capa de transporte

Redes

Gabriel Infante Lopez

El Servicio de Transporte

- Direccionamiento y control de flujo
- Servicio Orientado a la conexión
- tres fases: establecimiento, transferencia de datos y liberación.

Para que sirve el servicio de transporte?

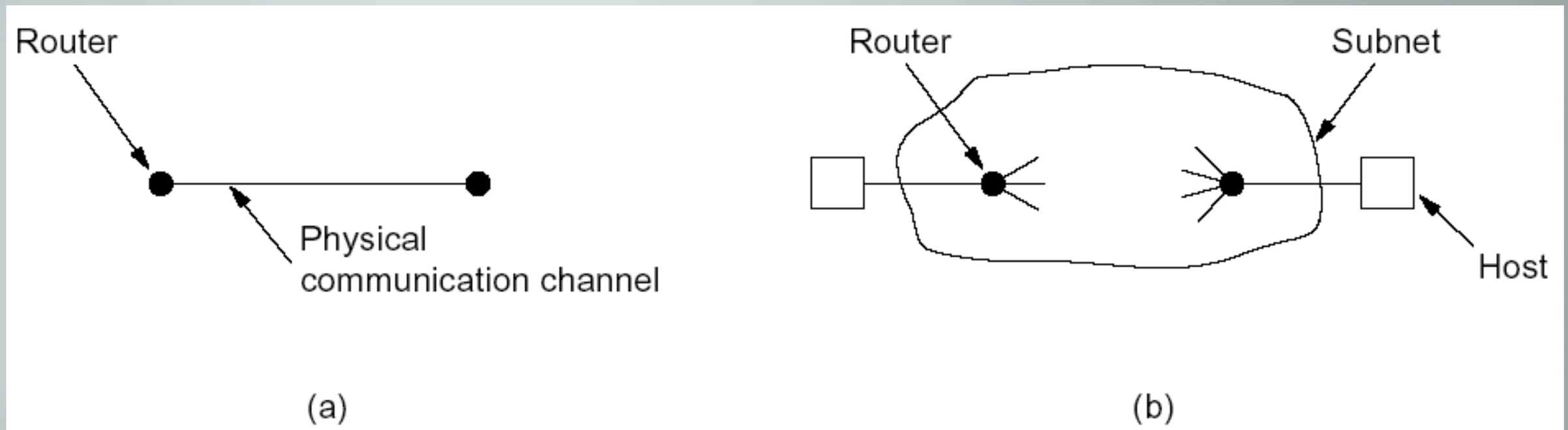
Quien corre cada uno de los protocolos?

Servicio de transporte

- El servicio de transporte es confiable.
- Un proceso manda algo por la red y otro proceso lo escucha tal cual fue enviado.
- el servicio de transporte es usado por programadores
- el servicio de transporte tiene que proveer primitivas fáciles de usar.

Transporte

Observación: Los protocolos de transporte se parecen a los de la capa de datos: e.g., control de errores y de flujo.



Las conexiones en la de datos son punto a punto

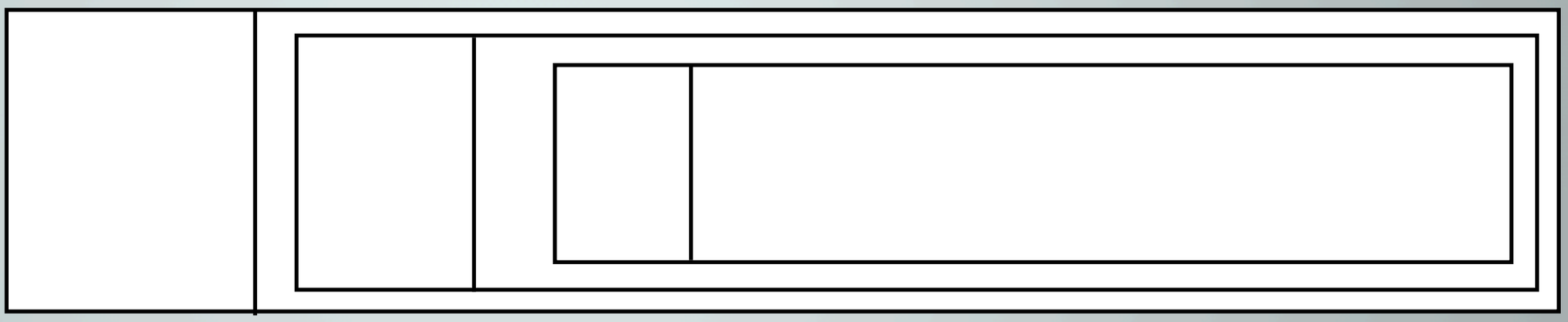
Un paquete no reaparece despues de haber viajado en la red

Paquete capa de transporte

Encabezado de la trama

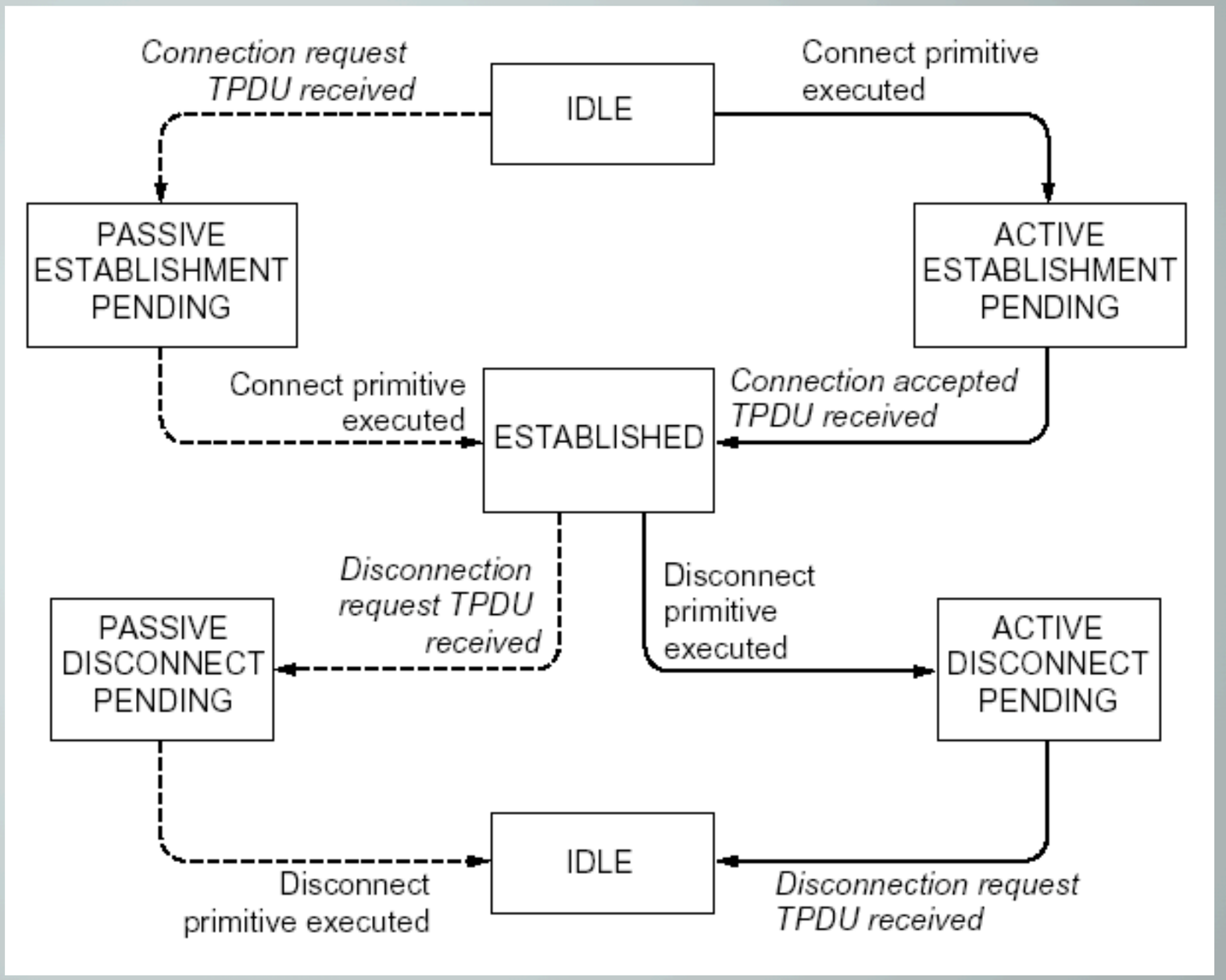
Encabezado del paquete

Encabezado de la TPDU



Unidad de Datos del Protocolo de Transporte (TPDU)

Conexiones



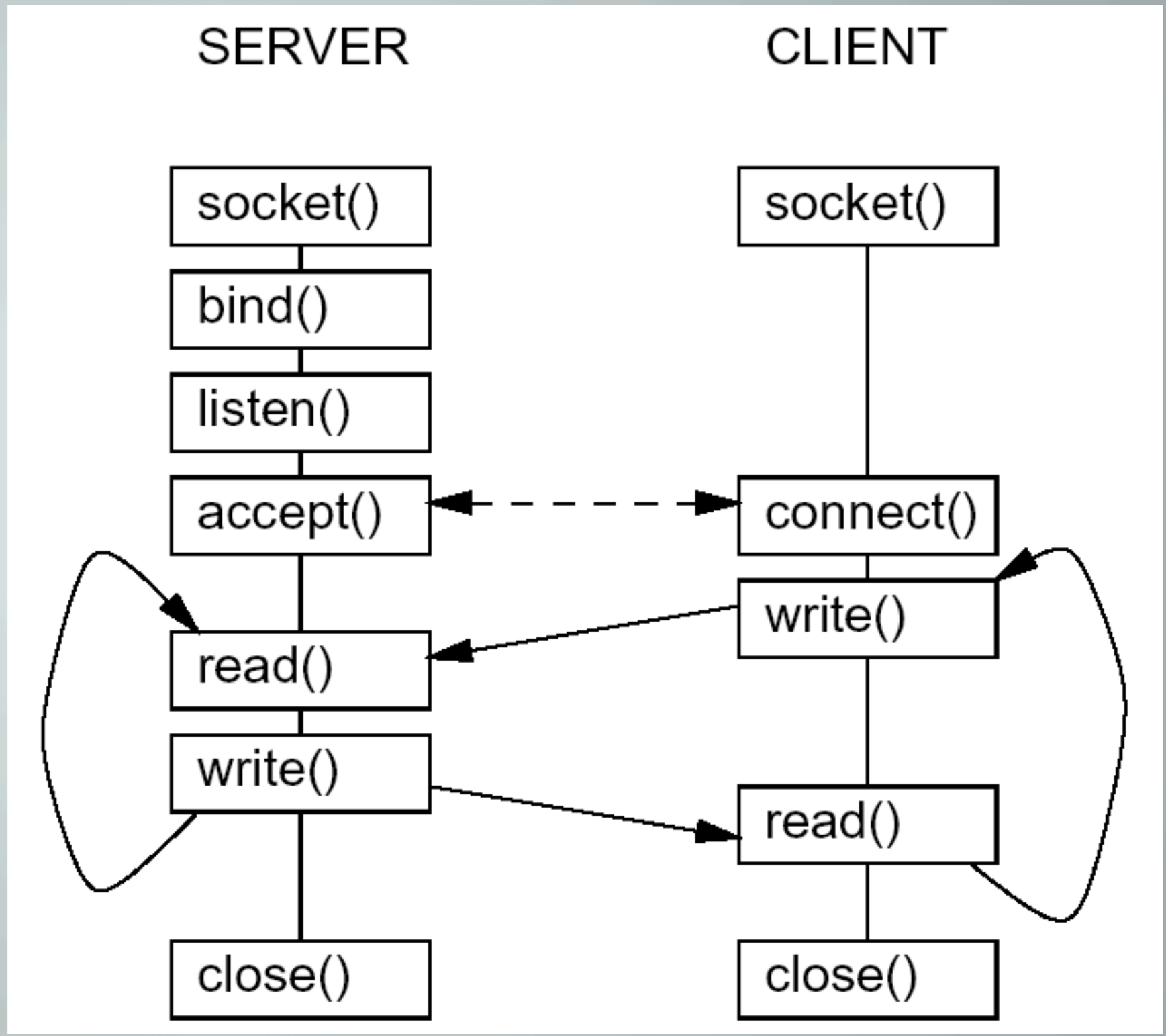
—— Cliente

····· Servidor

Primitivas Berkeley

Primitiva	Significado
LISTEN	Crea un nuevo punto terminal de comunicación
BIND	Adjunta una dirección local a un socket
LISTEN	Anuncia la disposición a aceptar conexiones; indica el tamaño de cola
ACCEPT	Bloquea al invocador hasta la llegada de intento de conexión
CONNECT	Intenta establecer activamente una conexión
SEND	Envia datos a través de la conexión
RECEIVE	Recibe datos de la conexión
CLOSE	Libera la conexión

Ejemplo


SERVER
CLIENT
socket()
bind()
listen()
accept()
connect()
write()
read()
write()
read()
close()
close()

Ejemplo Cont.

Sockets – Server Side

```
serverAddress : TransportAddress /* Publicly known address */
...
PROCESS Server IS
  clientSocket : Socket; /* Private socket */
  ...
  BEGIN
    serverSocket := NEW Socket;
    serverSocket.bind(serverAddress);
    serverSocket.listen(maxConnections);
    LOOP
        serverSocket.accept(clientSocket);
        clientSocket.read(request); /* receive */
        clientSocket.write(answer); /* send */
        clientSocket.close();
    END LOOP;
END Server;
```

Ejemplo Cont.

Sockets – Client Side

```
serverAddress : TransportAddress /* Publicly known address */
...
PROCESS Client IS
    clientAddress : TransportAddress; /* Private address */
    clientSocket : Socket;      /* Private socket  */
    ...
    BEGIN
    clientAddress := NEW TransportAddress;
    clientSocket := NEW Socket;

    clientSocket.bind(clientAddress);
    LOOP
        IF clientSocket.connect(serverAddress)
            THEN EXIT;
            ELSE sleep(1);
        END IF;
    END LOOP;

    clientSocket.write(request); /* send */
    clientSocket.read(answer); /* read */
    clientSocket.close();
    END Client;
```

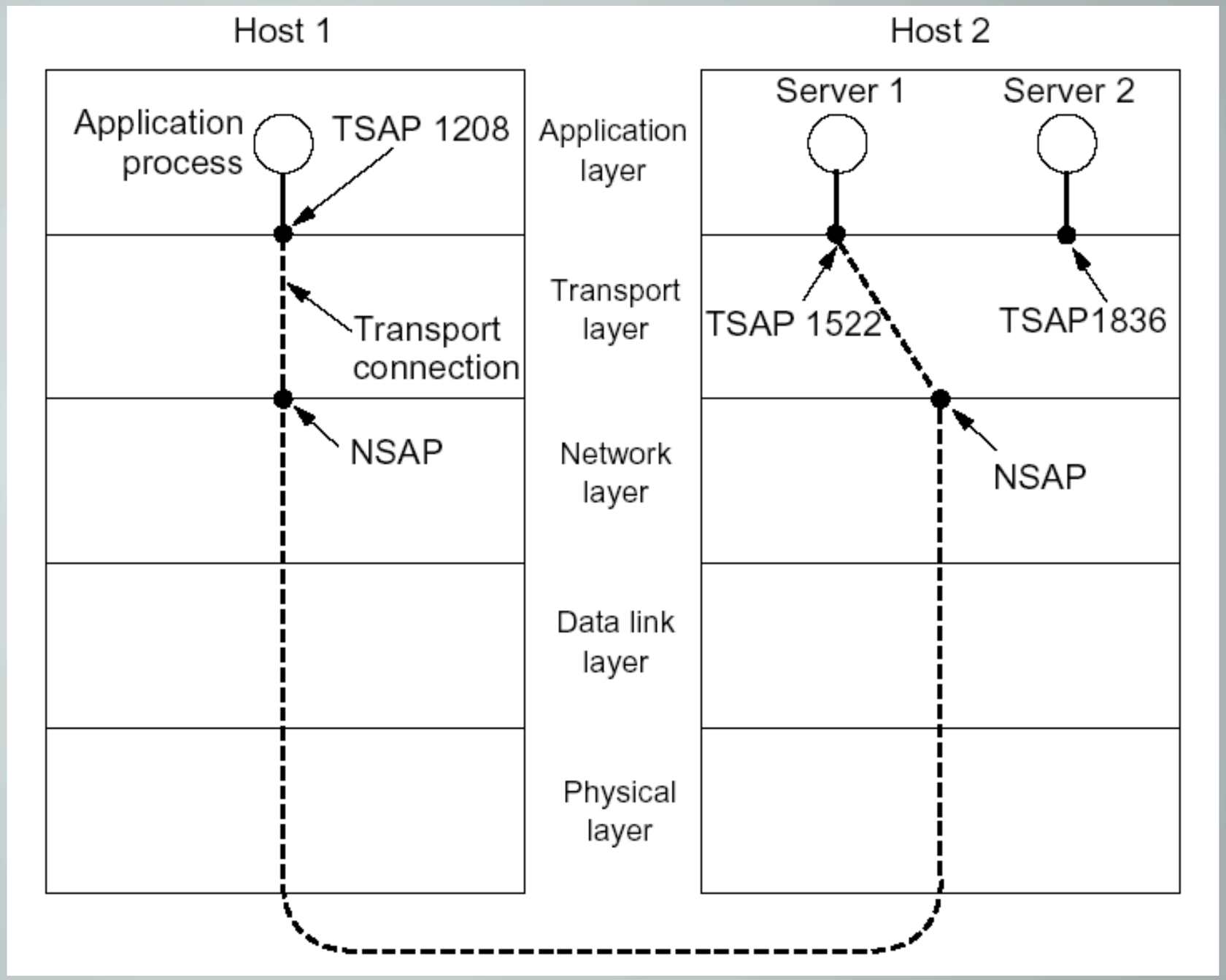
Elementos de la capa de transporte

- Direccionamiento
- Establecimiento de una conexión
- Liberación de una conexión
- Control de Flujo
- Multiplexión
- Recuperación de caídas.

Direccionamiento

- A quien mandar un paquete?
  - TSAP (Punto de acceso al servicio de Transportes), e.g., puertos
  - NSAP (Punto de acceso al servicio de Transporte), e.g., direcciones IP
  - Una direción en la capa de red es un numero de la capa red y un puerto

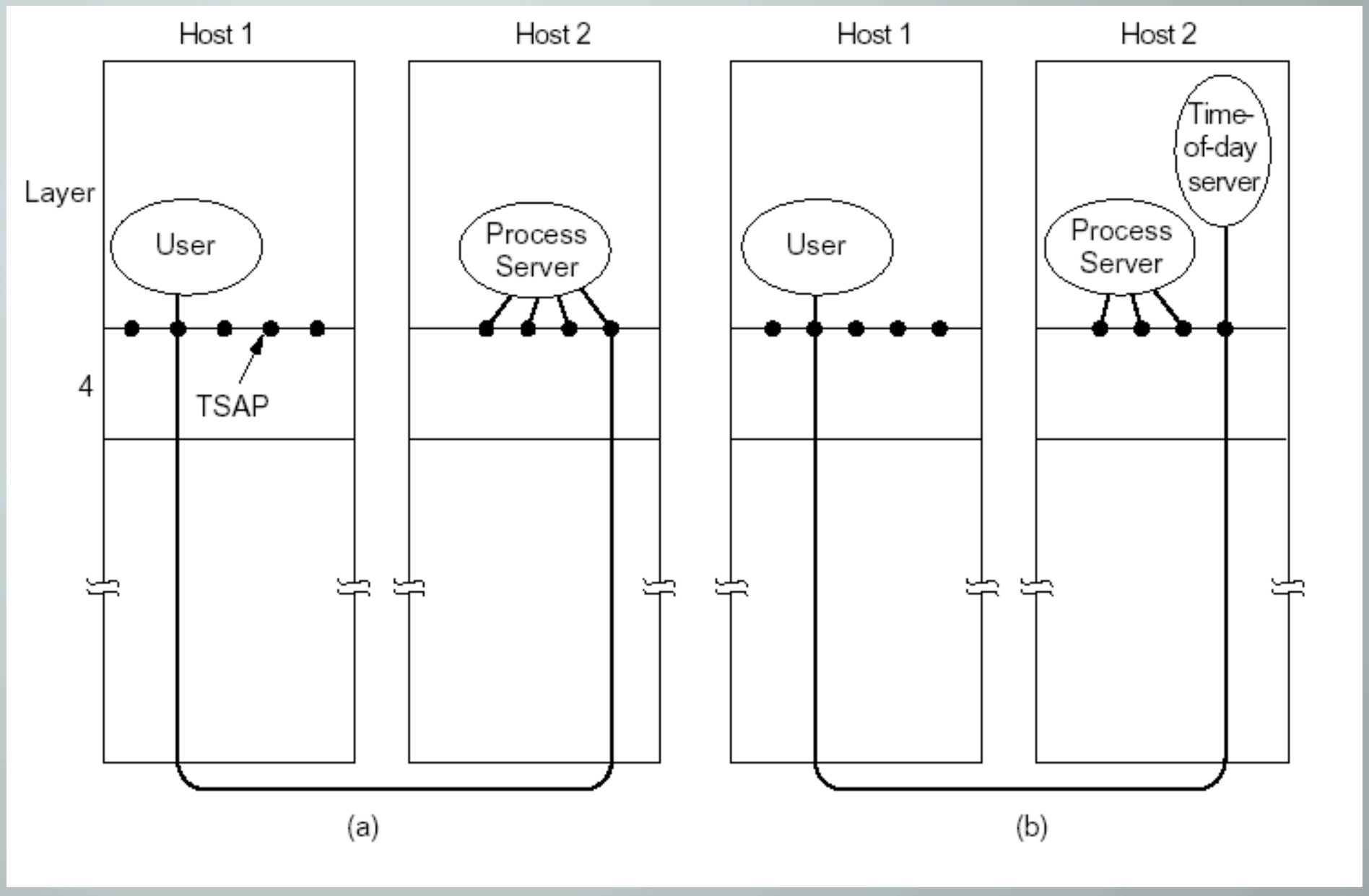
Ejemplo



Como sabemos el puerto correcto?

Direcciones Fijas para Servicios

Solución general: Mantener un solo proceso en un puerto fijo que maneje varias entradas (inetd)



Direcciones Desconocidas

Problema: Algunas veces no se puede tener un proceso que maneje todos los pedidos

Solución: Usar un servidor de nombres

Pregunta: Este servidor devuelve un TSAP, como encontramos la dirección NSAP?

Pregunta: En que nivel esta el servidor de nombres

Establecimiento de conexión

Liberación de Conexión

Control de flujo

Multiplexión

Recuperación de caídas

Transporte en Internet

UDP

TCP