará controlada por protocolos, los cuales indicarán cómo tienen que ser, de qué datos tienen que constar, cómo deberán enviar dichos datos, cómo deberán recibirlos, etc.

El protocolo más común es el dictado por el modelo TCP/IP, el cual consta de las siguientes capas:

- **Capa de aplicación:** Esta capa está compuesta por aplicaciones de red, las cuales usarán los niveles más inferiores para poder transferir mensajes entre ellas mismas. Algunos ejemplos de protocolos que trabajan en esta capa son:
    - **HTTP:** Este protocolo es el encargado de definir la manera en la que se van a comunicar los servidores y los navegadores web.
    - **SMTP:** Este protocolo es el encargado de definir la manera en la que se gestiona el correo electrónico.
    - **DNS:** Es el que traduce a direcciones IP los nombres de los dispositivos que se encuentra en la red.
    - **FTP:** Posibilita las transferencias de ficheros.
    - **NFS:** Permitirá que podamos compartir ficheros en diferentes ordenadores de una red.
    - **TELNET:** Posibilitará la conexión remota de terminales.
- **Capa de transporte:** Esta capa está compuesta por todos aquellos elementos software cuya función es crear el canal de comunicación, descomponer el mensaje que hayamos enviado en diferentes paquetes y gestionar la transmisión del mismo entre el emisor y el receptor. Aquí es donde actuarán los protocolos TCP y UDP.
- **Capa de Internet:** Esta capa está compuesta por todos aquellos elementos software encargados de dirigir los paquetes por la red; además, se asegurarán de que dichos paquetes lleguen a su destino.
- **Capa de red:** La forman todos aquellos elementos hardware de comunicaciones, tarjetas de red, cables, etc., y es la encargada de trasmitir todos los paquetes de información. Debemos tener en cuenta que los protocolos tienen que conocer los detalles físicos de la red para un correcto envío de los paquetes.

/ 4. Caso práctico 1: "¿Cómo se identifican las aplicaciones en diferentes PC?"

Planteamiento: Pilar y José están estudiando en qué consiste la comunicación de aplicaciones en red. Ya conocen los dos modelos en los que se basa toda comunicación: OSI y TCP/IP.

Una vez comprenden estos modelos, pueden empezar a indagar en cómo pueden realizar la comunicación entre varias aplicaciones que están en diferentes PC dentro de la red. «Hay una cosa que no me cuadra: si las aplicaciones están en diferentes ordenadores, ¿cómo podemos enviar o recibir el mensaje al ordenador donde está la aplicación?», le comenta Pilar a José.

Nudo: ¿Cómo crees que se podrán enviar los mensajes a los PC donde se encuentran las aplicaciones?

Desenlace: Es normal que, al principio, las aplicaciones en red nos resulten extrañas, ya que no estamos acostumbrados a ellas y siempre hemos realizado aplicaciones simples que se comunican con el usuario.

Si pensamos un poco, estas aplicaciones las vamos a tener en diferentes ordenadores que están conectados en red, seguramente una red local, y cada uno de estos ordenadores se va a identificar de forma única en dicha red, mediante su dirección IP.

A través de las diferentes direcciones IP de los ordenadores, podremos realizar las comunicaciones, ya que, cuando enviemos un mensaje a otra aplicación, lo tendremos que enviar a la dirección IP del ordenador donde se encuentra, y, de la misma forma, cuando nos envíen a nosotros un mensaje, lo deberán hacer a la dirección IP que tenga nuestro ordenador.

/ 5. Protocolos de comunicaciones: protocolo TCP

Ya hemos visto que en la capa de transporte del modelo TCP/IP y en la capa de transporte del modelo OSI, es donde se realiza todo lo relacionado con la transferencia de datos y corrección de errores de estos, entre el emisor y el receptor de la comunicación.

Como seguro que habéis adivinado, **la misión es proporcionar un transporte de información confiable entre el emisor y el receptor**, que sea totalmente independiente de la capa física que se esté utilizando en la misma.

Esto puede realizarse, por ejemplo, mediante el protocolo TCP (Protocolo de Control de Transmisión o Transmission Control Protocol en inglés), que es un protocolo orientado a conexión que creará un flujo de transmisión de datos entre el origen y el destino, que garantizará que la información se entregue sin errores. Este protocolo parte el mensaje que se quiere enviar en paquetes, enviándolos por el canal de comunicación.

A estos paquetes les irá asignando un número para que, una vez lleguen a su destino, puedan ser reconstruidos, volviendo a tener de una pieza el mensaje que se envió.

Otra tarea del protocolo TCP será controlar el flujo del canal de comunicación. Con esto, controlará si hay más o menos tráfico, haciendo que la red no se sature y evitando que un receptor que sea lento en el proceso de recepción de paquetes quede saturado por un emisor que sea muy rápido enviando los mensajes.

Este protocolo es **fiable**, lo cual va a hacer que se garantice la llegada de los paquetes al receptor.

Al contar con todas estas características, el protocolo TCP no es un protocolo sencillo de implementar, ya que tiene que cubrir muchos aspectos en el transporte de la información.

Ejemplos de protocolos que usan TCP son HTTP, FTP, Telnet, etc.

Fig. 2. Protocolo TCP.

Vídeo 1. "Correspondencia entre OSI y TCP/IP"
https://bit.ly/31E49j2

/ 6. Protocolos de comunicaciones: protocolo UDP

Además del protocolo TCP, contamos con otro protocolo llamado UDP (Protocolo de Datagramas de Usuario o User Datagram Protocol en inglés), el cual es un protocolo que no está orientado a conexión. Esto implica que no va a tener ningún tipo de sincronización para el envío de mensajes entre el emisor y el receptor.

Este protocolo se utiliza, principalmente, para **aplicaciones que no van a necesitar asignación de control de secuencia ni de control de flujo en las transmisiones que se hagan.**

Su funcionamiento se basa en **la partición del mensaje que se quiere enviar en datagramas, enviándolos por el canal de comunicación, sin control ninguno, pudiendo llegar o no al destinatario.** Al contrario de lo que pasaba con el protocolo TCP, en UDP no hay una numeración de cada datagrama para que más adelante se puedan unir y reconstruir el mensaje original.

Este protocolo se va a utilizar en aplicaciones en las que prime más la velocidad de entrega de paquetes, las conocidas como aplicaciones en tiempo real, como pueden ser aplicaciones de streaming o transmisión de voz.

Este protocolo no es fiable, ya que no garantiza la llegada de todos y cada uno de los datagramas enviados por el emisor del mensaje al receptor. Además, no realiza ningún tipo de control de flujo en las comunicaciones.

Al no realizar ni control de flujo ni de errores, el protocolo UDP es mucho más rápido que TCP en la transmisión de los datos de la comunicación. Además, esto también implica que será mucho menos complejo de implementar.

Este protocolo se encuentra en la capa de transporte.

Ejemplos de protocolos que usan UDP son DHCP, BOOTP, DNS, etc.

Fig. 3. El protocolo UDP es parecido a un río caudaloso. Veloz, pero no fiable.

Vídeo 2. "Clases para TCP y UDP en Java"

https://bit.ly/31ICjlH

/ 7. Los puertos

Ya hemos visto que, para lograr que la información vaya de un extremo a otro de la comunicación, deberemos utilizar la capa de red y enviar esos paquetes o datagramas a la dirección IP del destinatario.

Ciertamente, el proceso no es tan sencillo, ya que tendremos que indicar a dónde van dirigidos dichos paquetes. Para esto, podemos definir una serie de direcciones de transporte en las que los procesos de nuestras aplicaciones puedan estar de forma constante a la escucha, por si llega algún paquete o datagrama. Esto es lo que conocemos como **puertos.**

La gran mayoría de puertos se va a asignar de forma aleatoria, sin ningún tipo de orden, simplemente se asignará el primero que se encuentre libre.

No obstante, hay ciertas aplicaciones que sí tienen un puerto ya asignado para su funcionamiento, los cuales no podrán ser usados en las asignaciones aleatorias o en asignaciones manuales que hagamos nosotros, ya que eso llevaría a diversos errores en la transmisión de la información a través de la red.

Quien se encarga de asignar los puertos predefinidos a las aplicaciones que así lo necesiten es la Autoridad de Asignación de números de Internet o IANA, por sus siglas (Internet Assigned Numbers Authority).

La IANA tiene definidos los siguientes rangos de puertos:

- **Puertos conocidos:** Estos puertos son los que están reservados para aplicaciones estándar y van desde el puerto 0 al 1023. Algunos ejemplos de estos son el puerto 21 para el protocolo FTP, el puerto 80 para el protocolo HTTP, etc.
- **Puertos que están registrados:** Estos puertos han sido asignados para servicios o aplicaciones específicas y van desde el puerto 1024 al 49151. Este será el rango de puertos que deberemos utilizar para desarrollar nuestras aplicaciones.
- **Puertos dinámicos:** Estos puertos no están registrados para ningún servicio, sino que su uso es para atender a conexiones temporales entre diferentes aplicaciones y van desde el puerto 49151 al 65535.

Fig. 4. Comunicación de red.

/ 8. Caso práctico 2: "El esquema cliente/servidor"

Planteamiento: Una vez Pilar y José comienzan a entender el funcionamiento de los diferentes protocolos relativos a la comunicación en red, empiezan a entender que es posible realizar dicha comunicación entre aplicaciones.

«Tiene que haber algún modelo a seguir en este tipo de comunicaciones que nos tenemos controlado», le comenta José a Pilar, a lo que ella le responde que está de acuerdo, no cree que los mensajes se envíen a una dirección sin más.

Nudo: ¿Qué piensas al respecto? ¿Crees que nuestros amigos tienen razón y que hay que seguir algún esquema en el diseño de estas aplicaciones?

Desenlace: Cuando desarrollamos aplicaciones que hacen uso de las comunicaciones en red, no podemos hacerlo sin ningún tipo de orden y enviar los mensajes sin más: debemos seguir un protocolo estándar para este tipo de aplicaciones.

Estas aplicaciones se van a desarrollar mediante lo que conocemos como el protocolo cliente/servidor. Este protocolo es muy sencillo de comprender, ya que vamos a tener dos ordenadores diferentes: uno hará de servidor y otro de cliente. Básicamente, el servidor será el que provea de servicios a los ordenadores que se los piden, los cuales serán los clientes.

Una vez comprendida esta idea, cualquier aplicación que use las comunicaciones en red estará dividida en estas dos partes: un servidor que escucha peticiones y provee de servicios y uno o varios clientes que se conectarán al servidor para pedir un servicio.

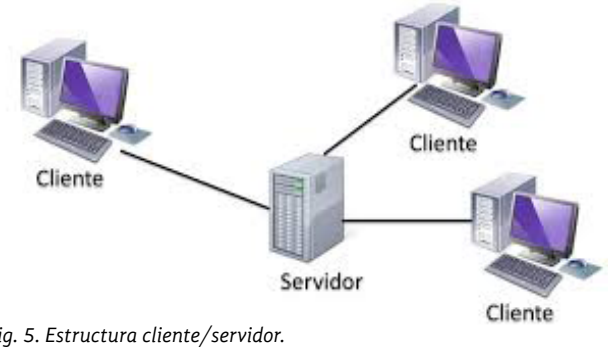

Fig. 5. Estructura cliente/servidor.

/ 9. Bibliotecas para networking en Java

El lenguaje de programación Java nos ofrece las siguientes clases y bibliotecas para poder desarrollar aplicaciones con comunicaciones en red:

- **InetAddress:** Esta clase nos va a permitir encontrar un nombre de dominio a partir de su dirección IP, y viceversa. Los objetos de esta clase tendrán dos elementos: el nombre del equipo y la dirección IP.

- **Socket:** Esta clase realiza o implementa la comunicación bidireccional entre un programa cliente y otro programa servidor, es decir, va a permitir tanto el envío como la recepción de mensajes. Si usamos objetos de esta clase, nuestros programas Java podrán comunicarse a través de la red de forma independiente de la plataforma. Esta clase se utilizará para clientes TCP.

- **ServerSocket:** Nos ayudará a implementar un socket que puede ser utilizado por los servidores para escuchar y aceptar peticiones de conexión de clientes. Esta clase se utilizará para servidores TCP y UDP.

- **DatagramSocket:** Con ella podremos implementar clientes que utilicen datagramas, siendo no fiables y no ordenados. Esta clase ofrece una comunicación muy rápida, ya que no hay que establecer la conexión entre cliente y servidor. Esta clase se utilizará para clientes UDP.

- **DatagramPacket:** Representará un datagrama, y contiene toda la información necesaria por el mismo: longitud de paquete, direcciones IP y número de puerto.

- **MulticastSocket:** Utilizada para crear una versión 'multicast' de la clase DatagramSocket. Mediante esta clase podremos enviar mensajes a múltiples clientes o servidores.

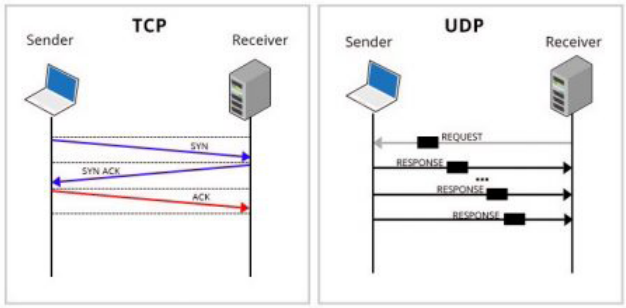

Fig. 6. Comunicación TCP y UDP.

/ 10. Resumen y resolución del caso práctico de la unidad

A lo largo de esta unidad, hemos comenzado a estudiar los fundamentos básicos de las comunicaciones en red de las aplicaciones.

Hemos visto los dos modelos fundamentales que existen en las comunicaciones en red de aplicaciones, los modelos OSI y TCP/IP, indicando en qué consisten, en qué capas se dividen y qué utilidad tiene cada una de ellas.

Seguidamente, hemos profundizado sobre los dos protocolos que se usan para realizar las comunicaciones en red entre aplicaciones, los protocolos TCP y UDP, indicando en qué consisten, y sus diferencias y similitudes.

También hemos aprendido qué son los puertos y el papel tan importante que ejercen en las comunicaciones en red.

Por último, hemos conocido una serie de bibliotecas para comunicaciones en red que nos ofrece el lenguaje de programación Java.

Resolución del caso práctico de la unidad

Hasta el momento, todas las aplicaciones que hemos realizado han consistido en un programa que interactúa con el usuario, pidiendo una serie de datos y mostrando el resultado de haberlos procesado, o a lo sumo, también, los datos se han podido obtener o almacenar en ficheros o una base de datos, pero nada más.

La forma de comunicación entre estas aplicaciones consiste en hacerlo mediante la red, por lo que las aplicaciones que se comunican pueden estar en diferentes equipos.

Esto se puede realizar fácilmente mediante sockets (que estudiaremos en la siguiente unidad), que nos permitirán hacer que las aplicaciones se comuniquen a través de la red mediante el envío de paquetes, en base a los protocolos estudiados.

/ 11. Bibliografía

Anexo: Puertos de red en Wikipedia, la enciclopedia libre. Recuperado el 15 de julio de 2020 de: https://es.wikipedia.org/wiki/Anexo:Puertos_de_red

Gómez, O. (2016). Programación de Servicios y Procesos Versión 2.1. https://github.com/OscarMaestre/ServiciosProcesos/blob/master/_build/latex/Servicios Procesos.pdf

Modelo de arquitectura del protocolo TCP/IP (Guía de administración del sistema: servicios IP). (s. f.). https://docs.oracle.com/. Recuperado el 15 de julio de 2020 de: https://docs.oracle.com/cd/E19957-01/820-2981/ipov-10/

Protocolo de datagramas de usuario en Wikipedia, la enciclopedia libre. Recuperado el 15 de julio de 2020 de: https://es.wikipedia.org/wiki/Protocolo_de_datagramas_de_usuario

Protocolo de control de transmisión en Wikipedia, la enciclopedia libre. Recuperado el 15 de julio de 2020 de: https://es.wikipedia.org/wiki/Protocolo_de_control_de_transmisi%C3%B3n

Sánchez, J. M. y Campos, A. S. (2014). Programación de servicios y procesos. Madrid: Alianza Editorial.

MEDAC