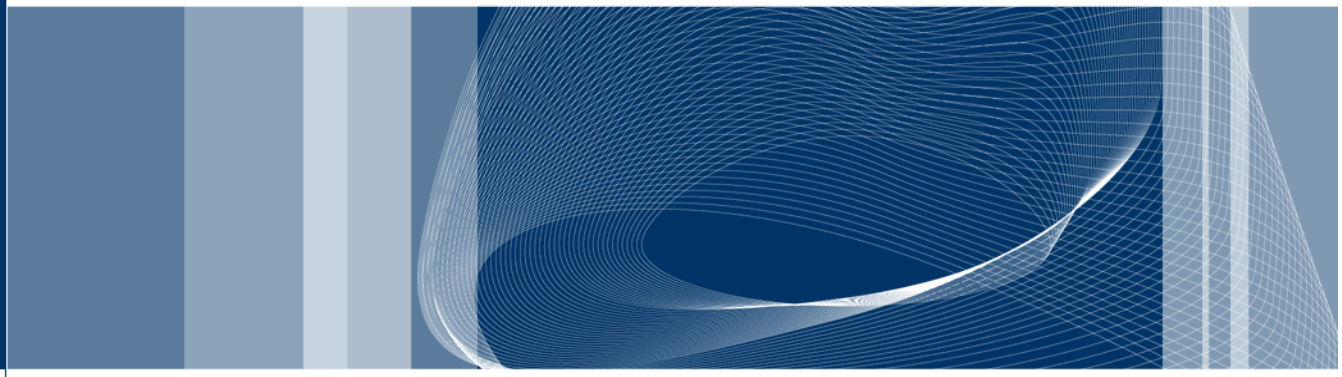
POLITECNICO DI MILANO



Sistemi Distribuiti

Informatica B

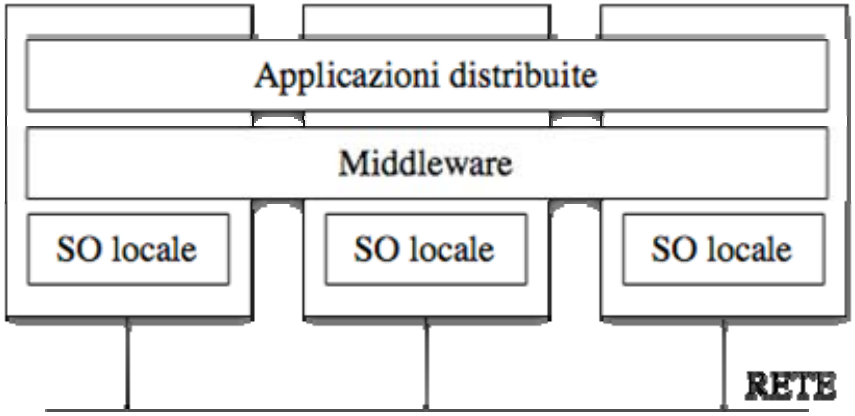
Introduzione

Che cos'è un sistema distribuito?

- Un sistema distribuito è una collezione di computer indipendenti che appare all'utente come un solo sistema coerente
- Da notare:
  - le macchine sono autonome (hardware)
  - l'utente pensa di lavorare su una sola macchina (software)

Organizzazione di un sistema distribuito

- Obiettivo: offrire una visione unica del sistema che in realtà è composto da computer e reti eterogenei
- Soluzione: organizzazione a strati (layer)
  - Livello superiore: utenti e applicazioni
  - Livello intermedio: strato software
  - Livello basso: sistema operativo
- Il livello intermedio è spesso chiamato **middleware**



Caratteristiche

- Quali sono le caratteristiche principali che un sistema distribuito deve avere?
  - consentire facilmente la connessione tra utenti e risorse
  - essere trasparente, cioè nascondere che le risorse sono distribuite
  - essere aperto
  - essere flessibile
  - essere scalabile

Facilità di connessione

- Un sistema distribuito deve rendere semplice l'accesso a risorse remote
  - Esempi: stampanti condivise, dati, file
- Perchè?
  - Motivi economici
  - La connessione rende più semplice la collaborazione e lo scambio di informazione
- Ma facilità di connessione e condivisione fanno insorgere anche problemi di **sicurezza**:
  - Informazioni sensibili
  - Tracciabilità delle comunicazione

Trasparenza

- Un sistema distribuito deve nascondere che i suoi processi e risorse sono fisicamente distribuite
  - **trasparenza di accesso**: nascondere le differenze di rappresentazione dei dati e del modo in cui gli utenti accedono alle risorse
  - **trasparenza di locazione**: nascondere la locazione fisica di una risorsa
  - **trasparenza di migrazione**: permettere il continuo accesso a risorse che possono essere spostate (**trasparenza di rilocazione** se avviene mentre una risorsa è in uso)
  - **trasparenza di duplicazione**: nascondere la duplicazione di delle risorse (es. per migliorare le prestazioni)
  - **trasparenza di concorrenza**: nascondere agli utenti che competono per le medesime risorse
  - **trasparenza ai fallimenti**: nascondere all'utente eventuali guasti di risorse
  - **trasparenza alla persistenza**: nascondere il tipo di memoria su cui si trova la risorsa (es. volatile o fissa)

Sistemi aperti

- Un sistema distribuito si dice aperto se offre servizi secondo regole standard per descrivere la sintassi e la semantica del servizio
- Le regole sono di solito specificate attraverso **interfacce**, che specificano i nomi delle funzioni disponibili e la loro intestazione
- Un'interfaccia dovrebbe essere:
  - **Completa**: specifica tutto quello che è necessario per implementarla
  - **Neutra**: non dà informazioni su come deve apparire un'implementazione
- Queste caratteristiche influenzano:
  - **Interoperabilità**: due implementazioni costruite diversamente possono coesistere e lavorare insieme
  - **Portabilità**: un'applicazione sviluppata per un sistema distribuito A può essere eseguita su un sistema B

Flessibilità

- Un sistema distribuito deve essere flessibile, cioè deve rendere semplice:
  - la configurazione del sistema
  - l'aggiunta di nuove componenti
- Quindi un sistema distribuito aperto deve essere estensibile

Scalabilità

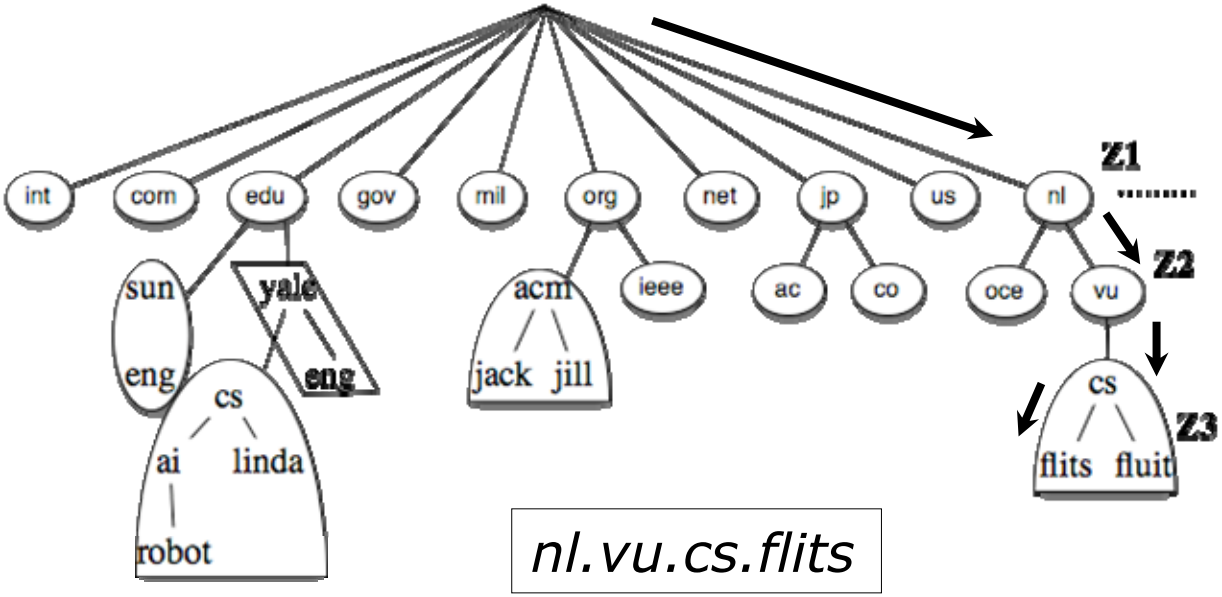
- La scalabilità di un sistema si può misurare secondo 3 dimensioni:
  - rispetto alla dimensione: posso aggiungere utenti e risorse
  - geograficamente: utenti e risorse possono essere fisicamente molto distanti
  - dal punto di vista amministrativo: facile da gestire anche in presenza di organizzazioni amministrative indipendenti
- Problematiche:
  - Servizi e dati centralizzati (una sola componente che offre un servizio è il collo di bottiglia del sistema)
  - Algoritmi centralizzati
  - Sistemi progettati per reti locali si basano spesso su comunicazione sincronizzata che non scala geograficamente
  - La comunicazione in reti estese è intrinsecamente inaffidabile

Scalabilità (2)

- Per evitare latenza in reti estese cercare il più possibile di avere una comunicazione tra componenti asincrona
  - Non sempre è possibile (applicazioni interattive)
- Replicazione: duplicare componenti nel sistema distribuito
  - Il caching è un caso speciale di replicazione
  - La replicazione dà problemi di consistenza
- Distribuzione: dividere le componenti in parti più piccole e distribuirle

Esempio: Domain Name System (DNS)

- Il DNS è un esempio di distribuzione
  - Organizza lo spazio gerarchicamente attraverso un albero di domini:
  - Ogni dominio è diviso in zone non sovrapposte
  - Dato un nome composto da parti diverse (che rappresentano le diverse zone) si risolve navigando l'albero



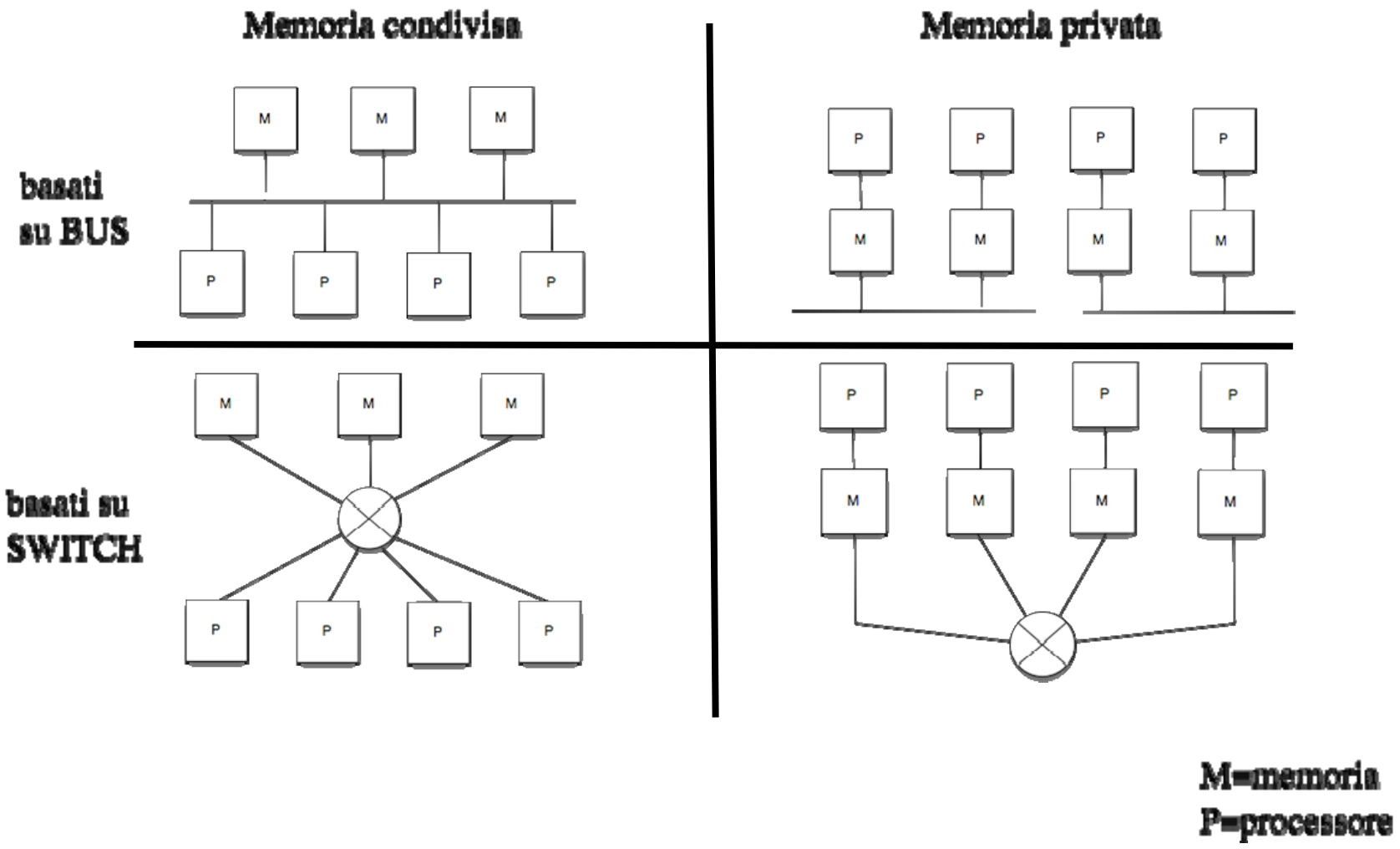
nl.vu.cs.flits

Architettura HW

Hardware

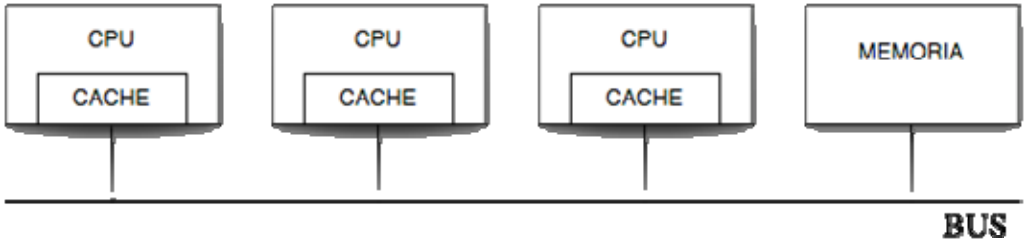
- Un sistema distribuito è formato da più CPU, ma queste possono essere organizzate in modo diverso:
- Possiamo distinguere tra:
  - **Multiprocessori**: un unico spazio di indirizzi fisici è condiviso da tutte le CPU (memoria condivisa)
  - **Multicomputer**: ogni macchina ha il suo indirizzo fisico (memoria privata)
- All'interno un'ulteriore distinzione è data dalla rete di interconnessione:
  - **basata su BUS**: c'è un'unica rete
  - **basata su SWITCH**: cavi individuali da macchina a macchina

Organizzazione



Multiprocessori (1)

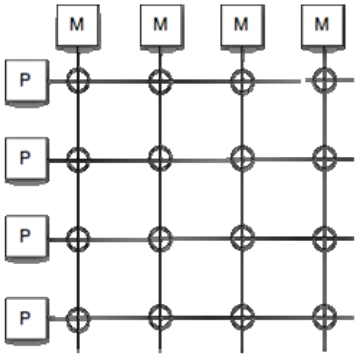
- Nei sistemi multiprocessori tutte le CPU condividono la stessa memoria
- Nei multiprocessori basati su bus, le CPU e il modulo di memoria sono direttamente connessi attraverso un bus
  - La memoria deve essere coerente (se A scrive una zona di memoria e B la legge poco dopo, B deve poter leggere ciò che ha scritto A)
  - Per ottenere coerenza spesso si aggiunge una memoria cache



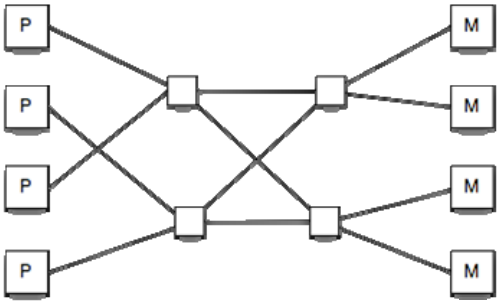
  - I multiprocessori basati su bus scalano poco

Multiprocessori (2)

- Per aumentare la scalabilità si può dividere la memoria in moduli e connetterli con uno switch multiingresso/multiuscita (crossbar switch)



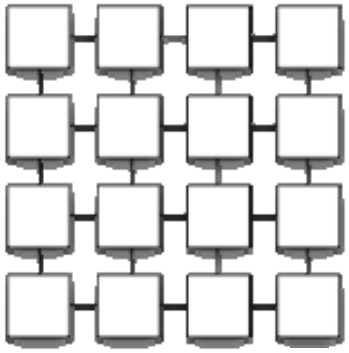
  - ci sono n^2 punti di switch (per n grande non va bene!)
- reti che richiedono meno switch: reti omega

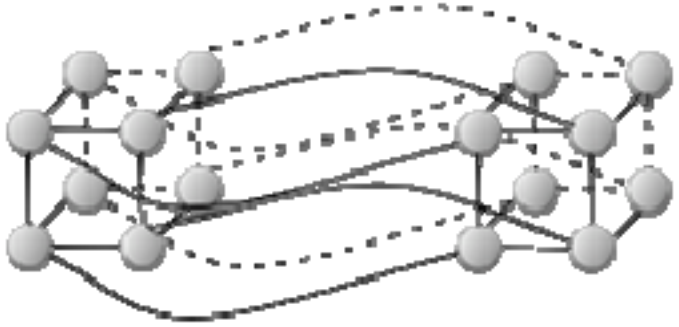


  - si attraversano più punti di switch per andare da P a M

Sistemi multicomputer omogenei

- Nei sistemi multicomputer ogni CPU ha accesso alla sua memoria locale
- Il problema diventa come far comunicare le CPU
- In sistemi basati su bus, i processori sono direttamente connessi a una rete multiaccesso condivisa
  - Problemi di scalabilità
- In sistemi basati su switch, i messaggi tra processori sono indirizzati attraverso una rete di interconnessioni
  - Esempi:





Sistemi multicomputer eterogenei

- Molti sistemi distribuiti sono composti da multicomputer eterogenei
  - Diverse componenti del sistema possono avere caratteristiche molto diverse
- Anche la rete di interconnessione può essere eterogenea
- Tali sistemi sono solitamente di grandi dimensioni, intrinsecamente eterogenei e non danno una visione globale, quindi richiedono software sofisticato

Software

Software

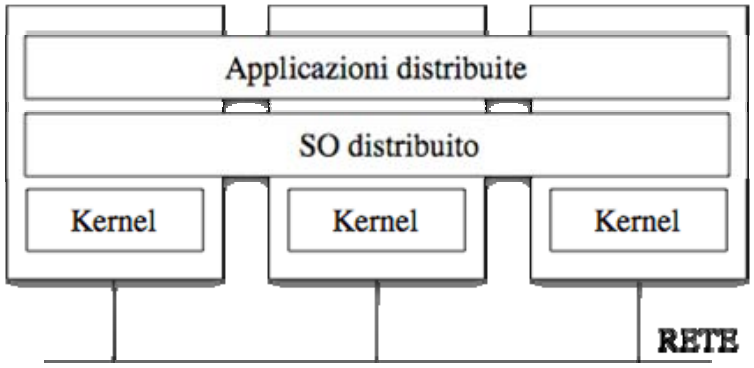
- I sistemi distribuiti hanno molto in comune con i sistemi operativi tradizionali:
  - Si comportano come manager delle risorse: permettono a più utenti e applicazioni di condividere risorse
  - Cercano di nascondere la natura eterogenea del sistema attraverso l'uso di una macchina virtuale
- Sistemi operativi per sistemi distribuiti:
  - Sistemi fortemente accoppiati (**SO distribuiti**): gestiscono multiprocessori e multicomputer omogenei
  - Sistemi debolmente accoppiati (**SO di rete**): sono usati per multicomputer eterogenei
- Per meglio supportare la trasparenza viene introdotto il **middleware**

SO distribuiti multiprocessore

- SO multiprocessore: offrono il supporto necessario per l'accesso alla memoria condivisa:
- Concettualmente: tutte le strutture dati necessarie vengono allocate nella memoria condivisa
- Praticamente: a questi dati si accede con più processori
- Primitive per l'accesso concorrente
  - semaforo: un booleano viene usato per dare accesso alle risorse e il cambio di valore del booleano avviene atomicamente
  - monitor: può essere visto come un modulo con variabili e procedure a cui si può accedere solo mediante le sue procedure e permette l'accesso a un solo processo alla volta

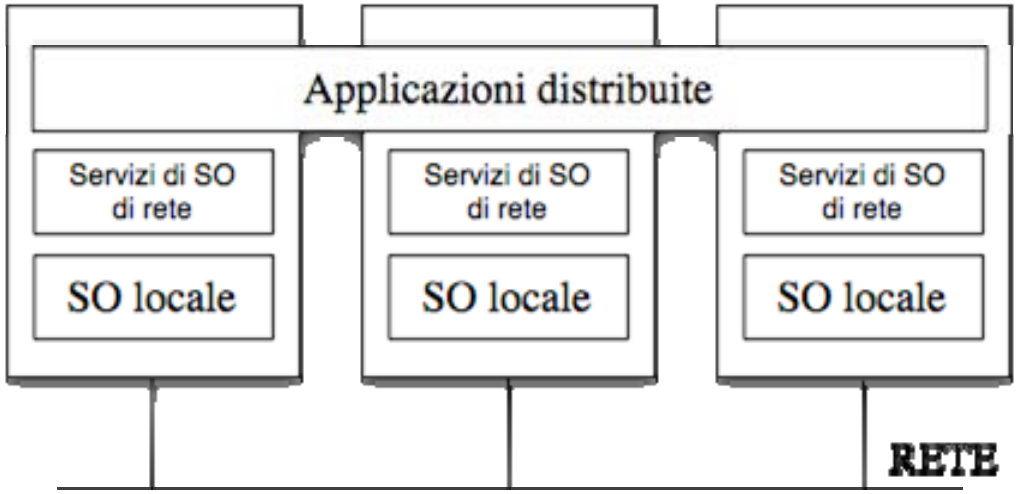
SO distribuiti multicomputer

- Non c'è una memoria fisicamente condivisa dove allocare l'informazione condivisa
- La comunicazione avviene attraverso il passaggio di messaggi
- La semantica del passaggio di messaggi può variare a seconda dei sistemi
- Ogni componente ha il suo kernel
- In alternativa si usa la memoria virtuale di ogni componente per creare una memoria virtuale condivisa (DSM)



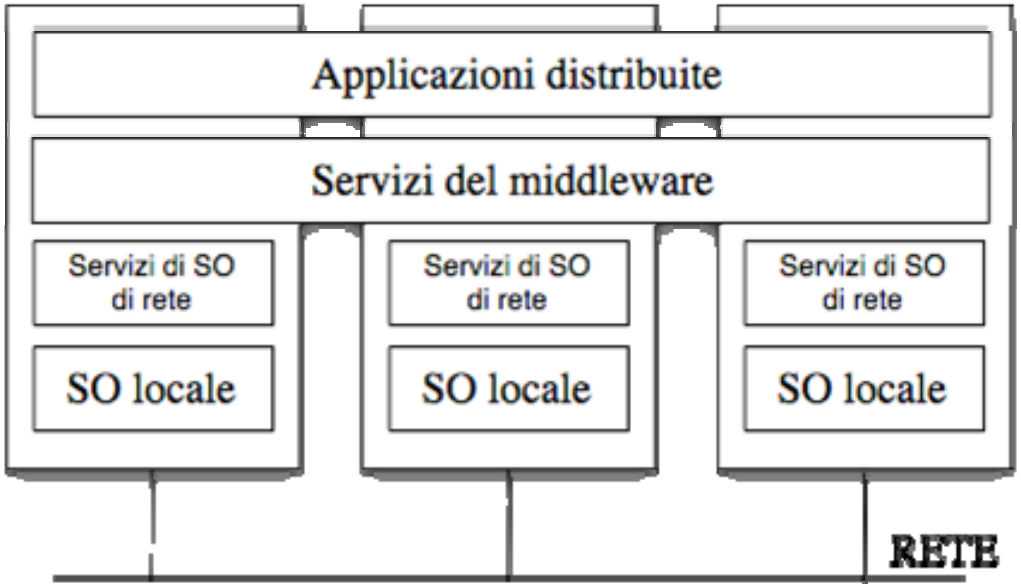
SO di rete

- L'hardware sottostante è eterogeneo, quindi non può essere gestito come un singolo sistema
- Il sistema operativo è locale a ogni componente



Middleware

- I SO distribuiti e di rete non sono davvero sistemi distribuiti rispetto alla nostra definizione
  - i primi non garantiscono una collezione indipendente di computer
  - i secondi non danno una visione coerente
- L'aggiunta di uno strato di software dà queste garanzie



Servizi del middleware

- **Servizi di comunicazione**: nascondono il passaggio di messaggi a basso livello
- **Naming**: permette alle componenti di essere condivise e consultabili come in un elenco telefonico
- **Salvataggi "persistenti"**: middleware avanzati sono integrati con database
- **Transazioni distribuite**: garantisce l'atomicità delle operazioni
- **Sicurezza**

Il modello client/server

- Un sistema distribuito può essere pensato come "clienti (client) che richiedono servizi da servitori (server)
- Secondo il modello client/server, in un sistema distribuito ci sono solo due tipi di processi: client e server:
  - Un server è un processo che implementa un servizio
  - Un client è un processo che richiede un servizio da un server inviando una richiesta
  - La comunicazione client/server è nota come richiesta/risposta

Comunicazione

Comunicazione

- Tutte le comunicazioni in un sistema distribuito multicomputer sono basate sullo scambio di messaggi (a basso livello)
  - non c'è memoria condivisa
- Se A comunica con B
  - A costruisce il messaggio nel suo spazio di indirizzamento
  - A esegue una chiamata di sistema affinché il sistema operativo mandi il messaggio sulla rete a B
  - A e B devono accordarsi sul significato della composizione del messaggio (e dei bit)

Modello ISO-OSI

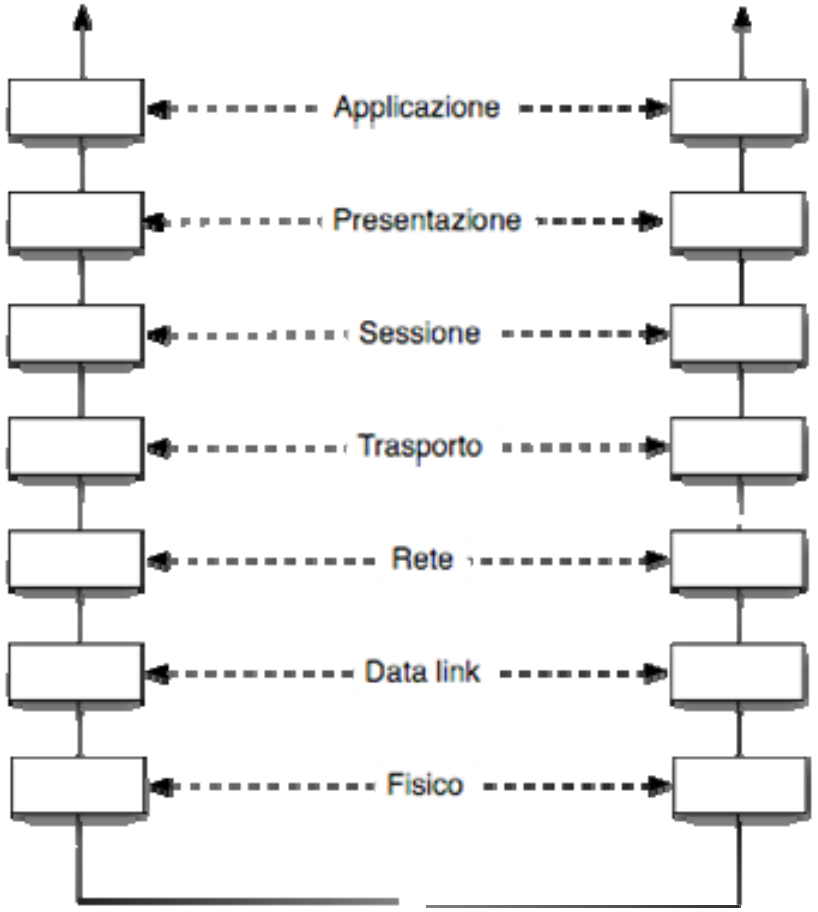
- La International Standard Organization (ISO) ha sviluppato un modello di riferimento che identifica i vari livelli coinvolti nella comunicazione.
- Il modello si chiama Open System Interconnection Reference Model (1983)
- Non è stato molto usato in pratica, MA aiuta a capire il funzionamento di reti di computer
- Il modello OSI permette a sistemi aperti di comunicare

Protocollo di comunicazione

- L'insieme di regole che specificano il formato, il contenuto e il significato dei messaggi mandati e ricevuti si dice **protocollo**
  - Un gruppo di computer per comunicare deve utilizzare un protocollo
- Ci sono due tipi di protocolli:
  - **Orientati alla connessione**: prima di scambiare i messaggi mittente e destinatario stabiliscono esplicitamente una connessione. La rilasciano alla fine dello scambio.
    - Esempio: il telefono
  - **Connectionless** (senza connessione): non è necessario stabilire la connessione a priori
    - Esempio: un lettera in casella

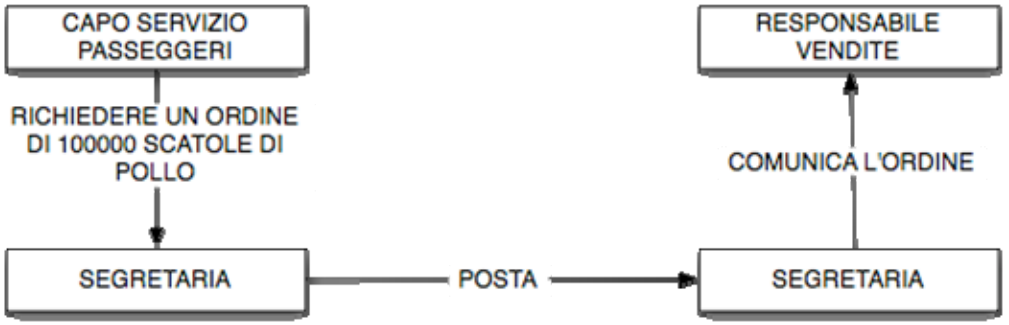
Protocollo a strati

- Nel modello ISO-OSI la comunicazione è divisa in 7 livelli o strati (layer)
- Ogni livello gestisce un livello specifico della comunicazione:
  - il problema può essere diviso in pezzi e ciascuno può essere risolto indipendentemente
- I livelli offrono un'interfaccia al livello sovrastante
  - un insieme di operazioni che insieme definiscono la funzionalità del livello



Esempio: perchè la comunicazione a strati è importante?

- Azienda aerea A e Azienda di catering B
- Ogni mese:

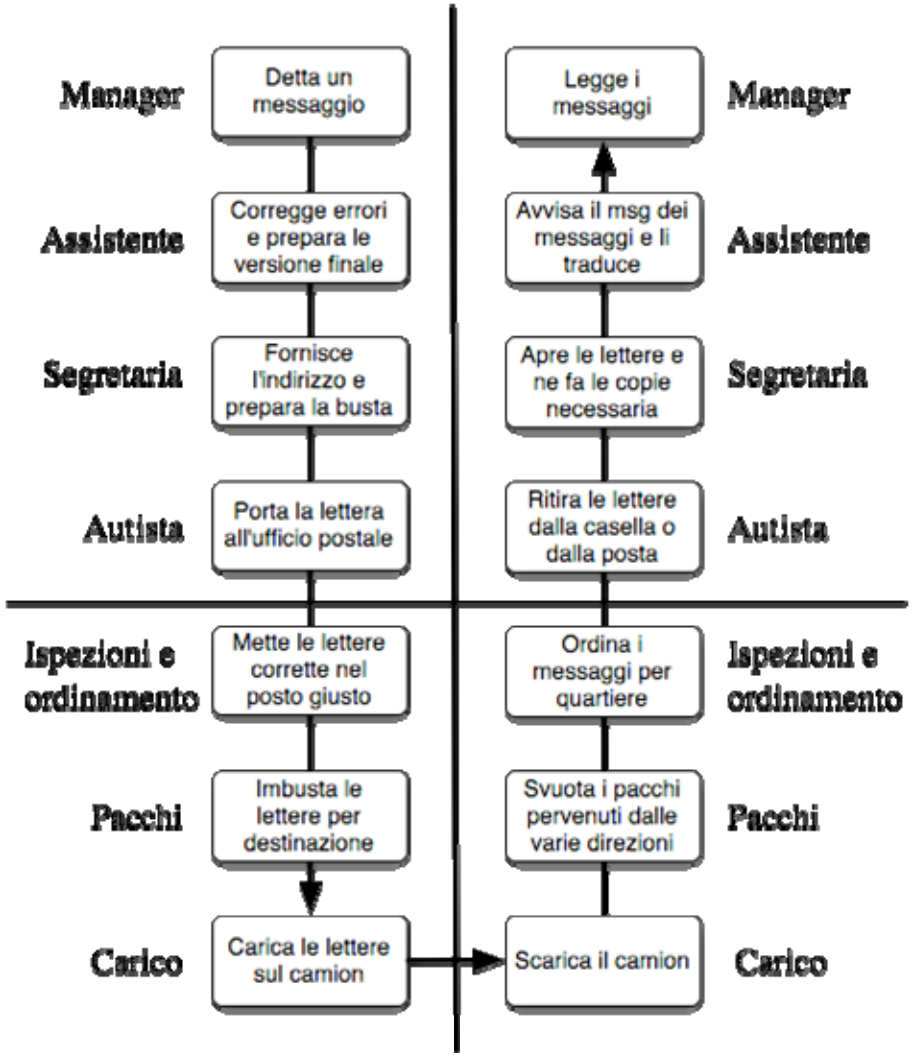


- Il servizio postale peggiora e le segretarie passano al fax
- Il cambiamento non viene percepitto dai capi

Più in dettaglio

Azienda

Ufficio Postale



… nelle reti …

- Quando il processo A sulla macchina 1 vuole comunicare col processo B sulla macchina 2, A costruisce un messaggio e passa il messaggio al livello di applicazione sulla sua macchina
- Il livello di applicazione aggiunge un'intestazione al messaggio (6) e passa il messaggio al livello di presentazione
- Il livello di presentazione aggiunge la sua intestazione (5) e lo passa al livello sottostante e così via
- Quando raggiunge il fondo, il livello fisico lo trasmette

- Quando il messaggio arriva alla macchina 2 (a livello fisico), viene passato verso l'alto e ogni livello gestisce la sua intestazione

Il protocollo ISO-OSI

- Il protocollo ISO-OSI ha 7 strati: e ciascuno ha le sue regole
- Il protocollo si dice anche **pila di protocolli**:
  - Protocolli di basso livello (fisico, data link, rete)
  - Protocolli di trasporto
  - Protocolli di alto livello (sessione, presentazione, applicazione): in pratica c'è solo il livello di applicazione

Protocolli di basso livello

- **Livello fisico:** si occupa della gestione fisica (meccanica ed elettrica) dell'interfaccia con il mezzo fisico usato per il collegamento
  - Ha il compito di trasmettere zeri e uno
  - Fissa la tensione per lo 0 e l'1, il numero di bit al secondo e se la comunicazione può essere bidirezionale simultaneamente
  - Fissa la forma e la dimensione dei connettori di rete
- **Livello data link**: ha il compito di gestire eventuali errori di comunicazione avvenuti a livello fisico
  - Raggruppa i bit in gruppi (frame)
  - Controlla se i frame sono corretti
  - Il mittente aggiunge un bit che rappresenta la somma dei bit nel frame (checksum)
  - Il ricevente risomma i bit e controlla la somma con la checksum
- **Livello di rete**: calcola il miglior cammino per inviare il messaggio (routing)
  - Il miglior cammino non è sempre il più corto

Protocolli di basso livello: il livello di rete

- Il livello di rete si occupa dell'indirizzamento dei messaggi lungo la rete, implementando gli opportuni meccanismi di commutazione
- Il servizio fornito è a livello funzionale ed è quindi indipendente dal particolare tipo di rete adottata
- Il protocollo più usato è il protocollo **IP** (Internet Protocol)
  - Caratteristiche:
    - protocollo connectionless
    - si occupa dell'instradamento e della rilevazione d'errore (nessuna correzione)
  - Non assicura:
    - la consegna,
    - l'integrità,
    - la non-duplicazione
    - l'ordine di consegna

Protocollo di trasporto

- Il livello di trasporto ha il compito di fornire una connessione affidabile
- Il livello di trasporto riceve il messaggio dal livello applicativo e lo spezza in pacchetti piccoli a sufficienza, assegna a ciascuno un numero in una sequenza e li invia tutti
- Si occupa di verificare:
  - quali pacchetti sono stati realmente inviati/ricevuti
  - quanti messaggi il ricevente può ancora ricevere
  - quali pacchetti devono essere ritrasmessi...
- Il protocollo di trasporto internet è TCP (transmittion Control Protocol)
- È supportato anche il protocollo UDP (Universal Datagram Protocol)

Il protocollo TCP

- Caratteristiche:
  - protocollo connection-oriented (indirizzo IP - porta TCP)
  - fornisce un servizio full-duplex, con acknowledge e correzione d'errore
- Due host connessi su Internet possono scambiarsi messaggi attraverso socket TCP
- TCP costituisce l'infrastruttura di comunicazione della maggior parte dei sistemi client-server su Internet

Il protocollo UDP

- Caratteristiche:
  - protocollo connectionless (indirizzo IP - porta UDP)
  - fornisce un servizio di rilevazione d'errore.
  - non assicura la consegna nè, tantomeno, l'ordine di invio (unreliable, best-effort protocol)
- Utilizzato nelle applicazioni client-server di tipo richiesta/risposta
  - Esempi: DNS

Protocolli di alto livello: protocolli applicativi

- I protocolli applicativi sono in pratica l'unico protocollo di alto livello usato in pratica
- Alcuni esempi:
  - FTP
  - SMTP
  - POP
  - HTTP
  - Telnt/SSH

FTP (File Transfer Protocol)

- Permette il trasferimento di file tra elaboratori diversi connessi in rete
- Vengono aperte due connessioni TCP per ogni sessione FTP:
  - una connessione di controllo (porta 20)
  - una connessione dati (porta 21)
- Il protocollo stabilisce il formato dei comandi e dei messaggi scambiati
- FTP include un meccanismo di autenticazione basato su username e password passato dal client al server

SMTP (Simple Mail Transfer Protocol)

- Gestisce l'invio di messaggi di posta elettronica attraverso la rete
- La connessione tra i diversi server di posta avviene attraverso una connessione TCP (porta 25)

- Ogni utente é identificato dall'indirizzo:
  nomeutente@indirizzo_host

- Il processo di invio é batch

POP (Post Office Protocol)

- Protocollo per la lettura della propria posta da un mail server
- Sfrutta una connessione TCP sulla porta 110
- Fornisce comandi per avere la lista dei propri messaggi, scaricare un messaggio dal server al client, cancellare un messaggio dal server
- L'autenticazione è basata su una coppia "username-password" che viene scambiata in chiaro tra client e server

IMAP

- Protocollo per la lettura della propria posta da un mail server
- Sfrutta una connessione TCP sulla porta 143
- Fornisce comandi per avere la lista dei propri messaggi, scaricare un messaggio dal server al client, cancellare un messaggio dal server
- Il client e il server sono sempre sincronizzati
  - I messaggi sono sul server (anche se una copia viene salvata sul client per renderne possibile la visione offline)
  - Le operazioni sulle mail sono comandi inviati al sever visibili sul client in quanto
- L'autenticazione è basata su una coppia "username-password" che viene scambiata in chiaro tra client e server

Riassumiamo: invio

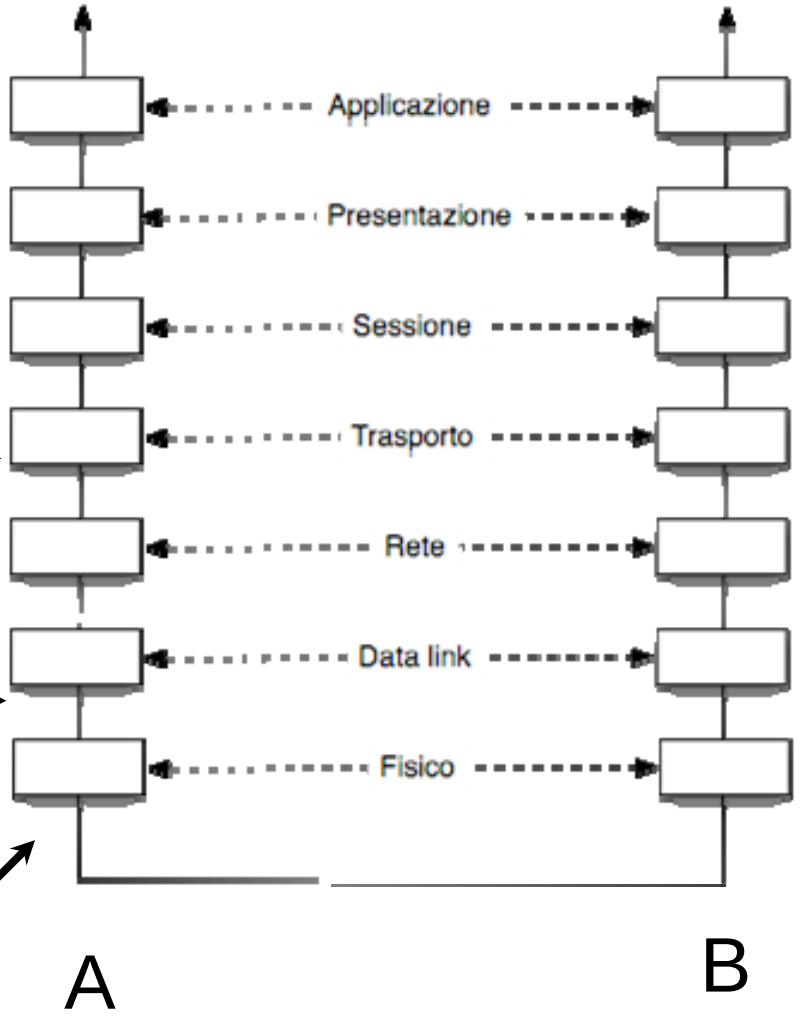
Il messaggio viene preparato con l'applicazione scelta dall'utente

Il messaggio viene diviso in pacchetti, a ciascun pacchetto viene aggiunta un'informazione sequenziale e tutti vengono spediti

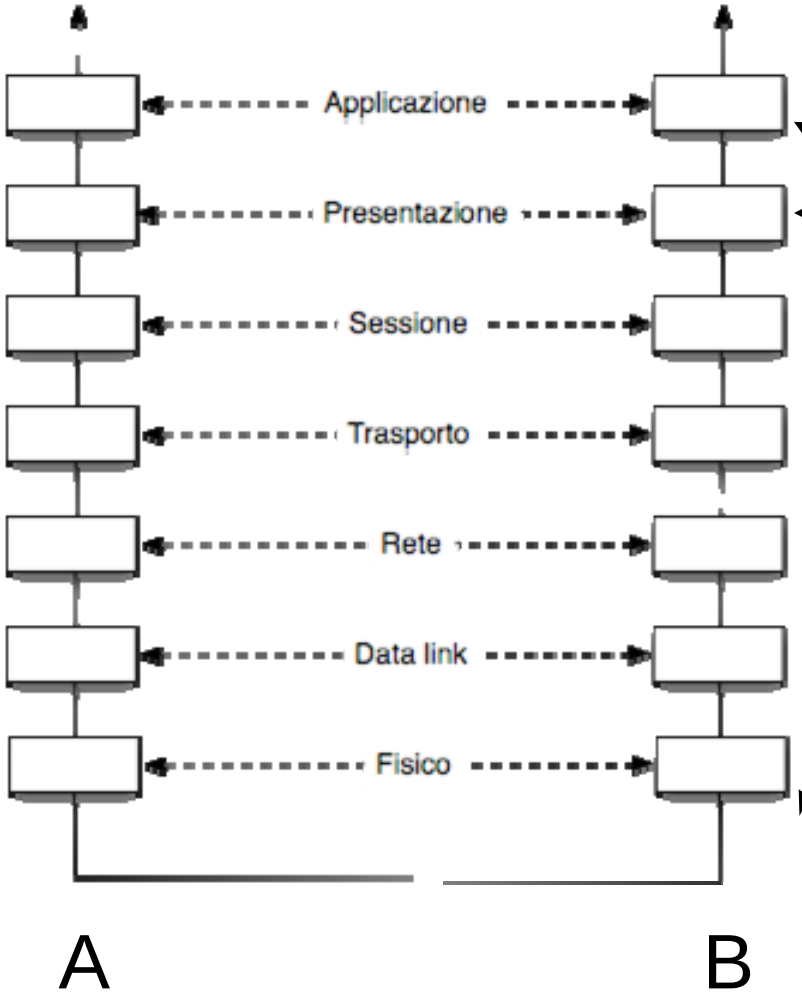
Per ciascun pacchetto viene calcolato il cammino su cui inviarlo

Da un pacchetto forma un frame, gruppi di bit che rappresentano il contenuto del pacchetto e informazioni aggiuntive per il controllo di correttezza

Converte i frame in sequenze di bit e li trasmette (si preoccupa del mezzo trasmissivo



Riassumiamo: ricezione



Il messaggio viene reso disponibile all'utente

Il messaggio viene ricomposto sfruttando l'informazione contenuta nei singoli pacchetti

Riordina i pacchetti ricevuti

Verifica che i frame siano corretti (controllando l'informazione aggiuntiva) e li passa sotto forma di pacchetto al livello superiore

Riceve una sequenza di bit e li passa sotto forma di frame al livello superiore