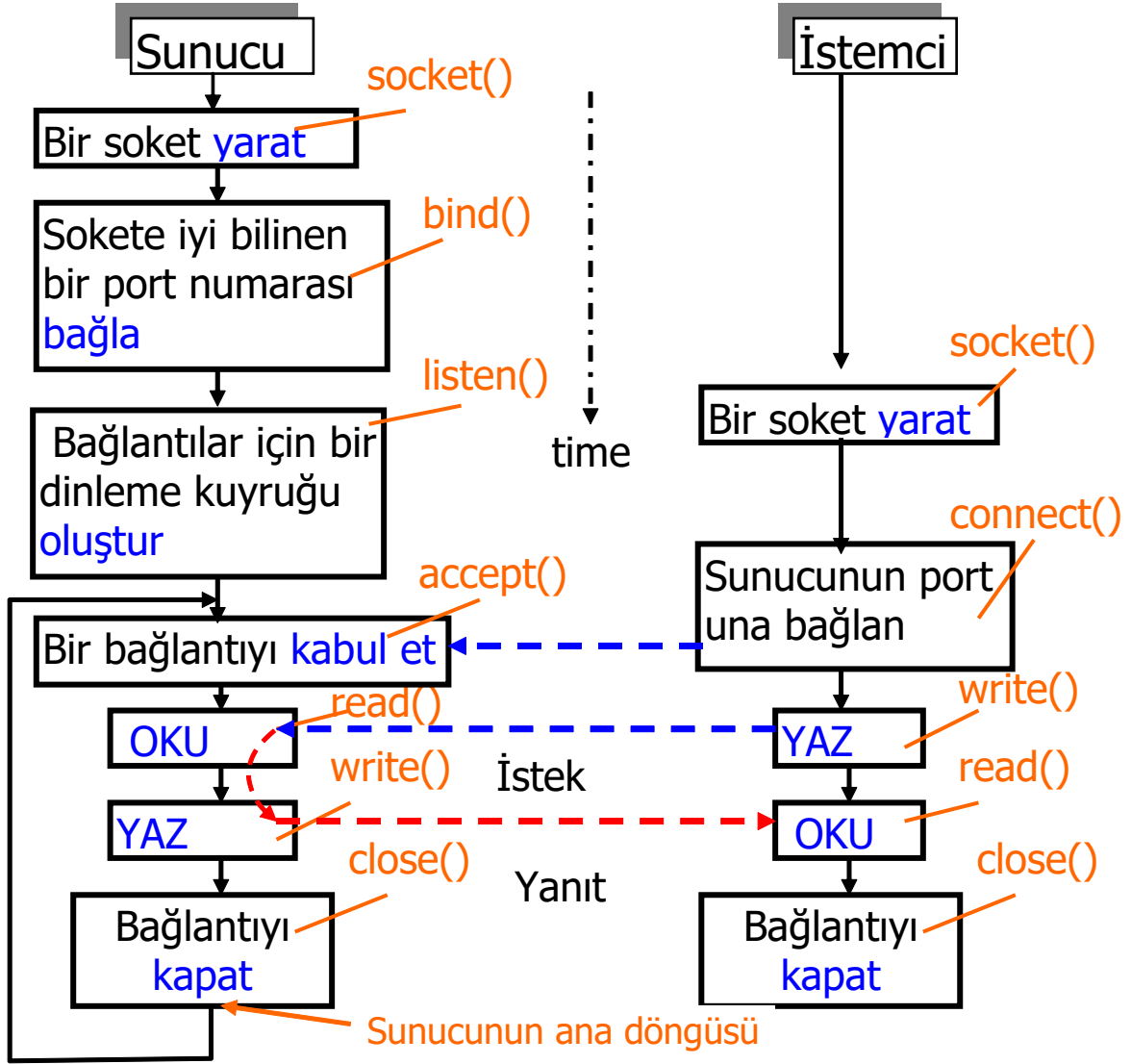
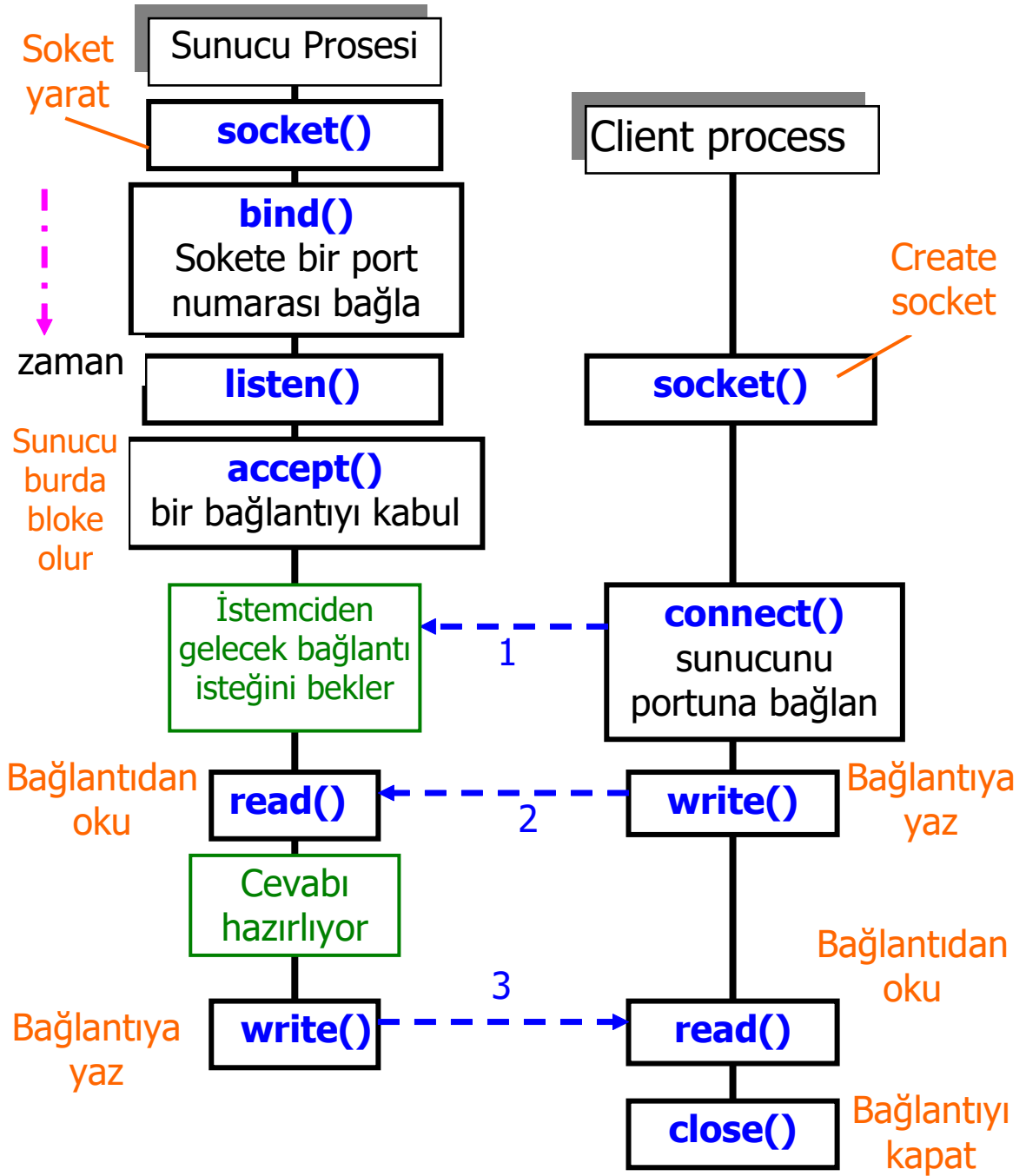


Soket IPC

(bağlantı doğrultusunda olan **istemci** and **sunucu** operasyonları)




Bağlantı-doğrultusunda olan istemci ve sunucu iletişimi

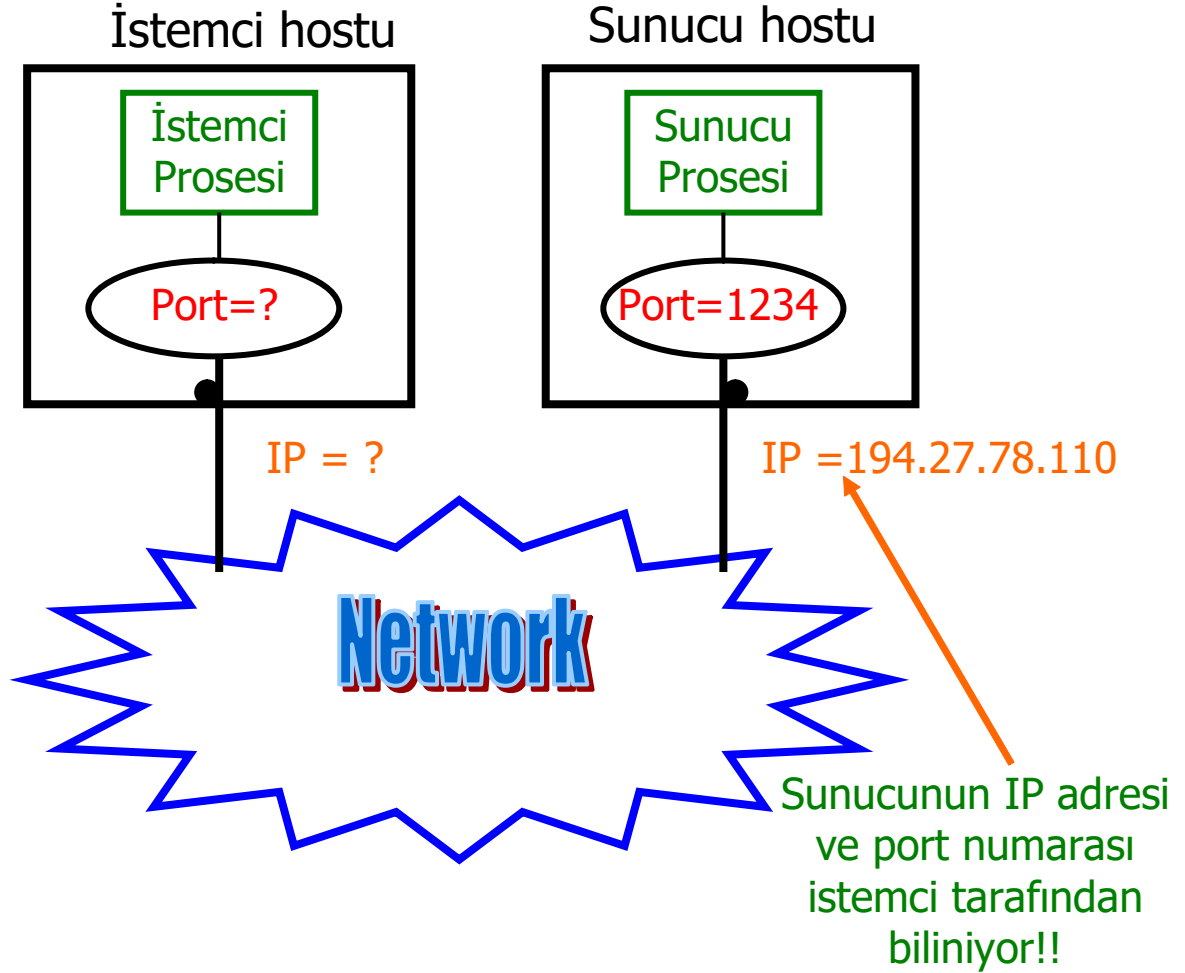


Sunucu programında tipik bir ana döngü (program parçası)

```
struct sockaddr_in client ; /* istemcinin adresi için */
int len, sd, ns ;
char in[1000] ; /* istemciden gelen*/
char out[1000] ; /* istemciye giden*/
. . . . .
len = sizeof( client ) ;
. . . . .
while( 1 )
{ ns = accept( sd, &client, &len ) ; /* bağlantıyı
                                     kabul et */
  while( read( ns, in, 1000 ) > 0 )
  { /* istemciden gelen veriyi işle */
    . . . . .
    write( ns, out, 1000 ) ; /* istemciye gönder */
  }
  close( ns ) ; /* soketi kapat */
}
```



Internet-alanı akış soketleri kullanılarak tasarlanmış
istemci/sunucu sistemine bir örnek



İstemci/sunucu sistemi için Internet-Alanı soketleri

Sunucu: 194.27.78.110 de başlar	İstemci: herhangi bir host ta başlar
<pre> #include ... #define PORT 1234 main() {int n, s, ns, len ; char buf[1024] ; struct sockaddr_in name ; if((s=socket(AF_INET,...))<0) { error } name.sin_family=AF_INET ; name.sin_port=htons(PORT); name.sin_addr.s_addr = htonl(INADDR_ANY) ; len=sizeof(name) ; if (bind(s,&name,len) < 0) { error } if (listen(s,5) < 0) { error } while (1) {if((ns=accept(s,&name,...))<0) { error } while((n=read(ns,buf,...))>0) write(1,buf,n) ; close(ns) ; } }</pre>	<pre> #include ... #define PORT 1234 #define IP "194.27.78.110" main() {int n, s, len ; char buf[1024] ; struct sockaddr_in name ; if((s=socket(AF_INET,...))<0) { error } name.sin_family=AF_INET ; name.sin_port=htons(PORT); name.sin_addr.s_addr = inet_addr(IP) ; len = sizeof(name) ; if (connect(s,&name,len)<0) { error } while((n=read(0,buf,...)) >0) {if (write(s,buf,n,...) < 0) { error } } close(s) ; exit(0) ; }</pre>

Detaylı program için bakınız (Curry), pp.404-407

istemci/sunucu sistemi için Internet-Alanı soketleri için başka bir örnek

İstemci prgramı

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <ctype.h>

#define PORT 7000 /* Sunucunun port adresi */
#define SIZE sizeof(struct sockaddr_in)

int main(void)
{
    char buf[1024];
    int n, s, ns, len;

    /* İstemci için port ve ip adresi işletim sistemi tarafından belirlenecek: */
    struct sockaddr_in cli = {AF_INET, INADDR_ANY, INADDR_ANY};

    /* Sunucu adresi */
    struct sockaddr_in srv = {AF_INET, PORT, inet_addr("127.0.0.1")};

    printf("İstemci: başlıyor ...\n");

    /* SOCK_STREAM tipinde Internet soketi oluştur */
    if ((s = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        perror("Soket problemi");
        exit(1);
    }

    /* Soketi port ve ip numarasına bağla */
    if (bind(s, (struct sockaddr *) &cli, SIZE) < 0)
    {
        perror("Bağlama hatası");
        exit(1);
    }
}
```

```

/* Sunucuya bağlanmayı dene */

len = SIZE;
if (connect(s, (struct sockaddr *)&srv, len) < 0)
{
    perror("Bağlantı hatası");
    exit(1);
}

/* Mesajı s soketi aracılığıyla sunucuya gönder */
n = send(s, "This is a request from client\n", 30, 0);
if (n < 0)
{
    perror("Gönderim hatası");
    exit(1);
}

printf ("İsteci: istek gönderildi ...\n");

/* Sunucudan gelen cevabı alıp yazdır */
n = recv(s, buf, sizeof(buf), 0);
write (1, buf, n);

/* Soketi kapat ve çık. */
close(s);
printf ("TCP istemcisi sonlandı...\n");
exit(0);
}

```

Sunucu Programı

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <ctype.h>

#define PORT 7000 /* Sunucunun port adresi */
#define SIZE sizeof(struct sockaddr_in)

int main(void)
{
    char buf[1024];
    int n, s, ns, len;
    struct sockaddr_in srv = {AF_INET, PORT, INADDR_ANY};
    struct sockaddr_in cli; /* İstemci adresi için */
    int cli_len = SIZE;

    printf("Sunucu: başlıyor...\n");

    /* SOCK_STREAM tipinde soket oluştur */
    if ((s = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        perror("Soket hatası");
        exit(1);
    }

    /* Soketi istenen port ve tüm ip adreslerine bağla */
    if (bind(s, (struct sockaddr *) &srv, SIZE) < 0)
    {
        perror("Bağlama hatası");
        exit(1);
    }

    /* Dinlemeye (bağlantı beklemeye) başla, en fazla bağlantı bekleyen 5
adet istemci olabilir */
    if (listen(s, 5) < 0)
    {
        perror("Dinleme hatası");
        exit(1);
    }
}
```



```

    }

    while(1)
    {
        /* Bağlantı kabul et, eğer sırada bekleyen yoksa, bir bağlantı gelinceye
        kadar bekler */
        ns = accept(s, (struct sockaddr *) &cli, &len);
        if (ns < 0) {
            perror ("Kabul etme sorunu");
            exit(1);
        }

        /* Bağlantı sağlanan soketten oku */
        n = recv(ns, buf, sizeof(buf), 0);
        if (n < 0) {
            perror("Okuma sorunu");
            exit(1);
        }

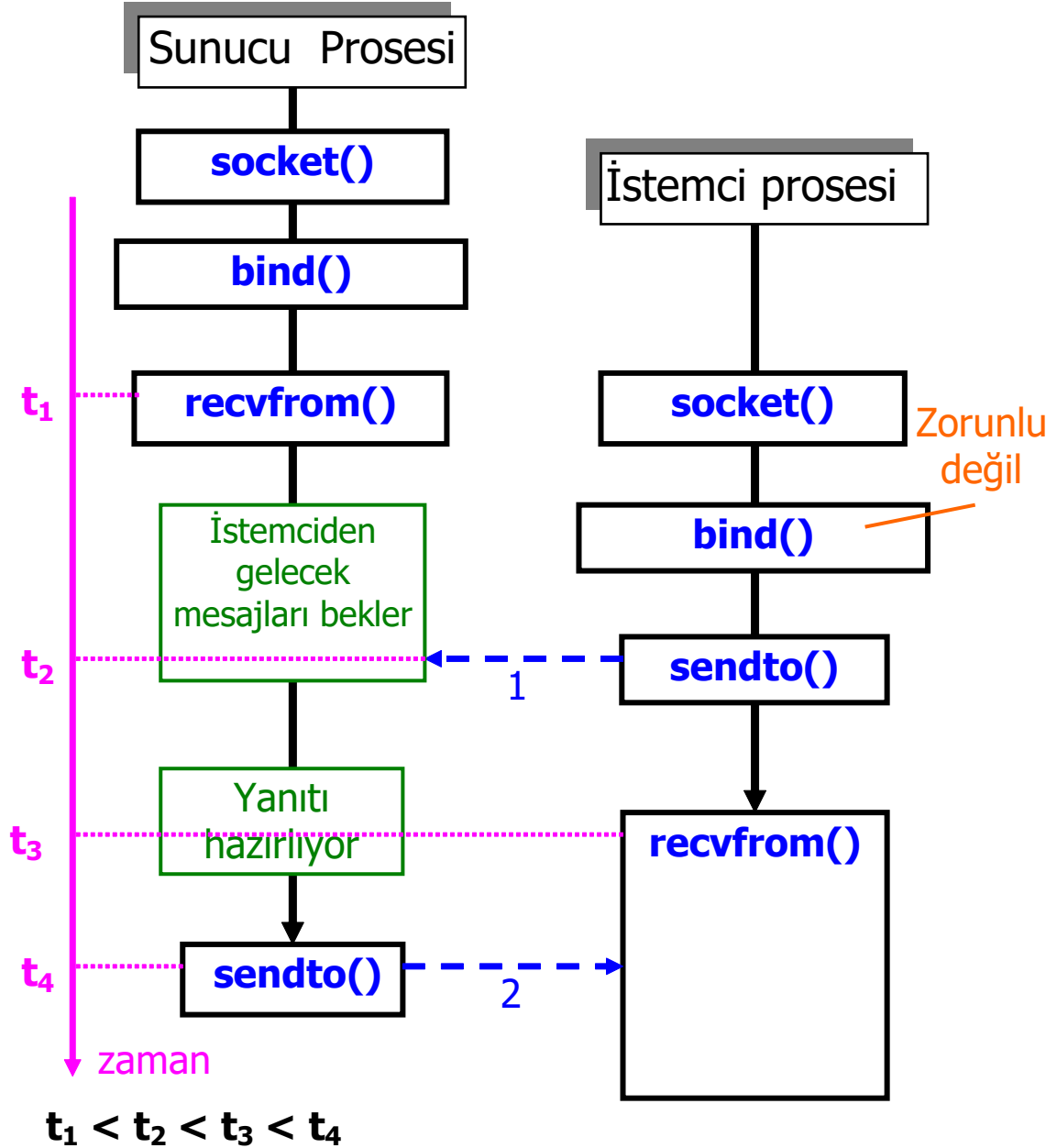
        write(1, buf, n);

        /* İstemciye cevap yolla */
        send (ns, "This is a reply from server\n",28,0);
        printf ("Sunucu: cevap gönderildi...\n");

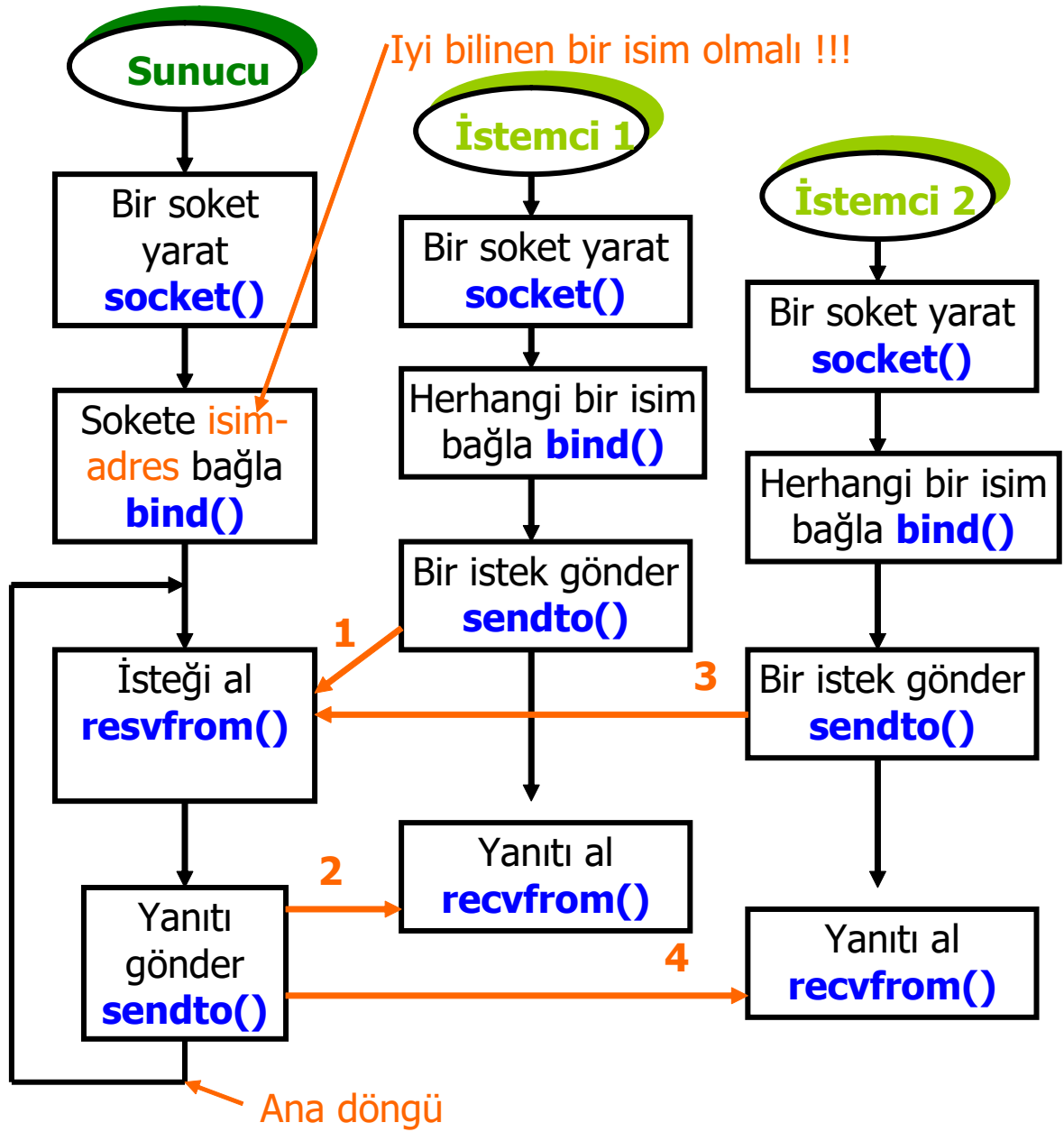
        /* Soketi kapat ve devam et */
        close (ns);
    }
}

```

Bağlantısız istemci-sunucu iletişimi



Soketlerle bağlantısız istemci/sunucu sistemi sockets



Bağlantısız soketle istemci programı

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#define SERV_PORT 7000 /* Sunucunun port numarası */
#define SERV_IP "194.27.78.05" /*Sunucunun IP adresi */

#define SIZE sizeof(struct sockaddr_in)

int main(void)
{
    int sd, ret, n, len;
    struct socaddr_in cln, srv;
    char buf[100], msg[] = "İstemcinin mesajı\n";

    /* SOCK_DGRAM tipinde Internet soketi yarat */
    if ((sd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
    { perror("socket problem"); exit(1); }

    /* Sokete bağlamak için istemcinin adres bilgilerini belirle */
    cln.sin_family = AF_INET;
    cln.sin_port = htons(0); /*Herhangi bir port*/
    cln.sin_addr.s_addr = htonl(INADDR_ANY);

    /* Soketi bu hostun IP adresine bağla */
    if (bind(sd, (struct sockaddr *) &cln, SIZE) < 0)
    { perror("bind problem"); exit(1); }

    /* Sunucunun adres bilgieri gönderi yapmadan önce
    belirlenir */
    srv.sin_family = AF_INET;
    srv.sin_port = htons (SERV_PORT);
    srv.sin_addr.s_addr = inet_addr(SERV_IP);

    /* Soket sd ye mesaj gönder */
    ret = sendto(sd, msg, 22, 0, (struct sockaddr*)&srv, SIZE);
    if (ret < 0) {perror("sendto problem"); exit(1);}
```

```

n=recvfrom(sd, buf, sizeof(buf), 0, (struct sockaddr*)&srv,
SIZE);
if (n < 0) {perror("recvfrom problem"); exit(1);}
buf[n] = NULL; printf("%s\n", buf);
close(sd);
exit(0);
}

```

Bağlantısız soketle sunucu programı

```

#include ... /* İstemci dekilerin aynisi */

#define PORT 7000 /* Sunucunun port numarası */
#define SIZE sizeof(struct sockaddr_in)

int main(void)
{
    int sd, ret, n, cln_len;
    struct socaddr_in srv, cln;
    char buf[100], reply[] = "Sunucunun yanıtı\n";

    /* SOCK_DGRAM tipinde Internet soketi yarat */
    if ((sd = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
    { perror("socket problem"); exit(1); }

    /* Sokete bağlamak için sunucunun adres bilgileri belirle */
    srv.sin_family = AF_INET;
    srv.sin_port = htons (PORT);
    srv.sin_addr.s_addr = htonl (INADDR_ANY);

    /* Soketi bu hostun IP adresine bağla */
    if (bind(sd, (struct sockaddr *) &srv, SIZE) < 0)
    { perror("bind problem"); exit(1); }

    /* Soket sd den oku */
    while(1) /* Sonsuz döngü */
    {
        n = recvfrom(sd, buf, sizeof(buf), 0, (struct
sockaddr*)&cln, &cln_len);
        if (n < 0) {perror("recvfrom problem"); exit(1);}
    }
}

```

```

    write(1, buf, n);
    ret = sendto(sd, reply, sizeof(reply), 0, (struct
sockaddr*)&cln, cln_len);
    if (ret < 0) {perror("sendto problem"); exit(1);}
}
close(sd);
exit(0);
}

```

Bağlantısız soketle başka bir istemci programı

```

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <ctype.h>

#define PORT 7000 /* Port numarası */
#define SIZE sizeof(struct sockaddr_in)

int main(void)
{
    char buf[1024];
    int n, s, ns, len;
    struct sockaddr_in cli = {AF_INET, INADDR_ANY, INADDR_ANY};
    struct sockaddr_in srv = {AF_INET, PORT}; /* serverin adresi için */

    printf("CLIENT: başlıyor ...\n");

    /* Sunucu adresi. */
    srv.sin_addr.s_addr = inet_addr("127.0.0.1");

    /* SOCK_DGRAM tipinde Internet soketini oluştur. */
    if ((s = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
    {
        perror("Soket hatası");
        exit(1);
    }
}

```

```

/* Açılan soketi belirtilen porta bağla. */
if (bind(s, (struct sockaddr *) &cli, SIZE) < 0)
{
    perror("Bağlama problemi");
    exit(1);
}

/* s soketi üzerinden veri yolla */
n = sendto(s, "İstemcinin isteği\n", 30, 0,
            (struct sockaddr*)&srv, SIZE);
if (n < 0) {perror("Gönderim problemi"); exit(1);}

printf ("İstemci: İstek gönderildi...\n");

/*Soketi kapat ve çık.*/
close(s);
printf ("... ve kapatıldı\n");
exit(0);
}

```

Bağlantısız soketle başka bir sunucu programı

```

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <netdb.h>
#include <arpa/inet.h>
#include <ctype.h>

#define PORT 7000 /* Sunucunun port adresi */
#define SIZE sizeof(struct sockaddr_in)

int
main(void)
{
    char buf[1024];
    int n, s, ns, len;
    struct sockaddr_in srv = {AF_INET, PORT, INADDR_ANY};
    struct sockaddr_in cli; /* client'in adresi için */
    int cli_len = SIZE;

    printf("Sunucu: başlıyor ...\n");
}

```

```

/* SOCK_DGRAM tipinde soket oluřtur */
if ((s = socket(AF_INET, SOCK_DGRAM, 0)) < 0)
{
    perror("Soket problemi");
    exit(1);
}

/* Soketi sunucunun ip ve port numarasına bađla */
if (bind(s, (struct sockaddr *) &srv, SIZE) < 0)
{
    perror("Bađlama problemi");
    exit(1);
}
printf ("... ve 10 saniye uyuyor\\n");

/* Okumadan önc 10 saniye uyu */
sleep(10); /* 10 sec */

/* řimdi s soketinden veri oku */
n = recvfrom(s, buf, sizeof(buf), 0, (struct sockaddr *)&cli, &cli_len);
if (n < 0) {perror("Okuma problemi"); exit(1);}

write(1, buf, n);
printf ("SUNUCU řini tamamladı...\\n");

/* Soketi kapat ve çık */
close(s);
exit(0);
}

```