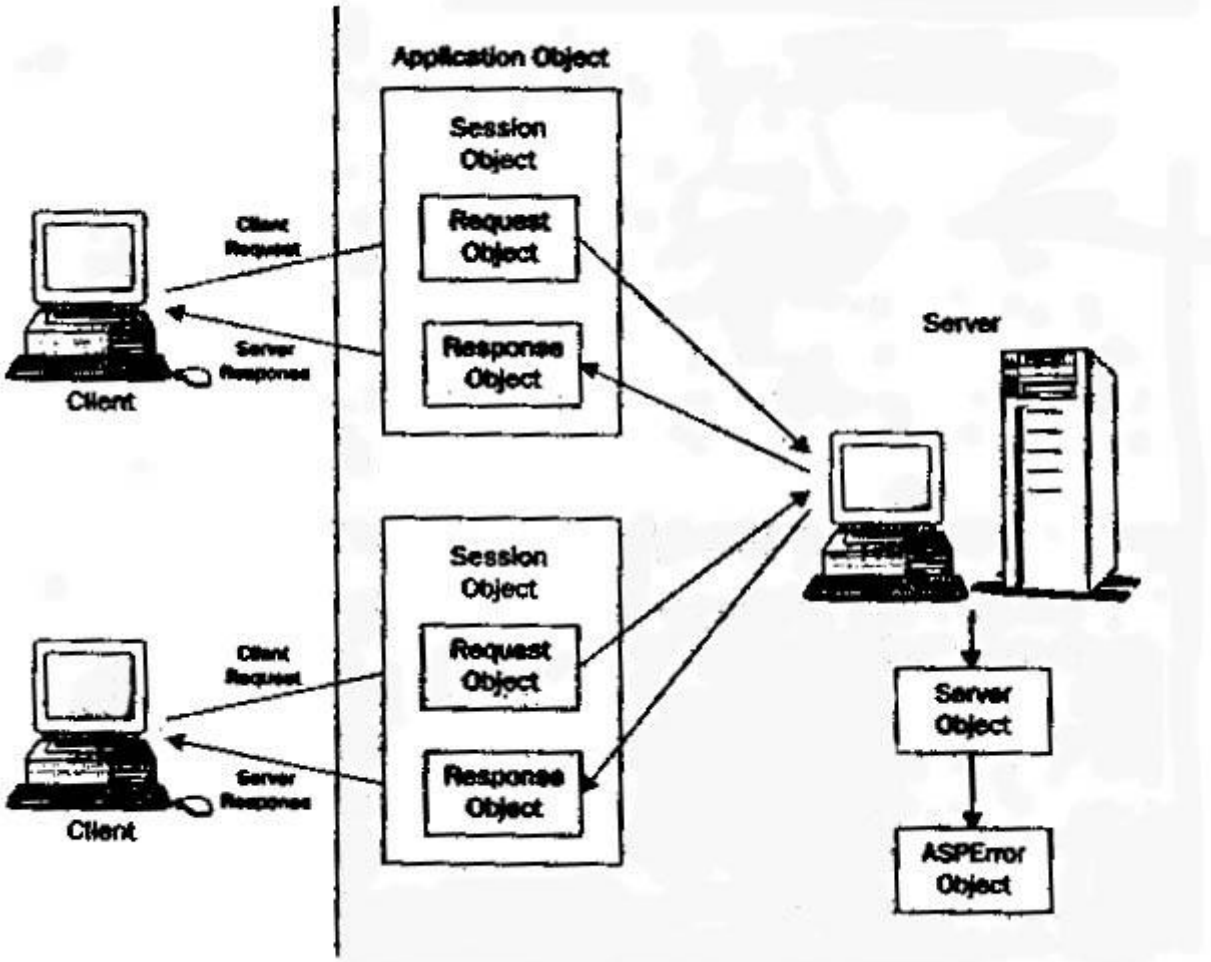
Pendahuluan

- Pemrograman Jaringan modern sekarang ini berbasis pada model client/server. Pada sebagian besar kasus, server biasanya mengirim data, sedangkan client menerimanya. Pembahasan tentang model Client/Server tidak akan lepas dari konsep sistem terdistribusi. Sebab client/server merupakan model dasar dari sistem terdistribusi.

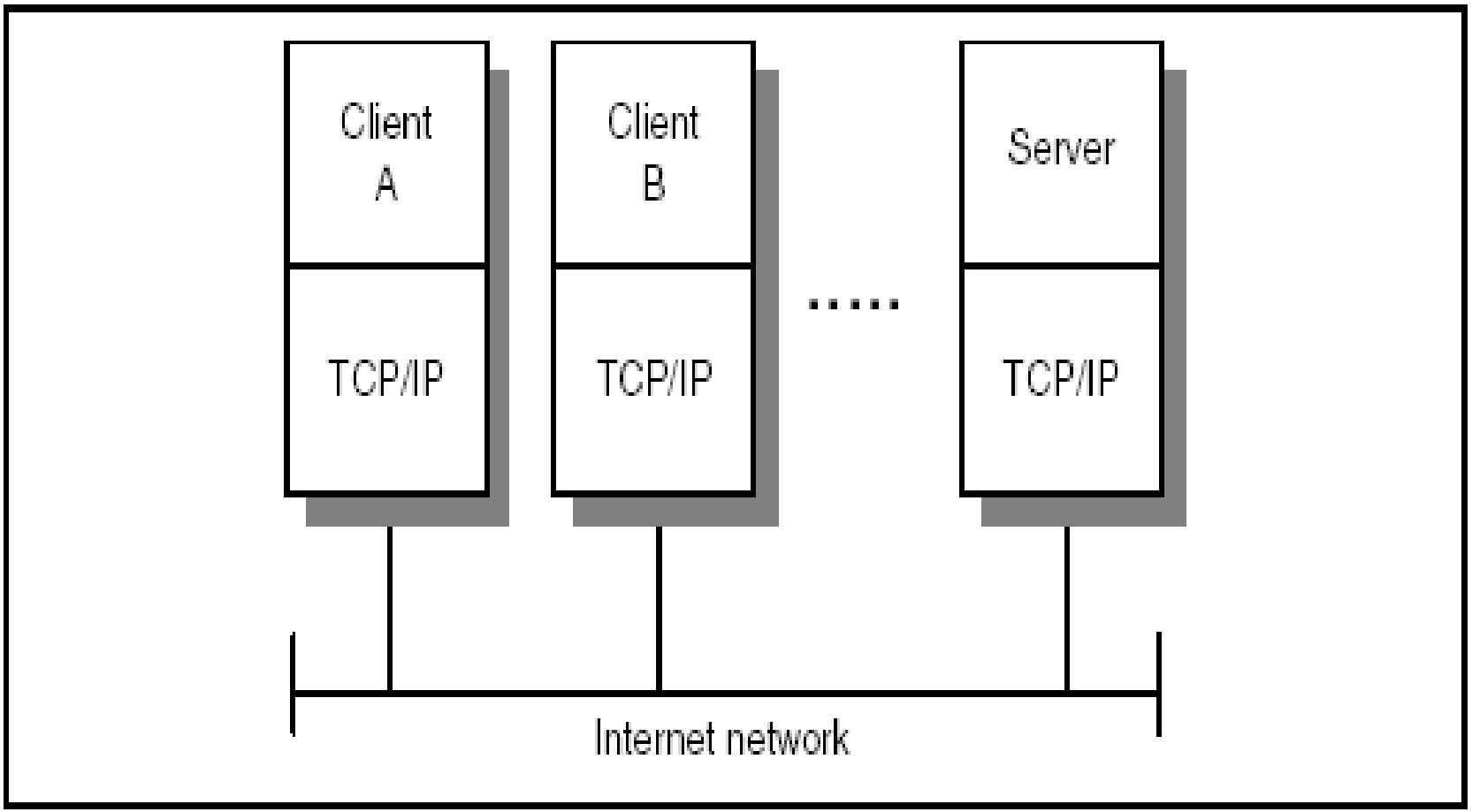
- Ada dua organisasi yang menstandarisasi pemrograman jaringan dan protokol di internet, yaitu :

1. Internet Engineering Task Force (IETF) : TCP/IP, MIME, dan SMTP.
2. World Wide Web Consortium (W3C) : HTTP, HTML, XHTML, MathML, dan XML.

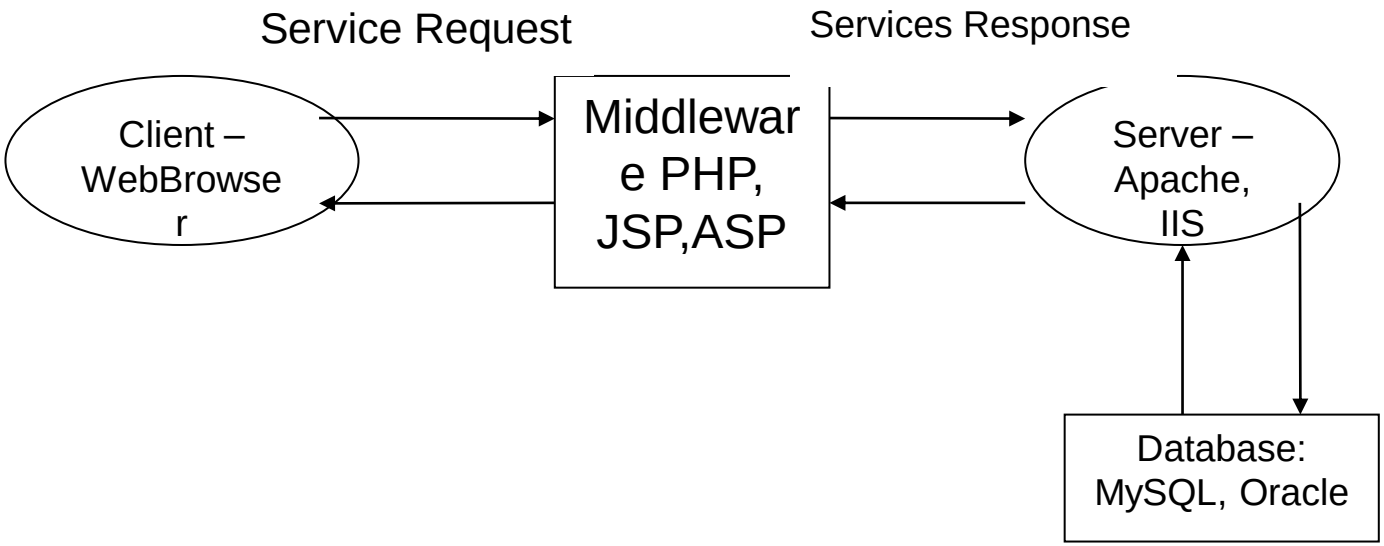
Hubungan Client-Server



Model aplikasi Client-Server



Arsitektur Client-Server



Ket:

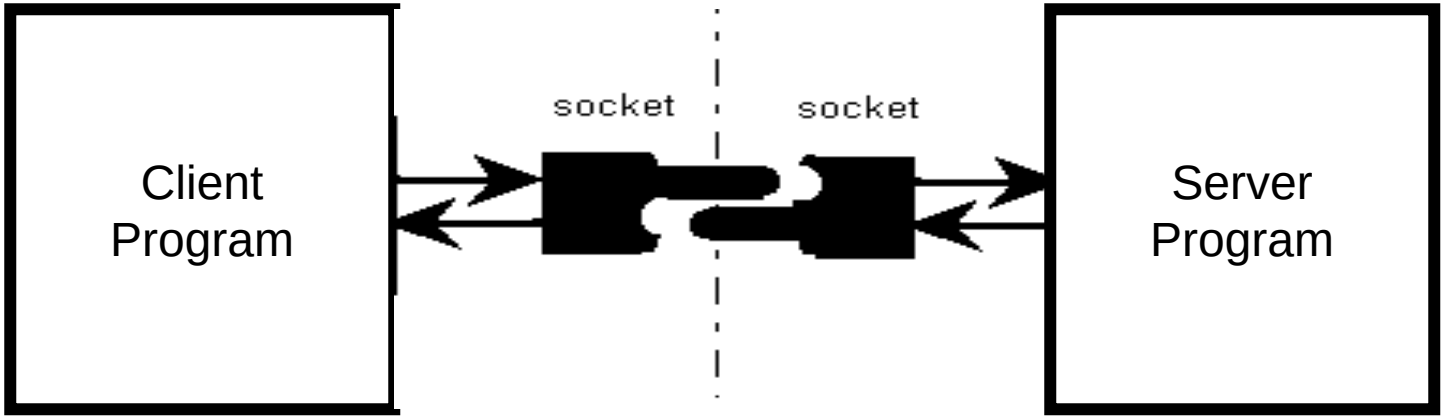
• ***Service Request*** adalah permintaan dari client baik berupa permintaan data maupun perintah ke server.

• ***Service Response*** berupa balasan dari server atas permintaan dari client berupa hasil proses.

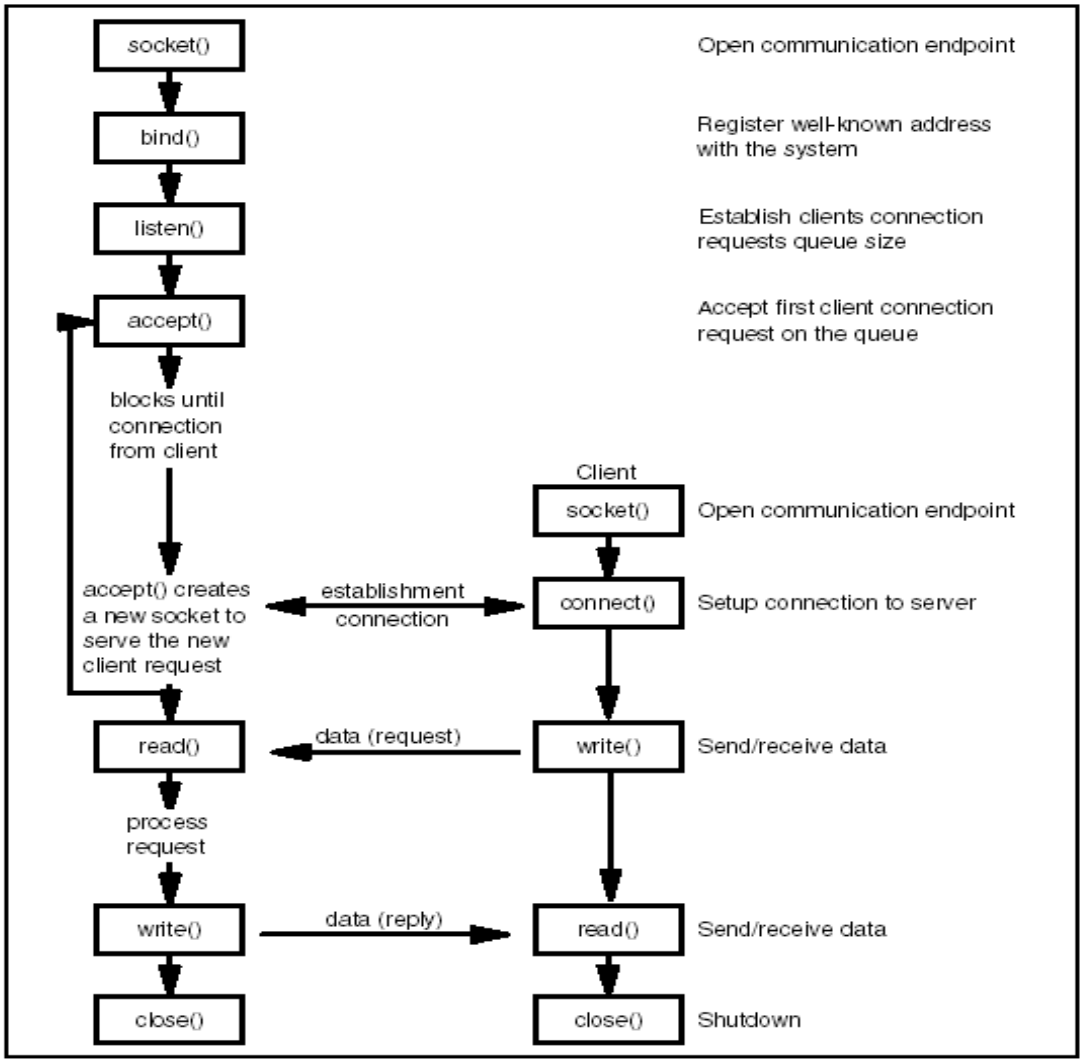
Socket

- ***Socket*** adalah sebuah special type of *file handle*, yang digunakan oleh sebuah process untuk layanan request jaringan dari operating system.

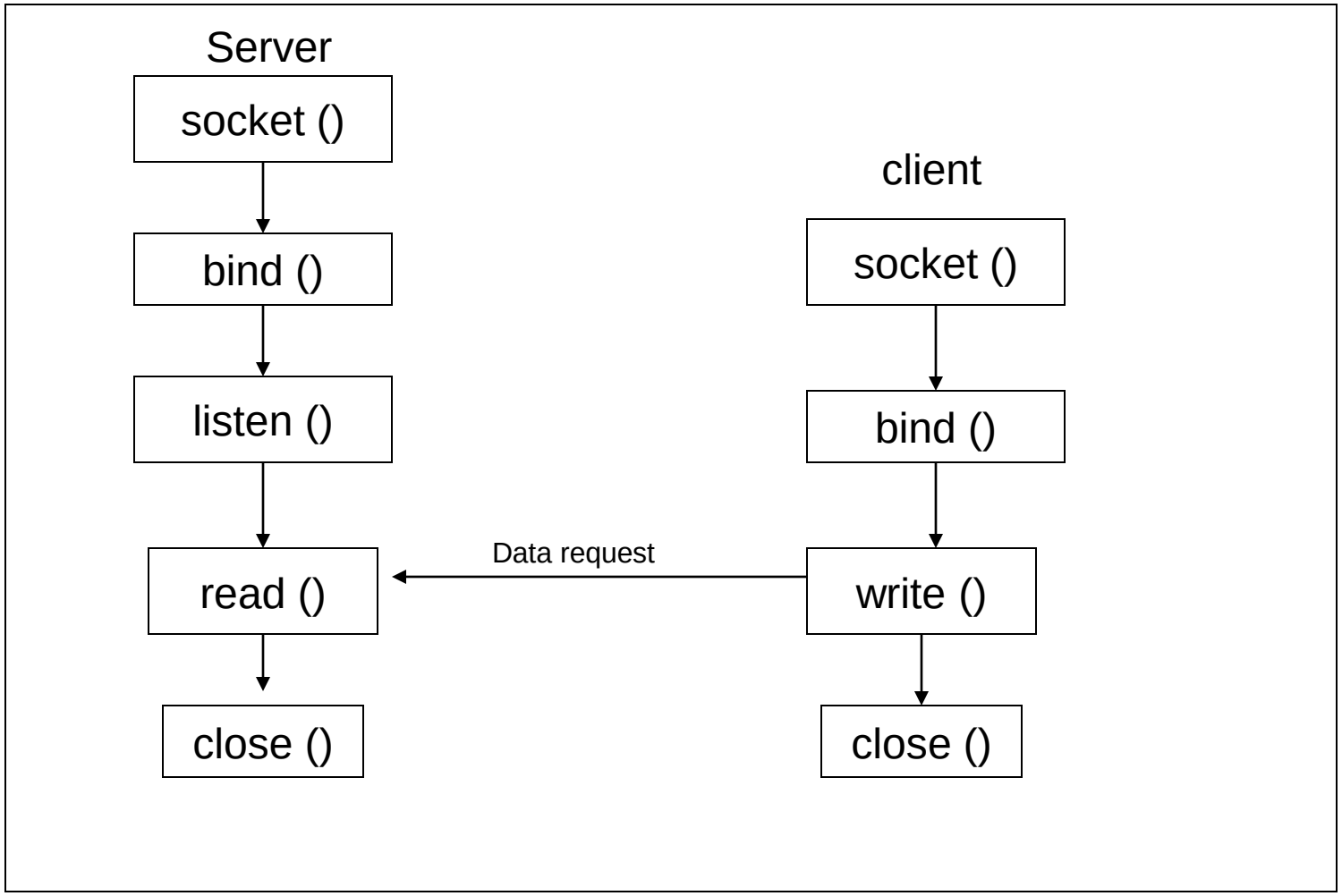
Blok Diagram Socket



Pemanggilan sistem socket untuk protokol *connection-oriented*



Pemanggilan sistem socket untuk protokol *connectionless*



Tabel Pemanggilan Sistem Socket

Type	Establish	Send	Receive
Connection-orientated server	bind, listen, or accept	write or sendto	read or recvfrom
Connection-orientated client	connect	write or sendto	read or recvfrom
Connectionless server	bind	sendto	recvfrom
Connectionless client	bind	sendto	recvfrom

Command Socket

- **Inisialisasi**

  `int sockfd = socket(int family, int type, int protocol)`

- **Registrasi socket ke alamat address**

  `int bind(int sockfd, struct sockaddr *localaddr, int addrlen)`

- **Penerimaan Koneksi**

  `int accept(int sockfd, struct sockaddr *foreign-address, int addrlen)`

- **Koneksi keluar dari server**

  `int connect(int sockfd, struct sockaddr *foreign-address, int addrlen)`

- **Send/receive data**

  `The read(),readv(sockfd, char*buffer int addrlen), recv(), readfrom(), send(sockfd,msg,len,flags), and write()`

- **Menutup socket**

  `int close(int sockfd)`

Pemrograman jaringan UDP/IP

- Mendefinisikan Layanan *connectionless*.

  Untuk server:

  ```
  sd=socket(AF_INET, SOCK_DGRAM, 0);
  ```

  Untuk client:

  ```
  sd = socket(AF_INET,SOCK_DGRAM,0);
  ```

- bind (registrasi ke alamat port)

  Untuk server :

  ```
  servAddr.sin_family = AF_INET;
  servAddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servAddr.sin_port = htons(LOCAL_SERVER_PORT);
  rc = bind (sd, (struct sockaddr *) &servAddr,sizeof(servAddr));
  ```

  Untuk client:

  ```
  cliAddr.sin_family = AF_INET;
  cliAddr.sin_addr.s_addr = htonl(INADDR_ANY);
  cliAddr.sin_port = htons(0);
  ```

- Send/Receive

Untuk server:

```
cliLen = sizeof(cliAddr);
n = recvfrom(sd, msg,MAX_MSG,0,(struct sockaddr *)
&cliAddr, &cliLen);
```

Untuk client:

```
for(i=2;i<argc;i++)
{   rc = sendto(sd, argv[i], strlen(argv[i])+1, 0,
       (struct sockaddr *) &remoteServAddr,
       sizeof(remoteServAddr));}
```

Pemrograman jaringan TCP/IP

- Mendefinisikan Layanan *connection-oriented*.

  Untuk server:

  ```
  sd = socket(AF_INET, SOCK_STREAM, 0);
  ```

  Untuk client:

  ```
  sd = socket(AF_INET, SOCK_STREAM, 0);
  ```

- bind (registrasi ke alamat port)

  Untuk server :

  ```
  servAddr.sin_family = AF_INET;
  servAddr.sin_addr.s_addr = htonl(INADDR_ANY);
  servAddr.sin_port = htons(SERVER_PORT);
  ```

  Untuk client:

  ```
  localAddr.sin_family = AF_INET;
  localAddr.sin_addr.s_addr = htonl(INADDR_ANY);
  localAddr.sin_port = htons(0);
  ```

- Send/Receive

Untuk server:

```
while(read_line(newSd,line)!=ERROR) {
printf("%s: received from %s:TCP%d : %s\n", argv[0],
inet_ntoa(cliAddr.sin_addr),
ntohs(cliAddr.sin_port), line);
```

Untuk client:

```
rc = connect(sd, (struct sockaddr *) &servAddr,
sizeof(servAddr));
```