# Contents



# 1 Sockets

[!NOTE] Idea:
Mecanismo que permite comunicación a través de TCP/IP entre procesos remotos. Describe una IP y un PUERTO.

- Ofrece una interfaz de acceso a la [[1728038869 - Transport network layer|Transport layer]] del protocolo TCP/IP y UDP

## 1.1 Tipos y dominios:

[!NOTE] Dominios:
Los dominios representan familias de protocolos a través de los cuales se comunicarán los procesos.

- AF_UNIX : Dentro de la propia máquina
- AF_INET : Comunicación usando [[1730826735 - Protocol - IPv4 protocol|IPv4]]
- AF_INET6: Comunicación usando [[1731411683 - Procotol - IPv6|IPv6]]

Existen diferentes **tipos de sockets:**

- **Stream:** Usando [[1727428451 - Definition - TCP|TCP]]
- **Datagrama:** Usando [[1727428429 - Definition - UDP|UDP]]
- **Raw:** Sin protocolo IP

## 1.2 POSIX:

Sockets se definen bajo `sys/socket.h`. Con las isguientes interfaces:

- `int socket(DOMINIO,TIPO,0)` → Devuelve el identificador del socket
- `int connect(<socket>, <address>,sizeof(<address>))` → Conectar el socket al address, del servidor o donde sea
- `int close(<socket>)` → Cerrar el socket
- `int read(<socket>, <buffer>, <size>)` → Lee del socket al buffer una cantidad size
- `int write(<socket>,<buffer>,<size>)` → Escribe al socket una cantidad e bytes size del buffer
- `int bind(<socket>,<address>,<size>)` → Asigna un address a un socket
- `int listen(<socket>,<backlog>)` → El socket empezará a escuchar para aceptar nuevas peticiones que le llegen

- `int close(<socket>)` → Cierra el socket
  La idea básica de uso es:

#### 1.2.1 Cliente:

1. Crear un socket
2. Connectar el socket
3. Usar ese socket para mandar/recibir información
4. Cerrar el socket

```
sc = socket(...);
connect(sc,...);
write(sc,...);
read(sc,...);
close(sc,...);
```

**Remarks:**

- Se utiliza un socket distinto para cada operación

#### 1.2.2 Server:

1. Crear un socket
2. Bind el socket al address del servidor
3. Empezar a escuchar en ese address
4. Ir aceptando peticiones, cada peticion crea un nuevo socket con el que interactuar con el cliente que creó la petición
5. Cerrar el socket en caso de que el servidor se cierre

```
sv = socket(...);
bind(sv,...);
listen(sv,...);
while (1) {
    nv = accept(sv,...);
    // Use that new socket to perform the required operation
    do_operation(nv);
}
// If something went wrong, or the server is stopping:
close(sv);
```

---

#### 1.2.3 Up

- [[Distribuidos]]

#### 1.2.4 Down