Definizioni e caratteristiche

Created	@February 28, 2023 11:22 AM
Class	Sistemi Distribuiti
Type	
Materials	ds01-Definizioni.pdf
Status	Done

- **Organizzazione a layer pura:** Solo chiamate ai livelli sottostanti (Reti ISO/OSI).
- **Organizzazione a layer misti:** Le chiamate possono essere effettuate a qualsiasi livello sottostante (Sistema operativo)
- **Upcalls e downcalls miste:** Possono essere effettuate chiamate sia a livelli sottostanti che a livello soprastanti.

Distributed Operating Systems

Sistema operativo distribuito

Gli utenti non sanno di star utilizzando più macchine, l'accesso alle risorse remote è come l'accesso alle risorse locali.

Data migration: si possono trasferire interi files oppure solo porzioni di file che ci interessano

Computation migration: trasferire la computazione, anziché i dati, tra i sistemi.

Process migration: Eseguire uno o diversi processi in diversi sistemi

Network Operating System

L'utente sa di utilizzare più macchine

Il **NOS** dà espliciti metodi di comunicazione:

- Comunicazione diretta tra processi
- Esecuzione concorrente di processi da applicazioni distribuite

- Servizi, come la migrazione di processi, sono gestiti da applicazioni

Accesso esplicito alle risorse di altre macchine:

- Login remoto alla propria macchina (telnet, SSH)
- Remote desktop
- Trasferimento dati dalla macchina remota alla macchina locale tramite FTP

Middleware

Il middleware è un software che si trova tra il sistema operativo e le applicazioni, fornendo funzionalità di gestione delle comunicazioni e dell'interoperabilità tra sistemi distribuiti.

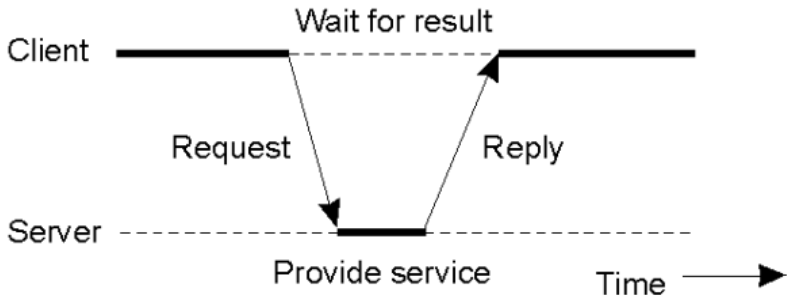
Servizi del middleware:

- Sicurezza: protezione dei dati e servizi
- Naming: nomi simbolici sono utilizzati per identificare parti di un SD. Possono essere utilizzati per identificare indirizzi reali o sono impliciti del middleware
- Accesso trasparente: modello di comunicazione che nasconde dettagli nell'invio dei messaggi
- Persistenza: servizio automatico di storage dei dati (file system o DB)
- Distributed transactions: offre un modello persistente che assicura le operazioni di lettura e scrittura

Item	Distributed OS		Network OS	Middleware-based OS
	Multiproc.	Multicomp.		
Degree of transparency	Very High	High	Low	High
Same OS on all nodes	Yes	Yes	No	No
Number of copies of OS	1	N	N	N
Basis for communication	Shared memory	Messages	Files	Model specific
Resource management	Global, central	Global, distributed	Per node	Per node
Scalability	No	Moderately	Yes	Varies
Openness	Closed	Closed	Open	Open

Modello Client-Server

Il modello di interazione tra un processo client e un processo server:



L'architettura di base prevede che un client acceda ad un server con una richiesta e che il server risponda con un risultato.

Problemi fondamentali

Generalmente ogni sistema distribuito ha 4 problemi da fronteggiare:

- Identificare la controparte: assegnamento di nomi
- Accedere alla controparte: come raggiungo un processo o una risorsa remoti?
- Comunicazione 1: è necessaria la condivisione di un protocollo
- Comunicazione 2: è necessaria la condivisione di una sintassi e di una semantica

Distribuzione trasparente

Trasparenza: nascondere i dettagli agli utenti

La trasparenza deve essere estesa a tutti gli ambiti. Non è possibile nascondere completamente i fallimenti, e la trasparenza ha ripercussioni sui costi e sulla velocità del sistema; a volte è più conveniente mostrare alcuni dati (es. lo stato di un server).

- Trasparenza all'accesso, permette l'accesso a risorse locali e remote con le stesse operazioni e con lo stesso formato dei dati;
- Trasparenza delle posizioni, permette l'accesso a risorse remote senza conoscere la loro locazione fisica nella rete;
- Trasparenza all'esecuzione concorrente, permette a più processi di operare con risorse condivise senza interferenze reciproche;
- Trasparenza alla replicazione, permette l'uso di istanze multiple per migliorare l'affidabilità e le prestazioni senza che gli utenti ne siano consapevoli;
- Trasparenza al fallimento, permette di ovviare ai fallimenti parziali per consentire di portare a termine un compito;
- Trasparenza alla migrazione, permette il trasferimento di utenti, dati e software su macchine o locazioni diverse senza che le operazioni effettuate ne risentano;
- Trasparenza alla scalabilità, permette la riconfigurazione dinamica di un sistema per migliorarne le prestazioni senza che gli utenti ne siano consapevoli;
- Trasparenza alla persistenza, permette di nascondere se una risorsa (software o dati) sia in memoria o su disco.

Nel campo dell'information hiding è importante distinguere tra che servizi un sistema offre (API) e come il servizio è stato implementato e sviluppato (frameworks e algoritmi specifici). Le interfacce devono essere progettate in modo da essere complete e neutrali; devono rispettare i criteri di interoperabilità, portabilità ed estendibilità per facilitarne il supporto.

I componenti che costituiscono un sistema distribuito sono logicamente indipendenti, cioè hanno un compito preciso svolto autonomamente, e ogni componente collabora con gli altri per svolgere un compito più complesso. C'è una netta separazione tra meccanismi e politiche: i meccanismi sono le capacità dei componenti, le politiche sono come le capacità vengono utilizzate per definire un comportamento (es. context switch e round robin). Se la separazione è rigida, vanno definiti meccanismi appropriati e di conseguenza la gestione diventa più complessa, quindi è necessario trovare un equilibrio tra flessibilità e complessità.

Distribueted Systems Basics

Per poter capire le richieste e formulare le risposte i processi devono concordare un protocollo che definisce il formato, l'ordine di invio e ricezione dei messaggi, il tipo di

dati e le azioni da eseguire quando viene ricevuto il messaggio. Le applicazioni TCP/IP si scambiano stream di byte (meccanismo) infiniti che vengono raggruppati in messaggi (politica) definiti dal protocollo condiviso.