

Laboratorio di Programmazione in rete

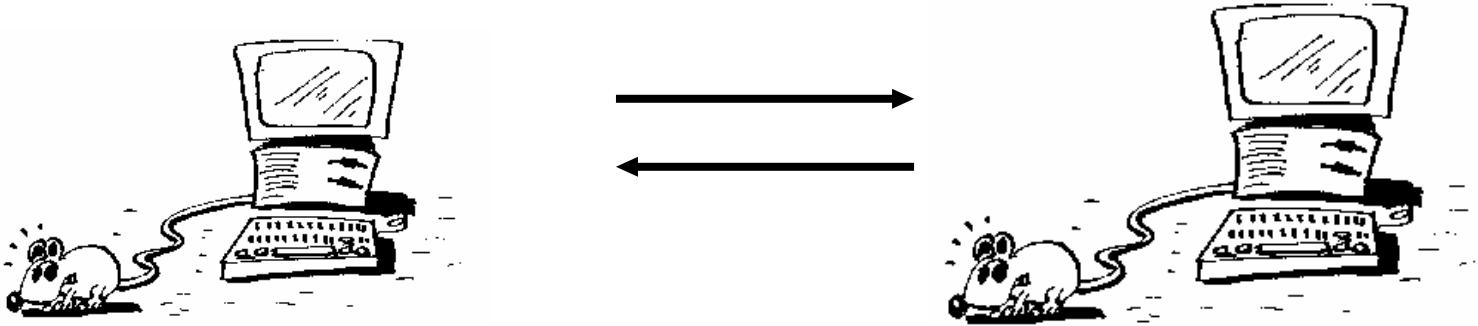
Introduzione
alla programmazione C di socket

A.A. 2005/06

Dott.ssa Valeria Carofiglio

Comunicazione tra computer

Come far comunicare più computer su una rete?



Una collezione di protocolli: TCP/IP

Internet Protocol (IP)

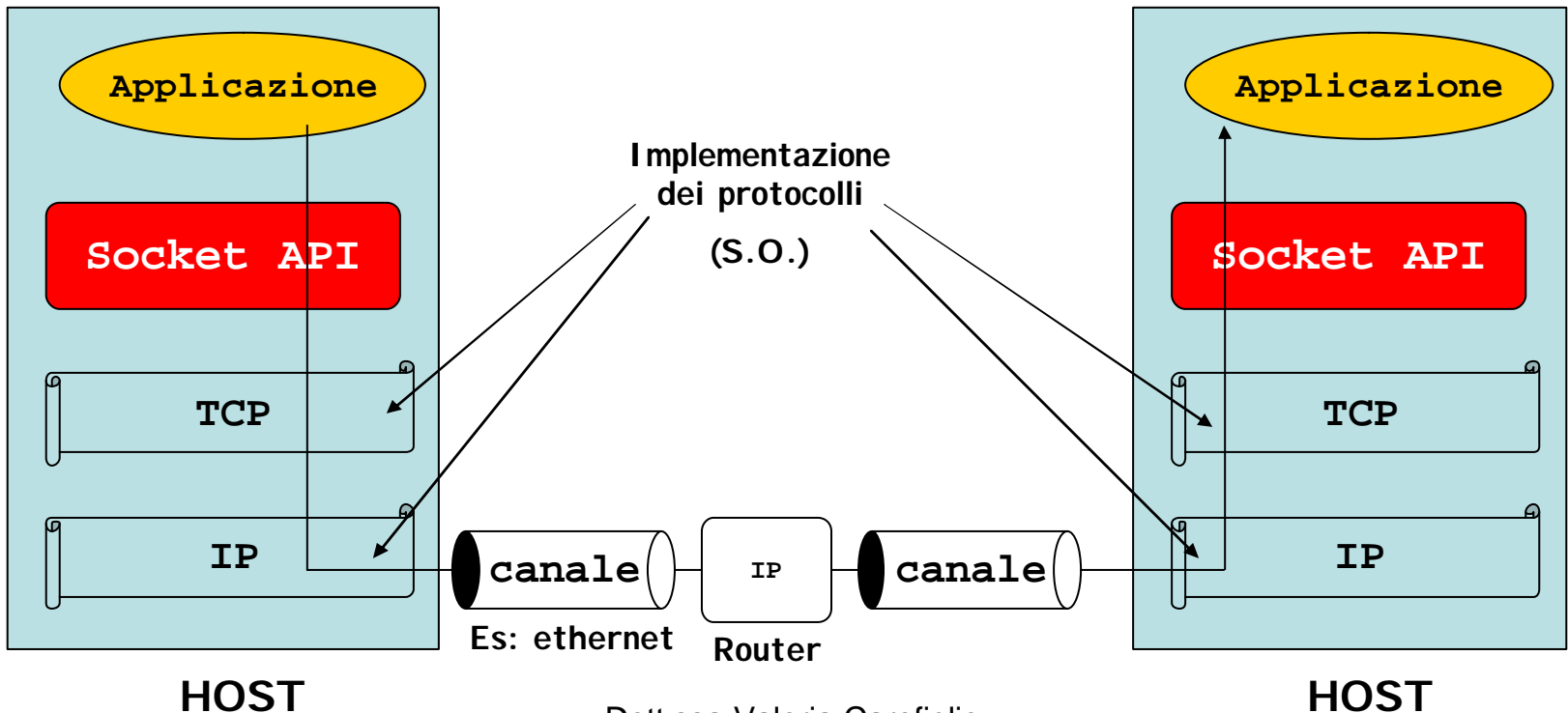
Trasmision Control Protocol (TCP)

User Datagam Protocol (UDP)

Dott.ssa Valeria Carofiglio

Una rete TCP/IP

Organizzazione a livelli:
relazione tra protocolli, applicazioni e API socket



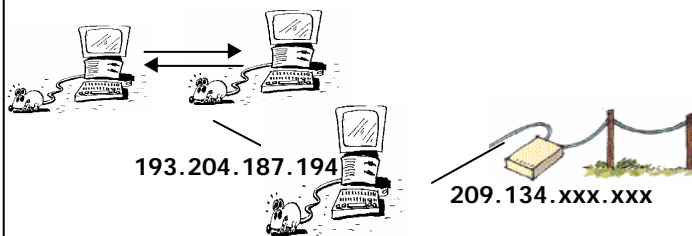
I indirizzi

La comunicazione tra programmi è possibile se si conosce *l'indirizzo* dei programmi che devono comunicare

Indirizzo Internet

(usato da IP)

- identificatori a 32 bit
- notazione: 193.204.187.194
- identifica una interfaccia



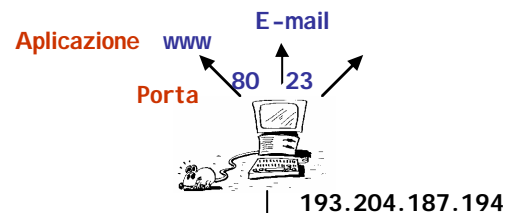
- `www.di.uniba.it` → 193.204.1..

Numero di porta

(interpretato da TCP o UDP)

- identificatori a 16 bit (sempre in relazione ad un IP)
- range:
 - Well-Known: 0 → 1023
 - Registered: 1024 → 49151
 - Dynamic (private) 49152 → 65535

<http://www.iana.org/assignments/port-numbers>



Una Socket: una visione di insieme

- **Una socket** è un dispositivo che consente la comunicazione (trasferimento di dati) tra due processi su internet, in una LAN, su un singolo computer...

Una Socket: una visione di insieme (cont.)

- Esistono varie **famiglie di socket**. Ogni famiglia
 - riunisce i socket che utilizzano gli stessi protocolli (Protocol Family) sottostanti,
 - supporta un sottoinsieme di stili di comunicazione e
 - possiede un proprio formato di indirizzamento (Address Family)

Alcuni esempi di famiglie

- **Unix Domain sockets**: file in una directory di un computer local host. Consentono il trasferimento di dati tra processi sulla stessa macchina Unix
- **Internet socket (AF_INET)**: consentono il trasferimento di dati tra processi posti su macchine remote connesse tramite una LAN o Internet

Una Socket: una visione di insieme (cont.)

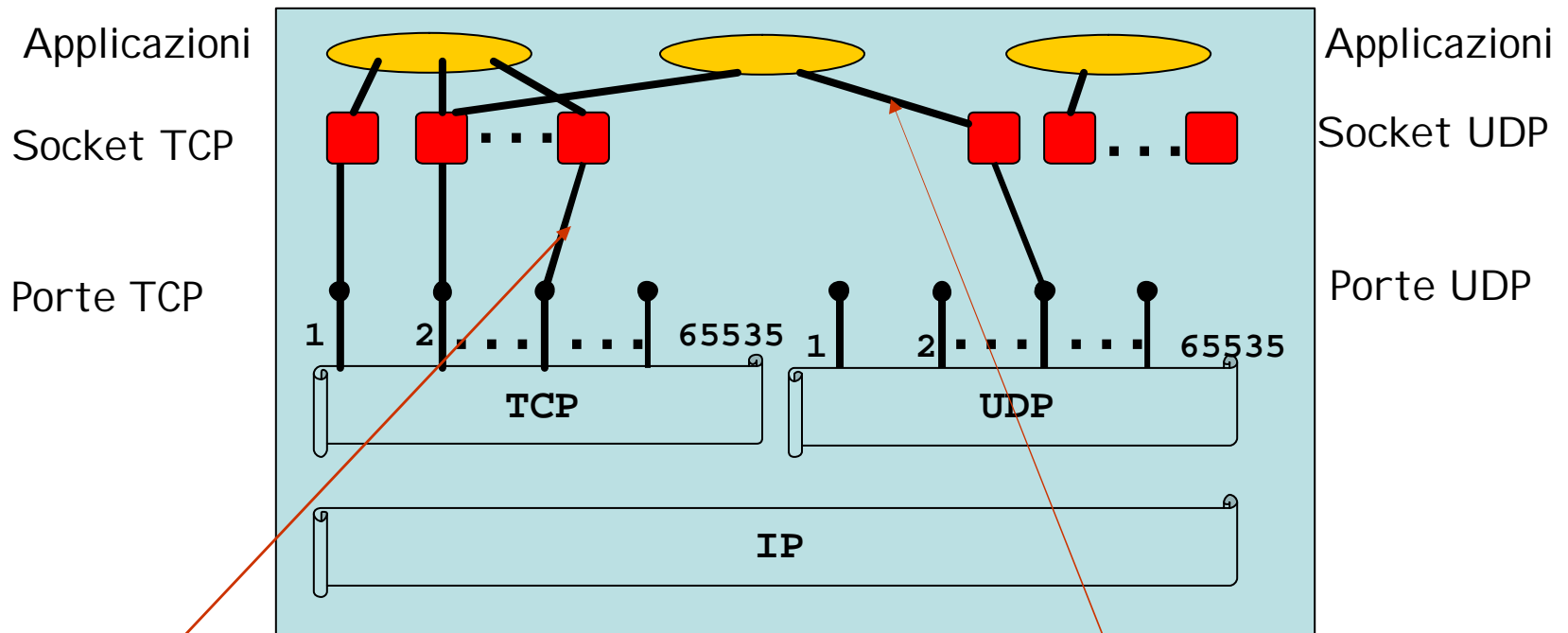
- Il **tipo** di una socket definisce una modalità di comunicazione che una socket usa per inviare dati:
 - **Streaming Socket** (SOCK_STREAM): Fornisce una connessione sequenziale, affidabile e full-duplex. Il protocollo TCP è basato su questo tipo di socket.
 - **Datagram socket** (SOCK_DGRAM): Supporta i datagrammi (privo di connessione, messaggi inaffidabili di una lunghezza massima prefissata). Il protocollo UDP è basato su questo tipo di socket

Osserviamo che:

AF_INET + SOCK_STREAM determineranno una connessione **TCP**,
AF_INET + SOCK_DGRAM determineranno una trasmissione **UDP**

Socket, Protocolli e Porte su un singolo host

Una socket che usa la famiglia di protocolli TCP/IP è univocamente determinata da *un indirizzo internet*, un protocollo di comunicazione (TCP o UDP) e un numero di porta



Sockets bound to ports

Dott.ssa Valeria Carofiglio

Descriptor reference

Una Socket: una visione di insieme (cont.)

- Nel gergo socket uno dei processi che comunicano è chiamato **Server** e l'altro **Client**. Tra i due processi il server è quello che ha controllo maggiore, poiché è il processo che inizialmente crea la socket. **Più client possono comunicare attraverso la stessa socket**, ma solo un server può essere associato ad una definita socket.

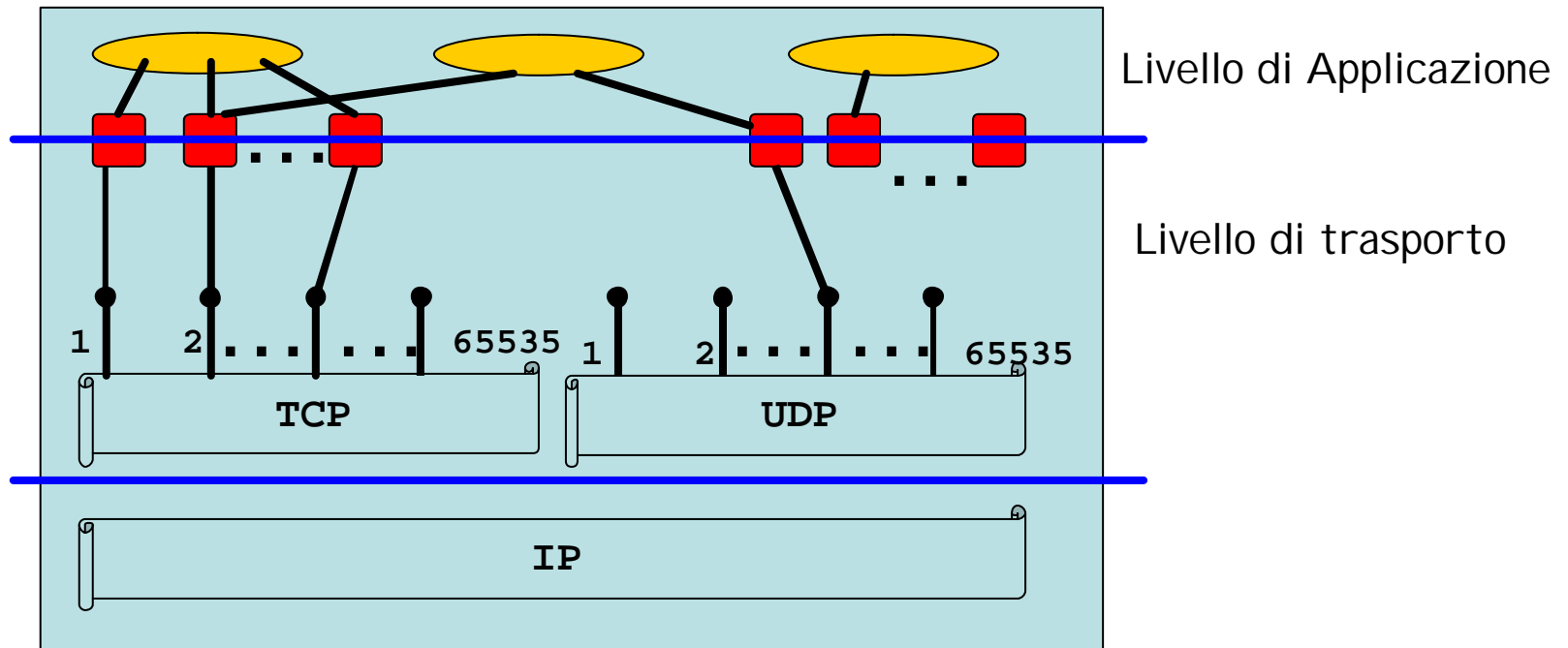
Client e Server

Il fatto che un programma agisca come client o come server determina un
differente uso delle API Socket

- Il client ha bisogno di conoscere l'indirizzo del server (ma non il viceversa)
- Il Server può apprendere informazioni sull'indirizzo del client una volta stabilita la connessione

Programmazione socket

Lo sviluppatore ha il controllo di tutto ciò che sta sul lato del livello applicativo della socket, ma ha poco controllo sul lato a livello di trasporto (alcuni parametri)



Host

Programmazione socket (cont.)

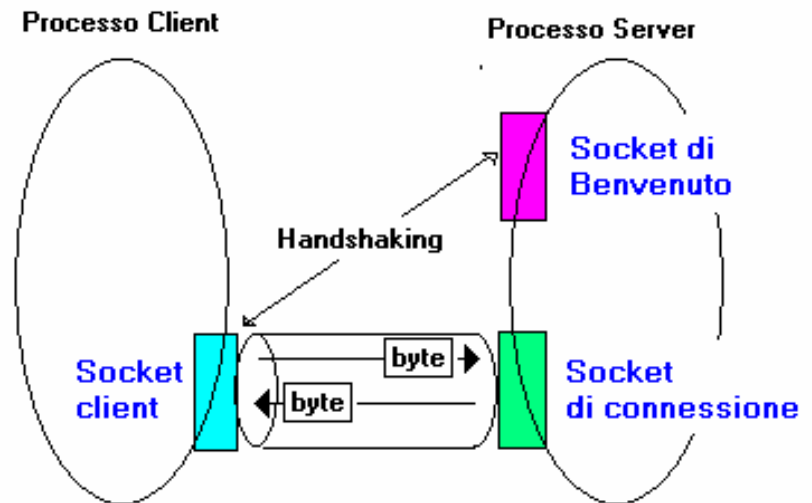
Il client deve contattare il server

- Il programma server deve essere in esecuzione come processo
- Il programma server deve avere una porta (socket) che dia il benvenuto al contatto iniziale stabilito da un processo client in esecuzione

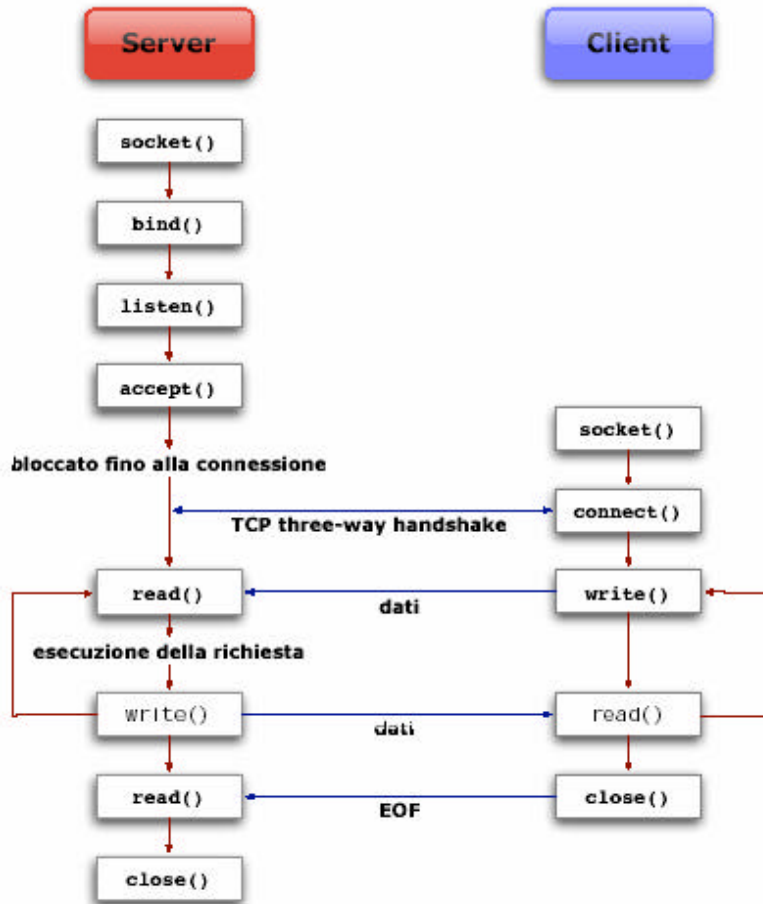
Il client contatta il server tramite:

- la creazione di una socket locale
- la specifica di un indirizzo del processo server (IP, numero di porta relativi al processo)
- *Dopo la creazione della socket nel client:* TCP avvia un handshake a tre vie e stabilisce una connessione TCP con il server

- *Durante l'handshake a tre vie:* il TCP server crea una nuova socket (dedicata a quel particolare client - socket di connessione)



Interazione TCP Client/Server



Server

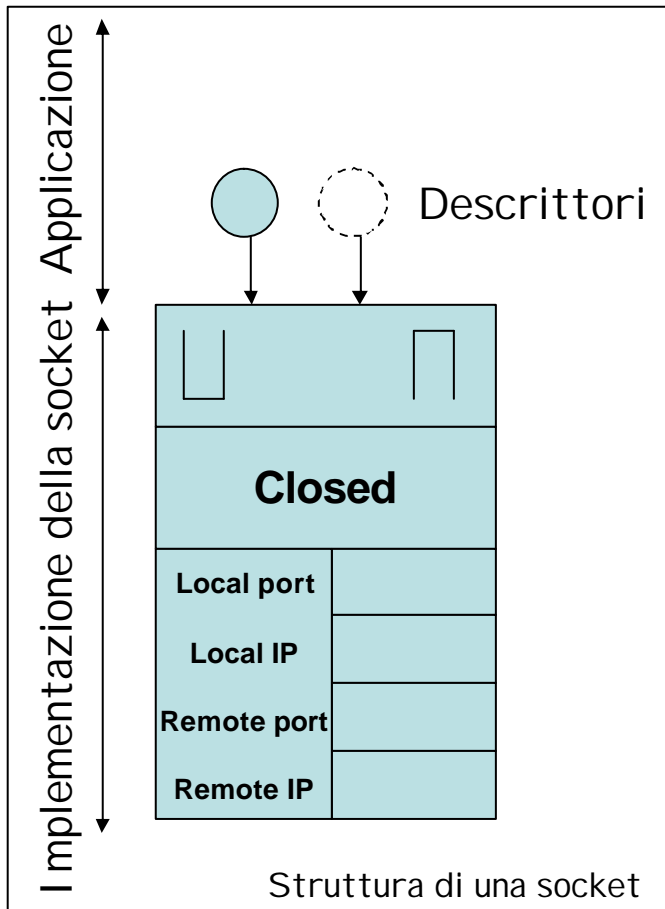
1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare un local address alla socket
4. Settare la socket all'ascolto
5. Iterativamente:
 - a. Accettare una nuova connessione
 - b. Inviare e ricevere dati
 - c. Chiudere la connessione

Client

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

Una astrazione di una socket:

Strutture dati associate ad una socket TCP



- L'applicazione fa riferimento alla struttura di una socket tramite *descrittori*

- differenti processi possono fare riferimento alla stessa struttura socket

- informazioni associate alla struttura socket:

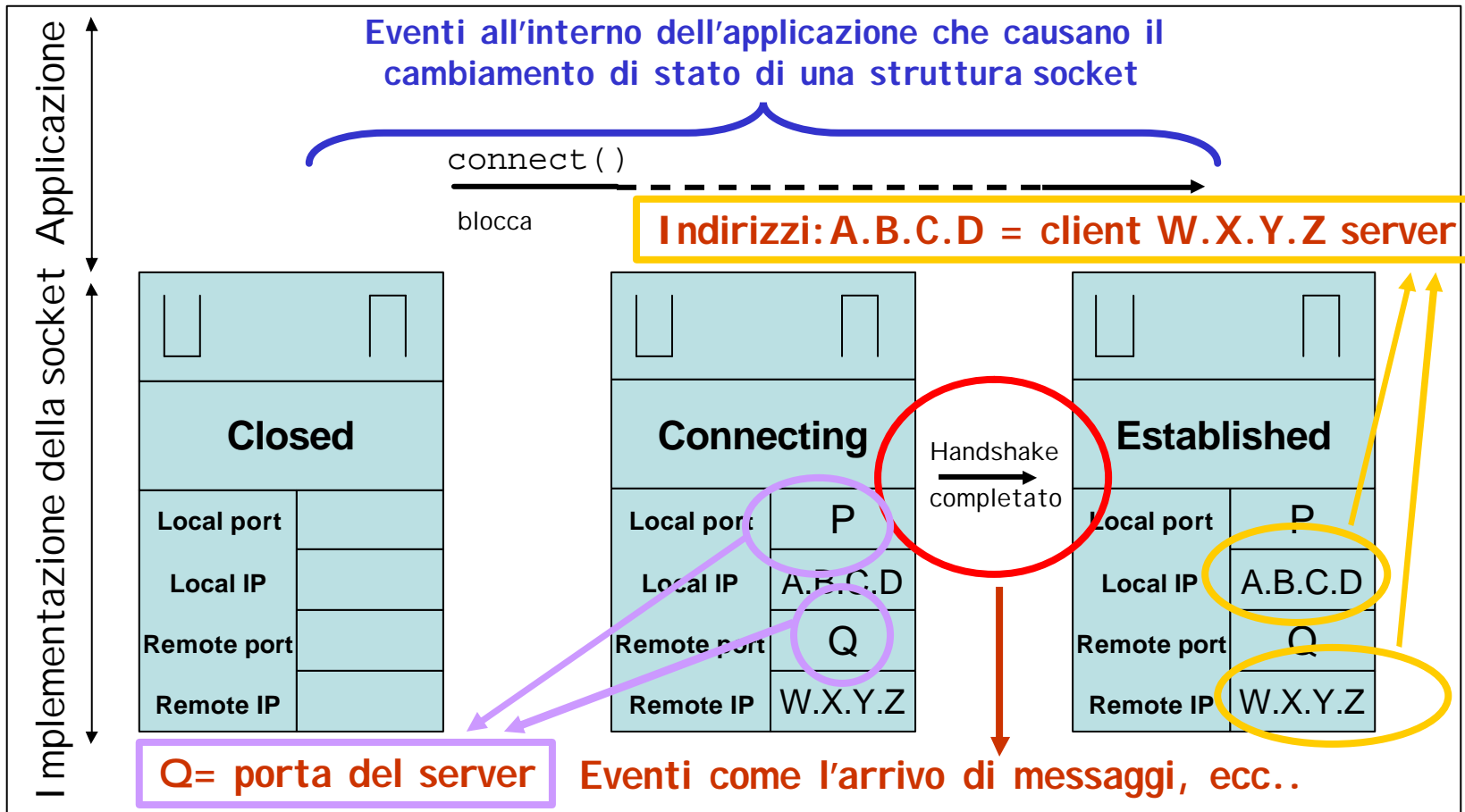
- code di ricezione e invio

- informazioni sullo stato dell'handshake (per una socket TCP)

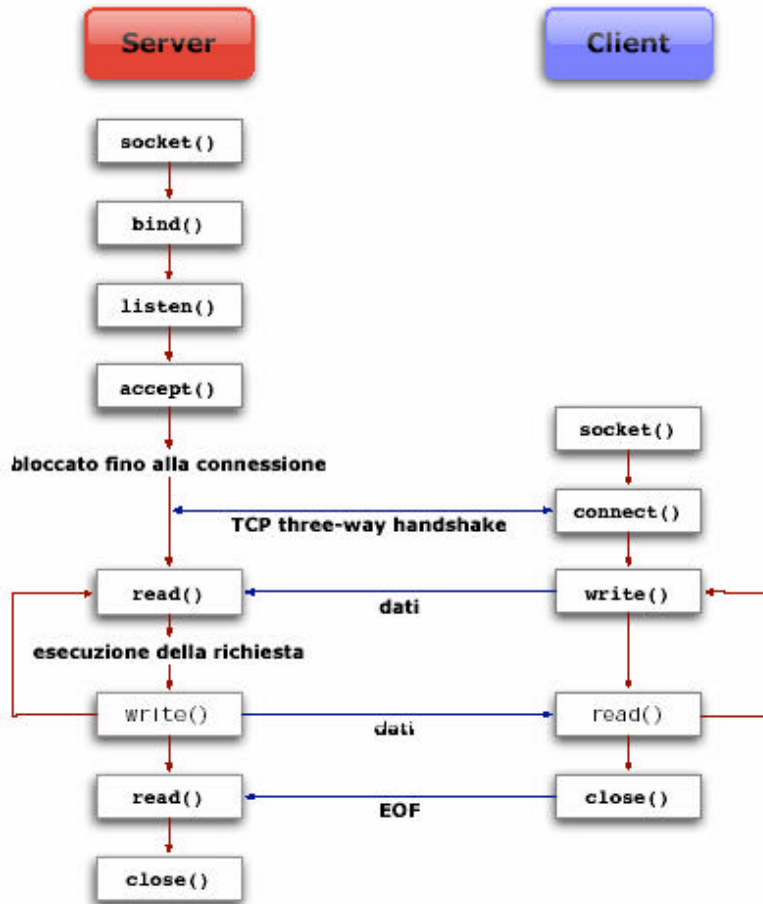
- indirizzi internet

Il ciclo di vita di una socket TCP

Notazione



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

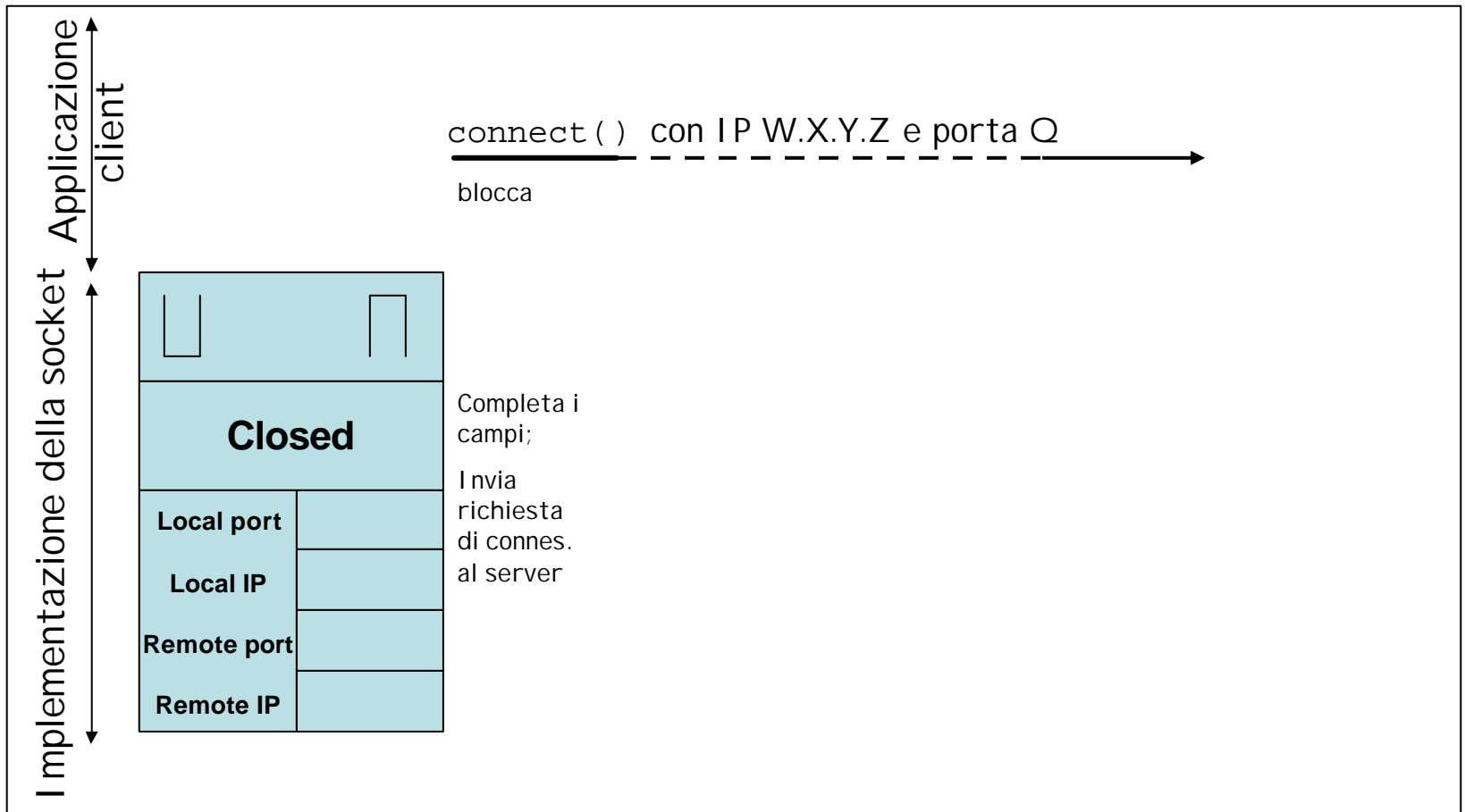
Il ciclo di vita di una socket TCP

Stabilire una connessione (lato client)



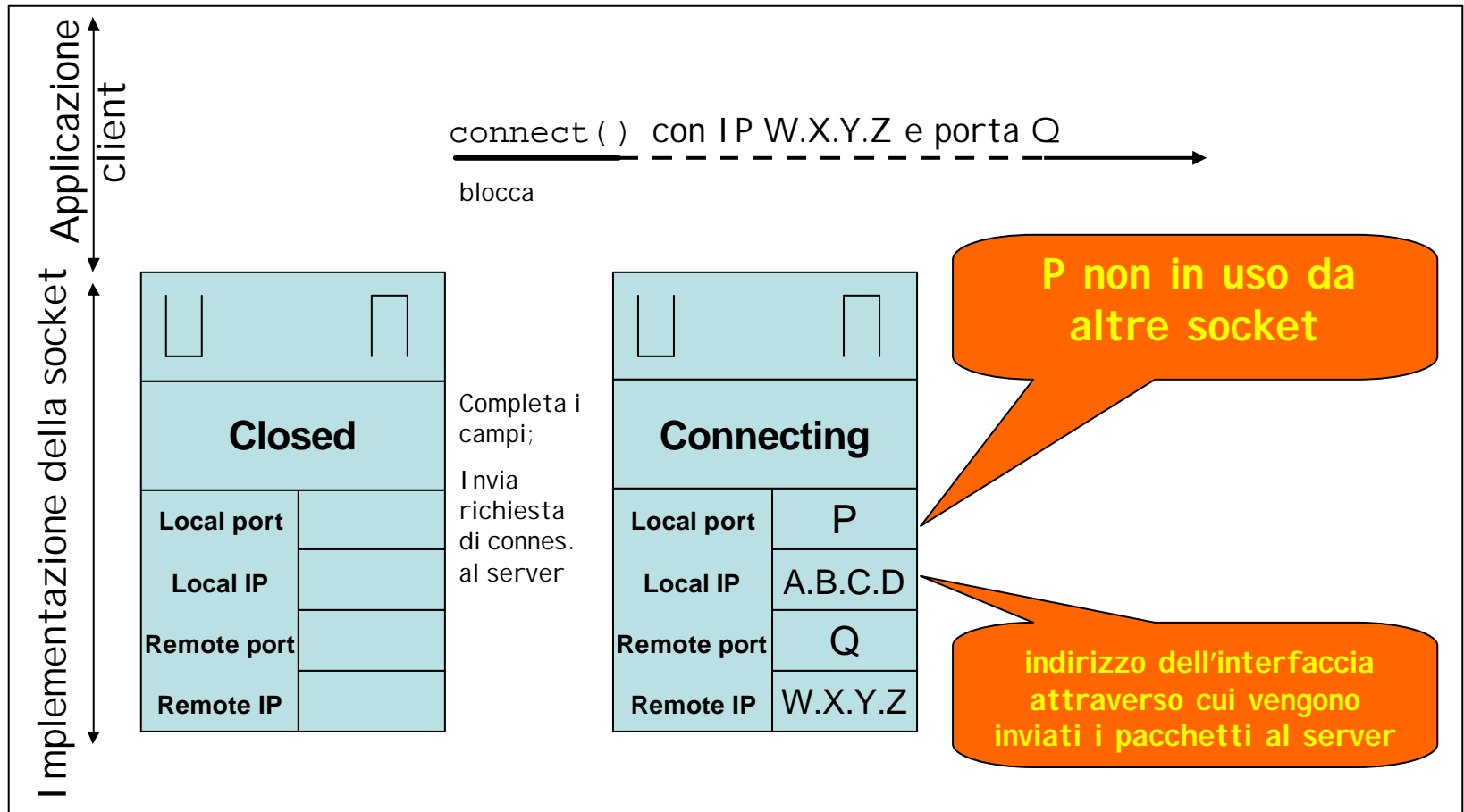
Il ciclo di vita di una socket TCP

Stabilire una connessione (lato client)



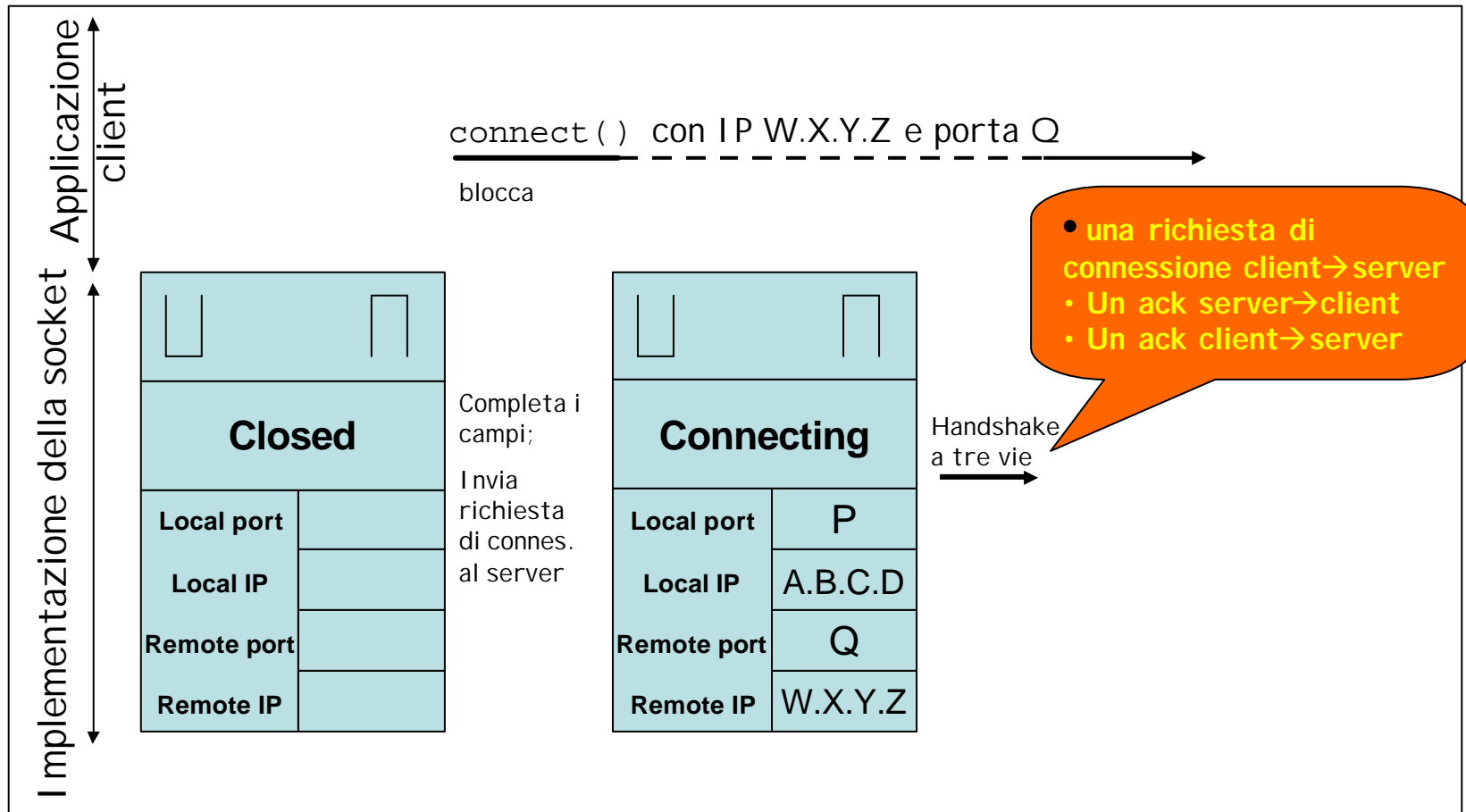
Il ciclo di vita di una socket TCP

Stabilire una connessione (lato client)



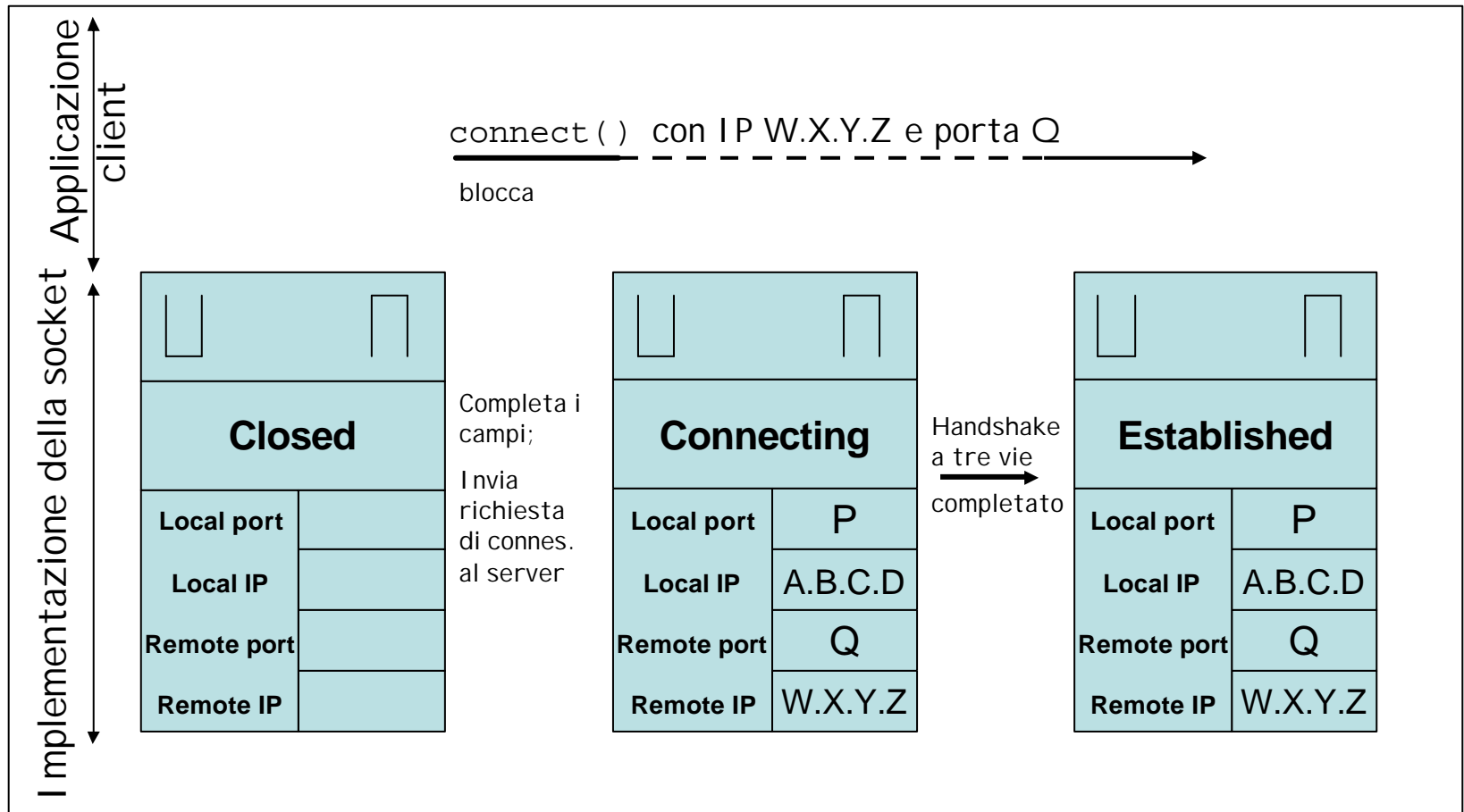
Il ciclo di vita di una socket TCP

Stabilire una connessione (lato client)



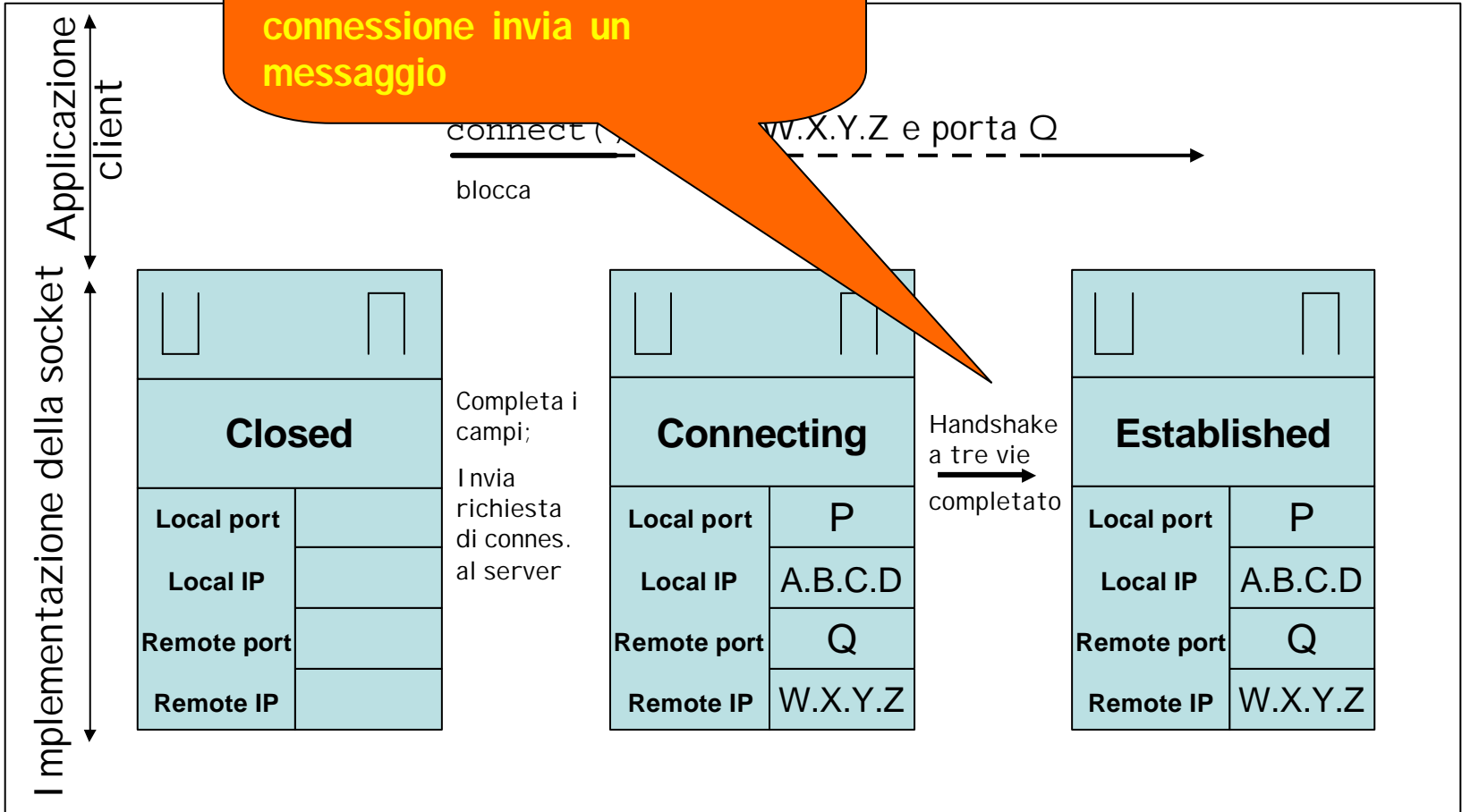
Il ciclo di vita di una socket TCP

Stabilire una connessione (lato client)



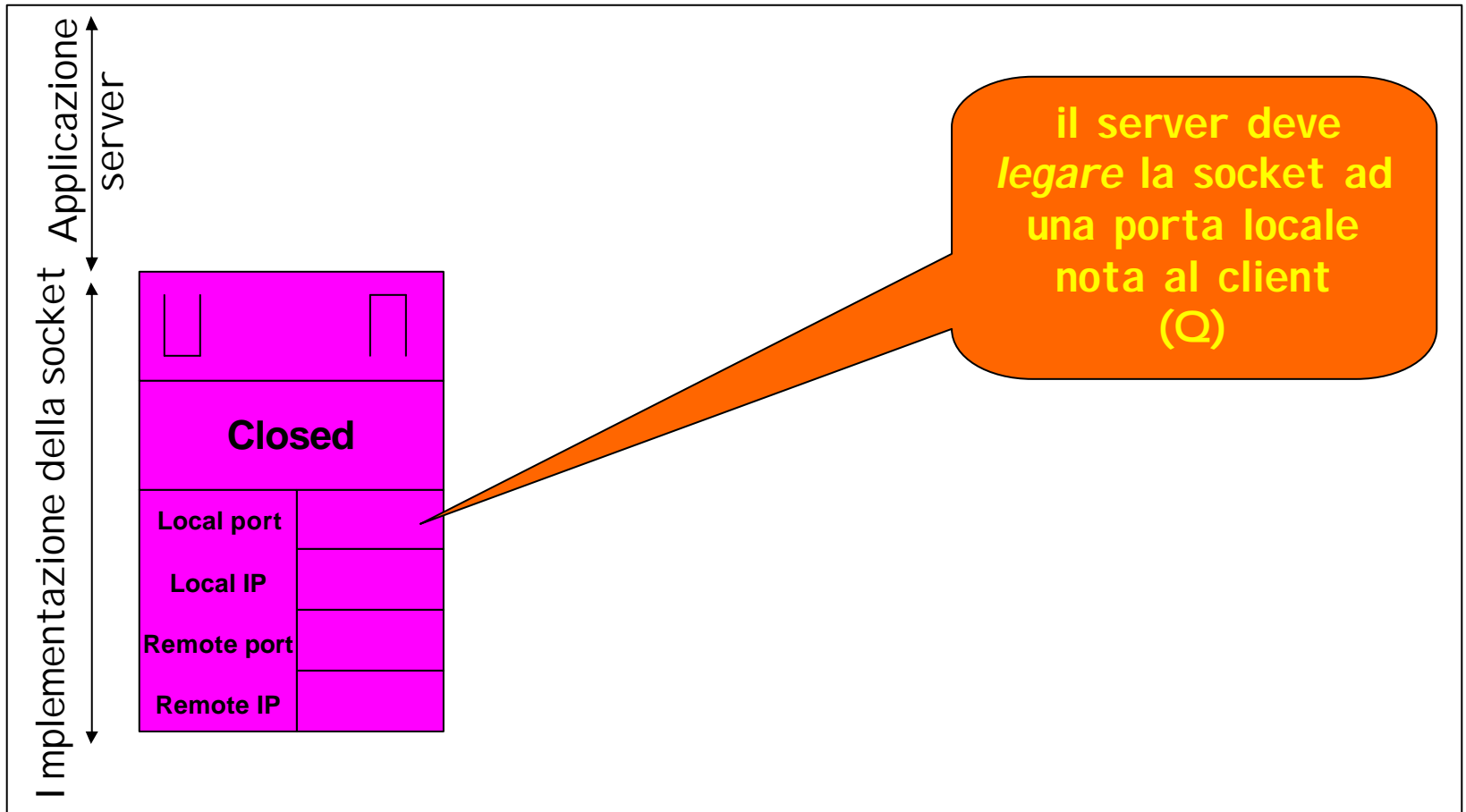
Il client e la socket TCP (lato client)

- il client considera la connessione stabilita non appena riceve l'ack dal server
- se il server non accetta una connessione invia un messaggio



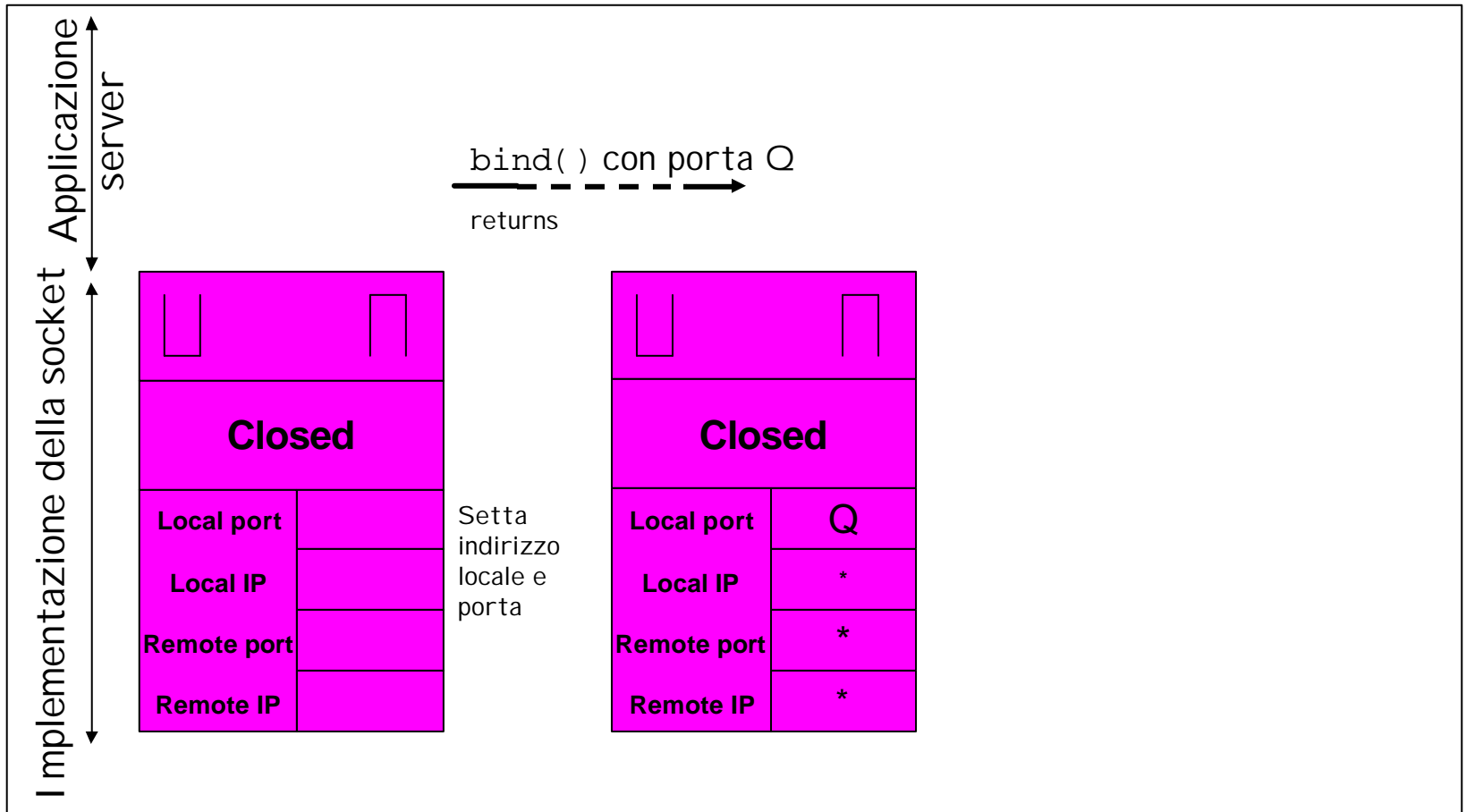
Il ciclo di vita di una socket TCP(cont.)

Setup della socket (lato server)



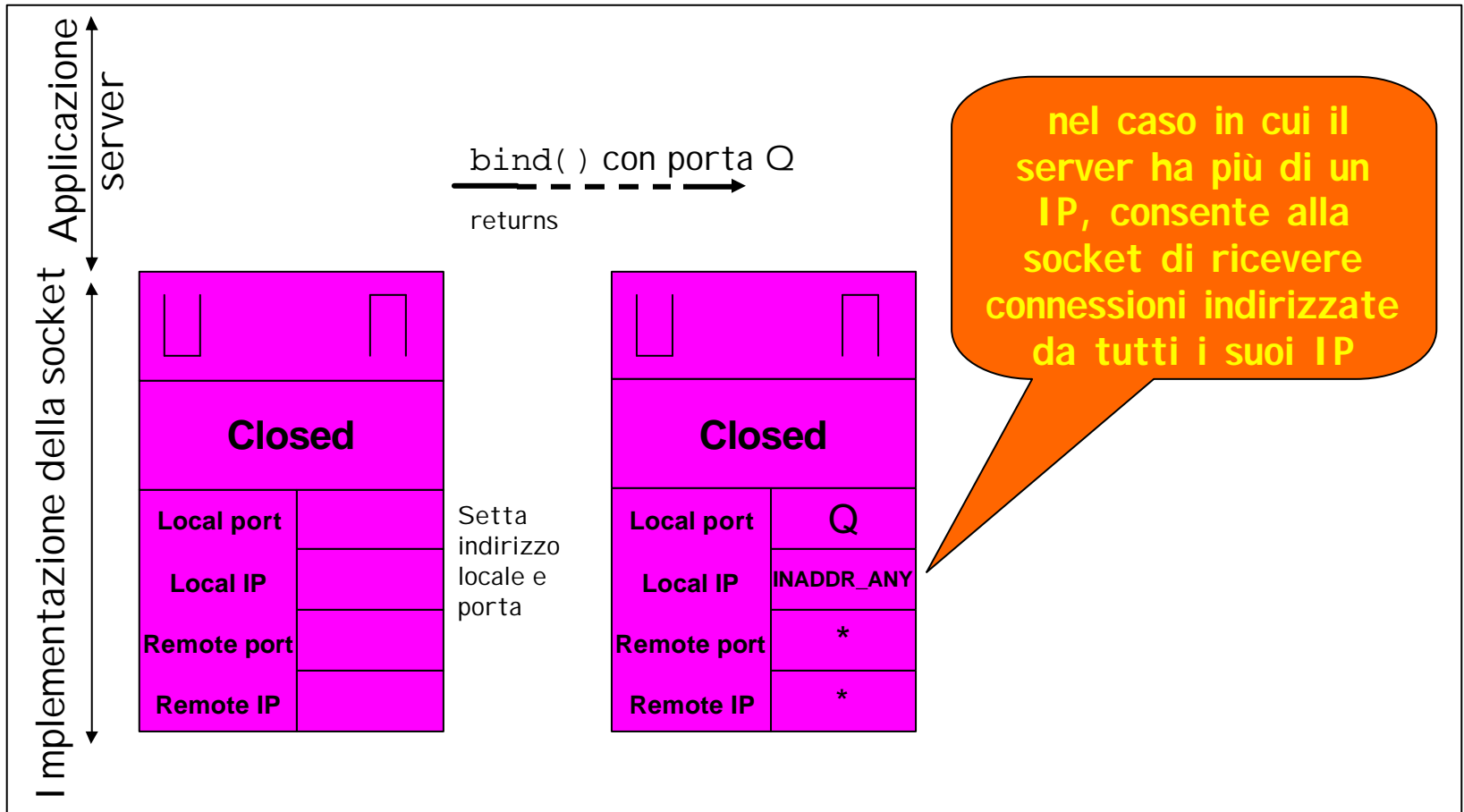
Il ciclo di vita di una socket TCP(cont.)

Setup della socket (lato server)



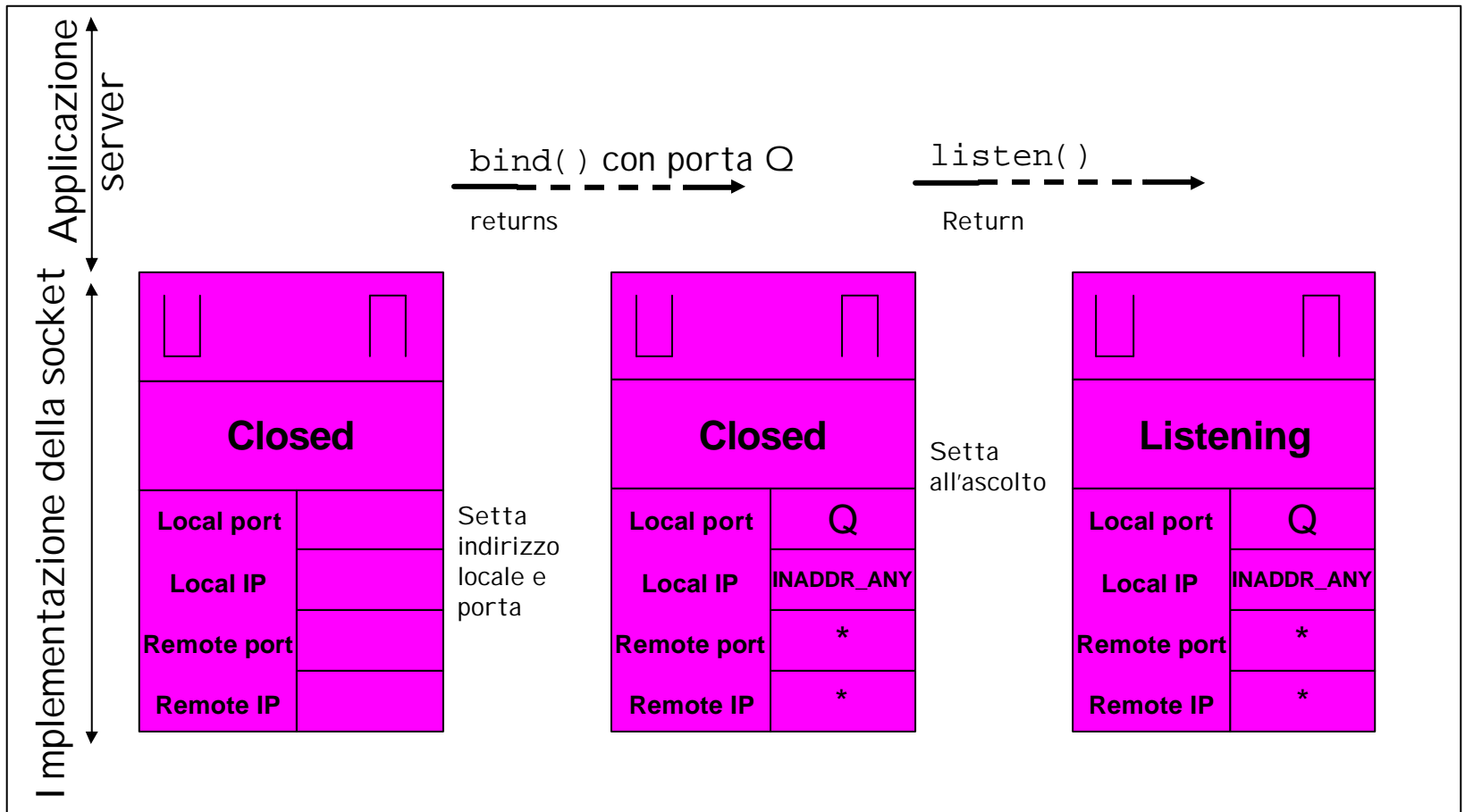
Il ciclo di vita di una socket TCP(cont.)

Setup della socket (lato server)



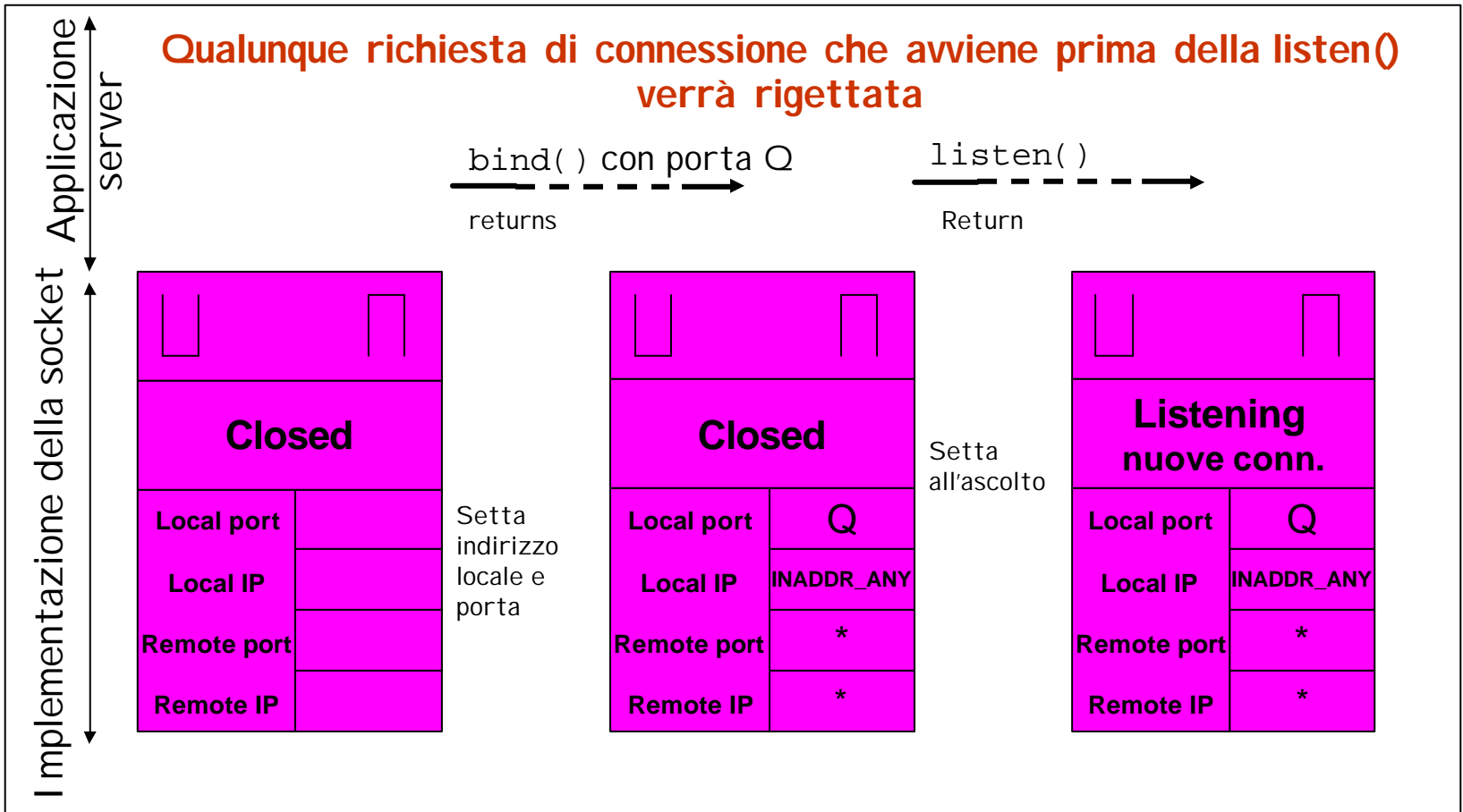
Il ciclo di vita di una socket TCP(cont.)

Setup della socket (lato server)



Il ciclo di vita di una socket TCP(cont.)

Setup della socket (lato server)



Il ciclo di vita di una socket TCP(cont.)

Accettare la connessione (lato server)



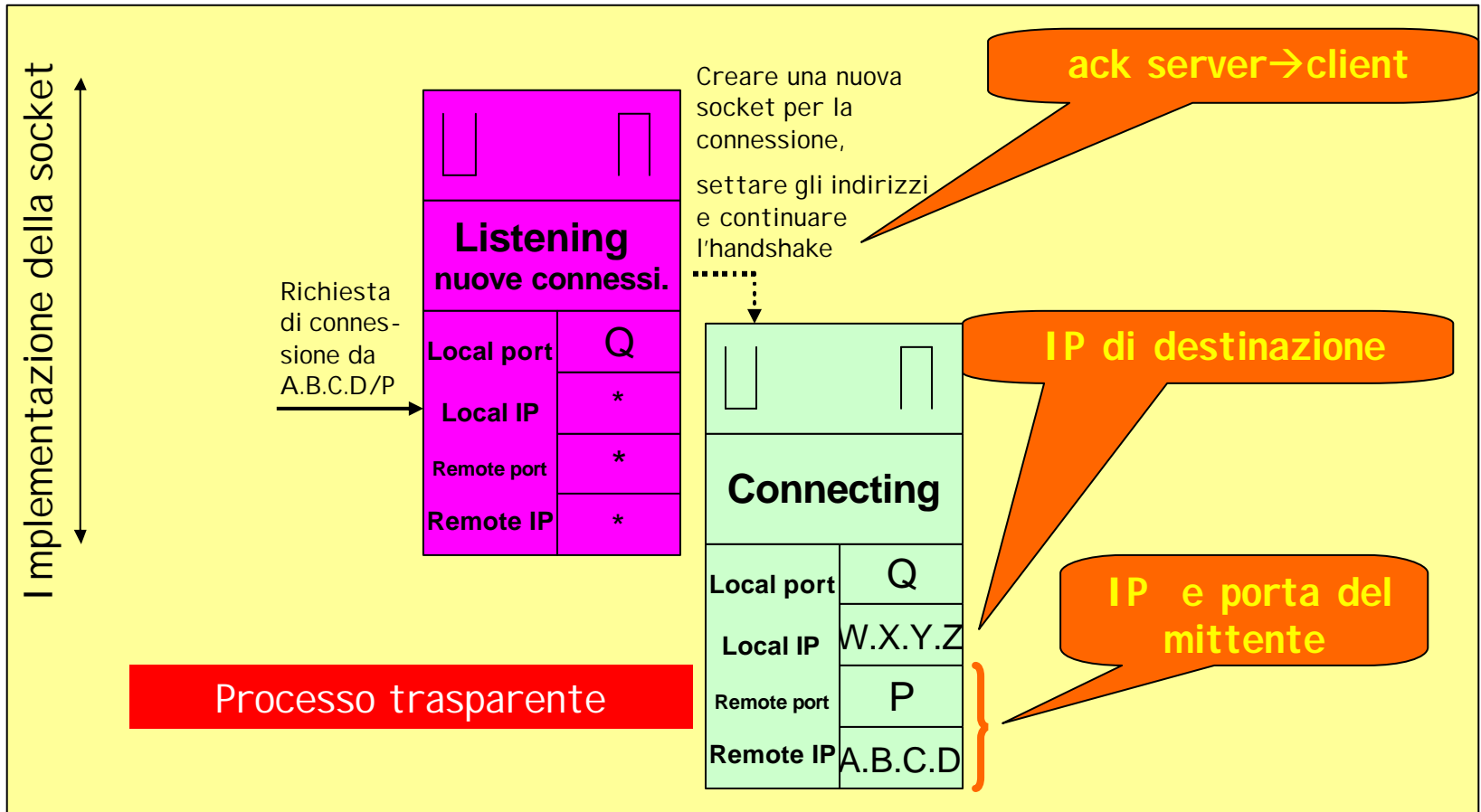
Il ciclo di vita di una socket TCP(cont.)

Gestione della richiesta di connessione in entrata



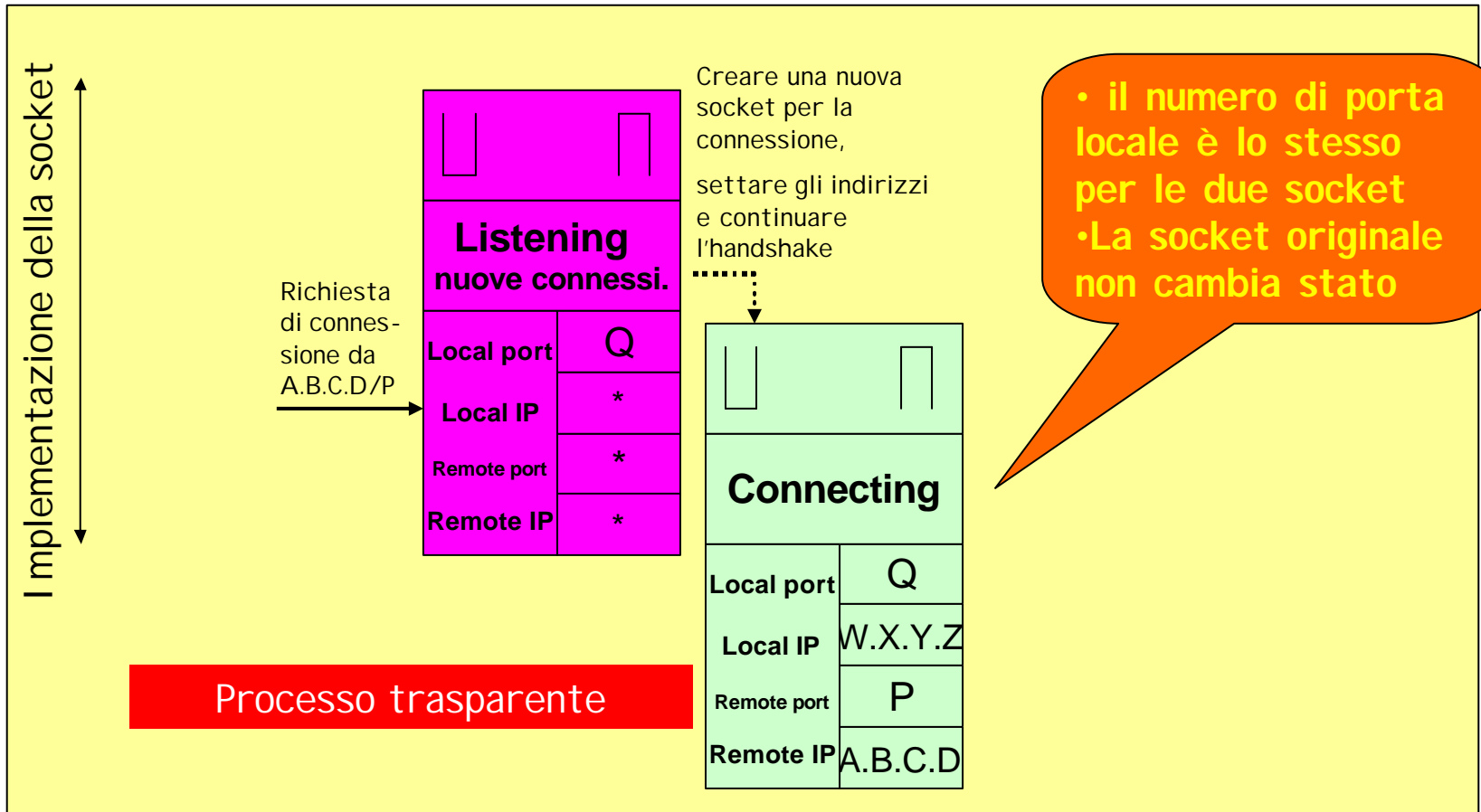
Il ciclo di vita di una socket TCP(cont.)

Gestione della richiesta di connessione in entrata



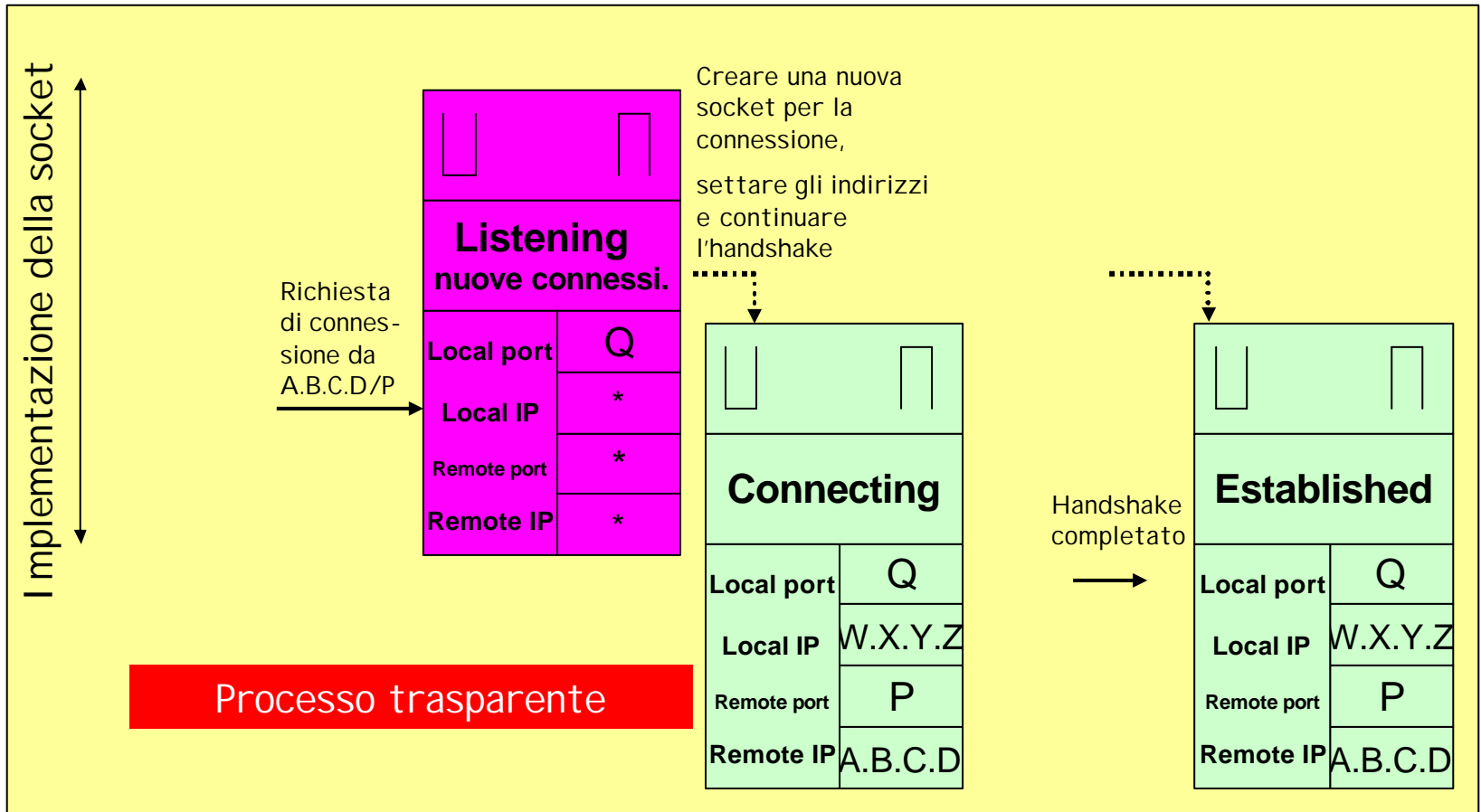
Il ciclo di vita di una socket TCP(cont.)

Gestione della richiesta di connessione in entrata



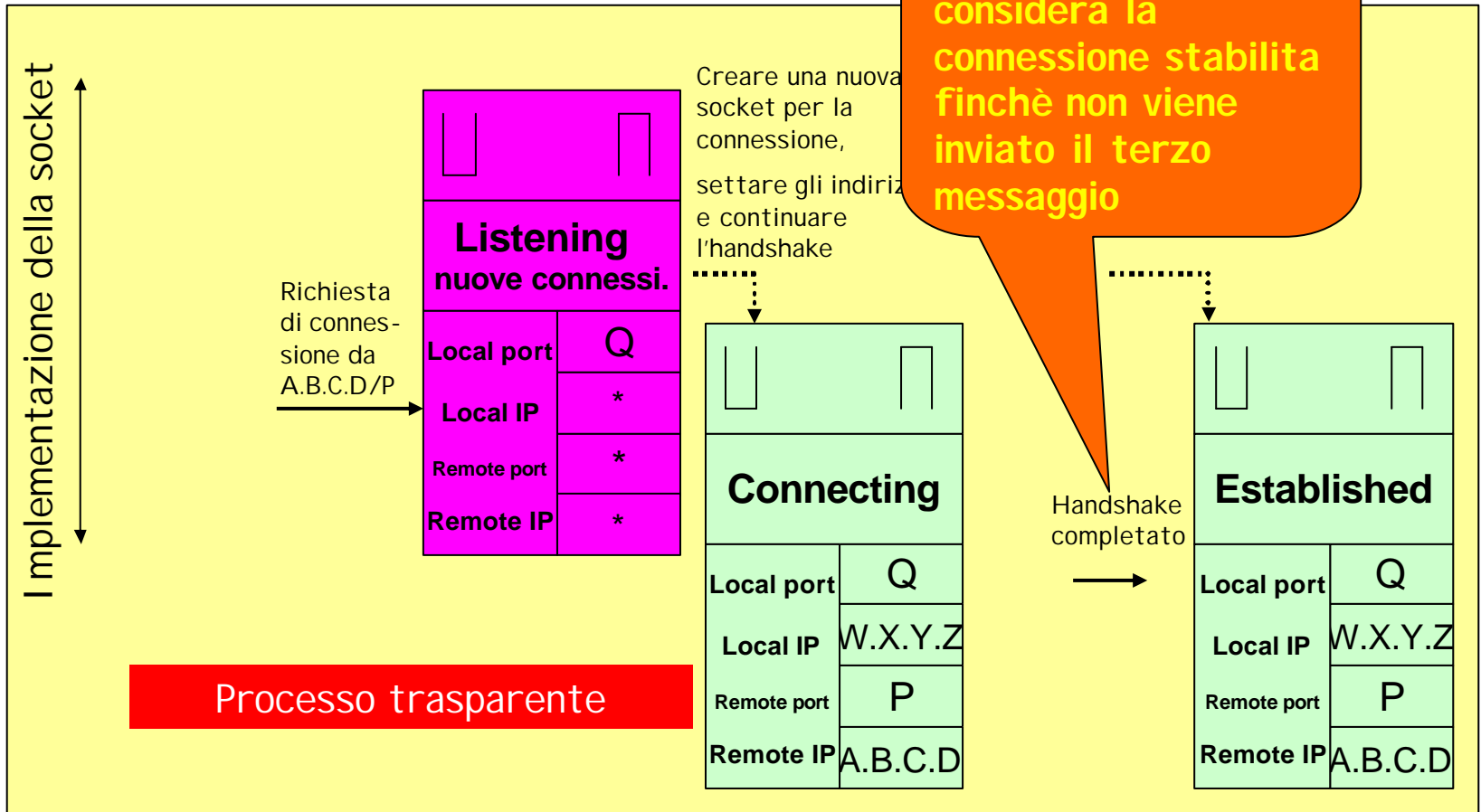
Il ciclo di vita di una socket TCP(cont.)

Gestione della richiesta di connessione in entrata



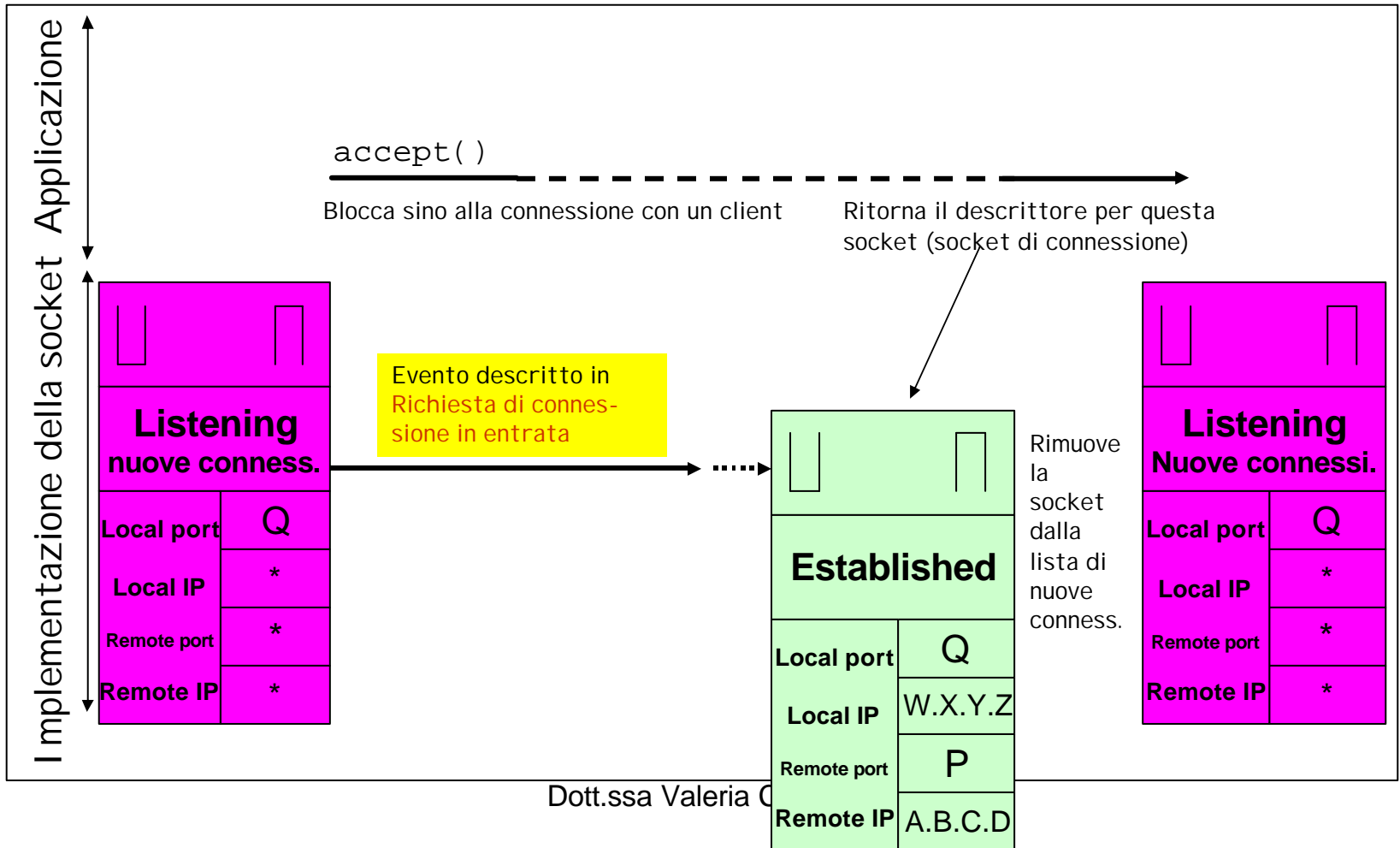
Il ciclo di vita di una socket TCP(cont.)

Gestione della richiesta di connessione



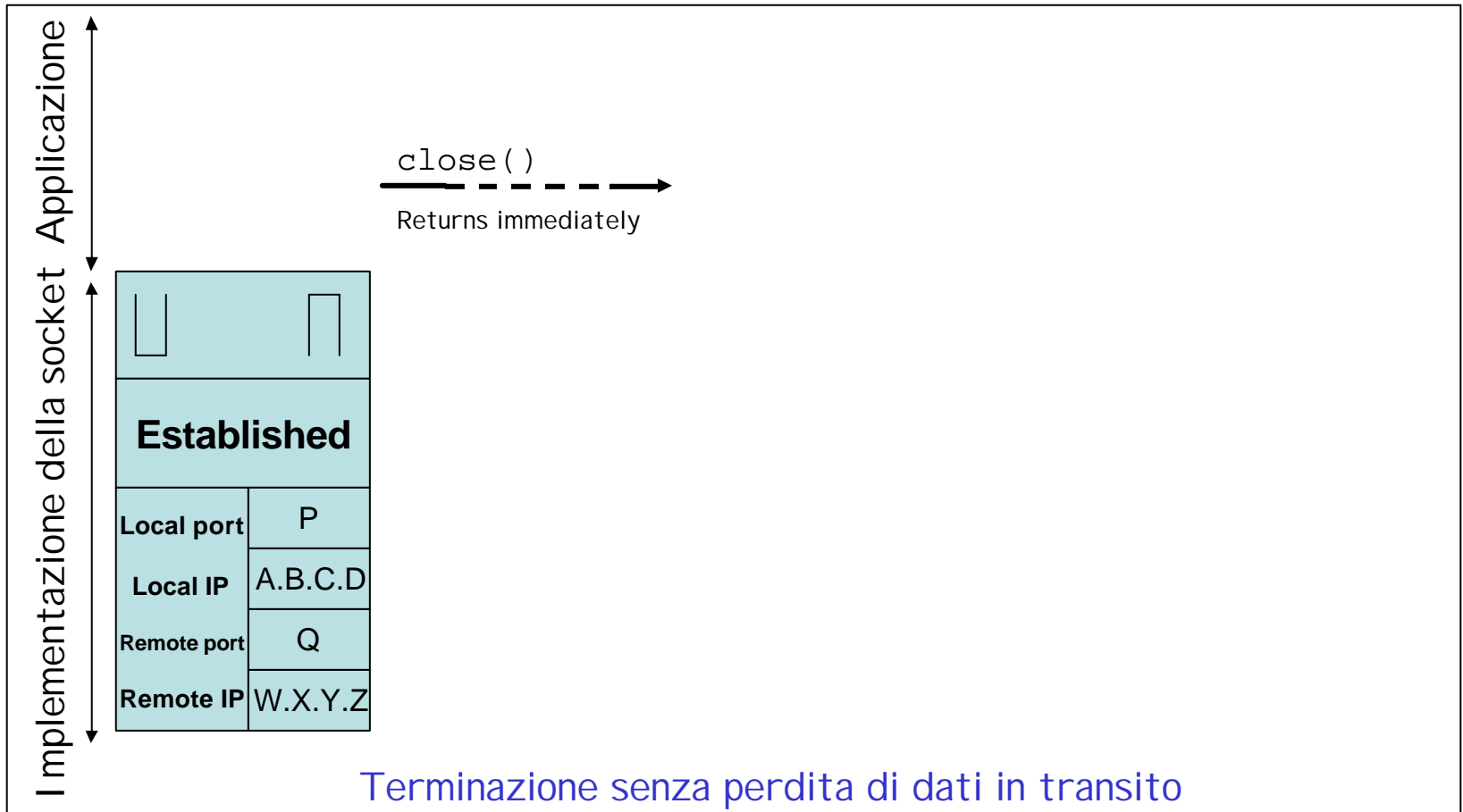
Il ciclo di vita di una socket TCP(cont.)

Accettare la connessione (lato server)



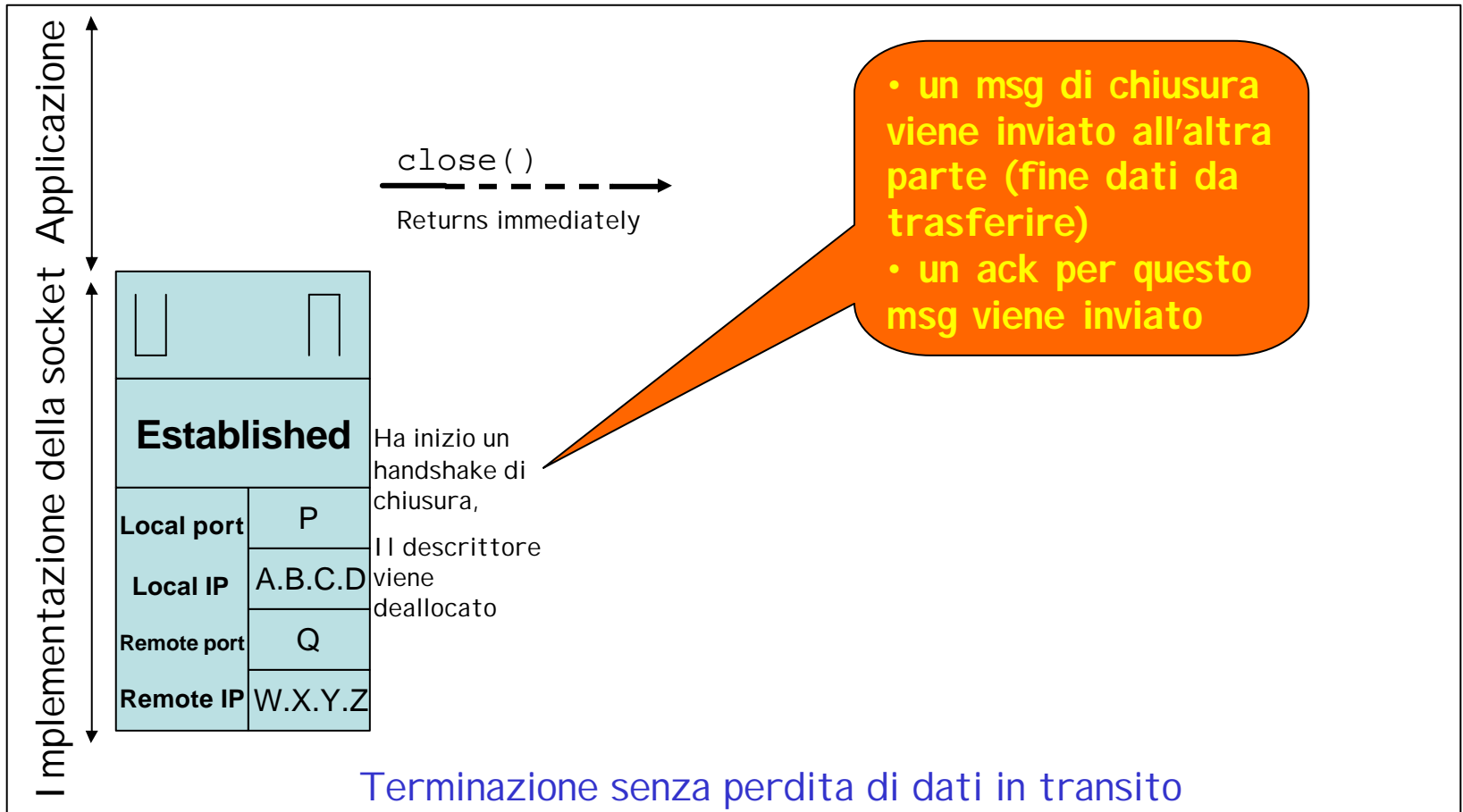
Il ciclo di vita di una socket TCP(cont.)

Chiudere la connessione (indipendente dal lato)



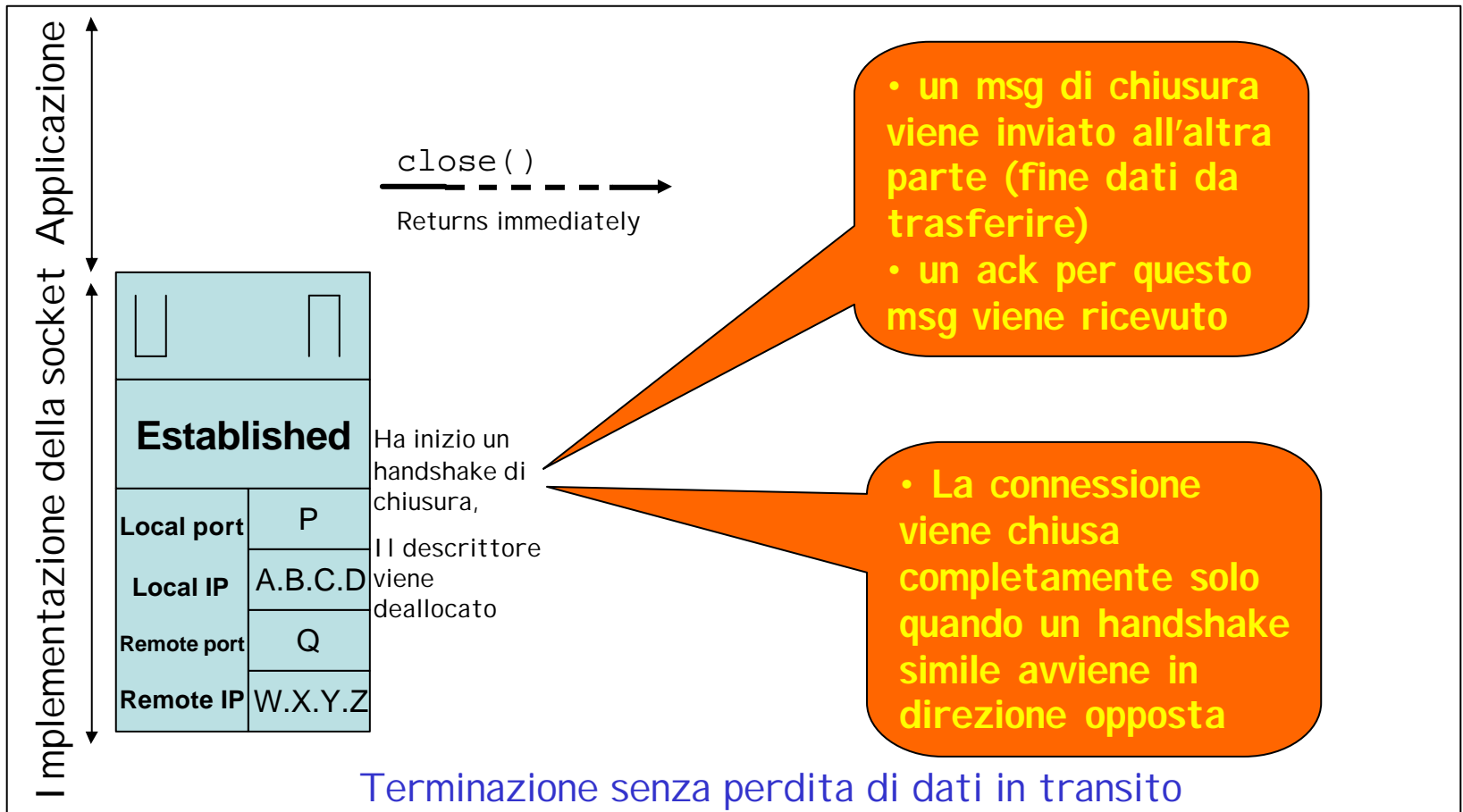
Il ciclo di vita di una socket TCP(cont.)

Chiudere la connessione (independente dal lato)



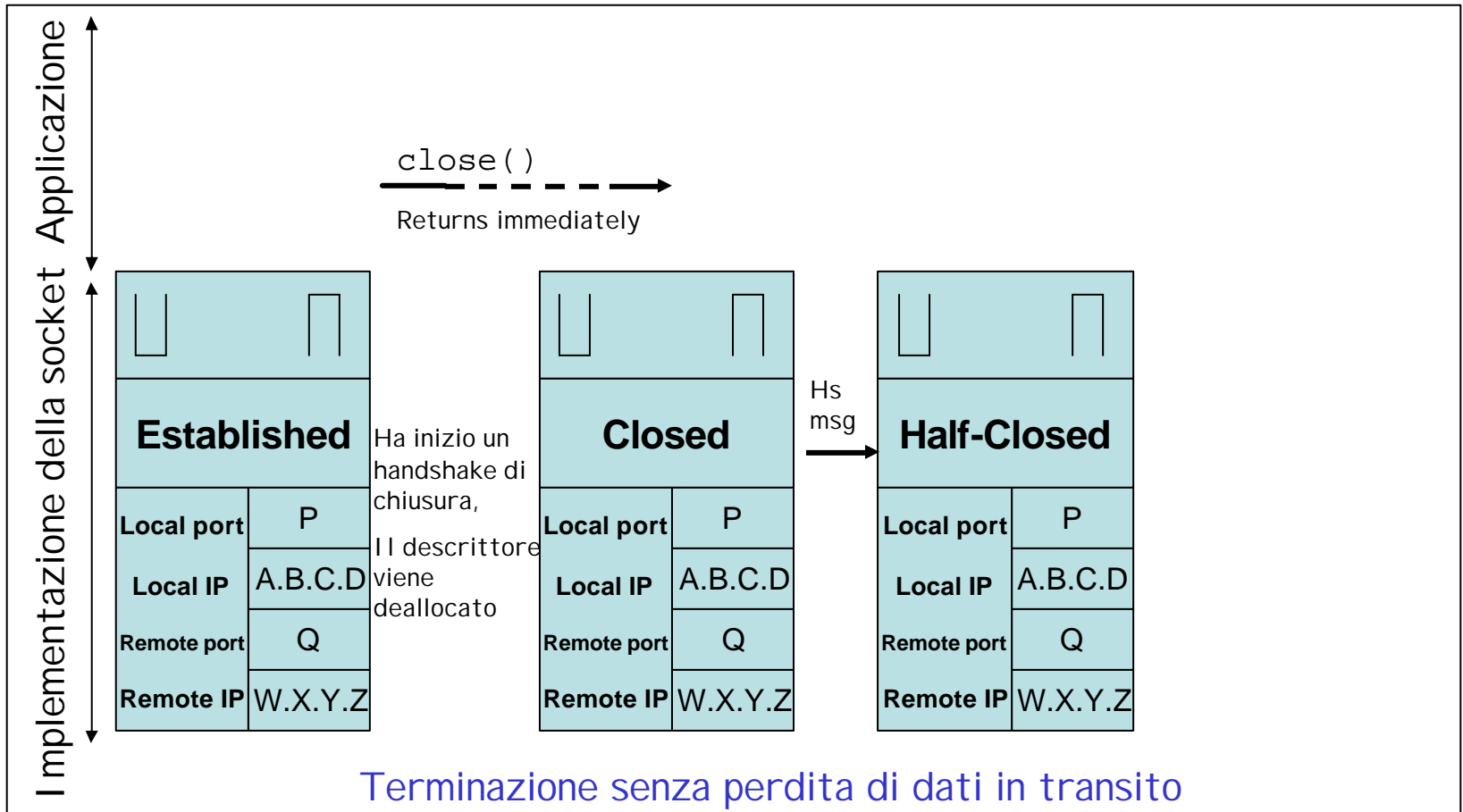
Il ciclo di vita di una socket TCP(cont.)

Chiudere la connessione (independente dal lato)



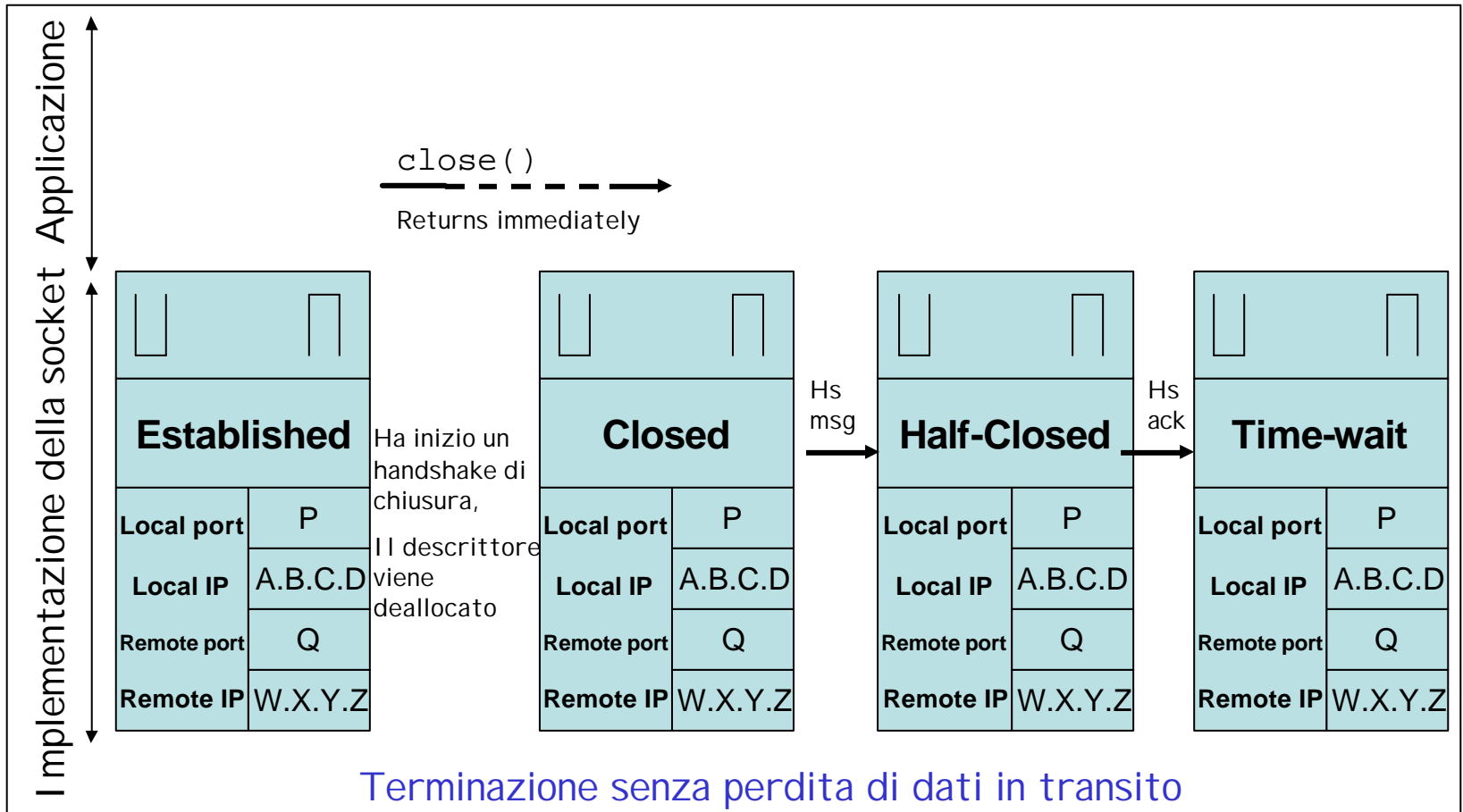
Il ciclo di vita di una socket TCP(cont.)

Chiudere la connessione (indipendente dal lato)

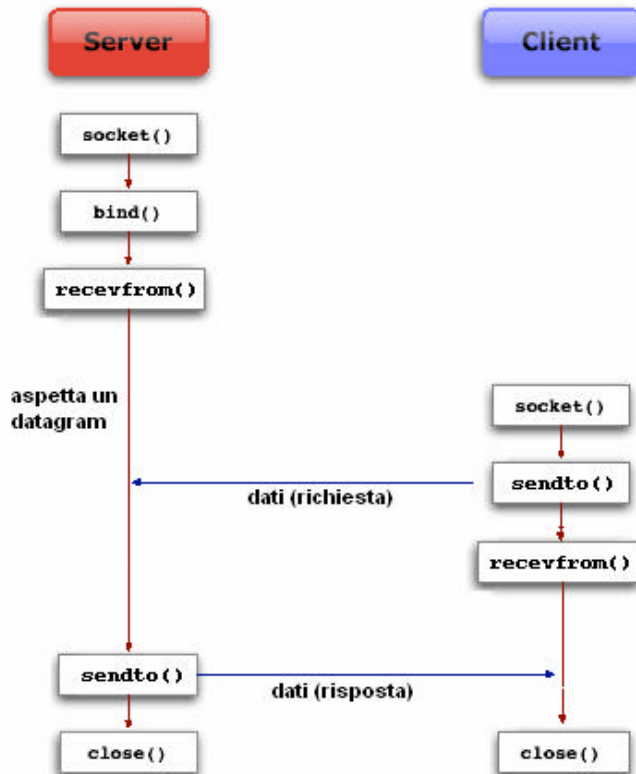


Il ciclo di vita di una socket TCP(cont.)

Chiudere la connessione (independente dal lato)



Interazione UDP Client/Server



Server

1. (Inizializzare una WSA)
2. Creare una socket
3. Assegnare un local address alla socket
4. Iterativamente:
 - a. Inviare e ricevere dati
 - b. Chiudere la connessione

Client

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

Informazioni

- Si presume che il computer in uso sia dotato di un sistema operativo ed un compilatore C in grado di supportare le sockets

- Lab Turing& Shannon: windows XP - Dev C++ (versione v)
- se volete cimentarvi: UNIX (+ compilatore c)

N.B. il nostro corso sarà Windows-Based. Ma mostreremo le differenze nell'uso delle API socket per applicazioni di base UNIX-Based.

- Il materiale didattico e alcuni degli esempi mostrati potranno essere scaricati dalla pagina del corso:

http://www.di.uniba.it/intint/people/vale_file/LabProRete/laboratorio.htm

N.B. Gli esempi scaricabili sono testati e possono essere compilati ed eseguiti in DEV C++ senza alcuna modifica. Tuttavia in base alla posizione dei file header (.h) e delle dipendenze varie (es. librerie) sul vostro sistema, talvolta potrebbero essere necessarie piccole modifiche.