Appunti di Enterprise Digital Infrastructures

Matteo Gianello

23 settembre 2013

Indice

1 Hard Disk

1.1 Caratteristiche base

Le caratteristiche che identificano un hard disk sono la capacità, ovvero la quantità di dati che esso può contenere; a questa caratteristica è stata associata una legge, simile a quella di Moore, la quale afferma che la capacità operazionale degli hd si moltiplica di un fattore 100 ogni 10 anni come mostrato in figura **??** Questo aumento di capacità è dovuto a diversi fattori,

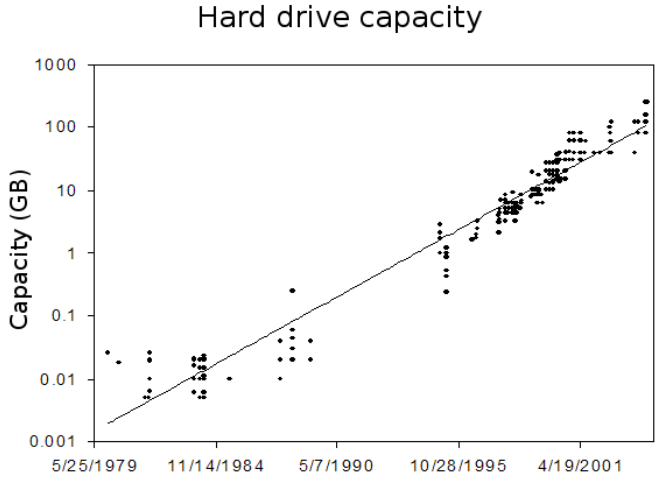

Figura 1: Andamento della capacità degli HD negli anni

aumento della densità dei dati sulla superfice, incremento della velocità di lettura, riduzione del tempo di accesso.
Questi miglioramenti non hanno però lo stesso grado di incremento; questo porta perciò a dei problemi, infatti, la capacità dei dischi cresce più velocemente di quanto la diminuzione del tempo di accesso si riduce. Questo porta alla formanzione di un collo di bottiglia che non permette di sfruttare a pieno le risorse.
Si utilizza così una gerarchia di memoria che permette un accesso più rapido alle risorse riducendo il fenomeno del bottlenecks**??**. Il tempo impiegato per accedere ad un dato in memoria utilizzando questa gerarchia di memoria può essere ricavato attraverso due approcci che portano allo stesso risultato. Il primo approccio considera $p(i)$ la probabilità di accedere allo strato $i - esimo$ della gerarchia di memoria per ricavare il dato. Ovviamente esistono alcune caratteristiche. Detta $p(i)$ la probabilità di accedere al layer i allora:

- $p(k) \leq p(j) \quad k > j$
- $\sum p(i) > 1$

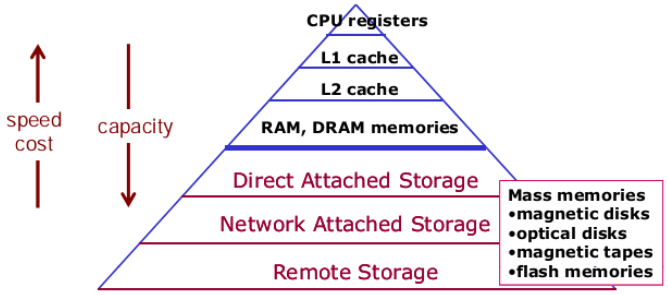

Figura 2: Gerarchie di memoria dalla più veloce alla più lenta

Da qui ricaviamo che il tempo di accesso alla memoria è dato da:

$$\begin{aligned} p(i) &= p(i-1) \times m(i-1) \\ p(i) &\times t(i) \\ total\ time &= \sum p(i) \times t(i) \end{aligned}$$

Il secondo approccio considera $P^*(i)$ che il dato si trovi nello strato i-esimo; in questo caso abbiamo che:

- $p^*(i) = (1 - m(i)) \times p(i)$ - è la probabilità di effettuare la ricerca nello strato i-esimo
- $\sum p^*(i) = 1$
- $t^*(i)$ è il tempo di accesso ad un dato nello strato i ed è uguale alla somma dei tempi di accesso a tutti gli strati sottostanti.

Il tempo di accesso in questo caso è dato da:

$$\begin{aligned} p^*(i) &= p(i) \times (1 - m(i)) = \\ &= (1 - \textstyle\sum_{j=1}^{i-1} p^*(j) \times (1 - m(i))) \\ total\ time &= \sum p^*(i) \times t^*(i) \end{aligned}$$

1.1.1 Hard Disk: componenti e caratteristiche

Gli hard disk sono formati da un disco che ha un diametro che varia dai 3,5 a 1,8 inch con due facce, raggiungono velocità di rotazione comprese tra 7200:15000 RPMed hanno densità delle traccie di circa 16000 tracce per inch divise in blocchi da 512Byte.

Testine Le testine viaggiano su un sottile strato di aria (alcuni nanometri) sopra il piatto; sono presenti una testina per ogni superfice di piatto disponibile. Queste quando sono a riposo, vengono parcheggiate o al centro del piatto o all'esterno.Il tempo impiegato dalla testina per passare dalla posizione di riposo al cilindro(insieme di tracce con lo stesso raggio) nel quale è contenuto il dato è detto *seek time* e può variare tra i 3 e i 14 ms

Rate di trasferimento Il rate di trasferimento è la quantità di dati che possono essere trasferiti dall'hard disk nell'unità di tempo. Nel caso di un hd che viaggia a 7200rpm si può avere un trasferimento fino a 1030 Mbits/sec. L'interfaccia di comunicazione SATA 3.0 permette un trasferimento di massimo 300MB/sec.

1.1.2 Solid State Disk

Architettura interna Nei solid state disk i dati sono immagazzinati in celle NAND che possono essere di due tipi: celle *SLC* che possono immagazzinare solo un bit, e celle *MLC* che possono immagazzinare più bit attraverso la tecnica del voltaggio a multilivello. Nel caso di utilizzo di MLC però si ha una tolleranza ai quasti minore a causa dell'alta deperibilità delle celle. Ad un livello più alto le memorie flash sono organizzate in *pagine* ovvero le unità più piccole che possono essere lette o scritte, e *blocchi* le più piccole unità ad essere cancellate. Una pagina può contenere fino a otto blocchi logici della dimensione di 512 byte; un blocco invece consiste solitamente in un insieme di 64 pagine ovvero 256KB

Gli inconvenienti Attualmente essi hanno un costo 10 volte superiore rispetto ai normali hd; in oltre la loro vità è inferiore in quanto le memorie flash possono essere scritte solo un limitato numero di volte. La differenza sostanziale tra tempi di lettura e scrittura comporta un incremento meno notevole rispetto a quello teorico nelle prestazioni. Le prestazioni di scrittura peggiorano col passare del tempo a causa delle cariche residue. Nel caso di SSD il controllore diventa il vero collo di bottiglia del sistema

1.2 Analisi delle performance

Il tempo di servizio del disco è dato da

$$seek\_time + rotational\_latency + data\_transfert\_time + controller\_overhead$$

dove le varie componenti sono rispettivamente:

- **seek time**: tempo impiegato dalla testina per raggiungere la traccia che contiene i dati

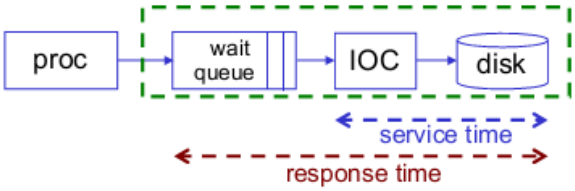

Figura 3: Tempi di servizio e di risposta di un HD

- **latency time**: tempo necessario per attendere che il settore richiesto passi sotto la testina
- **transfert time**: tempo di trasferimento da quando la testina inizia a leggere a quando il dato arriva al controllore. Dipende dal tempo di rotazione.
- **controller overhead**: tempo di managment del buffer.

Il tempo di risposta di un hard disk è variabile e dipende dal tempo speso in coda in attesa dell'esecuzione della richiesta e il tempo di esecuzione stesso; esso può dipendere da il numero di richieste in coda, il livello di utilizzazione della risorsa, il tempo di servizio dell'hd.

2 RAID

I RAID (Redundant Arrays of Independent Disk) sono stati introdotti intorno agli anni '80, con lo scopo di aumentare le dimensioni la sicurezza e le performance dei sistemi di storage. A livello di sistema i dischi vengono considerati come un unico disco con prestazioni e di dimensioni più elevate. I dati sono suddivisi sui vari dischi ai quali ci si accede in parallelo così da aumentare i rate di trasferimento e di I/O e bilanciare il carico di lavoro sui vari dischi.
Esistono due tecniche ortogonali per sfruttare il meccanismo dei RAID:

- il *data striping* per migliorare le prestazioni
- la *ridondanza* per migliorare l'affidabilità.

Il data striping prevede che i dati scritti in modo sequenziale siano divisi in unità su diversi dischi secondo un preciso algoritmo (es. round robin). Possiamo avere anche qui due tecniche di striping; la prima prevede che più richieste di I/O siano eseguite in parallelo su più dischi in modo da ridurre la coda di attesa e quindi il tempo di risposta del disco. La seconda tecnica prevede invece che una singola richiesta di I/O sia suddivisa in più blocchi che vengono scritti in parallelo su più dischi incrementando il rate di trasferimento per ogni richiesta.

Ridondanza Con l'aumentare delle dimensioni dei dischi e delle rispettive performance è aumentata anche la probabilità di errore, si è perciò cercato una soluzione a questo inconveniente, la ridondanza.
La probabilità di avere un errore in un array di 100 dischi risulta essere 100 volte più alta di quella che si ha su un singolo disco. Utilizzando però tecniche di correzione errore che utilizzano informazioni ridondanti scritte su dischi diversi è possibile recuperare le informazioni; questo meccanismo peggiora però le prestazioni di scrittura.
Possiamo suddividere i dischi RAID in categorie in base alla granularità dei dati e ai metodi di calcolo dei dati ridondanti che utilizzano.

Granularità dei dati Per quanto riguarda la granularità dei dati possiamo averne di due tipi: *fine grained* dove i dati sono suddivisi in piccole unità e ogni operazione di I/O accede a tutti i dischi, questa tecnica aumenta il troughtput dei dati ma solo una richiesta di input/output per volta può essere soddisfata. In una granularità di tipo *coarse* i dati sono disposti in unità più grandi così che le operazioni di I/O più piccole accedono ad un numero limitato di dischi mentre quelle più grandi accedono alla totalità dell'array; questo permette l'esecuzione in parallelo di più unità di I/O.

2.1 Architetture RAID

Esistono diversi tipi di architetture che si possono applicare alla tecnologia RAID che si suddividono in base alle tecniche con le quali implementano la ridondanza. In alcuni casi è possibile anche applicare più architetture allo stesso array di dischi.

RAID 0: striping Nel RAID di livello 0 i dati sono scritti su di un singolo disco logico e divisi in più blocchi distribuiti su più dischi fisici secondo un apposito algoritmo di striping. Questo sistema punta sulle prestazioni e non sull'affidabilità. Ha un costo basso in quanto non implementa la ridondanza e quindi ha le performance di scrittura migliori. Unico (e non piccolo) difetto è che un singolo errore su uno qualsiasi dei dischi comporta la perdita dell'intera totalità dei dati.

RAID 1: mirroring Nel RAID 1 quando i dati vengono scritti su un disco viene duplicato su un secondo disco. Questo permette di avere alta affidabilità e la lettura dei dati risulta essere molto veloce. Ma tutto questo comporta un alto costo infatti a parità di costo si utilizza solo il 50Teoricamente è possibile replicare i dati su più di un disco ma questa soluzione non è mai applicata in quanto troppo costosa a parità di affidabilità. Una soluzione più comunemente adottata è invece quella di avere una molteplicità di dischi ogniuno dei quali ha un suo mirror. Esistono due diverse configurazioni di questa architettura il RAID 0+1 e il RAID 1+0

RADI 0+1 In questa configurazione prima si applica lo striping e poi il mirroring (fig. **??**) Questa configurazione permette un'alta affidabilità e alte

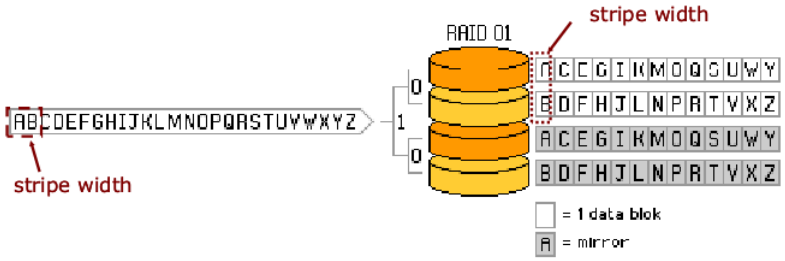

Figura 4: Configurazione RAID 0+1

prestazioni; ma richiede almeno quattro dischi e dopo un errore il sistema diventa di tipo RAID 0.

RAID 1+0 molto simile allo 0+1 il RAID 1+0 ha buone caratteristiche di velocità e tolleranza ai guasti, richiede sempre quattro dischi, si effettua lo stripping dei dati che vengono memorizzati sui dischi e sui loro mirror. Il grado di tolleranza ai guasti di un raid 1+0 è pari a quello di un raid 0. Questo meccanismo è usato soprattutto nei database con grandi carichi di lavoro.

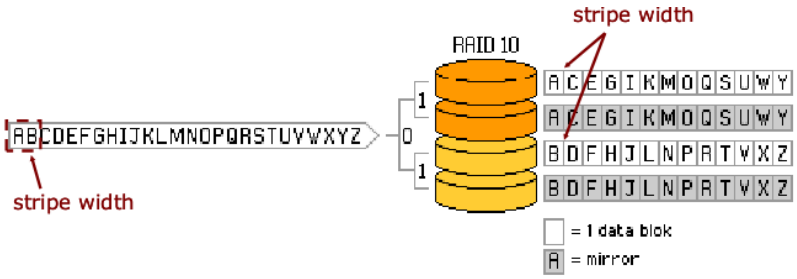

Figura 5: Configurazione RAID 1+0

Raid 0+1 contro 1+0 I blocchi memorizzati sono gli stessi e per molti controller non esiste alcuna differenza in quanto le operazioni di striping e di mirroring vengono eseguite simultaneamente. Nella configurazione 0+1 esiste un unica possibilità di errore un altro errore su un altro disco dell'array è un point of failure. Per effettuare il recupero dei dati sono necessari tutti i dischi dell'array. Nella configurazione 1+0 è tollerato un errore per ogni blocco di raid 1 senza compromettere la stabilità del sistema.

RAID 2 Il raid 2 sfrutta una serie di dischi nei quali viene calcolata la parità dei dati contenuti nei dischi di storage. Il numero di dischi di parità è uguale al logaritmo del numero di dischi di storage. Il costo di questa architettura è minore di quello dell'architettura RAID 1 ma comunque troppo elevato.
Questa configurazione può sopportare un numero di faliure pari al numero di subset usati per calcolare la parità dei dati.

Figura 6: Configurazione RAID 2

RAID 3 I dati sono interallacciati ciò permette l'utilizzo di un singolo disco per la parità dei dati. Viene usato in applicazioni che necessitano di una grande larghezza di banda ma che hanno un numero basso di applicazioni di I/O in quanto ogni operazione di lettura accede a tutti i dischi e ogni operazione di scrittura accede sia ai dischi dati che a quello di parità. Questo meccanismo permette l'asservimento di una singola operazione di I/O per volta.

Figura 7: Configurazione RAID 3

RAID 4 Simile alla configurazione RAID 3 in questo caso i dati sono interallacciati in blocchi; la lettura di dati più piccoli del blocco richiede l'accesso a un solo disco, la scrittura invece richiede l'aggiornamento del blocco richiesto e il ricalcolo della parità del blocco. In questo caso il disco di parità diventà il collo di bottiglia del sistema. Questa configurazione può sopportare la rottura di al massimo due dischi.

RAID 5 La configurazione a RAID 5 è un misto tra tutte quelle viste fino ad ora. I dati sono divisi in blocchi i quali subiscono lo striping sui vari dischi; la parità è distribuita su tutti i dischi come avveniva per i dati.

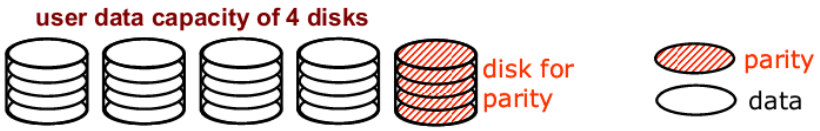

Figura 8: Configurazione RAID 4

Le operazioni di scritturasono più lente delle configurazioni 0 e 1 ma le operazioni di lettura sono più veloci del RAID 1. Il carico è bilanciato sui vari dischi e la configurazione può sopportare la rottura contemporanea di due dischi.

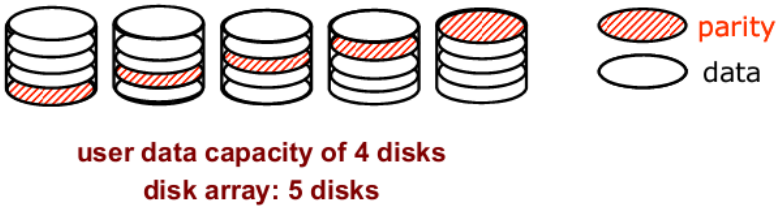

Figura 9: Configurazione RAID 5

Confronto tra le architetture 3, 4, 5 Le architetture RAID 3, 4 e 5 hanno le stesse caratteristiche a livello di overhead, sicurezza contro i guasti e capacità totale di immaggazinamento. Quello che cambia è la complessità del controller, la configurazione 3 richiede un controller molto semplice ma con la sincronizzazione dei dischi; la configurazione 4 ha nel controllore del disco di parità il suo collo di bottiglia. Il Raid 5 richiede invece un sofisticato algoritmo per il calcolo della parità. Visto che il costo delle tre architettura risulta uguale il raid 3 e 4 non vengono usati.

2.2 Raid: prestazioni

Le metriche principali per valutare le prestazioni in una architettura raid sono:

- **MTTF1: Mean time to failure** che indica il tempo che intercorre tra l'avvio del disco e il successivo errore.
- **MTTR:** indica il tempo necessario a riparare/sostituire un disco affetto da malfunzionamento.
- **MTTDLn: Mean time to data loss** indica il tempo richiesto per avere il numero di guasti nel disco da rendere irrecuperabile i dati sull'array di dischi.

Per calcolare l'MTTF dobbiamo innanzitutto assumere che i tempi di errore abbiano una distribuzione esponenziale e indichiamo con $F_x(t)$ la funzione di distribuzione dei tempi di failure.

$$F_X(t) = 1 - e^{-\frac{t}{MTTF}}$$

Considerando ora un array con n dischi otteniamo che il tempo di guasto dell'array è uguale al minimo dei quasti degli n dischi:

$$F_{min(X_1 \ldots X_n)}(t) = 1 - (1 - F_X(t))^n = 1 - e^{-\frac{nt}{MTTF}}$$

Possiamo approssimare questa quantità a:

$$F_{min(X_1 \ldots X_n)} \simeq \frac{nt}{MTTF}$$

in quanto la quantità $\frac{t}{MTTF} << 1$

3 Sistemi di storage

Le principali richieste che i nuovi sistemi di storage devono soddisfare sono di diversa natura; prima fra tutti la sicurezza dei dati in quanto essi non devono essere corrotti o persi. La reperibilità dei dati in quanto essi devono essere disponibili 7 giorni su 7 e 24 ore su 24 e disponibili per tutti gli utenti con determinati vincoli. Infine l'espansione dei sistemi di storage deve essere trasparente agli utenti finali. Queste caratteristiche hanno fatto in modo che i sistemi di storage si evolvessero da *Direct Attached Storage* (DAS) nel quale si aveva una visione centrata sull'host ad una logica di *Storage Area Network* (SAN) centrata sulla rete.
Il modello fisico di come sono strutturate le diverse architetture è mostrato in figura **??** Il modello logico è invece rappresentato in figura **??**

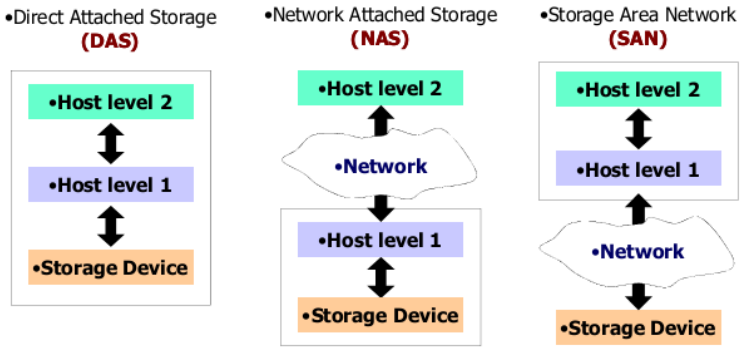

Figura 10: Modello fisico delle varie architetture di storage

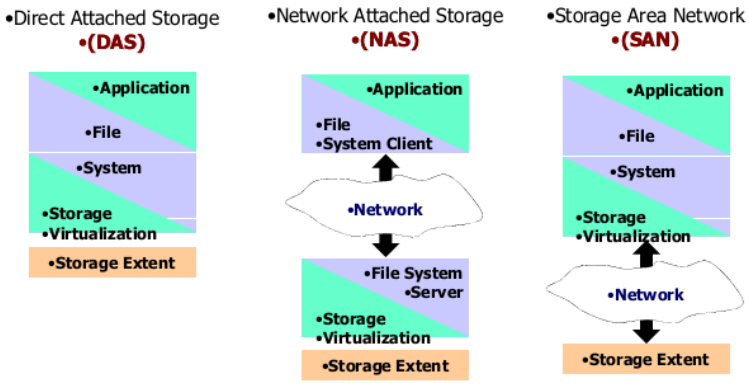

Figura 11: Modello logico delle varie architetture di storage

3.1 Direct Attached Storage

Come si nota schema logico di figura ?? nei sistemi DAS l'architettura è concentrata su un unica macchina, se si volesse accedere ai dati bisognerebbe passare dalla macchina che ospita il disco. Questo implica i classici problemi di compatibilità tra i vari sistemi operativi. Questo tipo di architettura

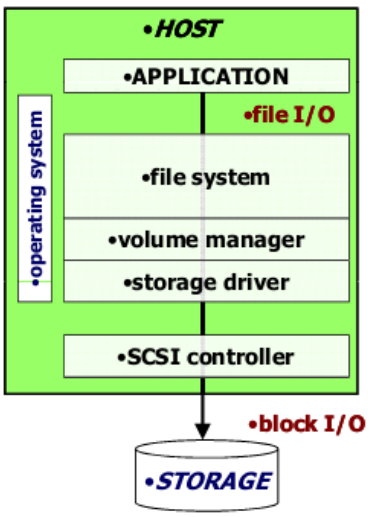

Figura 12: Modello logico dell'architettura DAS

rapportata ad un contesto di rete come quello di figura ?? implica grossi problemi di scalabilità una complessità molto elevata e prestazioni limitate.

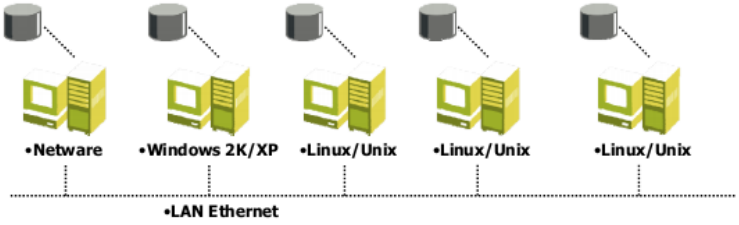

Figura 13: Modello di rete DAS

3.2 Network Attached Storage

Il modello logico dell'architettura NAS è rappresentato in figura ??; come si vede questo tipo di architettura prevede un unico sistema che gestisce i dischi questo permette di rendere trasparenti i meccanismi di gestione dei dischi ai diversi host come l'aumento o la sostituzioni di dischi. Il vantaggio

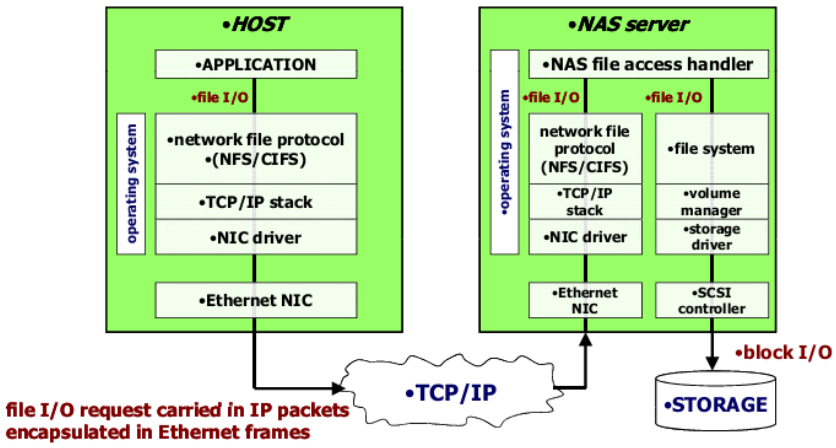

Figura 14: Schema logico dell'architettura NAS

principale è la grande scalabilità basta aggiungere un disco al NAS o una unità NAS. Come si vede dallo schema di rete in figura **??** ogni unità NAS ottiene un indirizzo IP dalla rete.

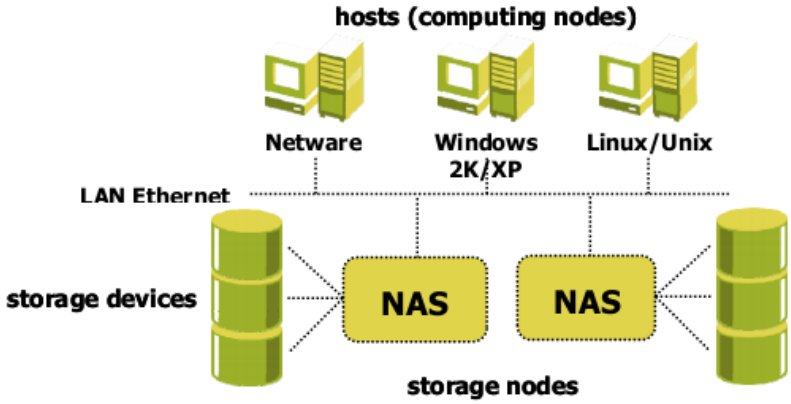

Figura 15: Schema di rete dell'architettura NAS

3.3 Storage Area Network

Parliamo di una rete dedicata all'accesso ai dischi. Di solito la comunicazione in questa rete avviene attraverso fibra ottica. Come front-end per gli host si può utilizzare un interfaccia NAS in modo da rendere ancora più trasparente il sistema di storage come in figura **??** Il vantaggio di questo tipo di rete è la grande scalabilità in quanto basta aggiungere dei device alla sottorete SAN per aumentare la capacità dell'intero sistema senza nessun'altra operazione.

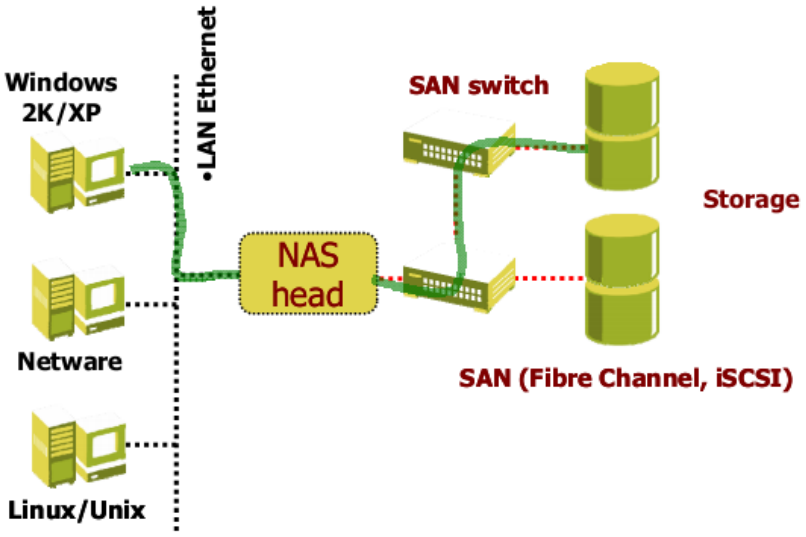

Figura 16: Schema di rete dell'architettura SAN con front-end NAS

Fiber Channel

Solitamente tutta la sottorete SAN è cablata in fibra ottica. In quanto un eventuale cablaggio ethernet implicherebbe numerosi problemi, il primo e più importante è quello dell'overhead introdotto dal protocollo TCP/IP a numerosi livelli, per manipolare le trame nella rete o per instradarle attraverso i vari indirizzi ip. Tutte queste informazioni sono inutili in un contesto di storage perciò è preferibile utilizzare un protocollo più performante come il *Fiber Channel*. Il Fiber Channel unisce tutte le caratteristiche migliori per un protocollo mirato allo scambio di grandi quantità di dati. Alta velocità nello scambio di dati, alta flessibilità e la capacità di realizzare reti estese. Le caratteristiche principali sono:

- Collegamenti full-duplex
- Throughput di 1600 Mbps
- Supporto per connessioni fino a 10 km
- Connettori piccoli
- Uso di componenti standard