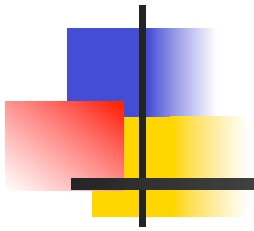
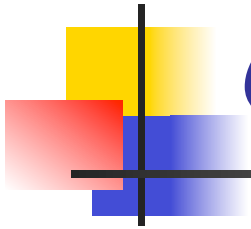


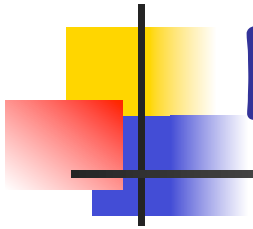
Il file system GPFS

Alessandro Brunengo
INFN-Genova

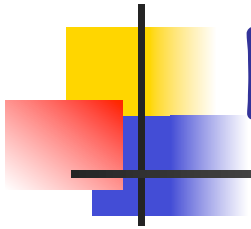


Contenuto

- Network Shared Disk
- Caratteristiche del file system
- Creazione e parametri del file system
- Gli storage pool
- File system management
- Fileset
- File attributes

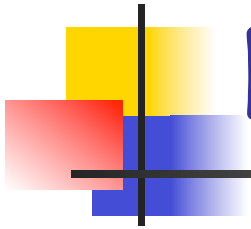


Network Shared Disk



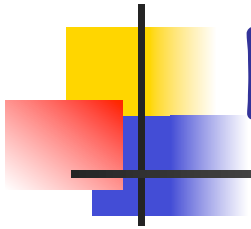
Network Shared Disk

- Il file system GPFS e' costituito da una collezione di dischi (Network Shared Disk), su cui GPFS memorizza dati e metadati
  - su linux: ogni block device con una entry in /dev/* (HD, partizioni, LUN esportate da RAID controller, multipath device, ...)
- Ogni NSD deve essere inizializzato tramite il comando **mmcrnsd**
- **mmcrnsd** registra il disco come NDS nei file di configurazione, e scrive sul device un **NSD descriptor**
- l' NSD descriptor contiene le informazioni del cluster di appartenenza, l' NSD name e l' NSD id, tramite i quali potra' essere riconosciuto



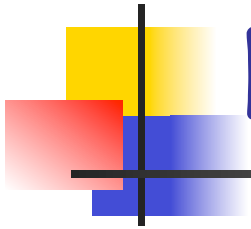
NSD discovery

- Allo startup GPFS esegue una procedura di **identificazione** degli NSD (**NSD discovery**)
- Su unix la procedura e' eserguita tramite lo script **/usr/lpp/mmfs/bin/mmdevdiscover**
  - questa procedura analizza **i device visti dal sistema operativo** e ne identifica il tipo in funzione delle caratteristiche di failover o multipath (**powerdisk** per *EMC power path*, **vpath** per *IBM virtual path*, dmm per *Device Mapper Multipath*, **gpt** per dischi visti da Windows, **generic** per dischi senza un multipath failover su linux)
  - per ciascun device stampa in output il nome ed il tipo
- E' possibile sovrascrivere, o integrare, il comportamento di questa procedura creando lo script **/var/mmfs/etc/nsddevices**
  - Se esiste, questa procedura viene eseguita prima di quella di default
  - Questa procedura puo' indicare a GPFS se si debba ulteriormente eseguire o meno la procedura di default
- Su ciascun device della lista viene quindi cercato l'NSD descriptor nel settore 2 del device
  - Il descriptor **identifica univocamente l'NSD**, che quindi viene riconosciuto indipendentemente dal nome che il sistema operativo ha dato al device



NSD discovery (cont.)

- Se il device viene identificato come NSD **noto al configuration database del cluster** GPFS, verra' utilizzato dal nodo per le operazioni di I/O
  - attenzione: **alla creazione il device type viene scritto nei file di configurazione** del cluster **(/var/mmfs/gen/mmsdrfs)**. Se il device type in seguito cambia (upgrade di OS, modifica dello script nsddevices, ...) l'NSD **non verra' identificato** come tale ed il device non verra' utilizzato dal nodo
- Tutti gli NSD registrati nei file di configurazione del cluster che non sono identificati come locali, vengono acceduti tramite un NSD server
  - questo e' possibile se almeno uno dei nodi direttamente connessi al device e' configurato come NSD server per quell'NSD
  - possono essere definiti fino a 8 NSD server per ciascun NSD
- La scelta della tipologia di accesso puo' essere forzata definendo un valore opportuno all'opzione *useNSDserver* del mount del file system:
  - **always**: utilizza solo l'accesso indiretto, tramite il primo server disponibile
  - **asfound**: utilizza prioritariamente l'accesso indiretto
  - **asneeded**: utilizza prioritariamente l'accesso diretto (default)
  - **never**: utilizza solo l'accesso diretto



NSD server e failover

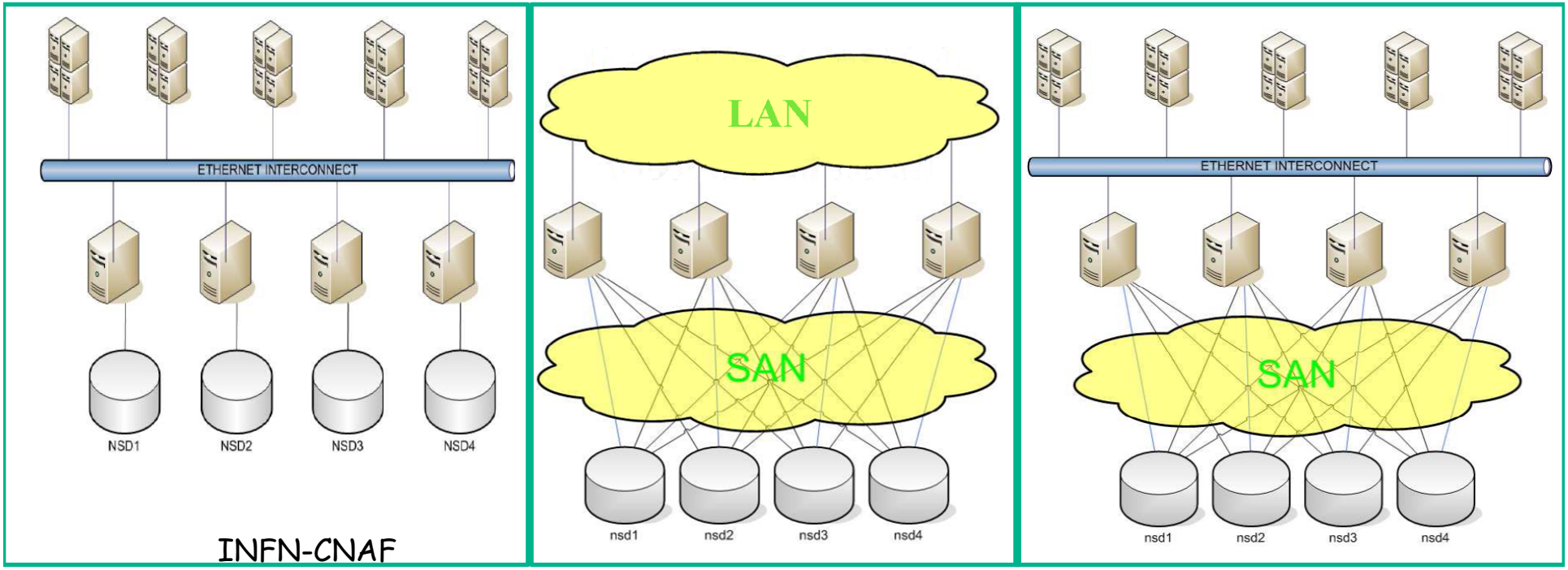
- Per ogni NSD possono essere definiti fino ad 8 NSD server
  - per ciascun NSD, solo un NSD server e' attivo in un dato istante
  - in occasione di server failure (server crash, server network failure, server SAN failure) l'NSD viene servito dall'NSD server successivo specificato nella lista
    - quando il server primario e' nuovamente disponibile, sul client si puo' riconfigurare l'accesso attraverso il primario manualmente, tramite il comando **mmnsddiscover**
- Il failover in occasione di un NSD server failure ed il recovery sono automatiche ed integrate in GPFS
  - l'operazione di **gestione della failure e di recovery** con l'assegnazione di un nuovo server per gli NSD comporta un ritardo inferiore al minuto sugli I/O in corso
  - tipicamente le applicazioni non falliscono e proseguono le attivita' di I/O senza problemi
  - il failover interviene anche quando l'NSD server viene fermato utilizzando la procedura ordinaria di shutdown di GPFS (**mmshutdown**)

Differenti topologie

Direct attached disks.
No failover available.

All nodes connected to the SAN.
No NSD server needed.

Mixed topology.
NSD server with failover.



Caratteristiche dell'NSD

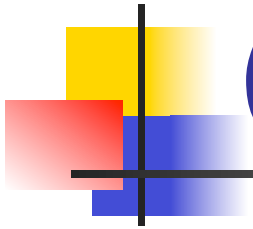
- **# mmcrnsd -F StanzaFile [-v {yes |no}]**

  StanzaFile deve contenere una serie di stanze che defniscono i psarametri di configurazione per ogni disco da inizializzare:

```
%nsd: device=DiskName
      nsd=NsdName
      servers=ServerList
      usage={dataOnly | metadataOnly | dataAndMetadata
             | descOnly}
      failureGroup=FailureGroup
      pool=StoragePool
```

  - **DiskName**: il block device name del disco da inizializzare (/dev/*)
    in assenza di ServerList, deve essere il device name con cui viene visto il volume dal nodo su cui si esegue il comando
  - **ServerList**: lista di NSD server (fino a 8)
    - opzionale: non necessario se tutti i nodi accedono alla SAN
    - se presente, il DiskName deve essere quello con cui il primo nodo in ServerList vede il device

Caratteristiche dell'NSD (cont.)



- **DiskUsage**: definisce cosa conterra' l'NSD (dataAndMetadata, dataOnly, metadataOnly, descOnly)
  - utilizzato all'atto della creazione del file system
- **FailureGroup**: intero (tra -1 e 4000) che indica il failure group dell'NSD. Informazione utilizzata all'atto della creazione del file system per decidere come replicare dati, metadati e filesystem descriptor (le repliche vengono collocate su NSD appartenenti a **failure group differenti**)
  - tutti i dischi con uno stesso point of failure dovrebbero essere configurati con lo stesso failure group (es: le LUN esportate da uno stesso controller o i dischi direttamente connessi ad un nodo)
- **DesiredName**: stringa che identifica l'NSD univocamente nel cluster
  - parametro utilizzato nei comandi GPFS per indicare l'NSD (o disk)
- **StoragePool**: nome dello storage pool a cui l'NSD deve appartenere
  - parametro utilizzato all'atto della creazione del file system
  - il nome deve essere univoco nell'ambito del file system

Note sui parametri dell'NSD

- **DesiredName**: non puo' essere cambiato
  - per cambiarlo si deve rimuovere l'NSD e ricrearlo
- **ServerList**: per modificare la lista degli NSD server si utilizza il comando **mmchnsd**:

 # mmchnsd {"DiskDesc[;DiskDesc...]" | -F StanzaFile}

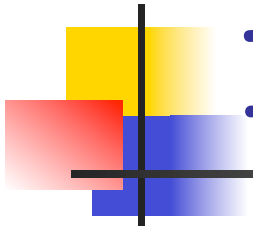
dove **DiskDesk** e' una stringa "**DiskName:ServerList**"

- se l'NSD fa parte di un file system, prima di eseguire mmchnsd il file system deve essere smontato su tutto il cluster
- **DiskUsage** e **FailureGroup**: e' possibile modificarli al volo senza interrompere l'I/O, tramite il comando **mmchdisk**
- **StoragePool**: per essere cambiato il disco deve essere rimosso dal file system (**mmdeldisk**) e poi riaggiunto (**mmadddisk**)

Visualizzazione degli NSD

- Per visualizzare gli NSD definiti nel cluster si usa il comando **mmlsnsd**
  - visualizza gli NSD definiti, l'eventuale file system di appartenenza, l'elenco degli NSD server definiti
  - l'opzione **-m** permette di visualizzare il nome del device con cui l'NSD viene visto dagli NSD server

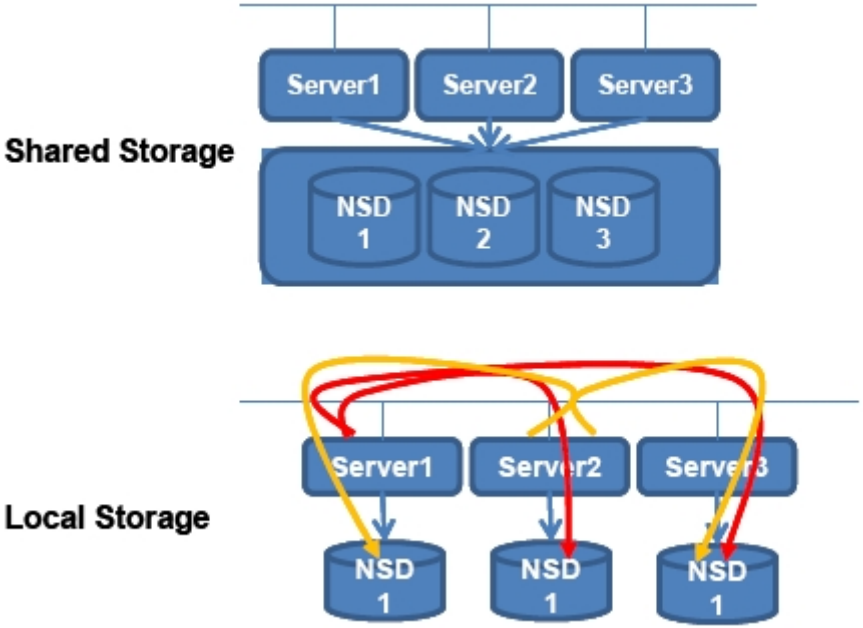
```
# mmlsnsd

 File system   Disk name    NSD servers
---------------------------------------------------------------------------
 backup_dev    f0_a4        bcksrv2.ge.infn.it
 test_dev      part1        (directly attached)
 test_dev      part2        (directly attached)
 (free disk)   part10       (directly attached)
 (free disk)   part3        (directly attached)
 (free disk)   part4        (directly attached)
 (free disk)   part5        (directly attached)
```

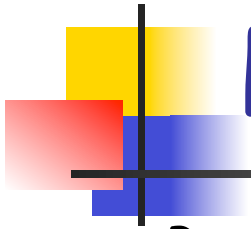


Il file system

Il file system GPFS

- Il file system GPFS e' costituito dalla colleazione di uno o piu' dischi (Network Shared Disk) connessi a nodi appartenenti al cluster
- I dischi possono essere locali (direttamente accessibili da un solo nodo) o volumi visibili via SAN (direttamente accessibili da piu' nodi contemporaneamente)



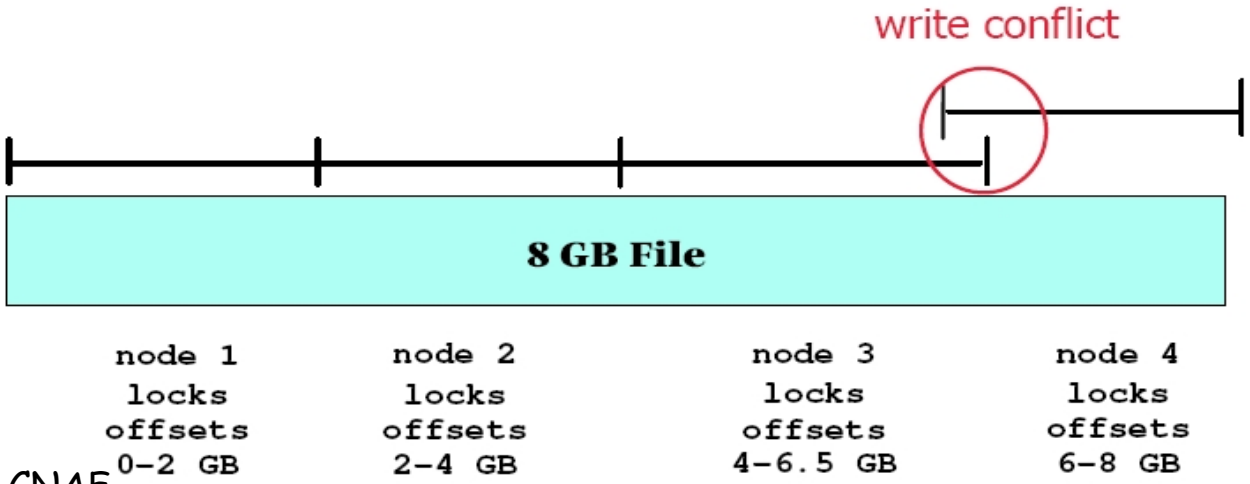


Posix e network file system

- Posix file system
  - supporto per le system call open(), read(), write(), close(), lseek(), unlink() etc..
  - supporto per tutti i comandi shell che operano sul file system (ls, rm, cd, …)
  - ownership e permission unix-like
  - supporto per ACL posix e NFSv4
- Network file system
  - il file system e' accessibile da tutti i nodi del cluster
    - su tutti i nodi il namespace e' omogeneo (stesso mount point)
  - supporto per l'accesso da parte di altri cluster GPFS in modalita' nativa (remote cluster export)
  - supporto per l'export via NFS o samba
    - supporta l'implementazione di NFS ad alta affidabilita' (CNFS)

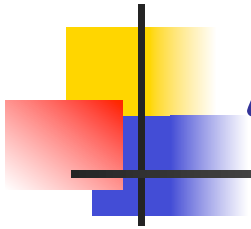
Parallel file system

- Parallel file system
  - shared disk: tutti i dischi vengono utilizzati contemporaneamente da tutti i nodi (direttamente o tramite NSD server), per dati o metadati
  - stripe: il singolo file viene suddiviso in blocchi che vengono collocati su tutti i dischi del file system in operazioni di I/O parallele (concomitanti)
  - il meccanismo di byte range locking ed il controllo di accesso eseguito dai token manager permettono l'accesso concomitante di piu' utenti allo stesso file (su regioni diverse)



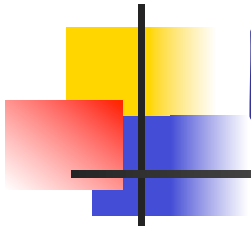
Solidita' ad effidabilita'

- High recoverability
  - GPFS e' un journaled file system
    - il file system dispone di *recovery logs* (uno per ogni nodo che accede al file system)
    - i recovery logs sono replicati su dischi appartenenti a diversi failure group
    - GPFS mantiene una rigida sequenza di operazioni e logging su data block e metadati
      - questo permette di recuperare la corretta struttura del file e del file system in occasione di crash di un nodo durante operazioni di I/O
  - GPFS implementa meccanismi idonei a configurare sistemi di **disaster recovery** sfruttando repliche sincrone su siti remoti



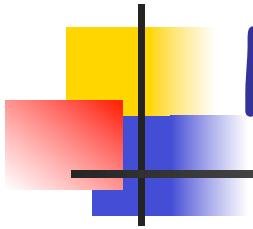
Alta disponibilita'

- Il cluster implementa meccanismi per la gestione della failure di **qualsiasi server**, operando il subentro di un altro nodo del cluster in failover, senza perdita di funzionalita'
  - mantiene la funzionalita' di I/O anche in occasione della perdita di un NSD server
  - mantiene la capacita' di recuperare lo stato del file system (compresi i lock) in caso di failure di file system manager o token manager
- Il file system supporta la **replica sincrona di dati e metadati** per sopportare la perdita di un disco
  - attraverso il concetto di failure group e' possibile automatizzare repliche su dischi senza point of failure in comune



Flessibilita' di management

- **Cluster file system**: management omogeneo e semplificato anche in ambienti di grosse dimensioni
  - la configurazione e le caratteristiche del file system sono note e condivise da tutti i nodi
- **Posix file system**: posso utilizzare comandi standard Unix
- Supporto per aggiunta/rimozione/sostituzione di dischi dinamicamente, senza interruzione di servizio
  - supporto anche per la ridistribuzione dinamica di dati e metadati in occasione di inserimento di nuovo spazio disco
- Supporto di **storage pool** e **policies** di movimentazione dati automatiche
  - permette anche di definire gerarchie di storage (tiering)
- Supporto per **separazione di dati e metadati**
  - impatto sulle prestazioni

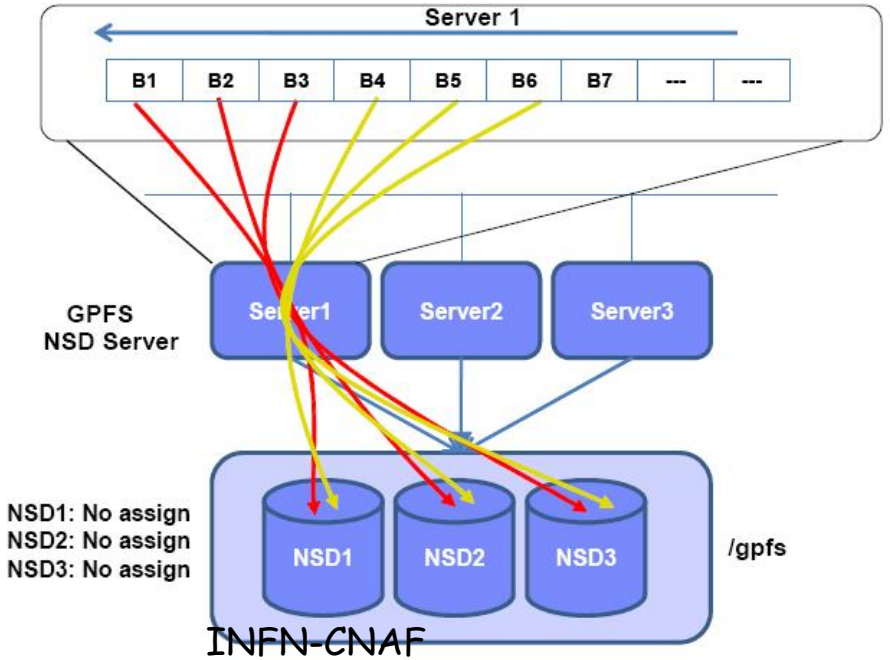


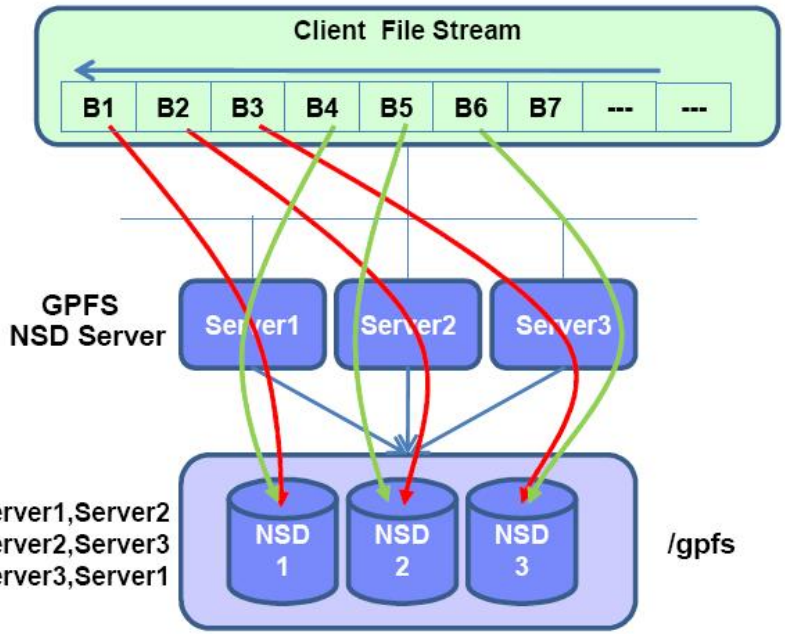
Performance e scalabilita'

- Accesso parallelo da parte di tutti i client su tutti i dischi utilizzati in modalita' stripe
  - permette di sfruttare la banda disponibile verso i dischi o verso gli NSD server
  - l'aggiunta di dischi o NSD server implica aumento di banda disponibile
- Riconoscimento di modalita' di accesso (sequenziale, strided) e sfruttamento della memoria per il read prefetch e per write behind
- Scalabile dinamicamente grazie al supporto per l'espansione del file system

Parallelismo dell'I/O sugli NSD server

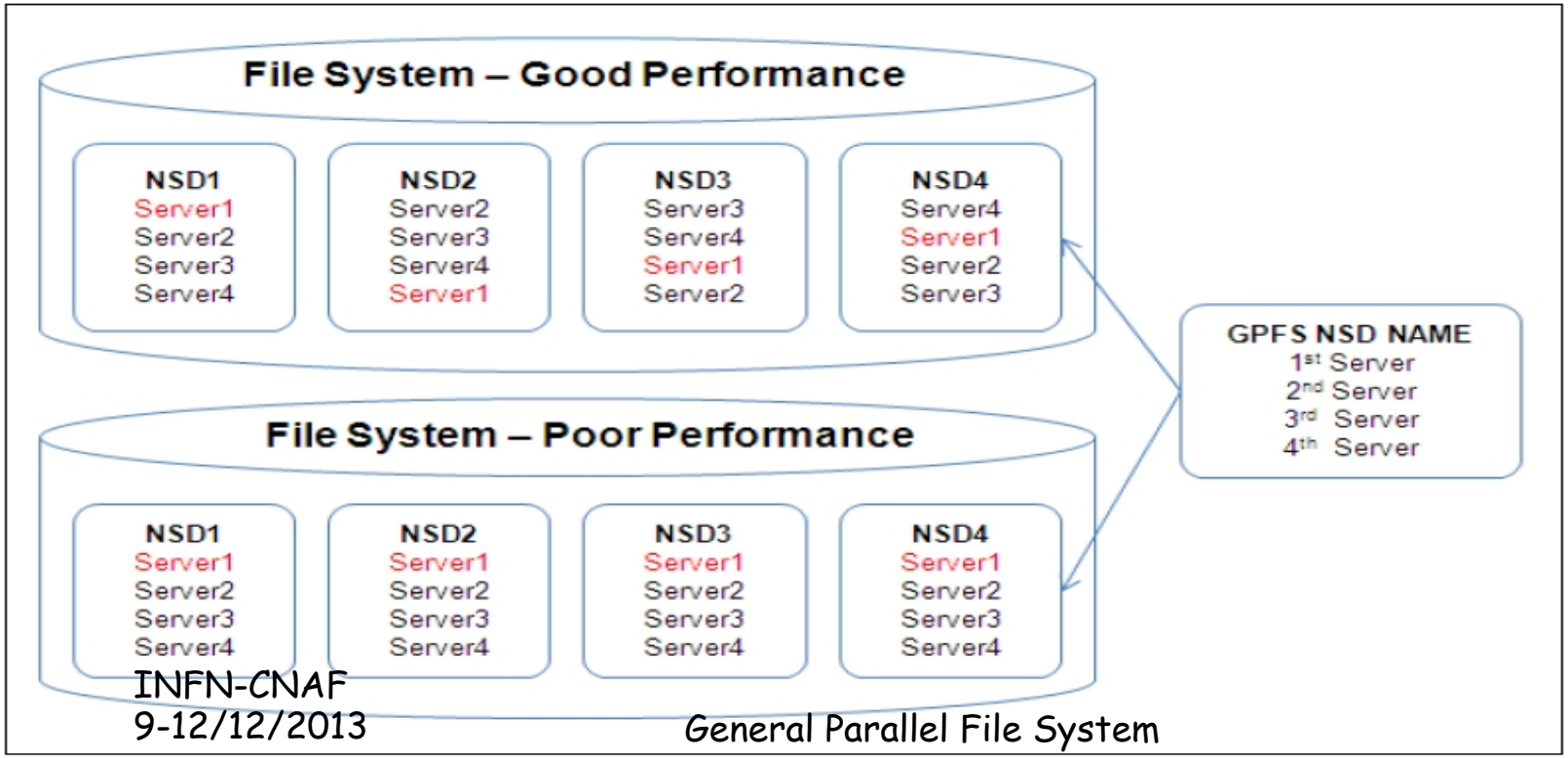
- Una configurazione bilanciata degli NSD server permette di sfruttare il parallelismo ed ottenere prestazioni migliori

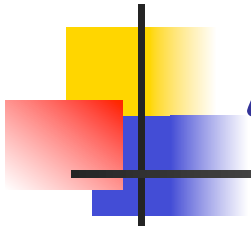




NSD server e performance

- Per ottenere le migliori prestazioni si devono adottare configurazioni opportune sulla scelta degli NSD server: nel secondo caso i server 2, 3 e 4 restano sempre inattivi:



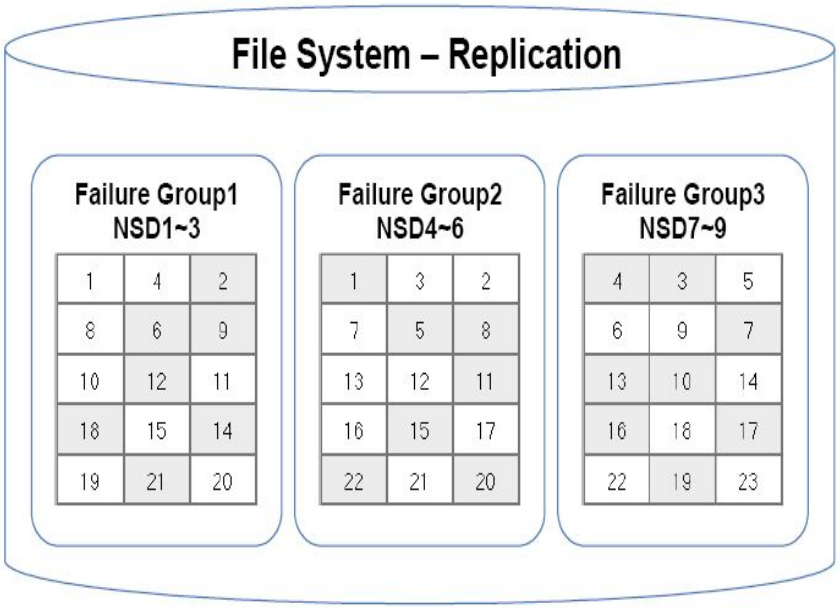


Altre caratteristiche

- Quota
- ACL (posix e NFS v4)
- Snapshot
- Fileset

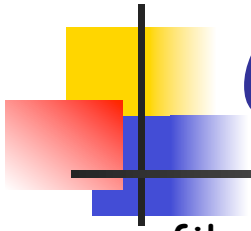
Le repliche

- GPFS supporta la replica (max tre copie) di dati e/o metadati
  - ogni disco viene assegnato ad un failure group (definito dall'amministratore all'inserimento del disco nel file system)
  - GPFS realizza la copia dei dati e/o metadati tra dischi appartenenti a failure group differenti
  - e' possibile attivare la replica dei dati, dei metadati o di entrambi



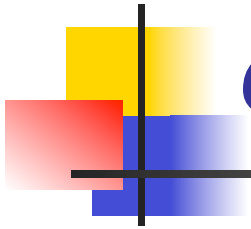
- Il comando per definire il fattore di replica **di un file** e' mmchattr
- A livello di file system si definiscono i valori di default e max replica factor per dati e metadati
- Attenzione alla disponibilita' di spazio disco ed inodes:
  - la replica dei metadati raddoppia il numero di i-nodes utilizzati (e di spazio usato per i metadati)
  - la replica dei dati raddoppia lo spazio utilizzato

Struttura del file system GPFS

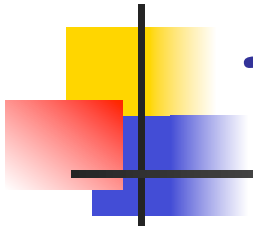


- file system descriptor
  - contiene, tra le altre cose, l'elenco ordinato dei dischi del file system ed il puntatore all'inode file
  - il file system descriptor viene creato in una o piu' repliche su dischi diversi alla creazione del file system o alla aggiunta di dischi al file system
- inode file: contiene alcuni inodes, tra cui quelli relativi alla root del file system, alla block allocation map, all'inode allocation file
  - la block allocation map contiene una mappa dei subblocks (1/32 della block size) allocati e liberi per ogni disco
    - la quantita' di subblocks indirizzabili per disco e' fissata alla creazione del file system (determina la massima dimensione di un disco per poter essere inserito nel disk pool del file system)
    - viene creata in porzioni distinte allocabili contemporaneamente: il numero determina quanti nodi possono contemporaneamente allocare/rilasciare data blocks sul file system
  - l'inode allocation file contiene l'inode allocation map, che contiene info sugli inode liberi ed occupati
    - puo' essere dinamicamente estesa fino al limite architetturale
- i file vengono messi su disco come in altri fs Unix: inodes, indirect blocks, data blocks

File system descriptor quorum



- Alla creazione del file system viene creato il file system descriptor
  - data la sue criticita', viene creato **in piu' repliche**
- Le repliche saranno collocate a seconda della disponibilita':
  - se esistono almeno 5 failure group differenti vengono create 5 copie (su dischi appartenenti a failure group diversi)
  - se esistono almeno tre dischi, vengono create tre copie su tre dischi diversi (su failure group diversi se disponibili)
  - se esistono due dischi, viene creata una copia per ogni disco
- L'accesso al file system viene consentito solo se c'e' il quorum sul file system descriptor
  - il quorum e' definito come la meta' piu' uno dei file system descriptor creati
  - quando vi sono due sole repliche, la perdita di un singolo disco impedisce l'utilizzo del file system
    - in casi come questi e' possibile definire un terzo disco di tipo "descOnly" allo scopo di utilizzarlo come quorum disk per il file system descriptor quorum
    - e' sufficiente un disco di 4 MB

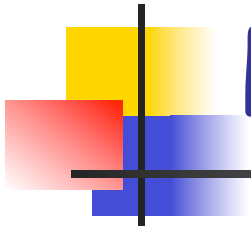


Creazione e parametri del file system

Creazione del file system

- Per creare un file system: **mmcrfs <device name> -F StanzaFile**
- Il comando crea un file system utilizzando gli NSD ed i POOL specificati nello StanzaFile, e crea (su tutti i nodi del cluster) un device file in /dev, associato al file system.
- La sintassi dello StanzaFile per specificare gli NSD e' la stessa vista per il comando **mmcrnsd**, mentre la configurazione di un disk pool ha la sintassi:

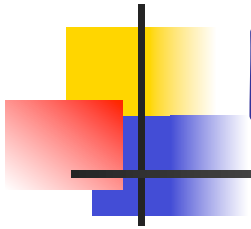
```
%pool:
        pool=StoragePoolName
        blockSize=BlockSize
        usage={dataOnly | metadataOnly | dataAndMetadata}
        layoutMap={scatter | cluster}
        allowWriteAffinity={yes | no}
        writeAffinityDepth={0 | 1 | 2}
        blockGroupFactor=BlockGroupFactor
```



Parametri dei pool

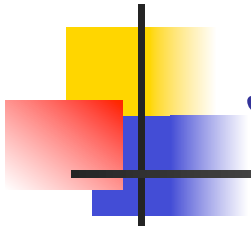
- storagePoolName: il nome del pool
- blockSize: deve essere uguale per tutti i pool diversi da system
- usage: solo system puo' contenere metadati
- layoutMap: definisce il criterio di allocazione dei blocchi (vedi oltre)
- allowWriteAffinity: definisce se GPFS deve utilizzare il File Placement Optimizer (si vedra' in seguito)
- writeAffinityDepth e blockGroupFactor: parametri specifici del FPO

- La block size per system e per gli altri storage pool puo' essere definita da opportuni switch del comando **mmcrfs**



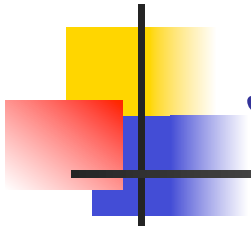
File system block size

- Si possono specificare numerosi parametri per definire le caratteristiche del file system. Attenzione: alcuni non sono piu' modificabili.
- La block size del file system e' un parametro non modificabile
- La block size (BS) e' la quantita' di dati scritti per singola operazione di I/O su ciascun disco del file system
  - se il file system e' costituito da N dischi, l'I/O verra' realizzata operando N operazioni di I/O ciascuna di BS bytes, in parallelo
    - un file di dimensione inferiore alla block size sara' scritto su parte di un unico blocco di un singolo NSD
  - la BS e' quindi la massima quantita' di dati che GPFS gestisce in una singola operazione di I/O
- GPFS supporta block size da 16 KB a 4 MB, con default 256 KB
  - per usare BS maggiori di 1M, si deve aumentare il parametro di configurazione del cluster **maxblocksize** in modo opportuno, con il comando **mmchconfig** (**richiede cluster shutdown**)



Subblocks

- GPFS divide un block in 32 subblocks
  - il subblock e' la minima quantita' di spazio allocabile
  - ogni file occupera' un certo numero di blocchi interi, piu' un certo numero di subblocks.
    - le porzioni di blocco utilizzate parzialmente sono dette frammenti
    - le operazioni di creazione e rimozione continua di file genera un aumento della frammentazione
  - l'occupazione minima di spazio dati un file e' pari ad un subblock (cioe' 1/32 della BS)



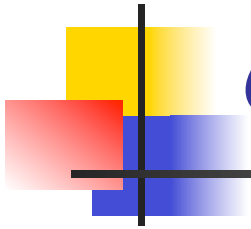
Scelta della block size

- La scelta opportuna dipende da diversi fattori
  - **ottimizzazione dell'occupazione**: scelta da operare in funzione della dimensione media dei file: se ci sono molti file piu' piccoli della dimensione del subblock si spreca molto spazio
  - **ottimizzazione delle prestazioni** di I/O sul volume RAID: e' sempre opportuno utilizzare una block size che sia multiplo della stripe size dei volumi RAID sottostanti
  - ottimizzazione in funzione del **pattern di accesso** dell'applicativo
    - applicativi che fanno I/O **sequenziale** di **grandi file** possono avere ottime **prestazioni** usando BS grandi (>= 1MB)
    - per accesso **randomico** di **piccole quantita' di dati** (general file service), si ottimizza **l'occupazione** con una BS piccola (64-512 KB)

Block Allocation Map type

- Il parametro –j di mmcrfs controlla il **modo** in cui, all'interno di un disco, vengono scelti i blocchi da allocare
  - **cluster**: l'allocazione viene fatta cercando di mantenere adiacenti i blocchi dei dati di uno stesso file
    - adatto a cluster piccoli (e' default per cluster con meno di 8 nodi)
    - prestazioni leggermente migliori inizialmente
    - causa un degrado delle prestazioni con l'aumentare dell'utilizzo di spazio e col numero dei nodi attivi sul file system
  - **scatter**: l'allocazione dei blocchi viene fatta con scelta random
    - adatto a cluster di dimensioni non piccole
    - le prestazioni sono inizialmente inferiori rispetto al cluster, ma costanti nel tempo (media sulla posizione del settore)
    - la frammentazione e' gestita meglio
- Non puo' essere modificato dopo la creazione del file system
  - Il parametro puo' essere definito differentemente su diversi storage pool entro lo stesso file system

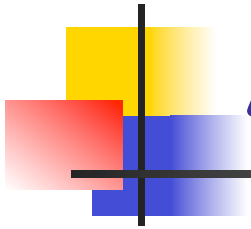
Numero di nodi con accesso concorrente al file system



- Su ciascun disco del file system viene creata una **Block Allocation Map** per indicare lo stato di utilizzo dei subblocks del disco (liberi/occupati)
- La Block Allocation Map viene creata divisa in parti **allocabili separatamente**, cioe' contemporaneamente
  - quando un nodo chiede o rilascia blocchi, la porzione della Block Allocation Map coinvolta viene marcata locked, modificata, quindi rilasciata
  - il numero di parti in cui e' divisa definisce il grado di parallelismo nella allocazione/deallocazione di blocchi sul disco
- Questo parametro viene definito all'atto della creazione del file system tramite l'opzione **-n** di **mmcrfs**
  - questo valore puo' essere in seguito modificato con il comando **mmchfs**, ma la nuova modifica riguardera' **solo i dischi di storage pool creati successivamente**
  - e' meglio sovrastimare questo valore che sottostimarlo

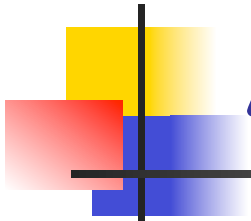
Max e default replica factor

- Il numero **massimo** di repliche supportate per i dati (**MaxDataReplica**) e per i metadati (**MaxMetadataReplica**) sono definite tramite le opzioni **–R** e **–M** di **mmcrfs**
  - i possibili valori per entrambe sono 1 (non si possono avere repliche), 2 o 3 (si possono' avere fino a due repliche)
  - questi parametri non sono modificabili dopo la creazione del file system
- Il numero di repliche per dati e metadati creati **per default** (alla creazione di un nuovo file o di nuovi metadati) e' definita tramite le opzioni **–r** e **–m** di **mmcrfs**
  - entrambe possono valere al minimo 1, al massimo il valore MaxDataReplica o MaxMetadataReplica rispettivamente
  - entrambe possono essere modificate tramite il comando *mmchfs*
    - solo i dati e metadati creati dopo la modifica adotteranno la nuova configurazione
    - per replicare tutti i dati e metadati secondo il corrente fattore di replica, si deve utilizzare il comando *mmrestripefs* con le opportune opzioni



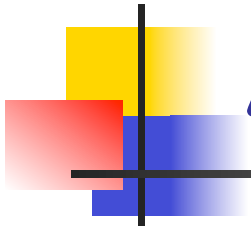
Altre opzioni

- **Deny-write open lock**: -D [posix | NFS4]
  - posix: per file system esportati via NFS v3 o non esportati
  - NFS4: per file system esportati via NFS v4, samba, o montati su nodi Windows
- **Mount allo startup**: -A [yes | no | automount]
- **Suppress atime update**: -S [no | yes]
  - la soppressione dell'update dell'access time riduce l'I/O sui metadati, ma puo' comportare problemi per policies basate sull'attributo ACCESS_TIME



Altre opzioni

- **Report exact mtime**: -E [no | yes]
  - yes: stat() e fstat() riportano il valore corretto
  - no: stat() e fstat() riportano il valore all'ultimo sync del client che modifica il file; in questo caso, possono esserci effetti non voluti per operazioni di backup o polocies che utilizzano l'attributo MODIFICATION_TIME
- **ACL type**: -k [posix | nfs4 | all]
  - posix per l'utilizzo di ACL tradizionali, nfs4 per il supporto di ACL NFS v4 o Windows
- **Strict replication**: -K [no | whenpossible | always]
  - no: se non puo' creare la replica, l'operazione di I/O non ritorna errore
  - whenpossibile: forza la replica se la configurazione dei dischi lo permette (abbastanza failure group)
  - always: forza sempre
    - la creazione del file o della directory fallisce se non e' possibile creare la replica quando deve essere forzata

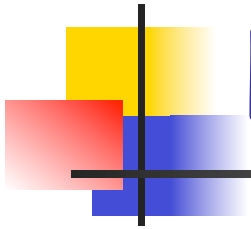


Altre opzioni (cont.)

- **Mountpoint directory**: -T <mountpoint>
  - defnisce il mount point presso il quale il file system verra' montato
    - la directory viene creata allo startup di GPFS se necessario
  - e' lo stesso su tutti i nodi del cluster
- **Attivazione quota**: -Q [yes | no]
  - attiva il supporto per la gestione della quota (user, group, fileset)
- **Visualizzazione quota per fileset**: --filesetdf | --nofilesetdf
  - se e' abilitata la quota, il comando df mostra valori corrispondenti alla quota del fileset e non al file system complessivo
- **Inode**: --inode-limit MaxNumInodes[:NumInodesToPreallocate]
  - definisce il numero massimo di inode per il file system, ed opzionalmente quanti sono preallocati
  - puo' essere modificato con il comando *mmchfs*

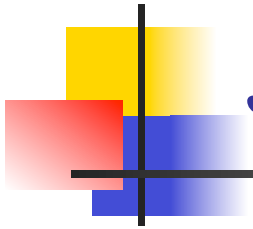
Altre opzioni (cont.)

- **Block size per i metadati**: --metadata-block-size <size>
  - definisce la block size per il pool system
- **Scopo di user/group quota**: --perfileset-quota | --noperfileset-quota
  - definisce i limiti di quota per user e group a livello di singolo fileset o di tutto il file system
- **Mount priority**: --mount-priority Priority
  - permette di specificare l'ordine in cui montare i file system allo startup: file system con maggiore priority vengono montati dopo; zero indica nessuna priorita, ed i file system verranno montati per ultimi



File system format level

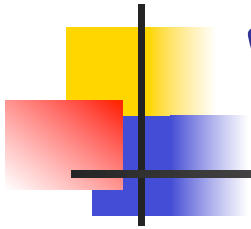
- Release successive di GPFS possono comportare modifiche nella struttura del file system
  - file system creati con versioni di GPFS precedenti possono essere migrati all'ultimo livello di format level, tramite il comando: **mmchfs** –V [full | compat]
  - compat abilita le sole modifiche compatibili con vecchie release di GPFS
  - full abilita tutte le nuove funzionalita'
    - in questo caso il file system potrebbe non essere piu' montabile da client con release di GPFS non up to date
    - in alcuni casi puo' essere necessario completare la migrazione con il comando **mmmigratefs**
- Alla creazione, il file system viene creato con l'ultimo format level supportato
  - e' possibile creare il file system con una formattazione compatibile con release precedenti di GPFS, tramite il parametro --version di **mmcrfs**



Storage Pools

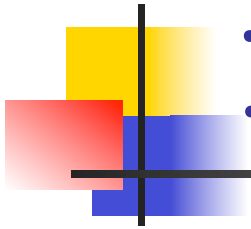
Storage pools

- Gli storage pools sono **collezioni di dischi** o volumi RAID di un file system
  - attraverso gli storage pools e' possibile **partizionare fisicamente** i device utlilizzati dal file system in gruppi
  - unitamente alle policies, e' possibile realizzare operazioni di **collocazione**, **movimentazioni**, **rimozione**, duplicazione di file tra storage pool diversi
- Esistono due tipi di storage pool
  - internal: pensati per lo storage on line; raggruppano i dischi veri e propri, volumi fisici gestiti direttamente da GPFS
  - external: pensati per il near line storage, o per l'archiviazione; questi sono oggetti gestiti da un applicativo esterno a GPFS (ad es. TSM)
    - GPFS mette a disposizione strumenti per definire una interfaccia tra GPFS e l'applicativo
    - GPFS non gestisce gli oggetti specifici che memorizzano i dati (dischi, tape), ma solo la movimentazione dei dati da e verso lo stprage pool esterno



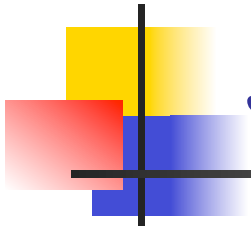
Vantaggi degli storage pools

- Sfruttando gli storage pool tramite le policies e' possibile
  - migliorare il rapporto prestazioni/prezzo, assegnando i dischi piu' costosi ai dati di maggior valore
    - e' possibile creare una gerarchia di gruppi di dischi e collocare i dati in funzione di svariati criteri
  - migliorare le prestazioni
    - ridurre le collisioni di accesso per i dischi migliori
    - ridurre l'impatto sulle prestazioni dei dischi piu' lenti o di rebuild
    - accedere a dati archiviati all'occorrenza, in modo semplice
  - migliorare l'affidabilita'
    - contenimento dell'indisponibilita' di dati in seguito a failure
    - creazione di nuovi storage pool all'occorrenza



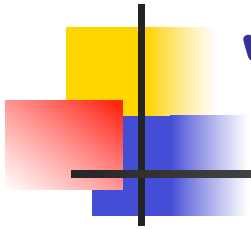
Internal storage pool

- L'appartenenza di un disco ad uno storage pool e' un attributo del disco, che viene specificato alla creazione del file system o alla aggiunta del disco al file system
  - lo storage pool di appartenenza e' identificato da una stringa (case sensitive) che deve essere unica per file system
- GPFS supporta fino ad 8 storage pools per file system, uno sempre esistente (***system*** storage pool), gli altri opzionali
- GPFS colloca dati sugli storage pool:
  - alla creazione del file, in base alla policy di placement
  - quando un attributo del file (size, access time, file set) soddisfa le condizioni di una policy di migrazione di file tra storage pool diversi
  - quando attributi del file system, del file set o di uno storage pool (percentuale di occupazione) soddisfano condizioni di policy di migrazione



Storage pool "*system*"

- Lo storage pool ***system*** contiene
  - strutture di controllo del file system
  - file speciali, directory, link simbolici
  - tutti i metadati associati ai file, compresi extended attributes e indirect blocks
- Opzionalmente lo storage pool ***system*** puo' contenere anche user data
- E' l'unico storage pool che puo' contenere metadati,gli altri possono contenere solo dati
  - cosa un disco possa ospitare (dati/metadati/entrambi) e' un differente attributo del disco
  - se un disco nello storage pool system puo' contenere dati, implicitamente lo storage pool system conterra' anche dati
  - se un disco puo' contenere metadati, deve appartenere allo storage pool system



"User" e "*system*" storage pool

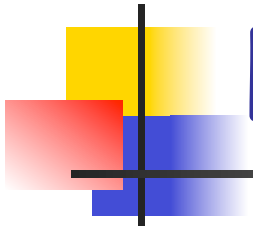
- In funzione della tipologia del suo contenuto e della sua criticita', e' caldamente suggerito utilizzare per lo storage pool system dischi ad alta affidabilita' e prestazioni
  - l'accesso ai metadati e' di tipo randomico
  - la replica dei metadati del file system e' fortemente consigliata
- Lo storage pool system viene creato automaticamente alla creazione del file system
- Gli user storage pool sono opzionali (max 7 per file system)
  - vengono automaticamente creati quando si inserisce nel file system un disco il cui attributo storage pool di appartenenza specifica uno storage pool inesistente
  - vengono rimossi automaticamente quando viene rimosso dal file system l'ultimo disco che vi appartiene
  - gli user storage pool possono contenere solo blocchi di user data

Listing storage pools

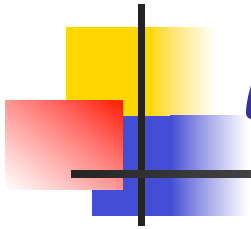
- Per visualizzare gli storage pool definiti in un file system, eseguire il comando:

```
# mmlsfs fs1 -P

flag  value                    description
---   ---------------          ---------------------------
-P    system;sp1;sp2 Disk storage pools in file system
```

- E' possibile definire e visualizzare l'appartenenza di un file ad uno storage pool (**mmlsattr**, **mmchattr**)

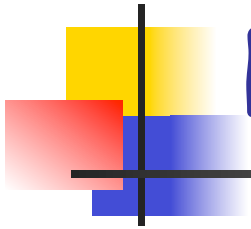


File system management



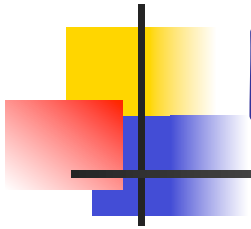
Mount e dismount

- **mmmount**: mount del file system
  - supporto anche per il comando Unix **mount**, ma senza features GPFS
  - supporto per binding mount su sistemi chrooted
- **mmumount**: dismount del file system
  - supporto per il comando Unix **umount**, ma senza features GPFS
  - non si puo' smontare se il kernel ha riferimenti attivi a file nel file system (NFS export, file aperti, CWD di processi)
- **mmlsmount**: mostra chi monta il file system (anche nodi remoti, o internal mount)
  - gli internal mount possono essere dovuti a
    - **binding del file system (sistemi chrooted)**
    - **esecuzione di operazioni di movimentazione dei dati (ad esempio restripe)**



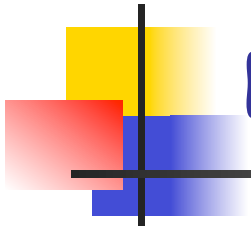
Rimozione del file system

- **# mmdelfs <filesystem-device>**
  - Il file system deve essere smontato
  - i dati non possono essere piu' recuperati
  - gli NSD del file system vengono nuovamente marcati come "available" per essere inseriti in un altro file system
- Se ci sono dischi non piu' accessibili, il comando fallisce
  - si deve utilizzare l'opzione -p per indicare di procedere ugualmente



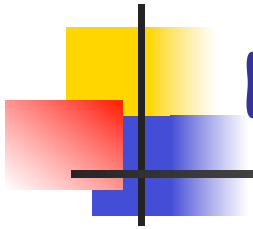
File system attribute

- **# mmlsfs <device> ...**
  - visualizza tutti i parametri di configurazione del file system
  - puo' richiedere il singolo parametro (usando il relativo switch)
- **# mmchfs <device> ...**
  - modifica uno o piu' parametri del file system
  - la modifica di alcuni parametri richiede il dismount del file system su tutto il cluster
    - vedere la man page



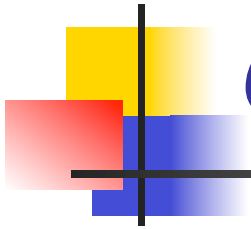
mmfsck

- **mmfsck** permette di analizzare il file system e di operare le necessarie modifiche per correggere inconsistenze
- opera in modalita'
  - **online** (sconsigliata): si limita a recuperare blocchi allocati ma non usati
    - puo' capitare per allocazioni fallite per problemi di spazio disco, concomitanti con node failure
    - blocchi rimangono marcati allocati ma non sono realmente usati
    - altre inconsistenze sono riportate ma non corrette
  - **offline** (a file system smontato): fa cose analoghe a fsck
    - trova file orfani (blocchi allocati ma directory entry missing): lost+found
    - dir entry che punta a i-node free: rimuove la dir entry
    - incorrect link count: corregge
    - incorrectly formed dir entry (non corrisponde il generation number dell'i-node): rimuove la dir entry
    - ...



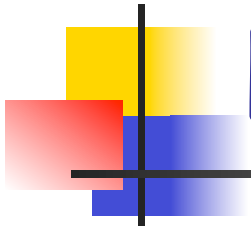
mmfsck

- Nota: mmfsck non puo' eseguire su file system con dischi in stato "down"
  - si deve eseguire mmchdisk per portarlo in stato "up" o "unrecovered"
  - vedremo domani cosa significa "stato down/up/unrecovered" di un disco



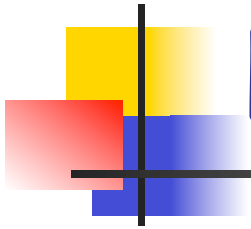
Occupazione del file system

- E' supportato il comando Unix "df"
- Esiste un comando GPFS: **mmdf**
  - visualizza l'occupazione **per disco**, **per storage pool**, e fornisce informazioni sui dischi (**failure group**, **store metadata**, ...)
  - visualizza anche l'occupazione di **i-nodes**
  - E' un comando che effettua I/O sui metadati del file system (eseguire con criterio): ci puo' mettere un po'
- **mmlspool** per un summary degli storage pool (senza i-node counting)



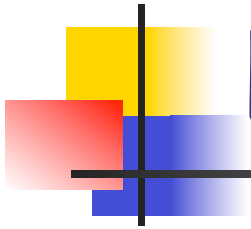
File system defragmentation

- La scrittura di un file viene all'occorrenza eseguita occupando l'ultimo blocco solo parzialmente
  - la scrittura si completa su alcuni dei 32 subblocks dell'ultimo blocco
- La frammentazione consiste nella presenza di tali blocchi parzialmente utilizzati
- La deframmentazione consiste nell'accorpare diversi frammenti nello stesso blocco, in modo da liberare blocchi interi
- **mmdefragfs** e' il comando per interrogare lo stato di frammentazione (flag **–i**) o per eseguire la deframmentazione del file system
  - l'operazione puo' essere eseguita a file system montato



Fileset

- GPFS supporta il concetto di fileset, che e' sostanzialmente un sottoalbero del file system che dal punto di vista amministrativo si comporta come un file system indipendente
  - e' possibile definire una quota per fileset
  - e' possibile definire user/group quota per fileset
  - si possono definire policy di collocazione e movmentazione di file per fileset
  - si possono creare snapshot di singoli fileset
- Il fileset e' identificato da una stringa di caratteri che deve essere univoca all'interno del file system

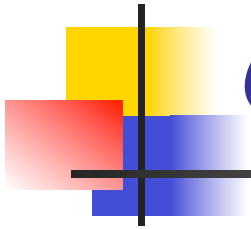


Fileset e i-node space

- Il fileset puo' essere
  - indipendente: lo spazio di i-node e' dedicato al fileset
    - questo ottimizza funzioni di scan dei file di un fileset, ad esempio nella applicazione di policy
  - dipendente: lo spazio di i-node e' quello del file set *root* o di un altro fileset indipendente
- Alla creazione del file system viene automaticamente creato il fileset *root*
  - non puo' essere cancellato
  - la radice del fileset *root* coincide con la root del file system
  - contiene file di systema, come i file di quota

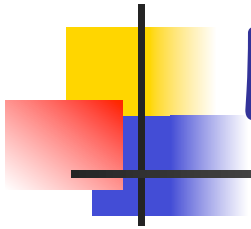
Junction del fileset

- Il fileset creato contiene una root directory vuota, e non e' visibile
- la accessibilita' del fileset viene realizzata creando un junction point all'interno del *root* fileset o di un fileset visibile
  - la junction ha l'aspetto di una normale directory (comprese le permission) ma non si possono eseguire le operazioni di rmdir e unlink su di essa



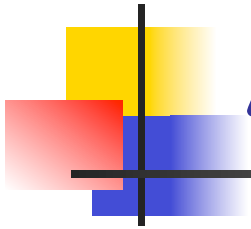
Creazione del fileset

- mmcrfileset <dev> <fs-name> [--inode-space <spec>] [-p <afm-attribute>...]
  - --inode-space new: fileset indipendente
    - si puo' specificare lo spazio di i-node per il fileset
  - --inode-space <existing-fileset>: crea un fileset dipendente che condivide l'i-node space con il fileset specificato



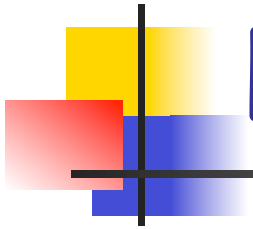
Link/unlink il fileset

- mmlinkfileset <dev> <fs-name> -J <junction path>
  - rende visibile il fileset posizionando la sua root sotto <junction path>, che viene creata col comando
- mmunlinkfileset <dev> {<fs-name>|-J <junction-path>} [-f]
  - rende il fileset invisibile
    - i file vengono conservati (ed i blocchi restano allocati!)
    - -f per forzare l'operazione, che fallisce se esistono file open entro il fileset

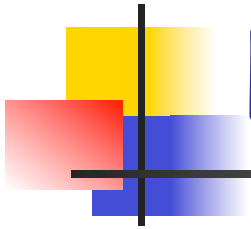


Altre operazioni sul fileset

- mmlsfileset <dev> …
  - visualizza le caratteristiche di uno o di tutti i fileset
  - vedere la man page
- mmchfileset <dev> <fs-name> …
  - modifica i parametri del fileset (nuova junction point, i-node space per fileset indipendenti, attributi AFM)
  - mmdelfileset <dev> <fs-name>

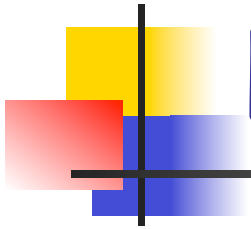


File attributes



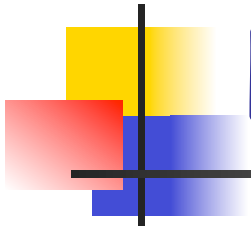
File attributes

- Gli attrtibuti controllabili a livello di file sono
  - **numero di repliche dei dati**
  - **numero di repliche dei metadati**
  - **storage pool di appartenenza**
  - **caching policy** (direct I/O)
  - **appendOnly mode**
  - **immutabilita'**
- Utilizzare **mmlsattr** per visualizzare gli attributi (oltre a fileset ed eventuale snapshot di appartenenza), **mmchattr** per modificarli
  - **mmlsattr** puo' mostrare gli eventuali stati di **exposed** (dati su dischi suspended), **ill replicated**, **ill placed** e **unbalanced**
  - **mmlsdisk** segnala l'esistenza di tali file a livello di file system



File attributes

- Replication factor e max replication factor
  - il replication factor deve essere non superiore a max replication factor (indipendenti per dati e metadati, valori possibili: 1 e 2)
  - in assenza di una definizione specifica tramite **mmchattr**, valgono i valori definiti a livello di file system al momento della creazione del file
  - in occasione di errore nella replica (mancanza di spazio o di failure group) o di rimozione di dischi, l'esistenza di repliche incomplete viene visualizzata dai comandi **mmlsattr** (per un file) o **mmlsdisk** (a livello di file system)
  - dopo la soluzione del problema, tramite i comandi **mmrestripefile** o **mmrestripefs** si sistemano le cose



File attributes

- **mmchattr –P <pool name> <file>** viene utilizzato per spostare un file in un dato storage pool
- GPFS supporta l'utilizzo del direct I/O sul file
  - **mmchattr –D <file>**
  - in questa modalita' le funzionalita' di caching non vengono utilizzate: i dati vengono copiati direttamente tra disco e user space buffer
  - largamente utilizzato da database manager
  - equivalente ad utilizzare O_DIRECT flag nella open()
- E' possibile proteggere file dalla involontaria rimozione o modifica, definendo le proprieta' di
  - **immutability**: il file non puo' essere rimosso, modificato, spostato
  - **appendOnly**: il file puo' essere modificato solo in append
  - in entrambi i casi e' possibile modificare il pool di appartenenza del file