UNIVERSITÀ DEGLI STUDI DI BOLOGNA

FACOLTÀ DI INGEGNERIA
Corso di Laurea in Ingegneria Informatica
Laboratorio di Amministrazione di Sistemi L-A

Configurazione di una rete Linux Terminal Server con server basato su OpenMosix

Tesi di Laurea di:
Mario Di Bacco

Relatore:
Dott. Ing. Marco Prandini

Correlatore:
Ing. Francesco Benincasa

Anno Accademico 2005/2006 - Seconda sessione

Indice generale

1 La Rivoluzione Digitale e l'impatto sociale ed ambientale....7
1.1 Il Divario Digitale....9
1.1.1 Alcuni dati sul Divario Digitale....10
1.1.2 I Paesi più sviluppati ed il Divario Digitale....15
1.1.3 Accessibilità e consumismo informatico....16
1.2 Il problema dell'hardware obsoleto....17
1.2.1 L'apparato normativo....19
1.2.2 La politica delle aziende....19
1.2.3 Recupero, Riciclo e Smaltimento....21
2 Il Trashware e il Free Software....24
2.1 Free Software e Software Open Source....27
2.1.1 Modello di sviluppo....30
2.1.2 I governi e l'Open Source....33
2.1.3 L'impatto nei Paesi in via di sviluppo....34
2.2 Il Trashware in Italia....35
2.2.1 L' associazione RaccattaRAEE....37
2.2.2 Ingegneria Senza Frontiere....38
3 I Cluster....41
3.1 OpenMosix....43
3.1.1 Sottosistemi di openMosix....44
3.1.2 La gestione dell'I/O....47
3.1.3 La memoria condivisa in Linux....50
3.1.4 Migshm....51
3.2 LTSP....53
3.2.1 Teoria del funzionamento....56
3.3 Considerazioni sul bilancio energetico....63
3.3.1 Considerazioni rispetto alla collettività....64
3.3.2 Considerazioni economiche....65
4 Linux Terminal Clustered Server Project....68
4.1 Architettura e funzionamento di LTCSP....69
4.2 Scalabilità....71
4.3 Il Package di installazione....73
5 I Test....76
5.1 Rendering....77
5.2 Editing di immagini....78
5.3 Encoding....79

5.4 Word processing.....79
6 Conclusioni.....84
7 Appendici.....88
7.1 Appendice A, Installazione e configurazione di openMosix.....88
7.1.1 Monitoraggio del cluster.....91
7.1.2 Aggiunta di Migshm ad openMosix.....94
7.2 Appendice B, Installazione di LTSP.....96
7.2.1 Il sistema LTSP in dettaglio.....96
7.2.2 Maggiori dettagli su XDMCP.....102
7.2.3 Installazione di LTSP.....104
7.2.4 Configurazione dei terminali.....107
7.2.5 Startup del terminale.....108
7.3 Appendice C, Indice di sviluppo ORBICOM.....109
8 Ringraziamenti.....110
9 Bibliografia.....111

Introduzione

Il progetto di tesi qui presentato è nato da una collaborazione tra il nostro team e l'associazione Ingegneria Senza Frontiere Bologna (ISF). Il gruppo Informatica di ISF ha infatti offerto degli spunti di lavoro ai tesisti, raccolti con entusiasmo e motivazione, che si sono conciliati molto bene con l'attività di tesi, sia dal punto di vista della pertinenza scientifica che in quello dell'interesse accademico. ISF ha fornito inoltre il supporto tecnico e organizzativo. Il lavoro è stato svolto nell'Hacklab, dello spazio sociale autogestito XM24, a Bologna, che ha offerto attrezzature e competenze, mentre l'hardware è stato fornito dall'associazione RaccattaRAEE di Bologna, attiva nell'ambito del trashware, oltre che da ISF. Il progetto prende vita come un'evoluzione naturale del lavoro svolto dal gruppo Tecnologie dell'Informazione di ISF Roma, che aveva già verificato i vantaggi dell'uso di sistemi di clustering, sviluppati possibilmente su piattaforme libere. ISF Roma aveva mostrato la possibilità di usare tali sistemi anche su architetture povere.

Lo scopo del nostro lavoro è stato quello di progettare una configurazione ibrida di un sistema di clustering, che impiegasse al meglio i mezzi disponibili, facendo particolare attenzione ai vincoli imposti dal tipo di risorse impiegate. Molti sinteticamente infatti, il progetto vuole riuscire a utilizzare hardware datato e software libero (che bene si intersecano tra loro, da diversi punti di vista) per realizzare una rete di terminali diskless, il cui server è un cluster di

macchine. Per realizzare questa soluzione, è stato necessario ibridare due differenti sistemi di clustering: uno che permettesse di distribuire servizi su terminali diskless, e l'altro che rendesse possibile integrare tra loro le risorse di diverse macchine, per raggiungere la potenza di calcolo necessaria. Solo questa configurazione, che non necessita di mezzi avanzati per i terminali, né di un server singolo, necessariamente tecnologicamente avanzato, ci ha permesso di utilizzare hardware di penultima generazione, raccolto grazie all'attività di trashware (di cui si parlerà nel Capitolo 2), raggiungendo risultati considerevoli. Inoltre tutto il software utilizzato, in quanto free software, ci ha permesso di attingere a una documentazione dettagliata, e in ogni caso è l'unica soluzione che permette di non essere vincolati a un produttore (si veda il Capitolo 2.1). Il progetto infatti è concepito per essere fruito da realtà che possano far evolvere tale progetto autonomamente, in particolare i Paesi in via di sviluppo (questo aspetto del progetto è trattato largamente nel Capitolo 1, insieme a importanti considerazioni sull'impatto ecologico dell'hardware). Il sistema costituito ha risposto in maniera corretta alle sollecitazioni, dimostrando la validità degli strumenti usati.

L'ambiente operativo GNU/Linux è stata la scelta obbligata per la piattaforma di sviluppo, sia perché si tratta di un software libero, sia perché è un sistema multiutente nativo e molto stabile. La comunità di sviluppo dei software liberi, inoltre, ci ha messo a disposizione degli strumenti di comunicazione molto efficienti, come le liste di discussione. Per l'architettura di clustering abbiamo scelto Linux Terminal Server Project (LTSP, si veda il Capitolo 3.2), per quanto concerne il sistema di terminali, integrandolo con openMosix (trattato nel Capitolo 3.1), per integrare i server in un cluster. Entrambi i sistemi sono molto attivi ed ormai decisamente maturi.

Abbiamo testato il comportamento del sistema con dei test specifici, riportati nel Capitolo 5, che conducono a una valutazione molto

positiva dei risultati. Inoltre abbiamo sviluppato un package, nominandolo LTCSP (Linux Terminal Clustered Server Project), con cui rilasciamo in rete il software testato e integrato nelle sue parti.

Nelle conclusioni (Capitolo 6), trattiamo i risultati generali dell'intero progetto, ed insieme prospettiamo gli sviluppi che il lavoro potrà avere in futuro.

Capitolo 1

1 La Rivoluzione Digitale e l'impatto sociale ed ambientale

Il divario esistente nell'accesso alle nuove tecnologie (Internet, Computer) presenti nel mondo è figlio della rivoluzione digitale, generata ed alimentata dallo sviluppo delle ICT, tecnologie dell'informazione e della comunicazione. Si sono generate nuove modalità di creare conoscenze, di educare e di diffondere informazioni di ogni tipo, si è modificato il modo in cui nel mondo si conducono gli affari economici e si gestiscono i governi. L'importanza centrale dell'ICT nello sviluppo mondiale è riconosciuta e ribadita anche dall'ONU[1], che afferma che l'accesso alle informazioni ed alla conoscenza è prerequisito fondamentale per il raggiungimento degli obiettivi di sviluppo della dichiarazione del millennio[2] [1].

Nella dichiarazione dei principi del WSIS (Summit Mondiale sulla Società dell'Informazione) viene riconosciuto l'immenso impatto che l'ICT ha su ogni aspetto della vita degli esseri umani. Il rapido

1 Il Segretario Generale dell'ONU aveva annunciato all'interno del Rapporto per il Millennio due iniziative di estremo rilievo:la realizzazione di una nuova Rete Sanitaria per i paesi in via di sviluppo e l'istituzione di un Servizio delle Nazioni Unite per la Tecnologia e l'Informazione chiamato UNITeS

2 Recentemente, a Kuala Lumpur in Malaysia, nel giugno 2006, è nata UnGaid, l'Alleanza globale delle Nazioni unite per le Information and Communication Technologies (ICT) e lo sviluppo. Si tratta di una nuova agenzia dell'Onu che lavorerà per cercare di colmare il *digital divide*

progresso di queste tecnologie apre opportunità completamente nuove per il raggiungimento di più alti livelli di sviluppo. La capacità dell'ICT di abbattere alcuni ostacoli tradizionali, primo tra tutti quello della distanza geografica, rende possibile per la prima volta nella storia, l'utilizzo delle potenzialità delle nuove tecnologie informatiche a vantaggio di milioni di persone in ogni angolo della Terra. Sotto condizioni favorevoli, queste tecnologie possono essere strumenti molto potenti per aumentare la produttività, generare crescita economica, ridurre la disoccupazione, favorire la partecipazione e l'inclusione delle popolazioni nella vita politico-economica in ciascun paese, consentendo un reale intervento delle persone sulle decisioni che le riguardano. L'ICT può garantire la creazione di network e quindi di spazi pubblici di dibattito tra le persone, canali attraverso cui far condividere conoscenze ed esperienze ed in cui far circolare notizie. Lo sviluppo delle ICT aprirà nuove strade per la diffusione e la socializzazione di servizi: è il caso della telemedicina, ma anche nel campo dell'educazione si aprono strade tutte nuove che potrebbero garantire un accesso più ampio all'istruzione. L'e-commerce garantirebbe la possibilità di comunicazione tra realtà locali isolate e il mercato globale, che le nuove tecnologie stanno contribuendo a sviluppare. É quindi molto importante trovare un modo per ridurre i costi legati alle tecnologie, e superare i problemi legati all'assenza di infrastrutture fondamentali, come le reti telefoniche cablate. Sarà particolarmente importante perciò riuscire a mettere a disposizione delle popolazioni computer ancora funzionanti, ma a prezzi più bassi (pensiamo a macchine di penultima generazione opportunamente ottimizzate), e fornire loro la capacità di accesso a Internet (ad esempio tramite sistemi wireless). Le tecnologie digitali potrebbero contribuire ad eliminare alcune delle disparità che si osservano a livello mondiale, dando più forza alla voce dei paesi in via di sviluppo, dissolvendo le barriere nazionali e rafforzando il processo di democratizzazione. Proprio la grande

importanza che l'ICT riveste per lo sviluppo della società in ogni suo aspetto, rende ancora più stridente il divario tra paesi ricchi e paesi poveri, cioè tra chi ha la possibilità di accompagnare quello sviluppo traendone i massimi benefici, e chi invece a causa di una arretratezza economica, culturale ed infrastrutturale, rimane tagliato fuori. Appare evidente quindi che il superamento del cosiddetto divario digitale (digital divide) tra il Nord ed il Sud del mondo diventi in questo contesto una delle sfide più urgenti che si pongono all'attenzione della comunità internazionale.

1.1 Il Divario Digitale

Divario digitale è il termine utilizzato per indicare le disuguaglianze nella fruizione (accesso ed utilizzo) delle tecnologie informatiche. Ciò che si osserva, nonostante l'importanza che tali tecnologie ricoprono per lo sviluppo economico e sociale a livello globale, è una sempre maggiore difficoltà nella loro utilizzazione per alcune categorie sociali e addirittura per interi Paesi. Se da un lato la rivoluzione tecnologica in atto, favorita dalla crescita del World Wide Web, offre alla popolazione mondiale enormi possibilità di sviluppo, dall'altra contribuisce ad alimentare nuove forme di disuguaglianza. Tale rivoluzione presuppone infatti grandi investimenti e la presenza di infrastrutture e servizi spesso assenti in numerosi paesi.

Dal punto di vista culturale ed educativo possiamo dire che va aumentando il divario tra società alfabetizzate e società in cui il tasso di analfabetismo è ancora molto alto. Oltre a questo fenomeno, che riguarda le differenze tra paesi con diversi gradi di sviluppo, anche nei paesi più sviluppati si vanno generando le condizioni per una

nuova forma di analfabetismo: l'analfabetismo elettronico.

1.1.1 Alcuni dati sul Divario Digitale

Per una analisi della situazione mondiale rispetto alla questione della fruizione delle tecnologie digitali, si prenderanno in considerazione due studi molto dettagliati condotti da Orbicom (Unesco Chairs in Communication), pubblicati nell'Ottobre del 2003 [2], e nell'Ottobre del 2005 .Questi studi sono basati sull'analisi dello stato di ciascun paese in termini di fruizione e diffusione dell'ICT, di investimenti nelle nuove tecnologie e di livello medio di istruzione, negli anni dal 1996 al 2003. In questa analisi, infatti, per ciascun Paese (si veda l'appendice C) vengono introdotti tre indicatori:

- **Infodensity**: somma di tutti gli indici della capacità produttiva in riferimento all'ICT (capitale, forza lavoro, infrastrutture ma anche il livello medio di istruzione della popolazione);
- **Infouse**: flussi di consumo delle ICT;
- **Infostate**: aggregazione di infodensity e infouse;

Sulla base del valore dell'infostate (che viene assunto come indice di sviluppo tecnologico dei paesi) è stato possibile stilare una graduatoria su base mondiale di tutti gli stati. É possibile suddividere le nazioni in cinque gruppi (da A ad E):

- Le nazioni che fanno parte del gruppo A (gruppo alto), includono i paesi dell'ovest europeo (inclusi i paesi scandinavi, Olanda, Svizzera, Belgio, Lussemburgo, UK, Germania), Stati Uniti d'America e Canada, Hong Kong, Singapore, Repubblica

Coreana, Giappone, Australia, Nuova Zelanda e Israele. Questo gruppo di 23 paesi, rappresenta il 13% della popolazione mondiale.

- Le nazioni che fanno parte del gruppo B (gruppo elevato) sono 24 e rappresentano solamente il 4% della popolazione mondiale. Qui troviamo gli stati del sud Europa (Portogallo, Italia, Spagna, Malta, Cipro e Grecia) e dell'Europa dell'est (Slovenia, Estonia, Repubblica Ceca, Ungheria, Repubblica Slovacca, Polonia, Croazia e Lituania), affiancati da Cile, Uruguay, Argentina, Emirati Arabi, Qatar, Macao nel Brunei e Barbados.
- Le nazioni che fanno parte del gruppo C (gruppo intermedio) sono 26 e rappresentano un terzo della popolazione mondiale. Questo gruppo e composto da paesi molto distribuiti a livello geografico tra cui i paesi dell'America Latina (Brasile, Costa Rica, Colombia, Venezuela, San Salvador, Panama, Perù), alcuni stati dai paesi Arabi (Kuwait, Arabia Saudita, Giordania), due paesi Africani “avanzati” (Mauritius e Sud Africa), Malesia, Thailandia e China dall'Asia. Inoltre sono presenti alcuni stati Balcani (Bulgaria, Romania, Jugoslavia), Giamaica, Trinidad e Tobago.
- Le nazioni che appartengono al gruppo D (gruppo moderato) sono 34 e rappresentano il 29% della popolazione mondiale. Tra essi molti paesi dell'America Latina (Ecuador, Bolivia, Paraguay, Guatemala, Nicaragua e Honduras) e dell'Asia (Iran, Armenia, Mongolia, Kyrgyzstan, Indonesia, Sri Lanka, Vietnam ed India). In questo gruppo si trovano inoltre alcune nazioni Nord-Africane (Tunisia, Egitto ed Algeria) e diversi stati dell'Africa sub-Sahariana (Botswana, Gabon, Zimbabwe, Togo e Gambia).
- Le nazioni che fanno parte del gruppo E (gruppo basso)

rappresentano il 15% della popolazione mondiale nel 2003. Esse sono 32, perlopiù concentrate in Africa, con Ciad, Etiopia, e Repubbliche Africane Centrali in coda. I paesi non africani nel gruppo sono Birmania, Cambogia, Bangladesh, Nepal, Laos, Yemen e Pakistan.

Gli studi condotti da Orbicom evidenziano una crescita dell'indice di sviluppo in tutti i gruppi di Paesi (illustrazione 1). Tale crescita evidenzia una tendenza mondiale, che però non dà una misura del Digital Divide. Sono le differenze nell'Infostate a fornire i dati per

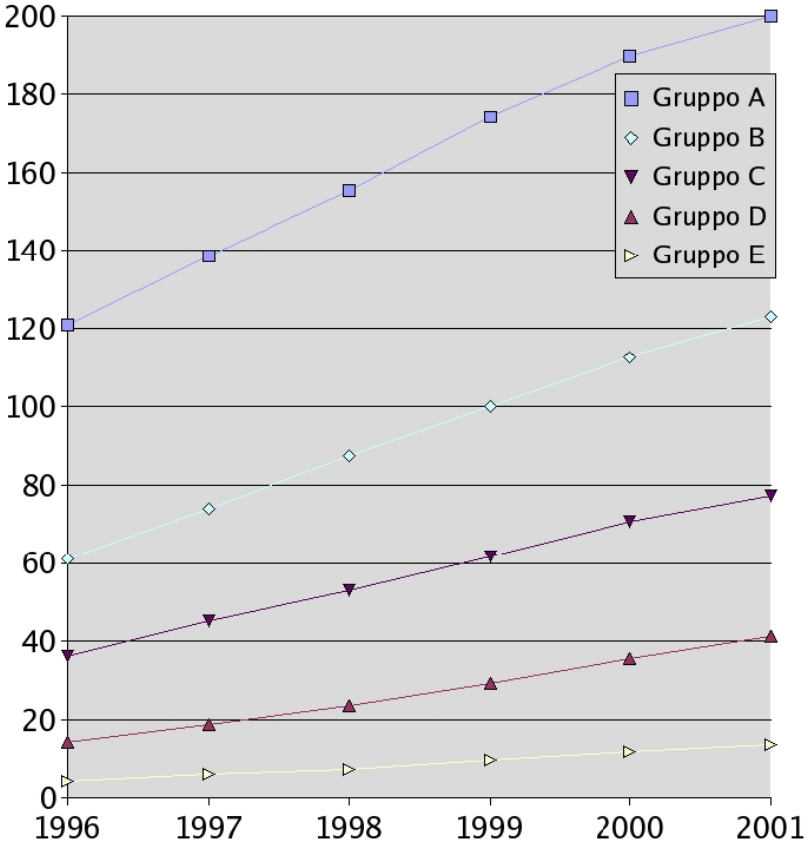

Illustrazione 1: Indice di sviluppo tecnologico per i gruppi

monitorare l'evoluzione del divario tra gli stati. Le differenze tra i Paesi più sviluppati e le nazioni più arretrate crescono in valore

assoluto ogni anno. L'unico dato confortante è che il tasso di crescita annuo dei Paesi con un livello di sviluppo più basso è più grande di quello dei Paesi maggiormente sviluppati (si veda l'illustrazione 2, confrontandola con i dati dell'appendice C).

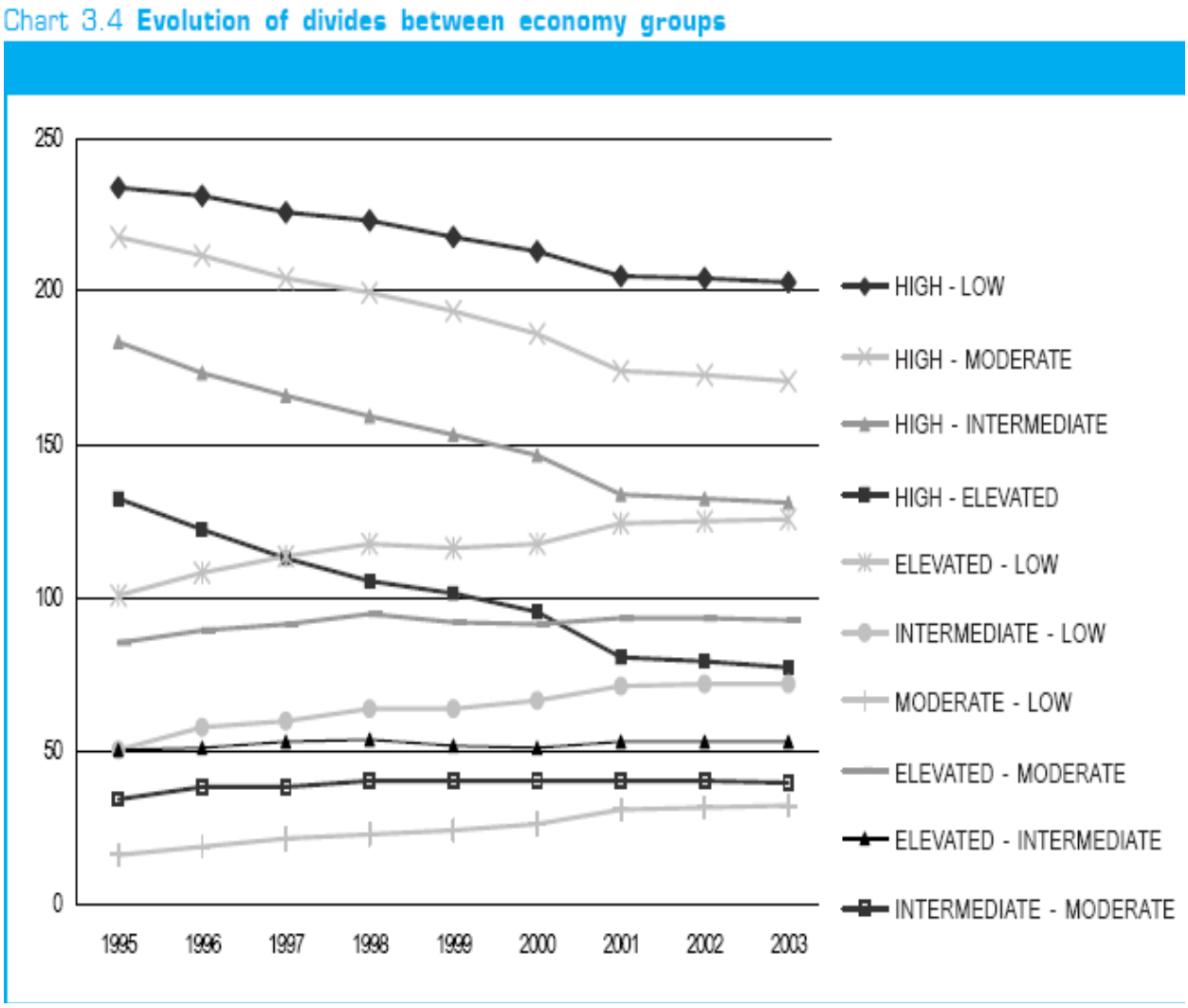

Illustrazione 2: Evoluzione del grado di divario tra gruppi di paesi

Se volessimo interpretare ottimisticamente i dati, potremmo dire che il fatto che i paesi meno sviluppati abbiano un tasso di crescita maggiore, è un indicatore del fatto che in essi si assiste ad una crescita percentuale degli investiment nell'ICT.

Nonostante questa tendenza ad un aumento degli investimenti, dall'interpretazione dei dati appare chiaro che la maggior parte della popolazione mondiale ancora esclusa da questa rivoluzione

tecnologica a da tutti quei benefici che essa può portare (telemedicina, telelavoro, e-learning, ecc).

Un dato di estremo interesse per i nostri studi è la diffusione dei PC rispetto al numero di abitanti. Il dato ci dice che nei paesi del gruppo A si assiste ad una crescita del 15,6% su base annua, mentre nei paesi meno tecnologicamente sviluppati si ha un numero di PC per 100 abitanti assolutamente insignificante e una crescita molto bassa (circa il 2%).

Facendo una valutazione generale, nei nei paesi del Sud del Mondo la difficoltà nello sviluppo e nella diffusione dell'ICT è data dalla carenza di infrastrutture di telecomunicazione, spesso concentrate perlopiù nelle grandi città, e totalmente assenti nelle zone rurali (nelle quali vive invece la maggior parte della popolazione), dai costi molto elevati delle connessioni alla rete telefonica e, per l'appunto, dalla scarsa disponibilità di computer. Oltre a queste motivazioni di carattere tecnico si dovrà aggiungere anche il fatto che il livello di istruzione è in molti casi assolutamente deficitario, tanto da indurre gli analisti a parlare di divario conoscitivo (knowledge divide). É indicativo, infatti, che proprio nei Paesi in cui l'accesso ad Internet è più basso, si abbia anche un livello di istruzione inferiore.

1.1.2 I Paesi più sviluppati ed il Divario Digitale

Nei Paesi più sviluppati si riscontra un problema diverso. In essi si ha una grande diffusione delle tecnologie digitali. Tale diffusione è legata ovviamente al livello di ricchezza, che oltre a consentire la fruizione delle tecnologie, ha anche incentivato la ricerca con opportuni investimenti.

Uno studio del Dipartimento del Commercio USA, condotto a partire dal 1993, dimostra che c'è una mancanza di accesso ai computer da parte delle classi più povere, delle persone in possesso di un titolo di istruzione secondaria, delle popolazioni di colore ed Ispaniche, delle comunità rurali e delle donne. Vediamo quali sono i fattori discriminanti:

- ***Reddito***: nello studio si evidenzia come il numero dei PC nelle case sia quadruplicato tra il 1984 e 1997, ma allo stesso modo questo periodo ha visto crescere le disparità nel possesso dei dispositivi hardware tra strati sociali. Uno studio dell'OCSE (Organizzazione per la Cooperazione e lo Sviluppo Economico) [3], che ha raccolto dati dalla Francia, dal Giappone e dagli USA, ha confermato la sostanziale disparità nell'accesso ai PC in base ai differenti redditi familiari;
- ***Occupazione***: uno studio condotto dai governi di Gran Bretagna, Finlandia e Danimarca, riporta un dato inequivocabile: i manager e i liberi professionisti che hanno più mezzi messi loro a disposizione fanno un utilizzo di tecnologie che è doppio rispetto a quello degli altri comuni impiegati e triplo rispetto agli operai ed ai disoccupati;
- ***Educazione***: il livello di educazione è una variabile molto importante, visto che scuole ed università educano all'utilizzo dei computer e forniscono talvolta la possibilità di navigare gratuitamente. Si può osservare che il divario che si ha in base all'occupazione è assolutamente analogo a quello che si vede rispetto all'educazione, essendo spesso l'una dipendente dall'altra;
- ***Sesso***: in questo campo possiamo dire che le differenze tra uomini e donne nell'utilizzo del PC e nell'accesso ad Internet rispecchiano sostanzialmente le differenze sociali tra i sessi;
- ***Età***: in Europa si evidenzia una situazione per la quale gli

utilizzatori delle tecnologie digitali sono sostanzialmente i giovani tra 18 e 25 anni, mentre la percentuale di utenti decresce vertiginosamente al crescere dell'età con una percentuale di utenti over-65 che in tutta Europa è molto vicino allo zero.

1.1.3 Accessibilità e consumismo informatico

Dal quadro delineato nel paragrafo precedente si evince che accanto alle persone che si possono permettere soluzioni hardware di ultima generazione, abbiamo tutta una serie di categorie sociali che invece per carenza di istruzione, problemi economici o legati a barriere culturali, rimangono escluse da questo meccanismo. É opportuno inoltre evidenziare una certa tendenza, cui assistiamo negli ultimi anni nei paesi più sviluppati, al radicamento di un atteggiamento generale di "consumismo" a livello informatico. Vengono messi sul mercato computer sempre più potenti con una capacità di calcolo che raddoppia ogni 18 mesi. Ad hardware sempre più performanti corrispondono software sempre più complessi e potenti, che hanno bisogno di girare su macchine migliori e viceversa.

In tale meccanismo, ogni applicativo è pensato per una determinata macchina e non per le precedenti. Si genera, così, una inaccessibilità all'indietro, per la quale, per avere accesso alle nuove funzioni di un sistema appena prodotto, si è obbligati a sostituire anche la macchina.

In questo sistema di duopolio anche utenti casalinghi con limitate necessità di calcolo si vedono costretti ad acquistare delle macchine assolutamente sovradimensionate, quando invece computer ancora

perfettamente funzionanti (anche se non di ultima o penultima generazione) vengono dismessi.

1.2 Il problema dell'hardware obsoleto

Da uno studio condotto dalle Nazioni Unite è emerso che per fabbricare un PC ed un monitor sono necessarie 1,8 tonnellate di materie prime. In particolare per produrre un normale desktop, equipaggiato con un monitor CRT da 17", sono richiesti 240 kg di combustibili fossili, 22 kg di sostanze chimiche e 1500 kg di acqua. Sembra certo poi, che alla fine di questo decennio, il numero di apparecchi finiti nelle discariche arriverà a quota 3 miliardi (cifra che comprende computer, cellulari, televisori, frigoriferi, stampanti e tutte le periferiche).

Le sostanze dannose per l'uomo e per l'ambiente presenti nelle apparecchiature elettriche ed elettroniche sono piombo, cadmio, mercurio, cromo esavalente, varie materie plastiche, ritardanti di fiamma bromurati. Attualmente, in Europa il 90% dei rifiuti elettronici viene avviato alla discarica, incenerito o frantumato senza che sia stato precedentemente trattato. Come se non bastasse, le componenti di piccole dimensioni vengono trattate come normali rifiuti domestici e, come tali, finiscono direttamente negli inceneritori.

Durante l'anno 2002 sono state dismesse, perché guaste o semplicemente perché sostituite con modelli più potenti ed aggiornati, circa 12.000 tonnellate di monitor, 12.400 tonnellate di computer tra desktop e portatili (compresi mouse, tastiere, modem ecc.), 1240 di server e workstation, 900 di scanner, 2600 di stampanti, 13.800 di fax e copiatrici, 5000 di toner, superando così la

soglia del **milione di tonnellate di spazzatura elettronica** prodotta nel nostro Paese nell'ultimo decennio. I dati riguardanti l'unione Europea ci dicono che in media ogni cittadino dell'unione produce ogni anno 20 Kg di spazzatura elettronica e che tale cifra è destinata ad aumentare con un tasso complessivo che oscilla tra il 16% e il 28% annuo. Secondo il WWF, l'incenerimento dei RAEE (rifiuti di apparecchiature elettriche ed elettroniche) emette nell'atmosfera circa 36 tonnellate di mercurio e 16 di cadmio all'anno, e contribuisce per più di metà del piombo immesso negli inceneritori. A queste cifre sull'inquinamento legato allo smaltimento dei rifiuti elettronici, dobbiamo aggiungere anche quelle legate alla produzione dei computer.

1.2.1 L'apparato normativo

Queste cifre rendono necessaria una corretta sensibilizzazione del cittadino e delle imprese ma anche l'emanazione di leggi che regolamentino in modo chiaro lo smaltimento dei RAEE, oltre alla creazione di centri specializzati per la separazione dei componenti delle apparecchiature elettroniche e per il loro riutilizzo. L'unione europea impone dal 1 luglio 2006 il divieto di costruire computer, telefonini ed elettrodomestici usando piombo, mercurio, cadmio e cromo esavalente. Il passo ulteriore che la UE ha previsto per gli stati membri è che entro il 31 dicembre 2006 venga obbligatoriamente raggiunto un tasso minimo di raccolta selettiva di apparecchi a fine vita provenienti dai nuclei domestici pari a 4 Kg/anno pro-capite.

L'Italia recepisce, anche se tardivamente, le direttive europee in materia di smaltimento di rifiuti elettrici ed elettronici con il Decreto

Legge approvato dal Consiglio dei Ministri il 22 luglio scorso. Tuttavia, rimane più di un dubbio sulla sua reale efficacia in quanto non sono chiare le modalità in cui verranno distribuiti i finanziamenti economici a chi ha il dovere di provvedere alla raccolta e al riciclaggio dei RAEE.

1.2.2 La politica delle aziende

Mentre i paesi dell'UE si preparano ad adeguarsi alle nuove normative comunitarie, alcune aziende hanno già da qualche tempo iniziato a sviluppare politiche di riciclaggio adottando nuove iniziative ed investendo denaro per renderle visibili alla maggioranza delle persone. Fu Hewlett-Packard la prima azienda ad offrire ai propri clienti la possibilità di dar indietro la macchina vecchia con l'acquisto di quella nuova. Una soluzione semplice che consente al produttore di avviare un computer inservibile allo smaltimento, dopo aver recuperato i componenti ancora utilizzabili. In particolare HP si muove su più fronti: ha avviato un programma per la ricerca di materiali meno inquinanti da utilizzare nella progettazione dei prodotti HW, inoltre viene data molta attenzione alla costruzione di apparati facilmente smontabili, ovvero le cui componenti, una volta arrivate a fine vita, possano essere facilmente separate. Per quanto riguarda i consumabili, HP ha istituito un servizio di ritiro di toner e cartucce esauste dai propri clienti, purché essi abbiano una grande quantità di apparecchiature. Il servizio di recupero è affidato ad aziende partner selezionate tra i trasportatori ed i riciclatori autorizzati. In più all'utente finale viene offerto un servizio a pagamento per la restituzione dell'hardware, che consiste nella distribuzione su tutto il territorio italiano dei kit di restituzione, ossia dei box che possono

contenere fino a 25 kg di materiale hardware obsoleto e che vengono ritirati, una volta pieni, dalla società partner. Per stessa ammissione della Direzione Affari governativi e pubblici di HP Italia, tale iniziativa ha avuto finora una risposta molto fredda da parte dell'utente finale, che continua ancora a preferire il cassonetto. Simili iniziative sono state intraprese anche dalla Epson, ma queste riescono a coinvolgere solo i grandi utenti. Molto attiva nel campo è anche la Fujitsu-Siemens, che ha creato in Germania un centro per lo stoccaggio di computer e periferiche dismesse. Molte altre aziende dichiarano di avere progetti in stand-by, o comunque di essere ancora nella fase di studio dei progetti.

Greenpeace lancia periodicamente la "Ecoguida ai prodotti elettronici", che premia le aziende che non fanno uso di sostanze chimiche pericolose e contribuiscono al riciclaggio dei rifiuti elettronici [4]. Nokia e Dell sono al primo posto nella classifica, ed entrambe affermano che come aziende produttrici dovrebbero essere responsabili dei propri prodotti e rifiuti che devono essere avviati al riciclaggio. Nello studio è valutata con maggiore attenzione l'eliminazione delle sostanze pericolose rispetto al riciclaggio, perché, finché non viene intrapreso questo passo, un riuso e riciclaggio sicuro dei componenti non è possibile.

Greenpeace fa comunque affidamento sulla correttezza delle dichiarazioni delle aziende e in caso si scopra la non veridicità dei dati, ciò viene tenuto in conto nella classifica stilata successivamente.

Come si vede lo scenario è molto complesso. Viene data grande enfasi a due aspetti fondamentali: l'eliminazione di materiali dannosi dal processo di fabbricazione dei dispositivi e il corretto smaltimento dei rifiuti elettronici (recupero, riciclo, smaltimento). Non viene data invece la giusta enfasi ad un terzo aspetto fondamentale: lutilizzo più corretto dei PC e di tutte le periferiche e il riuso del materiale

obsolescente ma ancora perfettamente funzionante.

1.2.3 Recupero, Riciclo e Smaltimento

Avvalendoci dei risultati di una ricerca condotta dalla Microelectronic and Computing Technology Corporation, possiamo dire che all'interno di un normale desktop (approssimativamente 27 kg di peso) ci sono oltre trenta elementi, caratterizzati da possibilità di riciclaggio anche molto differenti.

Di fronte ad una tale pluralità di elementi presenti in quantità anche minime e con tassi di efficienza di riciclaggio talvolta molto bassi, si intuisce come soltanto con volumi alti di hardware smaltito si riescono a ottenere quantità significative di materiali da recuperare. Un corretto processo di trattamento dei dispositivi elettronici obsoleti, dovrà constare di diverse fasi quali quella di raccolta delle apparecchiature, di cernita e collaudo dell'HW, di prelievo dei componenti riutilizzabili, di frantumazione e recupero di materiali che possono essere riciclati. Solo quando si è recuperato tutto il possibile si passerà allo smaltimento vero e proprio.

Questo modo di operare può permettere di ottenere il massimo in termine di profitti per le aziende, di abbattere i tassi di inquinamento e di consentire, prendendo in analisi la sola Europa, di risparmiare 120 milioni di giga-joule, equivalente a 2,8 milioni di tonnellate di petrolio ogni anno, con un risparmio energetico pari al 60-80% rispetto all'utilizzo di materia vergine.

Dal punto di vista commerciale, per le aziende il problema è ricavare profitto dal disassemblaggio e dal recupero dei componenti, oppure dal ricondizionamento e dal riutilizzo dei computer. É per questo che

l’attività di recupero dei PC obsoleti è svolta in Italia da associazioni senza fini di lucro, che destinano le macchine ad associazioni no-profit o a Paesi con economie più povere. Si parlerà infatti, più avanti, di **RaccattaRAEE**, che opera nell'area bolognese.

L’aspetto del recupero dell’hardware obsoleto in vista di un suo riutilizzo è per noi di centrale importanza. Sono due le idee che ci guidano nello studio che conduciamo: la prima è legata alla sempre più pressante necessità di smaltire grandi quantità di computer; l’altra, meno ovvia ma altrettanto importante, è dare all’obsolescenza tecnologica il suo corso naturale, e non il frenetico ritmo artificiale imposto negli ultimi anni. Spesso, infatti, i computer vengono dismessi dalle aziende ancora funzionanti, benché con prestazioni non al passo con il software di ultima generazione. Hanno un valore residuale che però non è sfruttabile dalle imprese, perché per esse tenere un computer lento in produzione fa indirettamente crescere altri tipi di costi, specie in termini di produttività di un impiegato. Tale valore può però essere utilizzato convenientemente in tutte le realtà in cui la presunta lentezza dell'hardware non causa un problema rilevante (come associazioni, scuole ecc). Riutilizzando in particolari realtà i computer dismessi, si ottimizza l’utilizzo del valore economico totale della macchina.

In sostanza, si vorrebbe promuovere un corretto utilizzo della risorsa tecnologica: il PC ormai ammortizzato, lento, poco produttivo, viene sostituito da uno più competitivo ed efficiente. É importante che le aziende imparino a liberarsi subito dei PC che non usano e che, altrimenti, occupano inutilmente spazio, ma soprattutto perdono quel valore residuo che può essere socialmente utile. Così facendo si ottengono due vantaggi economici: il risparmio dello smaltimento immediato e la possibilità di avere a disposizione risorse informatiche accessibili a tutti.

Capitolo 2

2 Il Trashware e il Free Software

Con il termine "Trashware" si intende la pratica di riutilizzare produttivamente hardware ormai dismesso (ed altrimenti destinato allo smaltimento) rendendolo di nuovo funzionante ed utile, magari assemblando anche dispositivi di computer diversi. La parola è stata coniata dal Gruppo Operativo Linux Empoli (Golem) e nasce da un merge dei termini “trashing” e “hardware”. Chi fa trashware recupera, ripristina, ridistribuisce e riutilizza le macchine dismesse da privati, enti pubblici ed aziende per concederle in uso e senza scopo lucrativo ad associazioni di volontariato o a progetti di solidarietà internazionale e cooperazione allo sviluppo.

Fino ad oggi, come detto precedentemente, lo smaltimento dell’hardware è avvenuto senza troppe preoccupazioni. Pur essendo l’e-waste molto difficile da trattare, questo tipo di rifiuto è stato tenuto in passato abbastanza facilmente sotto controllo, viste le ridotte quantità di materiale da gestire. Negli ultimi anni, invece, la quantità di macchine da dismettere ha raggiunto un numero elevatissimo (Tabella 1), tanto che quello dei rifiuti tecnologici è diventato un problema ecologico di grande attualità. Le motivazioni che spingono i gruppi italiani di Trashware sono sostanzialmente legate alla

questione ecologica, di cui si è precedentemente parlato, ma anche a una questione educativa, come il corretto utilizzo dei dispositivi hardware. Spesso infatti i computer vengono dismessi sebbene ancora funzionanti, perché troppo lenti, e tale lentezza è vista nei settori aziendali come un fattore di rallentamento inaccettabile della produttività. I computer, però, hanno un valore comunque ancora molto alto (non dimentichiamo che i computer più performanti di oggi saranno considerati obsoleti tra 18 mesi, come consegue dalla Legge di Moore) che può essere sfruttato in tutte quelle realtà in cui la supposta lentezza non è un grande problema, ma anche in generale utilizzando strumenti che aggregano tra loro le potenzialità delle singole macchine, per costruire una sistema più efficiente. Per chi non può permettersi un computer, come associazioni di volontariato, scuole, associazioni culturali, studenti, privati cittadini con redditi bassi ecc, avere una macchina di penultima generazione opportunamente ottimizzata e riconfigurata è una possibilità molto appettibile. In tal modo si fornisce un servizio utile alla comunità ed al contempo si ottimizza l'utilizzo del valore economico totale dei PC.

Il riutilizzo, di fatto, non elimina la necessità di smaltimento fisico dei computer, ma sicuramente minimizza e ottimizza gli sprechi e l'impatto sull'ambiente e sulla società. In ultima analisi possiamo dire che anche per le aziende può essere un vantaggio non indifferente, in quanto le macchine dismesse vengono spesso tenute ammassate nei magazzini facendo crescere i costi storici, mentre con la concessione dei materiali si promuove l'aggiornamento delle risorse tecniche di un'azienda, senza che questo sia un incentivo al consumo ed allo spreco. Le aziende investono, in termini di efficienza, la parte residuale di questo investimento (scarsamente sfruttabile economicamente), che può essere usata socialmente per la formazione di nuovi tecnici, i quali, a loro volta, potranno essere utili alle imprese.

Condizione fondamentale affinché il Trashware possa attecchire è

che le aziende abbandonino alcuni comportamenti che sono dannosi per il processo di riutilizzo. Uno di questi è lo stoccaggio per lunghissimi periodi di tempo delle macchine in luoghi inadatti; spesso vengono tolte parti vitali da conservare per casi di necessità (che quasi mai si realizzano) attuando quella che in gergo tecnico viene chiamata "cannibalizzazione dei PC".

L'attività di Trashware è inscindibilmente legata all'utilizzo di Software Libero o software Open Source. Il software proprietario ha, infatti, diversi tipi di incompatibilità con la natura stessa del progetto: un'incompatibilità economica, poiché l'acquisto delle licenze sarebbe troppo oneroso per associazioni di volontariato e per le popolazioni dei paesi in via di sviluppo; un'incompatibilità etica, poiché per la comunità Linux è di fondamentale importanza mantenere l'indipendenza dalle politiche delle grandi aziende (mentre installare software proprietario significherebbe diventare uno strumento commerciale di diffusione dei loro prodotti); infine un'incompatibilità di carattere tecnico nel senso che è estremamente difficile reperire versioni di qualunque software non libero in grado di funzionare in modo corretto sulle macchine oggetto del recupero e, anche qualora questo fosse possibile, risulterebbe spesso inutile poiché i prodotti nuovi e aggiornati spesso non hanno caratteristiche accettabili di retrocompatibilità, ma anzi contribuiscono in modo determinante all'obsolescenza tecnica e all'esclusione sociale.

Anni	Server	PC	Totale
1998	72.600	615.000	687.600
1999	105.500	1.109.500	1.215.000
2000	86.500	1.176.500	1.263.000
2001	116.000	861.500	977.500
2002	127.550	1.042.500	1.170.050

Tabella 1: Computer avviati alla dismissione in Italia 1998-2002 (dati in unità)

2.1 Free Software e Software Open Source

Negli ambienti scientifici, la libertà di scambio di idee ed informazioni è un principio tenuto in alta considerazione per la fecondità che ha dimostrato; ad esso infatti è generalmente attribuita molta parte dell'eccezionale ed imprevedibile crescita del sapere negli ultimi tre secoli. Il concetto di software libero discende naturalmente da quello di libertà di scambio. Quest'ultima non è tuttavia una questione puramente pratica: essa è infatti alla base dei concetti di libertà di pensiero e di espressione. Analogamente alle idee, il software è immateriale, e può essere riprodotto e trasmesso facilmente. Parte essenziale del processo che sostiene la crescita e l'evoluzione del software è la sua libera diffusione. Ed ogni giorno di più, come le idee, il software permea il tessuto sociale e lo influenza, producendo effetti etici, economici, politici e in un senso più generale culturali.

Il primo a formalizzare il concetto di software libero fu **Richard M. Stallman** nei primi anni Ottanta. La definizione che egli ha prodotto, assume la forma di quattro principi di libertà:

- ***libertà 0***, o libertà fondamentale: libertà di eseguire il programma per qualunque scopo, senza vincoli sul suo utilizzo
- ***libertà 1***: libertà di studiare il funzionamento del programma e di adattarlo alle proprie esigenze.
- ***libertà 2***: libertà di redistribuire copie del programma.
- ***libertà 3*****:** libertà di migliorare il programma, e di distribuirne i miglioramenti.

Il software distribuito con una licenza che rispetti questi principi è detto libero (free software). Nel 1984 Stallman diede vita al progetto GNU, con lo scopo di tradurre in pratica il concetto di software libero

e creò la Free Software Foundation, per dare supporto logistico, legale ed economico al progetto GNU. La licenza del progetto GNU, la Licenza Pubblica Generica GNU (GNU GPL), al contrario delle normali licenze d'uso, concede all'utilizzatore di un programma di godere di tutte e quattro le libertà suddette. Inoltre si occupa di tutelarle nel senso che chi modifichi un programma protetto da GPL e lo distribuisca lo deve fare ancora sotto licenza GPL. Con un gioco di parole, il nome dato a questo tipo di protezione è permesso d'autore (in inglese **Copyleft**): è il criterio che prevede che le modifiche ad un programma possano essere distribuite solo con la stessa licenza del programma originale. Le licenze dei software proprietari usano le norme sul diritto d'autore (copyright) per togliere libertà agli utenti di un programma; il permesso d'autore usa le stesse norme per garantire quelle libertà e per proteggerle.

Nel 1998 Bruce Perens, Eric Raymond ed altre personalità nel campo del software libero si convinsero che i principi di libertà associati ad esso fossero mal visti nel mondo degli affari, a causa della loro carica ideologica. Lanciarono, così, una campagna di promozione del free software che mettesse in luce i suoi numerosi vantaggi pratici come la facilità di adattamento, l'affidabilità, la sicurezza, la conformità agli standard, l'indipendenza dai singoli fornitori ecc. Scrissero a tal fine la Open Source Definition, il documento fondamentale del movimento Open Source. In più la nuova definizione che essi trovarono per il software prodotto consentiva loro di superare un'ambiguità fondamentale, quella legata alla parola "free" nella lingua Inglese che può significare "libero" ma anche "gratuito". Tale ambiguità è di non secondaria importanza per chi si vuole guadagnare da vivere con la scrittura di programmi. La definizione ufficiale di "software open source" come pubblicata dalla Open Source Iniziative, si avvicina molto a quella di free software; tuttavia è per alcuni aspetti un po' più ampia ed in più ha accettato alcune licenze che vengono considerate dal movimento di Stallman

come inaccettabilmente restrittive per gli utenti. La voluta neutralità del movimento open source verso gli aspetti etici e politici del software libero è la caratteristica sostanziale che lo distingue dalla filosofia di quest'ultimo. Possiamo dire, quindi, che la differenza che c'è tra i due movimenti è politica più che pratica, visto che essi concordano sulle licenze accettabili ed hanno obiettivi e mezzi comuni. I due movimenti sono cioè in disaccordo sui principi di base, ma sono più o meno d'accordo sugli aspetti pratici. Per questo essi possono lavorare insieme su molti progetti specifici. Al di là delle questioni ideologiche, possiamo sicuramente dire che la diffusione del software libero ed open source è diventato uno dei fenomeni più interessanti dell'intero panorama delle tecnologie informatiche, generando un livello di interesse e producendo un numero di appassionati che è paragonabile a quello dei primi anni della diffusione di Internet. Sebbene, come visto, il fenomeno non sia recentissimo, solo negli ultimi anni ha raggiunto un numero critico di persone, che gli ha consentito di entrare nel mercato del software tradizionale. Negli ultimi anni l'impatto di questo nuovo tipo di programmi sta diventando molto forte nell'industria del software e nella intera società. Sta diffondendo un nuovo tipo di modello di sviluppo, che trae vantaggio dal lavoro di sviluppatori sparsi per il mondo; proprio la presenza di comunità che contribuiscono allo sviluppo del software o anche che semplicemente risolvono problemi per i nuovi utenti è uno degli aspetti più affascinanti di questa realtà tutto sommato nuova. La tecnologia Open Source consente un modello di affari che è completamente nuovo e che ha in generale un impatto molto positivo nella creazione di nuovi mercati e di nuove opportunità di lavoro.

2.1.1 Modello di sviluppo

Oltre all'impatto che l'Open Source sta avendo sul mercato dell'Information Technology, esso sta producendo molti cambiamenti nel campo dello sviluppo del software, abbattendo quello che era il concetto tradizionale del software engineering, per il quale solo una gestione centralizzata ed un forte controllo sull'accesso al codice sorgente garantivano un prodotto di alta qualità. Il modello proposto e diffuso dall'Open Source è per certi aspetti opposto; in esso, infatti, un gran numero di sviluppatori sparsi nel mondo collaborano per costruire un prodotto dall'alto standard qualitativo. Le motivazioni che inducono a usare e sviluppare il software libero sono molte e molto diverse tra loro e sono di natura etica, filosofica o semplicemente dettate da aspetti pratici. Vediamo questi ultimi:

- L'accessibilità al codice sorgente e il diritto di modificarlo garantisce la possibilità di mettere a punto e di migliorare il prodotto. Consente di adattare il codice a nuovi hardware (aspetto fondamentale per la nostra ricerca), di adattarlo al variare delle condizioni, e di raggiungere una piena comprensione di come il sistema lavora. É per questo che molti esperti sono arrivati alla conclusione che per estendere la vita di un'applicazione, deve essere disponibile il suo codice sorgente. Inoltre l'avere a disposizione il codice sorgente consente agli sviluppatori di identificare bachi e correggerli;
- Il diritto di redistribuire le modifiche e le migliorie del codice consente a tutti di fruire dei miglioramenti apportati dagli sviluppatori al codice;
- Il diritto di utilizzo del software garantisce, insieme al diritto di redistribuzione, che ci sia una grande quantità di utenti potenziali. L'enorme diffusione garantisce un ambiente di testing

di scala globale, inoltre un mercato molto ampio che richiede sempre più supporto e adattamento per il software prodotto, attirando sempre nuovi sviluppatori e di conseguenza migliorando la qualità del prodotto e le sue funzionalità;

- Non c'è nessuno che abbia il potere di porre restrizioni all'uso del software. Cosa che invece si ha ogni qualvolta il management di una software-house decide di non aggiornare i programmi per vecchie piattaforme, costringendo gli utenti a rimanere legati alle vecchie versioni, a passare ad un nuovo prodotto o direttamente ad acquistare nuovo hardware;
- Non c'è un unico soggetto da cui dipenda il futuro del software. Questo accade, al contrario, nel caso di software proprietario: se un'azienda decide di non sviluppare un programma o semplicemente chiude, nessuno può continuare lo sviluppo di quel prodotto. Questo problema è amplificato, negli ultimi tempi, dalle acquisizioni o dalle fusioni a cui assistiamo nel mercato del software, che spesso instaurano un regime di monopolio. L'Open Source fattivamente protegge da questo pericolo, poiché è sempre possibile trovare qualcuno che si occupi di continuare lo sviluppo senza che ci siano limitazioni legali o pratiche (da citare, il caso di Netscape e Mozilla Firefox);
- Il software a codice sorgente chiuso non è, per definizione, studiabile. Questa limitazione ha ripercussioni negative fondamentali, perché lega lo sviluppo e la personalizzazione dei programmi all'azienda produttrice (crea una reale "dipendenza" da essa, in casi di aggiornamenti e modifiche), ma soprattutto nega all'utilizzatore la conoscenza approfondita degli strumenti che usa. La conseguenza diretta è che i soggetti interessati a sviluppare autonomamente prodotti software, se non usano free software, si vedono tolta questa opportunità. É il caso dei Paesi in via di sviluppo, che perdono la prospettiva di poter produrre

software in settori cruciali come l'educazione e la pubblica amministrazione.

- C'è sempre la possibilità di "forking". Cioè la possibilità di iniziare un codice alternativo se si percepisce che quello corrente sta intraprendendo una gestione sbagliata (un esempio di questo è proprio il progetto openMosix nato dal progetto Mosix nel momento in cui quest'ultimo stava diventando non open). Così nel momento in cui una release di un prodotto è rilasciata sotto licenze proprietarie si può usare quella immediatamente precedente come base di partenza per le successive rilasciate ancora sotto GPL.

Ovviamente il modello di sviluppo appena descritto porta con sé anche quelli che vengono percepiti come svantaggi. Per esempio la possibilità che un prodotto non continui ad essere effettivamente sviluppato, cioè se il prodotto morirà per mancanza di interesse. Ovviamente questo è un problema presente anche per il software proprietario per il quale, però, è la legge del mercato a decretare l'abbandono di un progetto. Altro limite è che alcune volte è difficile poter sapere se un progetto esiste e qual è il suo status corrente. Questo è particolarmente vero per i progetti più piccoli. Tuttavia a questo problema si ovvia grazie all'apparato informativo che si è creato intorno al fenomeno GNU/Linux, cioè grazie al Web, ai giornali, ai forum, alle mailing-list, ecc. Questo modello di sviluppo, insieme alle licenze di rilascio dell'Open Source, hanno generato un meccanismo di cooperazione e competizione genuino, che in ultima analisi ha portato all'alta qualità e all'alta efficienza dei progetti Open Source. Abbiamo dunque un meccanismo di cooperazione tra gli sviluppatori, all'interno di un progetto, ma anche tra programmatori impegnati in progetti differenti. É molto comune che uno sviluppatore legga e corregga i bachi del codice sviluppato da un altro. In più i miglioramenti apportati ad un codice da un programmatore sono in generale fruibili da tutti quelli che, anche in progetti diversi, stanno

utilizzando lo stesso codice. Per quanto riguarda la competizione possiamo dire che i diversi progetti Open Source competono tra loro per emergere all'interno della stessa nicchia di mercato, in tal modo sono forzati a mantenere dei livelli di qualità alti, in modo da non perdere utenti e sviluppatori.

Le caratteristiche fin qui descritte sono collegate alla nostra ricerca per quello che concerne gli aspetti tecnici dell'adattabilità e della flessibilità del software, allo scopo della riqualificazione dell'hardware. Rimane da comprendere come venga recepita la scelta del software libero dall'utente finale, ovvero dai fruitori della soluzione tecnologica. Per questo motivo effettueremo una valutazione di quale sia l'atteggiamento delle pubbliche amministrazioni rispetto alla diffusione del sistema operativo GNU/Linux. Successivamente ci concentreremo su alcuni progetti in cui il software open viene usato in Paesi in via di sviluppo (PVS) nel tentativo di colmare il digital divide.

2.1.2 I governi e l'Open Source

La diffusione del Free and Open Source Software (FOSS) nel mondo è un fenomeno molto ampio a cui molti governi stanno guardando con grande interesse. L'esempio lampante delle politiche sul software libero adottate nel mondo è quello dell'Unione Europea, che sosterrà la diffusione degli standard e dei software liberi su scala globale attraverso il progetto FLOSSWorld. In ogni caso, in tutto il mondo le iniziative dei governi che favoriscono la diffusione del FOSS sono molteplici. Esse riguardano sia l'uso dei software nelle amministrazioni e nelle imprese, come accade in Brasile, Danimarca,

Germania, Thailandia, Sud Africa, Svezia, ma anche nelle scuole, come nelle Filippine, in Spagna.

La Svezia, ad esempio, giustifica l'utilizzo di applicativi e sistemi operativi liberi, dichiarando che essi sono qualitativamente pari o migliori di quelli proprietari. La Gran Bretagna giunge alle stesse conclusioni, in più promuove l'iniziativa per garantire a tutti l'accesso uniforme ai dati (cosa che altrimenti sarebbe subordinata all'acquisto delle licenze dei software proprietari). La Cina, il Giappone, il Sud Corea, l'India promuovono il FOSS negli ambiti di ricerca, istituendo commissioni ed enti che si occupino della transizione.

L'aspetto straordinario che si denota in questa panoramica è che la sensibilità al software libero è alimentata dai governi di Paesi molto diversi tra di loro, in termini di sviluppo e di avanzamento economico.

2.1.3 L'impatto nei Paesi in via di sviluppo

Molto interessante a questo punto è vedere quali sono le caratteristiche che possono rendere l'OSS appetibile in particolare nei paesi del Sud del mondo e che ha fatto sì che si sviluppassero intorno al prodotto open numerosi progetti di cooperazione con lo scopo di colmare il digital divide.

L'impatto dell'Open Source nei paesi in via di sviluppo può essere ancora maggiore che nelle regioni in cui il settore dell'Information Technology è ben sviluppato. Tra le ragioni principali che ci hanno indotto questa valutazione, le più importanti sono innanzitutto la facilità d'accesso ai prodotti software, che possono essere facilmente messi a disposizione di aziende e singoli utenti. Al contrario del software proprietario, il cui costo dell'accesso diretto ai programmi é

vincolati a costose royalties, nel free software non ci sono costi legati a ciascuna copia (se non quelli legati alla distribuzione), per cui l'accesso della popolazione più povera al software può essere possibile praticamente in tutti i paesi. Andando oltre il solo livello di utenza, i paesi in via di sviluppo possono partecipare attivamente allo sviluppo di tecnologie avanzate. Di fatto è interessante notare come importanti progetti di free software (come alcuni di quelli trattati in questo testo) siano stati condotti in paesi con poca tradizione dello sviluppo di programmi. Infine, i PVS (Paesi in Via di Sviluppo) potrebbero contribuire alla ridistribuzione del software in maniera più efficace, poiché più intensamente interessati alla fruizione dello stesso.

In conclusione, possiamo dire che questo tipo di software, rispetto a quello proprietario, consente di ridurre le disparità tra chi è in grado di accedere alle risorse tecnologiche e chi invece non ha questa possibilità. Inoltre consente ai paesi meno sviluppati di contribuire attivamente allo sviluppo di tali tecnologie.

2.2 Il Trashware in Italia

Il Trashware è un fenomeno molto recente ed in grande espansione in Italia, sotto la spinta di associazioni di volontariato e di appassionati utilizzatori del sistema GNU/Linux. Il **Golem** è stato il primo gruppo a strutturare in Italia per il ricondizionamento di computer obsoleti, ed è stato poi seguito da numerosi altri gruppi sparsi in tutto il Paese.

I membri del Golem hanno tracciato le linee guida da seguire per realizzare un progetto di riutilizzo di hardware [5]. Riducendo il

problema ai minimi termini, sono state individuate tre attività principali per portare avanti il progetto. Esse richiedono attrezzature, competenze ed esperienze sostanzialmente diverse e specifiche. Le attività principali sono la raccolta dei computer dismessi, il ricondizionamento con GNU/Linux e Software Libero, e la redistribuzione dei computer a chi può utilizzarli con vantaggio. Per ciascuna delle attività è necessario reperire il soggetto più competente, attrezzato e predisposto.

Il soggetto chiamato a recuperare fisicamente i computer dismessi dall'azienda è il cosiddetto **Soggetto Collettore**. É importante che esso sia già il riferimento istituzionale per chi smaltisce i computer: in questo modo anche chi non è a conoscenza del progetto contatterà comunque il soggetto adatto allo scopo. Sembra in questo senso opportuno coinvolgere l'azienda municipalizzata o consortile che ritira i rifiuti ingombranti, almeno nell'aspetto logistico, organizzativo ed informativo.

In più diventa davvero essenziale la presenza di un soggetto dotato di adeguati mezzi di trasporto sia per raccogliere i computer recuperati, sia per gestire opportunamente tutto il materiale inutilizzabile. L'azienda consortile diventa l'interfaccia unica nei confronti di chi dismette. Essendo questa la sua funzione naturale, può ricoprirla senza sforzo, essendo già attrezzata per farlo. Quindi il meccanismo della raccolta può essere il seguente: chi dismette prende contatto con il Collettore, che si informa sulla presenza di macchine riutilizzabili. A questo punto, se le macchine sono visionabili, il **Soggetto Riqualificatore** sarà chiamato per scegliere i computer riutilizzabili. Si applicheranno, allora, due metodologie di ritiro differenti: i computer non recuperabili saranno ritirati come rifiuti, e i computer ancora buoni prenderanno la via del riutilizzo. Chi dismette dovrà pagare solo per le macchine non più riutilizzabili.

Il Soggetto Riqualificatore è il secondo soggetto, nonché il fulcro

attorno al quale ruota tutta l'attività. Per rendere operativi vecchi computer, si ha bisogno di competenze superiori e tecnici più competenti e preparati rispetto a quanto può bastare per un computer nuovo. Ricorrendo ad un'azienda, i costi di recupero lievitano enormemente, ma non è in genere possibile ricorrere ad un gruppo di volontari "inesperti" che non possono garantire la necessaria produttività del recupero. La scelta del Golem è ricaduta, allora, sui **LUG** (Linux Users Group). Nei gruppi di utenti Linux infatti ci sono persone che per la loro competenza e la loro passione, possono risolvere numerosi inconvenienti hardware e software; offrendo un aiuto preziosissimo.

Infine il terzo soggetto è il **Soggetto Ridistributore**, che potrebbe essere chi conosce e coordina le associazioni di volontariato presenti sul territorio, o chi gestisce e coordina gli aiuti alle associazioni o a gruppi di cittadini (anziani o studenti, per esempio). Quest'ultimo soggetto dovrebbe conoscere le esigenze di chi richiede un computer ricondizionato in modo che si possa operare sulle applicazioni utente da inserire nelle diverse macchine affinché i richiedenti ne traggano massimo vantaggio. Questo soggetto dovrà anche definire i criteri di redistribuzione e gestire la consegna dei PC.

2.2.1 L' associazione RaccattaRAEE

Un esempio attivo nel territorio bolognese nell'ambito del Trashware è rappresentato dall'associazione RaccattaRAEE; questa si occupa del "riciclaggio" di PC dismessi, ma ancora utilizzabili, sottraendoli dalla rottamazione e ridistribuendoli sul territorio. Inoltre l'associazione si occupa di promuove l'informazione al cittadino e alle

imprese, sul corretto fine vita delle apparecchiature elettriche ed elettroniche.

Negli ultimi anni l'associazione è impegnata in diversi progetti, di svariati tipi, tra i più importanti troviamo l'organizzazione di giornate dedicate alla raccolta di rifiuti elettronici (RaccattaRAEE DAY), l'introduzione al “riciclaggio informatico” ai bambini delle scuole elementari, i laboratori di "Monto, Smonto, Aggiusto e Creo"; il ricondizionamento PC per progetti nel terzo mondo ed il Progetto “Senegal - Un Computer per tutti" atto a fornire i mezzi per la creazione di un Laboratorio Telematico per la Scuola Elementare “Malik Kaire Diaw” a Thies[3].

L'Associazione RaccattaRAEE insieme all'Associazione A.C.A.B.A.S. lavora per inviare alla scuola il materiale necessario alla costruzione del laboratorio, il quale permetterà di mettere in comunicazione, entro la fine del 2006, due scuole, in Senegal e a Bologna. Il progetto permetterà scambi culturali tra gli studenti e lezioni sull'energia e sul rifiuto elettronico.

2.2.2 Ingegneria Senza Frontiere

Ingegneria Senza Frontiere è una associazione di volontariato con l'obiettivo di riunire studenti, corpo docente e laureati in Ingegneria ed Architettura che vogliano acquisire conoscenze ed impegnarsi direttamente nell'affrontare le problematiche della progettazione tecnica e della formazione professionale, nei Paesi del Sud del Mondo [6]. Allo stesso tempo l'associazione è impegnata, in Italia, in progetti di sensibilizzazione e di educazione allo sviluppo, nell'ambiente accademico e professionale.

3La scuola ospita 864 bambini (435 maschi e 429 femmine), in 12 classi.

ISF è sorta al Politecnico di Torino nel novembre 1995, sulla base delle esperienze e dei risultati ottenuti da "Ingenièurs Sans Frontiéres" in Francia e da "Ingenieria Sin Fronteras" in Spagna. Attualmente, in Italia si contano 7 Sedi Consolidate (Bari, Firenze, Genova, Pisa, Roma, Torino e Trento) e 10 Nascenti, ognuna di esse attiva presso gli Atenei delle rispettive città. Per far questo si avvale di progetti integrati nel contesto sociale, culturale, ambientale e religioso dei singoli Paesi, a stretto contatto con le realtà dell'Università, delle Organizzazioni Non Governative e delle imprese con cui essa collabora. ISF è impegnata sia nell'ambito della Cooperazione Internazionale, dello Sviluppo Sostenibile, del Risparmio Energetico, sia nel campo formativo, con i temi dell'Etica nella professione tecnica, la conoscenza e il rispetto dell'ambiente e delle culture.

Per quanto concerne il nostro lavoro, abbiamo potuto contare su uno stretto rapporto di aiuto e collaborazione da parte della sede ISF di Bologna. Da essa abbiamo ricevuto un supporto tecnico-logistico fondamentale per lo svolgimento dell'attività.

Di seguito sono presentati alcuni progetti in cui ISF è coinvolta.

- Nella città di Breza (Bosnia e Herzegovina), ISF Torino con l'aiuto di ISF Bologna è impegnata in due Progetti di Sviluppo. Il primo, in collaborazione con l'Associazione Alma Mater, è consistito nella costruzione di due serre con impianto di irrigazione a beneficio della associazione di donne Brezanke. Il secondo, in collaborazione con l'Associazione I.So.La di Torino, ha avuto come obiettivo, all'interno del Centro Giovani della Città, la realizzazione di un Internet Point a favore della Associazione giovanile di Breza "Desnek". In particolare, questo progetto prevede l'utilizzo di un software, scritto appositamente per la gestione di un Internet Point, e la formazione di base degli amministratori di rete locali. Il progetto, per ora, non è

stato portato a termine.

- Un altro progetto significativo che si colloca nell'ambito delle Tecnologie per l'Informazione e la Comunicazione (TIC), in partnership con l'associazione Centro Sviluppo Umano (CSU) di Viareggio e l'associazione Nasongdo, di Boulsa, Burkina Faso. Esso prevede l'installazione di un impianto satellitare VSAT, innanzitutto per la connessione ad Internet presso l'associazione. In progettazione vi è una rete wireless, da realizzare sfruttando tecnologie povere, da realizzare per dare connettività ad altre realtà vicine socialmente fondamentali (Ospedale, Comune, Dispensario, Provincia).

Capitolo 3

3 I Cluster

La parola "Cluster" (dall'inglese "grappolo") è usata nell'ambito informatico per riferirsi alla condivisione delle risorse di calcolo tra i nodi di una rete di macchine. Tipicamente un cluster integra le risorse di più computer che lavorano parallelamente, al fine di aumentare, a seconda dello scopo, la potenza di calcolo o l'affidabilità dell'intero sistema. L'idea che è alla base del clustering consiste nel distribuire il carico di lavoro su tutti i computer disponibili, utilizzando le risorse libere su ogni macchina del sistema.

Il singolo computer è l'unità base di un cluster ed è chiamato "**nodo**". I cluster possono crescere in dimensioni con l'aggiunta di più macchine, con una potenza ed una velocità dipendente da quella dei loro nodi e dalla velocità della rete. Per questa ragione un sistema di clustering deve saper utilizzare al meglio l'hardware anche al variare delle condizioni, deve cioè essere scalabile. Questo diventa una vera sfida quando il cluster è composto da tipi di hardware differenti (cluster eterogeneo), e quando la configurazione cambia in modo impredicibile, con macchine che si aggiungono o abbandonano il cluster.

Esistono tre tipi di cluster: Fail-over, Load-balancing ed High Performance Computing, con i primi due che sono probabilmente i più diffusi.

- ***Fail-over Cluster***: il funzionamento delle macchine è continuamente monitorato, e quando uno degli host smette di funzionare, un'altra macchina subentra. Lo scopo è garantire un servizio continuativo;
- ***Cluster con load-balancing***: è un sistema nel quale le richieste di lavoro sono inviate alla macchina con meno carico;
- ***HPC Cluster***: i computer sono configurati per fornire prestazioni estremamente alte. Le macchine suddividono i processi di un job su più macchine, al fine di guadagnare in prestazioni. La peculiarità saliente è che i processi sono parallelizzati e che le routines che possono girare separatamente saranno distribuite su macchine differenti invece di aspettare di essere eseguite una dopo l'altra. Gli HPC sono diffusi specialmente tra centri di elaborazione dati;

Il nostro studio si è focalizzato in particolare su due sistemi di cluster molto diversi tra di loro: openMosix ed LTSP. Molto diversi perché sono nati e si sono sviluppati per due scopi differenti. openMosix è un High-Performance Computing Cluster, perciò è stato pensato per fare calcoli fornendo alte prestazioni, mentre LTSP è un sistema Thin Client molto più orientato del primo alle normali applicazioni utente.

3.1 OpenMosix

Illustrazione 3: Logo di openMosix

OpenMosix è un software che consente di trasformare in un cluster una rete di computer interconnessi e dotati di sistema operativo GNU/Linux. Esso automaticamente ed in modo trasparente bilancia il carico tra i diversi nodi, e questi ultimi possono aggiungersi o lasciare il cluster senza che ci sia interruzione del servizio. Il carico è distribuito tra i nodi sulla base della velocità della loro CPU e della memoria disponibile.

OpenMosix è sviluppato sotto forma di una patch del Kernel Linux. Questo significa che un'applicazione utente, i file e le altre risorse continuano a funzionare senza alcun cambiamento di codice. L'utente casuale non noterà la differenza tra un sistema Linux semplice ed uno con openMosix; per questo, l'intero cluster funzionerà come un normale sistema GNU/Linux molto veloce. OpenMosix non ha un controllo centralizzato di tipo master/slave. Ogni nodo agisce come un sistema autonomo, le sue decisioni sono indipendenti e si basano su una conoscenza parziale degli altri nodi. Questa struttura consente di avere una configurazione molto dinamica nella quale nodi possono essere aggiunti o rimossi in modo trasparente, consentendo così un'agevole scalabilità del sistema e assicurando un funzionamento ottimale indipendentemente del numero dei nodi.

Per garantire la scalabilità del cluster, si adotta uno schema di diffusione delle informazioni sullo stato di ogni nodo di tipo probabilistico: ad intervalli regolari, ogni macchina del cluster invia ad un sottoinsieme casuale di altri nodi informazioni sulle risorse che ha a disposizione, mantenendo contemporaneamente in una tabella gli

ultimi valori ricevuti. Questo scambio avviene ogni secondo, utilizzando il protocollo UDP. Il funzionamento di openMosix si basa su un sistema di rilevazione dell'utilizzo delle risorse, un insieme di algoritmi adattativi per la gestione delle risorse ed un sistema di migrazione dei processi trasparente.

3.1.1 Sottosistemi di openMosix

Il sistema per la rilevazione dello stato del cluster è quello per cui ogni nodo è in grado di conoscere il reale stato di utilizzo delle risorse del sistema. Ciascuno, come detto, invia le informazioni sul suo stato di carico tramite UDP ad un numero di nodi scelti in modo casuale, e mantiene una tabella con le notizie che riceve dagli altri nodi. Le informazioni vengono aggiornate molto frequentemente, tipicamente ad ogni system call oppure ogni secondo. La creazione di un nuovo processo invalida tutte le informazioni visto che modifica il carico sui nodi. Gli algoritmi per la condivisione delle risorse sono gli strumenti che consentono ad openMosix l'ottimizzazione e il bilanciamento del carico.

Sulla base delle informazioni che i nodi si scambiano, un **algoritmo di load balancing** tenta di ridurre la differenza di carico tra le coppie di nodi, facendo migrare i processi dai nodi più carichi a quelli con più risorse disponibili. Questo sistema è decentralizzato e tutti i nodi utilizzano lo stesso algoritmo; la riduzione delle differenze di carico viene eseguita da coppie di nodi in modo autonomo. L'algoritmo determina le potenzialità di ogni nodo sulla base del numero delle CPU disponibili, della velocità della singola CPU. L'algoritmo è in grado di rispondere dinamicamente a variazioni del carico.

Il load balancing è affiancato da un altro algoritmo fondamentale per openMosix, che è quello di **memory ushering**, a cui è demandato il compito di bilanciare il carico di memoria dei nodi. Serve per evitare che ci sia un drastico peggioramento delle prestazioni a causa della mancanza di memoria libera. Quando un nodo inizia ad usare il file di swap, l'algoritmo va in esecuzione e tenta di migrare il processo verso un altro nodo con sufficiente memoria libera, anche se così facendo si dovesse ottenere un bilanciamento del carico non ottimale. Il bilanciamento della memoria infatti ha la priorità sul bilanciamento del carico della CPU.

Oltre a questi algoritmi, ce ne è un altro che serve a determinare su quale nodo un processo dovrebbe essere eseguito. Il modello matematico alla base di questo algoritmo deriva dalla scienza economica. **Determinare il nodo ottimo** non è una cosa semplice. La complicazione principale è che le risorse di ogni nodo sono eterogenee e non direttamente confrontabili, visto che non sono misurate nella stessa unità di misura. L'algoritmo utilizzato da openMosix è interessante in quanto tenta di ridurre queste differenze basandosi su principi economici di mercato. L'idea è, infatti, di associare a ciascuna risorsa un "costo" omogeneo e migrare i processi dove il costo è più basso. Questa strategia basata sull'economia più che sulle tecnologie informatiche, consente di sviluppare algoritmi che ottimizzano l'allocazione delle risorse nei nodi.

L'ultimo elemento che caratterizza fortemente il funzionamento di openMosix è il sistema di migrazione dei processi. Il meccanismo di **Preemptive Process Migration** (PPM) consente la migrazione trasparente dei processi, e possiamo dire che è quello che consente a openMosix di funzionare. Ogni processo ha un unico nodo di origine (UHN-Unique Home Node) in cui è stato creato; questo è il nodo da cui l'utente ha lanciato il processo. Finché l'utilizzo delle risorse del nodo rimane al di sotto di una certa soglia, il processo è

vincolato a rimanere sul nodo home; quando tale soglia viene superata, alcuni processi possono essere migrati verso altri nodi. I processi che vengono mandati ad un altro nodo sfruttano le risorse locali, ma continuano a interagire con l'ambiente dell'utente tramite l'UHN. L'obiettivo totale è quello di massimizzare l'efficienza dell'utilizzazione delle risorse dell'intera rete di calcolatori. Per implementare PPM occorre dividere i processi in due parti:

- La parte utente: è la parte che può essere migrata; in termini openMosix, si chiama anche "***remote***". Contiene anche l'eseguibile del programma, il suo stack, i dati, le mappe di memoria e i registri che competono al processo;
- La parte di sistema: è la parte che rimane nel nodo d'origine; si chiama per openMosix anche "***deputy***". É vincolata a rimanere nell'home node e non può essere migrata. Contiene una descrizione delle risorse che il programma sta utilizzando e un sistema di system call per conto del processo remoto.

I processi migrati verso un nodo non sono accessibili dagli altri processi che girano su di esso. L'unica cosa che un utente remoto può fare è forzare il processo a lasciare il nodo e migrare altrove. Il processo è in esecuzione immediatamente dopo il suo trasferimento, anche se non ha ancora tutte le pagine di memoria necessarie; le pagine di cui il processo ha bisogno sul nuovo nodo vengono trasferite fino a che sul nodo remoto non è stato ricostruito il working set del processo. Grazie alla tecnologia di "Demand paging" le pagine vengono trasferite solo quando sono effettivamente necessarie, riducendo così la dimensione del resident set.

La comunicazione tra deputy e remote avviene grazie a un protocollo molto leggero basato su UDP che usa porte oltre le prime 1024 (cioè non di sistema). L'interfaccia tra deputy e remote è ben definita, quindi è possibile intercettare tutte le interazioni tra le due parti e inviarle sulla rete. In tal modo è possibile captare tutte le system call

eseguite dal processo e quelle che non dipendono dal nodo sono eseguite in remoto senza ulteriori ritardi, mentre quelle che dipendono dal nodo di origine vengono trasferite al deputy che le esegue per conto del remote. Il deputy invia quindi i risultati a quest'ultimo che riprende l'esecuzione del codice utente. É in questo modo che si realizza la trasparenza nella migrazione dei processi. La divisione dei processi in deputy e remote comporta però un overhead nell'esecuzione delle system call e nella notifica dei segnali.

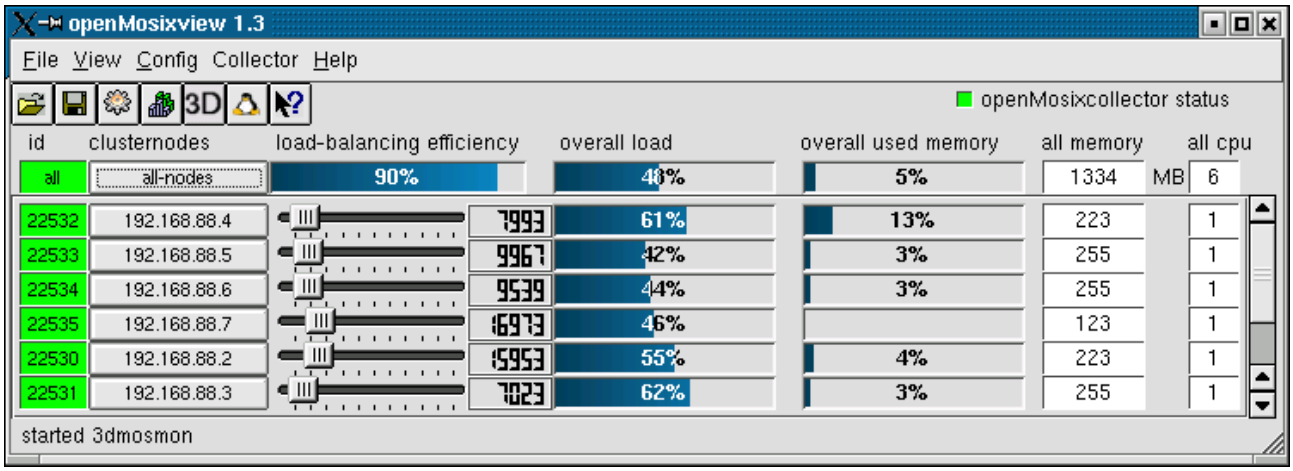

Illustrazione 4: Finestra principale di openMosixView

3.1.2 La gestione dell'I/O

L'I/O sui file eseguito via rete è un notevole collo di bottiglia, rallenta le prestazioni del nodo ed intasa la rete con traffico inutile, per questo è preferibile tenere tutte le operazioni di I/O locali. Così al contrario di tutti i network filesystem esistenti (per esempio NFS), che trasferiscono i dati attraverso la rete fino al nodo remoto in cui si trova il processo, il cluster openMosix tenta di migrare il processo nel nodo in cui il file effettivamente risiede. Questo ha richiesto l'utilizzo di un filesystem proprio: **openMosix filesystem** (oMFS). L'oMFS è

un filesystem del cluster e consente ad ogni nodo di accedere al filesystem di qualunque altro nodo come se fosse montato localmente; si occupa solo di rendere accessibili i filesystem ma non di utilizzarli in modo efficiente. OMFS è stato esteso con il **Direct filesystem Access** (DFSA) a partire dal kernel 2.4.26. Il DFSA è stato pensato per ridurre l'overhead dovuto all'esecuzione di operazioni di I/O da parte dei processi migrati. In aggiunta a questo filesystem per prevenire il problema di intasamento della rete, sono stati modificati gli algoritmi di load balancing in modo che un processo che deve eseguire molte operazioni sui file (processo I/O bound), venga migrato verso il nodo su cui deve compiere le suddette operazioni. I processi I/O bound hanno quindi maggiore flessibilità nel migrare dall'home node, per garantire il migliore utilizzo delle risorse dell'intero cluster.

Vediamo in conclusione quali sono i pro ed i contro di questo sistema di clustering.

***I Pro**:*

- Trasparenza: il sistema di migrazione dei processi è totalmente trasparente sia all'utente sia alle applicazioni; i processi standard di Linux vengono migrati verso i nodi più performanti ogni volta che è necessario;
- Nessuna ricompilazione del codice per le applicazioni: è possibile usare le normali applicazioni che funzionano su una singola macchina senza doverle riscrivere o ricompilare;
- Nessuna manutenzione aggiuntiva: una volta che il cluster è stato configurato tutte le operazioni sono automatiche e non richiedono interventi da parte dellutente.

***I Contro**:*

- Le applicazioni composte da un solo processo non traggono vantaggi dall'uso del cluster, eccetto ovviamente per il fatto che quel processo sarà migrato sul nodo più performante;
- Non tutti i processi possono essere migrati: alcuni tipi di processi sono vincolati al proprio home node e non possono essere migrati, come i processi con necessità real-time e i processi che hanno accesso diretto ai dispositivi di I/O e soprattutto i processi a memoria condivisa;
- La minima unità di lavoro di openMosix è il processo: questo significa i thread non vengono migrati ad eccezione dei Java Green Thread e degli FSU Pthread. Si noti che la versione in attuale sviluppo di openMosix (2.6) riuscirà a migrare anche i singoli thread.

Il fatto che openMosix non consenta la migrazione dei processi a memoria condivisa è stato uno dei problemi più grandi che abbiamo dovuto affrontare nelle nostre ricerche e nei nostri test. La nostra intenzione, come detto, è sempre stata quella di usare openMosix per scopi diversi rispetto a quelli per cui era stato progettato (HPC), cioè per fare in modo che le normali applicazioni utente traessero vantaggio dal girare su un cluster piuttosto che su un singolo computer. Il problema, per noi, nasce nel momento in cui abbiamo verificato che molte delle applicazioni che volevamo far funzionare erano a memoria condivisa. Vediamo ora come funziona la condivisione di memoria nel sistema GNU/Linux e su quale soluzione sperimentale sta lavorando la comunità openMosix per ovviare a questo inconveniente.

3.1.3 La memoria condivisa in Linux

Nel sistema Linux la condivisione della memoria è gestita utilizzando i meccanismi di comunicazione tra processi implementati nello standard UNIX System V; esso consente a due o più processi di accedere ad alcune strutture di dati comuni inserendole all'interno di un segmento di memoria. La memoria condivisa in Linux è una porzione di memoria contigua accessibile o in lettura o in scrittura da un certo numero di processi che abbiano gli appropriati permessi di accesso.

Normalmente, quando in openMosix un processo migra su un nodo remoto, la sua mappa di memoria viene distrutta e ricreata li dove il processo è migrato: ogni porzione di memoria condivisa ha un identificatore unico all'interno del sistema denominato chiave. Tale chiave è lo strumento che i processi utilizzano per accedere alla memoria condivisa. L'accesso avviene nel seguente modo: un processo apre la memoria condivisa attraverso la chiave (funzione `shmget()`) e l'attacca al proprio spazio di indirizzamento (funzione `shmat()`), da dove la staccherà quando non ne avrà più bisogno (funzione `shmdt()`). Per eseguire operazioni di controllo su una porzione di memoria condivisa si usa la funzione `shmctl()`.

Ora che abbiamo visto come la memoria condivisa è gestita nel sistema GNU/Linux andiamo ad analizzare la soluzione (ancora in fase di sperimentazione) che viene proposta alla comunità openMosix.

3.1.4 Migshm

Migshm (Migration of Shared Memory) è una patch del kernel addizionale a quella di openMosix. É stata creata da 5 studentesse di ingegneria dell'Università di Pune in India (il team MAASK), per cercare di risolvere uno dei problemi maggiori di openMosix, cioè, come detto, consentire la migrazione di processi che condividono la memoria. La patch è ancora in fase di sperimentazione, ed ancora non si sa quando verrà inserita stabilmente nella patch ufficiale openMosix. Data la scarsa stabilità che è emersa nei test di tale patch, il nostro team ha deciso di non includerla nel package definitivo, tuttavia essa desta un interesse accademico rilevante, che ne giustifica la trattazione. Migshm consiste sostanzialmente dei seguenti moduli di kernel:

- ***Migrazione dei processi a memoria condivisa***. É il più importante: consente ai processi a memoria condivisa di essere presi per la migrazione durante la loro esecuzione. Per fare questo, si deve innanzitutto fare in modo che openMosix riconosca tali processi e li consideri come migrabili: si usa un identificatore e un flag che dica al cluster se il processo è fortemente o debolmente legato alla regione di memoria condivisa. Esso garantisce che le system call possano essere fatte trasparentemente da remoto quando i processi migrano.
- ***Communication Module***. Quando un processo migra su un cluster openMosix è richiesto un canale di comunicazione tra la parte deputy e la parte remote, questo canale non può essere utilizzato, però, quando due processi entrambi remote devono comunicare tra di loro. Per superare questo problema, migshm prevede un demone MigShareMemD che è attivato su ogni nodo.

- ***Consistency Module***. Quando processi che utilizzano memoria condivisa migrano su differenti nodi, lavorano su copie locali della memoria condivisa. Così è necessario mantenere la consistenza tra le differenti copie. É necessario assicurare che venga letta sempre l'ultima copia dei dati e che le modifiche fatte in scrittura vengano replicate agli altri processi remoti. Il modello di consistenza è il cosiddetto "Eager release consistency": un processo propaga le modifiche solo quando rilascia la zona di memoria che condivide con gli altri processi. Il messaggio di invalidate (col quale si rende noto che i dati non sono più aggiornati) è gestito in remoto da un daemon (il mig-shm-daemon) che gira su ogni nodo. Ad ogni invalidate una nuova versione aggiornata della pagina di memoria deve essere trasferita dal nodo home attraverso la rete. Allo scopo di massimizzare gli accessi locali, la memoria condivisa è migrata con un processo, a patto che esso sia strettamente vincolato alla memoria stessa.
- ***Access log and migration decisions***. Registra gli accessi alle pagine di memoria condivisa. Con queste informazioni viene arricchito l'algoritmo di load-balancing openMosix, il quale usa un demone chiamato memsorter per raccoglie le informazioni; migshm ha esteso le funzionalità di quel demone. Viene utilizzato un counter per processo (incrementato secondo un particolare algoritmo), il quale conta il numero di accessi. Quando il contatore supera una certa soglia, il processo verrà considerato fortemente o debolmente legato alla regione di memoria condivisa (in base al valore del contatore) e di conseguenza la memoria verrà migrata con il processo o meno, incrementando così il numero degli accessi locali e riducendo quelli remoti.
- ***Migration of shared memory***. Gestisce la migrazione della regione condivisa. Quando una regione di memoria viene

migrata con il processo che è fortemente legato ad essa, il nuovo nodo diventa l'owner della memoria. Il possessore precedente notifica il cambiamento a tutti i nodi in cui si trovano processi che condividono quella regione di memoria, questi manderanno i messaggi di writeback al nuovo owner.

Chiudiamo così la trattazione teorica di migshm, sottolineando ancora una volta come questa patch di openMosix sia assolutamente ancora in una fase sperimentale. Il suo sviluppo sarà molto importante e ci permetterà di dotare il cluster di uno strumento molto potente che ci consentirà auspicabilmente di colmare quella che in questo momento è forse la più grande lacuna di openMosix: l'impossibilità di migrare applicazioni a memoria condivisa.

Accanto ad openMosix, abbiamo studiato e sperimentato un altro sistema per la creazione e la gestione di un cluster: LTSP. Questo sistema è pensato in un modo che potremmo dire diametralmente opposto a quello di openMosix. Vediamo nel dettaglio come funziona LTSP e poi cercheremo di fare un confronto tra i due sistemi, per poter comprendere come una loro integrazione possa rivelarsi un vantaggio per le finalità della nostra ricerca.

3.2 LTSP

LTSP (Linux Terminal Server Project) è un sistema Thin Client, cioè un sistema che consiste di un certo numero di workstation diskless che sono connesse ad un server. Tutte le applicazioni girano sul server, che realizza le funzioni di calcolo e di gestione dei dati. Le workstation sostanzialmente forniscono all'utente tastiera, video e mouse per l'accesso alle applicazioni ed ai dati.

La illustrazione 6 mostra la struttura del cluster LTSP: un nodo master, che è il server che svolge tutte le operazioni del sistema, e diverse macchine destinate agli utenti, le quali si comportano come semplici terminali.

Il motivo principale per cui LTSP è stato scelto per il progetto, è che esso fornisce un modo semplice per riutilizzare computer a basso costo come terminali grafici, collegandoli ad un server GNU/Linux. Il progetto LTSP, ideato da James A. McQuillan, è stato pensato e realizzato per tutte le situazioni in cui c'è bisogno di un numero elevato di computer, ma si hanno a disposizione poche risorse. Usando LTSP infatti si possono acquistare macchine datate, a basso costo, che abbiano una scheda di rete come unico prerequisito, ed usarle come comuni workstation.

Per semplificare molto il funzionamento di LTSP accade ciò che è descritto di seguito.

Il terminale si accende e carica dal floppy (o dalla rom della scheda di rete) il sofware (circa 16 Kbyte) che gli permette di contattare il server, dal quale scarica il kernel che deve lanciare per partire.

Fatto il boot, il terminale carica via NFS (Network File System) il suo filesystem di root. Questa procedura serve a eseguire il programma di login sul terminale. Il filesystem montato temporaneamente (4 Mb) viene caricato in RAM.

La workstation può essere configurata in tre modi:

- Grafico: si usa X window. La workstation può essere usata per lanciare qualsiasi applicazione sul server di partenza, o su qualunque altro server della rete.
- Testo basato su sessioni telnet: la workstation può collegarsi in telnet sul server. Ogni sessione sarà messa su un differente schermo virtuale per un totale di 9 sessioni separate.
- Prompt dei comandi: questa modalità è molto utile durante la

fase di configurazione per risolvere i vari problemi che possono presentarsi.

Partito X, il terminale grafico fa una query al server, chiedendo di far partire il desktop manager (XDM, KDM o GDM) Appare quindi la schermata con login e password (la stessa schermata proposta quando il sistema parte normalmente). Dopo l'autenticazione si possono lanciare le applicazioni, come se ci si trovasse direttamente sul server.

Al termine della sessione, dopo il logout il terminale può rimanere acceso in attesa di un altro utente, oppure può essere spento privandolo direttamente di alimentazione, poichè non essendo presenti dischi, il filesystem non può essere danneggiato.

É interessante che le prestazioni di tutte le workstation della rete dipendono solamente dalle prestazioni del server centrale (con disco) e non dalle potenze di calcolo delle singole macchine.

3.2.1 Teoria del funzionamento

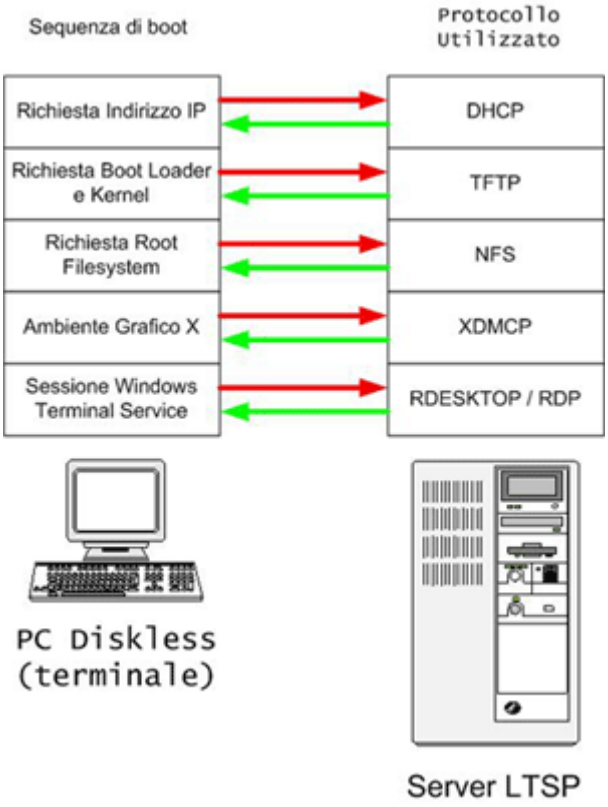

Illustrazione 5: Fasi del boot e servizi relativi

Far partire una workstation senza disco include diverse fasi. La caratteristica peculiare di LTSP consiste nel meccanismo con cui i terminali effettuano lo startup, come è dettagliatamente trattato in seguito. D'altra parte, a permettere il funzionamento del sistema sono tutti i diversi servizi che il server offre alle workstation. Per una trattazione dettagliata del funzionamento e della configurazione di tali servizi rimandiamo all'Appendice B.

Supponendo che il server abbia il pacchetto LTSP installato, nel momento in cui accendiamo la workstation, questa inizierà il suo test hardware, chiamato **POST** (Power On Self Test). Durante il POST, il BIOS cerca le eventuali ROM di espansione. La scheda di rete può contenerne una, la eeprom Etherboot. Il BIOS ne prenderà atto e una volta che il POST sarà terminato, verrà eseguito il **codice Etherboot** contenuto nella eeprom.

Il codice Etherboot cercherà una scheda di rete e una volta trovata, questa verra inizializzata.

Il codice Etherboot farà quindi una **richiesta DHCP** sulla rete locale, includendovi il MAC address della scheda di rete.

Il processo dhcpd, sul server, leggerà la richiesta e cercherà nel suo file di configurazione la presenza del MAC address corrispondente; lo stesso processo costruirà in seguito un pacchetto di risposta contenente molte informazioni. Questo pacchetto sarà inviato indietro alla workstation ed includerà:

1. Indirizzo IP per la workstation
2. Settaggio NETMASK per la rete locale
3. Directory e nome del kernel da scaricare (dal server)
4. Directory e nome del filesystem NFS da montare come root
5. Parametri opzionali da passare al kernel, tramite l'interfaccia comandi standard del kernel di Linux.

Il codice Etherboot riceverà la risposta del server, e configurerà l'interfaccia TCP/IP nella scheda di rete con i parametri ricevuti.

Usando il protocollo TFTP (Trivial File Transfer Protocol), il codice Etherboot contatterà il server e **scaricherà il kernel**.

Una volta che il kernel sarà stato completamente scaricato sulla workstation, il codice Etherboot lo metterà nella corretta locazione di memoria; dopodiché il controllo verrà passato al kernel, che inizializzerà l'intero sistema e tutte le periferiche che potrà riconoscere. Attaccata alla fine del kernel, c'è l'immagine di un filesystem. Questa verrà caricata in memoria come se fosse un normale ramdisk, e temporaneamente montata come root filesystem. Un parametro del kernel (`root=/dev/ram0`) gli dirà di montare l'immagine come filesystem di root.

Normalmente, quando il kernel finisce l'operazione di boot, viene

lanciato il programma init. In questo caso invece, istruiremo il kernel per fare altrimenti, e lanciare un nostro script, tramite un altro parametro (`init=/`**`linuxrc`**).

Lo script `/linuxrc` eseguirà una scansione del bus PCI cercando una scheda di rete. Ogni volta che ne troverà una, esso guarderà nel file `/etc/niclist` per vedere se corrisponde. Trovata una corrispondenza, il nome del modulo driver verrà stabilito e caricato. Nel caso di schede di rete ISA, questo driver dovrà essere specificato come parametro del kernel, assieme a qualunque parametro aggiuntivo richiesto, come indirizzi IRQ ecc.

Il programma `dhclient` verrà quindi eseguito per fare un'**altra richiesta al server DHCP**. Saremo costretti a fare questa richiesta perché la prima è utilizzata solo dal kernel, che ignorerà qualsiasi opzione specificata via DHCP (server NFS, ecc.), mentre la seconda configurerà il sistema generale. Questo è particolarmente importante nel caso si vorrà usare un server NFS diverso dal server TFTP.

Quando dhclient riceverà una risposta dal server, verrà eseguito il file `/etc/dhclient-script`, verranno recuperate le informazioni e utilizzando queste la scheda eth0 verrà configurata.

Fino ad allora, il filesystem di root sarà stato un ramdisk. A questo punto, lo script `/linuxrc` **monterà un nuovo filesystem su root attraverso NFS**. La directory che verrà esportata per questo scopo sarà generalmente `/opt/ltsp/i386`. Ma non sarà possibile montare semplicemente il nuovo filesystem su `/`. Si dovrà infatti prima montarlo su `/mnt`. Eseguire quindi un'operazione chiamata pivot_root. pivot_root scambierà il root filesystem corrente con un nuovo filesystem. Quando l'operazione sarà ultimata, il filesystem NFS sarà montato su `/`, ed il vecchio ramdisk su `/oldroot`.

Una volta che il filesystem NFS sarà montato, il compito di `/linuxrc` sarà finito, e potrà essere e**seguito il vero programma init**.

Init leggerà il file `/etc/inittab` ed inizierà a configurare l'ambiente della workstation.

Il processo init è caratterizzato da un runlevel. Ogni runlevel fa partire diversi set di servizi per la workstation. Le workstation LTSP partono di default con runlevel 2. Questa opzione potrà essere settata dalla riga initdefault nel file `inittab`.

Una delle prime operazioni che `inittab` eseguirà, sarà lanciare il **comando `rc.sysinit`** quando la workstation è nello stato `sysinit`.

Lo script `rc.sysinit` creerà un ramdisk di 1Mb che dovrà contenere tutti i file che necessiteranno di essere modificati durante il normale funzionamento. Il ramdisk verra montato sulla directory `/tmp`. Ogni file che dovrà essere scritto sarà messo qui, ed alcuni link provvederanno a far sì che questi file si trovino anche nella giusta collocazione nella root directory. Verrà poi montato il filesystem `/proc`.

Se la workstation è stata configurata per eseguire lo swap della RAM su NFS, la directory `/var/opt/ltsp/swapfiles` verrà montata su `/tmp/swapfiles`. Quindi, se qui non sarà presente nessun file di swap, ne verrà automaticamente creato uno. La grandezza di questo file viene stabilita all'interno del file `lts.conf`.

Il file di swap verrà infine abilitato tramite il comando `swapon`.

L'interfaccia di loopback per la rete verrà ora configurata. Questa rappresenta la scheda di rete virtuale che ha 127.0.0.1 come suo indirizzo IP.

Se le applicazioni locali (localapps) sono abilitate, la directory `/home` verrà montata, in modo che le varie applicazioni potranno accedere alle directory degli utenti.

Alcune directory verranno create in `/tmp` per contenere alcuni file temporanei che sono necessari quando la workstation lavora, ad

esempio: /tmp/compiled, /tmp/var, /tmp/var/run, /tmp/var/log, /tmp/var/lock, /tmp/var/lock/subsys.

Poi verrà creato il file /tmp/syslog.conf che conterrà informazioni che diranno al demone syslogd come e a quale host della rete inviare i dati di log (l'host è specificato sempre all'interno del file lts.conf). Esiste un link chiamato /etc/syslog.conf che punta al file /tmp/syslog.conf. Così il processo syslogd verrà fatto partire.

Una volta terminata l'esecuzione di rc.sysinit, il controllo ritornerà a init che porterà il runlevel da *sysinit* a 5. Questo causerà l'esecuzione di alcune entry nel file inittab.

Di default, in inittab ci sono delle entry che servono a lanciare lo script /etc/screen_session su tty1, tty2 e tty3. Questo significa che si possono eseguire tre sessioni alla volta, e il tipo di sessione è definito dalle entry SCREEN_01, SCREEN_02 e SCREEN_03 in lts.conf. Se si desiderano più sessioni è sufficiente configurare secondo le proprie esigenza il file inittab.

Se SCREEN_01 è settato con il valore "startx", allora lo script /etc/screen.s/startx lancerà l'**X Windows System**, rendendo disponibile un'interfaccia grafica. Nel file lts.conf, c'è un parametro chiamato XSERVER. Se questo parametro è mancante oppure settato su "auto", verrà tentata la ricerca automatica della scheda video. Se questa è una scheda PCI, otterremo il codice PCI Vendor e Device ID, e vedremo se questa scheda è presente nel file /etc/vidlist. Se la scheda è supportata da Xorg6.7, la routine pci_scan ritornerà il nome del modulo driver da caricare. Se la scheda è invece solo supportata da XFree86 3.3.6, pci_scan ritornerà il nome del server X da usare. Lo script startx potrà distinguere le differenze dell'output del comando: i moduli del server X in versione 3.3.6 iniziavano tutti con "XF86_", mentre i più recenti moduli del server Xorg sono tipicamente nomi lowercase, come "*ati*"

o “*trident .* Se viene utilizzato Xorg, verrà richiamato lo script `/etc/build_x4_cfg` che genererà il file di configurazione XF86Config per X4. Se invece viene usato XFree86 3.3.6, verrà richiamato lo script `/etc/build_x3_cfg` per produrre lo stesso file, ma in formato X3. Questi file sono posti nella directory `/tmp` che è il ramdisk visto solo dalla workstation. Il file XF86Config verrà creato in base alle informazioni contenute nel file di configurazione `/etc/lts.conf`. Dopo la creazione del file XF86Config, lo script `startx` lancerà il server X con quel nuovo file di configurazione.

Il server X manderà una query XDMCP al server, che offrirà una finestra di login. A questo punto l'utente può loggarsi, ottenendo una sessione sul server [7].

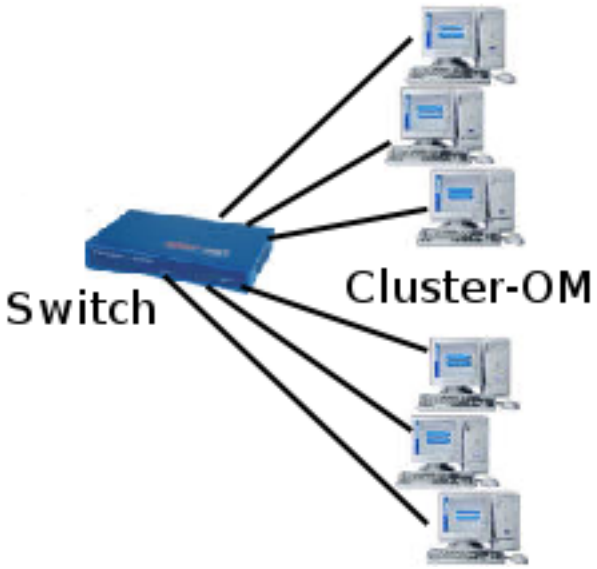

Illustrazione 5: Tipica configurazione del cluster openMosix

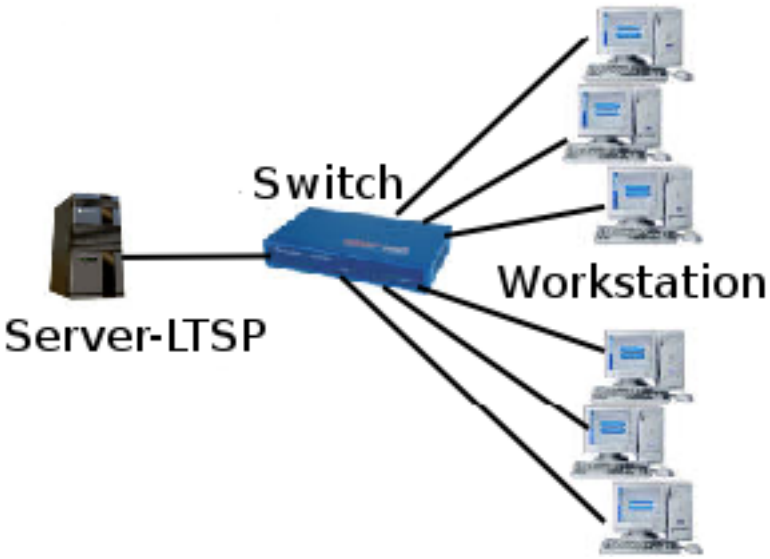

Illustrazione 6: Tipica configurazione di un cluster LTSP single server

3.3 Considerazioni sul bilancio energetico

L'attività di trashware e la pratica del riuso dell'hardware pongono in rilevanza l'aspetto del consumo di energia. In effetti, in generale, l'hardware obsoleto ha consumi elettrici ridotti, ma costruire un cluster implica che tali consumi vadano moltiplicati per il numero dei nodi, cosa che potrebbe rendere energeticamente svantaggioso (in prima analisi) usare delle macchine povere. L'hardware datato contiene processori meno esosi in termini di alimentazione, quindi richiede meno energie per essere raffreddato; inoltre le altre unità di elaborazione che incidono con una percentuale elevata sul consumo totale, come le schede video e le memorie, essendo meno esigenti riducono ulteriormente il dispendio elettrico. Resta da valutare come incidiamo sul bilancio energetico formando un cluster, rispetto all'uso di una singola macchina.

I fattori in gioco nel bilancio generico, oltre al consumo elettrico di ogni singolo pezzo, sono i seguenti:

- **costo di costruzione** di una nuova macchina performante
- **costo di dismissione** degli apparati
- **numero di utenti** che usufruiscono del sistema
- **tempo di vita** produttivo di una macchina performante

Tali fattori sono da analizzare da un punto di vista sia energetico che economico. Inoltre le considerazioni che verranno fatte saranno da valutare sia nel contesto della collettività, sia nel contesto più ristretto di una qualsiasi forma aggregativa di persone (associazioni, aziende, ecc ecc).

3.3.1 Considerazioni rispetto alla collettività

Iniziando ad analizzare la situazione dei consumi dal punto di vista della collettività, dobbiamo innanzitutto sottolineare l'enorme quantità di risorse energetiche che vengono utilizzate per la costruzione di un computer. Per la nostra analisi si è deciso di utilizzare come unità di confronto i chilogrammi di petrolio, poiché possiamo convertire in tale unità sia l'energia usata per far funzionare gli apparati che producono la macchina, sia per farla funzionare durante il suo tempo di vita. In ogni caso non sono da sottovalutare anche altre risorse che occorrono alla costruzione di un PC, come l'acqua e i diversi tipi di metalli.

Si è constatato che per la costruzione di un computer vengono utilizzati circa 240 Kg di petrolio, 1500 Kg di acqua, e 22 Kg di svariate sostanze chimiche. Si è stimato poi il consumo elettrico di una macchina performante e di una macchina obsoleta, di circa due o tre generazioni antecedente alla prima. Dai nostri studi risultano 289 W consumati da una macchina nuova e 156 W da una macchina obsoleta. Si sono inoltre lasciati in disparte i consumi dei monitor (poiché utilizzati in entrambe le formazioni).

Si è in seguito passati a calcolare il tempo in ore impiegato dalle due diverse architetture per consumare 1 KWh; si è calcolato che l'architettura moderna impiega 3.46 ore, mentre quella obsoleta ne impiega 6.41. Ciò equivale a dire che in un giorno lavorativo di otto ore, la macchina nuova consuma 2.312 KWh mentre quella obsoleta consuma 1.248 KWh. A questo punto è possibile trasferire il riferimento dai KWh ai Kg di petrolio. Con una semplice equivalenza ($1\ KWh = 0.22 \cdot 10^{-3}$ tep) si è potuto stimare che in un giorno lavorativo il PC nuovo consuma 0.509 Kg di petrolio, mentre quello obsoleto ne consuma 0.275. Se ora si moltiplicano i Kg di petrolio usati giornalmente dalle due macchine, per il numero medio di giorni

lavorativi annuali, stimati a 241 (escludendo sabati, domeniche e festività), si ha che la macchina nuova consuma 122.582 Kg di petrolio e quella obsoleta consuma 65.894 Kg.

Naturalmente non è corretto continuare la trattazione paragonando un solo PC obsoleto ad un PC nuovo, in quanto il primo non reggerebbe da solo il confronto dal punto di vista dell'efficienza. Lo regge invece il cluster openMosix, composto da più macchine (nel nostro laboratorio 4). Avendo ora a confronto la macchina performante ed il cluster di 4 macchine obsolete, e tenendo presente la legge di Moore (che impone il cambio generazionale della tecnologia elettronica ogni 1.5 anni) si potrà notare che il cluster avrà un consumo di petrolio per 1.5 anni di lavoro pari 395.37 Kg (65.894 Kg x 4 PC x 1.5 anni), contro i 183.87 Kg.

La differenza dei due dati sembra indicare un risparmio di 211.49 Kg di petrolio, tramite l'utilizzo di una macchina di ultima generazione. Considerando però che, scegliendo il cluster, la macchina nuova non viene usata e neanche prodotta, si avrà un risparmio di 240 Kg di petrolio (necessari per la creazione di quest'ultima) ed un conseguente bilancio nel **risparmio globale di circa 30 Kg di petrolio a favore del cluster**. Ciò è vero senza tener conto del risparmio di acqua e di molte altre materie prime, che avvantaggerebbero ulteriormente il cluster. In conclusione, il riutilizzo di hardware obsoleto porta ad un risparmio energetico per l'intero pianeta.

3.3.2 Considerazioni economiche

Si vuole ora passare ad una analisi di tipo più “materiale”,

tralasciando il vantaggio che trae generalmente l'ambiente dalla non produzione di una nuova macchina, e considerando il vantaggio economico che può trarre un'aggregazione di persone, sia essa un'azienda, un'associazione, o una comunità in via di sviluppo. Il ragionamento è stato dapprima sviluppato prendendo ad esempio un'azienda, poi ampliato ad una qualsiasi comunità.

Si è stimato il costo medio dell'energia elettrica, su dati Enel[4], corrispondente a 0.13 €/Kwh. Si è poi considerato il prezzo di un PC di ultima generazione pari ai 700 €. Per quel che riguarda il costo del cluster si è voluto considerare un prezzo di 50 € a macchina[5], che conduce ad una spesa di 200 € per la costruzione dell'intera struttura, tralasciando il costi irrisori di cavi ethernet e switch di rete.

Prendendo anche qui come riferimento la legge di Moore, che impone un salto generazionale ogni 1.5 anni, e mantenendo valide le considerazioni fatte precedentemente sui consumi delle due architetture, si può stimare il consumo energetico del cluster, per 1.5 anni di lavoro, in 239 €, ed il consumo della macchina performante in 110 €.

Sommando a questi consumi, i prezzi di acquisto precedentemente descritti, si avrà una spesa totale per il sostentamento del cluster per 1.5 anni di lavoro pari a 439 € ed una spesa per la singola macchina pari a 810 €.

Si può quindi affermare, alla luce di tale ragionamento, che mantenere operativo un cluster di macchine obsolete costa sicuramente più di mantenere operativa una singola macchina performante; ma il costo di acquisto, davvero minimo, del primo porta comunque ad un **risparmio notevole**, di circa 370 € **nel caso si scelga la soluzione di clustering**.

Si può ora estendere il ragionamento ad una situazione più generica, come quella di una scuola o di una associazione, realtà che

4 Costo ENEL supponendo 2640 KWh massimi annui, nella fascia oraria lavorativa (7:00-20:00 Lun-Ven)
5 Costo basato sui dati forniti da RaccattaRAEE, Bologna

solitamente non hanno la possibilità di investire ingenti quantità di denaro in innovazione tecnologica. Tali soggetti hanno spesso l'occasione di ricevere da associazioni che si occupano di trashware macchine obsolete a costo zero. In questa situazione il risparmio che deriva dall'utilizzo del cluster è ancora maggiore, in quanto si avrà un costo di attività pari al solo consumo elettrico.

Nelle considerazioni fatte non si tiene conto del fatto che la sostituzione di una macchina vecchia con una macchina nuova produrrebbe in aggiunta dei costi di smaltimento ingenti, che vanno ulteriormente a sfavore della scelta della sostituzione.

Volendo ora ampliare ulteriormente il ragionamento, si può analizzare realtà nelle quali la sostituzione del parco macchine non avviene seguendo la legge di Moore, ma seguendo un ritmo più blando, ad esempio di un cambio ogni 3 o persino 5 anni. Realizzando nuovamente il ragionamento ed i calcoli sopra descritti, è possibile affermare che l'utilizzo del cluster povero porta ad un risparmio notevole, di circa 250 €, anche nel caso della sostituzione delle macchine ogni 3 anni e di un risparmio più irrisorio, ma comunque non trascurabile (circa 75 €), nel caso di utilizzo del cluster per ben 5 anni.

Capitolo 4

4 Linux Terminal Clustered Server Project

Il presente progetto di tesi si è focalizzato sull'idea di far interagire i due sistemi sopra descritti. Di qui la sostituzione del termine “Server” con “Clustered Server” nel nome del progetto originale LTSP. La motivazione che ha spinto all'integrazione di LTSP con openMosix, è stata quella di voler creare un sistema completo e autonomo, reperendo risorse fisiche e computazionali da soli hardware recuperati. In effetti, i sistemi di terminali più diffusi fanno uso di server molto potenti, capaci cioè di gestire da soli il totale del carico di lavoro che tutto il cluster delle workstation richiede. Tuttavia, ciò implica che nella scelta della macchina server, normalmente bisogna considerare un carico di lavoro circa pari alle esigenze degli applicativi che verranno utilizzati, moltiplicato per il numero delle workstation. É quindi impensabile poter svolgere la mansione di server con una macchina dall'hardware poco sofisticato, che invece è un vincolo del nostro progetto. Si è ipotizzato, allora, che la quantità di risorse necessarie potesse essere raggiunta formando un cluster di macchine poco performanti, ovvero già obsolete, che cercasse di raggiungere in prestazioni un single-server: si è utilizzato quindi un sistema di clustering HPC (High Power Computing) non per

migliorare l'efficienza di diverse macchine già potenti, ma per raggiungere la quantità di risorse necessarie a un sistema di terminali, usando macchine povere. Per formare il cluster si è ricorsi a openMosix.

4.1 Architettura e funzionamento di LTCSP

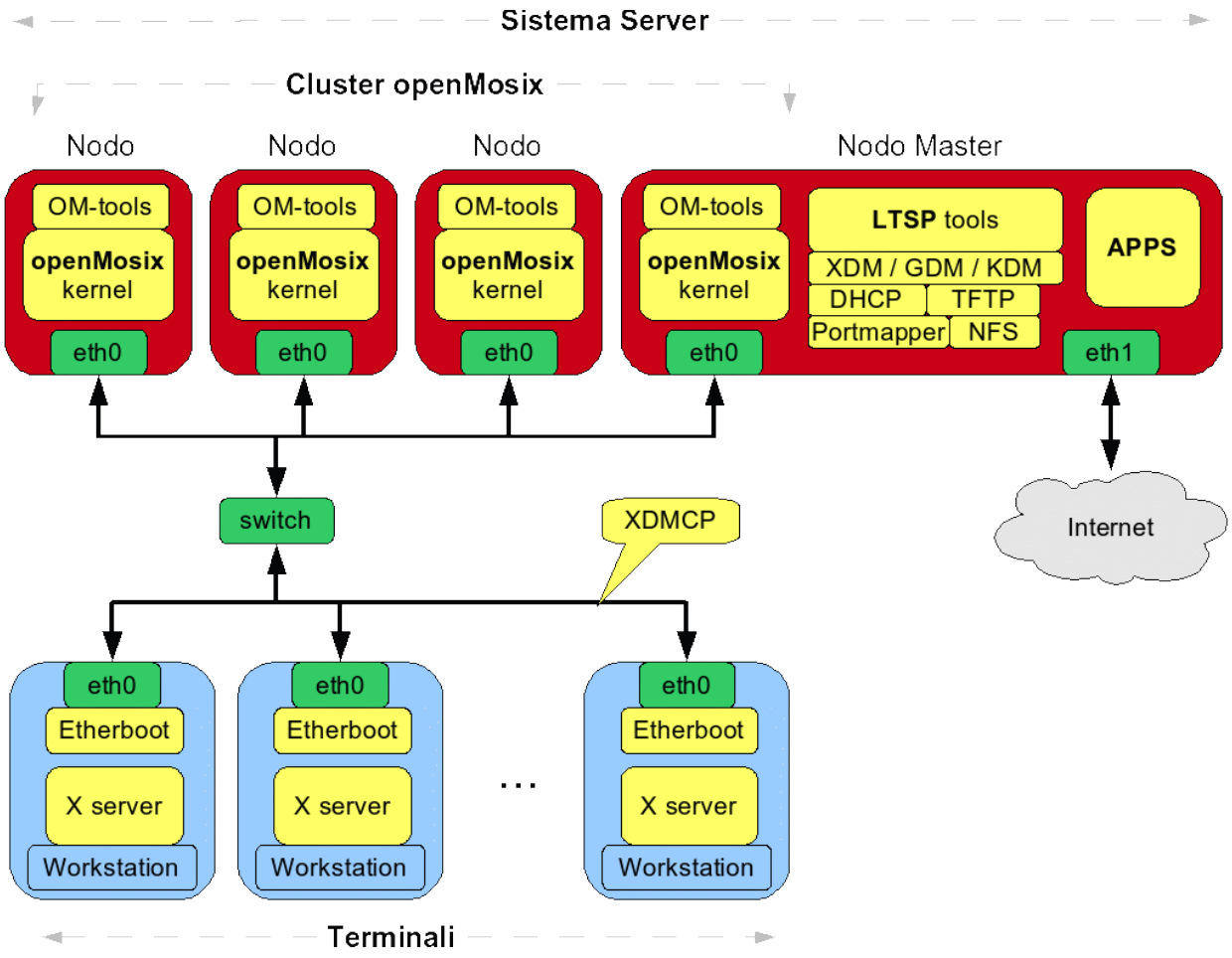

Illustrazione 7: Architettura di LTCSP

Alla luce di quanto detto precedentemente, il sistema completo (**openMosix + LTSP**) non ha bisogno in alcuna parte di risorse tecniche particolari, essendo formato dai seguenti componenti:

- ***Il cluster di server***, di cui uno è il server master, mentre gli altri eseguono pura elaborazione. Tutto il software è installato sul nodo master, che fornisce i servizi vitali al sistema (ovvero permette ai terminali di eseguire lo startup e rendersi operativi), e gli applicativi, che le workstation eseguiranno in remoto. Sugli altri nodi del server sono installati il solo sistema operativo (il kernel che esso usa è un kernel openMosix) e gli *userland tools* openMosix. Il kernel permette la parallelizzazione senza aver bisogno del software di cui sta eseguendo il processo, ma delle sole pagine di memoria coinvolte, come ben spiegato nel Capitolo 3.1. A parte il nodo master, nessun nodo necessita di configurazione, poiché sono i tools spazio utente che configurano real-time il sistema.
- ***Diversi terminali diskless***, che usufruiscono dei servizi. Sono terminali classici: forniscono video e periferiche di input, ed eseguono lo startup caricando il kernel dal server, tramite il boot da un dispositivo locale (si veda l'Appendice B). Una volta eseguito il boot del terminale, si opererà come se si fosse sulla macchina server.

L'interesse che LTSP e openMosix destano, per quanto concerne la nostra scelta di farli convivere insieme, è giustificato dal fatto che essi partizionano ottimamente le loro aree di funzionamento, non interferendo così in modo distruttivo. Il motivo per cui ciò avviene, risiede nel fatto che LTSP è, in ultima analisi, una collezione di tool e servizi quasi autonomi, che eseguono in spazio utente, mentre openMosix è una patch del kernelLinux, che per l'appunto, esegue in spazio kernel. La parallelizzazione delle applicazioni è effettuata dal kernel, e come si è detto, è questo il punto di forza fondamentale di openMosix. I sistemi attualmente sviluppati che forniscono supporto al clustering, invece, fanno uso di librerie personalizzate, che chi sviluppa software deve utilizzare esplicitamente, oppure devono creare nativamente dei processi tali che piattaforme specifiche sanno

parallelizzare.

4.2 Scalabilità

Esaminiamo ora gli aspetti che influenzano la scalabilità del sistema LTCSP: in un sistema a terminali essa è una caratteristica centrale. Si noti come essa interessi solamente le caratteristiche dei nodi server: i nodi diskless contano per il loro numero (che implica computazione per il cluster), mentre quasi per nulla per le prestazioni: eseguendo del codice di startup minimo, e delegando i server per ogni operazione, essi hanno bisogno di risorse di calcolo limitatissime (si parla di CPU 486 con 8 MB di RAM), e di una scheda grafica che supporti la risoluzione del monitor utilizzato [8].

OpenMosix possiede due strumenti che consentono a una macchina di unirsi al cluster dinamicamente. Essi sono il demone `openMosixInfoDaemon`, il quale si occupa di aggiornare ogni secondo lo stato del nodo su cui risiede agli altri componenti del cluster, mandando pacchetti UDP multicast, e il demone `omdiscd`, che, attraverso pacchetti UDP broadcast spediti a intervalli, scopre l'eventuale aggiunta di un altro nodo al cluster, svincolando l'amministratore dalla configurazione manuale dei server, e soprattutto permettendo ad un nodo che sia andato in *service down* di riunirsi agli altri con un semplice reboot. Tutto ciò avviene ad un costo di prestazioni relativamente basso: le analisi del traffico di rete, effettuate su ogni coppia di nodi (utilizzando lo sniffer open source testuale `iptraf`[6]) considerando i messaggi scambiati dai demoni prima citati, il traffico è di **276 Kbyte al secondo**. É una percentuale ridotta del throughput che sopporta l'hardware di rete, supponendo

6 Si veda http://cebu.mozcom.com/riker/iptraf/

che ogni nodo del cluster abbia una scheda ethernet da almeno 100 Mbit[7].

É stata effettuata un'altra analisi di scalabilità sul sistema messo a punto in laboratorio: il volume del traffico di rete generato complessivamente allo startup di un terminale. Il test è stato condotto esaminando il traffico ethernet sull'interfaccia del server master (quella della rete interna, sulla quale transitano tutti i pacchetti di gestione del sistema LTCSP), in particolare analizzando le seguenti porte, che rispecchiano i servizi che il terminale richiede per funzionare:

```
SERVIZIO   PORTA           DESCRIZIONE
om-mfs      723/tcp       # openMosix FileSystem port
om-disc    1334/udp       # openMosix autodiscovery protocol
om-mig     4660/tcp       # openMosix Migration Daemon port
om-info    5428/udp       # openMosix Info Daemon port
bootps       67/udp       # BOOTP server
bootpc       68/udp       # BOOTP client
nfsd       2049/udp/tcp   # NFS server daemon (nfsd) port
tftp         69/udp       # Trivial File Transfer Protocol
xdmcp       177/tcp/udp   # XDMCP protocol
```

Oltre quelle specificate, vi sono alcune ulteriori connessioni generate dal sistema su porte libere scelte dinamicamente. Dall'istante di accensione, il terminale completa tutte le fasi del caricamento (specificate in dettaglio nell'appendice B) nelle quali il numero dei pacchetti scambiati aumenta progressivamente nel tempo (poiché aumenta di fase in fase la complessità dei protocolli usati). Il picco del traffico è stato rilevato nello step in cui la workstation accorda e

7 La supposizione è da considerarsi verificata, considerando che la maggior parte dell'hardware fornitoci dall'associazione di trashware (RaccattaRAEE) aveva tali caratteristiche

abilita la connessione XDMCP (`startx`) con il gestore grafico del server: il valore è di circa **2 Mbit al secondo**, e decade pochi istanti dopo. Si tratta di un protocollo che distribuisce informazioni video, per cui era prevedibile aspettarsi un valore massimo piuttosto elevato, che tuttavia resta una frazione della banda offerta dall'hardware.

Considerando il vincolo di disponibilità di hardware datato, possiamo supporre che i punti nevralgici in cui smistiamo il traffico massimo (lo switch di rete e l'interfaccia ethernet del nodo master, che potrebbero rivelarsi colli di bottiglia) forniscano una **banda di almeno 100 Mbit/s**. Abbiamo considerato inoltre i test su LTCSP, trattati approfonditamente nel capitolo 5, e abbiamo intersecato i risultati ottenuti con le statistiche riportate nella documentazione di LTSP[8], per avere un confronto prestazionale del nostro cluster di laboratorio con i single server funzionanti, dislocati nel mondo: il nostro cluster fornisce una disponibilità di memoria e di elaborazione sufficiente a 20 o 25 terminali. Questo numero rientra perfettamente entro il limite di banda di rete fornita dall'apparato fisico, anche considerando i colli di bottiglia. Si deduce che LTCSP gode della proprietà di essere **ottimamente scalabile**.

4.3 Il Package di installazione

Rendere fruibile il lavoro sviluppato nel progetto è stato parte integrante del progetto stesso. Non appena si è rivelata possibile una integrazione dei software, sottoposti ad analisi, configurazione e test, è stata avanzata l'idea di racchiudere il codice maturo in un pacchetto di installazione, che fosse il più possibile multi-piattaforma,

8 Si veda LTSP, “Success stories”, sul sito ufficiale

in modo da rendere massima la disponibilità dello stesso. Tale pacchetto è sostanzialmente la collezione completa dei sotto-pacchetti che sono stati necessari a installare e gestire un cluster LTSP e openMosix. Inoltre il pacchetto comprende i tool, scritti ad-hoc, che cercano di affinare l'assetto generale del sistema.

Sul cluster del laboratorio, all'inizio del progetto, è stata installata una distribuzione GNU/Linux Debian Sarge, che offre un potentissimo sistema di gestione dei pacchetti (Advanced Package Tool). APT sa gestire autonomamente le dipendenze che correlano i pacchetti DEB (quelli scritti per APT), per cui rende facilissima l'installazione di nuovo software. Purtroppo, tale architettura non è presente sulle distribuzioni non basate su Debian Linux, per cui per raggiungere l'obiettivo di rendere portabile il nostro package, si è scelto di non usare APT, né altri formati dipendenti dalla piattaforma (i più comuni sono RPM e DEB). In rete, gli sviluppatori usano distribuire il software semplicemente archiviandolo e comprimendolo: viene distribuito il file in formato TAR.GZ, che contiene i file e gli script di installazione. Seguendo tale prassi è stato sviluppato il package `LTCSP-1.tar.gz`. Segue una descrizione piuttosto dettagliata delle sue caratteristiche.

Lo sviluppo del pacchetto è iniziato con la raccolta delle versioni più recenti dei sotto-pacchetti necessari. Essi sono sviluppati da team diversi, per cui sono anche rilasciati in formati eterogenei tra loro. Il primo lavoro è stato convertire tali formati in un formato base comune, ovvero una gerarchia di file e cartelle. É seguita l'analisi e la modifica dei file e degli script necessari al momento dell'installazione, che ha compreso la sostituzione dei comandi platform-dependent con altri comandi equivalenti, ma presenti su tutte le distribuzioni, e successivamente la replicazione sul file-system di destinazione dei file contenuti nei pacchetti. Infine si sono sviluppati gli script di installazione (`install.sh`) e disinstallazione finali. Come utimo passo di approfondimento, esaminiamo le operazioni dello script di

installazione (lo script `uninstall.sh` esegue le operazioni complementari):

- Reperisce dall'utente alcune informazioni indispensabili: dovendo predisporre un cluster, è necessaria una installazione per ogni macchina, ma di esse una sola è il master server. Questa informazione è richiesta all'utente.
- Verifica che siano installati gli strumenti necessari ai sotto-pacchetti, tra cui quelli necessari a costruire l'`initrd` del kernel.
- Installa gli strumenti di openMosix (kernel e pacchetti di gestione)
- Installa nel bootloader (nel caso sia installato GRUB) la voce corrispondente al kernel appena installato
- Installa gli strumenti di LTSP
- Verifica che siano installati i servizi (DHCP, TFTP, Portmapper, NFS e XDMCP) di cui LTSP ha bisogno. In caso contrario notificherà all'utente che tali servizi mancano.

É stato sviluppato infine uno script (`build-floppy.sh`) che aiuta l'utente a identificare la scheda di rete, montata sui terminali, e in seguito a scrivere su floppy l'immagine Etherboot, usata per eseguire lo startup sulla workstation. Bisogna notare che l'identificazione della NIC (Network Interface Card) non è realizzabile con l'uso di software (`lspci`) su una macchina su cui non c'è un sistema operativo (il terminale è diskless): si è per questo fatto ricorso all'elenco dell'hardware di rete che il progetto Etherboot supporta[9], filtrato attraverso i vendor e model ID, insieme alle informazioni reperite dall'utente. Nel caso in cui lo script non fosse sufficiente al build di una immagine boot corretta, si può ricorrere al software sviluppato da Marty Connor (`www.rom-o-matic.net`), che permette di configurare finemente l'immagine Etherboot.

9 Si veda http://rom-o-matic.net/5.4.2/etherboot-5.4.2/src/bin/NIC

Capitolo 5

5 I Test

In questo capitolo è trattata una sintesi dei test più rilevanti effettuati dal team. I test sono stati necessari a verificare innanzitutto il corretto funzionamento del cluster, successivamente a misurarne le prestazioni, dapprima in condizioni ideali, infine in condizioni di saturazione.

Sono occorse due fasi per testare LTCSP in modo completo: nella prima sono stati condotti i test, come descritto in seguito, in modo autonomo, mentre nella seconda si sono effettuati gli stessi test su di un single-server, circa di pari prestazioni. Il test bed su cui sono stati realizzati i test è composto dai seguenti nodi (si noti che il cluster è eterogeneo, come previsto in contesti trashware):

- Nodo 1: Intel Celeron 1000 MHz, 384 MB di RAM, scheda Ethernet SiS SiS900 (romolo)
- Nodo 2: AMD Athlon 700 MHz, 384 MB di RAM, scheda Ethernet 3Com 3c905 (tulloostilio)
- Nodo 3: Intel Celeron 500 MHz, 128 MB di RAM, scheda Ethernet Realtek RTL-8139C (numapompilio)
- Nodo 4: Intel Pentium II 400 MHz, 128 MB di RAM, scheda Ethernet Realtek RTL-8139C (ancomarzio)

Come single-server è stato scelto la seguente macchina:

- Intel Pentium 4 2600 MHz, 512 MB di RAM

Sulle macchine è stato installato il sistema operativo Debian Sarge, con i kernel 2.4.26 con l'ultima patch openMosix, e il 2.4.27 con patch openMosix e Migshm (si vedano le appendici). Come nodo master del cluster LTSP è stato scelto il nodo 1 (romolo), sul quale è stata installato il pacchetto LTSP (versione 4.2). Su tutti i nodi è stato installato Xorg, ma per risparmiare risorse di CPU solo sul nodo master esso è stato avviato durante i test. Tutti i PC hanno interfacce Ethernet da 100 Mbit/s, e sono interconnesse con uno switch di rete da 100Mbit/s.

Infine, il client usato in laboratorio è stato il seguente:

- Pentium MMX 200 Mhz, 32 MB di RAM, scheda Ethernet 3Com 3c595

Sono stati condotti i test eseguendo via terminale i seguenti software (in ordine di complessità crescenti): cicli dummy, LAME encoder, Povray, The Gimp, AbiWord. Il risultato discriminante è stato il tempo di esecuzione dei comandi, calcolato facendo precedere al comando che avvia il programma, la stringa `time`. Time è un comando che calcola il tempo di esecuzione dell'argomento che segue. Infine, per poter eseguire più programmi contemporaneamente, li si èeseguiti in background, cioè posponendo una “&” al comando. Tutte le istruzioni vengono eseguite lanciando contemporaneamente un certo numero di istruzioni identiche, con tale numero variabile da 1 a N (diverso a seconda del test).

5.1 Rendering

Il programma utilizzato in questo test è chiamato POVRAY: è in assoluto l’applicativo open source più utilizzato da chi si occupa di

”renderizzazione” di immagini. Il rendering grafico tipicamente richiede moltissimi calcoli, impegnando molto la CPU, perciò si è prestato molto bene a costituire un ambiente di benchmarking. Nel test si sono renderizzate alcune immagini di complessità crescente, fornite direttamente all’interno del pacchetto di povray. L’istruzione lanciata per realizzare il test è stata:

```
# povray <nomeImmagine.pov> +V
```

Essa restituisce a fine elaborazione il tempo trascorso. Si sono lanciate più istanze della stessa immagine (`abyss.pov`): come si evince dall'illustrazione 8, il single server ha prestazioni migliori del cluster fino al lancio parallelo di 5 istanze di povray. Al lancio di 6 istanze **le prestazioni del cluster raggiungono quelle del single server, per poi superarle** con il lancio di un numero di processi maggiore.

5.2 Editing di immagini

Una fase dei test è stata realizzata utilizzando The Gimp 2.2, uno strumento multipiattaforma per l'elaborazione di immagini fotografiche, distribuito secondo licenza GPL. Gimp (GNU Image Manipulation Program) è adatto ad una grande varietà di elaborazioni di immagini, come il foto ritocco, la composizione e la creazione di immagini, per cui è considerato uno dei migliori elaboratori di immagini per piattaforme GNU/Linux ed è per questo incluso in numerose distribuzioni. L'applicativo è lanciato da riga di comando tramite l'istruzione che segue un certo numero di volte:

```
# time gimp & xkill &
```

La misura del tempo impiegato dal cluster per eseguire i job è stata presa considerando il valore di output dell’istruzione time. Si noti che l’aggiunta di `xkill` è stata necessaria perché il valore di time viene

restituito solo quando la finestra dell'applicazione viene chiusa dall'utente e solo con xkill è possibile fare questo istantaneamente

Possiamo vedere nell'illustrazione 9, come la migrazione dei processi nel cluster oM fornisce un'**efficienza di gran lunga superiore** rispetto al singolo server, anche con un numero di processi molto limitato.

5.3 Encoding

Questa tipologia di test è stata realizzata misurando le prestazioni nello svolgere encoding di tracce audio da formato wave a formato mp3. Per realizzare ciò abbiamo utilizzato l'applicativo LAME 3.47, (LAME Ain't an Mp3 Encoder). Esso è utilizzato da un gran numero di applicazioni che elaborano audio e video e permette di realizzare l'encoding direttamente da riga di comando. L'istruzione lanciata è la seguente:

```
# lame -b 128 Track1.wav Track1.mp3
```

Il parametro `-b` ci permette di definire il bitrate (sono stati scelti 128 Kbps, in quanto hanno fornito i dati più significativi). Utilizzando lo stesso criterio è stata realizzata anche la scelta di una traccia audio di dimensioni medie, cioè circa 44 MB. Dal grafico nell'illustrazione 10 è possibile notare come, anche in questo caso, **le prestazioni del cluster OM superano quelle del singolo server** già al superamento di tre processi contemporanei, superamento che tende ad aumentare all'aumentare dei processi lanciati.

5.4 Word processing

L'ultimo test significativo è stato realizzato utilizzando l'applicativo

AbiWord, un progetto open source nato dalle ceneri di un progetto proprietario e portato avanti da una comunità di sviluppatori creatasi attorno ad esso. Questo word-processor è molto avanzato, ed è disponibile per le più diffuse piattaforme, secondo i termini definiti dal GPL. L'istruzione lanciata da riga di comando è:

```
# time abiword "Tutto è bene quel che finisce bene.txt" & xkill &
```

La scelta del testo è ricaduta su di un file di dimensioni medie, all'incirca 190 KB. L'istruzione è stata lanciata fino a quattro volte, ma non oltre, in quanto l'applicativo genera processi che richiedono molte risorse. Come visibile nell'illustrazione 11, ancora, in questo test, le prestazioni del singolo server sono migliori solo per l'esecuzione di uno o due processi: il vantaggio si ribalta quando vengono lanciati tre o più processi parallelamente.

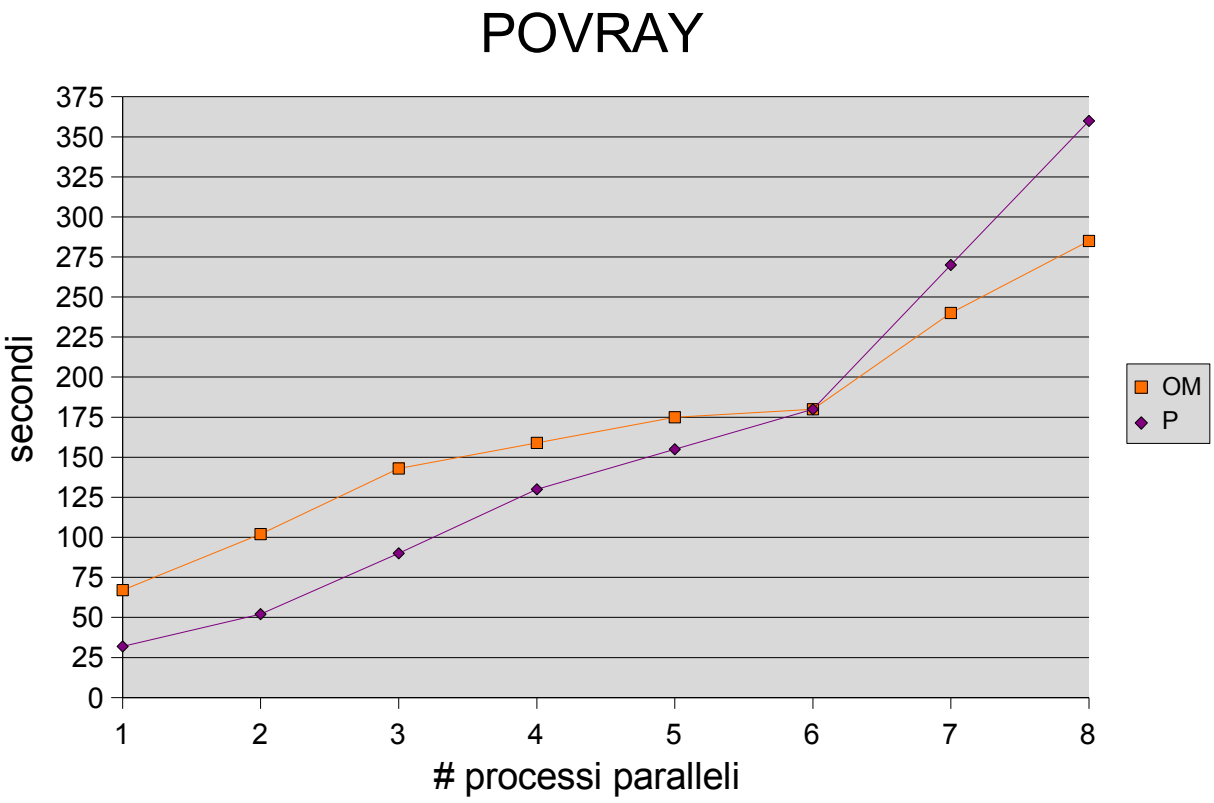

Illustrazione 8: Prestazioni di renderizzazione

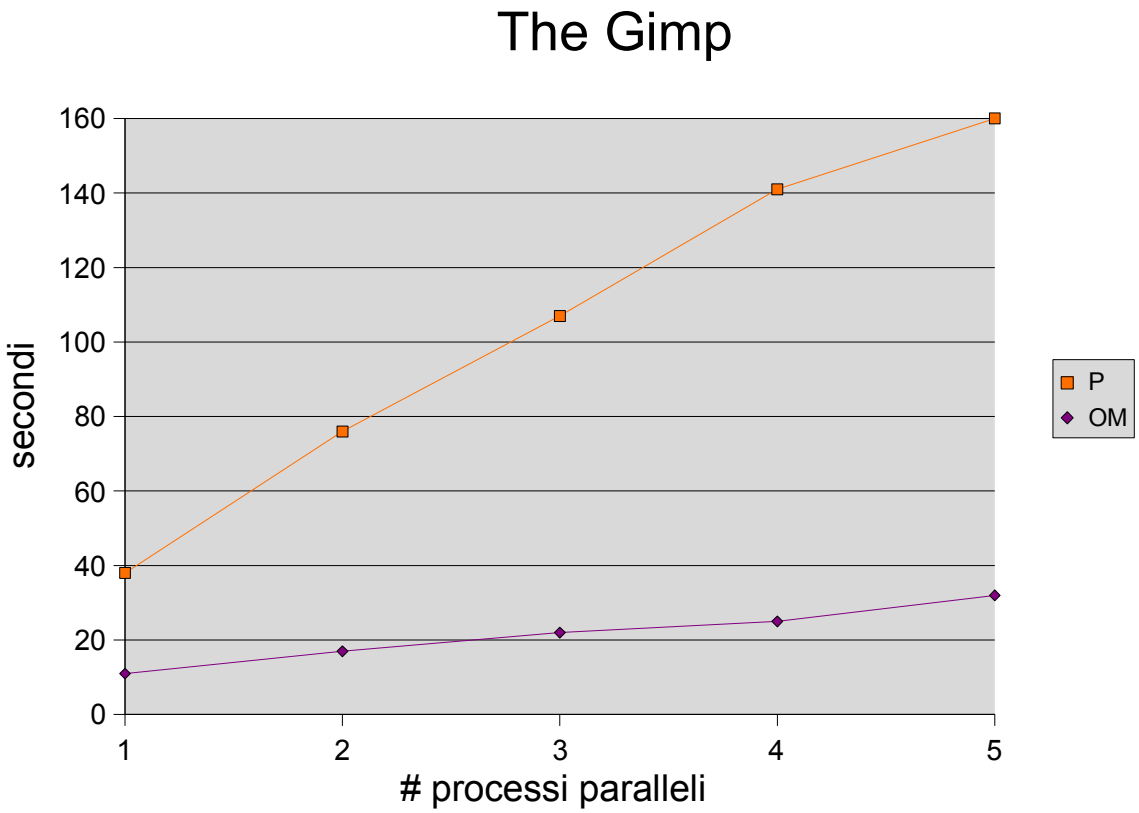

Illustrazione 9: Prestazioni di caricamento

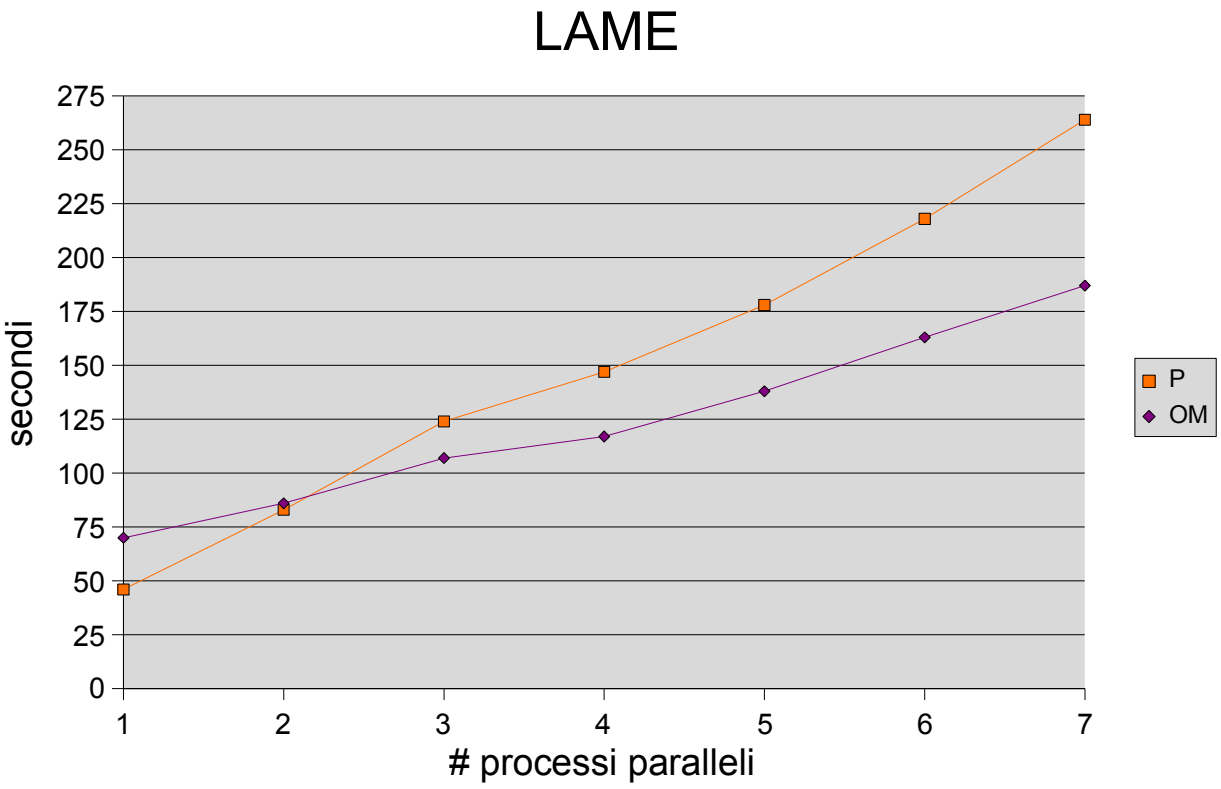

Illustrazione 10: Prestazioni di encoding

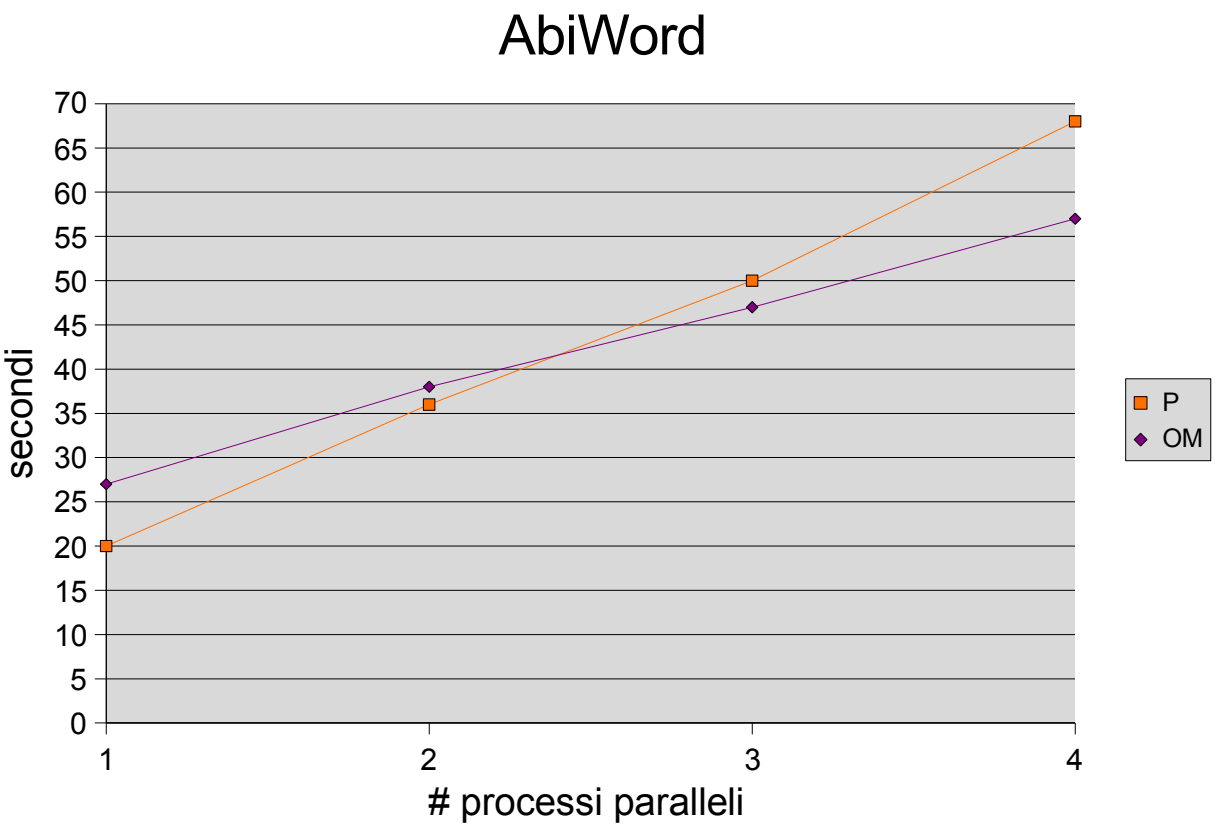

Illustrazione 11: Prestazioni di caricamento

Considerando che un server LTSP, in genere, deve sostenere il peso computazionale di numerosi client, con ottima probabilità dovrà eseguire un numero molto grande di processi parallelamente, per cui si troverà a lavorare nelle situazioni in cui il cluster OM presenta migliori prestazioni rispetto alla macchina performante.

6 Conclusioni

Il lavoro di tesi è stato portato a termine raggiungendo gli obiettivi genericamente prefissati. Il sistema ha rispettato i diversificati vincoli stabiliti, rispondendo in maniera brillante e confermando lo stato di maturità raggiunto dai progetti che hanno concorso a svilupparlo. Il nostro team, inoltre, ha acquisito in poco tempo abilità tecniche e teoriche notevoli, riguardanti la conoscenza del sistema operativo, di importanti aspetti del suo funzionamento, dei sistemi di clustering e dei sistemi di terminali, nozioni pratiche sull'uso e la gestione dell'hardware e degli strumenti fisici e software utili ad amministrarlo, inoltre conoscenze sull'impatto che le tecnologie comportano sulla società e sull'ambiente, e insieme alcune efficaci modalità per regolare alcune dinamiche che tale impatto comporta. In generale quindi, per poter sviluppare un sistema complesso, rispettando specifiche diversificate, il gruppo ha dovuto acquisire competenze sull'amministrazione di sistemi multiutente, sul software libero e sul management e riutilizzo dell'hardware. Tutti aspetti fondamentali nell'ingegneria.

Il progetto ha **dimostrato la fattibilità** di un sistema completo e complesso, composto esclusivamente da hardware datato, reperito grazie al trashware, e operante grazie a solo software libero. É stato dimostrato che **utilizzare hardware riciclato** per realizzare un terminale conviene, in termini di prestazioni (inalterate), impatto ambientale, possibilità economiche, potenzialità di utilizzo e sviluppo nei PVS. Esso sarebbe hardware altrimenti inviato in discarica o termovalorizzato precocemente. Abbiamo dimostrato per di più che è possibile formare un cluster di macchine che serva i terminali, anche qui restando in **contesti trashware e di free software**, con prestazioni che, all'aumentare dei nodi, superano quelle di un single

server. In particolare, GNU/Linux resta la soluzione ottima per rigenerare validità di computer obsoleti, grazie alla flessibilità con cui si possono aggregare o sottrarre funzionalità, adattandosi alle risorse disponibili. Il free software inoltre contrasta quel circolo vizioso voluto dalle grandi software house, in cui aumentano le richieste di calcolo dei software, le quali rendono indispensabili gli aggiornamenti dell'hardware: ciò ha soddisfatto in pieno la scelta del trashware come environment, ma soprattutto è in linea con le condotte di abbattimento del digital divide. La comunità che sviluppa software libero, inoltre, ci ha fornito in tempo reale diverse soluzioni a problemi sorti nel corso del lavoro, con modalità e strumenti che invece il software proprietario contrasta apertamente.

L'hardware dei terminali del sistema LTCSP **non ha praticamente bisogno di manutenzione**: esso non ha parti meccaniche in movimento (le più soggette a rottura) se non le ventole di raffreddamento. Ciò abbatte la rumorosità, fattore importante in ambienti di lavoro d'ufficio continuativo. Anche il software necessita di ridotto lavoro amministrativo: i programmi sono installati una sola volta e su di un singolo computer, inoltre è scarsa la possibilità di danni da parte degli utenti (il sistema di permessi ed il fatto che Linux sia nativamente un sistema operativo multiutente hanno reso ormai questo problema obsoleto). Conseguenze di questo aspetto sono la **semplificazione del sistema di logging e di backup**, che diventano centralizzati su di un solo PC senza operare nessuna modifica.

Linux Terminal Clustered Server Project, generato dalla fusione di LTSP ed openMosix, ha portato quindi con sé i pregi e i difetti di entrambe le architetture: dallo studio, LTCSP risulta **scalabile ed efficiente**, come dimostrato nei test e nelle analisi di carico computazionale e di rete. Resta il limite del non poter migrare i singoli thread e le applicazioni a memoria condivisa, sebbene tali limiti saranno abbattuti rispettivamente al rilascio di openMosix 2.6

(nei prossimi mesi) e della sua patch stabile migshm (per la memoria condivisa). Mentre la sezione **openMosix non necessita praticamente di configurazione**, in LTCSP resta purtroppo fondamentale qualche intervento manuale sull'assetto dei singoli servizi ai terminali: tali configurazioni riguardano strumenti che non appartengono esclusivamente a LTSP, per cui è difficile gestirli automaticamente senza prendere decisioni che spettano all'amministratore di sistema.

Un possibile progetto futuro potrebbe essere lo sviluppo di un tool grafico, che si interfacci con l'utente, che effettui le modifiche approfondite del sistema in modo comunicativo, che vada a sommarsi allo script di installazione sviluppato, e al limite a sostituire ltspadmin. Altre estensioni significative del progetto potrebbero riguardare aspetti più diretti all'uso comune del terminal server, come l'ampliamento all'uso dei device locali ai terminali. A tale scopo occorrerebbe includere la funzionalità `localdev`, che i progettisti di LTSP propongono in documentazione, nel progetto generico. Tale estensione permetterebbe di usare dispositivi quali lettori CD, lettore floppy disc, periferiche di memorizzazione USB, scanner e simili, direttamente sulle postazioni workstation.

Discorso a parte merita la possibilità di migliorare LTCSP dal punto di vista della continuità del servizio, ovvero renderlo un HA-Cluster (High Availability Cluster). Sebbene il cluster sia perfettamente scalabile, esso presenta l'inconveniente di avere un solo nodo master. La compromissione del nodo, o anche di un solo servizio (magari fondamentale allo startup dei terminali), che esso fornisce, causerebbe la mancanza di operatività dell'intero cluster. La comunità di sviluppo fornisce ad oggi alcuni strumenti, tra i più diffusi, il progetto Linux-HA (High-Availability Linux), il cui software di punta è Heartbeat. Si tratterebbe in sostanza di rendere ridondanti, su due o più nodi, i servizi di base e i dati sensibili, coordinando il lavoro di controllo con demoni installati appositamente e avendo a

disposizione dei tool che possano monitorare lo stato del sistema.

Concludendo, il sistema appare abbastanza maturo per poter essere installato negli ambienti per cui, più degli altri, esso è stato sviluppato, come le comunità del Sud del mondo, oppure i laboratori di scuole con poche risorse finanziarie. Ingegneria Senza Frontiere è già impegnata in progetti in Colombia, in Bosnia e nella periferia della città di Bologna. Probabilmente, il nostro team curerà la costruzione di un sistema simile su campo, in queste realtà: tale attività costituirebbe certamente un test più significativo di qualsiasi altro.

7 Appendici

7.1 Appendice A, Installazione e configurazione di openMosix

La procedura seguente illustra l'installazione e la configurazione di un cluster con il metodo classico di compilazione dei codici sorgenti. Vi sono procedure alternative, che implicano l'uso dei gestori dei pacchetti e variano da distribuzione a distribuzione, che rendono il processo di installazione più rapido.

Per l'installazione da sorgenti bisogna scaricare dalla rete i sorgenti del kernel 2.4.26, ottenuti ad esempio da `http://www.kernel.org`, successivamente copiarli nella directory `/usr/src`, poi scompattarli.

L'operazione genera una directory `linux-2.4.26/` all'interno della quale si trovano tutti i sorgenti del kernel. Successivamente si deve copiare in `/usr/src/` il file `openmosix-2.4.22.bz2`, scaricato da uno dei mirror di `http://sourceforge.net`. A questo punto si patchano i sorgenti del kernel con il comando:

```
bzcat /usr/src/openmosix-2.4.26.bz2 | patch -Np
```

Dopo averlo patchato, il kernel deve essere configurato e ricompilato, abilitando le voci specifiche di openMosix, oltre a quelle specifiche di ciascuna macchina tramite le istruzioni:

```
# make mrproper

# make clean

# make menuconfig
```

L'ultima istruzione in particolare avvia un menù che consente di configurare il kernel openMosix e di adattarlo alle esigenze hardware del cluster su cui dovrà lavorare. Sono state abilitate, quindi le voci specifiche di openMosix:

```
(*) openMosix process migration support

( ) Support cluster with a complex nethwork topology

(*) Stricter security on openMosix ports

(3)Level of process-identity disclosure (0-3)

(*) openMosix File-system

(*) Poll/Select exceptions on pipes

( ) Disable OOM Killer

( ) Load Limit
```

Una volta conclusa l'operazione di scelta dei moduli di cui si ha bisogno, segue la compilazione vera e propria del kernel con le istruzioni:

```
# make dep

# make clean

# make bzImage

# make modules

# make modules install
```

Successivamente bisogna installare il kernel patchato, e modificare il GRUB per poterlo avviare:

```
# cp arch/i386/boot/bzImage /boot/vmlinuzOM

# cp System.map /boot/System.map-2.4.22-OM

# vi /etc/lilo.conf
```

In GRUB è necessario aggiungere le righe che consentono di avviare il nuovo kernel:

```
image=/boot/vmlinuzOM

label=Linux-OM

read-only
```

Il GRUB è stato successivamente testato e reinstallato.

Ora, se il sistema è avviato con il kernel openMosix, è già in grado di migrare i processi.

Per poter gestire e monitorare openMosix sul nuovo kernel sono necessari alcuni tool, che devono essere installati con i seguenti comandi:

```
# tar xvfz openmosix-tools-0.3.6.tar.bz2

# cd openmosix-tools-0.3.6/

# ./configure

# make

# make install
```

Diversamente, anche essi possono essere installati più rapidamente tramite pacchetto.

Questi tools generano per prima cosa lo script con il quale e possibile lanciare openMosix su un generico nodo del cluster, inoltre forniscono alcuni strumenti per il monitoraggio, visti più avanti in questa appendice.

L'ultima operazione da eseguire, non obbligatoriamente, consiste nell'editare il file `openmosix.map` in cui sono contenuti gli indirizzi IP dei nodi del cluster.

Il file del cluster è fatto come segue:

```
# openmosix-node ID IP-Address (or hostname) Range-size

1 192.168.100.254 1
```

```
2 192.168.100.190 1

3 192.168.100.191 1

4 192.168.100.192 1
```

L'operazione non è obbligatoria perché è stato sviluppato il demone `omdiscd`, che si occupa di rimappare dinamicamente il numero dei nodi del cluster (attraverso l'invio periodico di un pacchetto multicast).

Con questa operazione si conclude la parte che illustra l'installazione e la configurazione del cluster. É interessante notare come, al limite, per creare un cluster funzionante basterebbe installare un kernel patchato. La semplicità con cui si costruisce un sistema che migra i processi è certo un punto di forza di openMosix.

7.1.1 Monitoraggio del cluster

Per monitorare l'ambiente openMosix sono disponibili diversi strumenti. Alcuni sono forniti direttamente con esso (come gli openmosix-tools), altri sono ideati e sviluppati dalla nutritissima comunità di developer che si è raccolta intorno al progetto.

Il pacchetto openmosix-tools fornisce cinque strumenti di gestione, tutti eseguibili esclusivamente da riga di comando: mosctl, mosmon, mosrun, mps, setpe.

- `mosmon`: monitorizza lo stato del cluster, fornendo un grafico aggiornato in tempo reale sulle condizioni di carico dei diversi nodi;

- `mosctl`: tool di amministrazione del cluster, consente di gestire la migrazione dei processi verso determinati nodi del cluster e l'accesso al filesystem di openmosix;
- `mosrun`: esegue un comando su un particolare nodo del cluster;
- `mps`: costituisce un potenziamento di ps, offrendo la possibilità di vedere tutti i processi lanciati nel cluster e i nodi su cui si trovano.
- `setpe`: consente la gestione del pool di indirizzi del cluster (mappatura dei nodi);

Oltre ai suddetti strumenti, merita essere menzionato `mtop`. Questa applicazione rappresenta la versione per il cluster di "top", applicazione che fornisce in tempo reale istantanee dell'attività del processore. In particolare quest'ultimo mostra i processi in esecuzione che fanno maggiore utilizzo della CPU ed il loro stato (running, sleeping, stopped), il PID (process ID) di ciascuno, da quanto tempo sono in esecuzione, l'utente che li ha lanciati, il loro grado di priorità, ed alcune altri parametri anche configurabili; mtop si differenzia da top per il fatto che mostra queste informazioni non per il singolo nodo ma per tutto il cluster, perciò viene fatto vedere anche il numero del nodo in cui un processo sta girando, ed il numero di migrazioni tra nodi del cluster che il task ha effettuato.

Come detto all'inizio del paragrafo, oltre a questi tools utente, la comunità openMosix ha sviluppato la suite openmosixview, composta da cinque applicazioni per la gestione, il monitoraggio ed il controllo del cluster, con veste grafica facilmente interpretabile. Gli applicativi contenuti nel pacchetto sono:

- `openmosixview`: l'applicazione principale di amministrazione

e monitoraggio;

- `openmosixprocs`: applicazione per la gestione dei processi;
- `openmosixcollector`: raccoglie le informazioni sullo stato dei nodi e sull'intero cluster;
- `openmosixanalyser`: analizza i dati ottenuti da openmosixcollector;
- `openmosixhistory`: registra la storia dei processi del cluster;

Tutte le applicazioni sono accessibili dalla finestra di openmosixview, le cui caratteristiche sono mostrate nell'illustrazione 4 (si veda il capitolo 3.1.1). In essa sono presenti più righe, ciascuna delle quali controlla un nodo. Percorrendo idealmente la finestra da sinistra a destra troviamo prima un rettangolo il cui colore ci dice se quel nodo del cluster è funzionante (verde) oppure no (rosso), con all'interno l'ID del nodo. Abbiamo poi un bottone con l'indirizzo IP del nodo; da questo si accede ad una finestra di dialogo con la quale si può per esempio veicolare la migrazione verso specifici nodi. Abbiamo successivamente il cursore che consente di modificare l'efficienza di ciascun nodo, altrimenti calcolata dall'algoritmo di openMosix. Successivamente abbiamo due barre di avanzamento che indicano per ciascun nodo le percentuali di carico complessivo e di memoria utilizzata. Infine le ultime due informazioni si riferiscono alla quantità di RAM di ogni nodo ed al numero di CPU presenti.

Altro tool molto utile è openmosixmigmon. Si tratta di un monitor per le migrazioni all'interno del cluster openMosix. Esso mostra tutti i nodi, intorno ad un nodo centrale che è quello su cui è in esecuzione openmosixmigmon. I processi che competono al nodo centrale sono disposti in circolo intorno ad esso. Quando un processo migra verso un altro nodo, viene indicato con il colore verde e di dispone in cerchio intorno al nodo verso cui si è spostato. Questo strumento consente non solo di osservare le migrazioni e di conoscere il PID del processo, ma soprattutto di spostare a piacimento i processi in

uno qualunque degli altri nodi semplicemente con l'utilizzo del mouse (si veda l'illustrazione 12).

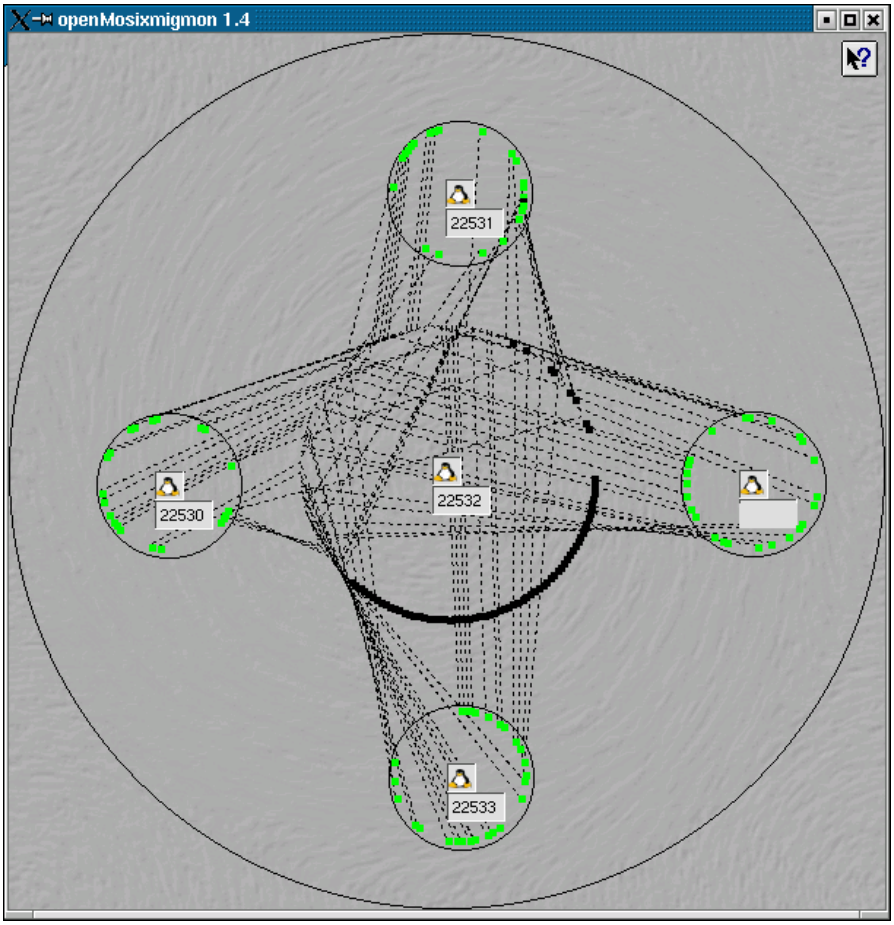

Illustrazione 12: openMosixMigmon

Come detto nello stesso capitolo, openMosix fornisce delle ottime prestazioni in tutte quelle applicazioni composte da processi CPU-bounded, ma non consente la migrazione dei processi che competono alle applicazioni che fanno uso di memoria condivisa. Per consentire a queste applicazioni di migrare è necessario aggiungere una patch ulteriore, ancora sperimentale.

7.1.2 Aggiunta di Migshm ad openMosix

Come detto precedentemente, migshm è una patch del kernel Linux, con le funzionalità openMosix attive. Il kernel di partenza, nella fase di patching, deve essere stato quindi già patchato con openMosix.

Il kernel deve essere patchato con le seguenti istruzioni:

```
# openmosix-2.4.26.bz2 | patch – Np1

# patch-Migshm-2.4.26-1.diff | patch – Np1
```

Successivamente è necessario abilitare alcuni parametri aggiuntivi nel kernel, per rendere operative le nuove proprietà di migrazione per applicazioni a memoria condivisa:

```
#Shared memory migration support(Exper.)(CONFIG-SHM)(N/y/?) y

#flush() support for consistency(CONFIG-SHM-FLUSH)(N/y/?)(NEW) y

#Kernel Debug messages (CONFIG-SHM-DEBUG)(N/y/?)(NEW) y
```

Dopo avere ricompilato il kernel ed averlo avviato correttamente su un nodo del cluster, l'installazione sulle altre macchine può essere più agevole e veloce se si decide di non procedere alla ricompilazione anche sugli altri nodi. Semplicemente è possibile copiare l'immagine del kernel su tutti gli altri nodi, e con essa copiare anche la cartella che contiene i moduli del kernel che vengono caricati all'avvio (`/lib/modules/2.4.26`), e quella che contiene i sorgenti e i tool di monitoraggio (`/usr/src/linux-2.4.26`). A questo punto è sufficiente modificare il GRUB per rendere il kernel avviabile.

Si hanno ora due ambienti diversi, nonostante il software sia lo stesso.

7.2 Appendice B, Installazione di LTSP

7.2.1 Il sistema LTSP in dettaglio

Il sistema LTSP si compone di un pacchetto di programmi iniziale, denominato generalmente *ltsp-utils.* con il quale si predispone la struttura di directory necessaria, si scaricano e si installano i pacchetti specifici del sistema LTSP, infine si configurano i servizi accessori, necessari all'elaboratore che ospita LTSP. Tali servizi accessori devono essere preventivamente stati installati, e in genere consistono nei seguenti package: `dhcp3-server`, `dhcp-common` (che di solito già installato), `dhcp3-dev`, `tftpd-hpa`, `nfs-kernel-server`, `xinetd`, `libwww-perl` (necessario per installare ltsp-utils).

Il sistema LTSP viene installato a partire dalla directory `/opt/ltsp/`, mentre i file del kernel, necessari per l'avvio dei terminali remoti, nella directory `/tftpboot/lts/`. Inoltre, il file `/etc/ltsp.conf` tiene traccia di alcune impostazioni generali del sistema LTSP.

Segue un riassunto dei file e delle cartelle coinvolte in LTSP:

File o directory	Descrizione
`/etc/ltsp.conf`	File di configurazione di LTSP, gestito attraverso i programmi di servizio di LTSP stesso
`/tftpboot/lts/`	File necessari per l'avvio di un terminale remoto: kernel, disco RAM iniziale, altri file di avvio

`/opt/ltsp/`	Sistema LTSP
`/opt/ltsp/pkg_cache/`	Directory di appoggio per il download dei pacchetti di LTSP
`/opt/ltsp/i386/`	Sottoalbero del filesystem del server, avviato presso i terminali remoti, in questo caso per achitettura i386
`/opt/ltsp/i386/etc/lts.conf`	File di configurazione avviato presso i terminali remoti
`/opt/ltsp/i386/etc/screen.d/`	Directory contenente gli script di controllo delle funzioni da attribuire alle console virtuali dei terminali remoti

Perché LTSP possa funzionare, è necessario che alcuni servizi siano disponibili presso lo stesso elaboratore in cui esso si installa. I programmi di servizio di LTSP sono in grado di controllare la configurazione della maggior parte dei servizi necessari; tuttavia, è bene rendersi conto di cosa serve a LTSP:

- ***DHCP****:* *`dhcpd.conf`*. Il servizio DHCP deve essere configurato in modo tale da poter assegnare, ai terminali e agli altri host, un gruppo di indirizzi IPv4, validi nella rete in cui si intende operare; inoltre ai terminali è necessario fornire l'indicazione del file da usare per l'avvio. In totale, la workstation prenderà le seguenti informazioni dal server DHCP: indirizzo IP, hostname, indirizzo IP del server, Default gateway, Directory e nome del kernel da scaricare. L'esempio seguente rappresenta la configurazione utilizzata per i test di laboratorio:

```
option option-128 code 128 = string;
option option-129 code 129 = text;
ddns-update-style ad-hoc;
```

```
default-lease-time            21600;
max-lease-time                21600;
option subnet-mask            255.255.255.0;
option broadcast-address      192.168.100.255;
option routers                192.168.100.254;
option domain-name-servers    213.140.2.12;
option domain-name            "indivia.net";
option root-path              "192.168.100.254:/opt/ltsp-4.2/i386";
get-lease-hostnames           true;
use-host-decl-names           on;

# extra IP per host ospiti: range 100-189
subnet 192.168.100.0 netmask 255.255.255.0
{
     range 192.168.100.100 192.168.100.189;
}

# IP statici: cluster di server: range 190-192
host tulloostilio {
        hardware ethernet     00:60:97:0D:0D:56;
        fixed-address         192.168.100.190;
}

host numapompilio {
        hardware ethernet     00:50:DA:BD:53:07;
        fixed-address         192.168.100.191;
}

host ancomarzio {
        hardware ethernet     00:10:A7:1D:09:1D;
        fixed-address         192.168.100.192;
}

# IP statici: workstation: range 200-202
group
{
```

```
    option log-servers        192.168.100.254;
  # The following is _NOT_ a MAC address!
    option option-128         e4:45:74:68:00:00;
    filename                  "/tftpboot/lts/vmlinuz-2.4.22-ltsp-2";

    host peter {
        hardware ethernet     00:50:04:BB:21:59;
        fixed-address         192.168.100.200;
    }

    host stewie {
        hardware ethernet     00:A0:24:E1:BE:58;
        fixed-address         192.168.100.201;
    }

    host brian {
       hardware ethernet     00:D0:09:30:6A:1C;
       fixed-address         192.168.100.202;
    }
}
```

Tabella 2: Il file `/etc/dhcp3/dhcpd.conf`

- ***TFTP****: `/etc/inetd.conf`*. Il servizio TFTP serve per trasferire i file necessari alla prima fase di avvio dei terminali remoti. Generalmente il servizio viene configurato in modo tale da offrire l'accesso esclusivamente alla gerarchia `/tftpboot/` (questo specificando anche l'opzione `-s` all'avvio del programma). Generalmente, il programma `tftpd` viene controllato dal supervisore dei servizi di rete; nel caso particolare di inetd, l'installazione di LTSP provvede a sistemare il file `/etc/inetd.conf` aggiungendo la riga:

```
tftp  dgram  udp  wait  root  /usr/sbin/tcpd        in.tftpd -s /tftpboot
```

  Nell'esempio di configurazione mostrato per il servizio DHCP, appare la direttiva `filename "/lts/vmlinuz-2.4.22-ltsp-2";` che serve a specificare il file del kernel da caricare

inizialmente. Nel file che configura DHCP appare anche la possibilità, per ogni singolo terminale, di caricare un file di avvio diverso da quello di default: in questo modo è possibile scaricare un kernel appropriato all'architettura specifica della macchina. Il servizio di tftd può essere abilitato, alternativamente, con un demone, senza ricorrere a inetd, che sarà configurato dallo stesso file.

- ***NFS**: `/etc/exports`*. Il sistema LTSP viene avviato presso i terminali remoti che innestano un file system di rete, offerto appunto tramite il protocollo NFS. Il servizio NFS viene configurato notoriamente tramite il file `/etc/exports` (nell'esempio), per stabilire cosa rendere accessibile all'esterno. La condivisione di `/opt/ltsp/i386/home/` è opzionale. Essa è utilizzata nel caso si vogliano attivare applicazioni locali ai terminali (*localapps*), che necessitano quindi di file di configurazione specifici.

```
/opt/ltsp
192.168.100.0/255.255.255.0(ro,no_root_squash,sync)

/var/opt/ltsp/swapfiles   192.168.100.0/255.255.255.0(rw,no_root_squash,async)
```

- ***Risoluzione dei nomi**: `/etc/hosts.`* I computer comunicano piuttosto bene usando gli indirizzi IP, ma questi si rivelano piuttosto scomodi per gli uomini, che preferiscono dare nomi ad essi; tale corrispondenza è mantenuta tramite il DNS o il file `/etc/hosts`: in un ambiente LTSP, senza la correlazione IP-nome, NFS darebbe errori sui permessi (quando la workstation tenta di montare il filesystem di root). Oltre ciò, se la workstation non è elencata nel file /etc/hosts , si potrebbero avere problemi anche con i display manager GDM e KDM. É quindi necessario aggiungere nel file `/etc/hosts` le coppie “ip nome” di tutti i terminali usati.
- ***XDMCP**.* Lo scopo principale di LTSP è quello di avviare presso

i terminali un sistema grafico che si colleghi al sistema corrispondente sul servitore (nel nostro caso il server X, tramite protocollo XDMCP). Il servizio di accesso remoto grafico XDMCP viene offerto da diversi display manager, per esempio da XDM. Bisogna osservare che programmi come XDM, GDM o KDM, che fanno eseguire l'accesso al sistema direttamente in sessione grafica, sono configurati in modo predefinito per ignorare completamente le richieste XDMCP provenienti dalla rete; pertanto occorre provvedere ad abilitare gli accessi. I tool di LTSP sono in grado di modificare tale configurazione, comunque è utile vedere un esempio (in questo caso usiamo XDM): Il file di configurazione `/etc/X11/xdm/xdm-config` contiene l'elenco degli altri file utilizzati e di altre opzioni; in particolare contiene una direttiva che impedisce l'accesso, che in questo caso va tolta o isolata (nell'esempio viene commentata):

```
 ...

 DisplayManager.servers:
/usr/X11R6/lib/X11/xdm/Xservers

 DisplayManager.accessFile:
/usr/X11R6/lib/X11/xdm/Xaccess

 ...

 ! SECURITY: do not listen for XDMCP or Chooser requests

 ! Comment out this line if you want to manage X terminals with
xdm

 # DisplayManager.requestPort:             0
```

Il file /etc/X11/xdm/Xaccess serve a limitare l'accessibilità del servizio XDMCP. La direttiva dell'esempio seguente abilita l'accesso a qualunque indirizzo:

```
 ...

 #
```

```
# Any host can get a login window:
#
*
...
```

7.2.2 Maggiori dettagli su XDMCP

Data l'importanza fondamentale che XDMCP svolge nel sistema costituito, è opportuno capire quanto esso influisce nell'architettura e in quale modo. XDMCP (X Display Manager Control Protocol) è un protocollo del sistema X Window System (chiamato anche X o X11), che fornisce software e protocolli per la gestione della grafica sui sistemi Unix e Unix-like. Il sistema X comprende l'X Display Manager (XDM), ovvero l'implementazione storica del gestore che fornisce la schermata di login e la sessione grafica che ne segue. Le altre implementazioni più diffuse sono GDM (Gnome Display Manager) e KDM (KDE Display Manager). Nell'architettura del sistema X il paradigma è il seguente: vi è un X server, che esegue sulla macchina che monta lo schermo e le periferiche di input, e diversi X client (tra cui anche il display manager), cioè i software che richiedono al server di mostrare le finestre a video. X è un sistema flessibile proprio perché, suddividendo i moduli software, permette di lavorare in distribuito in modo trasparente. Questa caratteristica costituisce il cuore di LTSP.

Il protocollo XDMCP costituisce il meccanismo con cui i client X comunicano con il server X, quando esso si trova su un altro host. Esso usa la porta UDP 177; su di questa si instaura la comunicazione con i pacchetti `Query` e `Willing`, rispettivamente di richiesta di connessione al display manager e di risposta all'X server. Successivamente l'X server manda un `Request packet` al display

manager, il quale risponde con un `Accept packet`. L'Accept packet contiene la risposta all'X server, con la quale si conferma l'autenticazione, grazie all'inclusione di una chiave segreta. Segue un `Manage packet` di conferma al display manager. Instaurata la sessione il server manda ad intervalli regolari un `KeepAlive`, ovvero un pacchetto che indica che il display manager è attivo. Se la risposta `Alive` non torna indietro entro un certo tempo, il display manager si considera inattivo.

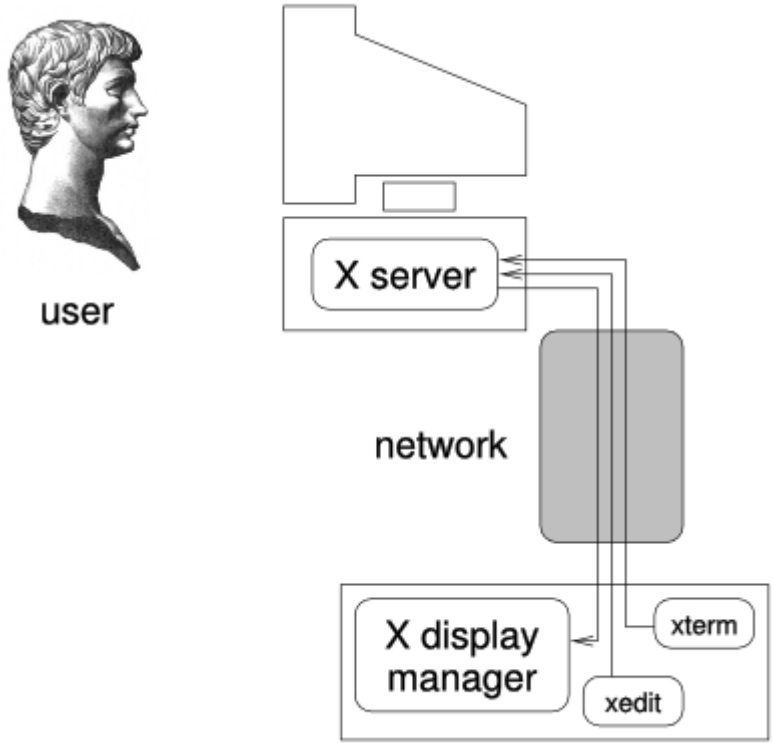

Illustrazione 13: Architettura semplificata di X

7.2.3 Installazione di LTSP

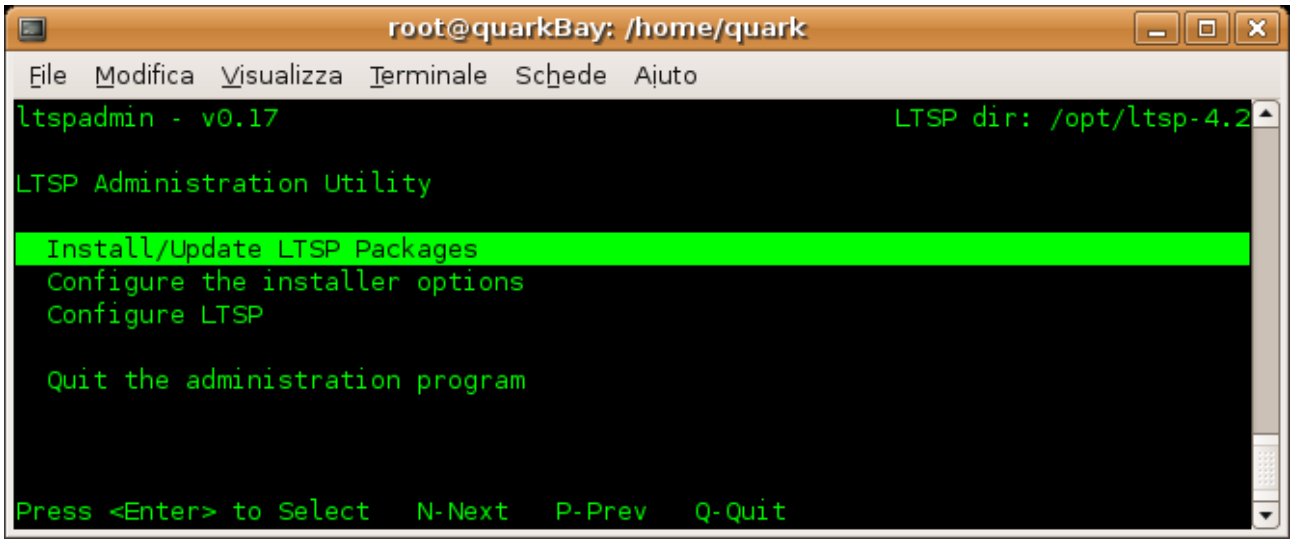

Illustrazione 14: ltspadmin

L'installazione di LTSP nell'elaboratore ospite avviene attraverso un pacchetto, denominato `ltsp-utils`, che deve essere a sua volta installato nel sistema. Fatto ciò, si procede all'utilizzo di Ltspadmin, con il comando `ltspadmin` (Illustrazione 14).

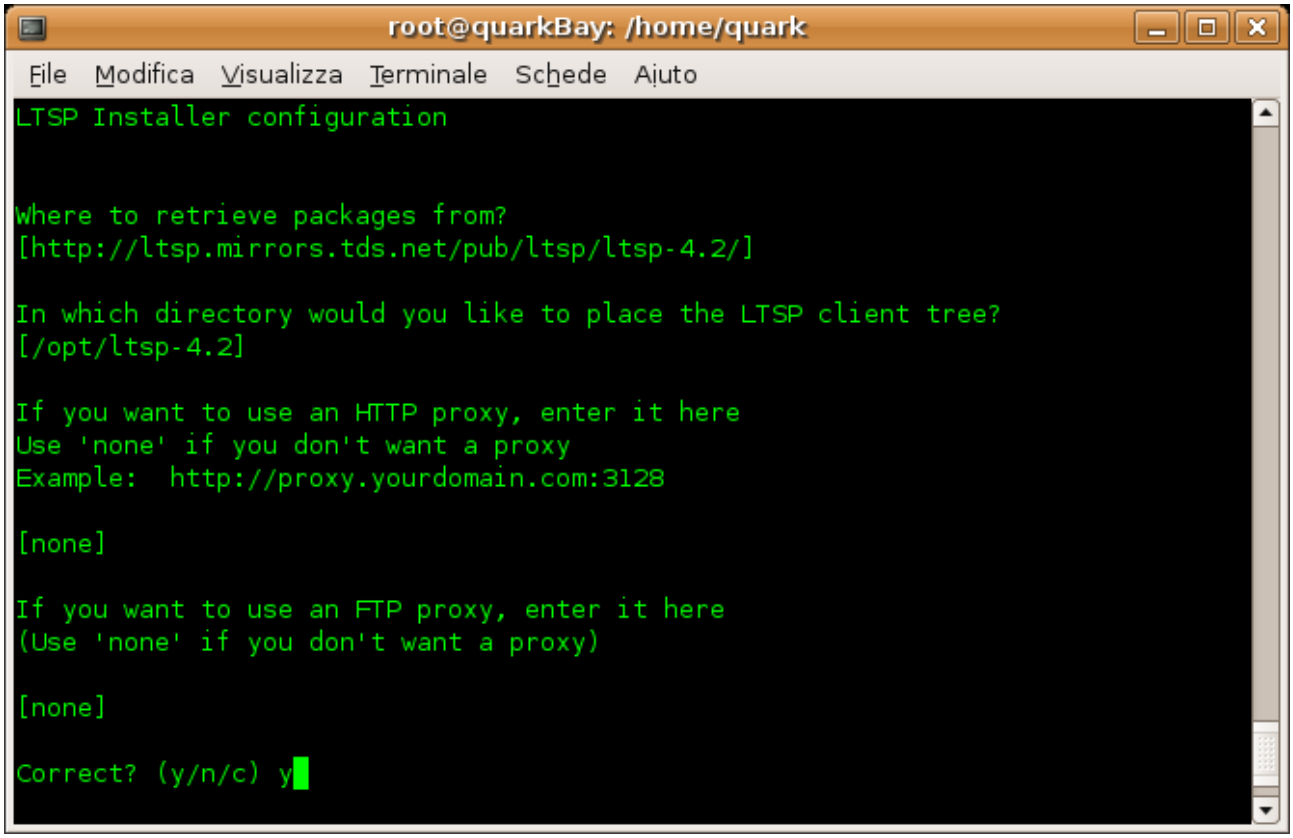

Illustrazione 15: ltspadmin, configurazione

Il primo passo è configurare l'installatore: si procede con `Configure LTSP`. Esso richiede alcune informazioni, che vanno a fissarsi nel file `/etc/ltsp.conf`.

```
...

LTSP_DIR=/opt/ltsp-4.2

PKG_SOURCE=http://www.ltsp.org/ltsp-4.2/

HTTP_PROXY=none

FTP_PROXY=none
```

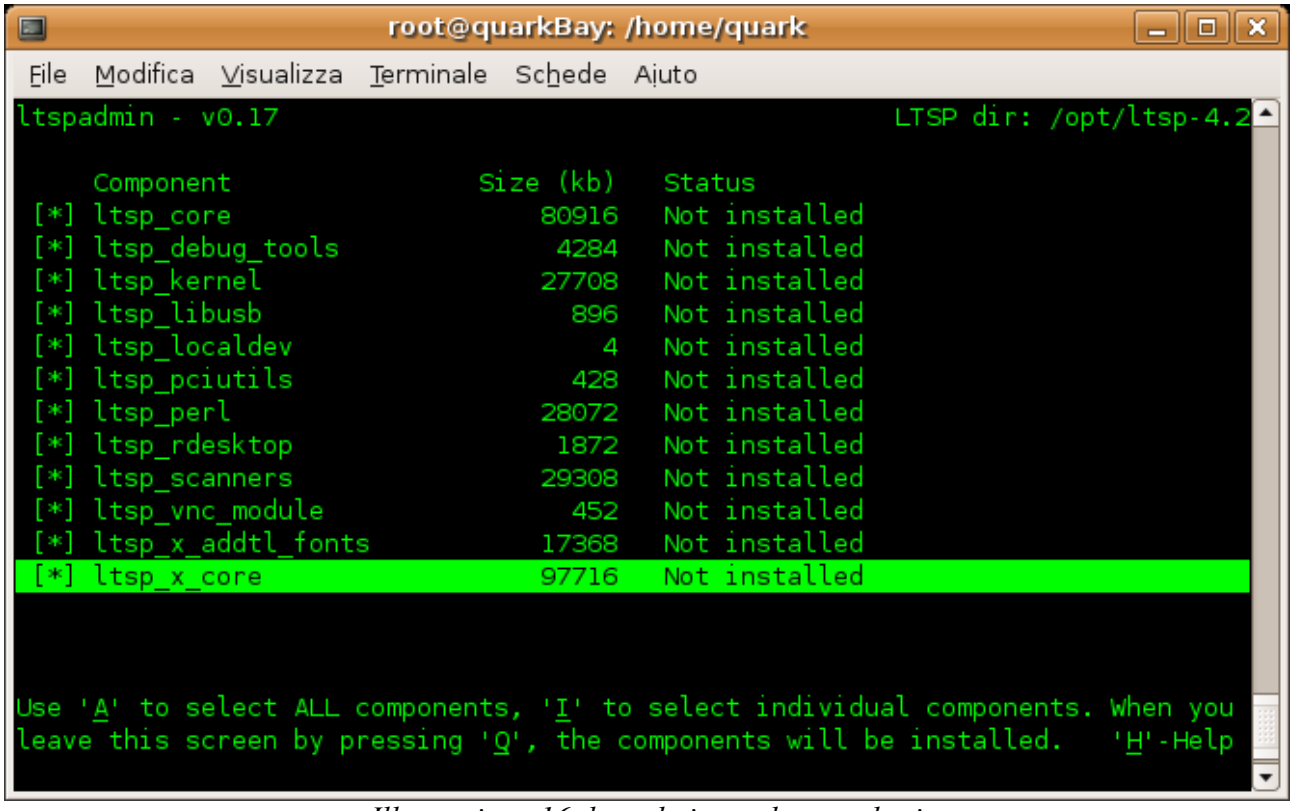

Illustrazione 16: ltspadmin, scelta pacchetti

I pacchetti raccolti (illustrazione 16) vengono depositati nella directory `/opt/ltsp/pkg_cache/`; una volta completata l'installazione, questi file possono essere rimossi.

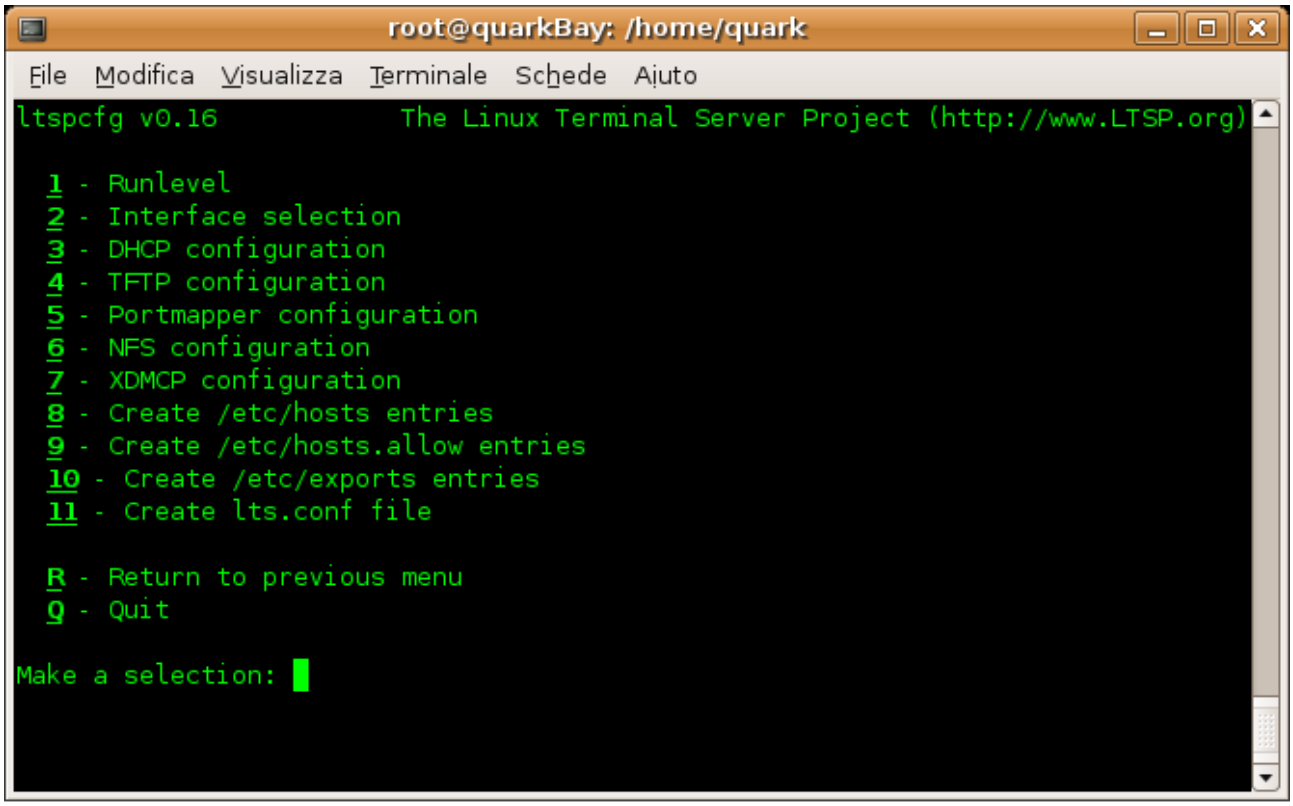

Illustrazione 17: ltsp, configurazione servizi

Fatto ciò, il programma permette di configurare in automatico e passo passo tutti i servizi (illustrazione 17) e di mostrare lo stato in tempo reale (illustrazione 18).

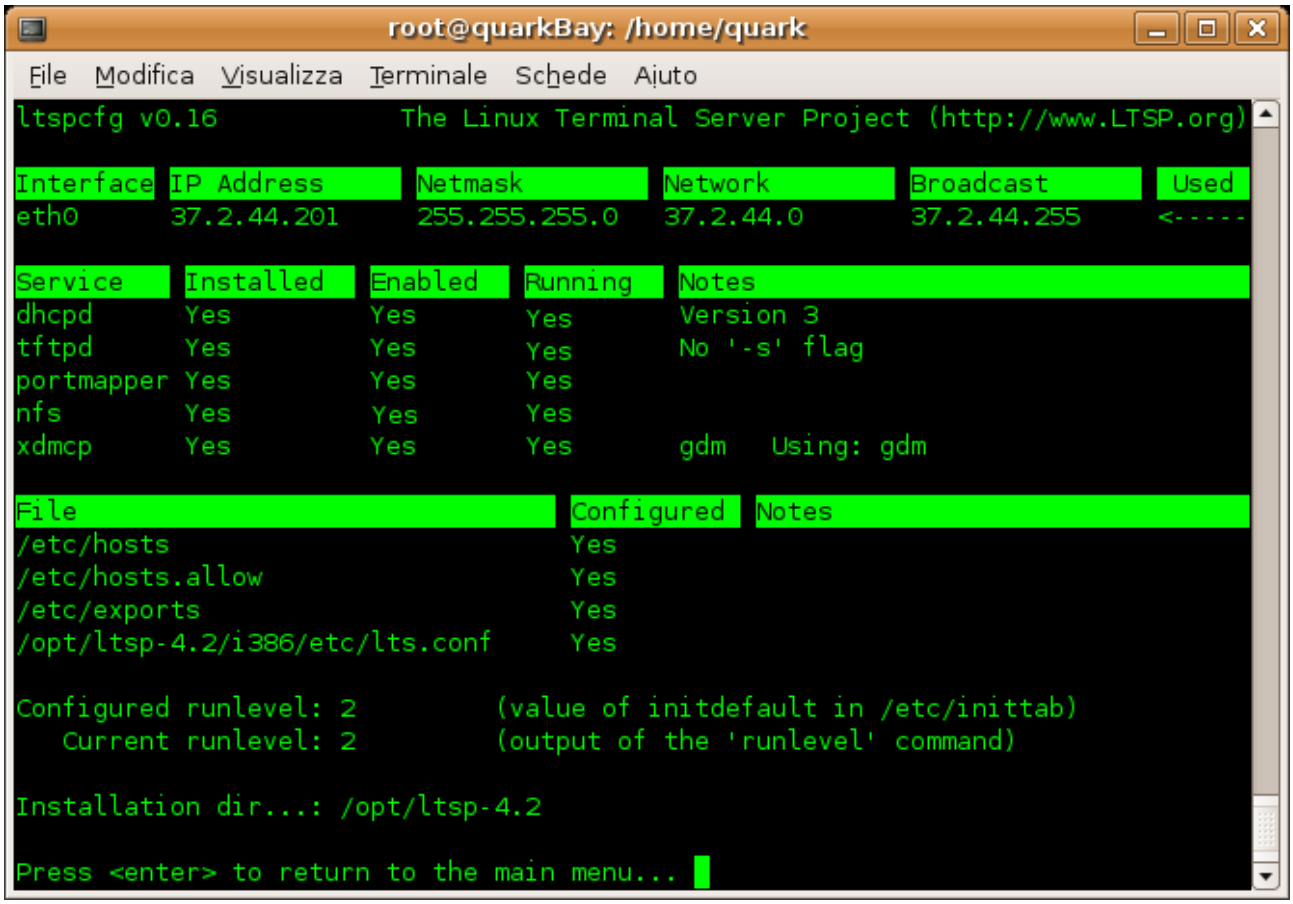

Illustrazione 18: ltsp, stato del sistema

7.2.4 Configurazione dei terminali

Una volta che il server è funzionante, si configurano le workstation. Dato che i terminali vanno semplicemente istruiti per scaricare il kernel che utilizzeranno dal server, l'unica operazione da compiere è permettere loro di caricare all'avvio del codice. Ci sono molti modi per far entrare il kernel in memoria, come ad esempio Etherboot (attraverso la eeprom a bordo della scheda), Netboot, PXE o un floppy disk. In genere, il floppy disc contenente il codice Etherboot è il mezzo più comodo.

7.2.5 Startup del terminale

Etherboot è un pacchetto software open source per creare immagini ROM che possono scaricare codice tramite un collegamento di rete, per essere eseguito su computer x86 (molte schede di rete hanno uno zoccolo dove può essere installata una eeprom o una ROM dove si è caricato il codice Etherboot). Si può scaricare il pacchetto Etherboot e configurarlo per il tipo di rom di cui si ha bisogno. Poi, si possono compilare i sorgenti per produrre un'immagine rom che può essere scritta in una eeprom.

Un'alternativa molto più semplice è stata sviluppata da Marty Connor: il sito `www.rom-o-matic.net` compila in modo automatico i sorgenti di Etherboot e fornisce l'immagine della rom, già pronta per essere scritta sul floppy disk.

Supponendo che il server e la workstation siano configurati correttamente, la procedura è inserire il dischetto nel drive della workstation (predisposta per il boot da floppy disc) ed accenderla.

Il codice Etherboot sarà letto dal floppy e trasferito in memoria, la scheda di rete verrà riconosciuta ed inizializzata, la richiesta dhcp verrà inviata sulla rete ed una risposta tornerà dal server, quindi il kernel verra scaricato alla workstation. Una volta che il kernel avrà inizializzato l'hardware della workstation, X window partirà e la finestra del login dovrebbe apparire sulla workstation.

A questo punto è possibile accedere tramite il login; una cosa importante da tenere a mente è che si sta accedendo al server. Tutti i comandi vengono quindi eseguiti sul server e mostrano il loro output sul monitor della workstation (è questa la potenza di X window). Ora è possibile eseguire qualunque programma installato sul server.

7.3 Appendice C, Indice di sviluppo ORBICOM

	Literacy (%)	Enrolment (%)			Skills
		Primary	Secondary	Tertiary	
	indicator	*indicators*			*index*
Sweden	99.0	110.0	148.8	70.0	155.0
Finland	99.0	101.7	125.9	85.3	154.0
Australia	99.0	102.4	153.8	64.6	153.3
United Kingdom	99.0	101.0	157.9	59.0	152.0
Belgium	99.0	105.2	154.4	58.3	151.3
Norway	99.0	101.5	114.6	70.0	145.0
New Zealand	99.0	99.0	113.2	71.7	145.0
Denmark	99.0	102.0	128.2	58.9	144.3
Korea (Rep.)	98.1	100.1	94.2	82.0	143.5
Netherlands	99.0	107.7	124.4	55.0	142.4
Russia	99.6	113.8	92.0	68.4	140.3
Estonia	99.8	103.0	110.1	59.3	140.2
United States	99.0	100.3	94.1	70.7	139.5
Spain	97.9	107.1	114.2	56.8	139.5
Slovenia	99.7	100.2	106.4	60.5	139.2
Canada	99.0	99.6	106.2	59.1	138.0
Lithuania	99.6	104.4	98.5	59.1	136.9
France	99.0	105.0	107.7	53.6	136.8
Latvia	99.8	98.8	92.6	64.3	136.7
Argentina	97.1	119.6	99.6	56.6	136.5
Ireland	99.0	118.9	109.1	47.3	136.5
Austria	99.0	103.3	98.6	57.2	135.5
Poland	99.8	99.5	101.3	55.5	135.5
Portugal	93.3	121.2	113.6	50.2	135.2
Belarus	99.7	110.3	84.1	62.1	134.9
Greece	97.5	96.6	95.7	61.0	134.3
Iceland	99.0	101.2	107.5	48.1	133.9
Ukraine	99.7	90.5	96.8	57.2	133.6
Macao, China	94.6	104.1	87.1	66.4	133.2
Japan	99.0	100.7	102.5	47.7	132.1
Israel	95.6	113.9	93.2	52.7	131.2
Italy	98.6	100.9	95.9	49.9	130.9
Germany	99.0	103.2	98.9	46.2	130.8
Barbados	99.7	108.3	103.3	38.8	130.0
Switzerland	99.0	107.3	99.6	42.1	129.8
Hungary	99.4	101.6	98.2	39.8	127.7
Uruguay	97.8	108.3	101.4	37.7	127.7
Libya	82.6	114.1	104.8	58.1	127.1
Bulgaria	98.6	99.4	94.3	40.1	125.8
Kyrgyzstan	97.0	102.0	85.4	43.8	124.1
Kazakhstan	99.5	99.3	88.8	38.7	123.9
Czech Republic	99.0	104.3	94.7	29.8	122.1
Croatia	98.5	95.6	88.4	36.4	121.5
Chile	96.2	102.7	85.5	37.1	120.7
Brazil	88.1	148.5	107.5	17.9	120.2
Slovak Republic	99.7	103.0	87.3	30.3	120.2
Yugoslavia (Serbia & Montenegro)	94.0	98.8	88.7	36.0	119.1
Thailand	96.0	97.7	82.8	36.8	118.7
Georgia	99.0	92.0	78.6	36.1	117.9
Philippines	95.6	112.1	81.9	30.4	117.4
Cuba	97.0	100.3	89.1	27.4	117.3
Mongolia	98.6	98.7	76.1	34.7	117.2
Bolivia	87.2	113.6	84.4	39.1	117.0
Singapore	93.1	94.3	74.1	43.8	117.0
Lebanon	87.5	102.7	77.4	44.7	116.0
Armenia	98.6	96.3	86.5	25.8	116.0
Cyprus	97.6	96.7	93.4	22.2	115.9
Romania	98.4	98.8	82.3	27.3	115.5
Malta	92.9	105.7	90.0	25.1	114.8
Bahamas	95.6	92.2	91.5	24.8	114.6
Jordan	91.5	98.6	86.3	30.5	114.2
Guyana	98.7	120.2	90.5	11.6	114.2
Peru	90.9	121.3	81.7	25.8	113.7
Panama	92.6	110.0	69.2	33.6	112.7
Luxembourg	99.0	100.2	96.1	9.7	111.8
Moldova	99.1	85.3	72.4	28.7	111.0
Tajikistan	99.4	106.8	82.0	14.8	110.9
Bahrain	89.1	98.0	95.0	20.7	110.7
Hong Kong, China	94.0	94.3	71.9	27.4	108.8
Brunei Darussalam	91.7	106.3	87.7	13.4	107.7
Qatar	82.5	105.9	90.2	23.3	107.5
Mexico	92.0	110.3	73.5	20.5	107.3
Fiji	93.9	108.8	80.4	13.5	107.0
Costa Rica	95.9	108.4	66.8	20.5	107.0
Colombia	92.4	109.6	65.2	24.0	106.4
Jamaica	88.0	100.5	83.6	16.9	104.9
Dominican Republic	84.7	126.1	67.4	23.1	104.8
South Africa	86.4	105.1	86.4	14.6	104.5
Malaysia	88.9	95.2	69.6	26.0	104.5
Turkey	86.5	94.5	76.0	24.5	104.3
Venezuela	93.4	105.9	68.6	17.7	104.3
Paraguay	93.9	111.8	63.5	17.7	103.8
Kuwait	83.5	94.3	85.2	20.9	103.6
Samoa	98.7	102.5	74.5	6.5	102.5
Albania	86.5	106.6	78.4	15.1	102.5
Ecuador	92.4	116.9	59.2	17.6	102.3
Sri Lanka	92.3	110.4	80.8	5.3	102.2
Trinidad & Tobago	98.6	105.1	70.4	7.0	101.7
PLANETIA	**82.5**	**99.7**	**69.9**	**25.4**	**101.2**
HYPOTHETICA	**82.5**	**99.7**	**69.9**	**25.4**	**101.2**
Mauritius	85.5	106.0	79.5	11.3	100.2
Vietnam	93.0	103.4	69.7	10.0	99.9
Tunisia	74.2	111.6	79.1	22.8	99.3
China	86.9	113.9	68.2	12.7	99.3
Iran	79.2	92.1	81.0	19.2	98.5
Belize	93.8	117.6	70.7	0.9	98.3
Indonesia	88.4	110.9	57.9	15.1	97.2
United Arab Emirates	77.8	92.2	79.4	10.2	92.8
El Salvador	80.1	111.8	55.9	16.6	92.8
Egypt	57.8	96.6	85.3	36.7	92.6
Saudi Arabia	78.7	67.3	69.2	21.9	91.6
Namibia	84.0	106.0	61.4	7.3	91.0
Botswana	79.7	103.3	72.7	4.7	90.8
Algeria	69.9	108.4	71.6	15.1	90.4
Oman	75.8	82.9	78.5	7.5	88.4
Syria	76.9	111.6	43.3	15.7	86.1
Gabon	71.0	134.4	50.9	6.6	84.5
Zimbabwe	90.7	99.0	42.9	4.2	83.9
Myanmar	85.6	89.6	39.3	11.5	82.5
Nicaragua	67.5	104.7	56.6	11.8	82.0
Lesotho	84.9	124.3	33.7	2.4	81.5
Swaziland	81.6	100.4	45.2	5.2	81.3
Honduras	76.7	105.8	32.0	14.3	80.1
Kenya	85.2	96.0	32.0	3.5	75.5
Lao P.D.R.	67.3	114.8	40.6	4.3	74.3
Cameroon	74.7	106.7	32.6	5.2	74.2
Guatemala	70.5	103.0	33.1	8.4	73.4
India	59.6	98.8	48.5	10.6	73.0
Congo	83.9	85.5	32.0	3.7	72.7
Togo	60.9	124.2	36.5	3.7	70.6
Ghana	74.9	81.4	37.7	3.2	70.0
Cambodia	70.1	123.4	22.2	2.5	69.5
Uganda	69.8	136.4	12.7	3.2	68.6
Nigeria	68.2	96.5	30.3	4.0	67.3
Morocco	51.7	107.0	40.9	10.3	66.8
Zambia	80.7	78.7	25.6	2.4	65.9
Liberia	57.0	105.4	30.5	8.3	65.4
Rwanda	70.5	117.0	14.4	1.7	64.7
Malawi	62.7	131.3	17.3	0.4	64.1
Yemen	50.3	81.0	46.3	10.5	63.7
Nepal	45.2	121.6	43.9	5.4	63.3
Madagascar	68.9	104.2	14.3	2.2	61.4
Haiti	52.9	110.4	29.3	1.2	59.9
Sudan	61.0	58.7	32.0	6.8	58.6
Bangladesh	41.6	97.5	46.9	6.1	58.5
Papua New Guinea	66.0	77.5	22.7	2.1	57.9
Côte d'Ivoire	51.8	80.3	22.8	6.7	54.5
Eritrea	58.7	60.5	27.6	1.5	52.7
Benin	41.0	104.1	26.0	3.6	51.8
Tanzania	78.1	69.9	5.8	0.7	50.7
Pakistan	45.8	73.2	25.8	3.5	49.4
Gambia	40.4	78.9	34.3	1.9	49.3
Mozambique	47.8	98.9	13.3	0.6	48.8
Congo D.R.	65.6	49.6	18.4	1.4	48.3
Mauritania	41.7	86.5	21.0	3.1	47.5
Djibouti	67.2	40.3	19.6	1.2	46.8
Senegal	40.2	75.3	18.7	3.7	44.3
Chad	47.4	73.4	11.5	0.9	43.0
Central African Rep.	51.2	66.1	9.7	1.8	42.8
Guinea	41.0	77.1	13.9	1.3	41.8
Angola	42.0	63.9	19.1	0.7	41.4
Ethiopia	42.7	61.6	17.1	1.7	41.2
Mali	28.0	57.1	13.6	2.5	31.7
Burkina Faso	26.8	47.5	10.2	0.9	27.3
Niger	17.6	40.0	6.5	1.5	20.0

Illustrazione 19: Graduatoria ORBICOM dello sviluppo tecnologico

8 Ringraziamenti

Il lavoro di tesi rappresenta il culmine di un ciclo di studi che, se pur non definitivamente, marca la fine di una fase della vita. E' quindi spontaneo per me ringraziare chi ha reso possibile e piacevole la vita universitaria, una possibilità che non è stato affatto ovvio vivere. Ringrazio i miei genitori, Paolo e Guido.

Ringrazio il team che ha partorito questo lavoro alexxx::shiva::quarK: ...noi compagni di cluster; ciccio@ISF per la possibilità data, insieme a tutta ISF; il prof. Prandini, sempre a disposizione; cesco@hackLab, jimmy e chi più o meno indirettamente ha regalato disponibilità all'XM24; monica@RaccattaRAEE per l'appoggio; ciccio@lenzi e tutta la casa, insieme agli amici d'abruzzo (gli original e quelli ormai acquisiti... ;)) con cui ho costruito la mia vita universitaria; gli amici di sempre, con cui ho costruito il resto. Ringrazio, ognuno sa perché, girasole, i borderline, il LUG, fausto, piero e

`` `find /home/ | grep -i "amici|parenti"` ``.

9 Bibliografia

[1] Alessandro Delfanti, *Verso l'alta tecnocrazia,* Il Manifesto

[2] Orbicom*, From the Digital Divide to the Digital Opportunities,* 2005

[3] OCSE, *Information technology outlook*. Technical report, Parigi:OCSE, 2000

[4] Mytech.it*, Greenpeace lancia ecoguida ai prodotti elettronici,* Agosto 2006

[5] Golem, *Trashware HowTo,* 2001

[6] Ingegneria Senza Frontiere Torino*, Sito Ufficiale,* 2006

[7] Appunti di informatica libera*, Introduzione a LTSP*

[8] Gabriele Zucchetta, *Xterminal con Linux*, Linux Journal

[9] Barry Wellman Wenhong Chen, *Charting and bridging digital divide*. Technical report, NetLab, 2003.

[10] WSIS. Declaration of principles. In world summit on the information society, Ginevra Dicembre 2003

[11] Giampaolo Bonora. *Digital divide*, Osservatorio Internet, 2001

[12] Jesus M. Gonzales, *Free software/open source: Information society opportunities for europe,* Information Society Directorate General of UE, 2001

[13] ITU, World telecommunication development report, ITU, 2003

[14] BGLug, *Panoramica sul kernel linux,* In Linux Day, Bergamo 2003

[15] George Neville-Neil, *Internet Protocol Frequently Asked Questions*, 1994

[16] Pippa Norris, *Digital divide.* New York:Cambridge University Press, 2001

[17] ONU, *Information and communication technology development indices,* Technical report, UNCTAD, 2003

[18] Paolo Palmerini and Tommaso Pucci, *Gnu/linux e il software libero nei paesi in via di sviluppo*, Atti del Terzo Linuxday Italiano, Firenze 2003

[19] George Sciadas, *Monitoring the digital divide...and beyond,* Technical report, Orbicom, 2003

[20] Richard Stallman, *The gnu project,* Linux Journal, 1998

[21] Ruggero Russo, *Piattaforme software distribuite per il recupero di hardware obsolescente*, 2004

[22] Carlo Buzzi. *I numeri ed i rischi della spazzatura informatica,* Punto Informatico, 2003

[23] WWF-Consorzio Ecoqual'IT, *Le-w aste ladri di futuro - le cause e gli effetti della mancata gestione dei rifiuti tecnologici*, 2002

[24] Alberto Cammozzo. *Il progetto pluto refun*, 2003

[25] John Clay, *Reduced Cost Network Computing Through, The Use Of Thin Clients and Low Cost Software*, 2003

[26] Corrado Giustozzi, *Obsolescenza programmata,* Byte Italia, 1999

[27] Werner Heuser and Wade Hampton, *Linux Ecology How-to*, 1999

[28] Alberigo Massucci, Computer a caccia di ecologia, Punto informatico, 2003

[29] Miur, *Ricondizionamento di vecchi pc,* Servizio Osservatorio Tecnologico del Miur, 2001

[30] Andrea Mosca. *Lo smaltimento dei componenti elettrici ed elettronici obsoleti: aspetti normativi e tecnologici*, Università a degli studi di Torino; Facoltà a di economia, 2001

[31] Kuehr Ruediger and Eric Williams. *Computers and the environment: Understanding and managing their impacts*. Technical report, Kluwer Academic Publishers, October 2003

[32] Amon Barak and Oren La'adan, *The mosix multicomputer operating system for high performance cluster computing,* Technical report, Institute of Computer Science, The Hebrew University of Jerusalem, 1998

[33] Amon Barak and Amnon Shiloh, *Scalable cluster computing with mosix for linux,* Institute of Computer Science, The Hebrew University of Jerusalem, 1998

[34] Kris Buytaert. *OpenMosix howto,* 2001

[35] Kris Buytaert, Introducing OpenMosix, 2004

[36] Richard Ferri, *The seacrets of OpenMosix*, 2003

[37] Gian Paolo Ghilardi. Considerations on openmosix. Technical report, Università a degli studi di Cremona, 2002.

[38] Linux Journal, *Discussing shared memory,* Linux Journal, 2002

[39] MAASK. *Migshm.* University of Pune,India, 2002

[40] Steve McClure and Richard Wheeler, *Mosix:how linux clusters solve real world problems,* EMC2 Corporation, 2001

[41] Davide Meloni, *OpenMosix@openLabs HowTo*, 2004

[41] Roberto Premoli*, Test Prestazionali di un Cluster di Calcolo OpenMosix,* 2001

[42] Mathias Rothenburg, *Monitoring and administration of openmosix clusters*, Bruxells 2004

[43] Sistemisti-indipendenti.org. *Load Balancing Cluster-OpenMosix con GNU/Linux*, 2004

[44] Mulyadi Santosa. *Checkpointing and distributed shared memory in openmosix*, Aprile 2004