

SISTEMI OPERATIVI “OPEN SOURCE”: IL CASO LINUX



In questo articolo vengono presentate le caratteristiche di **LINUX** dalla sua storia, all'evoluzione fino all'affermarsi all'interno del più generale fenomeno del software “*Open Source*”. Si dimostra come LINUX sia già pronto per il mercato e come permetta la realizzazione di soluzioni di elevata affidabilità, prestazioni, scalabilità, flessibilità, sicurezza ed economicità. A tale riguardo si analizzano ed approfondiscono le soluzioni basate su **CLUSTER LINUX** che si stanno affermando sul mercato.

Giampaolo Amadori
Gianfranco Bazzigaluppi
Walter Bernocchi
Mauro Gatti

1. BREVE STORIA DEI SISTEMI OPERATIVI CON PARTICOLARE RIFERIMENTO ALLO UNIX

In origine il termine “software” (SW) veniva usato per identificare quelle parti di un sistema di calcolo che fossero “modificabili” liberamente tramite differenti configurazioni di cavi e spinotti, diversamente dall’“hardware” (HW) con cui si identificava la componente elettronica. Successivamente all’introduzione di mezzi linguistici atti ad alterare il comportamento dei computer, SW prese il significato di “espressione in linguaggio convenzionale che descrive e controlla il comportamento della macchina”. In particolare occorre notare che ogni macchina aveva un suo proprio linguaggio convenzionale, comunemente detto *Assembler*.

Nella relativamente breve storia della “Information Technology”, tutte le società produttrici di HW hanno cominciato a fornire, assieme all’HW della macchina un certo numero di programmi in grado di svolgere le funzioni di base. Inizialmente si parlava di *Monitor*, poi di *Supervisor*, infine è stato adottato il nome

di “Sistema Operativo”. Tali programmi erano scritti sempre in *Assembler*, cioè nel linguaggio specifico della singola macchina.

Una eccezione a tale regola è stata il sistema operativo UNIX, realizzato da una organizzazione che non produceva HW, e precisamente i *Laboratori Bell*, e che pertanto fu progettato fin dall’inizio per risultare indipendente dalla specifica piattaforma HW su cui era stato inizialmente scritto. Non solo, i progettisti iniziali dello UNIX, proprio per renderlo indipendente dalla piattaforma HW, svilupparono un linguaggio, conosciuto come “Linguaggio C”, che contiene costrutti della programmazione strutturata (come i cicli FOR e DO...WHILE, le clausole IF...THEN...ELSE, ecc.) oltre a permettere la gestione precisa delle posizioni di memoria, possibile con l’*Assembler*.

Un’altra peculiarità, è che, avendo allora (anni ‘70) la *Bell* una causa in corso per violazione della legge statunitense sui monopoli, lo UNIX veniva inizialmente distribuito corredato dei sorgenti. Questo permise ad altri di portare il sistema UNIX stesso su piattaforme diverse da quelle usate dai laboratori

Bell e ne causò una rapidissima diffusione fra le comunità dei ricercatori e degli sviluppatori. Non solo: la disponibilità dei sorgenti permetteva a chiunque di contribuire al miglioramento ed all'espansione delle funzioni del sistema operativo. In pratica si formò una comunità di sviluppatori volontari, in genere dell'ambiente universitario, ma non solo, che contribuì, in un modo di operare che potremmo definire "collaborativo", alla crescita dello UNIX.

Nel 1984 la Bell perse la causa e la proprietà dei laboratori UNIX passò all'AT&T, che iniziò a rivendicarne i diritti, cioè cominciò a chiedere delle royalty, ma soprattutto mise un freno alla distribuzione dei sorgenti. Ne seguì una battaglia per i diritti di proprietà dello UNIX che durò una decina d'anni. Un folto gruppo di costruttori di sistemi informatici costituirono la Open Software Foundation, con l'obiettivo di farla diventare la proprietaria del software di base comune a tutti e sul quale tutti avrebbero costruito il loro sistema UNIX. Il risultato fu che quell'unico UNIX divenne una molteplicità di Sistemi Operativi proprietari: AIX, Digital UNIX, HP-UX, Sinix, Irix, UNIX386, ecc.

1.1. Il progetto GNU e la Free Software Foundation

La principale conseguenza della scomparsa della Bell e della trasformazione di UNIX in tanti sistemi proprietari fu che la comunità di programmatori, formatasi spontaneamente in quegli anni si trovò impossibilitata a continuare a lavorare. Uno degli "hacker" più lungimiranti, Richard M. Stallman, ricercatore presso il Laboratorio di Intelligenza Artificiale dell' M.I.T, nel 1985 diede origine al progetto GNU ed alla Free Software Foundation.

Vorrei sottolineare come l'uso recente della parola "hacker" con il significato di "pirata informatico" è una deformazione dovuta ai mass media. I veri "hacker" rifiutano questo significato e continuano ad usare la parola intendendo "Qualcuno a cui piace programmare e gode nell'essere bravo a farlo" [9].

L'acronimo GNU significa "GNU is Not Unix", ad indicare che l'evoluzione che aveva seguito il sistema operativo UNIX era sbagliata.

La Free Software Foundation (FSF) si occupa e si occupa tuttora di eliminare le restri-

zioni sulla copia, sulla redistribuzione, sulla comprensione e sulla modifica dei programmi per computer, concentrandosi in particolare sullo sviluppo di nuovo software libero, inserendolo in un sistema coerente che possa eliminare il bisogno di utilizzare software proprietario.

Stallman si premurò anche di concepire un'infrastruttura legale entro cui il sistema GNU potesse prosperare. Introdusse la **licenza GPL**, GNU Public License o General Public License, che garantisce le seguenti libertà:

- libertà di eseguire il programma per qualunque scopo, senza limitazioni;
- libertà di adattare il programma alle proprie necessità (l'accesso ai sorgenti è condizione essenziale);
- libertà di copiare il programma senza limitazioni;
- libertà di distribuire le copie modificate (l'accesso ai sorgenti è condizione essenziale).

Oltre a questi diritti, stabilisce un dovere: tutte le modifiche a software licenziato con la GPL devono essere rilasciate con la stessa licenza. La GPL è quindi una licenza persistente.

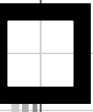
Il concetto di **Copyleft** - all rights reversed (gioco di parole con Copyright - all rights reserved) è stabilito nelle quattro libertà e nel dovere sanciti nella GPL e che vengono concessi (*left*, participio passato di leave).

1.2. Il kernel LINUX

La Free Software Foundation si occupò di riscrivere tutto il sistema UNIX per non dovere royalty a nessuno. Tutti i componenti venivano però innestati su un kernel UNIX, essendo il progetto del Kernel GNU (di nome Hurd) non ancora completato. A tutt'oggi il Kernel Hurd è ancora in fase di sviluppo.

Il 25 agosto 1991 uno studente finlandese, Linus Torvald, annunciò che stava lavorando ad un kernel UNIX-like ("non un progetto professionale come GNU" diceva il suo messaggio) e chiedeva suggerimenti su quali funzioni erano ritenute interessanti; e fornì quindi questo kernel, in seguito ribattezzato LINUX, alla Free Software Foundation.

Il **sistema GNU/Linux** (l'insieme del kernel Linux e delle componenti del progetto GNU) oggi è un sistema operativo completo, funziona su un numero notevolissimo di piat-



taforme hardware. Senza campagne pubblicitarie sponsorizzate da organizzazioni commerciali, ma solo per forza sua, è arrivato in pochi anni ad oltre dieci milioni di installazioni nel mondo. In particolare si stima che Linux sia il motore di oltre sei milioni di server Web in Internet.

1.3. Software Libero (Free software) ovvero "Open Source"

Un altro nome per il software libero è "Open Source". Questo termine è nato il 3 febbraio 1998 in una riunione a Palo Alto, California, a cui partecipavano Todd Anderson, Chris Peterson (del Foresight Institute), John "maddog" Hall and Larry Augustin (ambedue di Linux International), Sam Ockman (del Silicon Valley Linux User's Group), and Eric Raymond. Il concetto venne sviluppato durante una discussione sul doppio significato, in inglese, della parola "free": se usata nella frase "free beer tonight", significa "gratis", ma se usato come nel primo emendamento della Costituzione Americana con il concetto di "free speech right" allora significa "diritto alla libertà di parola".

Per evitare questa possibile confusione fu coniato il termine "Open Source". (La storia completa della nascita e della diffusione del termine si può trovare in [2] che, oltre a contenere la storia di OSI, intesa come Open Source Initiative, fornisce anche molti puntatori a documenti che riguardano la storia degli hacker e di Linux.)

1.3.1. IL SOFTWARE LIBERO È PIÙ AFFIDABILE E PIÙ EFFICIENTE

Il vantaggio principale del software libero non consiste, come invece molti vogliono credere, nel fatto che è a basso costo perché si può copiare senza limiti e senza royalty. Il vantaggio principale sta nella sua robustezza. Questa robustezza è dovuta al modo come viene sviluppato il software, che è diverso dal modo di sviluppo del software proprietario. I moduli del software libero vengono sviluppati da poche persone, spesso da una sola persona, ma poi, invece di essere protetti e tenuti nascosti, come accade per il software proprietario, vengono pubblicati in Internet, dove sono a disposizione di tutti i programmatori che desiderano rendersi utili verifican-

do software altrui. In pratica il software viene sottoposto a decine, centinaia di revisioni da parte di professionisti. Eventuali buchi o debolezze vengono evidenziati molto prima che possano emergere durante il funzionamento. Volendo banalizzare si tratta dell'applicazione del vecchio proverbio: "quattro occhi vedono meglio di due", che in questo caso diventa, in inglese, "Many eyes make all bugs shallow".

Per lo stesso motivo il software libero può essere anche più efficiente. Se un hacker, inteso come professionista del software, esaminando una porzione di codice si accorge che esiste un modo più efficiente di risolvere un problema, molto probabilmente lo comunicherà alla comunità degli sviluppatori, magari inviando la parte di codice già scritta. Alla lunga questo meccanismo spontaneo porta a software robusto ed efficiente.

Nel mondo del software libero inoltre il software viene rilasciato senza l'ansia di rispettare le scadenze di annuncio, tipiche del mondo del software proprietario. Non esiste un team di marketing che, avendo già annunciato delle nuove funzionalità, fa pressione sugli sviluppatori perché le date preannunciate siano rispettate. L'obiettivo è mettere a disposizione del software che funzioni bene, non del software che sia disponibile ad una data prefissata. Anzi, lo sviluppatore ha interesse a rilasciare del software che funzioni bene perché non c'è nessun guadagno a distribuire delle correzioni, a differenza di quanto invece avviene con gli "aggiornamenti" di software chiuso e proprietario.

1.3.2. IL SOFTWARE LIBERO È PIÙ SICURO

Nel software libero non possono esistere backdoor, cioè punti di ingresso conosciuti solo da chi ha sviluppato il codice. Essendo i sorgenti a disposizione di tutti, la presenza di una backdoor verrebbe segnalata in tempi brevi. Ovviamente, tutte le volte che si installa del software libero già compilato non ci può essere alcuna garanzia che uno sviluppatore malintenzionato non abbia inserito del codice non conforme ai sorgenti pubblici. Alla lunga però difetti di questo genere vengono scoperti e possono essere corretti, dato che il codice sorgente originale è sem-

pre a disposizione per essere ricompilato. È accaduto, per esempio, che le installazioni di un noto Data Base Relazionale proprietario contenessero un problema riguardante la sicurezza degli accessi al Data Base, perché banalmente gli sviluppatori avevano lasciato nel codice una backdoor di cui poi si erano dimenticati. Quando il Data Base in questione è diventato libero ed è stato messo a disposizione di tutti in formato sorgente, l'errore è stato scoperto e corretto nel giro di pochi mesi.

1.3.3. IL SOFTWARE LIBERO GARANTISCE LA SCELTA DEL FORNITORE

Al di là delle questioni tecniche finora elencate, un altro vantaggio tipico del software libero sta nella libertà di scelta: il cliente può davvero liberamente scegliere il fornitore di software o di servizi migliore disponibile sul mercato.

Forse paradossalmente, la massima libertà di scelta favorisce anche i fornitori stessi. Di primo acchito sembra che con il software libero per un cliente sia più semplice abbandonare un fornitore per un altro, ma questo significa anche che il cliente non è spaventato a dover lavorare con una piccola ditta che egli teme possa sparire in cinque o dieci anni. Molti piccoli sviluppatori software temono di perdere i loro già piccoli guadagni se sviluppassero software libero. In realtà molti programmatori e case di sviluppo vivono dignitosamente scrivendo e vendendo software libero, in un mercato in cui le loro scelte sono realmente libere.

Lo stesso non si può dire di chi basa i suoi programmi su piattaforme software proprietarie: devono costantemente subire "scelte di mercato" prese da altri, spendendo moltissima parte di quanto guadagnano solo per aggiornamenti di licenze per software che non fa quanto promesso dalle brochure pubblicitarie.

1.3.4. IL SOFTWARE LIBERO ABBASSA LA BARRIERA ALL'INGRESSO

Il software libero per sua natura abbassa le barriere di ingresso al mercato e questo può certamente essere salutato come un lieto evento da parte degli sviluppatori e degli utenti. Gli utenti avranno in tale modo la pos-

sibilità di ridurre i costi fissi legati alla loro soluzione salvaguardando quindi una maggiore liquidità all'acquisto di servizi per lo sviluppo di nuove soluzioni. Il fiorire di società che stanno migrando o sviluppando applicazioni su Linux così come l'affermarsi di palmari basati sul kernel Linux lo dimostra. Le imprese libere sono compensate con vendite e profitti per aver legalmente e correttamente soddisfatto gli acquirenti. Il software libero in quanto tale in realtà viene venduto a prezzi talmente bassi che i guadagni più significativi vengono fatti vendendo servizi e assistenza o soluzioni. RedHat, Suse, Caldera, TurboLinux e Mandrake sono alcuni esempi di aziende che si guadagnano da vivere con il software libero.

L'alternativa al software libero è la sorveglianza, ovvero assicurarsi che il software non sia usato o copiato illegalmente. Parlando in generale, la parola "sorveglianza" non è usata - si sente invece parlare di "controllo licenze" o frasi simili.

1.4. L'affermarsi di LINUX

Oggigiorno tutte le maggiori società di analisi e consulenza del mercato della Information Technology (IT) riconoscono il ruolo di primaria importanza già acquisito da Linux e pure ne prevedono una sempre maggiore affermazione nei prossimi anni. Al riguardo, una delle maggiori società di analisi del mercato IT, nel luglio 2001, sottolineava come LINUX già rappresentasse con ben il 26,9% di diffusione quale OS utilizzato sui nuovi ambienti server installati nel corso dell'anno 2000 il secondo OS più diffuso (Tabella 1).

Le cifre relative alle spedizioni sono, ovviamente, da intendersi in migliaia di unità.

È importante poi notare come il tasso di crescita annuale previsto per Linux sia notevolmente superiore a quello di qualsiasi altro OS e pertanto una sua sempre maggiore diffusione e rilevanza nei prossimi anni risulta cosa certa.

Per quanto riguarda poi le aree applicative per le quali Linux risulta essere maggiormente utilizzato queste sono (in ordine decrescente per quanto concerne la odierna diffusione): l'area dei Web Server, gli e-mail server, i Network server, i Firewall, i Web Application Server, l'area dello sviluppo delle

Piattaforma SW	1999 Spedizioni	1999 Share (%)	2000 Spedizioni	2000 Share (%)	1999-2000 Crescita (%)
Windows NT/2000	2.086	38,4	2.508	40,9	20,2
Linux	1.322	24,3	1.645	26,9	24,4
Netware 3.x,4.x,5.x	1.064	16,9	1.030	16,8	-3,1
Combined Unix	826	15,2	826	13,5	0
Other NOS	140	2,6	116	1,9	-17,4
Total	5.437	100	6.125	100	12,6

TABELLA 1

Confronto tra sistemi operativi in funzione del numero di Server installati (Fonte: IDC)

applicazioni, l'area dei DataBase Server, del Workgroup ed infine quella dei Transaction Server.

Riassumendo, LINUX si è da sempre più diffuso in nuove aree applicative in linea con la sua crescente evoluzione e crescita tecnologica; ovvero, non appena Linux è risultato sufficientemente maturo per essere utilizzato per le applicazioni legate al mondo di internet/intranet è stato da subito usato per esse ed ora che si è ancora più consolidato tecnicamente è pronto per svolgere anche funzioni di DB Server e Transaction Server e pertanto inizia ad essere ora molto usato anche per tali aree applicative.

1.5. Fattori che favoriscono e fattori che contrastano il rapido affermarsi di LINUX

Analisi di mercato hanno mostrato come i fattori che maggiormente stanno contribuendo all'affermarsi di Linux siano: il basso costo iniziale, la sua notevole affidabilità e robustezza, il basso costo a regime, il fatto che rappresenti una alternativa a mondi proprietari, la sua sempre crescente scalabilità, la sua portabilità fra piattaforme HW diverse, la crescente disponibilità di applicazioni ed il fatto che sia open source.

Le stesse analisi di mercato hanno invece evidenziato come i fattori che contrastano l'affermarsi di Linux siano: la mancanza di società che ne assicurino servizio e supporto, la mancanza di skill presso i clienti, la mancanza di applicazioni, il fatto che alcune applicazioni ritenute importanti da parte dei clienti debbano essere ancora migrate a tale piattaforma e la mancanza di cultura e conoscenza da parte del mercato.

1.6. Le iniziative di IBM atte a supportare l'utilizzo di LINUX

L'IBM ha fatto proprie le analisi e soprattutto le debolezze sintetizzate ai punti precedenti ed avendo riconosciuto in Linux un OS già adatto oggi per un utilizzo sempre più significativo da parte delle aziende ha deciso, ad inizio 2001, di investire un miliardo di dollari in tale area.

Con tale importante investimento l'IBM ha focalizzato le seguenti aree:

- tutte le piattaforme Server sono state rese in grado di supportare Linux offrendo così ai clienti delle "value proposition" ed opportunità rivoluzionarie assolutamente inimmaginabili fino a solo un anno fa;

- è stata intrapresa la migrazione di moltissimi dei prodotti SW IBM (quale DB2, Websphere, Domino, Tivoli, ecc.) su Linux per le diverse piattaforme HW;

- è stato costituito un cosiddetto Linux Technology Center per il quale lavorano a tempo pieno oltre 250 programmatori IBM e che ha la missione di lavorare assieme alla "Linux Community" per accelerare la maturazione di Linux attraverso lo sviluppo di utilities, tools, codice, ecc.;

- il supporto degli "Open Source Development Laboratories" assieme ad altri leader del mercato IT;

- la costituzione di 11 Centri a livello mondiale che sono in grado di aiutare sia Independent Software Vendor (ISV) sia clienti che volessero fare il porting su Linux di applicazioni attualmente utilizzate su altri OS;

- la costituzione ed il mantenimento di diversi siti web ai quali gli utenti possono collegarsi per ricevere una informazione puntualmente aggiornata relativamente al mondo Linux, al-

le applicazioni disponibili, ai tool a disposizione per gli sviluppatori, ecc.;

I ed infine, l'IBM ha iniziato a fornire con proprio personale una notevolissima quantità di Servizi per i clienti quali servizi di supporto di base ed avanzato, servizi di formazione, di consulenza, di implementazione e di sviluppo di applicazioni.

Per maggiori informazioni ci si può riferire al sito web <http://www.ibm.com/linux> [5].

1.7. Il ruolo delle Distribuzioni

Sebbene Linux sia liberamente disponibile e scaricabile attraverso Internet, lo scaricare il codice sorgente ed il ricompilare lo stesso al fine di farlo funzionare correttamente sulla piattaforma HW posseduta non è una attività così semplice da essere alla portata di tutti. Per tale motivo un discreto numero di società ha iniziato a distribuire Linux. Tali società non cambiano il sistema operativo, che è protetto dalla GPL ma si fanno bensì pagare perché portano Linux sul mercato sotto forma di un set consistente di CD, Manuali e pure assicurano il supporto all'installazione ed all'utilizzo di Linux. IBM ha sottoscritto partnership mondiali con le maggiori società che distribuiscono Linux (quali Red Hat, Suse, Caldera e TurboLinux) e pure alcune partnership locali (con Mandrake, RedFlag, ecc.).

1.8. LINUX: aree di utilizzo

Al paragrafo 1.4 abbiamo già velocemente analizzato le aree applicative per le quali Linux risulta essere oggi maggiormente utilizzato. Tali aree applicative possono essere sintetizzate in "quattro Modelli Infrastrutturali ed Applicativi" per i quali Linux può costituire una soluzione veramente innovativa.

Questi quattro modelli sono:

I Appliances: ovvero i sistemi dedicati allo svolgimento ottimale di una specifica funzione quali i sistemi che fungono da firewall, web-server, web application server, ecc.;

I Distributed Enterprise: Linux è un sistema robusto, affidabile, liberamente distribuibile ed economico ed è quindi perfetto per le aziende che hanno decine, centinaia o migliaia di sedi dove debbono utilizzare le stesse applicazioni con sicurezza, controllo ed in maniera economicamente efficace;

I Linux Cluster: ovvero insiemi di server che vengono collegati fra loro al fine di realizzare delle soluzioni infrastrutturali in grado di offrire alte prestazioni e/o superiori livelli di sicurezza a condizioni economicamente interessantissime e *dei quali parleremo più in dettaglio anche nei paragrafi successivi*;

I Workload Consolidation: ovvero il processo di razionalizzazione e riduzione del fenomeno di selvaggia ed incontrollata proliferazione di decine, centinaia e addirittura migliaia di server che sta oggi colpendo e mettendo in crisi la infrastruttura IT di tante aziende e Service Provider.

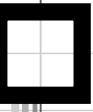
Grazie al fatto che oggi Linux può girare eccezionalmente bene ed assai efficacemente anche sui più grossi server IBM delle famiglie zSeries, iSeries e pSeries, e grazie all'estrema possibilità di portare e migrare applicazioni dai mondi proprietari verso il mondo Linux, i clienti possono pensare di "consolidare" centinaia di server su di un solo sistema con enormi vantaggi dal punto di vista del "Total Cost of Ownership" della soluzione, maggiore semplicità gestionale e riduzione delle risorse richieste.

2. CLUSTER LINUX: DESCRIZIONE GENERALE

Kai Hwang e Zhiwei Xu nel loro libro *Scalable Parallel Computing* [4], definiscono i *cluster* come insiemi interconnessi di interi computer (*nodì*) che lavorano congiuntamente come un singolo sistema (*single system image*) per fornire un servizio continuativo (*availability*) e servizi efficienti (*performance*). I cluster il cui scopo prioritario è quello di garantire la disponibilità del servizio sono noti come *HA cluster*, mentre quelli il cui scopo prioritario è di garantire alte prestazioni sono noti come *HPC cluster*. Nel seguito dedicheremo un paragrafo agli HA cluster e uno agli HPC cluster. In entrambi i casi ci limiteremo a considerare i sistemi basati sul sistema operativo Linux.

2.1. CLUSTER Linux per la realizzazione di sistemi e soluzioni Altamente Affidabili

Nel disegno di una soluzione ciò che l'architetto deve considerare prioritario, al fine di soddisfare le esigenze dell'azienda che



adotta quella soluzione, è la *realizzazione di un sistema che eroghi il servizio con prestazioni adeguate, nel momento in cui questo servizio effettivamente è richiesto, ad un costo il più piccolo possibile. Prestazioni adeguate* significa che, salvo alcune limitate eccezioni, le aziende non sono interessate ad avere il massimo delle prestazioni possibili, ma solo ad avere prestazioni adeguate alle loro esigenze. Per esempio, se il sistema ERP aziendale ha tempi di risposta per l'attività di inserimento ordini dell'ordine di 1 o 2 secondi, ben difficilmente i sistemi informativi saranno disposti ad investire ingenti somme di denaro per dimezzare il tempo di risposta. *Servizio effettivamente richiesto* significa che, anche se i sistemi informativi gradirebbero avere un sistema che non si guasti mai e non richieda alcuna interruzione di servizio per operazioni di manutenzione, questo non è certo l'obiettivo primario.

In effetti gli obiettivi sono, in ordine decrescente di importanza:

- 1. minimizzare il costo (danno) dovuto a interruzioni dell'erogazione del servizio:** ciò significa che le interruzioni pianificate di servizio sono considerate accettabili, benché ovviamente non siano auspicabili;
- 2. minimizzare il numero e la lunghezza delle interruzioni del servizio:** anche se le interruzioni di servizio pianificate non hanno un impatto grande come quelle non pianificate, nondimeno creano problemi di gestione, hanno comunque un impatto negativo sull'utenza se il sistema lavora 24 ore al giorno, possono ridurre le performance (buffering subottimale) e richiedono il riavvio dei processi di elaborazione batch;
- 3. minimizzare il numero dei guasti:** anche se con tecniche di mascheramento è possibile far sì che il guasto non produca un'interruzione di servizio, le componenti guaste devono però essere sostituite, con un conseguente incremento di lavoro per i sistemi informativi.

Notiamo che mentre l'obiettivo 1 è di importanza cruciale sia per gli utenti che per i sistemi informativi, l'obiettivo 2 è maggiormente importante per i sistemi informativi e l'obiettivo 3 è esclusivamente rilevante per i sistemi informativi.

Nel seguito del paragrafo faremo vedere come, facendo leva sulle caratteristiche avanzate della piattaforma IBM eServer xSeries e sulla robustezza del sistema operativo Linux, con un accurato disegno della soluzione sia possibile soddisfare la grande maggioranza delle esigenze aziendali per mezzo del sistema operativo Linux installato su server eServer xSeries. Vedremo in particolare come l'uso di tecnologie di clustering permetta di raggiungere livelli di disponibilità del servizio potenzialmente superiori al 99.9%.

2.1.1. EVITARE I GUASTI

Per evitare o minimizzare i guasti è necessario che il disegno di tutte le componenti della soluzione, separatamente prese, sia finalizzato all'obiettivo di costruire componenti affidabili, ma anche e soprattutto che nella fase di disegno si tenga esplicitamente conto delle complesse interazioni fra componenti durante l'esecuzione dei processi elaborativi. Quanto detto concerne ovviamente sia le componenti HW che le componenti SW. L'affidabilità del kernel Linux è stata soggetta a diversi studi che ne hanno dimostrato una stabilità paragonabile a quella dei più blasonati sistemi Unix proprietari. L'affidabilità complessiva del sistema operativo Linux su piattaforma Intel, ossia della combinazione del kernel Linux con il SW fornito a corredo dalle distribuzioni e della piattaforma Intel sottostante, è stato oggetto di un recente studio di D.H. Brown. La classificazione di D. H. Brown delle distribuzioni Linux in comparazione ai principali sistemi Unix relativamente al criterio RAS (acronimo che sta per le tre parole Reliability, Availability e Serviceability), ha visto prevalere SuSE Linux 7.2 sulle altre distribuzioni e ha evidenziato alcuni punti deboli di Linux, o più precisamente di Linux su piattaforma Intel. Va detto però che alcuni di questi limiti sono in fase avanzata di soluzione, quale per esempio il multi-path I/O, mentre altri sono in via di soluzione con l'arrivo dei primi server basati sull'IBM Enterprise X-Architecture (EXA).

Il problema estremamente complesso della qualità del SW e dei processi di debugging, verifica e test attraverso i quali si riduce la percentuale di difetti presenti nel SW non sa-

ranno esaminati in questo articolo. Un'analisi approfondita e aggiornata di questo argomento è contenuta nel numero monografico dell'IBM System Journal su *Software Testing and Verification* [3].

2.1.2. EVITARE LE INTERRUZIONI DI SERVIZIO

Evitare o minimizzare le interruzioni di servizio in presenza di guasti è possibile tramite tecniche di *mascheramento*. Un tipico esempio è l'uso di un *fault-tolerant team* di schede di rete. Se una scheda si rompe, o se si rompe lo switch al quale la scheda è connessa, viene attivata, in modo trasparente per gli applicativi, la scheda di backup. Le tecniche di mascheramento sono ampiamente utilizzate nel mondo Intel per dischi (RAID), schede di rete (team fault tolerant), ventole di raffreddamento, alimentatori. Fino a qualche mese fa rimaneva problematico la protezione attraverso ridondanza di CPU, memoria e sottosistema di I/O. La ridondanza nel sottosistema di I/O (multipath I/O) è ora possibile con il nuovo driver delle schede Qlogic. La protezione della memoria sarà possibile a breve con l'arrivo dei primi eServer xSeries che supportano il memory mirroring con sostituibilità a caldo dei banchi di memoria, una delle pietre miliari dell'Enterprise X-Architecture. Rimane come ultimo e più difficile ostacolo la ridondanza con sostituibilità a caldo delle CPU. Attualmente tutto ciò che si riesce a fare in caso di blocco di una CPU è il riavvio del server con successivo isolamento della CPU guasta e ripresa dell'attività sfruttando le altre CPU (in un sistema multiprocessore).

2.1.3. EVITARE LE INTERRUZIONI DI SERVIZIO QUANDO IL SISTEMA È UTILIZZATO

La trasformazione del tempo di arresto (downtime) da non pianificato in pianificato è di grande utilità per la grande maggioranza dei sistemi; fanno eccezione ovviamente i sistemi che devono lavorare 24 ore al giorno per 365 giorni all'anno. Fra le tecniche più promettenti per raggiungere questo scopo vale la pena citare la *Software Rejuvenation*. Numerosi studi sviluppati a partire dalle metà degli anni '90 hanno evidenziato che molti blocchi di sistemi sono dovuti a errori di programmazione che provocano durante l'e-

secuzione dei programmi fenomeni quali *memory leaks*, *unterminated threads*, *memory bloating*, ecc. In linea di principio è ovvio aspettarsi che il miglior modo per affrontare questi problemi sia definire migliori tecniche di controllo del SW e, in caso di problema, di identificare la componente malfunzionante per poterla riparare. Vi sono varie ragioni per le quali questa non è, quanto meno, l'unica via per affrontare il problema. Da un lato se il baco risiede in un SW proprietario per ottenere una *patch* che corregga l'errore può essere necessario aspettare mesi, e questa è in effetti una delle ragioni del successo del modello di sviluppo Open Source. D'altro canto alcuni degli errori possono essere così complessi da rendere l'individuazione della causa estremamente difficile (i cosiddetti Heisenbugs). In tutti questi casi è di grande beneficio per il sistema adottare tecniche di Software Rejuvenation, ossia di riavvio controllato di singoli gruppi di processi o di tutto il sistema operativo. Anche se il riavvio controllato è una prassi che molti amministratori di sistema applicano da molto tempo, la determinazione del momento ottimale per effettuare il riavvio e l'individuazione di metodiche che permettano di prevedere il presentarsi di guasti, sono problemi tutt'altro che banali che sono stati oggetto di numerosi studi sviluppati congiuntamente da IBM e dalla Duke University. Sulla base di questi studi, basati su sofisticate tecniche matematiche (*Stochastic Reward Nets*), è stato sviluppato il modulo del SW IBM Director, offerto gratuitamente su tutti gli eServer xSeries, che implementa il processo di SW Rejuvenation.

2.1.4. COME GESTIRE I GUASTI

Se non si riesce ad evitare il guasto e soprattutto se non si riesce ad evitare che il guasto si presenti nel momento in cui sistema è utilizzato, non resta che disporre di metodiche che rendano minimo il tempo che intercorre fra il guasto e la riparazione del sistema. Vari accorgimenti aiutano a ridurre il tempo necessario per la diagnosi e quello necessario per la riparazione. Ma sia la criticità sempre crescente dei sistemi informativi che la complessità degli stessi, stanno spingendo il mercato all'adozione sempre più ampia di meccanismi che automatizzino il processo di

ripristino del servizio. Rientrano in questa categoria in particolare le metodologie di replicazione e quelle di clustering. Le tecniche di replicazione, utilizzate in file system distribuiti come il GPFS di IBM e in tutti i maggiori DBMS, giocano un ruolo molto importante, ma per contenere l'esposizione non saranno analizzate in questa sede. Affronteremo invece i cluster per alta affidabilità.

I cluster per alta affidabilità possono essere classificati secondo molti punti di vista. Per presentarne le caratteristiche elencheremo nel seguito alcuni di questi punti di vista e faremo vedere come alcuni di questi cluster si comportano relativamente a questi. La trattazione ovviamente non ha alcuna pretesa di esaustività. Il lettore potrà trovare informazioni più complete nel sito <http://linux-ha.org> [1].

L'accesso ai dati

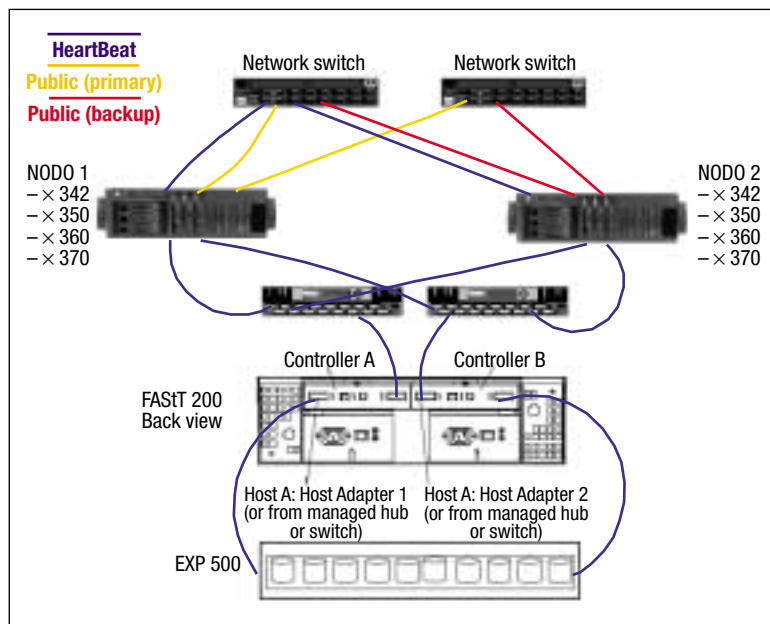
Vi sono sostanzialmente tre categorie di cluster:

- I cluster basati sulla totale replicazione dei dati (o clonazione dei server) quali per esempio TurboCluster della TurboLinux o LifeKeeper Data Replication della SteelEye Technologies;

- I cluster basati sulla condivisione dei dischi a livello HW ma non a livello SW (*shared nothing*) quali per esempio SteelEye LifeKeeper e Mission Critical DataGuard;

- I cluster basati sulla condivisione dei dischi sia a livello HW che a livello SW (*shared everything*) come per esempio Oracle 9i Real Application Cluster.

La prima categoria di cluster è prevalentemente utilizzata per le web farm o per altre attività a basso carico transazionale. La seconda categoria di cluster è quella con il più ampio spettro di applicazioni. Infatti, può essere utilizzata per proteggere virtualmente ogni applicativo: web server, database server, application server, ecc. La terza categoria di cluster richiede lo sviluppo di un complesso meccanismo per evitare danni all'integrità dei dati dovuti all'accesso simultaneo di più nodi, ossia di un distributed lock manager; per questa ragione non ha mai raggiunto un elevato livello di diffusione. Nel seguito ci concentreremo prevalentemente sulla seconda categoria.



Il sottosistema dischi condiviso

I nodi possono accedere al sottosistema dischi condiviso per mezzo del protocollo SCSI, del protocollo FC (SAN) o attraverso il protocollo IP (NAS). Di questi il protocollo FC è quello che garantisce la migliore combinazione di affidabilità e performance. Per questa ragione la grande maggioranza dei cluster per alta affidabilità, quantomeno per i server basati su piattaforma Intel, utilizzano la tecnologia FC. I NAS sono un'alternativa interessante per sistemi che non abbiano le esigenze di performance e integrità dei dati che sono tipiche dei sistemi transazionali. La figura 1 rappresenta un cluster Linux per alta affidabilità basato su IBM eServer xSeries e storage IBM FAST200.

Le distribuzioni

Un aspetto fondamentale è l'integrazione con le distribuzioni. La tabella 2 fornisce un elenco sintetico delle distribuzioni supportate.

Gli applicativi

In linea di principio un cluster per alta affidabilità *shared nothing* può proteggere ogni applicativo "che si comporti bene" ossia che sia in grado di ripristinare la propria attività in seguito ad un'interruzione irregolare. Per esempio un database transazionale può essere protetto con questa tecnica perché il DBMS riporta il sistema in condizioni di inte-

FIGURA 1

Cluster HA linux con IBM eServer xSeries e storage eServer FAST200

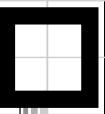


TABELLA 2
Distribuzione supportata dai Software

SW	Distribuzioni supportate
SteelEye LifeKeeper v. 4.0	Red Hat 6.x and 7.x, SuSE 7.x, Caldera eServer 2.3, Miracle Linux 1.1 and 2.x e TurboLinux (vedi http://www.steeleye.com/products/linux/system_conf.html per un elenco aggiornato)
Mission Critical Convolo Data Guard	Red Hat Linux 6.2, 7.1, SuSE Linux 7.1 e Debian GNU/Linux 2.2r3 (vedi http://www.mclinux.com/products/convolo-requirements.php per un elenco aggiornato)

grità sfruttando i log. Accanto al problema di quali applicativi possono essere protetti vi è il problema di come proteggere gli applicativi. In generale i SW di clustering forniscono meccanismi generici per controllare lo stato dei processi, spegnerli in caso di malfunzionamento e riavviarli su un altro nodo. L'implementazione specifica dei meccanismi di controllo, avvio e spegnimento dei processi è però demandata ad opportuni script. SteelEye Technologies vende degli Application Recovery Kit che possono essere utilizzati per agevolare il processo di configurazione di applicativi in cluster e un Software Development Kit per consentire lo sviluppo di opportuni script per applicativi per i quali non è disponibile un Application Recovery Kit. L'elenco degli ARK disponibili presso SteelEye Technologies è contenuto nella seguente pagina web http://www.steeleye.com/products/supported_apps.html [7]. Mission Critical fornisce invece supporto per un limitato numero di applicazioni sotto forma di servizio. L'elenco degli applicativi supportati è disponibile all'URL <http://www.missioncriticallinux.com/support/supported-applications.php> [8].

2.1.5. UN ESEMPIO DI LINUX CLUSTER PER LA HIGH AVAILABILITY (HA)

IBM e Open4.it hanno realizzato un cluster HA e Load Balanced, basato su architettura Pentium, fiber channel e software OpenSource per Linux con le seguenti caratteristiche:

- 40 cpu pentium 4;
- 35 GB ram;
- 1 Tera di storage.

Il cluster è articolato in: una coppia di bastioni in HA, una coppia di balancer in HA, venti server in LB come farm, una coppia di database server in HA, una coppia di publishing ser-

ver in HA, una coppia di file server in HA, uno storage in fiber channel con controller ridondato, due switch fiber channel e due switch Catalyst (Cisco) di rete.

Tale cluster è attualmente in funzione presso Aonet in Milano, dove assicura la continuità dei servizi di web publishing, content management, ecommerce, messaging, groupware web e wap che vengono offerti in modalità ASP.

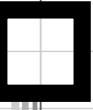
In particolare le applicazioni che vengono fatte girare sono: Apache, Interchange, Qmail, Courier, Twig, Zope, NFS e vario software aggiuntivo per sicurezza e intrusion detection

Secondo Maruzzelli (Resp. Architettura per Open4.it) "Le ragioni che hanno portato all'acquisto di tale cluster sono il vantaggio competitivo che si realizza da una installazione con cui riescono ad assicurare performance, sicurezza e continuità di servizio ad un gran numero di clienti, con costi decrescenti e con amministrazione unificata. Il vantaggio specifico dell'OpenSource e di Linux risiede invece nell'ampia comunità di sviluppatori software e amministratori che garantiscono un costante aggiornamento delle applicazioni sia in termini di funzionalità che, aspetto molto importante, di sicurezza".

2.2. CLUSTER LINUX per la realizzazione di sistemi di High Performance Computing(HPC)

I termini High Performance Computing e High Throughput Computing denotano un ampio gruppo di sistemi caratterizzati da grandi esigenze elaborative. Un elenco anche solo sommario dei settori che fanno uso di cluster per HPC richiederebbe molte pagine. Citiamo a puro titolo di esempio:

- fisica delle alte energie - per esempio il pro-



getto LHC del CERN per la ricerca dell'evanescente bosone di Higgs e la verifica delle teorie supersimmetriche;

■ prospezioni geologiche - particolarmente rilevante per le aziende del settore petrolifero;

■ studi geologici, in particolare per il problema della previsione dei terremoti;

■ Life Sciences - sia la genomica che la proteomica richiedono enormi potenze elaborative per analizzare Petabytes di dati;

■ previsioni meteorologiche - per analizzare modelli sempre più dettagliati dell'atmosfera;

■ Data mining - per analizzare gli enormi database generati dai sistemi di CRM al fine di evitare di perdere clienti e garantire un livello di servizio più elevato;

■ risk management - ad esempio per minimizzare i rischi operativi nei settori finanziari e lungo la supply chain;

■ analisi di portafoglio - particolarmente interessante per le società di intermediazione mobiliare che facciano uso di algoritmi statistici o di intelligenza artificiale al fine di prevedere l'andamento dei titoli e/o contenere il rischio di investimenti.

In moltissimi settori nei quali si fa uso di algoritmi per i quali la latenza della comunicazione fra i processi non è critica (*coarse grain parallelism*), i cluster sono il metodo preferenziale per la gestione di elaborazioni complesse. In particolare i cluster basati su Linux si stanno affermando come piattaforma preferenziale per l'High Throughput Computing. Le principali ragioni alla base di questa evoluzione sono:

■ il basso costo dei server basati su piattaforma Intel (per esempio gli IBM eServer xSeries);

■ l'accresciuta performance erogata dai server Intel, in particolare per i calcoli su interi e più recentemente con il rilascio dei Pentium IV anche per i calcoli in virgola mobile;

■ il basso costo del sistema operativo Linux;

■ la disponibilità del codice sorgente del sistema operativo Linux che permette di ottimizzarlo in funzione dell'esigenza; citiamo a titolo d'esempio la creazione di sistemi operativi con processi mobili (MOSIX, Qclusters OS);

■ la facilità con la quale gli esperti gestori dei sistemi HPC, che hanno prevalentemente una cultura UNIX, possono apprendere l'uso di un sistema Linux;

■ le ottime caratteristiche del kernel Linux,

particolarmente a partire dal rilascio del kernel della famiglia 2.4.x.

Notiamo altresì che i tentativi fatti di utilizzare Windows NT/2000 in luogo di Linux non hanno avuto molto successo. Le principali ragioni sono:

■ costo del sistema operativo;

■ difficoltà di ottimizzazione data l'indisponibilità del codice sorgente;

■ sistema di gestione meno flessibile.

2.2.1. BEOWULF: UNA DELLE ARCHITETTURE DEI LINUX CLUSTER

Introduciamo ora una delle architetture maggiormente utilizzate per la realizzazione di Linux cluster per HPC, dalla definizione di T. Sterling e Don Becker in "How to Build a Beowulf": *"un Beowulf è un gruppo di personal computers (PCs), interconnessi da una qualsiasi tecnologia di rete, su cui girano uno dei sistemi operativi open source con caratteristiche proprie dello Unix"*. A titolo esemplificativo, un cluster Beowulf potrebbe essere composto da semplicissimi PC, uno switch di rete che riconosca il protocollo TCP/IP e Linux come sistema operativo. Chiunque abbia una discreta conoscenza della tecnologia Intel, del sistema operativo Linux, e del protocollo TCP/IP può realizzare un Linux cluster.

I server e l'infrastruttura di rete sono però, per quanto importanti, solo due delle componenti di un cluster Linux per HPC. Occorre infatti in primo luogo un applicativo "parallelo", ossia in grado di sfruttare il parallelismo del sistema HW. Inoltre occorre un sistema di gestione efficiente, poiché fino a quando il numero di nodi è piccolo, la manutenzione cluster non è molto difficoltosa, ma quando il numero dei nodi diviene elevato il sistema diventa rapidamente ingestibile in mancanza di un sistema gestione efficiente. In particolare in virtù del progressivo aumento del numero di guasti al crescere del numero di nodi. A tal proposito esistono vari tool sia commerciali che non commerciali che aiutano i system administrator del cluster, per una descrizione di questi tools e delle componenti che fanno parte di un Linux cluster per HPC rimandiamo alla lettura dell'IBM redbook SG24-6041-00 scaricabile all'indirizzo <http://www.redbooks.ibm.com> [6].

2.2.2. LE COMPONENTI DI LINUX CLUSTER PER HPC

In termini generali un cluster Linux per High Performance Computing è costituito da

- nodi di elaborazione (per esempio eServer xSeries 330);
- un architettura di rete ad alte prestazioni per la comunicazione fra i nodi (per esempio basata su schede di rete e switch Myrinet) indicata spesso con l'espressione *system area network*;
- un sistema per l'immagazzinamento dei dati;
- sistema operativo Linux (spesso basato sulla distribuzione RedHat);
- compilatori paralleli (Portland Group Fortran F90, C, C++, GNU compilers, Intel F90 compiler);
- librerie parallele (per esempio MPI o PVM);
- Job Management System;
- un sistema di gestione integrato del cluster. Il tipici nodi utilizzati in un cluster Linux per HPC sono
 - server monoprocesori basati sull'architettura IA32 (per esempio eServer xSeries 330);
 - server biprocessori basati sull'architettura IA32 (per esempio eServer xSeries 330);
 - server quadriprocessori basati sull'architettura IA64 (per esempio eServer xSeries 380).

La scelta del tipo di server dipende essenzialmente dal tipo di applicativo, da considerazioni di costo e da valutazioni sulla complessità di gestione. Notiamo altresì che mentre Linux supporta teoricamente fino a un massimo di 16 processori, il grado di scalabilità al di sopra di 8 processori è tutt'altro soddisfacente con gli attuali kernel. Viceversa, i sistemi basati sull'architettura IA64 rendono possibile l'indirizzamento di grandi quantità di RAM, ma sono attualmente limitati ad un massimo di 4 processori. Un altro punto che occorre tenere presente è che, in un cluster costituito da molti nodi, una percentuale significativa del costo complessivo è rappresentata dai dispositivi a corredo (rack, cablaggio, monitor, ecc.). Al fine di minimizzare questa componente sono utili tecnologie quali l'IBM C2T technology che può consentire una riduzione dei costi dell'ordine di decine di migliaia di dollari.

La *system area network* può essere basata su schede di rete e switch Fast Ethernet, se gli applicativi possono tollerare un'alta la-

tenza nella comunicazione fra processi, da schede e switch Gigabit Ethernet, se è necessaria una più grande larghezza di banda o infine da dispositivi Myrinet, se è fondamentale minimizzare la latenza nella comunicazione fra processi. È importante sottolineare in proposito che in un cluster costituito da decine di nodi l'adozione di un'infrastruttura di rete ad alte prestazioni può far sì che il costo della rete eguagli o addirittura superi quello dei server. Una chiara comprensione delle effettive esigenze elaborative è quindi cruciale per contenere i costi.

Il sistema per l'immagazzinamento dei dati può consistere semplicemente dei dischi contenuti nei server o alternativamente di uno o più file server. In questo secondo caso un ruolo particolarmente cruciale è svolto non solo dal tipo di file server, per esempio un IBM Enterprise Storage Server o piuttosto un IBM Network Attached Storage, ma anche dal tipo di cluster file system con il quale si accede a questi dati (per esempio l'IBM General Parallel File System).

Un Job Management System è costituito da tre parti principali:

- uno *user server* che consente agli utenti di sottomettere i job a uno o più code di elaborazione;
- un *job scheduler* che decide come e quando assegnare le risorse ai processi di elaborazione;
- un *resource manager* che effettua l'allocazione delle risorse e controlla l'esecuzione dei job.

Il sistema per la gestione integrata del cluster deve garantire per quanto possibile

- una gestione integrata del cluster che consenta di soddisfare il requisito della *Single System Image*;
- l'installazione automatizzata, controllata e a blocchi della server farm;
- garantire un'elevata fault tolerance.

Per quanto concerne le problematiche di gestione ci limitiamo a citare il prodotto open source xCAT sviluppato da IBM, nonché il blasonato IBM Cluster Management System, recentemente portato su Linux. Relativamente alla fault tolerance è evidente che mentre è cruciale per un cluster costituito da centinaia di nodi minimizzare il costo per nodo ed è spesso accettabile in ca-

so di rottura di un nodo il dover riavviare i job in esecuzione su quel nodo, non è però al tempo stesso accettabile avere un'alta difettosità. Se per esempio un job richiede settimane per essere eseguito, il doverlo rilanciare comporta un ritardo di settimane. Tecniche di Predictive Failure Analysis quali quelli disponibili sugli eServer xSeries possono essere d'aiuto per raggiungere questo obiettivo.

2.2.3. UN'ALTERNATIVA AVANZATA

Accanto alle metodologie tradizionali che prevedono lo sviluppo di applicativi paralleli per mezzo di librerie quali PVM e MPI si sta diffondendo in vari settori l'alternativa più avanzata di utilizzare la migrazione di processi per ottimizzare la distribuzione del carico fra i nodi di elaborazione. Particolarmente interessante nel mondo Linux è il MOSIX sviluppato presso l'Università di Gerusalemme dal Prof. Amnon Barak del quale è stata recentemente presentata una versione commerciale (Qlusters OS) dall'azienda israeliana Qlusters Inc. Quest'ultimo prodotto contiene due importanti componenti che mancano nella versione open source (MOSIX): uno scheduler e un queue manager.

2.2.4. UN ESEMPIO DI LINUX CLUSTER PER HPC

Un esempio di Linux cluster per HPC in Italia è quello che IBM ha realizzato per il Cineca di Bologna e che ha le caratteristiche riportate in tabella 3.

Tale cluster è oggi in funzione presso il Cineca di Bologna dove è utilizzato per fare girare codici di calcolo relativi alle seguenti aree: astrofisica, chimica computazionale e fisica dello stato solido.

Le ragioni per le quali il Cineca ha deciso di installare un tale HPC Cluster sono dovute alla convinzione che sia già oggi utile sperimentare le tecnologie di clustering Linux e Beowulf per il supercalcolo a conferma della crescente maturità raggiunta da Linux.

Ringraziamenti

Giunti al termine di tale articolo ci fa piacere il ricordare e ringraziare il Dr. Stefano Maffulli (Associazione Culturale MILUG di Milano) per il prezioso supporto fornitoci e per la passione con la quale pro-

Compute node Model:	IBM x330 servers
Processor (PE):	Intel Pentium III 1.133 GHz
Number of PEs:	128
Processor per node:	64 nodes with 2 proc.
L2 Cache:	512 kbit per processor full speed
DRAM:	64 Gbytes (1GB/node)
Disk space:	2.7 TB
Peak performance:	145 Gflop/s
OS:	Linux
Internal Network:	Miricom LAN Card "C" Version
Available compilers:	Portland Group Fortran F90, C, C++ GNU compilers Intel F90 compiler
Parallel libraries:	MPI

muove il software Open Source, il Dr. Erbacci (CINCA di Bologna) per il supporto offerto per quanto riguarda le informazioni relative alla installazione CINECA ed il Dr. Maruzzelli per le informazioni e considerazioni forniteci relativamente alla installazione Open4.it.

Vogliamo poi concludere con una esortazione ed un incitamento al guardare già oggi a LINUX come una possibilità reale che può offrire un vantaggio competitivo all'interno della vostra azienda, del vostro istituto, della vostra scuola perchè sicuramente vi sono già oggi moltissime aree della vostra infrastruttura informatica che potrebbero trarre dei vantaggi, a volte impensabili, dall'utilizzo di Linux e di altro codice Open Source.

Bibliografia

- [1] High-Availability Linux Project: [HTTP://WWW.LINUX-HA.ORG](http://www.LINUX-HA.ORG)
- [2] History of the OSI: <http://www.opensource.org/docs/history.html>
- [3] IBM: Software Testing and Verification. *IBM System Journal*, Vol. 41, n. 1, 2002. <http://www.research.ibm.com/journal/sj41-1.html>
- [4] Kai Hwang, Zhiwei Xu: *Scalable Parallel Computing: Technology, Architecture, Programming*. McGraw-Hill Companies, 1998. <http://shop.barnesandnoble.com/booksearch/results.asp?WRD=hwang+xu&userid=66GToH2D8T>
- [5] Linux at IBM: <http://www-1.ibm.com/linux>
- [6] RedBooks: <http://www.redbooks.ibm.com>

TABELLA 3

Caratteristiche di un cluster HPC

- [7] Supported Apps: http://www.steeleye.com/products/supported_apps.html
- [8] Supported Layered Products: <http://www.missioncriticallinux.com/support/supported-applications.php>
- [9] The GNU Project: <http://www.gnu.org/gnu/the-gnu-project.html>

GIAMPAOLO AMADORI ingegnere nucleare con specializzazione successiva in marketing, è responsabile IBM delle attività di Vendita e Marketing Linux per la Regione Sud Europea. In IBM, dal 1989, ha ricoperto diversi ruoli di sempre maggiore responsabilità quali: Responsabile attività di Marketing per il Mercato Italiano delle Small and Medium Businesses, Direttore Vendite Italia per i Midrange Server.
E-mail: giampaolo_amadori@it.ibm.com

GIANFRANCO BAZZIGALUPPI è Manager of University Relations di IBM Italia.
Esperto di economia e organizzazione, attivo nella ricerca (IreR; Centro Studi di IBM) e nella docenza uni-

versitaria, dal 1994 al febbraio 2001 è stato responsabile della Fondazione IBM Italia.
Giornalista pubblicista, ha diretto dal 1993 al 1999 la rivista If, edita da G. Mondadori.
E-mail: bazzi@it.ibm.com

WALTER BERNOCCHI è un Advisor IT Specialist e lavora in IBM Italia per il gruppo xSeries Pre-Sale Technical Support. Ha iniziato la sua collaborazione con IBM nel 1989. Da marzo 2000 lavora a tempo pieno per sviluppare e supportare soluzioni in ambiente Linux. Ha esperienza di C/C++, Java, SunOS, Solaris e AIX. E' Red Hat Certified Engineer (RHCE).
E-mail: walter_bernocchi@it.ibm.com

MAURO GATTI dottore in Fisica e in Ricerca in Fisica, è team leader dell'IBM EMEA eServer xSeries Solution Architects team. La sua area di specializzazione è il disegno infrastrutturale di e-business server farms con eServer xSeries. In particolare si occupa di progetti di sistemi altamente affidabili, Linux, Grid Computing e ERP. Ha esperienze come free lance per alcune grandi aziende dell'IT (Microsoft, HP, IBM).
E-mail: mauro_gatti@it.ibm.com