Introduzione alle applicazioni di rete

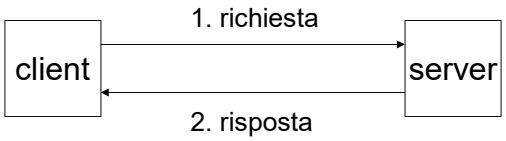
Definizioni base
Modelli client-server e peer-to-peer
Socket API
Scelta del tipo di servizio
Indirizzamento dei processi
Identificazione di un servizio
Concorrenza nei server
Interazione client/server orientata alla connessione e senza connessione
Collaudo di applicazioni di rete

Definizioni base

- Applicazione di rete (distribuita)
  - Fornisce/realizza un servizio per un utente finale
  - Composta da più programmi in esecuzione (processi) su host diversi
  - **Client**: processo che inizia la comunicazione
  - **Server**: processo che aspetta di essere contattato
- Applicazione Internet
  - La comunicazione tra processi è basata sui servizi di rete e trasporto di Internet
  - Può essere **basata su protocolli di livello applicazione**
    - protocolli pubblici o privati
    - **servizi via socket API**
  - Può essere invece basata su invocazione di servizi remoti
    - RPC: chiamata di procedura remota
    - Varianti RPC: Java RMI, .Net Remoting, Web services SOAP
- Applicazione conforme a standard Internet
  - Protocolli pubblici descritti da Request for Comments (RFC) e gestiti dall'Internet Engineering Task Force (IETF)
  - Esempi: FTP (file transfer), SMTP (email), HTTP (web), IRC (chat)

Modello client-server

- Molte applicazioni di rete usano come modello di interazione il modello client-server
- Il server offre un servizio (accesso alle risorse condivise) su richiesta (attesa passiva)
- Il client richiede un servizio iniziando la comunicazione
- Interazione asimmetrica: richiesta-risposta



Caratteristiche di un client e di un server

Client

- Esegue sul computer locale
- E' invocato dall'utente
- Inizia il contatto con il server
- Spedisce una richiesta, opzionalmente con dati
- Puo' accedere a piu' servizi (uno per volta)

Server

- Esegue su un computer remoto
- Parte all'inizializzazione del sistema
- Aspetta le richieste di servizio dai client: loop fino a che non arriva una richiesta
- Spedisce una risposta, opzionalmente con dati
- Accetta richieste da client arbitrari: un servizio per ogni client

Modello peer-to-peer

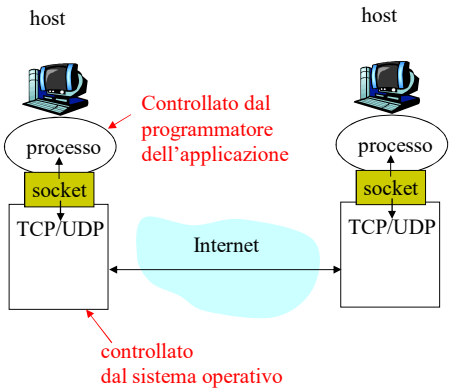
- I componenti dell'applicazione agiscono in modo paritetico
  - interazione simmetrica
    - Possono comportarsi sia da client che da server
- Le informazioni sono suddivise tra i nodi della rete piuttosto che essere concentrate in un singolo nodo
  - architettura decentralizzata
- Ogni pari può stabilire connessioni dirette con altri pari attivi nella rete

Applicazioni P2P

- Il modello P2P in realtà è utilizzato da lungo tempo nei livelli di rete e collegamento dati
  - Ethernet
  - Internet routing
- L'applicazione del modello P2P nella progettazione di applicazioni di rete è più recente
  - File sharing
  - Elaborazione distribuita
  - Comunicazione

Socket API

- API: Application Programming Interface
- Una socket e' un dispositivo di interfaccia tra un processo e il sistema operativo
- Un processo può sia spedire messaggi a un altro processo che ricevere messaggi mediante la propria socket
- La comunicazione via socket usa il modello di I/O di Unix
  - open-read-write-close
  - file descriptor = (pathname, flag)



Scelta del tipo di servizio

TCP - orientato alla connessione

- Il client stabilisce la connessione al server
- Il client e il server si scambiano messaggi multipli di dimensione arbitraria
- Il client termina la connessione

Garantisce l'affidabilità del servizio

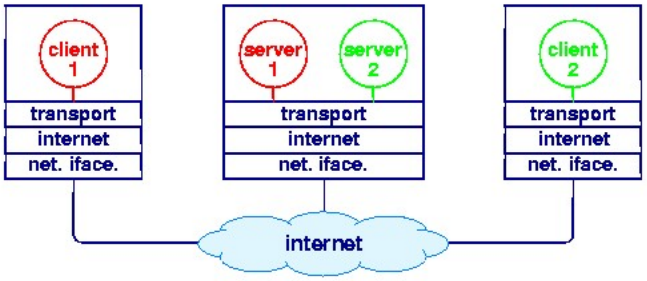
UDP - privo di connessione

- Il client costruisce il messaggio
- Il client spedisce il messaggio al server
- Il messaggio deve entrare in un datagramma UDP
- Il server risponde

Il servizio non è affidabile

Servizi multipli offerti da un host

- Un sistema operativo multi-tasking può ospitare più di un server
  - Ogni server offre un servizio specifico ai client che lo richiedono
  - I server sono processi indipendenti e possono gestire le richieste dei client simultaneamente



Indirizzamento dei processi

- Ogni processo comunicante è associato a un numero di porta a 16-bit, locale e unico rispetto all'host
- Necessita' di individuare un processo con un indirizzo di rete globalmente unico (IP address e port number)
- Tutti messaggi contengono due coppie di indirizzi:
  - Mittente: (IP address, port number)
  - Destinatario: (IP address, port number)
- Il sistema operativo dell'host destinatario consegna al processo un messaggio in arrivo sulla base del port number

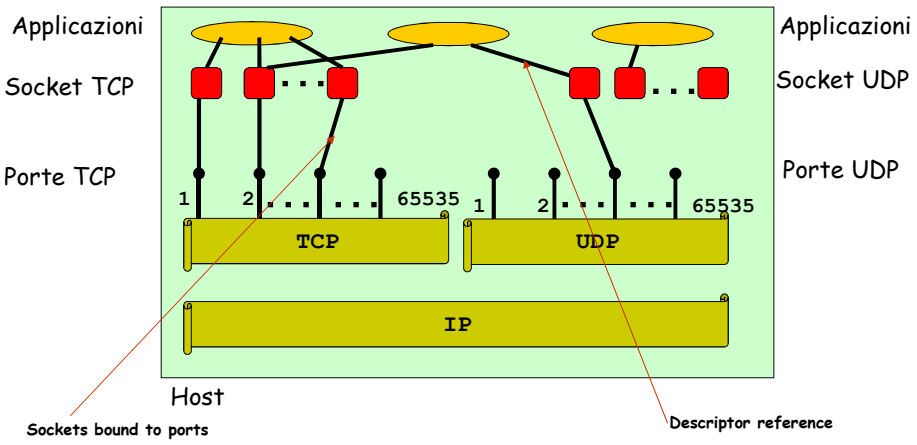
Numeri di porta

- 1 - 1023: porte di sistema (riservate da IANA a servizi noti di protocolli standard IETF)
  - 21: ftp
  - 23: telnet
  - 25: smtp
  - 80: www
- 1024 - 49151: porte utente (o porte registrate da IANA su richiesta)
- 49151 - 65535: porte private o dinamiche (automaticamente assegnate)

Identificazione di un servizio

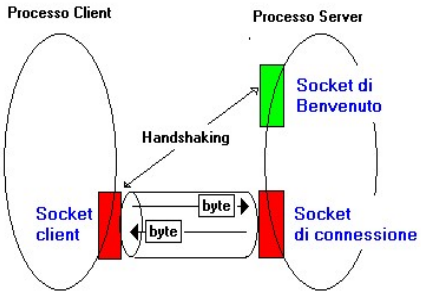
- A ogni servizio offerto da un server è assegnato un identificatore unico per l'host: numero di porta
- Il server si registra con il software del protocollo di trasporto usando il numero di porta (bind)
- Il client contatta il server usando identificatore dell'host (indirizzo IP) + l'identificatore del servizio (numero di porta)
  - Gli identificatori del server devono essere noti a priori
- Anche il client è identificabile tramite indirizzo IP + numero di porta
  - Il client comunica al server il proprio indirizzo completo al momento della richiesta
- Un numero di porta non può essere associato a più processi (server o client) contemporaneamente
- Un processo (server o client) può usare più numeri di porta contemporaneamente
  - Ogni thread di un server concorrente usa numeri di porta distinti

Socket e numeri di porta



Concorrenza nei server

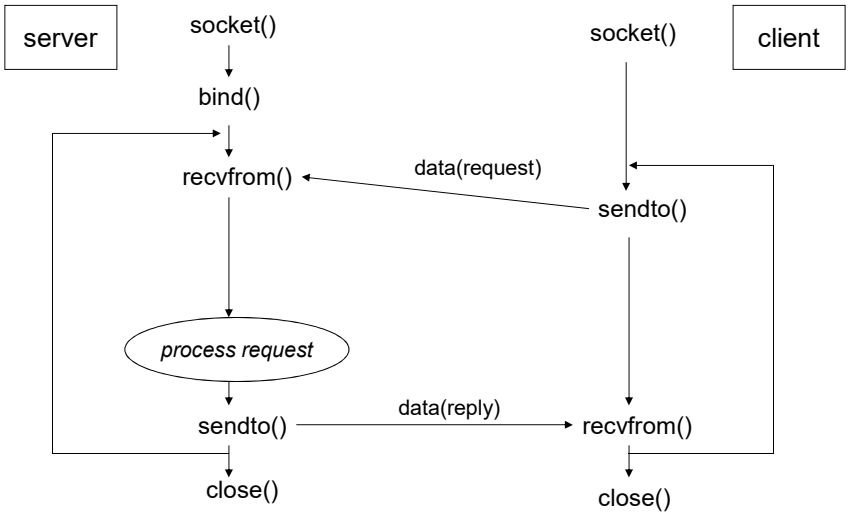
- Problema:
  - La risposta a una richiesta di un client può richiedere molto tempo
  - Altri client devono aspettare il completamento delle richieste precedent
- Soluzione
  - Server concorrente che permette di gestire contemporaneamente le richieste di client multipli indirizzate a una socket di benvenuto
  - Il server crea dinamicamente un nuovo thread (copia del processo) per ogni richiesta accettata e gli assegna una socket di connessione
  - Il thread concorrente (con il suo flusso di controllo autonomo operante su memoria condivisa) soddisfa la richiesta del client attraverso la socket di connessione
  - Il server si rende subito disponibile e attende le successive richieste dirette sempre alla socket di benvenuto



Connessioni e server concorrenti

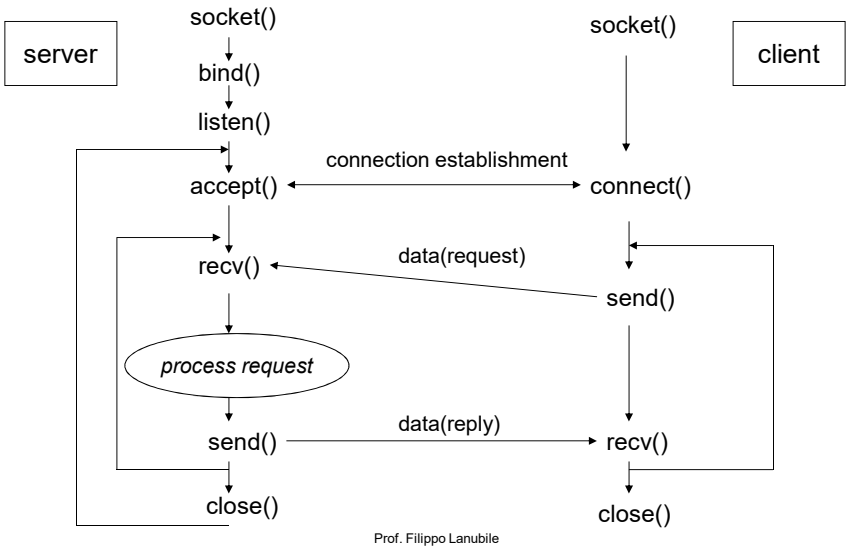
- Problema
  - I server concorrenti creano un thread per ogni richiesta accettata
  - Richieste simultanee sono servite da thread concorrenti
  - Ogni client è identificato da: (ind. IP, n.ro di porta)
  - Ogni servizio (server) è identificato da: (ind. IP, n.ro di porta)
  - Client e server interagiscono scambiandosi più messaggi
  - Come fa un client a far arrivare i messaggi (incoming messages) al thread assegnato?
- Soluzione
  - Utilizzare l'identificazione della sessione di trasporto
    - (ind. IP server, n.ro di porta server, ind. IP client , n.ro di porta client)
  - Il software che implementa il protocollo di trasporto associa ogni thread creato a una quadrupla distinta

Interazione client/server senza connessione



Prof. Filippo Lanubile

Interazione client/server orientata alla connessione



Prof. Filippo Lanubile

Collaudo di applicazioni di rete

- Test del server
  - uso di telnet per contattare il server
    - telnet server_address port_number
  - casi di test da standard input
  - comportamento osservato su standard output
- Test del client con server preesistente
  - installazione del server in locale
  - uso di localhost come indirizzo di rete