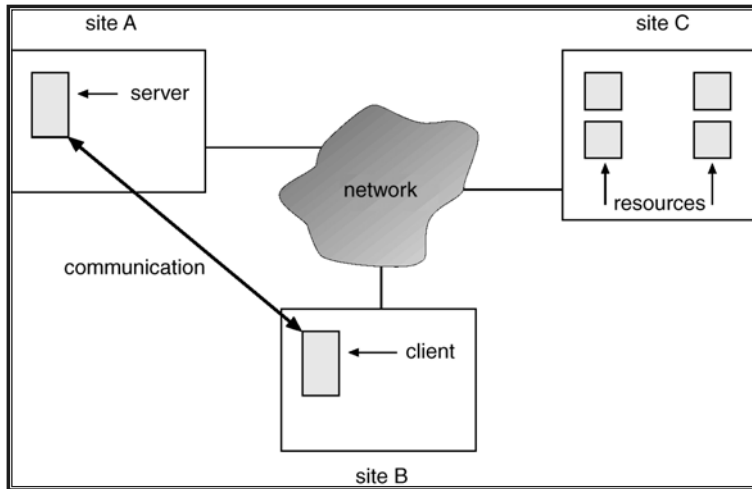


SISTEMI OPERATIVI DISTRIBUITI E FILE SYSTEM DISTRIBUITI

Sistemi Distribuiti

- Sistemi operativi di rete
- Sistemi operativi distribuiti
- Robustezza
- File system distribuiti
- Naming e Trasparenza
- Caching e Consistenza
- Network File System (NFS)

Un sistema distribuito



Motivazioni

■ **Condivisione di risorse**

Ad esempio:

- condivisione e stampa di file in siti remoti
- elaborazione di informazione in un database distribuito
- usare dispositivi hardware remoti.

■ **Accelerazione dell'esecuzione** – *condivisione di carico.*

■ **Affidabilità** – gestire fallimenti di siti remoti.

■ **Comunicazione** – scambio di dati e messaggi.

Sistemi Operativi di Rete

- Gli utenti sono a conoscenza dell'insieme dei calcolatori collegati in rete.
- L'accesso alle risorse dei vari calcolatori è realizzato esplicitamente tramite:
 - login remoto su un particolare computer

```
telnet si.deis.unical.it  
rlogin pc.deis.unical.it
```

- trasferimento di file tramite i meccanismi File Transfer Protocol (FTP)

```
ftp deis.unical.it  
get lezione12.pdf
```

Sistemi Operativi Distribuiti

- Gli utenti non sono a conoscenza dei computer disponibili. L'accesso alle risorse remote è simile all'accesso alle risorse locali.
- Le operazioni locali e remote avvengono sotto il controllo del sistema operativo.
- **Migrazione di dati** – i dati sono trasferiti in parte o tutti sul computer remoto e quindi sono riportati indietro al termine delle operazioni.
- **Migrazione delle computazioni** – spostamento delle operazioni sui nodi dove risiedono i dati da elaborare.

Sistemi Operativi Distribuiti

- **Migrazione dei Processi** – esecuzione di un processo o di parte di esso su calcolatori remoti.
- Motivazioni
 - Bilanciamento del carico
 - Accelerazione dell'elaborazione
 - Richiesta di hardware specifico
 - Richiesta di software specifico
 - Accesso ai dati – spostare i processi invece dei dati.
- Migrazione implicita e/o esplicita.

Robustezza

- Un sistema distribuito presenta aspetti specifici riguardanti la robustezza.
- Guasti più frequenti:
 - guasti ai collegamenti
 - guasti ai calcolatori
 - perdita di messaggi.
- Gestione dei guasti:
 - Individuazione del guasto
 - Riconfigurazione
 - Ripristino.

Rilevamento dei Guasti

- Non è facile capire che tipo di guasto si sia verificato.
- Generalmente si usa un protocollo di *handshaking*.
- Se il Sito **A** e il Sito **B** sono collegati direttamente si devono scambiare un messaggio ad intervalli temporali fissati Δt .
- Se il sito **A** non riceve un messaggio dopo un dato tempo Δt suppone la presenza di un guasto.
- Il sito **A** potrà inviare un messaggio di richiesta.
- Se **A** non riceve risposta potrà riprovare o assumere che esista un guasto e provare a contattare B su un percorso alternativo.

Rilevamento dei Guasti

- Se **A** non riceve nessuna risposta deduce che si è in presenza di uno dei seguenti guasti:
 - Il sito B non è attivo
 - Il collegamento non funziona
 - Il collegamento alternativo non funziona
 - Il messaggio si è perduto.
- **Ma A non ha certezza di cosa sia successo (quale tipo di guasto si sia verificato)!**

Riconfigurazione

- Quando un sito **A** verifica la presenza di un guasto deve avviare la procedura di ripristino:
 - Se il collegamento tra **A** e **B** è guasto deve informare tutti i siti del sistema.
 - Se il sito **B** è guasto bisogna informare tutti i siti che i servizi offerti da **B** non sono più disponibili.
- Nel caso il collegamento o il sito vengano ripristinati bisogna notificare nuovamente tutti i siti del sistema.

Problemi di Progetto

- **Trasparenza** – il sistema distribuito dovrebbe apparire come un calcolatore unico centralizzato.
- **Tolleranza ai guasti** – il sistema distribuito dovrebbe continuare a funzionare in presenza di guasti.
- **Scalabilità** – al crescere della domanda di elaborazione il sistema dovrebbe allocare le risorse necessarie per adattarsi al carico richiesto.
- **Clustering** – una collezione di macchine semi-autonome che si comportano come un sistema di calcolo singolo.

File System Distribuito

- Un **distributed file system (DFS)** è una implementazione distribuita su più calcolatori del modello classico di file system.
- Un DFS gestisce un insieme di dispositivi di memoria remoti e li mostra come fossero un **unico dispositivo**.
- I dispositivi di memoria possono essere eterogenei.
- Ogni insieme di file memorizzato su una singola macchina è detto **unità componente**.

Struttura dei File System Distribuiti

- Basata sui concetti di
 - **Servizio** – entità software che è eseguita su uno o più calcolatori per offrire una funzione a clienti sconosciuti.
 - **Server** – software di servizio eseguito su un calcolatore.
 - **Client** – processo che chiede un servizio eseguendo delle operazioni che costituiscono la sua interfaccia.
- Esempio:
 - Un interfaccia client per un servizio di gestione file è basato su operazioni come : *create, delete, read, write*.
 - L'interfaccia di un DFS non distingue tra file locali e file remoti.

Naming e Trasparenza

- **Naming** (*nominazione*) – associazione tra oggetti logici e oggetti fisici.
- Esempio – l’astrazione di un file che nasconde i dettagli che descrivono dove siano memorizzati i blocchi che lo compongono.
- In un DFS *trasparente* viene “nascosta” la indicazione del calcolatore dove è memorizzato il file.
- In caso di replicazione il naming restituisce le locazioni dove le diverse copie del file sono memorizzate.

Strutture di Naming

Buona astrazione

- **Trasparenza della locazione** – il nome del file non indica dove esso sia memorizzato fisicamente.
 - Ma esso è associato staticamente ai dei blocchi di disco su una particolare macchina.

Maggiore astrazione

- **Indipendenza della locazione** – il nome del file non deve essere cambiato quando la sua locazione fisica cambia (il file viene spostato fisicamente).
 - Maggiore separazione tra la gerarchia di naming e la gerarchia di memorizzazione.

Schemi di Naming: Tre Approcci

1. Nomi di file come **combinazione di nome della macchina e nome locale**: *macchina: nome_locale*.
2. Le **directory remote sono collegate alle directory locali** mostrando un albero coerente e un accesso trasparente. Le directory remote devono essere state montate. (NFS)
3. **Totale integrazione** tra le componenti del file system.
 - Una singola struttura globale di naming.
 - Se un server non è disponibile alcune directory potranno non essere disponibili. (Locus, Andrew)

Accesso a File Remoti

- Accesso a file remoti tramite richiesta ad una macchina remota (RPC).
- Per ridurre il traffico di rete e I/O sul disco si usa una cache che contiene i blocchi richiesti più frequentemente
 - Nuovi dati vengono trasferiti sul nodo client.
 - Per i blocchi disponibili l'accesso è eseguito localmente nella cache.
 - I file si possono trovare in più cache dei nodi client.
 - *Problema della consistenza della cache* – mantenere le copie nelle cache coerenti con la copia sul disco.

Locazione della Cache – Dischi o Memoria Centrale

- Vantaggi della cache sul disco locale:
 - Più affidabile.
 - I dati sono memorizzati in maniera stabile anche durante il ripristino.

- Vantaggi della cache in memoria centrale:
 - Permette nodi client senza disco (diskless).
 - L'accesso è più veloce.
 - Le memorie sono sempre più grandi.
 - Si può definire un unico meccanismo di caching sia sui client sia sui server.

Aggiornamento della Cache

- **Scrittura diretta** – i dati vengono scritti sul disco appena vengono aggiornati su una cache. Affidabile ma non efficiente. (NFS)

- **Scrittura ritardata** – i dati aggiornati su una cache vengono scritti sul disco in un secondo tempo. Efficiente ma molto meno affidabile. (Sprite)

- Variante: **scrittura su chiusura** - i dati vengono scritti alla chiusura del file. (Andrew)

Consistenza/Coerenza

- La copia nella cache è consistente con la copia sul disco sul server ?



- **Iniziativa del Client**

- richiesta al server di verificare la coerenza dei dati.

- **Iniziativa del Server**

- il server memorizza le parti sottoposte a caching. Controlla le eventuali incoerenze e le risolve.

Servizio con Informazione di Stato

- **Informazioni di stato sul server**

- **Meccanismo.**

- Un cliente apre un file.
- Il Server carica le informazioni sul file dal disco e le mette in memoria assegna un identificatore per la connessione con il cliente e apre il file.
- L'identificatore è usato per le operazioni sul file.
- L'area di memoria viene liberata sul server quando il cliente chiude il file.

- **Migliori prestazioni.**

- Minori accessi al disco e maggiori accessi alla cache.
- I server con stato possono ottimizzare l'accesso al file (es: accesso sequenziale).

File Server senza Informazione di Stato

Informazioni di stato sul client

- Le informazioni stanno tutte sul nodo cliente.
- Ogni richiesta deve identificare il file e la posizione richiesta.
- Non c'è la necessità di aprire e chiudere una connessione tramite *open* e *close*.
- Non presenta problemi in caso di guasto del server.
- Esempio: NFS.

Differenze

- Ripristino da guasti.
 - Un server con informazione di stato è critico in caso di guasto.
 - ❖ Comunica con i client per il ripristino.
 - ❖ Deve essere informato di guasti sui client.
 - Un server senza informazione di stato non è critico in caso di guasto.
 - ❖ L'informazione è sui clienti
 - ❖ Un nuovo server riceve dai client lo stato delle operazioni.

Replicazione dei File

- La replicazione di un file su macchine differenti migliora l'affidabilità, la disponibilità e l'efficienza del file system distribuito.
- Lo schema di naming associa il nome di un file ad una sua replica. L'esistenza di repliche dovrebbe essere invisibile all'utente ma non al sistema.

Problema:

- L'aggiornamento di un file replicato deve essere eseguito su tutte le repliche.

Sun Network File System (NFS)

- Modello e Implementazione per l'accesso a file remoti realizzato dalla SUN e usato in molti sistemi UNIX.
- Realizzato come file system distribuito di SunOS e Solaris.
- Basato sull'uso del protocollo UDP/IP basato su datagram non affidabili e Ethernet.

NFS

- Una rete di workstation interconnesse sono viste come un insieme di macchine indipendenti con il proprio file system che possono condividere i loro file in maniera trasparente.
- Una directory remota viene montata su una directory locale e viene vista come un sotto-albero locale.
- Nell'operazione di mount occorre specificare la macchina e la directory remota.
- Ogni directory remota può essere montata localmente e può soggetta a restrizioni di accesso come le directory locali.

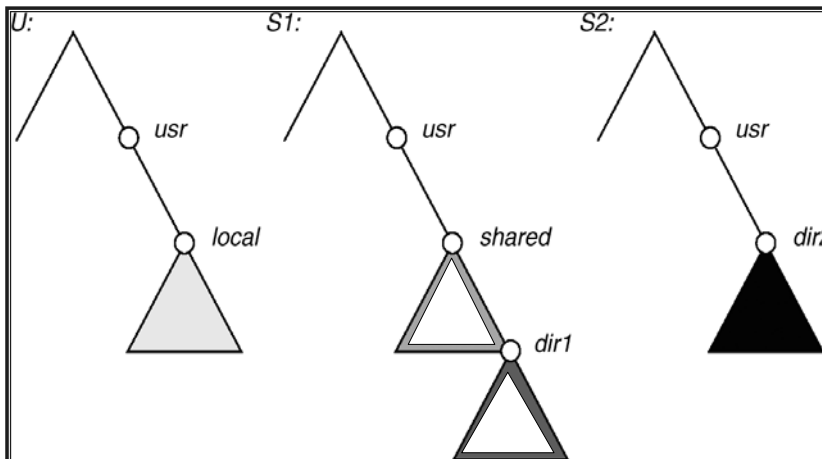
NFS

- NFS è progettato per essere usato in ambienti eterogenei di macchine diverse, di sistemi operativi diversi architetture di rete differenti. La specifica di NFS è indipendente dalle architetture fisiche.
- L'indipendenza è realizzata tramite l'uso di primitive di *chiamate di procedura remota (RPC)* costruite sopra il protocollo *External Data Representation (XDR)* usato tra due interfacce indipendenti dall'implementazione.
- La specifica di NFS distingue tra i servizi basati sul meccanismo di *mount* e i servizi di accesso remoto ai file.

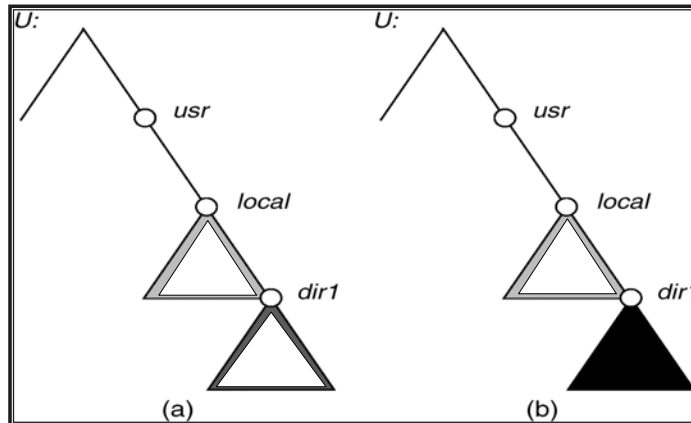
Protocol di mount NFS

- Stabilisce la connessione logica tra il server e il client.
- L'operazione di *mount* include il nome della directory remota e il nome del server su cui è memorizzata.
 - Una richiesta di *mount* è tradotta in una corrispondente RPC e viene inviata al server di *mount* sulla macchina remota.
 - *Export list* – specifica i file system locali che un server esporta per essere montati su macchine client, con l'indicazione di quali macchine client possono montarli.
- L'operazione di *mount* cambia solo il punto di vista dell'utente ma non cambia la configurazione del server.

Esempio: Tre File System Indipendenti



Esempio: Montaggio in NFS



Mount di
S1:/usr/shared su ***U:/usr/local***

Mount di
S2:/usr/dir2 su ***U:/usr/local/dir1***

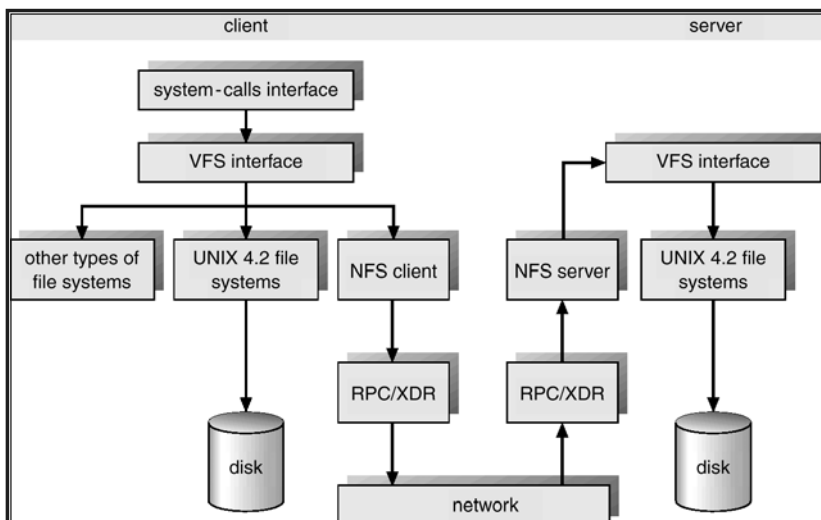
Protocollo NFS

- Fornisce un insieme di chiamate di procedure remote per operazioni remote file.
 - Ricerca di un file in una directory
 - Lettura di un insieme di entry di directory
 - Manipolazione di link e directory
 - Accesso agli attributi dei file
 - Lettura e Scrittura dei file.
- I server NFS sono senza stato ed ogni richiesta deve contenere tutti i parametri necessari.
- I dati modificati devono essere salvati sul disco del nodo server prima di essere disponibili per il client.
- Il protocollo NFS non fornisce meccanismi di controllo della concorrenza.

Livelli principali dell'architettura NFS

- Livello dell'*Interfaccia del file system UNIX* - basata su **open**, **read**, **write**, e **close**, e descrittori di file.
- Livello del *Virtual File System (VFS)* – distingue tra file locali da file remoti
 - Il VFS invoca il file system locale per i file locali (se ci sono più tipi di file system attiva quello opportuno)
 - Invoca le procedure del protocollo NFS per le richieste di file remoti.
- Livello del *Servizio NFS* – implementa il protocollo NFS.

Architettura NFS



Operazioni Remote NFS

- Corrispondenza (quasi biunivoca) tra system call di UNIX e system call delle RPC del protocollo NFS
- Uso di cache per migliorare le prestazioni degli accessi a file remoti.
- *File-blocks cache* – contiene blocchi di file remoti. Può essere usata solo se contiene dati aggiornati (necessaria una validazione sul server).
- *File-attribute cache* – aggiornata con gli attributi di file che arrivano dal server.
- Un client non libera i blocchi di scrittura ritardata finché i server non confermi che i dati siano stati scritti sul disco.

Domande

- Descrivere le differenze tra la trasparenza di locazione e l'indipendenza di locazione.
- Paragonare l'approccio con informazione di stato (sul server) e l'approccio senza informazione di stato (informazione sul client). Discutere vantaggi e svantaggi.
- Spiegare le motivazioni dell'uso di XDR nel SUN NFS.
- Quale è il ruolo delle chiamate di procedura remota nel Network File System ?