Programación Avanzada (TC2025)

Tema 3. Administración de procesos

Instituto Tecnológico y de Estudios Superiores de Monterrey. Campus Santa Fe
Departamento de Tecnologías de Información y Electrónica
Dr. Vicente Cubells (vcubells@itesm.mx)

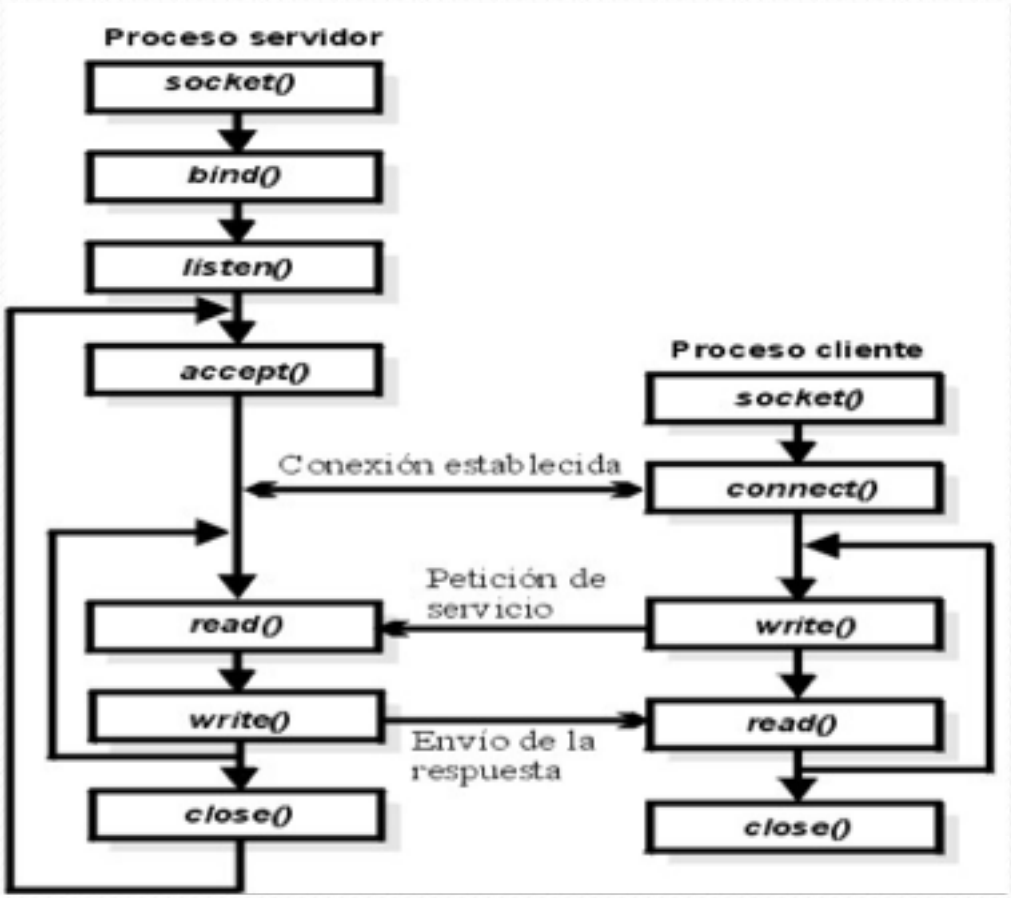
Temario

- Sockets
  - Definición
  - Arquitectura cliente-servidor
  - Tipos de direcciones
  - Funciones primitivas
  - Ejemplo

¿Qué es un socket?

- Es un mecanismo que provee comunicación bidireccional punto a punto entre dos procesos

Proceso de comunicación cliente-servidor



Familias de direcciones

```
#include <sys/socket.h>
struct sockaddr
{
        u_short sa_family;    //familia de direcciones
        char sa_data[14];     //dirección concreta
};
```

Familia de direcciones	Familia de protocolos	Descripción
AF_UNIX	PF_UNIX	Sockets de entorno UNIX
AF_INET	PF_INET	TCP/IP
AF_AX25	PF_AX25	Protocolo AX.25 para radioaficionados
AF_IPX	PF_IPX	Protocolo Novell IPX
AF_APPLETALK	PF_APPLETALK	Protocolo Apple Talk DDS

Manejo de direcciones…

```
struct sockaddr_in
{           u_short sin_family;             // tipo de dirección
            u_short sin_port;               // número de puerto
            struct in_addr sin_addr;        // dirección IP
            char sin_zero[8];
};

struct in_addr
{
            u_long s_addr;
};
```

El protocolo TCP/IP es de tipo big-endian (almacena el bit más significativo de los números multibyte en la dirección de memoria más baja)
Sin embargo, los procesadores Intel x86 y sus compatibles, son de tipo little-endian (almacenamiento a la inversa)

Manejo de direcciones…

```
#include <netinet/in.h>

unsigned long int htonl (unsigned long int hostlong);
unsigned short int htons (unsigned short int hostshort);
unsigned long int ntohl (unsigned long int netlong);
unsigned short int ntohs (unsigned short int netshort);
```

Ejemplo:

```
#define PUERTO_SERVIDOR_TCP 7000
int main(void)
{
        struct sockaddr_in servidor_addr;
        ......
        servidor_addr.sin_port = htons (PUERTO_SERVIDOR_TCP);
        ......
}
```

Manejo de direcciones

```
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int inet_aton (const char *cad_dir, struct in_addr *ip_dir);
char *inet_ntoa (struct in_addr direccion);
```

Ejemplo:

```
......
#define DIRECCION_SERVIDOR_TCP "192.168.1.1"
....
int main(void)
{
struct sockaddr_in servidor_addr;
......
inet_aton ( DIRECCION_SERVIDOR_TCP, &servidor_addr.sin_addr);
......
}
```

Obtener información del host

```c
#include <unistd.h>
int gethostname(char *nombre, size_t longitud_nombre);
```

```c
#include <netdb.h>
struct hostent *gethostbyname(const char *nombre);

struct hostent
{
        char *h_name;         //nombre oficial del host
        char **h_aliases;     //lista de alias alternativos
        int h_addrtype;       //tipo de dirección: AF_INET
        int h_length;         //longitud de la dirección
        char **h_addr_list;   //lista de direcc. para host
};
```

Trabajando con sockets…

- Crear un canal de comunicación bidireccional

```
#include <sys/types.h>
#include <sys/socket.h>

int socket (int familia, int tipo, int protocolo);

SOCK_STREAM para TCP
SOCK_DGRAM para UDP
```

EPROTONOSUPPORT	Servicio requerido o protocolo especificado no son válidos.
EINVAL	Familia o protocolo desconocidos.
EMFILE	La tabla de descriptores de ficheros del proceso está llena.
ENFILE	La tabla de ficheros del sistema está llena.
EACCESS	Carece del permiso necesario para la creación del socket.
ENOBUFS	El sistema no tiene espacio de buffer disponible.

Trabajando con sockets…

- Enlazar un socket con una dirección de red

```
#include <sys/types.h>
#include <sys/socket.h>

int bind (int sfd, struct sockaddr *dir_local, socklen_t longitud_dir);
```

EBADF	El primer argumento no es un descriptor válido.
ENOTSOCK	El primer argumento no especifica un descriptor de socket.
EADDRNOTAVAIL	La dirección especificada no está disponible (p.e., si la dirección IP especificada no coincide con la dirección de la máquina local) .
EADDRINUSE	La dirección especificada está ocupada (p.e., otro proceso ha podido conectarse a este puerto).
EFAULT	El puntero que representa la dirección local es inválido.
EINVAL	El socket ya está ligado a otra dirección.
EACCES	El proceso no tiene permisos para acceder a ese puerto.

Trabajando con sockets…

- El servidor se pone en modo pasivo, en espera de peticiones de los clientes

```
#include <sys/socket.h>

int listen (int sfd, int longitud_cola);
```

EBADF	El primer argumento no es un descriptor válido.
ENOTSOCK	El primer argumento no especifica un descriptor de socket.
EOPNOTSUPP	El tipo de socket no soporta la función *listen()*.

Trabajando con sockets…

- El cliente inicia una conexión con el servidor

```
#include <sys/types.h>
#include <sys/socket.h>

int connect (int sfd, const struct sockaddr *direccion, socklen_t longitud_dir);
```

EBADF	El primer argumento no es un descriptor válido.
ENOTSOCK	El primer argumento no especifica un descriptor de socket.
EAFNOSUPPORT	La familia de direcciones especificada en el punto remoto no puede ser utilizada en este tipo de socket.
EADDRNOTAVAIL	La dirección especificada no está disponible.
EISCONN	El socket ya está conectado.
ETIMEDOUT	El tiempo para intentar la conexión ha excedido del *timeout* prefijado (sólo para TCP).
ECONNREFUSED	Conexión rechazada por la máquina remota (sólo para TCP).
ENETUNREACH	No es posible encontrar la red (sólo para TCP).
EADDRINUSE	La dirección especificada está ocupada.
EINPROGRESS	El socket no está bloqueado y una posible conexión podría bloquearlo (sólo para TCP).
EALREADY	El socket no está bloqueado y una posible llamada esperaría a que se completara una conexión previa (sólo para TCP).

Trabajando con sockets…

- El servidor lee peticiones de servicio de la cola
- Se elimina la petición
- Se crea un nuevo socket

```
#include <sys/types.h>
#include <sys/socket.h>

int accept (int sfd, struct sockaddr *dir, socklent_t *longitud_dir);
```

EBADF	El primer argumento no es un descriptor válido.
ENOTSOCK	El primer argumento no especifica un descriptor de socket.
EFAULT	El puntero que representa la dirección remota es inválido.
EWOULDBLOCK	El socket se ha definido como E/S no bloqueada y no se esperan conexiones a aceptar.
EOPNOTSUPP	El tipo de socket no es SOCK_STREAM

Trabajando con sockets…

- Transmitir datos entre máquinas

```
int write (int socket, char *buffer, int longitud_buf);
```

EBADF	El primer argumento no es un descriptor válido.
EPIPE	Intento de escritura en un socket desconectado.
EFBIF	La cantidad de datos excede de la capacidad del sistema.
EFAULT	La dirección del puntero buffer es incorrecta.
EINVAL	El valor del socket es inválido.
EIO	Error de entrada/salida.
EWOULDBLOCK	El socket no puede aceptar todos los datos, si no, se bloquea, pero se ha definido una entrada/salida no bloqueada.

Trabajando con sockets…

- Recibir datos

```
int read(int socket, char *buffer, int longitud_buf);
```

EBADF	El primer argumento no es un descriptor válido.
EFAULT	La dirección del puntero buffer es incorrecta.
EINTR	Una señal ha interrumpido la lectura.
EIO	Error de entrada/salida.
EWOULDBLOCK	Se ha especificado una E/S no bloqueada pero el socket no contiene datos.

Trabajando con sockets

- Cerrar la conexión

```
int close (int socket);
```

EBADF	El primer argumento no es un descriptor válido.

Resumiendo

- Los sockets son un mecanismo de IPC más avanzado que permite la comunicación entre procesos de sistemas diferentes