Actividad 1. Escenarios de red

PROGRAMACIÓN DISTRIBUIDA

Una razón para construir un sistema distribuido es compartir recursos.

Un sistema distribuido es un componente hardware o software dentro de máquinas que están unidos mediante red, comunican y coordinan sus acciones mediante el paso de mensajes. Esta definición tiene las siguientes consecuencias:

- **Concurrencia:** Ejecución de programas concurrentes.
- **Inexistencia de reloj global:** Los programas coordinan sus acciones mediante el paso de mensajes. Los relojes de los host no están sincronizados.
- **Fallos independientes:** Cada componente del sistema puede fallar independientemente sin afectar a los otros.

Una arquitectura típica es la arquitectura **cliente-servidor**.

- **Clientes:** Elementos activos que demandan servicios a los servidores.
- **Servidores:** Elementos pasivos que realizan tareas necesarias de los clientes.

Existen varios modelos de programación para la comunicación entre los procesos:

- **Sockets:** Proporcionar los puntos extremos de la comunicación entre procesos.
- **Llamada de procedimientos remotos (RPC):** Permite a un cliente llamar a un procedimiento de otro programa en ejecución en un servidor.
- **Invocación remota de objetos:** Un objeto en un proceso puede invocar métodos de un objeto que reside en otro proceso.

VENTAJAS E INCONVENIENTES

Ventajas

- Compartir recursos y datos
- Capacidad de crecimiento incremental
- Mayor flexibilidad
- alta disponibilidad
- Soporte de aplicaciones inherentemente distribuidas
- Carácter abierto y heterogéneo

Desventajas

- Aumento de complejidad
- Problemas con las redes de comunicación
- Problemas de seguridad

Actividad 2. Arquitectura TCP/IP

CLASES JAVA PARA COMUNICACIONES EN RED

TCP/IP es una familia de protocolos desarrollados para permitir la comunicación entre ordenadores de cualquier red o fabricante, respetando los protocolos de cada red individual. Tiene 4 capas:

- **Aplicación:** FTP, SMTP, Telnet, HTTP, etc.
- **Transporte:** Suministra a aplicaciones, servicio de comunicaciones extremo a extremo usando TCP y UDP.
- **Red:** Selecciona la mejor ruta para enviar paquetes por la red. Su protocolo principal es el IP.
- **Enlace / Interfaz de red:** Recibe los datagramas de la capa de red y los transmite al hardware de la red.

TCP es un protocolo basado en la conexión. Garantiza que los datos enviados desde un extremo llegue al otro extremo en el orden que se envia. En caso contrario, notificará de un error.

UDP no está basado en la conexión. Envía paquetes de datos independientes (*datagramas*) de una aplicación a otra sin importar el orden de entrega. No se garantiza la recepción de los paquetes.

El paquete **java.net** tiene clases e interfaces para la implementación de aplicaciones de red:

- **URL:** Puntero a un recurso en la Web.
- **URLConnection:** Admite operaciones más complejas en la URL
- **ServerSocket y Socket:** Dan soporte a sockets TCP
    - **ServerSocket:** Usado por el servidor. Crea un socket en el puerto en el que escucha peticiones de conexión.
    - **Socket:** Usado por cliente y servidor para comunicarse entre sí usando streams
- **DatagramSocket, MulticastSocket y datagramPacket:** dan soporte a la comunicación vía datagramas UDP.
- **InetAddress:** Representa direcciones de Internet.

LOS PUERTOS

Los protocolos TCP y UDP usan **puertos** para asignar datos entrantes a un proceso en particular.

Los datos transmitidos por internet tienen información de direccionamiento que identifica a la máquina y el puerto para el que está destinada. La máquina se identifica por su dirección IP y los puertos se identifican por un número que TCP y UDP utilizan para entregar datos a la aplicación correcta.

En una comunicación TCP, una aplicación de servidor vincula un socket a un número de puerto específico para poder registrar el servidor en el sistema y recibir todos los datos destinados a ese puerto. Una aplicación cliente podrá comunicarse con el servidor a través de ese puerto.

En comunicaciones basadas en datagramas, el paquete contiene el número de puerto de su destino y UDP enruta el paquete a la aplicación adecuada.