Hálózati alkalmazások készítése

A Socket koncepció

Források

- Csizmazia Balázs: Hálózati alkalmazások fejlesztése
- http://www.uwo.ca/its/doc/courses/notes/socket
- http://pont.net/socket/
- http://www.ibrado.com/sock-faq/
- UNIX manual

Történelem – socket mechanizmus

- Eredetileg a BSD vezette be (1983: 4.2 BSD), mint IPC mechanizmust.
- Futótűzként elterjedt.
- Mára a számos IPC mechanizmus közül a legnépszerűbb.

Mi is az a Socket?

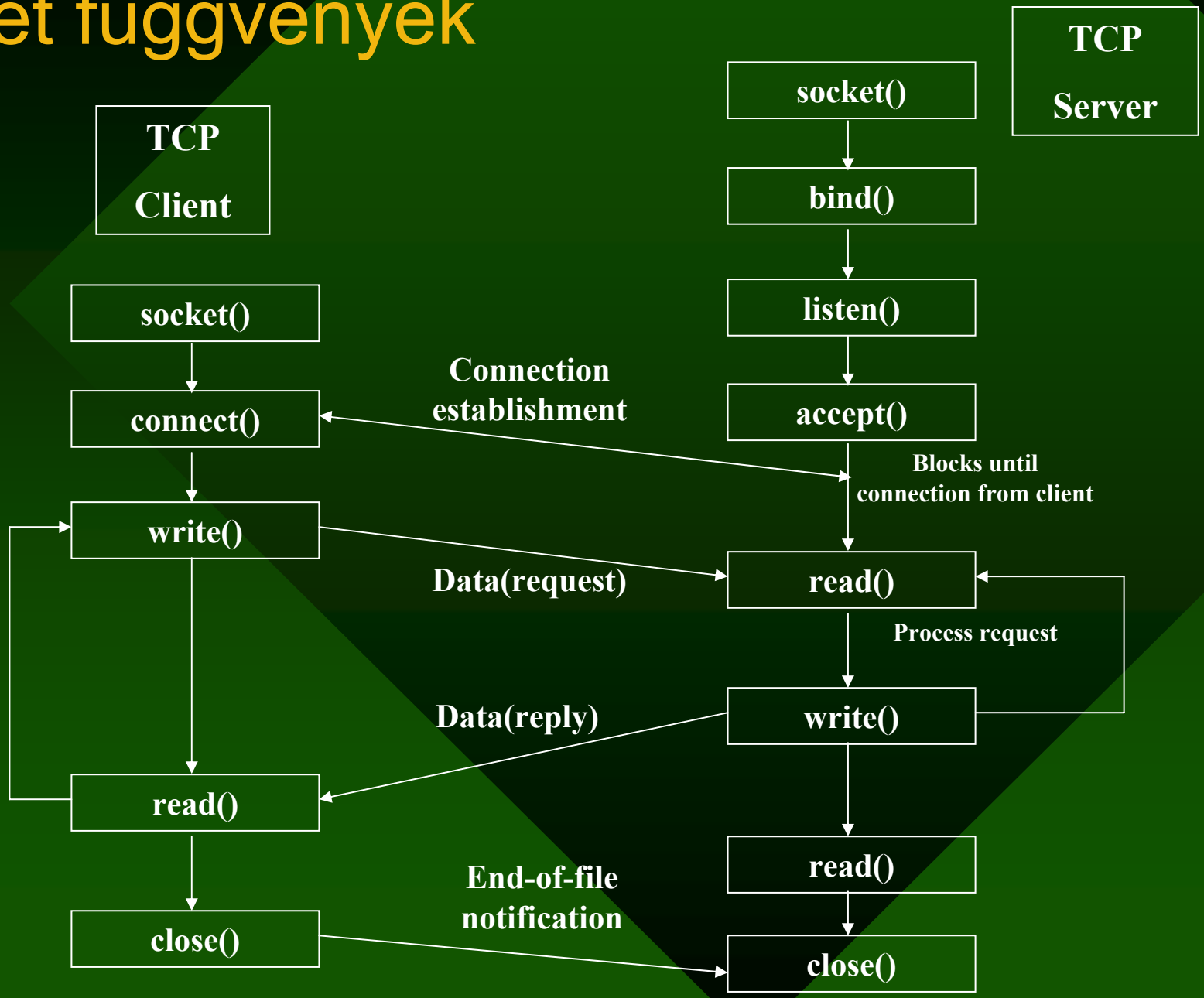
- Middleware.
- Független az operációs rendszertől (de UNIX-on kicsit többet tud)
- Hálózattranszparens.
- Kötöttsége: „csak” TCP/IP támogatás

Definició

A socket a UNIX fájl hozzáférési mechanizmusának általánosítása, mely egy kommunikációs végpontként szolgál.

- Full-duplex
- Két (alap) szolgáltatása:
  - Kapcsolat orientált kommunikáció (TCP)
  - Kapcsolat nélküli kommunikáció (UDP)

Socket függvények



Socket típusok

- PF_INET: /Internet domain socket/ egy IPv4 címmel rendelkező interfészhez kapcsolódik.
- PF_INET6: ua. mint PF_INET csak IPv6
- PF_UNIX (PF_LOCAL): /UNIX domain socket/ csak helyi IPC-t valósít meg

Socket típusok 2

- PF_IPX: Novell IPX/SPX protokoll alapú kommunikációt valósít meg.
- PF_X25: ISO-8208 protokoll alapú kommunikációt valósít meg.
- egyéb.

Kommunikáció típusok

- SOCK_STREAM: /TCP/ kapcsolat alapú, kétirányú, előzés mentes, megbízható
- SOCK_DGRAM: /UDP/ kapcsolat nélküli, nem megbízható, kötött üzenetméret
- SOCK_SEQPACKET: előzés mentes, megbízható, kétirányú, kapcsolat alapú, kötött üzenetméretű
- SOCK_RDM: megbízható, kapcsolat nélküli, kapcsolat nélküli

Kommunikáció típusok

- SOCK_RAW: /átjáró/ IP szintű csomag kezelés. Különlegessége, hogy nem porthoz kötődik, hanem protokollhoz.

TCP szerver írása

1. Socket készítés

```
Sock = socket(PF_INET, SOCK_STREAM, 0);
```

2. Kötés interfészhez

```
Addr.sin_family = AF_INET;
Addr.sin_port   = htons(8999);
inet_aton("127.0.0.1", &Addr.sin_addr);
bind(Sock, (struct sockaddr *) &Addr, sizeof Addr)
```

TCP szerver írása 2

3. „Hallgatózás” beállítása

   listen(Sock, 5)

4. Kapcsolat fogadása

   Conn = accept(Sock, NULL, NULL);

5. Adat küldés/fogadás

   write(Conn, HELO, strlen(HELO));

   read(Conn,buf, 30);

TCP szerver írása 3

6. Kapcsolat bontása

   close(Conn);

7. Vissza a kapcsolat fogadásához

8. Socket megszüntetése

   close(Sock);

TCP kliens írása

1. Socket létrehozása

   Sock = socket(PF_INET, SOCK_STREAM, 0);

2. Kapcsolódás a szerverhez

   Addr.sin_family = AF_INET;

   Addr.sin_port = htons(8999);

   inet_aton("127.0.0.1", &Addr.sin_addr);

   *Conn=connect(Sock, (struct sockaddr *) &Addr, sizeof Addr)*

TCP kliens írása 2

3. Adat küldés/fogadás

   read(Sock, buf, 30);

   write(Sock, HELO, sizeof HELO);

   /* Nem a Conn-t használjuk !!! */

4. Kapcsolat bontása

   close(Sock);

UDP szerver írása

1. Socket létrehozása

   sd=socket(AF_INET, SOCK_DGRAM, 0);

2. Kötés interfészhez

   servAddr.sin_family = AF_INET;

   servAddr.sin_addr.s_addr = htonl(INADDR_ANY);

   servAddr.sin_port = htons(LOCAL_SERVER_PORT);

   *bind (sd, (struct sockaddr *) &servAddr,*
   sizeof(servAddr));

UDP szerver írása 2

3. Üzenet fogadás (küldés)

   *recvfrom(sd, msg, MAX_MSG, 0, (struct sockaddr *) &cliAddr, &cliLen);*

4. Socket megszüntetés

   close(sd);

UDP kliens írása

1. Socket létrehozása

   sd = socket(AF_INET,SOCK_DGRAM,0);

2. Kötés (tetszőleges) interfészhez

   cliAddr.sin_family = AF_INET;

   cliAddr.sin_addr.s_addr = htonl(INADDR_ANY);

   cliAddr.sin_port = htons(0);

   *bind(sd, (struct sockaddr *) &cliAddr, sizeof(cliAddr));*

UDP kliens írása 2

3. Adat küldés (fogadás)

```
remoteServAddr.sin_family = AF_INET;
inet_aton("127.0.0.1", &remoteServAddr.sin_addr);
remoteServAddr.sin_port =
    htons(REMOTE_SERVER_PORT);
sendto(sd, HELO, strlen(HELO), 0, (struct sockaddr
    *) &remoteServAddr, sizeof(remoteServAddr))
```

4. Socket megszüntetés

```
close(sd);
```

UNIX domain socket

- Csak UNIX rendszereken implementálják
- Csak lokális kommunikációra használható (a szerver és a kliens ugyanazon a gépen fut)
- Megengedett kommunikációs típusok: SOCK_STREAM, SOCK_DGRAM
- A fájlrendszer egy speciális fájlján keresztül valósul meg

UNIX domain socket

- Létrehozása

  Sock = socket(PF_UNIX, SOCK_STREAM, 0);

- Kötése

  Addr.sun_family = AF_UNIX;

  strcpy(Addr.sun_path, "unix_sock\0");

  *bind(Sock, (struct sockaddr *) &Addr, sizeof Addr)*

UNIX domain socket 2

- Kapcsolódás

```
Addr.sun_family = AF_UNIX;
strcpy(Addr.sun_path, "unix_sock\0");
Conn=connect(Sock, (struct sockaddr *) &Addr,
  sizeof Addr)
```

- Az adatátvitelben nincs különbség