# Definizione

Un SD è un sistema dove i componenti hardware o software, comunicano e coordinano le proprie azioni solo attraverso lo scambio dei messaggi. Tale sistema deve dare all’utente l’impressione di lavorare con un singolo coerente sistema.

Se ogni nodo è autonomo allora non c’è una nozione di tempo globale, ognuno ha il proprio clock.

* No memoria condivisa
* no *global failure*
* no stato globale, no clock globale
* comunicazione attraverso scambio messaggi (message passing)
* autonomia di ogni nodo (debugging di un SD molto complicato)
* Best-Effort Programs

# Architettura Software

L’architettura definisce la struttura del sistema, i protocolli e le interfacce tra i nodi.

* **A livello** -> tipo client, server… I livelli = ruoli, ovvero ogni livello è una divisione FUNZIONALE del sistema.
* **Basate sugli oggetti** -> java-RMI
* **Basate su eventi** -> Applicazioni WEB basate sul callback (AJAX).
* **Centrale sui dati** -> tipo un file system condiviso
* **a strati** -> lo strato nasconde quello sottostante. Gli strati sono divisioni FISICHE del sistema. Usato da SO e Middleware

### ARCHITETTURE A STRATI

I layer superiori sono più specifici per i servizi, mentre quelli inferiori sono più generalisti. GLi strati superiori posso “chiamare” gli stati inferiori ma non il contrario.

### D.O.S.

* multiprocessori e multicomputer omogenei.
* sistema come un'unica grande macchina.
* Si preferisce distribuire la computazione piuttosto che i dati, cercando di bilanciare il carico di ogni macchina (**Computation Migration**).
* Si trasferiscono i dati usando interi file o porzioni del file (**Data Migration**).
* Si esegue un intero processo, o parti di esso, in siti diversi (**Process Migration**):
  + **Load Balancing**: distribuisci i processi sulla rete per uniformare il carico di lavoro
  + **Computation speedup**: i sottoprocessi possono essere eseguiti contemporaneamente su nodi diversi
  + **Accesso ai dati**: esegui il processo in remoto, anziché localmente

Il DOS scrive programmi - implementa servizi.

### N.O.S.

* risorse/servizi vengono condivisi e vengono usati come se fossero in locale.
* macchine eterogenee
* Governato dall’applicazione.

NOS e DOS generico

### MIDDLEWARE

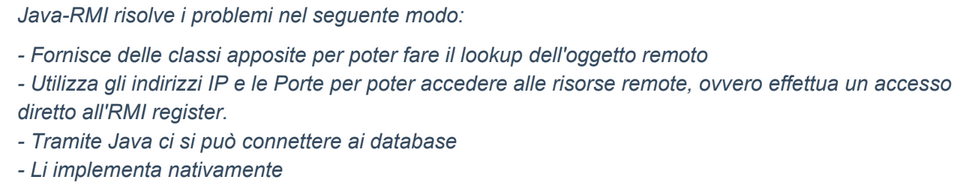
È uno strato software che sta sopra gli SO e fornisce servizi alle applicazioni soprastanti. È UN INTERFACCIA.

Invece che implementare servizi per ogni singola applicazione si scarica il lavoro al middleware che fornisce lui i servizi. Migliora la trasparenza di un sistema fornendo servizi specifico:

* Offre naming, trasparenza di accesso, persistenza (dati), transazioni distribuite, sicurezza
* middleware supportano NOS
* DOS è sempre chiuso, non posso aggiungere una CPU mentre il pc è acceso, devo spegnerlo e poi possa aggiungerla. Nei NOS e Middleware posso aggiungere livelli/macchine al fly.

# Problemi Fondamentali degli SD

* Bisogna saper individuare la risorsa, quindi servono nomi/indirizzi
* Bisogna saper trovare l’indirizzo per contattare il nodo, un processo o una risorsa. Usiamo un access point.
* Serve un protocollo per scambiare informazioni tra i vari nodi (TCP/IP)
* Devo saper interpretare le informazioni ricevute, bisogna stabilire una sintassi/semantica dei dati (è tutt’ora una problematica, negli SO è semplice ma tra i vari programmi no)



### TRASPARENZA REFERENZIALE

È difficile applicare la trasparenza durante un *system failure*, che può avvenire in qualunque momento.

* **Migration transparency**: spostamento di dati
* **Access transparency**: se accedo localmente o in remoto
* **Location transparency**: dove una risorsa è tenuta
* **Relocation/Mobility transparency**: risorsa viene trasferita mentre è ancora in uso
* **Concurrency transparency**: risorsa condivisa da user diversi
* **Persistence transparency**: in memoria o su disco

**Cosa si intende per protocollo?**

Si dice protocollo un insieme di regole che definisce il formato, l’ordine di invio e la ricezione dei messaggi tra dispositivi, il tipo dei dati e le azioni da eseguire quando si riceve un messaggio. Il protocollo non definisce la semantica.

* meccanismo = come viene implementato un servizio
* politica = comportamento protocollo