Laboratorio di Programmazione in Rete

a.a. 2005/2006

http://www.di.uniba.it/~lisi/courses/prog-rete/prog-rete0506.htm

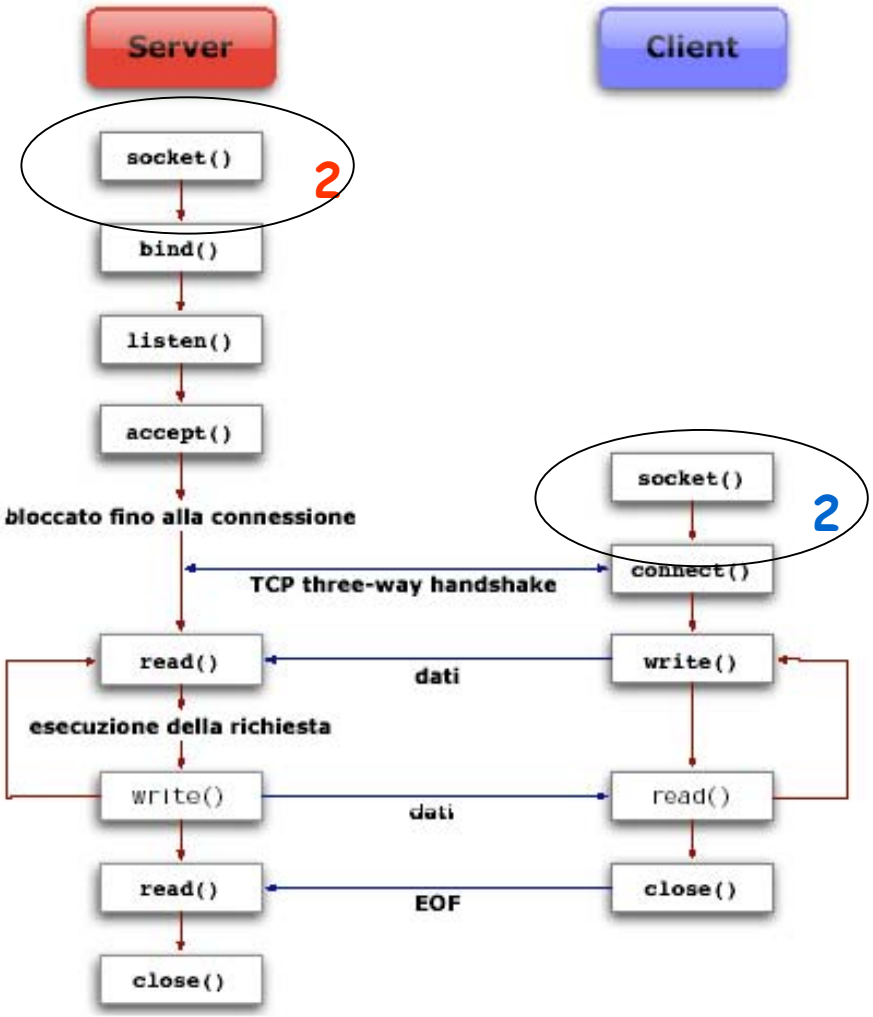
dott.ssa Francesca A. Lisi
lisi@di.uniba.it

Orario di ricevimento: mercoledì ore 10-12

N.B. Il presente materiale didattico è stato
prodotto da: dott.ssa V. **Carofiglio**
rielaborato da: dott.ssa F.A. **Lisi**

dib

Interazione TCP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare un local address alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
   a. Accettare una nuova connessione
   b. Inviare e ricevere dati
   c. Chiudere la connessione

Client

1. (Inizializzare una WSA)
2. Creare una Socket
3. Connettersi al server
4. Inviare e ricevere dati
5. Chiudere la connessione

Funzione socket()

Crea una socket dedicata ad un fornitore di servizi specifico

```
SOCKET socket(  int af, int type, int protocol );
```

Address family

(AF_INET: Internet Address Family)

Tipo di socket

Protocollo da usare con la socket per l'address family indicata

(solitamente posto a 0 indica il protocollo derivato dalla coppia [af, type])

Tipo	*Significato*
SOCK_STREAM	*Fornisce una connessione sequenziale, affidabile e full-duplex. Il protocollo TCP è basato su questo tipo di socket.*
SOCK_DGRAM	*Supporta i datagrammi (privo di connessione, messaggi inaffidabili di una lunghezza massima prefissata). Il protocollo UDP è basato su questo tipo di socket.*

Funzione socket( ): valori di ritorno

Crea una socket dedicata ad un fornitore di servizi specifico

```
SOCKET socket(  int af, int type, int protocol );
```

La funzione restituisce un intero che è interpretato come un *descrittore che referenzia la nuova socket* in caso di successo. Altrimenti restituisce un codice di errore

Funzione socket( ): valori di ritorno

Crea una socket dedicata ad un fornitore di servizi specifico

```
SOCKET socket(  int af, int type, int protocol );
```

La funzione restituisce un intero che è interpretato come un *descrittore che referenzia la nuova socket* in caso di successo. Altrimenti restituisce un codice di errore

ATTENZIONE!!!

Una applicazione client usa indirizzo IP e porta per connettersi

La funzione crea un socket senza nome

- Address Family
- Indirizzo IP
- Porta che identifica l'applicazione

↓

Bind()

Funzione socket(): uso

Dopo l'inizializzazione dell'applicazione WinSock, deve essere creata una variabile di tipo socket

Per creare una socket:

- Creare una variabile m_socket di tipo socket:

```
SOCKET m_socket;
```

- Avvalorare m_socket con il valore di ritorno della funzione socket():

```
m_socket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
```

Funzione socket(): uso

Dopo l'inizializzazione dell'applicazione WinSock, deve essere creata una variabile di tipo socket

Per creare una socket:

- Creare una variabile m_socket di tipo socket:

```
SOCKET m_socket;
```

- Avvalorare m_socket con il valore di ritorno della funzione socket():

```
m_socket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
```

In questo esempio la funzione usa l'internet address family, con socket di tipo stream e protocollo TCP/IP (↔ "0")

Funzione socket(): uso

Dopo l'inizializzazione dell'applicazione WinSock, deve essere creata una variabile di tipo socket

Per creare una socket:

- Creare una variabile m_socket di tipo socket:

```
SOCKET m_socket;
```

- Avvalorare m_socket con il valore di ritorno della funzione socket():

```
m_socket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
```

- Valutare la presenza di errori per assicurarsi che la socket sia valida

```
if ( m_socket == INVALID_SOCKET ) {
    printf( "Error at socket(): %ld\n", WSAGetLastError() );
    WSACleanup();
    return;
}
```

Funzione socket(): gestione dell'errore

Crea una socket dedicata ad un fornitore di servizi specifico

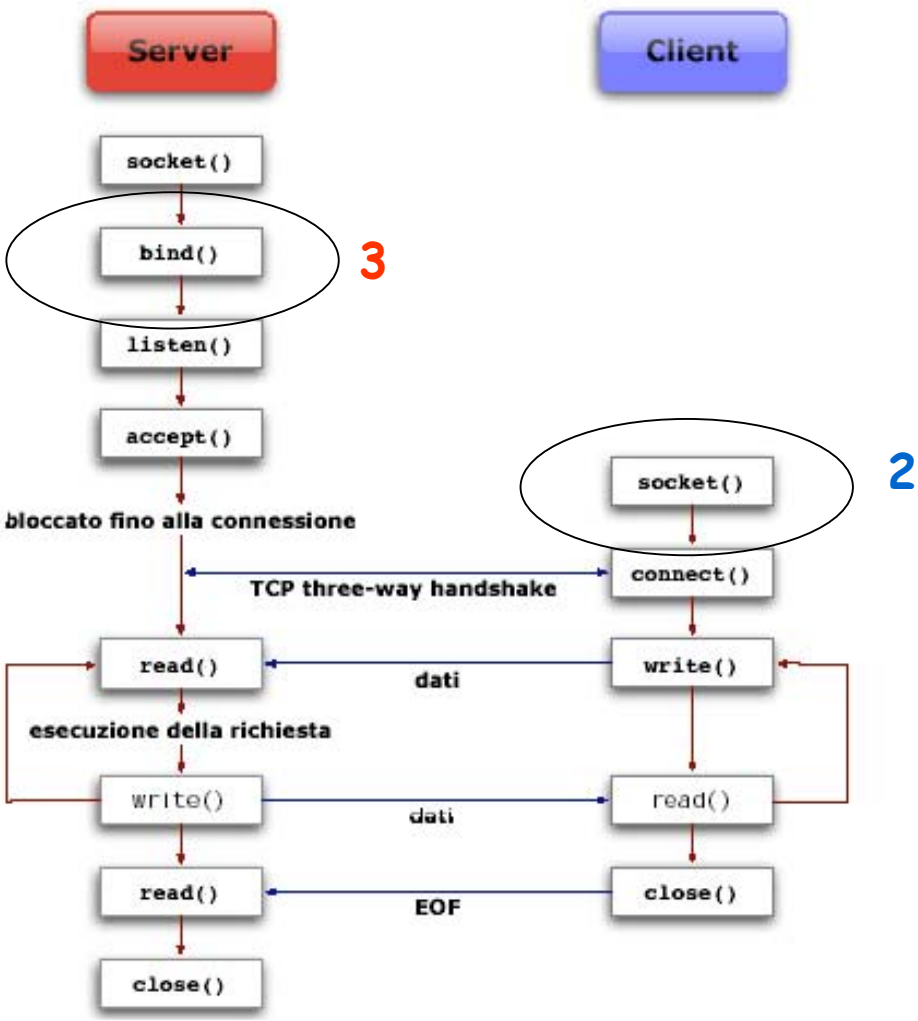
```
SOCKET socket(  int af, int type, int protocol );
```

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAENETDOWN	The network subsystem or the associated service provider has failed.
WSAEAFNOSUPPORT	The specified address family is not supported.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAEMFILE	No more socket descriptors are available.
WSAENOBUFS	No buffer space is available. The socket cannot be created.
WSAEPROTONOSUPPORT	The specified protocol is not supported.
WSAEPROTOTYPE	The specified protocol is the wrong type for this socket.
WSAESOCKTNOSUPPORT	The specified socket type is not supported in this address family.

Esempi di errori.

L'MSDN fornisce descrizioni di questo tipo per tutte le funzioni

Interazione TCP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare un local address alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
   a. Accettare una nuova connessione
   b. Inviare e ricevere dati
   c. Chiudere la connessione

Client

1. (Inizializzare una WSA)
2. Creare una Socket
3. Connettersi al server
4. Inviare e ricevere dati
5. Chiudere la connessione

Funzione bind()

Associa un *nome* alla socket creata in precedenza

```
int bind( SOCKET s, const struct sockaddr* name, int namelen);
```

Descrittore di un socket

Indirizzo da assegnare alla socket

Lunghezza in byte di name

Struttura sockaddr_in

La struttura **sockaddr**
è interpretata differentemente a seconda dei contesti determinati dalle differenti address family (AF_XXXX).

Forma di **sockaddr** In AF_INET (con protocollo IPv4)

```
struct sockaddr_in {
        short   sin_family;
        u_short sin_port;
        struct  in_addr sin_addr;
        char    sin_zero[8];
};
```

Un puntatore ad una **sockaddr** non è rigorosamente interpretato come tale

Le funzioni Winsock che fanno uso di un puntatore ad una struttura di tipo sockaddr devono necessariamente effettuare una operazione di "cast"

```
bind( m_socket, (SOCKADDR*) &service, sizeof(service)
```

Funzione bind( ): valori di ritorno

Associa un *nome* alla socket creata in precedenza

```
int bind( SOCKET s, const struct sockaddr* name, int namelen);
```

La funzione restituisce 'O' in caso di successo.
Altrimenti restituisce un codice di errore

Per il TCP/IP se la porta è specificata come zero, il fornitore di servizi assegna una porta tra 1024 e 5000
L'applicazione può usare la funzione getsockname (dopo la bind) per apprendere l'indirizzo IP e la porta assegnati

Funzione bind(): uso

Dopo la creazione di una socket

Per assegnare una porta ed un indirizzo IP alla socket

- Creare una variabile `service` di tipo `sockaddr_in`:

```
Sockaddr_in service;
```

Funzione bind(): uso

Dopo la creazione di una socket

Per assegnare una porta ed un indirizzo IP alla socket

- Creare una variabile service di tipo sockaddr_in:

```
Sockaddr_in service;
```

```
struct sockaddr_in {
        short   sin_family;
        u_short sin_port;
        struct  in_addr sin_addr;
        char    sin_zero[8];
};
```

Address family (AF_XXXX)

Porta TCP UDP

Indirizzo IP

Non Usato

```
struct in_addr {
        unsigned long s_addr;
};
```

Indirizzi internet 32 bit

Funzione bind(): uso

Dopo la creazione di una socket

Per assegnare una porta ed un indirizzo IP alla socket

- Creare una variabile `service` di tipo `sockaddr_in`:

```
Sockaddr_in service;
```

- Avvalorare la variabile creata

```
service.sin_family = AF_INET;
service.sin_addr.s_addr = inet_addr( "127.0.0.1" );
service.sin_port = htons( 27015 );
```

Local address

Funzione bind(): uso

Dopo la creazione di una socket

Per assegnare una porta ed un indirizzo IP alla socket

- Creare una variabile `service` di tipo `sockaddr_in`:

```
Sockaddr_in service;
```

- Avvalorare la variabile creata

```
service.sin_family = AF_INET;
service.sin_addr.s_addr = inet_addr( "127.0.0.1" );
service.sin_port = htons( 27015 );
```

- Assegnare porta e Ip alla socket. Valutare errori

```
if ( bind( m_socket, (SOCKADDR*) &service, sizeof(service) ) ==
   SOCKET_ERROR ) {
    printf( "bind() failed.\n" );
    closesocket(m_socket);
    return;
}
```

Funzione bind( ): gestione dell'errore

Crea una socket dedicata ad un fornitore di servizi specifico

```
int bind( SOCKET s, const struct sockaddr* name, int namelen);
```

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAENETDOWN	The network subsystem has failed.
WSAEACCES	Attempt to connect datagram socket to broadcast address failed because **setsockopt** option SO_BROADCAST is not enabled.
WSAEADDRINUSE	A process on the computer is already bound to the same fully-qualified address and the socket has not been marked to allow address reuse with SO_REUSEADDR. For example, the IP address and port are bound in the af_inet case). (See the SO_REUSEADDR socket option under **setsockopt**.)
WSAEADDRNOTAVAIL	The specified address is not a valid address for this computer.
WSAEFAULT	The *name* or *namelen* parameter is not a valid part of the user address space, the *namelen* parameter is too small, the *name* parameter contains an incorrect address format for the associated address family, or the first two bytes of the memory block specified by *name* does not match the address family associated with the socket descriptor *s*.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAEINVAL	The socket is already bound to an address.
WSAENOBUFS	Not enough buffers available, too many connections.
WSAENOTSOCK	The descriptor is not a socket.

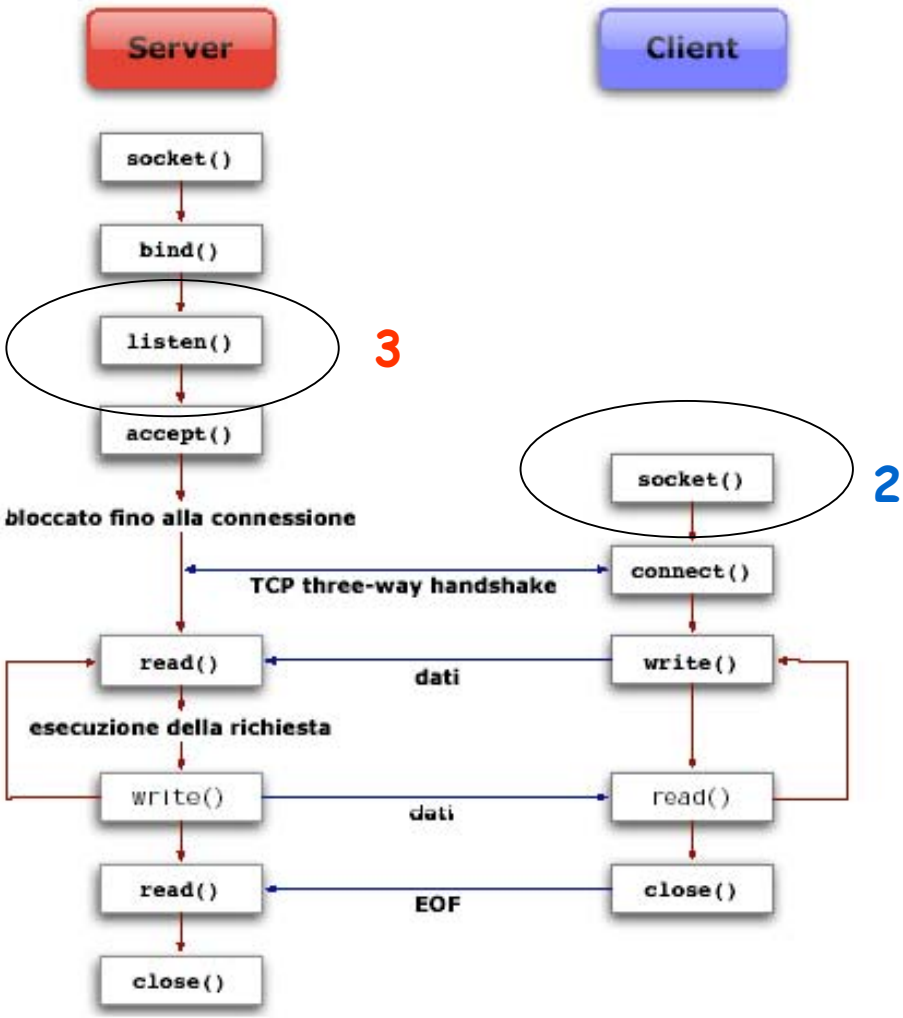
…un esempio di codice

```
#include <stdio.h>
#include "winsock2.h"

void main() {
 // Initialize Winsock
 WSADATA wsaData;
 int iResult = WSAStartup(MAKEWORD(2,2), &wsaData);
 if (iResult != NO_ERROR)
      printf("Error at WSAStartup()\n");

// Create a SOCKET for listening for incoming connection requests
  SOCKET ListenSocket;
  ListenSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
  if (ListenSocket == INVALID_SOCKET) {
       printf("Error at socket(): %ld\n", WSAGetLastError());
       WSACleanup();
       return;
      }

// The sockaddr_in structure specifies the address family, IP address,
and port for the socket that //is being bound.
  sockaddr_in service;
  service.sin_family = AF_INET;
  service.sin_addr.s_addr = inet_addr("127.0.0.1");
  service.sin_port = htons(27015);
```

…un esempio di codice (cont.)

```
// Bind the socket.
  if (bind( ListenSocket,   (SOCKADDR*) &service,   sizeof(service)) ==
SOCKET_ERROR) {
        printf("bind() failed.\n");
        closesocket(ListenSocket);
         return;
  }

  WSACleanup();
  return;
}
```

Interazione TCP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare una porta alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
   a. Accettare una nuova connessione
   b. Inviare e ricevere dati
   c. Chiudere la connessione

Client

1. (Inizializzare una WSA)
2. Creare una Socket
3. Connettersi al server
4. Inviare e ricevere dati
5. Chiudere la connessione

Funzione listen()

Setta la socket in uno stato in cui rimane in attesa di richiesta di connessioni

```
int listen(  SOCKET s,  int backlog);
```

Descrittore di un socket

Massima lungezza della coda di connessioni entranti

Funzione listen(): valori di ritorno

Setta la socket in uno stato in cui rimane in attesa di richiesta di connessioni

```
int listen(  SOCKET s,  int backlog);
```

Descrittore di un socket

Massima lungezza della coda di connessioni entranti

La funzione restituisce 'O' in caso di successo. Altrimenti restituisce un codice di errore

Funzione listen(): uso

Il server deve essere in grado di "ascoltare " una richiesta di connessione

Per settare l'ascolto

- Chiamare la funzione `listen()`, passando come parametri la socket creata e il massimo numero di connessioni che accetta

```
if ( listen( m_socket, 1 ) == SOCKET_ERROR )
    printf( "Error listening on socket.\n");
```

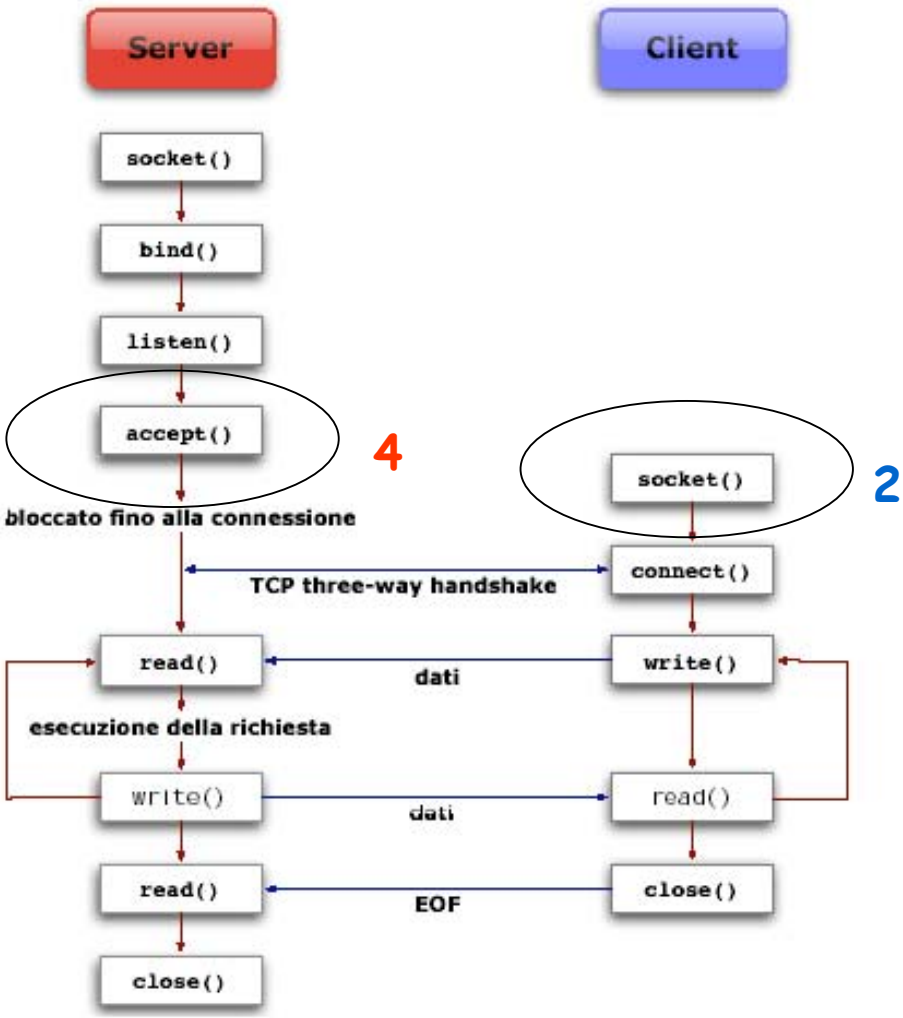
Funzione listen(): gestione dell'errore

Crea una socket dedicata ad un fornitore di servizi specifico

```
int listen(  SOCKET s,  int backlog);
```

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAENETDOWN	The network subsystem has failed.
WSAEADDRINUSE	The socket's local address is already in use and the socket was not marked to allow address reuse with SO_REUSEADDR. This error usually occurs during execution of the **bind** function, but could be delayed until this function if the **bind** was to a partially wildcard address (involving ADDR_ANY) and if a specific address needs to be committed at the time of this function.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAEINVAL	The socket has not been bound with **bind**.
WSAEISCONN	The socket is already connected.
WSAEMFILE	No more socket descriptors are available.
WSAENOBUFS	No buffer space is available.
WSAENOTSOCK	The descriptor is not a socket.
WSAEOPNOTSUPP	The referenced socket is not of a type that supports the **listen** operation.

Interazione TCP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare una porta alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
   a. Accettare una nuova connessione
   b. Inviare e ricevere dati
   c. Chiudere la connessione

Client

1. (Inizializzare una WSA)
2. Creare una Socket
3. Connettersi al server
4. Inviare e ricevere dati
5. Chiudere la connessione

Funzione accept()

Consente un tentativo di connessione in entrata su una socket

```
SOCKET accept(  SOCKET s,  struct sockaddr* addr,  int* addrlen);
```

Descrittore di un socket

Puntatore opzionale ad un buffer che riceve l'indirizzo dell'entità che fa richiesta di connessione

Puntatore opzionale che contiene la lunghezza di addr

Funzione accept()

Consente un tentativo di connessione in entrata su una socket

```
SOCKET accept(  SOCKET s,  struct sockaddr* addr,  int* addrlen);
```

- La funzione estrae la prima connessione dalla coda di pendenza delle connessioni sulla socket s.
- Successivamente crea e restituisce un riferimento ad una nuova socket.
- Questa nuova socket è quella che abbiamo chiamata socket di connessione.
- Mantiene le stesse proprietà della socket s

Funzione `accept()`: uso

Una volta in ascolto, il server deve essere in grado di gestire le richieste di nuove connessioni

Per accettare una connessione su una socket

- Creare una variabile `AcceptSocket` di tipo `SOCKET`

```
SOCKET AcceptSocket;
```

Funzione accept():uso

Una volta in ascolto, il server deve essere in grado di gestire le richieste di nuove connessioni

Per accettare una connessione su una socket

- Creare una variabile `AcceptSocket` di tipo `SOCKET`

```
SOCKET AcceptSocket;
```

- Effettuare un ciclo continuo per determinare se ci sono richieste di connessione e chiamare la funzione `accept()` per accettare eventuali connessioni

```
printf( "Waiting for a client to connect...\n" );
while (1) {
    AcceptSocket = SOCKET_ERROR;
    while ( AcceptSocket == SOCKET_ERROR ) {
        AcceptSocket = accept( m_socket, NULL, NULL );
    }
```

Funzione accept( ):gestione dell'errore

Consente un tentativo di connessione in entrata su una socket

```
SOCKET accept(  SOCKET s,  struct sockaddr* addr,  int* addrlen);
```

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAECONNRESET	An incoming connection was indicated, but was subsequently terminated by the remote peer prior to accepting the call.
WSAEFAULT	The *addrlen* parameter is too small or *addr* is not a valid part of the user address space.
WSAEINTR	A blocking Windows Sockets 1.1 call was canceled through **WSACancelBlockingCall**.
WSAEINVAL	The **listen** function was not invoked prior to **accept**.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAEMFILE	The queue is nonempty upon entry to **accept** and there are no descriptors available.
WSAENETDOWN	The network subsystem has failed.
WSAENOBUFS	No buffer space is available.
WSAENOTSOCK	The descriptor is not a socket.
WSAEOPNOTSUPP	The referenced socket is not a type that supports connection-oriented service.
WSAEWOULDBLOCK	The socket is marked as nonblocking and no connections are present to be accepted.

Funzione accept():uso

Una volta in ascolto, il server deve essere in grado di gestire le richieste di nuove connessioni

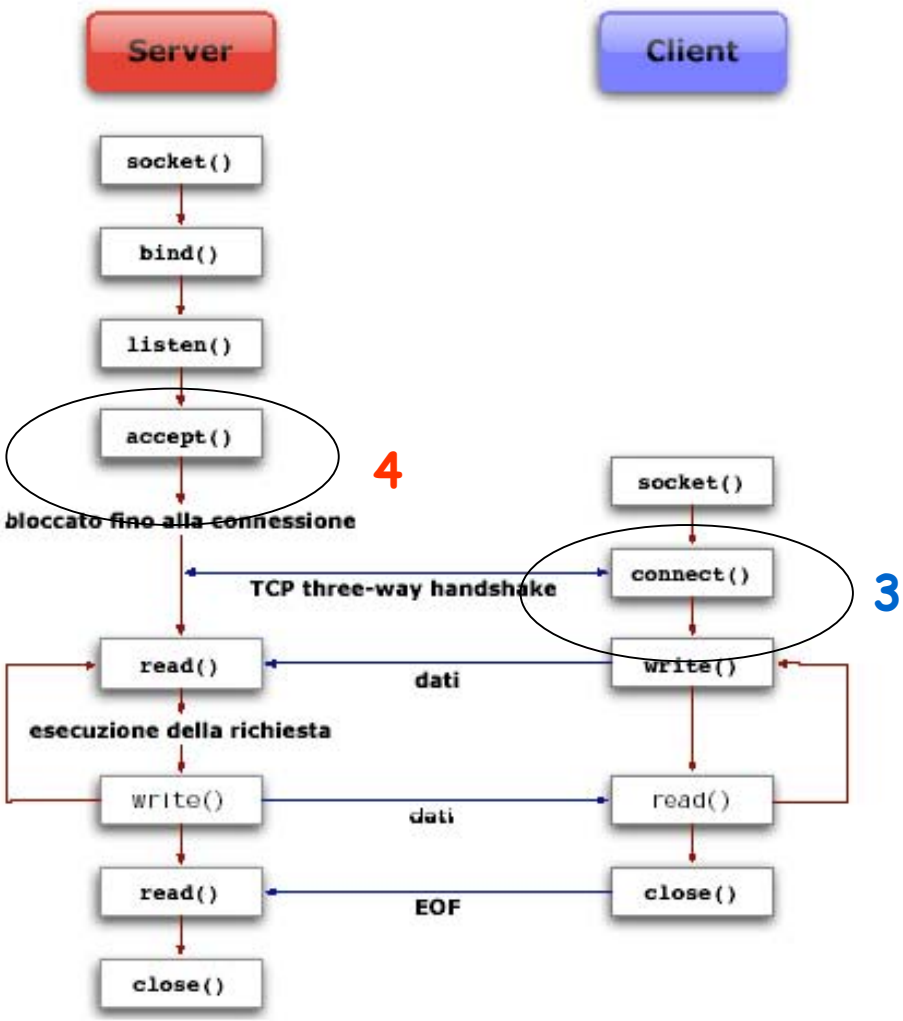
Per accettare una connessione su una socket

```
    SOCKET AcceptSocket;

    printf( "Waiting for a client to connect...\n" );
    while (1) {
        AcceptSocket = SOCKET_ERROR;
        while ( AcceptSocket == SOCKET_ERROR ) {
            AcceptSocket = accept( m_socket, NULL, NULL );
        }
```

- Quando la connessione del client è stata accettata: trasferire il controllo dal socket temporanea a quella originale e interrompere la ricerca di nuove connessioni

```
printf( "Client Connected.\n");
    m_socket = AcceptSocket;
    break;
}
```

Interazione TCP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare una porta alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
   a. Accettare una nuova connessione
   b. Inviare e ricevere dati
   c. Chiudere la connessione

Client

1. (Inizializzare una WSA)
2. Creare una Socket
3. Connettersi al server
4. Inviare e ricevere dati
5. Chiudere la connessione

Funzione connect()

Stabilisce una connessione ad una socket specificata

```
int connect( SOCKET s, const struct sockaddr* name, int namelen);
```

Descrittore di una socket non connessa

Nome della socket con cui dovrebbe essere stabilita la connessione

lunghezza di name

Funzione connect()

Stabilisce una connessione ad una socket specificata

```
int connect( SOCKET s, const struct sockaddr* name, int namelen);
```

Per una socket di tipo "connectionless" (per esempio SOCK_DGRAM), la connect() semplicemente stabilisce un indirizzo di destinazione di default

Qualunque datagram ricevuto da un indirizzo diverso da quello di destinazione verrà scaricato

Se il campo indirizzo della struttura che specifica il name è zero, la socket verrà disconnessa

Funzione connect():uso

Affinchè un client possa comunicare con un server, si deve connettere al server

Per connettersi ad un server

- Creare un elemento `clientService` di tipo `sockaddr_in` e settare il suo valore

```
sockaddr_in clientService;
clientService.sin_family = AF_INET;
clientService.sin_addr.s_addr = inet_addr( "127.0.0.1" );
clientService.sin_port = htons( 27015 );
```

Funzione connect( ):uso

Affinchè un client possa comunicare con un server, si deve connettere al server

Per connettersi ad un server

- Creare un elemento `clientService` di tipo `sockaddr_in` e settare il suo valore

```
sockaddr_in clientService;
clientService.sin_family = AF_INET;
clientService.sin_addr.s_addr = inet_addr( "127.0.0.1" );
clientService.sin_port = htons( 27015 );
```

- Chiamare la funzione `connect ( )` passando come parametri la socket creata e settata all'ascolto e la struttura `sockaddr_in`. Verificare la presenza di errori

```
if ( connect( m_socket, (SOCKADDR*) &clientService, sizeof(clientService) )
   == SOCKET_ERROR) {
    printf( "Failed to connect.\n" );
    WSACleanup();
    return;
}
```

Funzione connect( ):gestione dell'errore

Stabilisce una connessione ad una socket specificata

```
int connect( SOCKET s, const struct sockaddr* name, int namelen);
```

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAENETDOWN	The network subsystem has failed.
WSAEADDRINUSE	The socket's local address is already in use and the socket was not marked to allow address reuse with SO_REUSEADDR. This error usually occurs when executing **bind**, but could be delayed until this function if the **bind** was to a partially wildcard address (involving ADDR_ANY) and if a specific address needs to be committed at the time of this function.
WSAEINTR	The blocking Windows Socket 1.1 call was canceled through **WSACancelBlockingCall**.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAEALREADY	A nonblocking **connect** call is in progress on the specified socket. **Note** In order to preserve backward compatibility, this error is reported as WSAEINVAL to Windows Sockets 1.1 applications that link to either Winsock.dll or Wsock32.dll.
WSAEADDRNOTAVAIL	The remote address is not a valid address (such as ADDR_ANY).
WSAEAFNOSUPPORT	Addresses in the specified family cannot be used with this socket.
WSAECONNREFUSED	The attempt to connect was forcefully rejected.

Funzione connect( ) : gestione dell'errore(cont.)

Stabilisce una connessione ad una socket specificata

```
int connect( SOCKET s, const struct sockaddr* name, int namelen);
```

WSAEFAULT	The *name* or the *namelen* parameter is not a valid part of the user address space, the *namelen* parameter is too small, or the *name* parameter contains incorrect address format for the associated address family.
WSAEINVAL	The parameter *s* is a listening socket.
WSAEISCONN	The socket is already connected (connection-oriented sockets only).
WSAENETUNREACH	The network cannot be reached from this host at this time.
WSAEHOSTUNREACH	A socket operation was attempted to an unreachable host.
WSAENOBUFS	No buffer space is available. The socket cannot be connected.
WSAENOTSOCK	The descriptor is not a socket.
WSAETIMEDOUT	Attempt to connect timed out without establishing a connection.
WSAEWOULDBLOCK	The socket is marked as nonblocking and the connection cannot be completed immediately.
WSAEACCES	Attempt to connect datagram socket to broadcast address failed because **setsockopt** option SO_BROADCAST is not enabled.

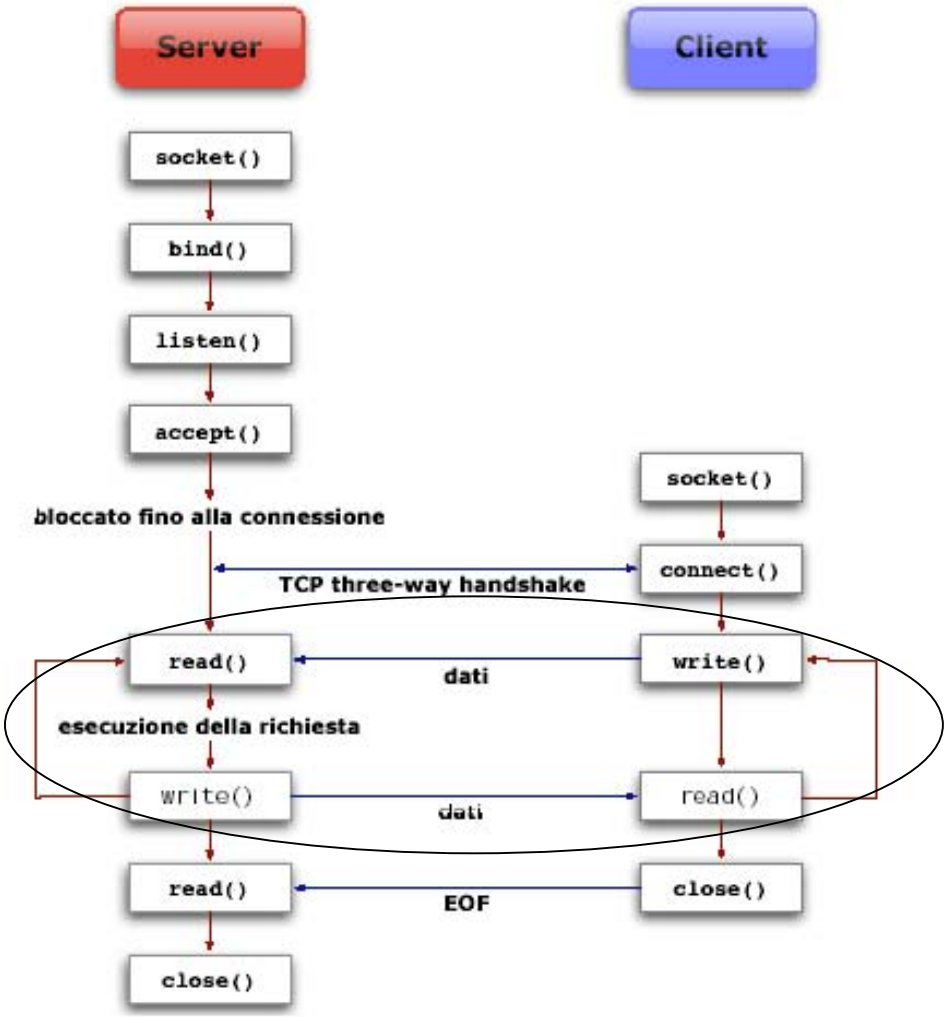
…un esempio di codice

```
#include <stdio.h>
#include "winsock2.h"
void main() {

// Initialize Winsock
  WSADATA wsaData;
  int iResult = WSAStartup(MAKEWORD(2,2), &wsaData);
  if (iResult != NO_ERROR)
    printf("Error at WSAStartup()\n");


// Create a SOCKET for connecting to server
  SOCKET ConnectSocket;
  ConnectSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
  if (ConnectSocket == INVALID_SOCKET) {
    printf("Error at socket(): %ld\n", WSAGetLastError());
    WSACleanup();
    return;
  }
```

…un esempio di codice (cont.)

```
// The sockaddr_in structure specifies the address family,
  // IP address, and port of the server to be connected to.
  sockaddr_in clientService;
  clientService.sin_family = AF_INET;
  clientService.sin_addr.s_addr = inet_addr( "127.0.0.1" );
  clientService.sin_port = htons( 27015 );


// Connect to server.
  if ( connect( ConnectSocket, (SOCKADDR*) &clientService,
sizeof(clientService) ) == SOCKET_ERROR) {
    printf( "Failed to connect.\n" );
    WSACleanup();
    return;
  }
  printf("Connected to server.\n");
  WSACleanup();
  return;
}
```

Interazione TCP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare un local address alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
   a. Accettare una nuova connessione
   b. Inviare e ricevere dati
   c. Chiudere la connessione

Client

1. (Inizializzare una WSA)
2. Creare una Socket
3. Connettersi al server
4. Inviare e ricevere dati
5. Chiudere la connessione

Funzione send( )

Invia dati ad una socket connessa

int send(SOCKET *s*, const char* *buf*, int *len*, int *flags*);

Descrittore di una socket connessa

Puntatore al Buffer contenente I dati da trasmettere

Lunghezza dei dati in buf, in byte

Indicatore che specifica il modo in cui la chiamata è fatta

Il flag può essere usato per influenzare il comportamento della funzione

Funzione send( ): valori di ritorno

Invia dati ad una socket connessa

int send(SOCKET *s*, const char* *buf*, int *len*, int *flags*);

Descrittore di una socket connessa

Buffer contenente I dati da trasmettere

Lunghezza dei dati in buf, in byte

Indicatore che specifica il modo in cui la chiamata è fatta

Il flag può essere usato per influenzare il comportamento della funzione

La funzione restituisce il numero di byte trasmessi in caso di successo. Un codice di errore, altrimenti

Funzione recev()

Riceve dati da una socket connessa (o "legata")

int recev(SOCKET *s*, char* *buf*, int *len*, int *flags*);

Descrittore di una socket connessa

Puntatore al Buffer contenente I dati da ricevere

Lunghezza dei dati in buf, in byte

Indicatore che specifica il modo in cui la chiamata è fatta

Il flag può essere usato per influenzare il comportamento della funzione

Inviare e ricevere dati (lato Server)

Per inviare e ricevere dati

```
int bytesSent;
int bytesRecv = SOCKET_ERROR;
char sendbuf[32] = "Server: Sending Data.";
char recvbuf[32] = "";

bytesRecv = recv( m_socket, recvbuf, 32, 0 );
printf( "Bytes Recv: %ld\n", bytesRecv );
bytesSent = send( m_socket, sendbuf, strlen(sendbuf), 0 );
printf( "Bytes Sent: %ld\n", bytesSent );
```

Due interi sono usati per tenere traccia del numero di byte che vengono inviati e ricevuti.

Inviare e ricevere dati (lato Client)

Per inviare e ricevere dati

```
int bytesSent;
int bytesRecv = SOCKET_ERROR;
char sendbuf[32] = "Client: Sending data.";
char recvbuf[32] = "";

bytesSent = send( m_socket, sendbuf, strlen(sendbuf), 0 );
printf( "Bytes Sent: %ld\n", bytesSent );
if ( bytesRecv == 0 ||
   (bytesRecv == SOCKET_ERROR && WSAGetLastError()== WSAECONNRESET ))
   {
      bytesRecv = recv( m_socket, recvbuf, 32, 0 );
           if ( bytesRecv == -1 ) {
               printf( "Connection Closed.\n");
               break;
                }
          if (bytesRecv < 0)
             return;
          printf( "Bytes Recv: %ld\n", bytesRecv );
        }
```

Due interi sono usati per tenere traccia del numero di byte che vengono inviati e ricevuti.

Funzione send( ):gestione dell'errore

Invia dati ad una socket connessa

int send(SOCKET *s*, const char* *buf*, int *len*, int *flags*);

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAENETDOWN	The network subsystem has failed.
WSAEACCES	The requested address is a broadcast address, but the appropriate flag was not set. Call **setsockopt** with the SO_BROADCAST socket option to enable use of the broadcast address.
WSAEINTR	A blocking Windows Sockets 1.1 call was canceled through WSACancelBlockingCall.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAEFAULT	The *buf* parameter is not completely contained in a valid part of the user address space.
WSAENETRESET	The connection has been broken due to the keep-alive activity detecting a failure while the operation was in progress.
WSAENOBUFS	No buffer space is available.
WSAENOTCONN	The socket is not connected.
WSAENOTSOCK	The descriptor is not a socket.
WSAEOPNOTSUPP	MSG_OOB was specified, but the socket is not stream-style such as type SOCK_STREAM, OOB data is not supported in the communication domain associated with this socket, or the socket is unidirectional and supports only receive operations.

Funzione send( ):gestione dell'errore(cont.)

Invia dati ad una socket connessa

int send(SOCKET *s*, const char* *buf*, int *len*, int *flags*);

WSAESHUTDOWN	The socket has been shut down; it is not possible to send on a socket after **shutdown** has been invoked with *how* set to SD_SEND or SD_BOTH.
WSAEWOULDBLOCK	The socket is marked as nonblocking and the requested operation would block.
WSAEMSGSIZE	The socket is message oriented, and the message is larger than the maximum supported by the underlying transport.
WSAEHOSTUNREACH	The remote host cannot be reached from this host at this time.
WSAEINVAL	The socket has not been bound with **bind**, or an unknown flag was specified, or MSG_OOB was specified for a socket with SO_OOBINLINE enabled.
WSAECONNABORTED	The virtual circuit was terminated due to a time-out or other failure. The application should close the socket as it is no longer usable.
WSAECONNRESET	The virtual circuit was reset by the remote side executing a hard or abortive close. For UDP sockets, the remote host was unable to deliver a previously sent UDP datagram and responded with a "*Port Unreachable*" ICMP packet. The application should close the socket as it is no longer usable.
WSAETIMEDOUT	The connection has been dropped, because of a network failure or because the system on the other end went down without notice.

Funzione recev( ):gestione dell'errore

Riceve dati da una socket connessa (o "legata")

int recev(SOCKET *s*, char* *buf*, int *len*, int *flags*);

Error code	Meaning
WSANOTINITIALISED	A successful **WSAStartup** call must occur before using this function.
WSAENETDOWN	The network subsystem has failed.
WSAEFAULT	The *buf* parameter is not completely contained in a valid part of the user address space.
WSAENOTCONN	The socket is not connected.
WSAEINTR	The (blocking) call was canceled through **WSACancelBlockingCall**.
WSAEINPROGRESS	A blocking Windows Sockets 1.1 call is in progress, or the service provider is still processing a callback function.
WSAENETRESET	The connection has been broken due to the *keep-alive* activity detecting a failure while the operation was in progress.
WSAENOTSOCK	The descriptor is not a socket.
WSAEOPNOTSUPP	MSG_OOB was specified, but the socket is not stream-style such as type SOCK_STREAM, OOB data is not supported in the communication domain associated with this socket, or the socket is unidirectional and supports only send operations.
WSAESHUTDOWN	The socket has been shut down; it is not possible to receive on a socket after **shutdown** has been invoked with **how** set to SD_RECEIVE or SD_BOTH.
WSAEWOULDBLOCK	The socket is marked as nonblocking and the receive operation would block.

Funzione recev( ):gestione dell'errore (cont.)

Riceve dati da una socket connessa (o "legata")

int recev(SOCKET *s*, char* *buf*, int *len*, int *flags*);

WSAEMSGSIZE	The message was too large to fit into the specified buffer and was truncated.
WSAEINVAL	The socket has not been bound with **bind**, or an unknown flag was specified, or MSG_OOB was specified for a socket with SO_OOBINLINE enabled or (for byte stream sockets only) *len* was zero or negative.
WSAECONNABORTED	The virtual circuit was terminated due to a time-out or other failure. The application should close the socket as it is no longer usable.
WSAETIMEDOUT	The connection has been dropped because of a network failure or because the peer system failed to respond.
WSAECONNRESET	The virtual circuit was reset by the remote side executing a hard or abortive close. The application should close the socket as it is no longer usable. On a UPD-datagram socket this error would indicate that a previous send operation resulted in an ICMP "Port Unreachable" message.

Esercizi

- Ricomporre l'esempio riportato in queste slide
- Fare il porting su Windows degli esempi riportati nel cap. 2 del testo di Donahoo & Calvert
- Svolgere esercizi 2, 5, e 6 a pag. 23 del testo