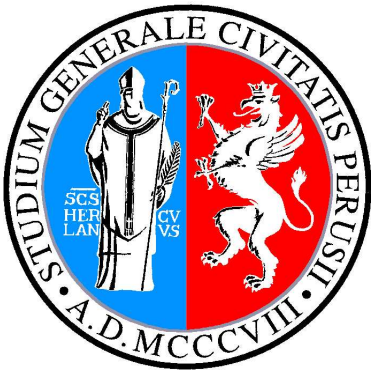

Università degli Studi di Perugia

Facoltà di Scienze Matematiche, Fisiche e Naturali

Corso di Laurea in Informatica

Tesi di Laurea

Installazione, configurazione e monitoraggio di un sito INFN-Grid.

Candidato

Igor Neri

Relatore

Prof. **Leonello Servoli**

Correlatore

Mirko Mariotti

Anno Accademico 2004-2005

Indice

Elenco delle figure

Introduzione

Introduzione generale

Nel corso degli ultimi anni la sezione INFN di Perugia ha deciso di installare un sito Grid. La prima versione installata, INFN-Grid 2.3.0, non si adeguava alla configurazione di rete della sezione, così, per permetterne il funzionamento, si è reso necessario modificare il software di installazione. Attualmente è stato necessario modificare la nuova versione per poter permettere il suo funzionamento con la configurazione di rete scelta, in quanto, con INFN-Grid 2.4.0, è stato introdotto un nuovo sistema di istallazione e configurazione. Le esigenze dei gruppi locali di ricerca, che partecipano al progetto Grid, hanno reso necessario modificare il software perchè esso consentisse agli utenti di utilizzare il sito come una farm locale senza la necessità di autenticarsi in Grid. In seguito è stato sviluppato un software che permettesse all'utente di consultare via web i dati relativi a vari parametri monitorati nel sito sotto forma di grafici.

Struttura della tesi

La tesi è composta da cinque capitoli, le conclusioni e le appendici:

- Capitolo 1: in questo capitolo viene introdotto il concetto di Computing Grid, ed in particolare di INFN-Grid.

- Capitolo 2: in questo capitolo viene illustrata la struttura attuale del sito INFN-Grid di Perugia prima del lavoro svolto.

- Capitolo 3: in questo capitolo viene illustrata la procedura standard di installazione di un sito INFN-Grid.

- Capitolo 4: in questo capitolo viengono illustrate le modifiche effettuate per far funzionare INFN-Grid con i Worker Node su rete nascosta e per la sottomissione locale di job.

- Capitolo 5: in questo capitolo vengono descritit gli strumenti di monitoraggio forniti da INFN-Grid, il software aggiuntivo installato nel sito di Perugia, il disegno e lo sviluppo di un nuovo strumento per il monitoraggio del sito Grid, ed i primi risultati delle analisi dei dati raccolti.

- Appendice A: nell'appendice A viene riportato il contenuto del file ig-site-info.def utilizzato nell'installazione del sito INFN-Grid di Perugia, le funzioni di configurazione modificate e le configurazioni delle code del batch-system locali e dello scheduler.

- Appendice B: nell'appendice B viene riportato il codice del pacchetto di monitoraggio sviluppato a Perugia.

Capitolo 1

Grid

1.1 Introduzione a Grid

1.1.1 La storia

Il problema della potenza di calcolo è stato affrontato più volte nel corso degli ultimi 30 anni; i primi calcolatori a mettere a disposizione un'elevata potenza di calcolo furono i mainframe[1] e soltanto i grandi enti potevano permettersi macchine di tale portata visti i costi elevati sia di acquisto che di manutenzione. L'architettura di queste macchine era spesso progettata ad-hoc da ditte specializzate. Tra la fine degli anni '80 e i primi anni '90, la diminuzione dei prezzi dei PC (Personal Computer) ne facilitò la diffusione presso i singoli utenti, mentre parallelamente nasceva l'architettura SIMD[2] destinata alle macchine dedicate al supercalcolo. Già nei primi anni '90, si pensò di sfruttare la potenza di calcolo offerta da ogni singola workstation e la tecnologia per l'interconnessione delle stesse, ormai a basso costo, per creare delle macchine parallele virtuali; nacquero così i primi cluster[3]. Il punto di forza dei cluster non era la potenza di picco di ogni singola macchina o architetture particolari bensì:

- L'uso di macchine destinate all'uso da ufficio come nodi di calcolo.
- L'uso di tecnologie di rete standard (ethernet).
- L'uso di un sistema operativo non progettato specificatamente per il calcolo ad alte prestazioni (Linux o FreeBSD).

Questi requisiti permettevano di avere elevata potenza di calcolo associata a bassi costi di manutenzione e di investimenti iniziali, con il vantaggio di un'elevata scalabilità.

[1] Calcolatori seriali di enormi dimensioni.
[2] Single Instruction Multiple Data.
[3] Un insieme di macchine configurate in modo tale da comunicare attraverso una rete locale LAN.

Gli investimenti precedentemente impiegati per la ricerca di nuove architetture che permettessero maggiori prestazioni, venivano ora destinati allo sviluppo di software. Nascono infatti nuovi software per il calcolo distribuito come MPI (Message Passing Interface), PVM (Parallel Virtual Machine) e HPF (High Performance Fortran) e paradigmi di programmazione parallela quali Task Farm e Pipeline, assieme ad un software (middleware) in grado di minimizzare l'eterogeneità delle risorse di calcolo e creare un singolo strato visibile all'utente.

1.1.2 Cos'è una Computing Grid

Secondo la definizione del progetto Globus[4] "Le Grid sono ambienti persistenti che rendono possibile realizzare applicazioni software che integrino risorse di strumentazione, di visualizzazione, di calcolo e di informazione che provengono da domini amministrativi diversi e che sono geograficamente distribuite". La definizione di Computing Grid (d'ora in avanti Grid per semplicità) si basa quindi sul concetto di condivisione di risorse hardware e software fra utenti o gruppi di utenti in una WAN[5].
I primi standard relativi a Grid furono definiti nei primi anni '90 da sviluppatori e ricercatori prima nel Grid Forum e in seguito nel Global Grid Forum.
Oltre ai vantaggi portati dall'aumento della potenza di calcolo, Grid permette di:

- Integrare e coordinare risorse non sempre appartenenti allo stesso ambito, stabilendo un insieme di regole e permessi di condivisione garantite da procedure di autenticazione e autorizzazione.
- Gestire le risorse in modo trasparente in quanto agli utenti è permesso accedere alle risorse remote come se fossero locali.
- Garantire servizi in termini di performance, sicurezza, tolleranza agli errori, disponibilità e tempi di risposta.
- Aprire sessioni in qualsiasi macchina agli utenti autorizzati ovunque essi si trovino, in quanto è una tecnologia disponibile ovunque.

La dislocazione geografica dei calcolatori rende poco adatta Grid per programmi paralleli, visti i tempi di latenza e vengono quindi sviluppate applicazioni dove a crescere è l'utilizzo delle CPU[6] (High Peak) piuttosto che l'utilizzo della banda per lo scambio di messaggi e/o dati (High Throughput). Applicazioni di questo genere si basano sul

[4]Globus Alliance è una comunità di individui ed organizzazioni che sviluppano le tecnologie fondamentali di Grid. http://www.globus.org
[5]Wide Area Network, una rete distribuita in un territorio geografico.
[6]Central Processing Unit.

paradigma farmer-worker[7], come simulazioni di tipo Montecarlo o, più in generale, quelle in cui è necessario eseguire una stessa sequenza di operazioni su di un enorme insieme di dati. In questo modo, poiché i nodi svolgono i calcoli in modo indipendente, non ci sono particolari requisiti sulle modalità di comunicazione.
Negli ultimi anni vari tipi di Grid sono in fase di progettazione, costruzione e sperimentazione, e fra non molti anni la la loro integrazione potrà dare vita a quella che, forse, sarà la Grid globale.
Attualmente, questa tecnologia che in passato era utilizzata maggiormente in ambito scientifico, si è rivelata molto efficiente anche per grosse multinazionali, laboratori privati, organi governativi, enti pubblici che possono così beneficiare della conseguente capacità di aggregazione e accesso a risorse distribuite.

1.1.3 Tipologie di applicazioni su Grid.

Esistono diverse tipologie di applicazione:

- Supercalcolo Distribuito, dà la possibilità di poter utilizzare più supercomputer o cluster garantendo di superare le limitazioni di calcolo esistenti.
- On Demand Computing, le risorse vengono rese disponibili quando un utente ne fa richiesta e vengono rilasciate e rese di nuovo disponibili non appena l'utente cessa di utilizzarle.
- Data Intensive Computing, permette la gestione di enormi quantità di dati distribuite geograficamente.
- Calcolo Collaborativo, ha lo scopo di facilitare e favorire la comunicazione e la collaborazione tra coloro che utilizzano le risorse e che appartengono allo stesso spazio o dominio virtuale.

1.1.4 Servizi richiesti ad una Grid

I servizi messi a disposizione da una griglia computazionale sono:

- **Sicurezza del Sistema**, in modo da stabilire con certezza l'identità degli utenti che possono accedere e il modo di utilizzo delle risorse. Questa autenticazione

[7]Il modello computazionale Farmer-Worker viene comunemente utilizzato nel caso di applicazioni decomponibili in task di grana medio-grande (Bag of Work) tra di loro indipendenti, cioè che per il loro svolgimento non necessitano di comunicare con altri. Un Farmer genera un insieme di Bag of Work che vengono messe a disposizione dei Worker. Ogni Worker che si trova in stato ideale estrae una Bag of Work dall'insieme ed effettua l'esecuzione del task richiesto. Alla conclusione del task, il Worker restituisce i risultati ottenuti al Farmer e, trovandosi, di nuovo nello stato ideale, torna a controllare se esiste altro lavoro da fare.

avviene tramite un certificato rilasciato da un apposito ente denominato CA (Certification Autority).

- **Affidabilità**, garantisce all'utente di poter contare su un sistema che esegue le richieste senza interruzioni e perdita di dati.
- **Consistenza**, la tecnologia è basata su protocolli e interfacce standard, quindi è possibile eseguire applicazioni non scritte appositamente per essere utilizzate in Grid, semplicemente imparando le modalità di utilizzo di Grid.
- **Accounting**, gestione della ripartizione delle risorse in base a vari parametri.

1.2 Le realtà esistenti.

Il primo software in grado di implementare una Grid fu sviluppato negli anni '90 e venne presentato in una conferenza sul calcolo distribuito in America con il nome di I-WAY, ed era in grado di mettere in comunicazione 17 farm[8] scientifiche.
Oggi esistono vari progetti che hanno lo scopo di implementare una griglia:

- CONDOR. É un progetto sviluppato all'università del Wiscolnsin da Miron Livny dal 1985. Ha dimostrato che all'interno dell'ateneo molte postazioni di lavoro inattive potevano essere utilizzate attraverso il Multitasking di UNIX. (CPU Intensive).
- Seti@home. Sviluppato da David Anderson; l'idea era quella di utilizzare l'enorme quantità di pc, anche di privati, dislocati in tutto il territorio mondiale per analizzare i dati provenienti dal radiotelescopio di Arecibo, sfruttandone i tempi morti; il tutto era possibile grazie alla ridotta quantità di dati che ogni singola postazione doveva scambiare con il server. (CPU Intensive).
- GLOBUS. É un toolkit Open Source su cui si basano tutti i moderni software di Grid che viene rilasciato con licenza GTPL (Globus Toolkit Public License).
- EUROPEAN DATAGRID. Questo progetto, ormai terminato, è stato finanziato dalla Comunità Europea e comprende sia realtà nazionali di ricerca come l'INFN[9] che partner industriali ed ha prodotto una prima versione di middleware, LCG.
- OSG. Open Science Grid è un progetto sviluppato negli USA, è una Grid che supporta il calcolo scientifico tramite una collaborazione fra enti di ricerca, sviluppatori di software e network providers.

[8] Farm è un termine inglese utilizzato in informatica per indicare una serie di server collocati in un ambiente unico in modo da poterne centralizzare la gestione, la manutenzione e la sicurezza.
[9] Istituto Nazionale Fisica Nucleare.

- EGEE. Progetto successore di EUROPEAN DATAGRID, finanziato dalla Comunità Europea, ha come scopo quello di realizzare un'infrastruttura Grid robusta e coerente, a partire da LCG.

- LCG. É un progetto concepito al CERN[10] per gestire tutto il computing per il progetto LHC[11], con lo scopo di realizzare un'infrastruttura Grid globale integrando i vari centri di ricerca scientifica sparsi in Europa, Asia ed America. Lo sviluppo della prima fase, iniziato nel 2002, terminerà nel 2008 accompagnando la prima fase operativa di LHC.
  Attualmente, visto che l'acceleratore di particelle non è ancora completo, l'hardware gestito dal progetto LCG viene utilizzato per effettuare dei test e delle analisi su dati simulati, in modo da poter testare, oltre all'infrastruttura di Grid, anche gli applicativi che verranno utilizzati per analizzare i dati prodotti dall'acceleratore non appena sarà ultimato. Al momento circa 100 ricercatori per esperimento lavorano alla simulazione in sedi dislocate in tutta Europa, mentre a regime saranno circa 2000; si stima che la quantità di dati prodotti dall'LHC sarà di circa 1 Pbyte per anno. (Data Intensive e CPU Intensive).

- INFN-GRID è la personalizzazione italiana di LCG, nata nel 1999 con lo scopo di realizzare la prima Grid italiana usando la rete del GARR[12], comprende una ventina di siti dislocati presso le università nazionali, dove sono presenti le sedi INFN. D'ora in avanti con il termine Grid ci riferiremo a questo tipo particolare di Grid.

1.3 Architettura di Grid

Per architettura si intende l'insieme delle componenti API[13] e SDK[14] che definiscono le interfacce e le librerie che compongono il sistema, evidenziandone gli scopi, le funzionalità e le interazioni che avvengono tra le parti in questione.

1.3.1 Livelli

L'architettura di Grid è formata da quattro diversi livelli (vedi figura 1.1):

- LIVELLO APPLICATIVO: è l'interfaccia con la quale l'utente può interagire

[10] Il CERN nasce nel 1945 ed è il più importante centro di ricerca per la fisica delle particelle.
[11] Large Hadron Collider, è il nuovo acceleratore di particelle sviluppato dal CERN che dovrebbe iniziare a funzionare nel 2007.
[12] Gruppo Armonizzazione Reti Ricerca.
[13] Appication Program Interface.
[14] Software Development Kit.

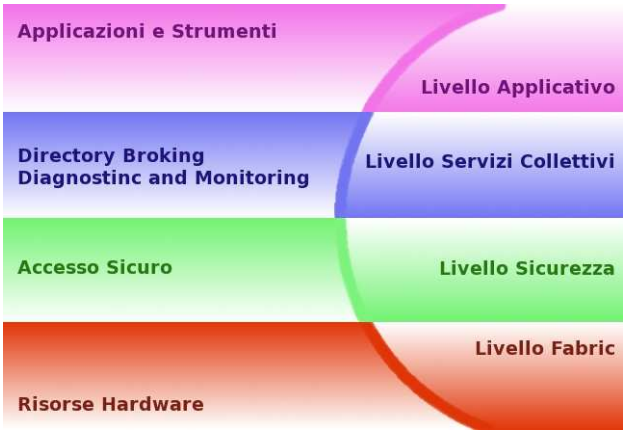

Figura 1.1: Livelli e funzionalità che caratterizzano l'architettura di Grid.

con il sistema, e contiene le applicazioni rivolte all'utente ed il meccanismo della gestione delle organizzazioni virtuali.

- LIVELLO SERVIZI COLLETTIVI: contiene i servizi API utilizzati per la gestione e la condivisione delle risorse che rendono nota all'utente l'allocazione delle risorse di cui ha bisogno. É composto da:
  - Directory Service, gestisce la visibilità delle risorse in base alla VO (Virtual Organization) (sez. 1.3.3) di appartenenza.
  - Monitoring, si occupa di controllare l'utilizzo delle risorse del sistema.
  - Diagnostic, ha il compito di dare un supporto diagnostico in caso di necessità.
- LIVELLO PROTOCOLLI (Resource and Connectivity):
  - RESOURCE, definisce l'insieme dei protocolli API e SDK che si occupano del controllo e del lancio di operazioni su delle risorse. L'insieme di questi protocolli si avvale dei protocolli di comunicazione appartenenti alla Connectivity, determinando due classi di protocolli: Informations Protocols che visualizzano lo stato delle risorse e Management Protocols che gestiscono l'accesso e l'utilizzo delle risorse.
  - CONNECTIVITY, definisce i protocolli di comunicazione e autenticazione per le transazioni di rete abilitando lo scambio dei dati con il livello Fabric. I primi permettono lo scambio di dati tra le risorse del livello inferiore, i secondi forniscono un meccanismo di autorizzazione e protezione delle comunicazioni.

- LIVELLO FABRIC: è composto dalle risorse distribuite tramite la rete, possono essere entità logiche o fisiche.

1.3.2 Le strutture di gestione di Grid

All'interno di Grid ci sono varie strutture, organizzate in modo gerarchico, che hanno il compito di gestire e controllare il funzionamento dell'intera Grid.

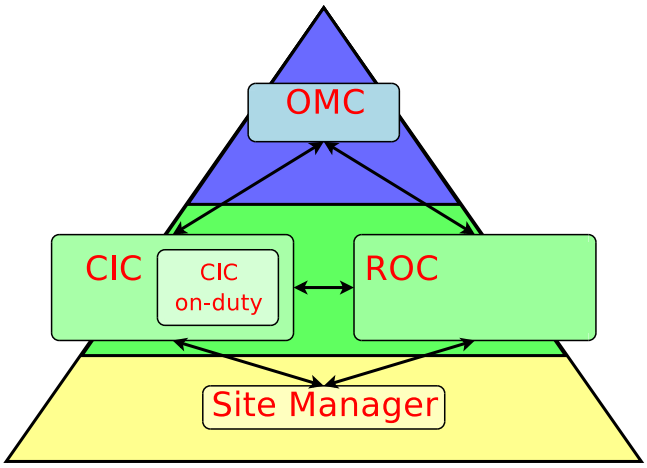

Figura 1.2: Schema della struttura di gestione di Grid.

Il livello più alto è rappresentato dall'OMC (Operation Manager Centre), è situato al CERN e consiste in una piccola squadra di gente con una notevole esperienza di infrastrutture Grid. Ha responsabilità generiche di coordinazione, del corretto funzionamento della Grid e definisce la politica operativa che deve essere attuata dai CIC (Core Infrastructure Centre) e dai ROC (Regional Operation Centre).
I CIC hanno il ruolo di far funzionare l'infrastruttura essenziale di Grid in ogni singolo sito e forniscono controllo e servizi operativi di analisi guasti. Sono quattro situati al CERN, nel Regno Unito, in Italia e in Francia; i CIC svolgono anche il compito di supportare i ROC nella risoluzione di problemi. Il CIC-on-Duty è uno dei CIC (a rotazione settimanale) che è responsabile di mantenere un operatore in servizio 24/24 ore 7/7 giorni.
I ROC sono attualmente nove, hanno il ruolo di coordinare e supportare all'interno della nazione di riferimento sia l'implementazione di nuovi siti che di nuove versioni di middleware, oltre che di coordinare e supportare le operazioni dei vari siti presenti in quella nazione.
I Site Manager sono i responsabili locali di un sito Grid, hanno il compito della manutenzione ed upgrade del sito oltre a quello di risolvere i problemi cooperando con i

CIC e i ROC.

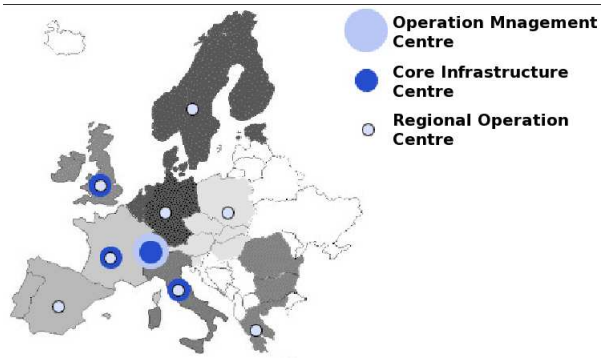

Figura 1.3: Dislocazione geografica di OMC, CIC e ROC.

1.3.3 Il concetto di Virtual Organization

Le Virtual Organization (VO) sono delle comunità di utenti che concordano con il management della Grid, le politiche d'uso e di condivisione delle risorse, accedendo all'infrastruttura di produzione.
Le VO, organizzate gerarchicamente, sono una struttura dinamica in quanto il numero degli utenti che ne fa parte può variare nel corso del tempo insieme alle risorse messe a disposizione. Inoltre un utente può appartenere a più VO contemporaneamente ed esistono relazioni fra VO diverse.
All'interno del Globus Toolkit (sez. 1.4) le VO sono implementate con il meccanismo dei grid-mapfile, un file che contiene l'elenco completo degli utenti appartenenti a ciascuna VO. Questo file è organizzato per righe, ciascuna delle quali contiene informazioni su un determinato utente, come il nome e il contact name per essere riconosciuto dal sistema.
Inizialmente le VO erano rappresentate in un albero LDAP (Lightweight Directory Access Protocol) contenente le informazioni sugli utenti e gruppi di appartenenza. Le informazioni presenti sull'albero LDAP possono essere utilizzate per la creazione automatica del grid-mapfile. L'unica informazione che è possibile ricavare dalla consultazione del server VO LDAP è l'appartenenza di un utente ad un gruppo. Non è possibile fare distinzione in base al ruolo e alle potenzialità o capability dell'utente. Molto spesso gli utenti appartengono a piu di una VO quindi il sistema risulta poco flessibile.

É stato quindi sostituito dall'Authorization Working Group dell'INFN con un nuovo

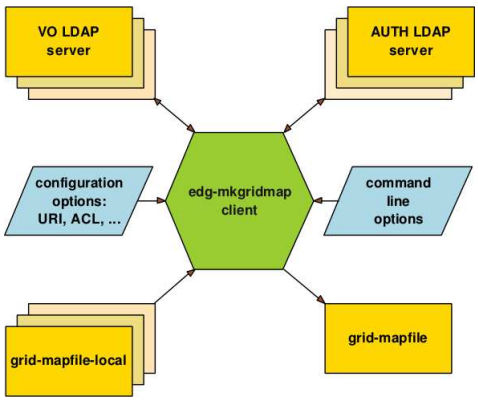

Figura 1.4: Schema di gestionde delle VO tramite LDAP.

modello chiamato VOMS (Virtual Organization Membership Service) dove ogni server VOMS può contenere informazioni relative a più VO. Il VOMS fornisce il supporto ai gruppi, ai ruoli e alle capability dell'utente.

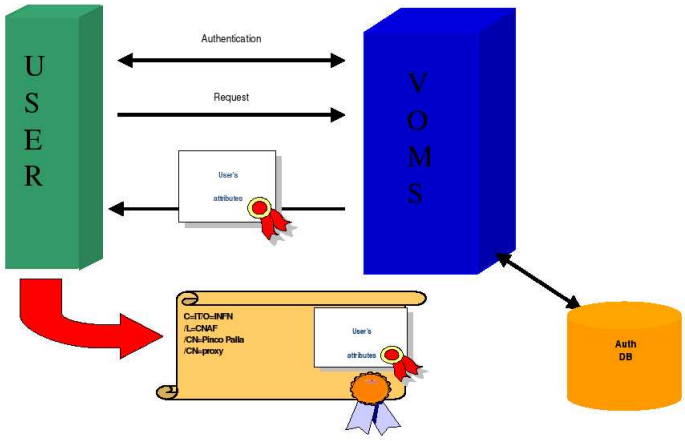

Figura 1.5: Schema gestione delle VO tramite VOMS.

1.4 Globus Toolkit

Globus Toolkit è un progetto sviluppato con lo scopo di creare un insieme di servizi utilizzabili in modo indipendente o in modo congiunto per poter sviluppare applicazioni Grid.
Si tratta di un insieme di tools che facilita la costruzione delle computing Grid abilitando le applicazioni alla gestione delle risorse di calcolo eterogenee e distribuite;

inoltre definisce i protocolli e le interfacce standard ed è completamente Open Source protetto dalla Globus Toolkit Public License. É costituito da quattro parti differenti:

1.4.1 Gestione della sicurezza.

GSI (Grid Security Infrastructure) è un meccanismo di autenticazione che si occupa della sicurezza nell'ambiente Grid, garantendo l'accesso solamente agli utenti autorizzati. É basato sullo standard dei certificati X.509, la crittografia asimmetrica e sul protocollo SSL.

1.4.2 Gestione delle risorse.

Il GRAM (Globus Resource Allocation Manager) si occupa della gestione delle risorse ed ha il compito di abilitare un accesso sicuro e controllato alle risorse computazionali gestendo l'esecuzione remota di operazioni sulle risorse stesse. Il Globus Toolkit mette a disposizione una struttura a livelli per la gestione delle risorse computazionali: ai livelli più alti ci sono tutti i servizi per la gestione di un insieme di risorse, mentre ai livelli inferiori troviamo tutti i servizi di allocazione delle singole risorse. Il GRAM rappresenta il livello più basso di questa struttura.
Il gatekeeper è un processo del GRAM che gestisce la richiesta di un nodo cliente inviandola al job manager, il quale dopo aver creato un processo per la richiesta ne controlla l'esecuzione comunicandone lo stato all'utente remoto (vedi fig. 1.6).
Il protocollo usato dal GRAM è basato su una serie di RPC[15] fondate sul protocollo HTTP[16].

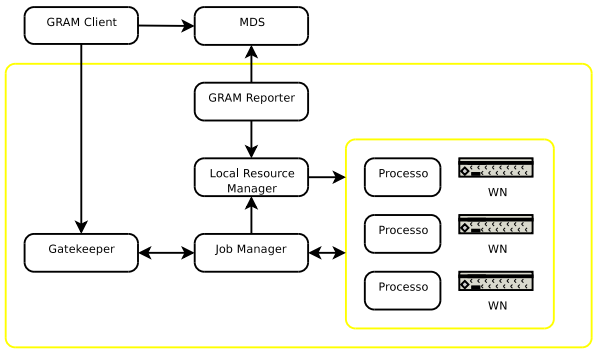

Figura 1.6: Architettura del GRAM.

[15]Remote Procedure Call, chiamata a procedura remota.
[16]HyperText Transport Protocol.

1.4.3 Gestione dei dati.

La gestione dei dati avviene grazie a tre sistemi:

- GRIDFTP, si tratta di un protocollo utilizzato per realizzare lo scambio di files tra le varie risorse all'interno della griglia. É stato realizzato migliorando ed estendendo il protocollo FTP[17], permettendo così di aumentare la capacità e la sicurezza del trasferimento dati grazie all'utilizzo dei meccanismi definiti dal GSI.
- GLOBUS REPLICA CATALOG, al fine di massimizzare la velocità di accesso ai dati, questo sistema, permette la replicazione degli stessi su diversi storage, anche remoti.
- GLOBUS REPLICA MANAGER, permette la replicazione e il trasferimento dei dati grazie alla cooperazione del protocollo GRIDFTP con il REPLICA CATALOG.

1.4.4 Gestione delle informazioni.

Il servizio GIS (Grid Information Service) raggruppa le informazioni di stato delle varie risorse e viene suddiviso in tre principali servizi:

- MDS, Monitoring and Discovering Service.
- GIIS, Grid Index Information Service.
- GRIS, Grid Resource Information Service.

Il primo fornisce un servizio di informazione creando un sistema di directory di servizi che viene utilizzato per reperire informazioni sullo stato delle risorse presenti.
Ogni risorsa dispone di un software, GRIS, che reperisce le informazioni sul proprio stato e le invia periodicamente ad un server GIIS. Generalmente questo è attivo su una sola macchina della farm e fa parte di una struttura gerarchica formata da più GIIS server, ognuno dei quali provvede a recuperare le informazioni e ad inviarle al GIIS di livello superiore fino ad arrivare a quello di livello massimo chiamato top MDS (vedi fig. 1.7).

1.5 Componenti di Grid (INFN-GRID)

Grid è costituita da vari siti dislocati nel territorio e da alcuni elementi esterni che ne permettono il funzionamento. Questi ultimi sono:

[17] File Transfert Protocol.

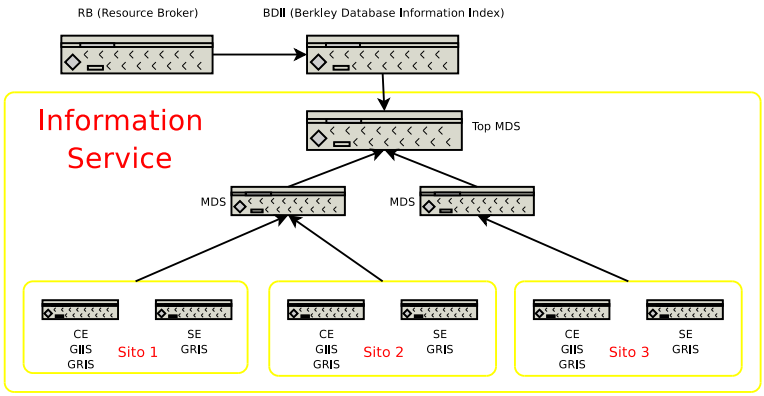

Figura 1.7: Schema dell'architettura dell'Information Service.

- RB (Resource Broker); è la macchina che si occupa di fare il matching dei job con le risorse disponibili, in base alle specifiche espresse nella descrizione del job. Se non viene specificata una risorsa su cui eseguire il job, il RB ne sceglie una tra quelle disponibili. Inoltre il RB funziona anche da database contenente lo stato attuale dei job sottomessi, che l'utente su una UI può interrogare tramite gli appositi comandi.

- BDII (Berkley Database Information Index); è un database che serve per memorizzare lo stato delle risorse. Infatti al momento della richiesta da parte di un RB effettua una interrogazione simultanea a tutte le risorse coinvolte e memorizza il loro stato all'interno del database; inoltre ha anche funzionalità di cache per le informazioni statiche come tipi di hardware, sistema operativo ecc.

- UI (User Interface); è la macchina che gli utenti usano in interattivo (in essa risiedono gli account utente) e che definisce in sostanza l'interfaccia con cui l'utente puo sottomettere un job su Grid. Questa può essere anche un pc desktop o un notebook.

All'interno di un singolo sito possiamo invece trovare diversi calcolatori che differiscono fra loro a seconda dei compiti che devono svolgere:

- CE (Computing Element); è la via d'accesso alla farm, infatti su questa macchina risiedono il gatekeeper e lo scheduler che gestisce la sottomissione dei job ai WN.

- SE (Storage Element); è la macchina che si occupa della memorizzazione dei dati, del loro accesso e della loro replica.

- WN (Worker Node); è il nodo che esegue effettivamente i calcoli.

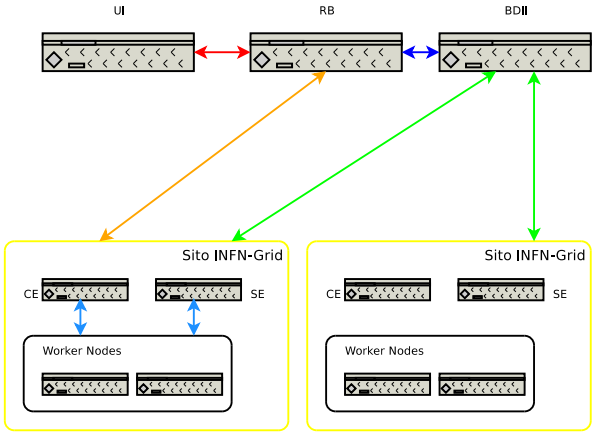

Figura 1.8: Esempio di sottomissione di un job.

In figura 1.8 si vede un esempio di sottomissione di un job, la UI contatta il RB chiedendo di sottomettere un job, questo interroga il BDII che tiene costantemente aggiornate le informazioni sui vari siti comunicando al RB lo stato degli stessi. Il RB sceglie un sito fra quelli che risultano disponibili secondo il BDII e vi sottomette il job. Dalla UI è poi possibile interrogare il RB per chiedere lo stato e l'output dei job sottomessi.

Capitolo 2

Il sito INFN-Grid di Perugia

Le esigenze dei vari gruppi di ricerca e dei singoli ricercatori che lavorano a Perugia nell'ambito delle tematiche proprie dell'INFN, dal punto di vista del calcolo, sono variegate; si va dal calcolo parallelo al calcolo distribuito, al calcolo che contenga una grande quantità di I/O.
Dal punto di vista gestionale, è importante poter contare su una gestione unificata delle risorse hardware, viste anche le limitate disponibilità di personale.
Nel 2004 la sezione INFN di Perugia ha deciso di installare un proprio sito INFN-Grid così da permettere ai gruppi di ricerca locali di usufruire delle potenzialità di Grid. Inizialmente solo il gruppo CMS ha partecipato all'installazione del sito, apportando anche modifiche alla configurazione originale di INFN-Grid 2.3.0, quella attualmente installata, per permetterne il funzionamento con WN su rete nascosta. Nel corso dell'ultimo anno anche i gruppi NA48 e THEOPHYS hanno messo a disposizione le proprie strutture per potenziare il sito INFN-Grid di Perugia e per poter sfruttare i vantaggi che Grid mette a disposizione.
É importante inoltre prevedere una struttura per il batch system che permetta un accesso duale, sia attraverso il versante Grid che quello normale, perché non tutte le richieste di calcolo sono immediatamente trasferibili in ambiente Grid, e perché occorre anche prevedere apporti hardware da gruppi che non vogliono entrare in Grid, ma che vogliono usufruire della gestione unificata delle risorse.
Il sito Grid di Perugia comprende una Grid di produzione affiancata da una Grid di sviluppo dove vengono fatti i test prima di apportare modifiche alla Grid di produzione.

2.1 La Grid di produzione

La struttura della Grid di produzione è in continua evoluzione, le macchine sono interconnesse fra loro attraverso una rete privata mentre alcune macchine, con doppia

interfaccia di rete, sono connesse alla rete internet; in particolare le macchine connesse ad internet sono: le User Interface, il Computing Element, lo Storage Element e il nodo per l'installazione; sono invece connessi solo tramite rete privata i Worker Node e i File Server.
Le macchine apportate dai vari gruppi sono rispettivamente:

- GRID: CE, SE, Install Server.
- CMS: nove WN, due UI e due fileserver con sei TB di spazio disco.
- NA48: nove WN, due UI ed un fileserver con quattro TB di spazio disco.
- CMS: quattro WN ed una UI.

I WN sono:

- Cinque biprocessori pentium III ad 1Ghz.
- Sei biprocessori pentium IV a 2,4Ghz.
- Undici biprocessori pentium IV a 2,8Ghz.

2.2 La Grid di sviluppo

La Grid di sviluppo è composta da sei macchine, queste sono interconnesse fra loro con due reti private, una che rispecchia la rete esterna della Grid di produzione e una interna, per fare ciò abbiamo usato gli stessi indirizzi IP della Grid di produzione. La Grid di sviluppo comunica con l'esterno tramite un gateway/firewall, questo è una macchina *diskless* (senza disco) dotata di due interfacce di rete, una collegata alla rete del dipartimento di Fisica e una collegata alla rete virtuale "pg.infn.it".

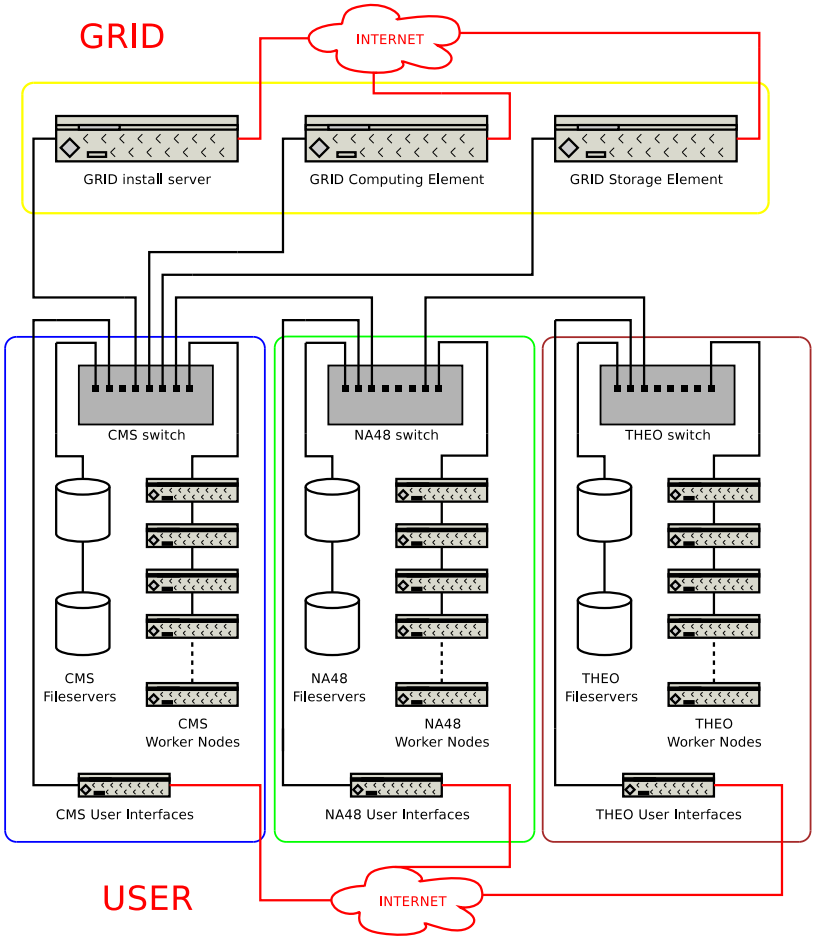

Figura 2.1: Strutura fisica del sito INFN-Grid di Perugia

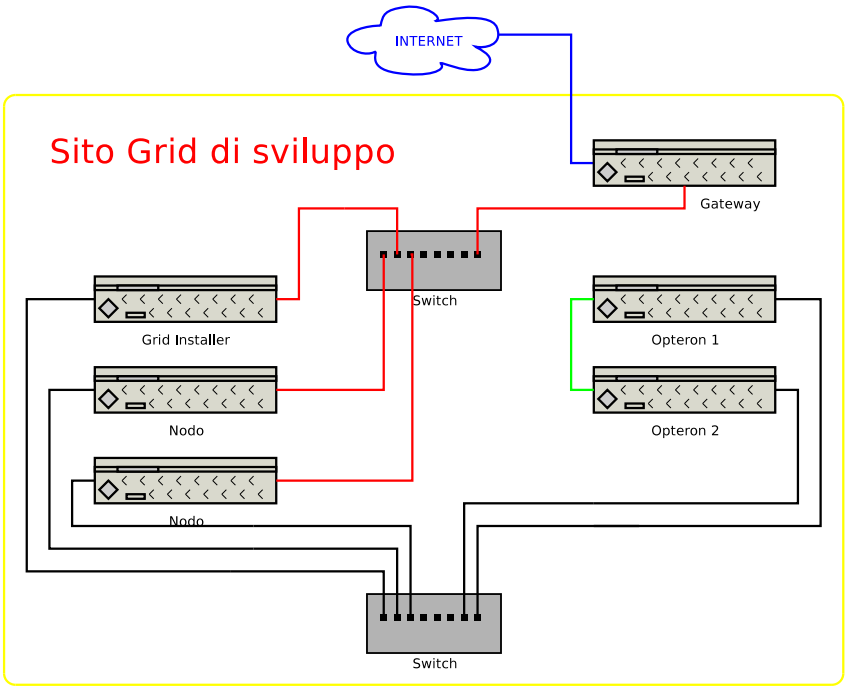

Figura 2.2: Strutura fisica della Grid di sviluppo

Capitolo 3

Installazione di un sito INFN-Grid 2.4.0

I passi per l'installazione di INFN-Grid 2.4.0 (d'ora in avanti con il termine INFN-Grid ci riferiremo alla versione 2.4.0 se non diversamente specificato) sono: l'installazione del sistema operativo, l'installazione del software, la sua configurazione e la fase di test. INFN-Grid richiede come sistema operativo Scientific Linux 3[1] (SL3), che può essere installato su tutte le macchine manualmente o mediante un'apposita macchina (Install Server) configurata in modo tale da automatizzare l'installazione della farm.

3.1 Install Server

Il setup dell'Install Server è opzionale nell'installazione di un sito INFN-Grid, l'unico suo scopo è facilitare l'installazione di un elevato numero di macchine con caratteristiche simili e ridurne il tempo di installazione.
Al primo avvio ogni macchina esegue il boot via PXE (Preboot eXecution Environment), contatta l'Install Server che gli fornisce un indirizzo IP in base al proprio MAC-address[2], un kernel per il primo avvio e alcuni file di configurazione. Il kernel usato è quello di installazione di SL3, che, grazie al sistema di installazione *anaconda* ed alcuni file di configurazione configura il sistema (partizioni, rete, indirizzo del server per l'installazione, etc...), richiede i pacchetti necessari all'installazione all'Install Server e li installa. Al termine dell'installazione ogni nodo contatta via web l'Install

[1] Distribuzione GNU/Linux basata su RedHat modificata dal CERN.

[2] Il MAC-address viene detto anche indirizzo fisico o indirizzo ethernet, ed è un codice di 48 bit (6 byte) assegnato in modo univoco ad ogni scheda di rete ethernet prodotta al mondo. Per questo tipo di indirizzi di solito si preferisce la notazione esadecimale anche per differenziarla dagli indirizzi IP che usano la notazione decimale.

Server eseguendo lo script `/var/www/cgi-bin/install_ack.cgi` che modifica la modalità di boot del nodo da boot via rete a boot da hard-disk.

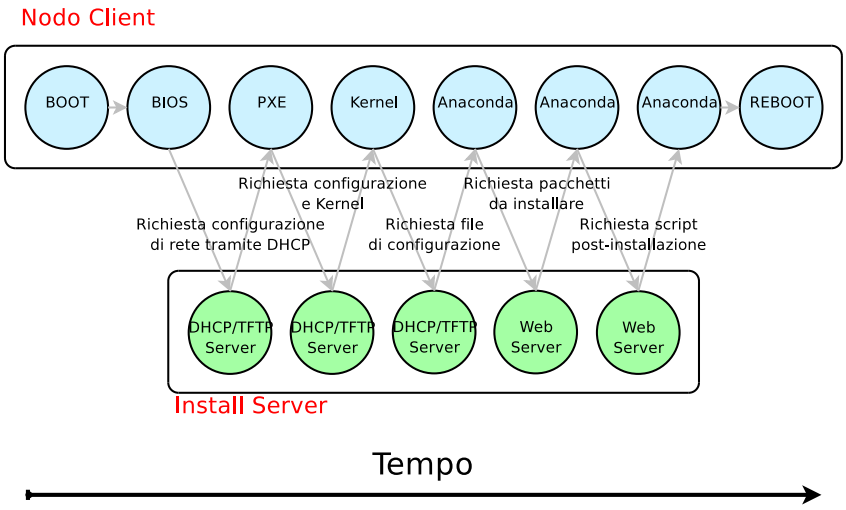

Figura 3.1: Sequenza di boot dei nodi durante l'installazione dall'Install Server.

Le componenti software richieste dall'Install Server sono quindi:

- Un server DHCP (Dynamic Host Configuration Protocol) per fornire informazioni ai nodi riguardo la configurazione di rete degli stessi (necessario solo in fase di installazione);
- Un server TFTP (Trivial File Transfer Protocol) per fornire i file richiesti durante la prima fase di installazione del SO (Sistema Operativo);
- Il *bootloader* pxelinux per l'installazione via PXE;
- I file contenenti kernel e il *ramdisk image* (initrd);
- Un file kickstart dove sono descritti tutti i parametri necessari per effettuare l'installazione del SO;
- Un web server contenente il repository locale del sistema operativo e tutto il software necessario;

I file di esempio e alcuni tools per facilitare l'installazione dell'Install Server sono contenuti nei pacchetti `ig-installserver` e `ig-yam` forniti rispettivamente da LCG e INFN-Grid.

3.1.1 Configurazione del server DHCP

Tramite il server DHCP i nodi ottengono la propria configurazione di rete in base al proprio MAC-address, e quindi come prima cosa è necessario installare il pacchetto del server DHCP per poi passare alla sua configurazione partendo dal file di esempio:

```
/etc/dhcpd.conf.example
```

Una volta configurato il servizio questo deve essere reso attivo con i comandi:

```
/sbin/chkconfig dhcpd on
/sbin/service dhcpd start
```

3.1.2 Configurazione del server TFTP

Quando i client eseguono il boot via PXE per prima cosa contattano il server DHCP per ottenere una configurazione di rete, poi i client necessitano di un kernel e di un *ramdisk* iniziale. Questi file vengono caricati utilizzando il protocollo TFTP, quindi un server TFTP deve essere in esecuzione nell'Install Server. Dopo l'installazione del server TFTP questo deve essere configurato modificando il file:

```
/etc/xinetd.d/tftp
```

e deve essere attivato tramite i comandi:

```
/sbin/chkconfig xinetd on
/sbin/service xinetd restart
```

I file necessari al boot via *pxelinux* risiedono della directory `/tftpboot/`, questi file sono forniti dal pacchetto `syslinux`.

3.1.3 Configurazione del server web

I pacchetti necessari all'installazione del SO, del middleware ed i file *kickstart* per i nodi vengono presi mediante protocollo HTTP da un server web. Nel nostro caso utilizzeremo *Apache 2*. Al termine dell'installazione e configurazione del server web è necessario renderlo attivo tramite i comandi:

```
/sbin/chkconfig httpd on
/sbin/service httpd restart
```

3.1.4 Aggiornamento dei pacchetti del repository tramite YAM

Il pacchetto YAM permette di mantenere un mirror locale di un repository APT[3]/YUM[4]; tramite il pacchetto *ig-yam* la configurazione standard di YAM viene personalizzata per permettere di creare un repository contenente oltre ai pacchetti del SO anche i pacchetti per l'installazione di INFN-Grid.
Tramite il comando:

```
yam -guxvv
```

viene eseguita la sincronizzazione dei pacchetti fra i server elencati nel file di configurazione (nel nostro caso linuxsoft.cern.ch, grid-deployment.web.cern.ch e grid-it.cnaf.infn.it rispettivamente per il SO, i pacchetti di LCG e i pacchetti di INFN-Grid) ed il repository locale, viene generata la lista dei pacchetti locali disponibili e le dipendenze fra essi, e vengono generati tutti i link simbolici nella directory del server web.

3.1.5 Kernel e initrd

Il kernel e l'initd vengono prelevati tramite TFTP per iniziare l'installazione via rete. Questi file sono indicati nel file di configurazione pxelinux e risiedono nella directory `/tftpboot/slc304-i386`.

3.1.6 Preparazione dei file pxelinux

Al boot il firmware PXE viene scaricato per mezzo del TFTP, questo viene eseguito e cerca di scaricare il file di configurazione. Tale file contiene:

- Il nome del kernel.
- Il nome del file initrd.
- L'interfaccia di rete da utilizzare per l'installazione e la sua configurazione.
- L'URL del file kickstart da utilizzare.

3.1.7 Preparazione dei file kickstart

I file *kickstart* contengono le informazioni riguardo i pacchetti da installare, le partizioni da preparare nel disco, i settaggi dell'hardware etc...
La preparazione dei file *kickstart* può essere effettuata manualmente, prendendo un file

[3]APT è un software di gestione per l'installazione di pacchetti software.

[4]YUM è un programma per l'aggiornamento automatico di pacchetti RPM (RedHat Packet Manager), YUM controlla le dipendenze dei pacchetti e tutto il necessario per installare correttamente programmi da vari repository.

kickstart di un'installazione precedente o generato mediante il tool grafico di RedHat "Kickstart Configurator".

3.1.8 Installazione dei nodi

Prima di partire con l'installazione del SO è necessario controllare che:

- Nel BIOS[5] sia attivo il boot via rete.
- Il nodo sia correttamente registrato nel server DNS (Domanin Name Server).
- Il MAC-address sia contenuto nella configurazione del server DHCP.
- Controllare che la configurazione del file *pxelinux* sia corretta.
- Controllare che la configurazione del file *kickstart* sia corretta.

Il passo successivo è eseguire lo script:

`/usr/sbin/ig-bootselect`

fornito dal pacchetto `ig-installserver`, inserire il FQDN[6] ed associarlo alla configurazione pxelinux da usare. In questo modo viene associato il file di configurazione pxelinux all'indirizzo IP del nodo.
Se tutti i passi sono stati eseguiti in modo corretto all'avvio successivo del nodo questo dovrebbe contattare il server DHCP, ottenere la configurazione dell'interfaccia di rete e iniziare l'installazione del SO.

3.2 Installazione del middleware

Il software di INFN-Grid viene installato nella directory `/opt` del sistema. Abbiamo al suo interno la directory `/opt/lcg/yaim` che è quella che contiene il software di installazione e configurazione:

- In `/opt/lcg/bin` ci sono gli script di installazione e configurazione.
- In `/opt/lcg/yaim/script` ci sono gli script che vengono richiamati dagli script di installazione e configurazione.

[5]Il Basic Input-Output System o BIOS è il primo codice che viene eseguito da un Personal Computer dopo l'accensione, ed ha la funzione principale di localizzare e caricare il sistema operativo nella RAM. Il BIOS garantisce la comunicazione a basso livello tra l'hardware e le periferiche e fornisce il supporto per le chiamate di sistema verso la gestione della tastiera e alcune primitive per la gestione dell'output sul video, seppur limitate al modo testuale. Il BIOS è scritto di solito nel linguaggio assembler nativo della famiglia di CPU utilizzata.

[6]Fully Qualified Domain Name, nome dell'host seguito dal dominio.

- In `/opt/lcg/yaim/exaple` ci sono i file di configurazione di esempio.
- In `/opt/lcg/yaim/function` e `/opt/lcg/yaim/function/local` ci sono le funzioni di configurazione richiamate a seconda del tipo di nodo da installare.

3.2.1 Preparazione all'installazione

Prima di effettuare l'installazione del middleware è necessario pianificare la fase di installazione con il Central Management Team (CMT) e in caso di upgrade da una versione di Grid precedente, è necessario inserire un avviso di *downtime* tramite il sistema di *ticketing*; il periodo di *downtime* deve essere sufficientemente lungo da poter permettere la completa migrazione e la certificazione del sito.

3.2.2 Aggiornamento del repository tramite YAM

Se si usa un repository locale è necessario modificare la configurazione di YAM così da poter effettuare il mirror dei pacchetti necessari all'installazione di INFN-Grid oltre a quelli del SO.
Dopo aver modificato la configurazione è necessario rilanciare il comando:

```
yam -guxvv
```

così da poter scaricare i pacchetti mancanti e rigenerare i metadata contenenti le dipendenze fra i pacchetti.

3.2.3 Installazione di Java

La licenza di SUN relativa a Java non permette che questo venga distribuito assieme al middleware, è perciò necessario scaricarlo dal sito web della SUN.
Se si usa un repository locale per l'installazione dei nodi è necessario salvare il pacchetto di Java nella directory:

```
/var/rep/slc304-i386/localrpms/
```

del repository e poi eseguire i seguenti comandi:

```
cd /var/rep/slc304-i386/localrpms/
sh j2sdk-1_4_2_08-linux-i586-rpm.bin
mv j2sdk-1_4_2_08-linux-i586.rpm j2sdk-1.4.2_08-fcs.i586.rpm
rm j2sdk-1_4_2_08-linux-i586-rpm.bin
```

e quindi aggiungere il pacchetto al repository locale con il comando:

```
yam -gvv -d slc304-i386
```

In caso non venga usato un repository locale è necessario installare manualmente il pacchetto Java su SE, CE, UI e su tutti i WN con il comando:

```
sh j2sdk-1_4_2_08-linux-i586-rpm.bin
```

Il pacchetto Java è necessario per il funzionamento di RGMA, il tool di monitoraggio usato da INFN-Grid.

3.2.4 Controllo del FQDN

Nei nodi è necessario controllare il FQDN tramite il comando:

```
hostname -f
```

questo deve restituire il nome completo dell'host (es. gridce.pg.infn.it).

3.2.5 Installazione dei pacchetti lcg-yaim e ig-yaim

Occorre scaricare l'ultima versione dei pacchetti `lcg-yaim-2.4.0` e `ig-yaim-2.4.0` che forniscono una serie di mini-script e di file di configurazione necessari all'installazione e alla configurazione del middleware.

3.2.6 Configurazione lcg-yaim e ig-yaim

Prima di passare alla configurazione del tool di installazione è necessario rimpiazzare il file `node-info.def`, fornito da LCG, con quello fornito da INFN-Grid. Per fare ciò dobbiamo eseguire i seguenti comandi:

```
cd /opt/lcg/yaim/scripts/
mv node-info.def node-info.def_lcg
mv ig-node-info.def node-info.def
```

É necessario poi modificare il file contenente la lista dei Worker Node:

```
/opt/lcg/yaim/examples/wn-list.conf
```

inserendovi la lista dei WN del sito.
L'ultimo file da modificare è `ig-site-info.def` personalizzandolo secondo la propria configurazione, tale file contiene tutti i parametri necessari per l'installazione e per la configurazione di ogni singolo nodo, come l'indirizzo del repository dal quale prelevare i pacchetti, gli hostname dei nodi da installare, la locazione di Java, le informazioni relative alle VO supportate e alle relative code nel batch-system etc...

3.2.7 Installazione dei certificati X.509

I nodi CE ed SE per essere identificati in Grid devono avere installati i certificati rilasciati dalla Certification Autority.
I due file, `hostcert.pem` e `hostkey.pem` vanno copiati dentro la directory

`/etc/grid-security/`

e gli vanno dati i permessi appropriati tramite i comandi:

```
chmod 644 /etc/grid-security/hostcert.pem
chmod 400 /etc/grid-security/hostkey.pem
```

3.2.8 Installazione del middleware

L'installazione del middleware avviene tramite il comando:

`/opt/lcg/bin/install_node <ig-site-info.def> <metapackage>`

La prima operazione della funzione `install_node` è la configurazione di APT usando la configurazione stabilita nel file `ig-site-info.def`, dopodichè installa il *metapackage* tramite APT. Ogni elemeto di INFN-Grid corrisponde ad un *metapackage*, nella tabella 3.2.8 sono elencati i vari *metapackage* e la loro descrizione.

Metapackage	Descrizione
ig_CE_torque	Computing Element con Torque+MAUI.
ig_CE_LSF	Computing Element con LSF.
ig_LCG_BDII	LCG BDII.
ig_MON	Sistema di monitoraggio basato su RGMA e GridICE.
ig_NM	Network Monitor.
ig_RB	Resource Broker.
ig_RB_DAG	Resource Broker con supporto DAG.
ig_SECLASSIC	Storage Element.
ig_UI	User Interface.
ig_UI_DAG	User Interface con supporto DAG.
ig_WN_torque	Worker Node con Torque.
ig_WN_LSF	Worker Node con LSF.
ig_WN	Worker Node.
ig_CE	Computing Element senza batch system.

Tabella 3.1: Elenco dei metapackage disponibili e loro descrizione

3.2.9 Configurazione del middleware

La configurazione del middleware avviene tramite il comando:

```
/opt/lcg/bin/configure_node <ig-site-info.def> <metapackage>
```

In caso sia necessario lanciare una funzione in modo indipendente si esegue il comando:

```
/opt/lcg/yaim/scripts/run_function <ig-site-info.def>
<function_name>
```

Le funzioni che è possibile specificare come argomento <`function_name`> sono i file nelle directory:

```
/opt/lcg/yaim/functions/
/opt/lcg/yaim/functions/local
```

3.2.10 Certificazione

Il CMT è l'organismo responsabile per la certificazione di un sito Grid, deve quindi occuparsi di controllare il corretto funzionamento di un sito prima che questo entri in produzione. I test che vengono effettuati per certificare un sito sono:

- Funzionalità MPI (se abilitate).
- Consistenza delle informazioni del GIIS.
- Sottomissione di job locali.
- Sottomissione dei job via Globus.
- Sottomissione dei job via Resource-Broker.
- Corretto funzionamento del Replica Manager.

Il CMT per effettuare questi test si avvale delle funzionalità del RB e del BDII, e al termine dei test viene comunicato al site manager l'esito della certificazione ed eventuali problemi e suggerimenti per la risoluzione attraverso un sistema di ticketing.

Capitolo 4

Modifiche per installazione di INFN-Grid a Perugia

4.1 Quali modifiche e perchè

La configurazione originale di un sito INFN-Grid 2.4.0, prevede che tutte le macchine debbano avere un indirizzo IP pubblico. La sezione INFN di Perugia non poteva accettare una situazione simile, dato che non era in grado di assegnare molti indirizzi IP a tale progetto. Si è pensato quindi di modificare il middleware, rendendo possibile l'installazione di alcune di queste macchine all'interno di una rete nascosta o privata; le uniche che non necessitano di essere raggiungibili direttamente dall'esterno sono i WN, saranno quindi queste le macchine che saranno connesse al sito soltanto tramite rete privata. Un'altra richiesta dei gruppi di ricerca che contribuiscono al progetto, era quella di poter usare il sito INFN-Grid anche come una farm locale, e poter quindi sottomettere job senza doversi autenticare in Grid; è stato quindi necessario modificare la configurazione del batch-system e dello scheduler, per permettere che questi accettassero job provenienti, non solo da Grid, ma anche direttamente dalle UI. La struttura logica e hardware implementata è rappresentata in figura 4.1.

4.2 Modifiche per l'installazione del SO su rete privata

4.2.1 Installazione di un server DNS interno

Normalmente, gli elementi di un sito INFN-Grid risolvono gli hostname in indirizzi IP e viceversa tramite un server DNS esterno al sito. Nella configurazione che vogliamo

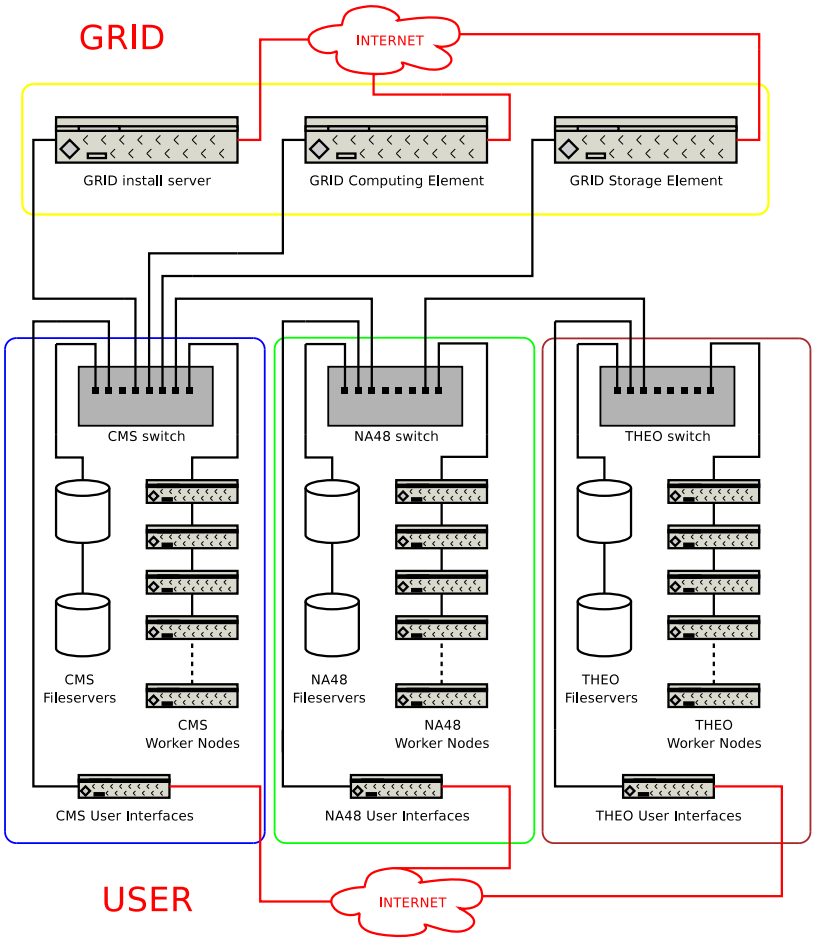

Figura 4.1: Struttura obiettivo del sito INFN-Grid di Perugia

raggiungere i WN sono interconnessi con le altre macchine, soltanto tramite una rete privata, si è quindi reso necessario installare un server DNS interno alla rete locale, che permettesse almeno la risoluzione degli hostname del dominio di rete interno; questo sarà anche il server DNS per le macchine che hanno due interfaccie di rete. Abbiamo scelto di installatre il server DNS *bind*, fornito dal pacchetto `named`, nell'Install Server. La configurazione del server DNS è composta da tre parti:

- Configurazione generica del server DNS (`/etc/named.conf`).
- Configurazione per la risoluzione di indirizzi IP in hostname.
- Configurazione per la risoluzione di hostname in indirizzi IP.

La configurazione generica del server DNS che abbiamo usato è la seguente:

```
options {
        directory "/var/named";
        dump-file "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
};
controls {
        inet 127.0.0.1 allow { localhost; } keys { rndckey; };
};
zone "." IN {
        type hint;
        file "named.ca";
};

zone "localhost" IN {
        type master;
        file "localhost.zone";
        allow-update { none; };
};

zone "0.0.127.in-addr.arpa" IN {
        type master;
        file "named.local";
        allow-update { none; };
};

zone "grid.pg.infn.it" IN {
        type master;
        file "grid.zone";
        allow-update { none; };
};

zone "254.168.192.in-addr.arpa" IN {
        type master;
        file "grid.rev";
        allow-update { none; };
};

include "/etc/rndc.key";
```

all'interno del file sono specificati i file `grid.zone` e `grid.rev` rispettivamente per la risoluzione di indirizzi IP in hostname e viceversa.
Riportiamo un esempio di grid.zone:

```
$TTL 3600
@       IN      SOA      gridrep.grid.pg.infn.it. grid-prod.pg.infn.it. (
                2005051401 ; Serial ; Increment by one after every change
                3600       ; Refresh every hour
                900        ; Retry every 15 minutes
                3600000    ; Expire 1000 hours
                3600 )     ; Minimum 1 hour

grid.pg.infn.it.   IN      NS      gridrep.grid.pg.infn.it.

gridce          IN      A       192.168.254.3
gridse          IN      A       192.168.254.4
gridrep         IN      A       192.168.254.5
cmsgridui       IN      A       192.168.254.7
node10          IN      A       192.168.254.10
```

e di grid.rev:

```
$TTL 3600
@       IN      SOA     gridrep.grid.pg.infn.it. grid-prod.pg.infn.it. (
                2005051401 ; Serial ; Increment by one after every change
                3600       ; Refresh every hour
                900        ; Retry every 15 minutes
                3600000    ; Expire 1000 hours
                3600 )     ; Minimum 1 hour

        IN      NS      gridrep.grid.pg.infn.it.

3       IN      PTR     gridce.grid.pg.infn.it.
4       IN      PTR     gridse.grid.pg.infn.it.
5       IN      PTR     gridrep.grid.pg.infn.it.
7       IN      PTR     cmsgridui.grid.pg.infn.it.
10      IN      PTR     node10.grid.pg.infn.it.
```

4.2.2 Modifiche ai file di kickstart

Nel file *kickstart* sono presenti le informazioni necessarie all'installazione del sistema, alla configurazione hardware etc... Fra queste informazioni, ci sono i parametri per la configurazione delle interfacce di rete. Nella configurazione standard è prevista la configurazione per una sola interfaccia di rete, con assegnazione dell'indirizzo IP tramite DHCP; il nostro sito avrà alcune macchine con due interfacce di rete ed alcune con una sola interfaccia, si è reso quindi necessario modificare il file, per impostare

entrambe le interfacce. Abbiamo deciso poi di impostare gli indirizzi IP in modo statico e non tramite DHCP per eliminare così un possibile "point of failure".
La parte relativa alla configurazione di rete per le macchine con due interfacce (CE, SE, UI), diventa la seguente:

```
network --device eth0 --bootproto static --ip <internal_ip> --netmask
  255.255.255.0
network --device eth1 --bootproto static --ip <external_ip> --netmask
  255.255.255.0  --gateway <gateway_ip> --nameserver <nameserver_ip>
  --hostname <external_hostname>
```

per le macchine con una sola interfaccia di rete risulta invece:

```
network --device eth0 --bootproto static --ip <internal_ip>  --netmask
  255.255.255.0  --gateway <gateway_ip>  --nameserver <nameserver_ip>
```

Abbiamo deciso di utilizzare per i WN come *gateway*[1] di default il CE; il *gateway* è necessario ai WN affinché possano comunicare con l'esterno.

4.2.3 Modifiche ad ig-bootselect

Lo script ig-bootselect, quello che associa il file kickstart al nome dell'host, creava confusione fra nomi interni ed esterni. Abbiamo deciso di installare i nodi tramite l'interfaccia di rete interna, fornendo così allo script il nome dell'host seguito dal dominio interno (es. gridce.grid.pg.infn.it). In fase di caricamento dei collegamenti fra gli indirizzi IP e i file di configurazione *pxelinux* già presenti, lo script convertiva l'indirizzo IP in hostname troncando il dominio. Il nome risultante corrispondeva al nome associato all'interfaccia esterna, creando così confusione. La parte della funzione *GetHostname* dello script, è stata quindi modificata da:

```
$res[0]=(split('\.', gethostbyaddr(inet_aton($res[1]), AF_INET)))[0];
```

in:

```
$res[0]=(gethostbyaddr(inet_aton($res[1]), AF_INET))[0];
```

per permetterne il corretto funzionamento.

4.2.4 Post installazione

Al termine dell'installazione, dopo aver controllato che la configurazione di rete fosse corretta e funzionante, abbiamo abilitato il CE a processare i pacchetti ricevuti dai WN destinati alla rete pubblica, per far si che questi potessero comunicare con l'esterno. Per fare ciò abbiamo utilizzato lo strumento *Netfilter*. Con il comando:

[1] Il gateway è un dispositivo di rete che opera al livello di rete e superiori del modello ISO/OSI. Il suo scopo principale è quello di veicolare i pacchetti all'esterno della rete locale (subnet). Da notare che gateway è un termine generico che indica il servizio di inoltro dei pacchetti verso l'esterno; il dispositivo hardware che porterà a termine questo compito è tipicamente un router.

```
echo "1" /proc/sys/net/ipv4/ip_forward
```

abbiamo abilitato l'"IP forwarding", e con il comando:

```
iptables -t nat -A POSTROUTING -s <INT_NETWORK> -d ! <INT_NETWORK>
-j SNAT --to <IP_CE>
```

abbiamo aggiunto una regola alla catena POSTROUTING della tabella NAT, che si occupa di modificare i pacchetti destinati verso l'esterno. La regola aggiunta dice al CE di modificare l'indirizzo IP del sorgente (-j SNAT) per i pacchetti che provengono dalla rete interna (-s <INT_NETWORK>) e che non sono destinati alla rete interna (-d ! <INT_NETWORK>). In questo modo, la macchina di destinazione ha un indirizzo IP valido a cui rispondere, e non un indirizzo di rete interna irraggiungibile. La modifica sarà poi resa permanente tramite un'apposita funzione spiegata in seguito. Abbiamo poi scaricato ed installato su ciascuna macchina i pacchetti relativi al middleware.

4.3 Modifiche necessarie alla configurazione del middleware

Le funzioni che provvedono a configurare il middleware sono contenute nelle directory `/opt/lcg/yaim/function/` e `/opt/lcg/yaim/function/local`; nel file `ig-node-info.def` è invece contenuta la lista delle funzioni associate ad ogni singolo metapackage. Elenchiamo le funzioni relative ad ogni singolo metapackage che vogliamo installare:

CE: il metapackage relativo al tipo di CE che vogliamo installare, quello con *torque* come batch-system, è ig_CE_torque. Le funzioni che esso richiama in fase di configurazione sono:

```
config_ntp
config_ldconf
config_sysconfig_edg
config_sysconfig_globus
config_sysconfig_lcg
config_crl
config_host_certs
config_users
config_edgusers
config_mkgridmap
config_gip
config_globus
config_nfs_sw_dir_server
```

```
config_fmon_client
config_lcgenv
config_java
config_rgma_client
config_bdii
config_workload_manager_env
config_wm_locallogger
config_apel_pbs
config_lcmaps
config_lcas
config_dgas_ce
config_torque_server
```

SE: la nostra configurazione richiede un solo SE e dobbiamo quindi installare il metapackage ig_SECLASSIC che richiama le seguenti funzioni:

```
config_ntp
config_ldconf
config_sysconfig_edg
config_sysconfig_globus
config_sysconfig_lcg
config_crl
config_host_certs
config_users
config_edgusers
config_mkgridmap
config_gip
config_globus
config_nfs_sw_dir_server
config_seclassic
config_java
config_rgma_client
config_rfio
config_fmon_client
config_lcmaps
config_lcas
```

e il metapackage ig_MON che richiama le seguenti funzioni:

```
config_ldconf
config_ntp
config_sysconfig_edg
config_edgusers
```

```
config_java
config_rgma_client
config_rgma_server
config_fmon_client
config_apel_rgma
```

WN: come per il CE, vogliamo installare i WN con il supporto per il batch-system *torque*, abbiamo scelto quindi il metapackage ig_WN_torque che richiama le seguenti funzioni:

```
config_ntp
config_ldconf
config_sysconfig_edg
config_sysconfig_globus
config_sysconfig_lcg
config_crl
config_nfs_sw_dir_client
config_globus
config_lcgenv
config_replica_manager
config_users
config_sw_dir
config_java
config_rgma_client
config_fmon_client
config_workload_manager_env
config_afs
config_torque_client
```

UI: per l'installazione delle UI è necessario installare il metapackage ig_UI, le funzioni che questo richiama in fase di configurazione sono:

```
config_ntp
config_ldconf
config_sysconfig_edg
config_sysconfig_globus
config_sysconfig_lcg
config_crl
config_globus
config_lcgenv
config_replica_manager
config_java
config_rgma_client
config_workload_manager_client
```

Le funzioni che è stato necessario modificare per permettere il funzionamento dei WN su rete nascosta sono le seguenti: `config_nfs_sw_dir_client`, `config_torque_client`, `config_bdii`.
Analizziamo nel dettaglio il funzionamento e lo scopo di ogni singola funzione:

- `config_nfs_sw_dir_client`: questa funzione, richiamata dai WN, si occupa di definire il modo in cui il software di ogni singolo esperimento viene montato dalle macchine che ne fanno uso. Nella nostra configurazione è il CE ad esportare il software dell'esperimento e lo fa tramite il protocollo NFS[2], questo avviene tramite la funzione `config_nfs_sw_dir_server`. I client, in una configurazione standard, montano il software tramite l'unica interfaccia di rete presente. Nella nostra configurazione, richiamando la funzione standard, i WN monterebbero il software tramite l'interfaccia di rete esterna. Per far si che i WN montino il software dell'esperimento facendo riferimento all'interfaccia di rete interna del CE occorre modificare la funziona di configurazione. Per prima cosa è stato necessario introdurre una variabile, che ci servirà anche nelle altre funzioni di configurazione, che stabilisse se il tipo di installazione che volgiamo effettuare, è con WN su rete privata o pubblica. Abbiamo chiamato questa variabile `PRIVATE_NETWORK` e l'abbiamo aggiunta alle altre variabili contenute nel file `ig-site-info.def`. La funzione che dobbiamo modificare, controlla che tutte le variabili necessarie alla configurazione siano impostate, poi provvede a modificare il file `/etc/fstab`. Tale file descrive quali sono i *device*, locali e remoti, che devono essere montati nel File System locale e in che modo. La funzione aggiunge la linea relativa al software degli esperimenti, indicando NFS come protocollo usato. Tale linea nella configurazione standard risulta:

  ```
  echo "$HOST_SW_DIR:$BASE_SW_DIR $VO_SW_DIR nfs rw,defaults 0 0"
   >> /etc/fstab.tmp
  ```

  Abbiamo poi bisogno di una nuova variabile che ci dica qual'è il nome interno del server che monta il software degli esperimenti; chiameremo la variabile `INT_HOST_SW_DIR`. La funzione esegue un controllo sul tipo in installazione e se è su rete privata allora controlla che tutte le variabili necessarie siano impostate, ed esegue il seguente comando anziché quello precedentemente descritto:

  ```
  echo "$INT_HOST_SW_DIR:$BASE_SW_DIR $VO_SW_DIR nfs rw,defaults 0 0"
   >> /etc/fstab.tmp
  ```

- `config_torque_client`: questa funzione, richiamata dai WN, si occupa di configurare l'*execution daemon* del batch-system *torque* e di configurare SSH[3] affinché

[2]Network File System (NFS) è un protocollo sviluppato inizialmente da Sun Microsystems nel 1984 e definito dagli RFC 1094, 1813, (3010) e 3530. L'NFS è un File System che consente ai computer di utilizzare la rete per accedere ai dischi remoti come fossero dischi locali.

[3]SSH (Secure SHell, shell sicura) è un protocollo che permette di stabilire una sessione remota

i WN possano comunicare con il CE e lo SE.
La configurazione dell'*execution daemon*, che normalmente risiede nel file

`/var/spool/pbs/mom\_priv/config`

è di default composta da:

- `$clienthost`: definisce il nome degli host che hanno il permesso di contattare l'*execution daemon* per sottomettere job, ricevere informazioni sullo stato del nodo etc... tramite una porta privilegiata. La variabile può essere ripetuta più volte per permettere a più di un host di contattare l'*execution daemon*.
- `$restricted`: definisce quali host hanno accesso all'*execution daemon* senza necessità di utilizzare una porta privilegiata.
- `$logevent`: definisce la maschera in base alla quale viene deciso quali eventi registrare sul file di log.
- `$ideal_load`: definisce il carico ideale per il nodo.
- `$max_load`: definisce il carico massimo per il nodo.

La funzione di configurazione imposta `$clienthost` e `$restricted` con il valore `CE_HOST`, impostato al nome esterno del CE, contenuto nel file di configurazione. Il CE contatta i WN tramite l'interfaccia di rete interna, quindi gli *execution daemon* ricevono un nome di host non contenuto nelle variabili `$clienthost` e `$restricted`, e rifiutano la connessione. Per ovviare a questo problema abbiamo deciso di introdurre una nuova variabile, chiamata `CE_INT_HOST`, che contenesse il nome interno del CE; la funzione quindi, in caso si sia scelta l'installazione su rete nascosta, imposta nella configurazione dell'*execution daemon* il valore di `CE_INT_HOST` alla variabile `$clienthost`, così da permettere al CE di poter comunicare con l'*execution daemon*.

- `config_bdii`: richiamata dal CE, si occupa di scrivere la configurazione degli script `lcg-bdii` e `lcg-bdii-update`, di aggiungere l'esecuzione del primo all'avvio e di aggiungere il secondo in *cron*[4]; tali script permettono al server BDII di raccogliere le informazioni relative al sito contattando il CE.
  Le parti degli script che ci interessano sono quelle che usano `iptables`, quindi per il primo:

cifrata ad interfaccia a linea di comando con un altro host. Analogamente ad altri protocolli come telnet ed rlogin, SSH ha una interfaccia a linea di comando, tipicamente una shell, ma l'intera comunicazione avviene in maniera cifrata. Per questo motivo, SSH è diventato uno standard di fatto per l'amministrazione remota di sistemi unix e di dispositivi di rete.

[4]Cron è un demone che permette l'esecuzione programmata a determinati orari o intervalli di tempo di un comando.

```
iptables -t nat -I PREROUTING 1 -p tcp --dport ${BDII_PORT_READ}
   -j REDIRECT --to-ports ${BDII_PORT_READ}
```

che provvede a creare una regola sulla catena PREROUTING della tabella NAT. Tale regola redirige i pacchetti destinati alla porta `BDII_PORT_READ` nella porta `BDII_PORT_READ`; questa non ha alcun effetto pratico ma è inserita per assicurare il corretto funzionamento dello script `lcg-bdii-update`. E per il secondo:

```
system("iptables -t nat -R PREROUTING 1 -p tcp --dport
   $bdii_port_read -j REDIRECT --to-ports $bdii_port_write");
```

che rimpiazza la regola precedente con una nuova, dove cambia ciclicamente la porta di destinazione.
Tale regola, visto che abbiamo impostato il CE come gateway di default, modifica anche la porta di destinazione delle richieste fatte dai WN verso l'esterno della rete o verso il CE, cambiandogli porta di destinazione. Per ovviare a questo problema abbiamo introdotto una variabile, nel file di configurazione, che indica la rete interna utilizzata (`INT_NET`) e verrà usata per generare le regole di `iptables`. La funzione di configurazione in caso si sia scelto di installare i WN su rete privata, aggiunge al file di configurazione usato da `lcg-bdii` e `lcg-bdii-update` due variabili, una per stabilire se si sta installando il sito con WN su rete privata, e una che contiene il valore di `INT_NET`. Nel primo script viene controllato se la variabile `PRIVATE_NETWORK` è impostata a "true" e in caso affermativo viene eseguito:

```
iptables -t nat -I PREROUTING 1 -p tcp -s ! $INT_NET -d
   `hostname -f` --dport ${BDII_PORT_READ} -j REDIRECT
   --to-ports ${BDII_PORT_READ}
```

al posto del comando riportato in precedenza. Nel secondo script, visto che è scritto in perl, abbiamo introdotto la dichiarazione di due nuove variabili, che vengono poi lette dal file di configurazione:

```
my $private_network;
my $int_net;
```

e abbiamo aggiunto il controllo sull'installazione di WN su rete privata. In tal caso, anziché la riga riportata in precedenza, viene eseguita:

```
system("iptables -t nat -R PREROUTING 1 -p tcp -s ! $int_net
   -d `hostname -f` --dport $bdii_port_read -j REDIRECT
   --to-ports $bdii_port_write");
```

Le nuove regole di `iptables` hanno lo stesso effetto delle precedenti, con la differenza che viene modificata la porta solo per le connessioni non provenienti dalla rete interna e che hanno come destinazione il CE, cosicché i WN possano contattare server BDII esterni.

É stato poi necessario scrivere una nuova funzione che si occupasse di abilitare in modo permanente il CE come gateway ed il funzionamento di AFS[5] sui WN.
La funzione, quando viene chiamata, controlla se l'host sul quale viene eseguita è il CE e se è abilitata l'installazione con WN su rete nascosta; in caso affermativo permette ai WN di comunicare con l'esterno. Per prima cosa viene ripulita la tabella NAT di `iptables`, viene aggiunta una nuova regola che abilita lo SNAT, e viene abilitato l'"IP forwarding"; tali impostazioni vengono poi rese definitive. Lo SNAT viene inserito in uno script di configurazione richiamato all'avvio della macchina e l'"IP forwarding" viene reso sempre attivo scrivendo un "1" nel file `/etc/sysctl.conf`. I nodi che usano AFS, oltre a poter comunicare con l'esterno, devono essere anche direttamente raggiungibili dall'esterno per poter rispondere alle chiamate di "callback" di AFS. Per fare ciò, ciascun WN deve avere una propria porta, sul CE, che venga reindirizzata sulla porta del servizio AFS di ogni singolo WN. Abbiamo deciso di assegnare, per ciascun WN, la porta calcolata su un offset (4100), più l'ultima parte dlel'indirizzo IP del nodo. Per fare un esempio, un nodo con indirizzo IP 192.168.0.13 avrà come porta nel CE la 4113. La lista dei nodi viene prelevata dal file di configurazione necessario in fase di installazione e definito dalla variabile WN_LIST, per ogni nodo viene poi risolto il suo indirizzo IP tramite il comando `host` e presa l'ultima parte dell'indirizzo IP. Vengono poi generate due regole per ciascun nodo, tali regole parametrizzate sono:

```
$IPTABLES -t nat -A PREROUTING -p udp --dport $PORT -j DNAT
--to-destination $NODE_IP:$AFSPORT
$IPTABLES -t nat -A POSTROUTING -p udp -s $NODE_IP --sport
$AFSPORT -j SNAT --to-source $CE_IP:$PORT
```

dove PORT è la porta ricalcolata per ogni singolo nodo,NODE_IP è l'indirizzo IP del nodo, AFSPORT è la porta standard del servizio AFS e CE_IP è l'indirizzo IP del CE. La prima regola, fa si che le richieste ricevute dal CE destinate alla porta PORT con protocollo udp, quello usato da AFS, vengano inoltrate al nodo corrispondente (NODE_IP) sulla porta standard di AFS (AFSPORT). La seconda regola invece, dice al CE che le richieste provenienti dal nodo NODE_IP e aventi come porta sorgente la porta AFSPORT, vengano traslate ed escano dalla rete come provenienti dal CE e dalla porta PORT.
Viene poi generato uno script che, eseguito in *cron*, preleva la lista dei WN del sito

[5]Andrew file system (AFS) è un File System distribuito sviluppato dall'università di Carnegie Mellon. Prende il nome da Andrew Carnegie e Andrew Mellon. Trova il suo uso principale nel calcolo distribuito.

con il comando `pbsnodes` e in base al risultato, se non è già presente una regola per tale nodo, la aggiunge in modo analogo al precedente. Si è scelto di prelevare la lista dei WN con il comando `pbsnodes`, in quanto i WN possono essere aggiunti anche in seguito, e quindi non sono presenti nella lista usata per la prima configurazione.

4.4 Sottomissione delle modifiche per WN su rete privata al gruppo INFN-Grid di sviluppo

Le modifiche apportate agli strumenti di installazione e configurazione sono state poi comunicate, con la relativa documentazione aggiornata, al CNAF[6] che è il centro italiano che si occupa della distribuzione di INFN-Grid. Ci è stato poi chiesto di testare il nuovo sistema per la gestione del BDII, che è stato introdotto nella versione 2.6.0 sostituendo gli script lcg-bdii e lcg-bdii-update, per verificare se ci fossero problemi per l'utilizzo di tale strumento su una configurazione con WN su rete nascosta. I test non hanno riscontrato problemi e le nostre modifiche sono state integrate in INFN-Grid 2.6.0.

4.5 Modifiche per la sottomissione locale di job

Per rendere possibile l'utilizzo, da parte degli utenti Grid locali, delle macchine del sito come una farm, abbiamo deciso di utilizzare il batch-system *torque*, che viene installato insieme a INFN-Grid.

4.5.1 Installazione e configurazione di torque client

Gli utenti, per poter utilizzare il batch-system *torque*, necessitano degli strumenti forniti dal pacchetto `torque-client`. Tale pacchetto fornisce tutti gli strumenti per la sottomissione, cancellazione e controllo dello stato dei job, oltre ad alcuni strumenti per il controllo dello stato dei nodi del batch-system. La configurazione di `torque-client` consiste nell'indicare il nome del server *torque* nel file di configurazione `/var/spool/pbs/server_name`.

4.5.2 Sincronizzazione degli utenti

Visto che *torque* esegue i job mandati in esecuzione nel nodo da parte di un utente mappandoli con lo stesso username e UID locale, abbiamo dovuto per prima cosa creare su tutte le macchine del sito, escluso l'Install Server, un account per ciascun

[6]IL CNAF e' un Centro Nazionale dell'INFN il cui scopo e' la ricerca e lo sviluppo nel campo delle discipline informatiche applicate agli esperimenti di fisica nucleare e delle alte energie.

utente avente stesso username e stesso UID; la configurazione standard prevede che gli utenti Grid abbiano il proprio account solo sulle UI. Per problemi di sicurezza abbiamo comunque limitato l'accesso da parte degli utenti alle sole UI dell'esperimento di cui fanno parte.

4.5.3 Configurazione di SSH

L'*execution daemon* eseguito sui WN, dopo aver processato il job, ne restituisce l'output sulla macchina dalla quale è stata fatta la richiesta di esecuzione, nel nostro caso quindi sulle UI. Normalmente l'output viene trasmesso tramite il protocollo SSH, le UI quindi debbono consentire ai WN l'accesso tramite SSH senza necessità di autenticarsi con la password. Per fare ciò abbiamo utilizzato il metodo che è anche usato dal CE per ricevere l'output dei job inviati tramite Grid: l'autenticazione in base all'host. Abbiamo così installato nelle UI il pacchetto `edg-pbs-knownhost`, modificando poi la sua configurazione in tutte le macchine del sito, aggiungendo i nomi delle UI alla lista dei nodi conosciuti.
La configurazione del server SSH delle UI è stata modificata aggiungendo le seguenti linee:

```
HostbasedAuthentication yes
IgnoreUserKnownHosts yes
IgnoreRhosts yes
```

HostbasedAuthentication abilita l'autenticazione in base al nome dell'host;
IgnoreUserKnownHosts specifica che il server SSH ingori nelle autenticazioni di tipo HostbasedAuthentication il file `$HOME/.ssh/known_hosts`;
IgnoreRhosts specifica che il server SSH ingori i file `.rhosts` e `.shosts` nelle autenticazioni di tipo HostbasedAuthentication.
É poi stato generato il file `/etc/ssh/shosts.equiv` nelle UI contenente la lista di tutti i WN, tale file è usato per specificare quali sono gli host autenticati a eseguire il login tramite SSH senza password.
Appena lo script `edg-pbs-knownhost` che è eseguito in *cron*, viene lanciato, i nodi si scambiano le chiavi SSH e i WN possono autenticarsi sulle UI in base al proprio hostname.

4.5.4 Configurazione del server del batch-system torque

Ogni esperimento ha esigenze particolari per quanto riguarda il numero di code e la relativa politica di gestione. Si è quindi creato un gruppo di code per ciascun esperimento, dove è possibile lanciare job soltanto dagli utenti appartenenti al gruppo dell'esperimento corrispondente, e da una delle UI del proprio gruppo. Un esempio di configurazione di una coda del batch-system è il seguente:

```
create queue cms-long
set queue cms-long queue_type = Execution
set queue cms-long acl_host_enable = True
set queue cms-long acl_hosts = cmsgridui.pg.infn.it
set queue cms-long acl_hosts += cmsgridui2.pg.infn.it
set queue cms-long resources_max.cput = 12:00:00
set queue cms-long resources_max.walltime = 24:00:00
set queue cms-long acl_group_enable = True
set queue cms-long acl_groups = cms
set queue cms-long enabled = True
set queue cms-long started = True
```

in questo caso, la coda appartenente al gruppo di ricerca CMS, ha come host abilitati soltanto cmsgridui.pg.infn.it e cmsgridui2.pg.infn.it, che sono le due UI del gruppo, ed ha una limitazione al gruppo CMS per la sottomissione dei job. Le altre opzioni definiscono il tempo massimo di esecuzione di un job, il tipo di coda creata etc...

4.5.5 Configurazione dello sheduler

Lo sheduler, nel nostro caso `maui`, è il processo che si occupa di decidere la destinazione dei job, il numero di quelli in esecuzione in contemporanea etc... In una configurazione standard il suo ruolo non è rilevante, visto che tutti i nodi sono riservati ai job provenienti da Grid, e che l'operazione di gestione dei job è affidata principalmente al Resource Broker. Nella nostra configurazione è necessario stabilire una politica, per la sottomissione di job, in base all'harware che ogni singolo esperimento mette a disposizione. Abbiamo deciso di assegnare una diversa priorità per ciascuna coda, sia locale che remota, e un numero massimo di job, sia per utente, gruppo e coda. La priorità più alta è stata assegnata alle code cert e test, usate rispettivamente per i job di certificazione via Grid, e per i test locali fatti dagli amministratori del sito. Le altre code hanno priorità inversamente proporzionale al limite massimo di durata del job. I diversi gruppi hanno tutti la stessa priorità a parità di coda, la limitazione sta invece nel numero massimo di job che possono eseguire contemporaneamente, questa avviene in base al numero di processori che hanno messo a disposizione del sito Grid. In una situazione di pieno carico ogni gruppo può eseguire un numero di job pari al numero di processori dei propri nodi, a farm scarica, invece, può eseguire contemporaneamente un numero di job pari al numero dei propri processori più la metà dei restanti.

Capitolo 5

Monitoraggio

Per controllare lo stato della Grid e dei singoli siti, INFN-Grid mette a disposizione due strumenti, *GridIce*[1] e *GridAT* (ancora in fase di testing). Questi strumenti però, non permettono di monitorare nel dettaglio ogni singola macchina, si è scelto quindi di installare nel sito di Perugia il software *Ganglia*.

5.1 Ganglia

Ganglia è uno strumento di monitoraggio distribuito che permette di controllare diversi parametri di un gruppo di macchine, come carico del processore, spazio disco, uso della memoria etc...
Ogni nodo su cui è instalato *Ganglia* comunica con gli altri attaverso un canale MULTICAST, inviando i valori di ciascuna metrica monitorata utilizzando il formato XML[2] per la rappresentazione dei dati. *Ganglia* mette a disposizione un'interfaccia web per la visualizzazione dello stato dei nodi monitorati, si è quindi scelto di installarla nell'Install Server visto che era già presente *Apache*. Le metriche monitorate sono personalizzabili ed è possibile scriverne di nuove a seconda delle proprie esigenze, nel nostro caso abbiamo aggiunto ad esempio una metriche per monitorare il numero di processi totali in esecuzione.

[1] GridICE è uno strumento di controllo distribuito progettato per i sistemi Grid. É progettato su diversi livelli a seconda del tipo di utente che ne deve fare utilizzo: il livello Virtual Organization, il livello Grid Operation Center, il livello di amministrazione locale ed il livello dell'utente finale.

[2] XML è l'acronimo per eXtensible Markup Language. A dispetto del nome non si tratta propriamente di un linguaggio, ma di un meta linguaggio, cioè un linguaggio per costruire altri linguaggi.

5.2 Sviluppo di un nuovo strumento di monitoraggio: NodeGraph

L'unico difetto di *Ganglia*, per le nostre necessità, è quello che sono consultabili solo i parametri monitorati nell'ultima ora e non vengono archiviati i precedenti. Si è quindi pensato di creare uno strumento che permettesse di archiviare e rendere consultabili i dati rilevati da *Ganglia*: è nato così *NodeGraph*. Il suo funzionamento può essere schematizzato in figura 5.1.

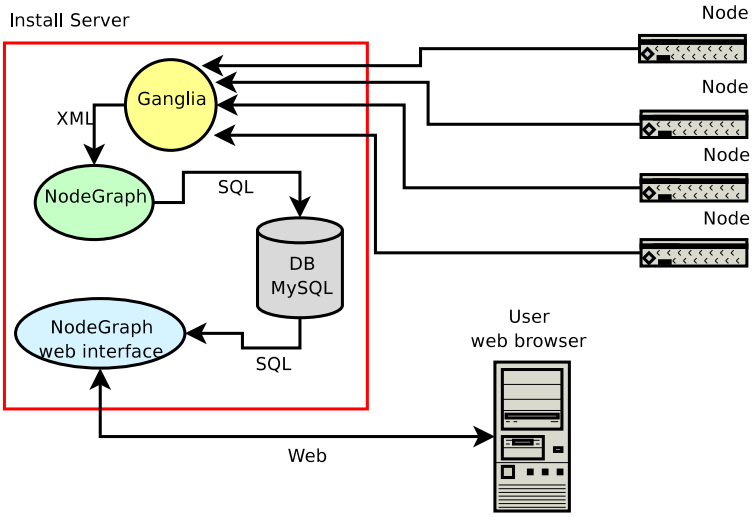

Figura 5.1: Schema di funzionamento di NodeGraph.

Abbiamo deciso di raccogliere i dati trasmessi dai client al server ogni 5 minuti, e di memorizzarli in un database MySQL; i dati del giorno vengono poi processati ed archiviati per creare una media giornaliera di ogni singola metrica per ogni singolo nodo. Il database generato, chiamato "monitoring", risulta composto dalle seguenti tabelle:

- *metrics*: utilizzata per archiviare le metriche monitorate, è composta da sei campi: il nome delle metriche usato su *Ganglia*, il nome esteso della metrica, l'unità di misura, la scala utilizzata, il tipo di visualizzazione per gruppi di nodi (somma o media) e il parametro su cui normalizzare la metrica se possibile (numero di CPU, numero di nodi o NULL se non è possibile normalizzare la media).
- *nodes*: contiene per ogni nodo monitorato le informazioni statiche, è composta da quattro campi: il FQDN, il nome visualizzato, il numero di processori e i gruppi a cui appartiene.

- <*nome_metrica*>: una tabella per ogni metrica contenente il FQDN, l'ora e la data del rilevamento e il valore di ogni metrica per ogni nodo ogni 5 minuti.

- <*nome_metrica*>*_daily*: una tabella per ogni metrica contenete per ogni giorno: il FQDN, il giorno e il valore medio giornaliero di quella metrica, se il valore è NULL significa che il numero di rilevazioni per quel giorno non era sufficiente per rendere la media attendibile.

5.2.1 Archiviazione dei dati

Come prima cosa, è necessario prelevare dal software *Ganglia* i valori di ogni metrica per ogni nodo. Tramite il comando `netstat` è possibile verificare la lista delle connessioni di rete elencando, con le opzioni "`lpn`", quelle in ascolto, il nome del processo che genera la connessione e la porta in forma numerica. Fra gli output del comando abbiamo:

```
tcp      0      0      *:8649      *:*      LISTEN      2294/gmond
tcp      0      0      *:8651      *:*      LISTEN      2150/gmetad
tcp      0      0      *:8652      *:*      LISTEN      2150/gmetad
```

che sono tutti e tre processi di *Ganglia*.
Consultando la documentazione, notiamo che il processo `gmetad`, nella porta 8651, esporta l'output XML. Proviamo quindi a contattare il servizio sulla porta 8651 con il comando "`/usr/bin/nc 127.0.0.1 8651`" e come ci aspettavamo riceviamo un output simile al seguente:

```
<HOST NAME="node12.grid.pg.infn.it" IP="192.168.254.12" REPORTED=
"1130928018" TN="7" TMAX="20" DMAX="0" LOCATION="unspecified"
GMOND_STARTED="1127893426">
<METRIC NAME="disk_free" VAL="27.470" TYPE="double" UNITS="GB" TN=
"175" TMAX="180" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="bytes_out" VAL="20.65" TYPE="float" UNITS="bytes/sec"
TN="301" TMAX="300" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="proc_total" VAL="49" TYPE="uint32" UNITS="" TN="186"
TMAX="950" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="pkts_in" VAL="4.80" TYPE="float" UNITS="packets/sec"
TN="301" TMAX="300" DMAX="0" SLOPE="both" SOURCE="gmond"/>
<METRIC NAME="cpu_nice" VAL="0.0" TYPE="float" UNITS="%" TN="66"
TMAX="90" DMAX="0" SLOPE="both" SOURCE="gmond"/>
...
...
...
</HOST>
```

Utilizziamo uno script in bash[3] che salva l'otput e lo invia ad uno script in Perl[4], quest'ultimo lo processa e lo salva sul database. Lo script in bash è il seguente:

```
#!/bin/bash
if [ ! -f /tmp/pars.xml ]
then
        /usr/bin/nc 127.0.0.1 8651 > /tmp/pars.xml
        /bin/cat /tmp/pars.xml | /usr/local/sbin/gangliaparser.pl
        /bin/rm -f /tmp/pars.xml
fi
```

Lo script, eseguito ogni 5 minuti tramite il processo di sistema `cron`, controlla se esiste il file `/tmp/pars.xml`, per evitare che sia ancora in corso il processo precedente prima di avviarne uno nuovo, in caso non esista viene creato il file avente come contenuto l'output del processo "`/usr/bin/nc 127.0.0.1 8651`". Il contenuto del file viene passato come standard input allo script in Perl `gangliaparser.pl` che lo processa, al termine viene rimosso il file temporaneo `/tmp/pars.xml`.
Lo script `gangliaparser.pl` prende in input il file XML, per ogni host del file controlla le metriche relative allo stesso, per ogni metrica, se essa è fra quelle da archiviare, esegue una query SQL che inserisce nell'apposita tabella <nome_metrica> il nome dell'host, il valore della metrica misurata e la data ed ora della rilevazione.
L'ultimo script che esegue l'operazione di archiviazione è `dailygenerator.pl`. Questo prende come argomenti la data ed un'opzione che serve per forzare (force) o non forzare (noforce) la rigenerazione della media giornaliera. Lo script preleva dal database i dati relativi al giorno passato in argomento (o al giorno precedente a quello corrente se non viene passato nessun argomento) per ogni nodo ed ogni metrica monitorata, se il numero di rilevazioni è inferiore a 48, corrispondenti a quattro ore, il nodo viene marcato *down* e viene inserito NULL nel valore della metrica relativa a quel giorno, altrimenti si esegue una media fra tutti i rilevamenti relativi al giorno e la si inserisce nel database. In caso la media sia già presente, se lo script viene chiamato con l'opzione noforce, questa non viene ricalcolata e sovrascritta; in caso invece questo venga chiamato con l'opzione force, la media viene ricalcolata e sovrascritta.

[3]Bash (acronimo per Bourne again shell) è una shell del progetto GNU usata nei sistemi operativi Unix e specialmente in Linux, si tratta di un interprete di comandi che permette all'utente di comunicare col sistema operativo attraverso una serie di funzioni predefinite, o di eseguire programmi.

[4]Perl, detto anche Pratical Extraction and Report Language, è un Linguaggio di programmazione procedurale interpretato creato nel 1987 da Larry Wall. Perl ha un singolare insieme di funzionalità mediate dal C, scripting shell Unix (sh), awk, sed e in diversa misura da molti altri linguaggi di programmazione.

5.2.2 Consultazione dei dati archiviati

Abbiamo deciso di visualizzare i dati archiviati via web sotto forma di istogrammi aventi sull'asse delle ascisse la data, sull'asse delle ordinate il valore della metrica monitorata. Per fare ciò abbiamo utilizzato *jpgraph*, un tool in php[5] che genera grafici partendo da un insieme di dati.
La parte di consultazione è divisa in quattro file: `select.php`, `generate.php`, `graph.php` e `graph-average.php`.
Spieghiamo ora nel dettaglio lo scopo e il funzionamento di ogni singolo file:

select.php

Questo è il *frontend* per l'utente (vedi figura 5.2), la sua struttura è divisa in due parti: a sinistra ci sono le opzioni da selezionare per la generazione del grafico, a destra vi è inizialmente una breve introduzione all'uso di *NodeGraph* e, una volta scelta la generazione del grafico, il risultato. Le opzioni sono composte da:

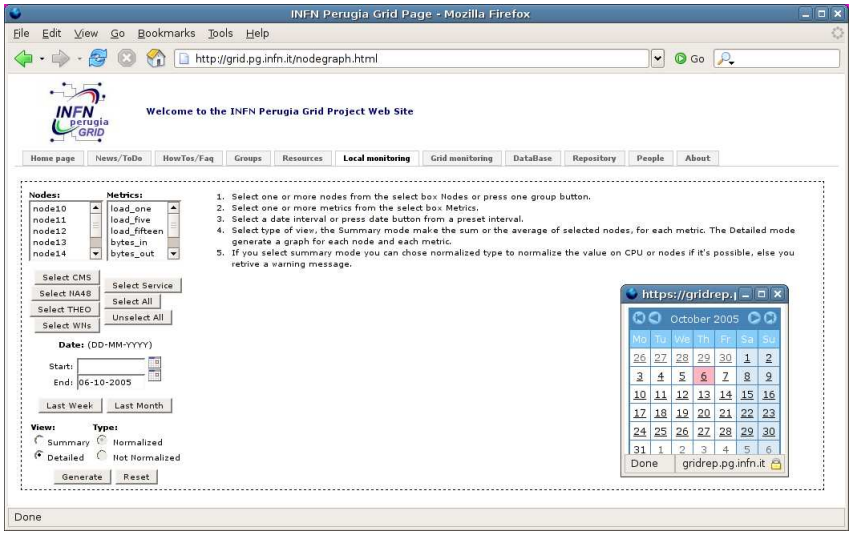

Figura 5.2: Pagina iniziale di NodeGraph: è possibile scegliere uno o più nodi, una o più metriche, un intervallo temporale (è possibile aiutarsi con un calendario), e il tipo di grafico che si vuole produrre. Nel IFRAME a destra c'è un breve how-to sull'utilizzo dello strumento.

- Due *select* multiple contenenti una la lista dei nodi monitorati, l'altra la lista delle metriche; i valori di ciascuna *select* vengono prelevati dal database.

[5]PHP è un linguaggio di programmazione Open Source utilizzato principalmente per la realizzazione di applicazioni lato server e pagine web dinamiche. Da qui il suo nome: un acronimo ricorsivo che sta per PHP: Hypertext Preprocessor.

- Una serie di pulsanti che permettono di selezionare i nodi facenti parte di uno stesso gruppo, questo avviene tramite delle funzioni in *JavaScript*[6].
- Due campi testo che devono contenere la data di inizio e di fine delle rilevazioni da mostrare.
- Due pulsanti per selezionare l'intervallo di generazione del grafico relativo all'ultima settimana o all'ultimo mese, anche questo è realizzato grazie a delle funzioni in *JavaScript*.
- Quattro *option box* per selezionare il tipo di vista:
  - *Summary*: genera un solo grafico per ogni metrica, i dati generati dipendono dalle opzioni:
    - *Normalized*: i dati sono una media normalizzata sul valore contenuto nella tabella relativa alla metrica; non tutte le metriche sono normalizzabili.
    - *Not Normalized*: i dati sono o la somma o una media semplice a seconda del valore contenuto nella tabella relativa alla metrica.
  - *Detailed*: genera un grafico per ogni metrica e per ogni nodo selezionato.
- Due pulsanti: uno per generare il grafico, richiamando il file `generate.php`, e uno per resettare i campi della form.

generate.php

Il file `generate.php`, richiamato dal file `select.php`, controlla se tutti i campi necessari sono inseriti e se il loro formato è corretto, altrimenti si interrompe. In caso si sia scelta la modalità *Summary* e *Normalized*, viene eseguito un ciclo per ogni metrica selezionata e controllato se è possibile normalizzare i dati per le metriche scelte; se questo non è possibile lo script restituisce un messaggio di errore e si interrompe. Lo script richiama poi `graph.php` se l'opzione *Detailed* è selezionata, o `graph-average.php` se è selezionata l'opzione *Summary*.

graph.php

Il file `graph.php` riceve come argomenti: il FQDN del nodo, la metrica e la data di inizio e di fine. Viene poi eseguita una query SELECT sulla tabella <nome_metrica>

[6] JavaScript è un linguaggio di scripting orientato agli oggetti comunemente usato nei siti web. Fu originariamente sviluppato da Brendan Eich della Netscape Communications con il nome di Mocha e successivamente di LiveScript, ma in seguito è stato rinominato "JavaScript" ed è stato formalizzato con una sintassi più vicina a quella del linguaggio Java di Sun Microsystems. JavaScript è stato standardizzato per la prima volta tra il 1997 e il 1999 dalla ECMA con il nome ECMAScript.

prelevando da essa i valori relativi al nodo e compresi nell'intervallo di tempo stabilito. Ogni valore, ordinato per data in modo crescente, viene controllato ed inserito in un vettore, se è NULL significa che il nodo è stato marcato *down* per quel giorno, viene quindi inserito il giorno in un apposito vettore. Vengono poi prelevati dalla tabella metrics del database la scala e l'unità di misura per la metrica selezionata. In base ai dati raccolti, viene generato l'istogramma contenente una banda rossa sui giorni marcati *down*.

graph-average.php

Il file `graph-average.php` riceve come argomenti: la lista dei FQDN dei nodi, la metrica, la data di inizio e di fine ed un flag per indicare se normalizzare o non i dati. Viene eseguito un ciclo per ogni nodo passato come argomento, all'interno del ciclo viene controllato il valore della metrica. Se è diverso da NULL vengono eseguite le seguenti operazioni: il valore relativo al nodo e al giorno in questione viene sommato con tutti i valori precedenti di quel giorno, viene incrementata una variabile che segna il numero di rilevamenti utili per quel giorno e viene incrementato il numero di CPU attive in quel giorno. Se invece il valore è NULL il giorno viene inserito in un vettore che contiene i giorni nei quali almeno un nodo, fra quelli selezionati, era *down*. Secondo il tipo di media scelta, normalizzata o meno, e secondo il tipo di normalizzazione, viene generato un vettore che viene poi usato per la creazione del grafico. Il vettore, contenente i giorni nei quali almeno un nodo sia marcato *down*, viene utilizzato per generare una banda di colore rosso nel grafico. Tale banda sta a significare che il rilevamento del giorno non è generato da tutti i nodi selezionati, ma solo da una parte.

5.2.3 Output ed Analisi dei dati

Lo strumento così sviluppato ha permesso una prima analisi dei dati raccolti negli ultimi sei mesi di funzionamento della farm e del sito Grid.
Prendendo solo i nodi di servizio (Install Server, CE ed SE) dell'infrastruttura notiamo in fig. 5.3 che il carico (load_five) è pressoché costante nel tempo, ed è quasi tutto concentrato nell'Install Server. Quindi da questo punto di vista non paiono esserci problemi.
Se prendiamo il carico su tutti i WN si notano in fig. 5.4 alcuni periodi di uso intenso, dove tuttavia non si raggiunge la saturazione a causa delle politiche di allocazione delle risorse tra i vari gruppi.
Nel grafico 5.5 invece, ottenuto selezionando i WN con il pulsante "Select WN", la metrica load_one (carico rilevato ogni minuto), e il tipo di visualizzazione "Summary" e "Normalized", si vedono fluttuazioni ampie a seconda dei periodi di uso. I risultati

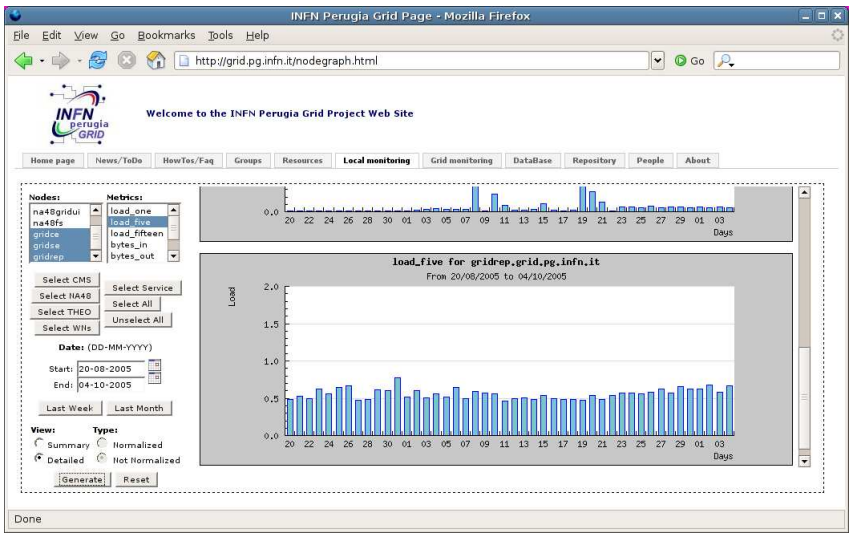

Figura 5.3: Grafico sul carico del processore per le macchine di servizio (Install Server, CE ed SE).

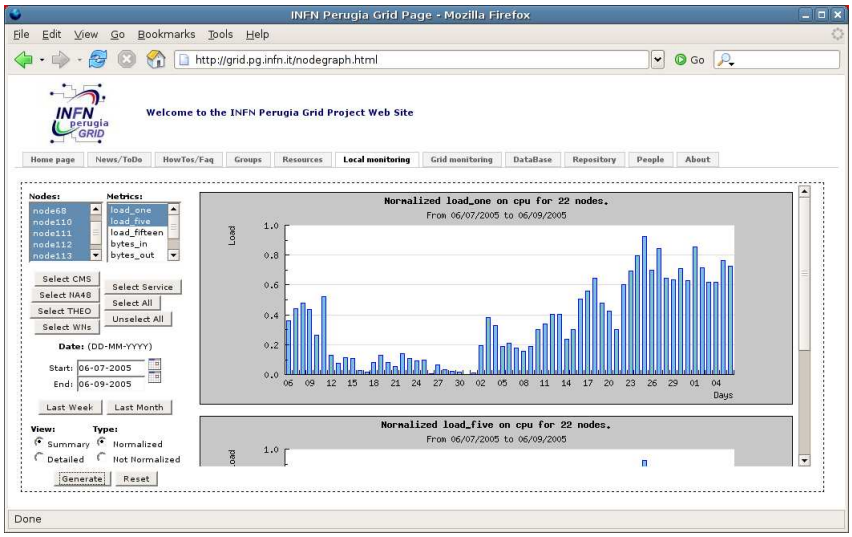

Figura 5.4: Grafico sul carico della CPU per i WN.

sono normalizzati sul carico medio per un processore dei WN della farm.
É possibile selezionare anche le macchine di un solo gruppo come in figura 5.6 e vediamo che l'uso ha le solite variazioni temporali, correlate con quelle delle figure precedenti, perché CMS è uno dei gruppi di utenti più attivi.
Si possono anche controllare i dati di uso per una singola macchina (cmsgridui ad

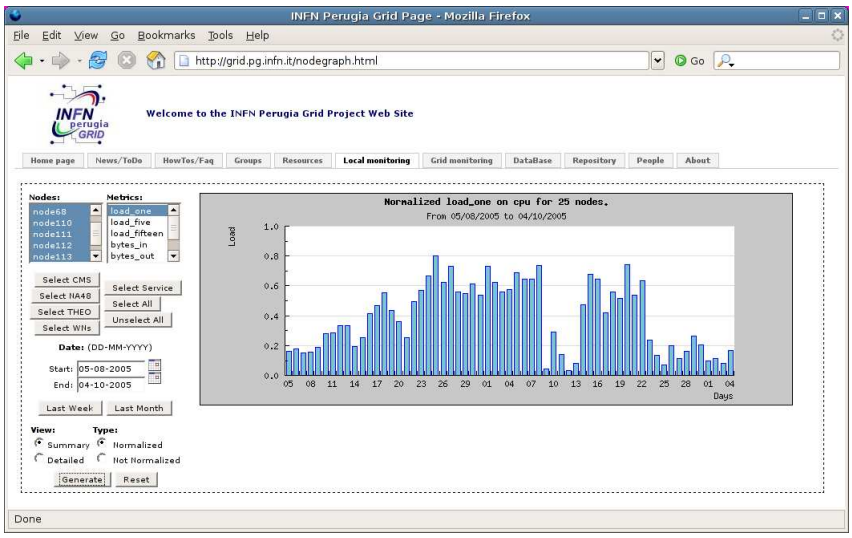

Figura 5.5: Grafico sul carico della CPU per i WN.

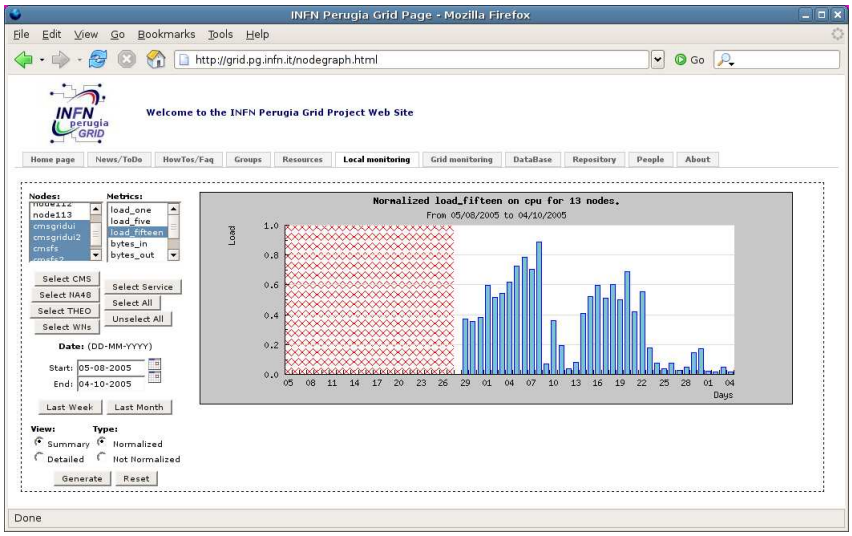

Figura 5.6: Grafico sul carico della CPU per le macchine del gruppo CMS: la metrica load_fifteen è stata introdotta in seguito, infatti nel grafico compare un banda rossa per il periodo nel quale i nodi selezionati non hanno rilevamenti utili per la metrica selezionata.

esempio) dedicata oltre che ad essere UI anche per far girare job in interattivo per gli utenti di quel gruppo. Si vedono il fig. 5.7 alcuni periodi di uso intensi, ed infatti tale funzionalità è stata ripartita su una seconda macchina (cmsgridui2).

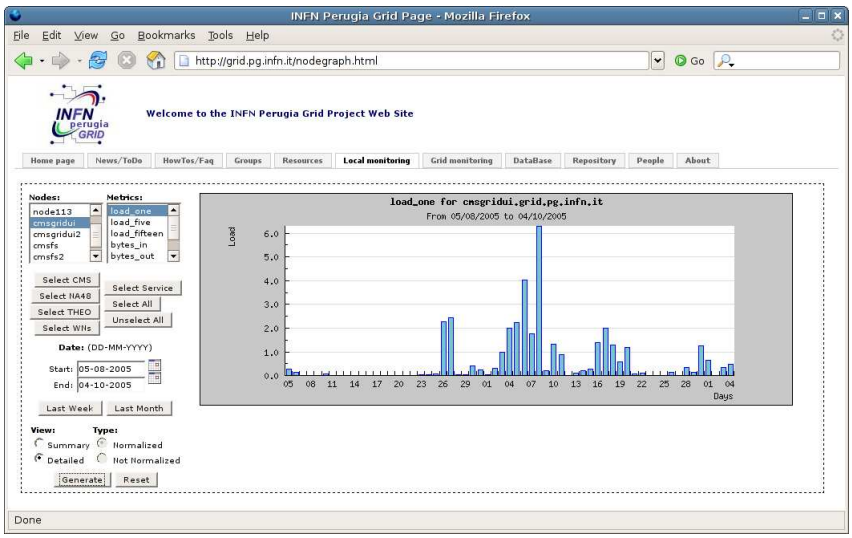

Figura 5.7: Grafico sul carico della CPU per cmsgridui.

Un'altra quantità che si può misurare è la percentuale di spazio disco libero nei vari fileserver. Ad esempio in fig. 5.8 vediamo l'evoluzione verificata per il gruppo CMS.

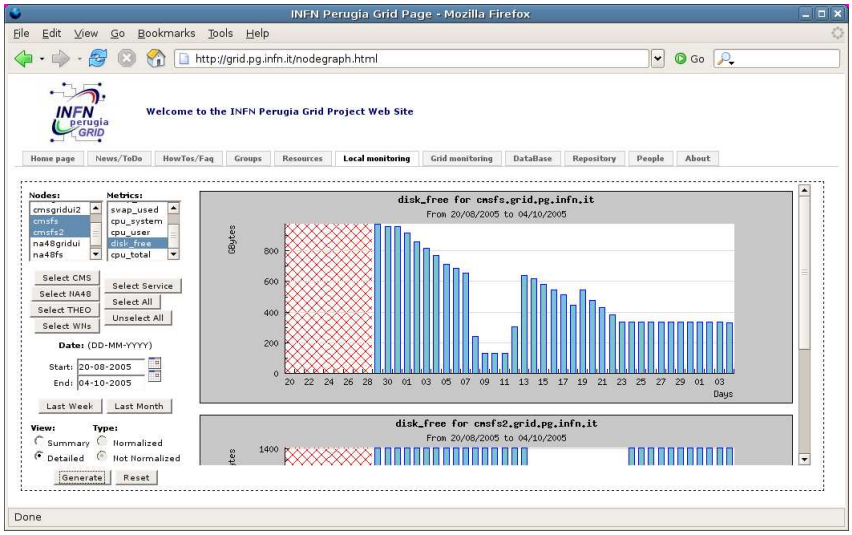

Figura 5.8: Grafico sullo spazio libero nei fileserver del gruppo CMS.

Infine si può stilare l'andamento dell'uso della banda passante per vedere se ci sono eventuali colli di bottiglia nell'uso della farm. In fig. 5.9 si vede l'evoluzione nel tempo di tale grandezza e anche qui si conclude che non sembrano esserci colli di bottiglia. A questo strumento si analisi nell'uso effettivo delle risorse andrebbe affiancata una

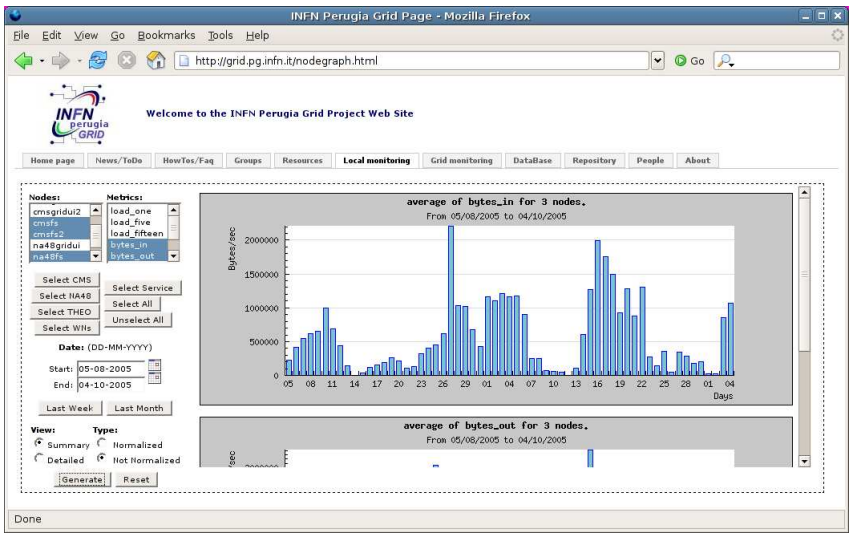

Figura 5.9: Grafico sull'uso della banda da parte dei fileserver.

analoga basata sui dati dello scheduler per verificare quale sia l'andamento delle richieste di uso, il numero di job in coda, i tempi medi di attesa, l'uso delle risorse da parte di singoli gruppi, anche in vista di una ottimizzazione dei parametri dello scheduler a seguito di una revisione delle politiche di allocazione delle risorse.

Conclusioni

In questa tesi ho descritto il lavoro svolto durante il periodo di stage presso la sezione INFN di Perugia. Inizialmente ho dovuto studiare e comprendere il funzionamento della tecnologia Grid raccogliendo numerose guide on-line e leggendo tesi di laurea inerenti a questo ambito. Contemporaneamente al mio ingresso si stava ultimando il *setup* della Grid di sviluppo, ho potuto così prendere confidenza con gli elementi di un sito standard. Nel periodo successivo ho iniziato dei test su *torque*, il batch system usato in Grid, per poter così capirne e testarne il funzionamento più a fondo. In vista dell'uscita della nuova versione del software di installazione e configurazione, ho iniziato a testarlo nella Grid di sviluppo per comprenderne il funzionamento e capire dove potevamo intervenire per adattarlo alle nostre esigenze. In seguito ho apportato le prime modifiche agli strumenti di installazione e configurazione fino ad ottenere una versione stabile da poter utilizzare nella Grid di produzione. Contemporaneamente alla conclusione delle modifiche la nuova versione è stata installata nel sito di produzione e testata a fondo. Le modifiche sono state poi sottoposte al CNAF, che si occupa della distribuzione di INFN-Grid, e sono state ritenute utili ed integrate nella release successiva. La configurazione standard è stata poi modificata per permettere ai gruppi di ricerca locali di poter usufruire delle macchine del sito come di una farm locale.

Con il rilascio delle nuove versioni di INFN-Grid sarà necessario probabilmente continuare ad eseguire le modifiche agli strumenti di configurazione.

La sezione di Perugia aveva la necessità di poter archiviare e consultare vari parametri che indicassero l'uso dei nodi della farm Grid, così da poter decidere se e come investire in nuovo hardware; ho così scritto uno strumento in grado di compiere questo compito partendo dai dati rilevati dallo strumento di monitoraggio *Ganglia*.

Attualmente lo strumento di monitoraggio non ha un interfaccia *user friendly* per l'amministrazione, sarà quindi necessario scrivere alcune pagine che permettano di gestirlo in modo semplice ed intuitivo. Sarà poi necessario scrivere una parte di codice che si occupi di archiviare i dati relativi al numero di job eseguiti, al loro tipo e al gruppo di appartenenza.

Ringraziamenti

Un sentito ringraziamento al Prof. Leonello Servoli per avermi dato la possibilità di lavorare a questo progetto, a Mirko Mariotti per il supporto tecnico, per il sostegno nei momenti più duri e per tutti i pranzi consumati insieme (a volte anche di sabato!!!). Alla mia famiglia per il supporto economico e **non solo**. A tutto il gruppo "multi-player", Daniele, Marco, Massimo, Michele, Mirko e Tiziano. A Fabrizio e Riccardo, gli amici di sempre. A Claudio, Daniele, Giacomo, Luca, Simone e Tiziano che mi hanno accompagnato nella carriera scolastica. Un saluto particolare a Valerio, anche se ultimamente ci siamo persi un po' di vista, abbiamo passato belle serate insieme. A tutti gli amici che mi hanno sostenuto durante questi anni passati all'università e con i quali ho condiviso molto. A Linus Torvalds e Richard Stallman per averci dato la possibilità di scegliere.
E per ultima, non per importanza, alla persona che mi ha consentito di raggiungere i miei obiettivi, a colei che mi ha sostenuto nei momenti più difficili e con la quale ho condiviso i momenti più belli: a Barbara.

Appendice A

Funzioni modificate e configurazione del batch-system

A.1 Il file ig-site-info.def utilizzato

```
# YAIM example site configuration file - adapt it to your site!
# INFN-GRID: set PRIVATE_NETWORK=true to use WN on private network
PRIVATE_NETWORK=true
# INFN-GRID: if PRIVATE_NETWORK=true, uncomment and write the internal domain name
MY_INT_DOMAIN=grid.pg.infn.it
# INFN-GRID: if PRIVATE_NETWORK=true, uncomment and write your internal network
INT_NET=192.168.254.0/24
# INFN-GRID: if PRIVATE_NETWORK=true, uncomment and write the internal FQDN hostname of CE
CE_INT_HOST=gridce.$MY_INT_DOMAIN
# INFN-GRID: IF PRIVATE_NETWORK=true, uncomment and write the internal hostname
# of the hostt exporting the directory used to install the application software
INT_HOST_SW_DIR=$CE_INT_HOST


# You must set an RB even if you are not installing it.
# You can uncomment one of these two RB (or select another one if you like)
# * RB that can access to all EGEE + LCG + INFN-GRID resources
RB_HOST=egee-rb-01.cnaf.infn.it
# * RB that can access to Italian Region (INFN-GRID) resources
# RB_HOST=gridit-rb-01.cnaf.infn.it
PX_HOST=myproxy.cnaf.infn.it
BDII_HOST=egee-bdii-01.cnaf.infn.it
MON_HOST=$SE_HOST
REG_HOST=lcgic01.gridpp.rl.ac.uk        # there is only 1 central registry for now

# INFN-GRID: DAG RAB. Uncomment and write here if you are installing an RB DAG
# RB_DAG_HOST=my-rb-dag.$MY_DOMAIN
# INFN-GRID: Network Monitoring (theodolite). Uncomment it if you are installing it
# NETWORKMON_HOST=my-nm.$MY_DOMAIN
# INFN-GRID: space separed list of the IP addresses of the NTP servers
NTP_HOSTS_IP="131.154.1.53"

# Set this if you are building a LFC server
# not if youre just using clients
# LFC_HOST=my-lfc.$MY_DOMAIN

WN_LIST=/opt/lcg/yaim/examples/wn-list.conf
```

```
# INFN-GRID: directory exported for SGM (it will be mounted by WN on VO_SW_DIR, see below)
BASE_SW_DIR=/opt/exp_soft
# INFN-GRID: host exporting the directory for SGM (a CE or a SE)
HOST_SW_DIR=$CE_HOST
# INFN-GRID local users list
USERS_CONF=/opt/lcg/yaim/examples/ig-users.conf
FUNCTIONS_DIR=/opt/lcg/yaim/functions

# If you are using a an apt+kickstart server set the INSTALL_SERVER_HOST
INSTALL_SERVER_HOST=192.168.254.5
OS_REPOSITORY="rpm http://$INSTALL_SERVER_HOST/rep slc304-i386 os updates extras localrpms"
LCG_REPOSITORY="rpm http://$INSTALL_SERVER_HOST/rep lcg_sl3-i386 2_4_0 2_4_0_updates"
IG_REPOSITORY="rpm http://$INSTALL_SERVER_HOST/rep ig_sl3-i386 2_4_0 utils"

# Uncomment the following lines and comment the previous ones if you
# are a not using a local apt repository:
# OS_REPOSITORY="rpm http://linuxsoft.cern.ch/ cern/slc304/i386/apt os updates extras"
# LCG_REPOSITORY="rpm http://grid-deployment.web.cern.ch/grid-deployment/gis apt/LCG-2_4_0/
en/i386 lcg_sl3 lcg_sl3.updates"
# IG_REPOSITORY="rpm http://grid-it.cnaf.infn.it/ apt/ig_sl3-i386 2_4_0 utils"

CA_REPOSITORY="rpm http://grid-deployment.web.cern.ch/grid-deployment/gis apt/LCG_CA/en/
i386 lcg"
CA_WGET="http://grid-deployment.web.cern.ch/grid-deployment/download/RpmDir/security/index.
html"

# For the relocatable (tarball) distribution, ensure
# that INSTALL_ROOT is set correctly
INSTALL_ROOT=/opt

# Youll probably want to change these too for the relocatable dist
OUTPUT_STORAGE=/tmp/jobOutput
JAVA_LOCATION="/usr/java/j2sdk1.4.2_08"

# Set this to '/dev/null' or some other dir if you want
# to turn off yaims installation of cron jobs
CRON_DIR=/etc/cron.d

GLOBUS_TCP_PORT_RANGE="20000 25000"

# WARNING: plain text password!
MYSQL_PASSWORD=fjhek4hjk

GRID_TRUSTED_BROKERS="'/C=IT/O=INFN/OU=Host/L=CNAF/CN=gridit-rb-01.cnaf.infn.it' '/C=IT/O=
INFN/OU=Host/L=CNAF/CN=edee-rb-01.cnaf.infn.it'"

# GRIDMAP_AUTH="ldap://lcg-registrar.cern.ch/ou=users,o=registrar,dc=lcg,dc=org"

GRIDICE_SERVER_HOST=$MON_HOST
# INFN-GRID: enable monitoring on WN of generic processes/daemon using gridice
GRIDICE_MON_WN=yes

SITE_EMAIL=grid-prod@pg.infn.it
SITE_NAME=INFN-PERUGIA
SITE_VERSION=LCG-2_4_0

SE_TYPE=disk
JOB_MANAGER=lcgpbs
# INFN-GRID: write here pbs even if you are using torque for compatibility
# with MPI support on Globus
CE_BATCH_SYS=pbs
CE_CPU_MODEL=PIII
CE_CPU_VENDOR=intel
CE_CPU_SPEED=1001
CE_OS=SLC
CE_OS_RELEASE=3.0.4
CE_MINPHYSMEM=1024
CE_MINVIRTMEM=2048
CE_SMPSIZE=2
CE_SI00=381
CE_SF00=0
CE_OUTBOUNDIP=TRUE
CE_INBOUNDIP=TRUE
CE_RUNTIMEENV="LCG-2 LCG-2_1_0 LCG-2_1_1 LCG-2_2_0 LCG-2_3_0 LCG-2_3_1 LCG-2_4_0 R-GMA
```

```
AFS PERUGIA MPICH"
CE_CLOSE_SE="SE1"
CE_CLOSE_SE1_HOST=$SE_HOST
# IMPORTANT: IF YOU ARE UPGRADING AN EXISTING STORAGELEMENT PLEASE CHECK THAT
# THIS PATH IS THE ONE USED IN THE PAST OTHERWISE FILES REGISTERED IN THE
# CATALOG WILL NOT BE CONSISTENT!!
CE_CLOSE_SE1_ACCESS_POINT=/flatfiles/SE00
# CE_CLOSE_SE2_HOST=another-se.$MY_DOMAIN
# CE_CLOSE_SE2_ACCESS_POINT=/somewhere

# dCache-specific settings
# DCACHE_ADMIN="my-admin-node"
# DCACHE_POOLS="my-pool-node1:/pool-path1 my-pool-node2:/pool-path2"
# Optional
# DCACHE_PORT_RANGE="20000,25000"

# SE_dpm-specific settings
# DPM_POOLS="lxb1727:/dpmpool2"
# Optional
# DPM_PORT_RANGE="20000,25000" ??
# DPMDATA=$CE_CLOSE_SE1_ACCESS_POINT
# DPMDB_PWD=dpmu_Bar
# DPMUSER_PWD=dpmu_Bar
# DPMCONFIG=/home/dpmuser/DPMCONFIG
# DPMLOGS=/var/tmp/DPMLogs
# DPMFSIZE=200M
# DPM_HOST=$SE_HOST
# # Temp
# DPMPOOL=dpmpool2

BDII_HTTP_URL="http://grid-deployment.web.cern.ch/grid-deployment/gis/lcg2-bdii/dteam/
lcg2-all-sites.conf"
BDII_REGIONS="CE SE"    # list of the services provided by the site
BDII_CE_URL="ldap://$CE_HOST:2135/mds-vo-name=local,o=grid"
BDII_SE_URL="ldap://$SE_HOST:2135/mds-vo-name=local,o=grid"
# BDII_RB_URL="ldap://$RB_HOST:2135/mds-vo-name=local,o=grid"
# BDII_PX_URL="ldap://$PX_HOST:2135/mds-vo-name=local,o=grid"

# INFN-GRID: remove the VO you do not support (gilda
VOS="cms dteam \
     infngrid theophys \
     babar gridit"
# others VOs: gilda e

QUEUES="cms cert grid"

VO_SW_DIR=/opt/exp_soft

VO_ATLAS_SW_DIR=$VO_SW_DIR/atlas
VO_ATLAS_DEFAULT_SE=$SE_HOST
VO_ATLAS_SGM=ldap://grid-vo.nikhef.nl/ou=lcgadmin,o=atlas,dc=eu-datagrid,dc=org
VO_ATLAS_USERS=ldap://grid-vo.nikhef.nl/ou=lcg1,o=atlas,dc=eu-datagrid,dc=org
VO_ATLAS_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/atlas
VO_ATLAS_QUEUES="atlas"

VO_ALICE_SW_DIR=$VO_SW_DIR/alice
VO_ALICE_DEFAULT_SE=$SE_HOST
VO_ALICE_SGM=ldap://grid-vo.nikhef.nl/ou=lcgadmin,o=alice,dc=eu-datagrid,dc=org
VO_ALICE_USERS=ldap://grid-vo.nikhef.nl/ou=lcg1,o=alice,dc=eu-datagrid,dc=org
VO_ALICE_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/alice
VO_ALICE_QUEUES="alice"

VO_CMS_SW_DIR=$VO_SW_DIR/cms
VO_CMS_DEFAULT_SE=$SE_HOST
VO_CMS_SGM=ldap://grid-vo.nikhef.nl/ou=lcgadmin,o=cms,dc=eu-datagrid,dc=org
VO_CMS_USERS=ldap://grid-vo.nikhef.nl/ou=lcg1,o=cms,dc=eu-datagrid,dc=org
VO_CMS_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/cms
VO_CMS_QUEUES="cms"

VO_LHCB_SW_DIR=$VO_SW_DIR/lhcb
VO_LHCB_DEFAULT_SE=$SE_HOST
VO_LHCB_SGM=ldap://grid-vo.nikhef.nl/ou=lcgadmin,o=lhcb,dc=eu-datagrid,dc=org
VO_LHCB_USERS=ldap://grid-vo.nikhef.nl/ou=lcg1,o=lhcb,dc=eu-datagrid,dc=org
VO_LHCB_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/lhcb
```

```
VO_LHCB_QUEUES="lhcb"

VO_DTEAM_SW_DIR=$VO_SW_DIR/dteam
VO_DTEAM_DEFAULT_SE=$SE_HOST
VO_DTEAM_SGM=ldap://lcg-vo.cern.ch/ou=lcgadmin,o=dteam,dc=lcg,dc=org
VO_DTEAM_USERS=ldap://lcg-vo.cern.ch/ou=lcg1,o=dteam,dc=lcg,dc=org
VO_DTEAM_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/dteam
VO_DTEAM_QUEUES="cert"

VO_SIXT_SW_DIR=$VO_SW_DIR/sixt
VO_SIXT_DEFAULT_SE=$SE_HOST
VO_SIXT_USERS=ldap://lcg-vo.cern.ch/ou=lcg1,o=sixt,dc=lcg,dc=org
VO_SIXT_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/sixt
VO_SIXT_QUEUES="grid"

VO_BIO_SW_DIR=/opt/exp_soft/bio
VO_BIO_DEFAULT_SE=$SE_HOST
VO_BIO_SGM=ldap://grid-vo.cnaf.infn.it:10389/ou=bioadmin,o=bio,c=it
VO_BIO_USERS=ldap://grid-vo.cnaf.infn.it:10389/ou=Testbed-Firb,o=bio,c=it
VO_BIO_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/bio
VO_BIO_QUEUES="grid"

VO_ENEA_SW_DIR=$VO_SW_DIR/enea
VO_ENEA_DEFAULT_SE=$SE_HOST
VO_ENEA_SGM=
VO_ENEA_USERS=ldap://grid-vo.cnaf.infn.it:10389/ou=Testbed-enea,o=enea,c=it
VO_ENEA_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/enea
VO_ENEA_QUEUES="grid"

VO_INAF_SW_DIR=$VO_SW_DIR/inaf
VO_INAF_DEFAULT_SE=$SE_HOST
VO_INAF_SGM=
VO_INAF_USERS=ldap://grid-vo.cnaf.infn.it:10389/ou=Testbed-Firb,o=inaf,c=it
VO_INAF_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/inaf
VO_INAF_QUEUES="grid"

VO_PLANCK_SW_DIR=$VO_SW_DIR/planck
VO_PLANCK_DEFAULT_SE=$SE_HOST
VO_PLANCK_SGM=
VO_PLANCK_USERS=
VO_PLANCK_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/planck
VO_PLANCK_VOMS_RB=vomss://voms.cnaf.infn.it:8443/edg-voms-admin/planck?/planck
VO_PLANCK_QUEUES="grid"

VO_BIOMED_SW_DIR=$VO_SW_DIR/biomed
VO_BIOMED_DEFAULT_SE=$SE_HOST
VO_BIOMED_SGM=ldap://vo-biome.in2p3.fr/ou=lcgadmin,o=biomedical,dc=lcg,dc=org
VO_BIOMED_USERS=ldap://vo-biome.in2p3.fr/ou=lcg1,o=biomedical,dc=lcg,dc=org
VO_BIOMED_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/biomed
VO_BIOMED_QUEUES="grid"

VO_ESR_SW_DIR=$VO_SW_DIR/esr
VO_ESR_DEFAULT_SE=$SE_HOST
VO_ESR_SGM=ldap://grid-vo.sara.nl/ou=lcgadmin,o=esr,dc=eu-egee,dc=org
VO_ESR_USERS=ldap://grid-vo.sara.nl/o=esr,dc=eu-egee,dc=org
VO_ESR_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/esr
VO_ESR_QUEUES="grid"

VO_INFNGRID_SW_DIR=$VO_SW_DIR/infngrid
VO_INFNGRID_DEFAULT_SE=$SE_HOST
VO_INFNGRID_SGM=
VO_INFNGRID_USERS=
VO_INFNGRID_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/infngrid
VO_INFNGRID_VOMS_RB=vomss://voms.cnaf.infn.it:8443/edg-voms-admin/infngrid?/infngrid
VO_INFNGRID_QUEUES="cert"

VO_THEOPHYS_SW_DIR=$VO_SW_DIR/theophys
VO_THEOPHYS_DEFAULT_SE=$SE_HOST
VO_THEOPHYS_SGM=
VO_THEOPHYS_USERS=ldap://grid-vo.cnaf.infn.it/o=TheoPhys,dc=eu-datagrid,dc=org
VO_THEOPHYS_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/theophys
VO_THEOPHYS_QUEUES="grid"

VO_CDF_SW_DIR=$VO_SW_DIR/cdf
```

```
VO_CDF_DEFAULT_SE=$SE_HOST
VO_CDF_SGM=
VO_CDF_USERS=
VO_CDF_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/cdf
VO_CDF_VOMS_RB=vomss://voms.cnaf.infn.it:8443/edg-voms-admin/cdf?/cdf
VO_CDF_QUEUES="grid"

# VO_GILDA_SW_DIR=$VO_SW_DIR/gilda
# VO_GILDA_DEFAULT_SE=$SE_HOST
# VO_GILDA_SGM=
# VO_GILDA_USERS=ldap://grid-vo.cnaf.infn.it:10389/ou=Testbed-gilda,o=gilda,c=it
# VO_GILDA_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/gilda
# VO_GILDA_QUEUES="grid"

VO_INGV_SW_DIR=$VO_SW_DIR/ingv
VO_INGV_DEFAULT_SE=$SE_HOST
VO_INGV_SGM=
VO_INGV_USERS=ldap://grid-vo.cnaf.infn.it:10389/ou=Testbed-Firb,o=ingv-bologna,c=it
VO_INGV_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/ingv
VO_INGV_QUEUES="grid"

VO_VIRGO_SW_DIR=$VO_SW_DIR/virgo
VO_VIRGO_DEFAULT_SE=$SE_HOST
VO_VIRGO_SGM=ldap://grid-vo.cnaf.infn.it/ou=virgosgm,o=virgo,dc=eu-datagrid,dc=org
VO_VIRGO_USERS=ldap://grid-vo.cnaf.infn.it/ou=testbed1,o=virgo,dc=eu-datagrid,dc=org
VO_VIRGO_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/virgo
VO_VIRGO_QUEUES="grid"

VO_BABAR_SW_DIR=$VO_SW_DIR/babar
VO_BABAR_DEFAULT_SE=$SE_HOST
VO_BABAR_SGM=ldap://babar-vo.gridpp.ac.uk/ou=babarsgm,dc=gridpp,dc=ac,dc=uk
VO_BABAR_USERS=ldap://babar-vo.gridpp.ac.uk/ou=babar,dc=gridpp,dc=ac,dc=uk
VO_BABAR_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/babar
VO_BABAR_QUEUES="grid"

VO_COMPCHEM_SW_DIR=$VO_SW_DIR/compchem
VO_COMPCHEM_DEFAULT_SE=$SE_HOST
VO_COMPCHEM_SGM=
VO_COMPCHEM_USERS=
VO_COMPCHEM_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/compchem
VO_COMPCHEM_VOMS_RB=vomss://voms.cnaf.infn.it:8443/edg-voms-admin/compchem?/compchem
VO_COMPCHEM_QUEUES="grid"

VO_GRIDIT_SW_DIR=$VO_SW_DIR/gridit
VO_GRIDIT_DEFAULT_SE=$SE_HOST
VO_GRIDIT_SGM=
VO_GRIDIT_USERS=ldap://grid-vo.cnaf.infn.it:10389/ou=Testbed-Firb,o=gridit,c=it
VO_GRIDIT_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/gridit
VO_GRIDIT_QUEUES="grid"

VO_MAGIC_SW_DIR=$VO_SW_DIR/magic
VO_MAGIC_DEFAULT_SE=$SE_HOST
VO_MAGIC_SGM=ldap://grid-vo.sara.nl/ou=lcgadmin,o=magic,dc=eu-egee,dc=org
VO_MAGIC_USERS=ldap://grid-vo.sara.nl/ou=magicians,o=magic,dc=eu-egee,dc=org
VO_MAGIC_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/magic
VO_MAGIC_QUEUES="grid"

VO_ZEUS_SW_DIR=$VO_SW_DIR/zeus
VO_ZEUS_DEFAULT_SE=$SE_HOST
VO_ZEUS_SGM=
VO_ZEUS_USERS=ldap://grid-vo.desy.de/ou=zeus,ou=vo,o=desy,c=de
VO_ZEUS_STORAGE_DIR=$CE_CLOSE_SE1_ACCESS_POINT/zeus
VO_ZEUS_QUEUES="grid"
```

A.2 Il file ig-bootselect

```
#!/usr/bin/perl
#
# ig-bootselect
# ================================================================
# Select a pxelinux configuration file for the boot
# Author: Enrico Ferro <enrico.ferro@pd.infn.it>
```

```
#

use strict;
use Socket;
use English;

my $program = "ig-bootselect";
my $version = "1.0";

# PXELINUX configuration directory. Here for each node there
# should be a link called as <IpAddressInHex> to the pxelinux
# configuration file to be used
my $pxelinux_dir="/tftpboot/pxelinux.cfg";

# Array of available pxelinux configurations
my @cfg;

# Nodes (keys) and their config (values)
my %nodes;

# Number of terminal lines
my $lines;

# Default boot
my $default_boot = "boot-hd.cfg";

# dialog's return codes
my $DIALOG_EXIT  = 256;
my $DIALOG_ABORT = 65280;
my $DIALOG_EXTRA = 768;
my $DIALOG_OK    = 0;

################################################################################
sub GetHexAddr ($) {
################################################################################
# Get the hostname or an IP address and return the IP address in
# hex (as required by pxelinux)
my $hostname = shift;

  # The 4th field is an array of the IP address of this node
  my @all_address = (gethostbyname($hostname))[4];
  if ($#all_address < 0) { # The array is empty
    return ;
  }
  # We unpack the IP address
  my @tmp_address = unpack('C4',$all_address[0]);
  return sprintf ("%02X%02X%02X%02X",$tmp_address[0], $tmp_address[1],
                                     $tmp_address[2], $tmp_address[3]);
}

################################################################################
sub GetHostname ($) {
################################################################################
# Get the hostname from HEX-IP address
  my @res;
  my $ip_address = hex($_);
  # Of course there is should be already something better to do it... :-(
  for (my $i=0; $i<4; $i++) {
    $res[1] = ($ip_address & 255) . '.' . $res[1];
    $ip_address = $ip_address >> 8;
  }
  chop($res[1]);

# ---------------------- Start INFN Perugia mod -----------------
  $res[0]=(gethostbyaddr(inet_aton($res[1]), AF_INET))[0];
# --------------------- end ---------------------------
  return @res;
}

################################################################################
sub Initialize {
################################################################################
# Initialize the list of configurations available, size of the terminal
  if (! -d $pxelinux_dir) {
```

```
    print ("Error: PXE Linux configuration directory not present ($pxelinux_dir)\n");
    exit(1);
  }
  if (! -f $pxelinux_dir . '/' . $default_boot) {
    print ("Error: missing default boot configuration ($pxelinux_dir/$default_boot)\n");
    exit(1);
  }
  if ($EUID ne 0) {
    print ("Error: you must be root to run this program\n");
    exit(1);
  }
  # Load the configurations list
  opendir(DIR, $pxelinux_dir);
  @cfg = sort(grep(/(\.cfg$)|(default)/, readdir(DIR)));
  closedir(DIR);
  # Get the terminal size
  $lines =`stty size`;
  chomp($lines);
  $lines =~ s/ .+//;
}

################################################################################
sub ExecDialog($) {
################################################################################
# Run the dialog tool. Return the button choosed (1st value) and
# the element/elements selected (2nd value, if applicable)
  my $cmd = shift;
  open (F,"dialog --backtitle \"$program $version\" $cmd 2>&1|");
  my $r = <F>;
  close(F);
  my $c = $?;
  $r =~ s/"//g;
  return ($c,$r);
}

################################################################################
sub ScanSymlinks() {
################################################################################
# Return true if some nodes where found. For each node we get its
# current cfg file in a hash table. Return false if no nodes are found, true
# otherwise.

  my ($hostname, $cfg);
  %nodes = ();
  # Load the nodes list
  opendir(DIR, $pxelinux_dir);
  my @hex_nodes = grep(/^[[:xdigit:]]{8}$/, readdir(DIR));
  closedir(DIR);
  foreach (@hex_nodes) {
    $hostname = (GetHostname($_))[0];
    # Load the configuration
    $cfg = "";
    $cfg = readlink($pxelinux_dir . "/" . $_);
    # Set default config is symlink is missing
    if ($cfg && $hostname) {
      $nodes{$hostname}= $cfg;
    } else {
      unlink($pxelinux_dir . "/" . $_);
      Message("Host not found for $_: the symlink was removed");
    }
  }
  return (%nodes !=());
}

################################################################################
sub Message ($){
################################################################################
# A shortcut to print an information message...
 my $text = shift;
 ExecDialog("--title \" Information \" --msgbox \"$text\" 8 60");
}

################################################################################
sub UpdateSymlinks ($$) {
```

```
##############################################################################
# Receive a as parameters a 1. space-separed list of nodes 2. their new
# configuration and updates their symlinks
 my $selected = shift;
 my $new_cfg  = shift;
 my $hex_addr;
 my $error_found = 0 ;
 $selected =~ s/"//g;
  my @sel = split(/\ /,$selected);
  foreach  (@sel) {
    $hex_addr = GetHexAddr($_);
    if (!$hex_addr) {
      Message("Failed to detect the IP address for $_.\n" .
              "Verify that /etc/resolv.conf contains a line like:\n\n" .
              "domain your.domain");
      $error_found++;
    } elsif (system ("cd $pxelinux_dir && rm -f $hex_addr && ln -fs $new_cfg
$hex_addr && " . "chown apache:apache $hex_addr") !=0) {
      Message("Failed update for \"$_\" with the configuration of \"$new_cfg\"\n");
      $error_found++;
    }
  }
  if ($error_found > 1) {
    Message("$error_found errors found");
  }
}

##############################################################################
sub RemoveSymlinks ($) {
##############################################################################
# Receive a space separed list of nodes to remove and remove their symlinks
  my $selected = shift;
  my $hex_addr;
  my $error_found = 0 ;
  my @sel = split(/\ /,$selected);
  foreach (@sel) {
    $hex_addr = GetHexAddr($_);
    if (system ("rm -f $pxelinux_dir/$hex_addr") !=0) {
       Message("Failed removal of \"$_\"");
       $error_found++;
    }
  }
  if ($error_found > 1) {
    Message("$error_found errors found during removal");
  }
}


##############################################################################
sub ShowDialogOperationSelect($) {
##############################################################################
# Show a dialog box where you can select the new boot type or to remove
  my $selected = shift;
  my $cmd = "--clear --title \" Boot type / remove selection \" " .
     " --cancel-label Exit " .
     " --ok-label 'Apply' " .
     " --cancel-label Back " .
     " --menu \"Select the new boot type or to remove the selected nodes:\" " .
     ($lines - 5) . " 75 " . ($lines - 12);
  foreach (@cfg) {
    if ($_ eq $default_boot) {
      $cmd .= " $_ \"Normal boot (no OS installation)\" ";
    } else {
      $cmd .= " $_ \"Install OS using $_\" ";
    }
  }
  $cmd .= " remove \"Remove from the list\" ";
  my ($code,$operation)=ExecDialog($cmd);
  if ($code == $DIALOG_OK) {
    if ($operation eq "remove") {
      RemoveSymlinks($selected);
    } else {
      UpdateSymlinks($selected, $operation);
    }
```

```
  }
}

##############################################################################
sub ShowDialogNodesSelect() {
##############################################################################
# Main dialog, return 1. the button chosen 2. a space separed 1list of selected nodes
  my $cmd = "--clear --title \" Hosts and boot type \" " .
     " --cancel-label Exit " .
     " --ok-label Modify/Remove " .
     " --extra-button --extra-label 'Add hosts' " .
     " --checklist \"Select the hosts with SPACE.\nPress \'Modify/Remove\'" .
     " to change their boot type or to remove them from the list." .
     "\nPress \'Add hosts\' to add hosts not present in this list.\" " .
     ($lines - 5) . " 60 " . ($lines - 15) ;
  foreach (sort(keys(%nodes))) {
    $cmd = " $cmd $_ \"" . $nodes{$_} . "\" off ";
  }
  return ExecDialog($cmd);
}

##############################################################################
sub ShowDialogAddHosts() {
##############################################################################
# Dialog where to add new nodes. Button pushed is returned to rerun
# it until the user is tired to add
  my $cmd = "--title \" Add new nodes \" " .
            " --cancel-label Close" .
            " --ok-label Add " .
            " --inputbox \"Write a list of hostnames separated by blanks:\" 8 70 ";
  my ($code,$to_add)=ExecDialog($cmd);
  if ($code == $DIALOG_OK && $to_add ne '') {
    UpdateSymlinks($to_add,$default_boot);
  }
  return $code;
}

##############################################################################
# MAIN
##############################################################################
Initialize;

my ($code,$selected);
while (1) {
  if (!ScanSymlinks) {
    Message('No nodes configured: add some nodes in the next dialog box or exit');
    do {
      $code=ShowDialogAddHosts;
    } while ($code == $DIALOG_OK);
    # If no nodes where added the user wants to exit
    if (!ScanSymlinks) {
      last;
    }
  } else {
     # Standard operations: show the nodes list
    ($code,$selected)=ShowDialogNodesSelect;
    if ($code == $DIALOG_OK) {
      if ($selected) {
         ShowDialogOperationSelect($selected);
       } else {
         Message("No nodes selected");
       }
    } # Add some new nodes
    elsif ($code == $DIALOG_EXTRA) {
      do {
        $code=ShowDialogAddHosts;
      } while ($code == $DIALOG_OK);
    } else {
      last;
    }
  }
}

exit(0);
```

A.3 La funzione config_nfs_sw_dir_client

```
function config_nfs_sw_dir_client () {

# Author: Enrico Ferro (enrico.ferro <at> pd.infn.it)
# Configure /etc/fstab to mount via nfs a directory with software for SGM

if [ -z "$VO_SW_DIR" ] ; then
   echo "\$VO_SW_DIR not defined"
   return 1
fi
if [ -z "$BASE_SW_DIR" ] ; then
   echo "\$BASE_SW_DIR not defined"
   return 1
fi
if [ $PRIVATE_NETWORK == "true" ]; then
  if [ -z "$INT_HOST_SW_DIR" ] ; then
     echo "\$INT_HOST_SW_DIR not defined"
     return 1
  fi
else
  if [ -z "$HOST_SW_DIR" ] ; then
     echo "\$HOST_SW_DIR not defined"
     return 1
  fi
fi

chkconfig portmap on
service portmap restart

grep -v "$VO_SW_DIR" /etc/fstab > /etc/fstab.tmp
if [ $PRIVATE_NETWORK == "true" ]; then
  echo "$INT_HOST_SW_DIR:$BASE_SW_DIR $VO_SW_DIR nfs rw,defaults 0 0" >> /etc/fstab.tmp
else
  echo "$HOST_SW_DIR:$BASE_SW_DIR $VO_SW_DIR nfs rw,defaults 0 0" >> /etc/fstab.tmp
fi
mv -f /etc/fstab.tmp /etc/fstab

mkdir -p $VO_SW_DIR

mount -a

return 0

}
```

A.4 La funzione config_torque_client

```
config_torque_client(){
#------INFN-Perugia's Modification for private network-----
if [ $PRIVATE_NETWORK == "true" ]; then
for x in CE_HOST CE_INT_HOST SE_HOST; do
     if [ "x`eval echo '$'$x`" = "x" ]; then
         echo "\$$x not set"
         return 1
     fi
done
else
for x in CE_HOST SE_HOST; do
     if [ "x`eval echo '$'$x`" = "x" ]; then
echo "\$$x not set"
         return 1
     fi
done
fi
#-----------------end-------------------

# Not used, as torque isn't managed by the relocatable dist
# INSTALL_ROOT=${INSTALL_ROOT:-/opt}

echo "$CE_HOST" > /var/spool/pbs/server_name
```

```
if [ "x`grep pbs_mom /etc/services`" = "x"  ]; then
    echo "pbs_mom     15002/tcp" >> /etc/services
fi

if [ "x`grep pbs_resmon /etc/services | grep tcp`" = "x"  ]; then
    echo "pbs_resmon     15003/tcp" >> /etc/services
fi

if [ "x`grep pbs_resmon /etc/services | grep udp`" = "x"  ]; then
    echo "pbs_resmon     15003/udp" >> /etc/services
fi

cat <<EOF > /etc/ssh/ssh_config
Host *
Protocol 2,1
   RhostsAuthentication yes
   RhostsRSAAuthentication yes
   RSAAuthentication yes
   PasswordAuthentication yes
   EnableSSHKeysign yes
   HostbasedAuthentication yes
EOF

cat << EOF >  /opt/edg/etc/edg-pbs-knownhosts.conf
NODES     = $CE_HOST $SE_HOST
PBSBIN    = /usr/bin
KEYTYPES   = rsa1,rsa,dsa
KNOWNHOSTS = /etc/ssh/ssh_known_hosts

EOF

# workaround for duplicate key entries (Savannah bug 5530)
for hostname in $CE_HOST $SE_HOST; do
   if [ -f /etc/ssh/ssh_known_hosts ];then
      grep -v $hostname /etc/ssh/ssh_known_hosts  > /etc/ssh/ssh_known_hosts.tmp
      /usr/bin/ssh-keyscan -t rsa $hostname >> /etc/ssh/ssh_known_hosts.tmp 2>/dev/null

      if [ $? = 0 ]; then
         mv /etc/ssh/ssh_known_hosts.tmp  /etc/ssh/ssh_known_hosts
      fi
   fi
done

/opt/edg/sbin/edg-pbs-knownhosts

#------INFN-Perugia's Modification for private network-----
if [ $PRIVATE_NETWORK == "true" ]; then
cat << EOF > /var/spool/pbs/mom_priv/config
\$clienthost $CE_INT_HOST
\$clienthost localhost
\$restricted $CE_HOST
\$logevent 255
\$ideal_load 1.6
\$max_load 2.1
EOF
else
cat << EOF > /var/spool/pbs/mom_priv/config
\$clienthost $CE_HOST
\$clienthost localhost
\$restricted $CE_HOST
\$logevent 255
\$ideal_load 1.6
\$max_load 2.1
EOF
fi
#-------------------end----------------
/sbin/chkconfig pbs_mom on
/etc/rc.d/init.d/pbs_mom stop
/etc/rc.d/init.d/pbs_mom start

cron_job edg-pbs-knownhosts root "03 1,7,13,19 * * * /opt/edg/sbin/edg-pbs-knownhosts"

cron_job mom_logs root "33 3 * * * find /var/spool/pbs/mom_logs -mtime +7 -exec gzip -9
```

```
{} \; 2> /dev/null"

return 0
}
```

A.5 La funzione config_bdii

```
config_bdii(){
#
# Configures the BDII.
#
# If SITE_BDII=yes configures as a site BDII otherwise top level
#
# Uses CE_HOST SE HOST RB_HOST and PX_HOST.
#
# These values should be changed the common ones.
#

for x in BDII_HOST; do
    if [ "x‘eval echo ’$’$x‘" = "x" ]; then
        echo "\$$x not set"
        return 1
    fi
done

INSTALL_ROOT=${INSTALL_ROOT:-/opt}

mkdir -p $INSTALL_ROOT/lcg/bdii/var

if ( ! echo "${NODE_TYPE_LIST}" | grep BDII > /dev/null ); then
        for x in BDII_REGIONS; do
            if [ "x‘eval echo ’$’$x‘" = "x" ]; then
                echo "\$$x not set"
                return 1
            fi
        done

        for REGION in $BDII_REGIONS; do
            if [ "x‘eval echo ’$’BDII_${REGION}_URL‘" = "x" ]; then
                echo "\$BDII_${REGION}_URL not set"
                return 1
            fi
        done

        BDII_BIND=mds-vo-name=$SITE_NAME,o=grid
        BDII_AUTO_UPDATE=no

        rm -f $INSTALL_ROOT/lcg/bdii/var/lcg-bdii-update.conf
        for REGION in $BDII_REGIONS; do
            echo "$REGION  ‘eval echo ’$’BDII_${REGION}_URL‘" >> $INSTALL_ROOT/
lcg/bdii/var/lcg-bdii-update.conf
        done

else
        BDII_BIND=mds-vo-name=local,o=grid
        BDII_AUTO_UPDATE=yes
        if [ "x$BDII_HTTP_URL" = "x" ]; then
                echo "\$BDII_HTTP_URL not set"
        return 1
    fi
fi

pass=‘mkpasswd -s 0 2> /dev/null‘ || pass=$RANDOM

cat << EOF > $INSTALL_ROOT/lcg/bdii/var/lcg-bdii.conf
BDII_PORT_READ=2170
BDII_PORTS_WRITE="2171 2172 2173"
BDII_USER=edguser
BDII_BIND=$BDII_BIND
BDII_PASSWD=$pass
BDII_SEARCH_FILTER=’*’
BDII_SEARCH_TIMEOUT=30
```

```
BDII_BREATHE_TIME=60
BDII_AUTO_UPDATE=$BDII_AUTO_UPDATE
BDII_AUTO_MODIFY=no
BDII_DIR=$INSTALL_ROOT/lcg/bdii/
BDII_UPDATE_URL=$BDII_HTTP_URL
BDII_UPDATE_LDIF=http://
SLAPD=/opt/openldap/libexec/slapd
SLAPADD=/opt/openldap/sbin/slapadd

EOF


#------------- Start INFN Perugia mod ---------
if [ $PRIVATE_NETWORK ]; then
cat << EOF >> $INSTALL_ROOT/lcg/bdii/var/lcg-bdii.conf
PRIVATE_NETWORK=true
INT_NET=$INT_NET

EOF
fi
#-------------- End ----------------

cd /etc/rc.d/init.d/
/etc/rc.d/init.d/lcg-bdii stop
/etc/rc.d/init.d/lcg-bdii start
/sbin/chkconfig lcg-bdii on
cd -

# Remove any bdii-update crons
crontab -l | grep -v lcg-bdii-update | crontab -

return 0
}
```

A.6 Lo script lcg-bdii

```
#! /bin/sh
#
# lcg-bdii system startup script
#
# Original version David Groep, NIKHEF
#
# Restructured by L.Field (2004-01-22)
#
# Restructured by L.Field (2005-02-11)
#
# $Id: lcg-bdii,v 1.18 2005/03/31 08:28:13 lfield Exp $
#
# chkconfig: 345 95 5
#
# description: BDII
#
# config: /opt/lcg/var/bdii/lcg-bdii.conf
#
. /etc/rc.d/init.d/functions

if [ "x$BDII_CONF" = "x" ]; then
   BDII_CONF=/opt/lcg/bdii/var/lcg-bdii.conf
fi

if [ -f $BDII_CONF ]; then
    . $BDII_CONF
else
    echo "Configuration file $BDII_CONF not found ..."
    exit 1
fi

case "$1" in
start)

   #Check to see if the process is already started
    if [  -f  ${BDII_DIR}/var/lcg-bdii-update.pid ] ; then
```

```
        failure && echo "BDII already running"
        exit 1
    fi

    for PORT in $BDII_PORTS_WRITE; do

        #Make the database directories.
rm -rf ${BDII_DIR}/var/${PORT}
        mkdir -p ${BDII_DIR}/var/${PORT}

        #Make the slapd config files.
       cat <<EOF > ${BDII_DIR}/var/${PORT}/lcg-bdii-slapd.conf
# Copies of OpenLDAP/Globus schemas.
include  /opt/lcg/bdii/etc/schema/core.schema
include  /opt/lcg/bdii/etc/schema/grid-info-resource.schema
include  /opt/lcg/bdii/etc/schema/grid-info-gram-reporter.schema
# The standard schemas.
include  /opt/lcg/bdii/etc/schema/Glue-CE.schema
include  /opt/lcg/bdii/etc/schema/Glue-CESEBind.schema
include  /opt/lcg/bdii/etc/schema/Glue-SE.schema
include  /opt/lcg/bdii/etc/schema/SiteInfo.schema
schemacheck off

pidfile  ${BDII_DIR}/var/${PORT}/slapd.pid
argsfile ${BDII_DIR}/var/${PORT}/slapd.args

idletimeout     120
sizelimit       80000
timelimit       2400

########################################################################
# ldbm database definitions
########################################################################

database  ldbm
cachesize 30000
dbcachesize 30000000
dbnosync
suffix  "o=grid"
rootdn  "o=grid"
rootpw  ${BDII_PASSWD}
defaultaccess read
directory ${BDII_DIR}/var/${PORT}

EOF

cat <<EOF > /etc/logrotate.d/lcg-bdii
# Logrotation for lcg-bdii.
${BDII_DIR}/var/lcg-bdii.log {
    weekly
    rotate 5
    missingok
    compress
}

EOF


    # Touch the pid files.
    touch  ${BDII_DIR}/var/${PORT}/slapd.pid

    #Enable the firewall
    iptables -t filter -I INPUT -m state --state NEW -m tcp -p tcp --dport ${PORT}
-j ACCEPT

    done

    echo -n "Starting BDII"

#-------------- Start INFN Perugia mod -----------------
if [ $PRIVATE_NETWORK ]; then
    iptables -t nat -I PREROUTING 1 -p tcp -s ! $INT_NET -d `hostname -f` --dport
${BDII_PORT_READ} -j REDIRECT --to-ports ${BDII_PORT_READ}
else
```

```
    iptables -t nat -I PREROUTING 1 -p tcp --dport ${BDII_PORT_READ} -j REDIRECT
--to-ports ${BDII_PORT_READ}
fi
#-------------- End ---------------------

    iptables -t nat -I OUTPUT 1 -p tcp --dport ${BDII_PORT_READ} -d `hostname -f`
-j DNAT --to-destination `host $(hostname) | awk '{print $4}'`:${BDII_PORT_READ}
    rm -f  ${BDII_DIR}/var/lcg-bdii.log
    export LANG=C
    ${BDII_DIR}/sbin/lcg-bdii-update ${BDII_CONF} > ${BDII_DIR}/var/lcg-bdii.log 2>&1 &
    echo $! > ${BDII_DIR}/var/lcg-bdii-update.pid

    if [ $? = 0 ]; then
        success && echo
    else
        failure
        echo
    fi

 ;;

stop )

    #Check to see if the process is already stopped
    if [ ! -f  ${BDII_DIR}/var/lcg-bdii-update.pid ] ; then
failure && echo "Stopping BDII"
exit 1
    fi

    #Remove iptables settings
    for PORT in $BDII_PORTS_WRITE; do
      iptables -t filter -D INPUT -m state --state NEW -m tcp -p tcp --dport ${PORT}
-j ACCEPT
    done
    iptables -t nat -D PREROUTING 1
    iptables -t nat -D OUTPUT 1

    #Try to term all slapd processes.
    su $BDII_USER -c 'killall -15 slapd' 2>/dev/null

    kill -15 `cat ${BDII_DIR}/var/lcg-bdii-update.pid` 2>/dev/null
    rm -f ${BDII_DIR}/var/lcg-bdii-update.pid
    rm -f /etc/logrotate.d/lcg-bdii

    sleep 1

    #Send kill -9 to any remaining slapd processes.
    su $BDII_USER -c 'killall -9 slapd' 2>/dev/null

    if [  "x`ps -ef | grep slapd | grep $BDII_USER`" = "x" ]; then
success  && echo "Stopping BDII"
    else
failure && echo "Stopping BDII"
    fi
     ;;

restart )
    $0 stop
    $0 start
    ;;

* )
   echo "Usage: $0 {start|stop|restart}"
    ;;
esac

exit
```

A.7 Lo script lcg-bdii-update

```
#!/usr/bin/perl -w
#
```

```
# lcg-bdii-update, populates the bdii database.
# Original version David Groep, NIKHEF
# Restructured by L.Field (2004-01-22)
# Peformance enhacements by L.Field (2004-05-24)
# Restructured by L.Field (2005-02-11)
#
# $Id: lcg-bdii-update,v 1.29 2005/04/11 09:38:53 lfield Exp $

use strict;
use POSIX;
use LWP::Simple;
use Sys::Hostname;

#Variables used for the time mesurements.
my $start_time;
my $end_time;
my $elapsed;

#Variables from the bdii configuration file
my $bdii_dir;
my $bdii_host=hostname();
my $bdii_ip=gethostbyname($bdii_host);
my $bdii_passwd;
my $bdii_user;
my $bdii_port_read;
my $bdii_port_write;
my $bdii_ports_write;
my @bdii_ports_write;
my $bdii_bind;
my $bdii_auto_update;
my $bdii_auto_modify;
my $bdii_search_timeout;
my $bdii_breathe_time;
my $bdii_update_url;
my $bdii_update_ldif;
my $bdii_search_filter;
my $slapd;
my $slapadd;

#---INFN Perugia mod---
my $private_network;
my $int_net;
#---end----

#Reads the bdii configuration file.
if (($ARGV[0])){
    open (CONFIG,$ARGV[0]) || die "Couldn't open config file $ARGV[0]\n";
    while (<CONFIG>) {
if (m/^(\w+)=(.*)$/) {
    my $var = $1;
    my $val = $2;
    $val=~s/^\s+//;
    $var =~ tr/A-Z/a-z/;
    $val =~ s/\'//g;
    eval "\$$var = '$val';";
}
    }
    close (CONFIG);

}else{
    print "Usage: $0 <bdii configuration file>\n";
    exit 1;
}

#Main Program Loop
while (1){
$bdii_ports_write=~s/\"//g;
@bdii_ports_write=split (/ /,$bdii_ports_write);

foreach(@bdii_ports_write){
    $bdii_port_write=$_;
    print "\nUpdating DB on port $bdii_port_write\n";

my $newConfig;
```

```
#Updates the file containing the ldif sources.
if ( $bdii_auto_update=~m/^yes/i ){
    eval {
local $SIG{ALRM} = sub { die "GOT TIRED OF WAITING" };
alarm (5);
$newConfig=get($bdii_update_url);
alarm(0);
    };
    if ($@ =~ /GOT TIRED OF WAITING/) {
print "Timed out while downloading configuration.\n";
    }else{
if($newConfig){
    open (CONFIG,">$bdii_dir/var/lcg-bdii-update.conf") || die "Couldn't open file
$ARGV[0]\n";
    print CONFIG $newConfig;
    close (CONFIG);
    print "Updated configuration.\n";
}
    }
}

my @urls;     # An arry of URLS that return ldif.

#Read the ldif urls from the file.
open (CONFIG, "$bdii_dir/var/lcg-bdii-update.conf" ) || die "Couldn't open config file
$bdii_dir/var/lcg-bdii-update.conf\n";
while (<CONFIG>) {
    s/\#.*//;
    if (m/^.*ldap:\/\// || m/^.*file:\/\//){
push @urls, $_;
    }
}
close (CONFIG);

#Ensure that the temp directory exists and is empty
`mkdir -p $bdii_dir/tmp`;
`rm -f $bdii_dir/tmp/*.ldif`;
`rm -f $bdii_dir/tmp/*.log`;
$start_time = `date -u +%s`;

my @pid;      # The pids of the forked processes.
my $pid;      # A pid for one process.
my $region;   # A region name for the ldif source.
my $command;  # A command that will return ldif.

print "Waiting $bdii_search_timeout s for query results.\n\n";

#Loop through for each ldif source
foreach(@urls){

    if(m/ldap:/){
#Split the information from the url.
m|^(\S*)\s+ldap://(.+):([0-9]+)/(.*)|;
$region=$1;
$command="ldapsearch -x -LLL -h $2 -p $3 -b $4 '$bdii_search_filter' >
$bdii_dir/tmp/$region.ldif 2>&1 \n ";
    }

    if(m/file:/){
#Split the information from the url.
m|^([^\s]*)\s+file://(.*)|;
$region=$1;
$command="$2 > $bdii_dir/tmp/$region.ldif";
    }

    # Fork the search.
    unless ($pid=fork){

        # Set our process group to a distinct value.
setpgrp();

        # Eval will kill the process if it times out.
eval {
```

```
    local $SIG{ALRM} = sub { die "GOT TIRED OF WAITING" };
    alarm ($bdii_search_timeout);  #Will call alarm after the timeout.

    if(system("$command")){
print "$region: ";
system("cat $bdii_dir/tmp/$region.ldif")
    }
    alarm(0);           # Cancel the pending alarm if responds.
};

        # This sections is executed if the process times out.
        if ($@ =~ /GOT TIRED OF WAITING/) {
    `rm -f $bdii_dir/tmp/$region.ldif`;
    print "$region: Timed out\n";
    my $PGRP=getpgrp();
    kill (-SIGKILL(), $PGRP);
    print "\n";
    exit 1;
}
exit 0;
    }
    push @pid, $pid;
}

    foreach(@pid){
waitpid($_, 0);
    }

$end_time = `date -u +%s`;
$elapsed = $end_time - $start_time;
print "Time for searches: $elapsed s\n";

my $entry="";     #An ldap entry.
my @ldif;   #An array of ldap entries.

# Reads in the ldif files and puts them into and array.
open (COMMAND, "cat $bdii_dir/tmp/*.ldif  |") || exit 1;
while (<COMMAND>) {
    chomp;
    if(m/^dn:\s/){
$entry="";
    }elsif(m/^\s/){
s/^\s//;
    }else{
$entry.="\n";
    }
    if(m/^\s*$/){
push @ldif, $entry;
    }else{
$entry.=$_;
    }
}
close COMMAND;

#Modify dn.
for(@ldif){
s/,\s+/,/g;
s/dn:(.*)?mds-vo-name=local,o=grid\s*\n/dn:$1o=grid\n/i;
s/dn:(.*)?o=grid\s*\n/dn:$1$bdii_bind\n/;
}

$start_time = `date -u +%s`;
#sort the dns, shortest first so that parent will exist.

my @tmp;
my $dn;

for(@ldif){
    $dn=$_;
    $dn=~s/\n/#/g;
    $dn=~s/#.*$//g;
    push @tmp, sprintf("%04u %s", length($dn), $_);
}
```

```
@ldif = sort @tmp;   # numerical sort
for(@ldif){
    s/^\d\d\d\d //;
    if(m/^(dn: $bdii_bind)/){
$_="";
    }
}

$end_time = `date -u +%s`;
$elapsed = $end_time - $start_time;
print "Time to sort: $elapsed s\n";
$start_time = `date -u +%s`;

    #Stop the DB to kill connections
    my $PGRP =`cat $bdii_dir/var/$bdii_port_write/slapd.pid`;
    if ($PGRP){
my $result=kill (-SIGTERM(), $PGRP);
        while ( $result > 0 ){
            sleep 1;
            $result=kill (-SIGKILL(), $PGRP);
        }
    }
    system("rm -f $bdii_dir/var/$bdii_port_write/dn2id.*
      $bdii_dir/var/$bdii_port_write/id2entry.*  $bdii_dir/var/$bdii_port_write/nextid.*");

    #Add the bind entry.
    open (COMMAND, "| $slapadd -v -f $bdii_dir/var/$bdii_port_write/lcg-bdii-slapd.conf >
/dev/null 2>&1") || die "SLAPD add failed on ldap://$bdii_host:$bdii_port_write";
    print COMMAND "dn: o=grid\nobjectClass: GlueTop\n\n";
    print COMMAND "dn: $bdii_bind\nobjectClass: GlueTop\n\n";
    close COMMAND;
    system("chown -R ${bdii_user}:${bdii_user} ${bdii_dir}");
    system("chmod -R 700 ${bdii_dir}");

    #Add all the entries.
    open (COMMAND, "| $slapadd -c -d 4 -v -f $bdii_dir/var/$bdii_port_write/
lcg-bdii-slapd.conf>/dev/null 2>$bdii_dir/tmp/stderr.log")
|| die "SLAPD add failed on ldap://$bdii_host:$bdii_port_write";
    for(@ldif){
print COMMAND "$_\n";
    }
    close COMMAND;

    #Start the DB.
    system("$slapd -f $bdii_dir/var/$bdii_port_write/lcg-bdii-slapd.conf -h
ldap://$bdii_host:$bdii_port_write -u $bdii_user");

    #Reads the log file and print out the errors.
    undef @tmp;
    my $i=0;
    open (ERROR, "$bdii_dir/tmp/stderr.log") || die "Can't open $bdii_dir/tmp/stderr.log";
    while (<ERROR>) {
if( m/str2entry:/){
    $i = $i + 1;
    $dn = $ldif[$i-1] ;
    $dn =~s/\n.*//g;
    print "$dn\n";
    print "$_\n";
}
if( m/added:/){
    $i = $i + 1;
}
    }
    close ERROR;

    $end_time = `date -u +%s`;
    $elapsed = $end_time - $start_time;
    print "Time to update DB: $elapsed s\n";

    #Updates the file containing the ldif modifications.
    undef $newConfig;
    if ( $bdii_auto_modify=~m/^yes/i ){
eval {
    local $SIG{ALRM} = sub { die "GOT TIRED OF WAITING" };
```

```
    alarm (5);
    $newConfig=get($bdii_update_ldif);
    alarm(0);
};
if ($@ =~ /GOT TIRED OF WAITING/) {
    print "Timed out while downloading ldif.\n";
}else{
    if($newConfig){
open (CONFIG,">$bdii_dir/var/lcg-bdii-mod.ldif") || die "Couldn't open file
$bdii_dir/lcg-bdii-mod.ldif\n";
print CONFIG $newConfig;
close (CONFIG);
print "Updated modification ldif file.\n";
    }
}
    }
    if ( -f "$bdii_dir/var/lcg-bdii-mod.ldif" ){
system("ldapmodify -c -x -f $bdii_dir/var/lcg-bdii-mod.ldif -h $bdii_host -p
$bdii_port_write -D o=grid -w $bdii_passwd > /dev/null 2>&1");
    }
    print "Grabbing port $bdii_port_read for $bdii_port_write\n";

#-------------- Start INFN Perugia mod -----------------

    if ($private_network eq "true"){
system("iptables -t nat -R PREROUTING 1 -p tcp -s ! $int_net -d `hostname -f`
--dport $bdii_port_read -j REDIRECT --to-ports $bdii_port_write");
    }
    else {
system("iptables -t nat -R PREROUTING 1 -p tcp --dport $bdii_port_read -j REDIRECT
--to-ports $bdii_port_write");
    }

#-------------- End ----------------------

    system("iptables -t nat -R OUTPUT 1 -p tcp --dport $bdii_port_read -d $bdii_host -j
DNAT --to-destination `host $bdii_host | awk '{print \$4}'`:$bdii_port_write");
    system("date");
    print "Sleeping for $bdii_breathe_time\n";
    sleep $bdii_breathe_time;
}

}
```

A.8 La funzione config_nat

```
config_nat(){
IPTABLES=/sbin/iptables
CE_IP=`host $CE_HOST | cut -d" " -f4`

if [ "X$CE_IP" = "Xfound:" ] ; then
        echo "Can't resolv $CE_HOST"
        return 1
fi

if [ "X$CE_HOST" = "X`hostname -f`" ] && [ $PRIVATE_NETWORK ]; then

        echo "Cleaning NAT table..."
        $IPTABLES -t nat -F
        echo "Done"

        echo "Enabling ip-forwarding..."
        echo "1" > /proc/sys/net/ipv4/ip_forward
        cat /etc/sysctl.conf | sed 's/net\.ipv4\.ip_forward\ =\ 0/net\.ipv4\.ip_forward\
=\ 1/g' > /tmp/sysctl.conf
        mv /tmp/sysctl.conf /etc/sysctl.conf
        echo "Done"

        echo "Enabling SNAT..."
        $IPTABLES -t nat -A POSTROUTING -s $INT_NET -d ! $INT_NET -j SNAT --to $CE_IP
        echo "Done"
```

```
        echo "Enabling AFS nat..."
        let BASEPORT=4100
        AFSPORT=7001
        for afsclient in $(cat $WN_LIST)
        do
                echo "Adding $afsclient..."
                NODE_IP=‘host $afsclient | cut -d" " -f4‘
                if [ "X$NODE_IP" != "Xfound:" ] ; then
                   let OFFSET=$(echo $NODE_IP | cut -d. -f4)
                   let PORT=$BASEPORT+$OFFSET
                   $IPTABLES -t nat -A PREROUTING -p udp --dport $PORT -j DNAT
--to-destination $NODE_IP:$AFSPORT
                   $IPTABLES -t nat -A POSTROUTING -p udp -s $NODE_IP --sport $AFSPORT
-j SNAT --to-source $CE_IP:$PORT
                else
                   echo "Can’t resolv $afsclient..."
                fi
        done
        echo "Done"

cat << EOF > /etc/init.d/snat
#!/bin/bash
# chkconfig: - 99 01
# description: Enable Nat for WN

start() {
        $IPTABLES -t nat -A POSTROUTING -s $INT_NET -d ! $INT_NET -j SNAT --to $CE_IP
}

stop() {
        $IPTABLES -t nat -D POSTROUTING -s $INT_NET -d ! $INT_NET -j SNAT --to $CE_IP
}

restart() {
        stop
        start
}

case "\$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  restart)
        restart
        ;;
  *)
        echo $"Usage: \$0 {start|stop|restart}"
        exit 1
esac
EOF

chmod 0755 /etc/init.d/snat
chkconfig snat on

cat << EOF > $INSTALL_ROOT/lcg/sbin/afs_nat
#!/bin/sh
IPTABLES=/sbin/iptables
let BASEPORT=4100
AFSPORT=7001
for afsclient in \‘pbsnodes -a | egrep -v ’^ *$| ’\‘ ; do
        NODE_IP=\‘host \$afsclient | cut -d" " -f4\‘
        if [ "X\$NODE_IP" != "Xfound:" ] ; then
                if [ ! iptables -t nat -L | grep ’DNAT’ | grep ’\$NODE_IP:7001’ ]; then
                   let OFFSET=\$(echo \$NODE_IP|cut -d. -f4)
                   let PORT=\$BASEPORT+\$OFFSET
                   \$IPTABLES -t nat -A PREROUTING -p udp --dport \$PORT -j DNAT
--to-destination \$NODE_IP:\$AFSPORT
                   \$IPTABLES -t nat -A POSTROUTING -p udp -s \$NODE_IP --sport
\$AFSPORT -j SNAT --to-source \$CE_IP:\$PORT
                fi
        else
```

```
                echo "Can't resolv \$afsclient..."
        fi
done
EOF
chmod 0755 $INSTALL_ROOT/lcg/sbin/afs_nat
cron_job afs_nat root "*/5 * * * * $INSTALL_ROOT/lcg/sbin/afs_nat"
fi
return 0
}
```

A.9 La configurazione delle code del batch-system

```
#
# Create queues and set their attributes.
#
#
# Create and define queue cert
#
create queue cert
set queue cert queue_type = Execution
set queue cert acl_host_enable = True
set queue cert acl_hosts = gridce.pg.infn.it
set queue cert resources_max.cput = 48:00:00
set queue cert resources_max.walltime = 72:00:00
set queue cert acl_group_enable = True
set queue cert acl_groups = dteam
set queue cert acl_groups += infngrid
set queue cert enabled = True
set queue cert started = True
#
# Create and define queue grid
#
create queue grid
set queue grid queue_type = Execution
set queue grid acl_host_enable = True
set queue grid acl_hosts = gridce.pg.infn.it
set queue grid resources_max.cput = 48:00:00
set queue grid resources_max.walltime = 72:00:00
set queue grid acl_group_enable = True
set queue grid acl_groups = theophys
set queue grid acl_groups += babar
set queue grid acl_groups += gridit
set queue grid enabled = True
set queue grid started = True
#
# Create and define queue na48-infinite
#
create queue na48-infinite
set queue na48-infinite queue_type = Execution
set queue na48-infinite acl_host_enable = True
set queue na48-infinite acl_hosts = na48gridui.pg.infn.it
set queue na48-infinite acl_hosts += na48gridui2.pg.infn.it
set queue na48-infinite resources_max.cput = 768:00:00
set queue na48-infinite resources_max.walltime = 780:00:00
set queue na48-infinite acl_group_enable = True
set queue na48-infinite acl_groups = na48
set queue na48-infinite enabled = True
set queue na48-infinite started = True
#
# Create and define queue cms
#
create queue cms
set queue cms queue_type = Execution
set queue cms acl_host_enable = True
set queue cms acl_hosts = gridce.pg.infn.it
set queue cms resources_max.cput = 48:00:00
set queue cms resources_max.walltime = 120:00:00
set queue cms acl_group_enable = True
set queue cms acl_groups = cms
set queue cms enabled = True
set queue cms started = True
#
```

```
# Create and define queue test
#
create queue test
set queue test queue_type = Execution
set queue test acl_user_enable = True
set queue test acl_users = mariotti
set queue test acl_users += mongardini
set queue test resources_max.cput = 12:00:00
set queue test resources_max.walltime = 24:00:00
set queue test enabled = True
set queue test started = True
#
# Create and define queue cms-short
#
create queue cms-short
set queue cms-short queue_type = Execution
set queue cms-short acl_host_enable = True
set queue cms-short acl_hosts = cmsgridui.pg.infn.it
set queue cms-short acl_hosts += cmsgridui2.pg.infn.it
set queue cms-short resources_max.cput = 00:15:00
set queue cms-short resources_max.walltime = 02:00:00
set queue cms-short acl_group_enable = True
set queue cms-short acl_groups = cms
set queue cms-short enabled = True
set queue cms-short started = True
#
# Create and define queue cms-infinite
#
create queue cms-infinite
set queue cms-infinite queue_type = Execution
set queue cms-infinite acl_host_enable = True
set queue cms-infinite acl_hosts = cmsgridui.pg.infn.it
set queue cms-infinite acl_hosts += cmsgridui2.pg.infn.it
set queue cms-infinite resources_max.cput = 48:00:00
set queue cms-infinite resources_max.walltime = 72:00:00
set queue cms-infinite acl_group_enable = True
set queue cms-infinite acl_groups = cms
set queue cms-infinite enabled = True
set queue cms-infinite started = True
#
# Create and define queue cms-long
#
create queue cms-long
set queue cms-long queue_type = Execution
set queue cms-long acl_host_enable = True
set queue cms-long acl_hosts = cmsgridui.pg.infn.it
set queue cms-long acl_hosts += cmsgridui2.pg.infn.it
set queue cms-long resources_max.cput = 12:00:00
set queue cms-long resources_max.walltime = 24:00:00
set queue cms-long acl_group_enable = True
set queue cms-long acl_groups = cms
set queue cms-long enabled = True
set queue cms-long started = True
#
# Create and define queue theo-infinite
#
create queue theo-infinite
set queue theo-infinite queue_type = Execution
set queue theo-infinite acl_host_enable = True
set queue theo-infinite acl_hosts = theogridui.pg.infn.it
set queue theo-infinite acl_hosts += theogridui2.pg.infn.it
set queue theo-infinite resources_max.cput = 48:00:00
set queue theo-infinite resources_max.walltime = 72:00:00
set queue theo-infinite acl_group_enable = True
set queue theo-infinite acl_groups = theophys
set queue theo-infinite enabled = True
set queue theo-infinite started = True
#
# Create and define queue theo-long
#
create queue theo-long
set queue theo-long queue_type = Execution
set queue theo-long acl_host_enable = True
set queue theo-long acl_hosts = theogridui.pg.infn.it
```

```
set queue theo-long acl_hosts += theogridui2.pg.infn.it
set queue theo-long resources_max.cput = 12:00:00
set queue theo-long resources_max.walltime = 24:00:00
set queue theo-long acl_group_enable = True
set queue theo-long acl_groups = theophys
set queue theo-long enabled = True
set queue theo-long started = True
#
# Create and define queue na48-medium
#
create queue na48-medium
set queue na48-medium queue_type = Execution
set queue na48-medium acl_host_enable = True
set queue na48-medium acl_hosts = na48gridui.pg.infn.it
set queue na48-medium acl_hosts += na48gridui2.pg.infn.it
set queue na48-medium resources_max.cput = 12:00:00
set queue na48-medium resources_max.walltime = 24:00:00
set queue na48-medium acl_group_enable = True
set queue na48-medium acl_groups = na48
set queue na48-medium enabled = True
set queue na48-medium started = True
#
# Create and define queue na48-short
#
create queue na48-short
set queue na48-short queue_type = Execution
set queue na48-short acl_host_enable = True
set queue na48-short acl_hosts = na48gridui.pg.infn.it
set queue na48-short acl_hosts += na48gridui2.pg.infn.it
set queue na48-short resources_max.cput = 01:00:00
set queue na48-short resources_max.walltime = 02:00:00
set queue na48-short acl_group_enable = True
set queue na48-short acl_groups = na48
set queue na48-short enabled = True
set queue na48-short started = True
#
# Create and define queue theo-short
#
create queue theo-short
set queue theo-short queue_type = Execution
set queue theo-short acl_host_enable = True
set queue theo-short acl_hosts = theogridui.pg.infn.it
set queue theo-short acl_hosts += theogridui2.pg.infn.it
set queue theo-short resources_max.cput = 00:15:00
set queue theo-short resources_max.walltime = 02:00:00
set queue theo-short acl_group_enable = True
set queue theo-short acl_groups = theophys
set queue theo-short enabled = True
set queue theo-short started = True
#
# Create and define queue na48-long
#
create queue na48-long
set queue na48-long queue_type = Execution
set queue na48-long acl_host_enable = True
set queue na48-long acl_hosts = na48gridui.pg.infn.it
set queue na48-long acl_hosts += na48gridui2.pg.infn.it
set queue na48-long resources_max.cput = 48:00:00
set queue na48-long resources_max.walltime = 52:00:00
set queue na48-long acl_group_enable = True
set queue na48-long acl_groups = na48
set queue na48-long enabled = True
set queue na48-long started = True
#
# Set server attributes.
#
set server scheduling = True
set server acl_host_enable = False
set server managers = root@gridce.pg.infn.it
set server operators = root@gridce.pg.infn.it
set server default_queue = dteam
set server log_events = 511
set server mail_from = adm
set server query_other_jobs = True
```

```
set server scheduler_iteration = 600
set server node_ping_rate = 300
set server node_check_rate = 600
set server default_node = lcgpro
set server node_pack = False
```

A.10 La configurazione dello scheduler

```
# MAUI configuration example

SERVERHOST              gridce.pg.infn.it
ADMIN1                  root
ADMINHOST               gridce.pg.infn.it
RMCFG[base]             TYPE=PBS
SERVERPORT              40559
SERVERMODE              NORMAL

# Set PBS server polling interval. If you have short
# queues or/and jobs it is worth to set a short interval. (10 seconds)

RMPOLLINTERVAL        00:00:10

# a max. 10 MByte log file in a logical location

LOGFILE               /var/log/maui.log
LOGFILEMAXSIZE        10000000
LOGLEVEL              1

# Set the delay to 1 minute before Maui tries to run a job again,
# in case it failed to run the first time.
# The default value is 1 hour.

DEFERTIME       00:01:00

###### General scheduling policies

#Components weight
CREDWEIGHT 1
FSWEIGHT 1

#Credential weight
CLASSWEIGHT 1
USERWEIGHT 1
GROUPWEIGHT 1

#Fairshare options
FSDECAY                 0.8

#Fairshare weight
FSUSERWEIGHT            5
FSGROUPWEIGHT           10
FSCLASSWEIGHT           20

#Anti-starvation
XFACTORWEIGHT           3
XFWEIGHT                7
XFCAP                   1000000

# MPI oriented
ENABLEMULTIREQJOBS TRUE

# per utilizzare la gestione dell'aquisizione di priorità
# da parte dei job in coda
JOBPRIOACCRUALPOLICY
MAXJOBQUEUEDPERUSERPOLICY ON

# i job che superano il limite di 100 per ogni utente non aumentano
# la loro priorità  anche se restano in coda
MAXJOBQUEUEDPERUSERCOUNT 100

#######Work nodes partitions
```

```
NODECFG[node10.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=2400
NODECFG[node11.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=2400
NODECFG[node12.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=2400
NODECFG[node13.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=1000
NODECFG[node14.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=1000
NODECFG[node15.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=1000
NODECFG[node16.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=1000
NODECFG[node17.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=1000
NODECFG[node18.grid.pg.infn.it] PARTITION=cms-wns PROCSPEED=2400

NODECFG[node60.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node61.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node62.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node63.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node64.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node65.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node66.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node67.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800
NODECFG[node68.grid.pg.infn.it] PARTITION=na48-wns PROCSPEED=2800

NODECFG[node110.grid.pg.infn.it] PARTITION=theo-wns PROCSPEED=2800
NODECFG[node111.grid.pg.infn.it] PARTITION=theo-wns PROCSPEED=2800
NODECFG[node112.grid.pg.infn.it] PARTITION=theo-wns PROCSPEED=2400
NODECFG[node113.grid.pg.infn.it] PARTITION=theo-wns PROCSPEED=2400

# Per CLASS config
CLASSCFG[cert]  PRIORITY=1000
CLASSCFG[test]  PRIORITY=1000
CLASSCFG[grid]  PRIORITY=20 MAXJOB=5,10
CLASSCFG[cms]  PRIORITY=40 MAXJOB=18,27
CLASSCFG[cms-short]  PRIORITY=80
CLASSCFG[cms-long]  PRIORITY=40 MAXJOB=10,20
CLASSCFG[cms-infinite]  PRIORITY=20 MAXJOB=2,5
CLASSCFG[theo-short]  PRIORITY=80
CLASSCFG[theo-long]  PRIORITY=40 MAXJOB=10,20
CLASSCFG[theo-infinite]  PRIORITY=20 MAXJOB=2,5
CLASSCFG[na48-short]  PRIORITY=80
CLASSCFG[na48-medium]  PRIORITY=60 MAXJOB=10,20
CLASSCFG[na48-long]  PRIORITY=40 MAXJOB=10,20
CLASSCFG[na48-infinite]  PRIORITY=20 MAXJOB=2,5

# Per GROUP config
GROUPCFG[cms] MAXJOB=18,27 PLIST=cms-wns:na48-wns:theo-wns PDEF=cms-wns
GROUPCFG[na48] MAXJOB=18,27 PLIST=cms-wns:na48-wns:theo-wns PDEF=na48-wns
GROUPCFG[theophys] MAXJOB=8,20 PLIST=cms-wns:na48-wns:theo-wns PDEF=theo-wns
GROUPCFG[gridit] MAXJOB=5,10
GROUPCFG[babar] MAXJOB=1,2

# Per USER config
USERCFG[DEFAULT]        MAXJOB=10,30
```

Appendice B

Codice di NodeGraph

B.1 Il file select.php

```
<HEAD>
<style type="text/css">
<!--
@import url("style.css");
-->
</style>
<script language="JavaScript">
function today(){
dateObj=new Date();
var day=dateObj.getDate();
var month=dateObj.getMonth()+1;
var year=dateObj.getYear()+1900;
if (day < 10) {
                day = "0"+day;
        }
if (month < 10) {
month = "0"+month;
}
document.graph.enddate.value=(day+"-"+month+"-"+year);
}
function last(days){
        today();
dateObj=new Date();
var nToday=dateObj.getTime();
var nDays = days * 24 * 60 * 60 * 1000;
var n2daysAgo = nToday - nDays;
dateObj.setTime(n2daysAgo);
        var day=dateObj.getDate();
        var month=dateObj.getMonth()+1;
        var year=dateObj.getYear()+1900;
if (day < 10) {
                day = "0"+day;
        }
        if (month < 10) {
                month = "0"+month;
        }
        document.graph.stdate.value=(day+"-"+month+"-"+year);
}
function excludenorm(opt) {
        document.graph.norm[0].disabled=(opt);
        document.graph.norm[1].disabled=(opt);
}
function selall(sm) {
  for(var i=0; i<sm.length; i++) {
    sm.options[i].selected = true;
  }
}
```

```
function docheck(sm,nodes) {
node = nodes.split(',');
for(var i=0; i<=node.length; i++) {
sm.options[node[i]].selected = true;
}
}
function deselall(sm) {
  for(var i=0; i<sm.length; i++) {
    sm.options[i].selected = false;
  }
}
</script>
<script language="JavaScript" src="calendar1.js"></script>
</HEAD>
<BODY onLoad="today();excludenorm(true);">
<FORM name="graph" action="generate.php" method="get" target="graph">
<TABLE WIDTH=100% style="border:1px dashed black">
<TR VALIGN=TOP><TD>
<TABLE>
<TR><TD>
<TABLE align=center>
<TR><TD>
<B>Nodes:</B><BR>
<select size=5 multiple name="nodes[]">
<?php
$connection = mysql_connect("localhost", "root", "") or die("Connection refused: " .
 mysql_error());
mysql_select_db("monitoring") or die("Error on DB selection");
$query = "SELECT fqdn, name, groups FROM nodes";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
$i=0;
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
echo "<option value=".$row['fqdn']."">".$row['name']."</option>";
if (preg_match("/wn/i", $row['groups'])) {
if (!isset($wn)){
$wn=$i;
}else{
$wn=$wn.','.$i;
}
}
if (preg_match("/cms/i", $row['groups'])) {
if (!isset($cms)){
$cms=$i;
}else{
$cms=$cms.','.$i;
}
}
if (preg_match("/na48/i", $row['groups'])) {
if (!isset($na48)){
$na48=$i;
}else{
$na48=$na48.','.$i;
}
}
if (preg_match("/theo/i", $row['groups'])) {
if (!isset($theo)){
$theo=$i;
}else{
$theo=$theo.','.$i;
}
}
if (preg_match("/service/i", $row['groups'])) {
if (!isset($service)){
$service=$i;
}else{
$service=$service.','.$i;
}
}
$i++;
}
?>
</select>
</TD><TD>
<B>Metrics:</B><BR>
```

```
<select size=5 multiple name="metrics[]">
<?php
$connection = mysql_connect("localhost", "root", "") or die("Connection
refused: " . mysql_error());
mysql_select_db("monitoring") or die("Error on DB selection");
$query = "SELECT value, name FROM metrics";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
        echo "<option value=".$row['value']."">".$row['name']."</option>";
        }
?>
    </select>
</TD></TR>
</TABLE>
<TR><TD align=center>
<TABLE>
<TR><TD align=right>
<?php
echo '<input type="button" value="Select CMS" onclick="docheck(this.form.
elements[\'nodes[]\'],\''.$cms.'\');"><br>';
echo '<input type="button" value="Select NA48" onclick="docheck(this.form.
elements[\'nodes[]\'],\''.$na48.'\');"><br>';
echo '<input type="button" value="Select THEO" onclick="docheck(this.form.
elements[\'nodes[]\'],\''.$theo.'\');"><br>';
echo '<input type="button" value="Select WNs" onclick="docheck(this.form.
elements[\'nodes[]\'],\''.$wn.'\');"><br></TD><TD align=left>';
echo '<input type="button" value="Select Service" onclick="docheck(this.form.
elements[\'nodes[]\'],\''.$service.'\');"><br>';
echo '<input type="button" value="Select All" onclick="selall(this.form.
elements[\'nodes[]\']);"><br>';
echo '<input type="button" value="Unselect All" onclick="deselall(this.form.
elements[\'nodes[]\']);"><br>';
?>
</TD></TR>
</TABLE>
</TD></TR>
<TR><TD align=center>
    <B>Date:</B> (DD-MM-YYYY)
</TD></TR>
<TR><TD>
<TABLE align=center>
<TR><TD align=right>
    <LABEL for="stdate">Start: </LABEL>
    <INPUT size=10 type="text" name="stdate" id="stdate"><br>
    <LABEL for="enddate">End: </LABEL>
    <INPUT size=10 type="text" name="enddate" id="enddate">
</td><td align=left>
    <a href="javascript:cal1.popup();"><img src="img/cal.gif" width="16"
height="16" border="0"></a><br>
    <a href="javascript:cal2.popup();"><img src="img/cal.gif" width="16"
height="16" border="0"></a><br><br>
</TD></TR>
</table>
<TABLE align=center><TR><TD align=right>
<input type="button" value="Last Week" onclick="last(7);"></TD><TD align=left>
<input type="button" value="Last Month" onclick="last(30);"></TD></TR>
</TABLE>
<TR><TD>
<TABLE align=center>
<TR><TD>
    <B>View:</B><BR>
    <INPUT type="radio" name="type" value="average"  onclick="excludenorm
(false)">Summary<BR>
    <INPUT type="radio" name="type" checked="true" onclick="excludenorm
(true)" value="noaverage">Detailed<BR>
</TD>
<TD>
    <B>Type:</B><BR>
    <INPUT type="radio" name="norm" checked="true" value="norm"> Normalized<BR>
    <INPUT type="radio" name="norm" value="notnorm"> Not Normalized<BR>
</TD></TR>
</table>
<TR><TD align=center>
    <INPUT type="submit" value="Generate"> <INPUT type="reset">
```

```
</TD></TR>
</TABLE>
<TD>
    <iframe name="graph" src="./how-to.html" width="770" height="350" border="0"
scrolling="auto" style="border: none;"></iframe>
</TD></TR>
</TABLE>
</FORM>
<script language="JavaScript">
var cal1 = new calendar1(document.forms['graph'].elements['stdate']);
cal1.year_scroll = true;
cal1.time_comp = false;
var cal2 = new calendar1(document.forms['graph'].elements['enddate']);
cal2.year_scroll = true;
cal2.time_comp = false;
</script>
</BODY>
```

B.2 Il file generate.php

```
<HTML>
<HEAD>
<style type="text/css">
<!--
@import url("style.css");
-->
</style>
</HEAD>
<BODY>
<?php
$nodes=$_GET['nodes'];
$metrics=$_GET['metrics'];
$stda = explode ("-", $_GET['stdate']);
$endda = explode ("-", $_GET['enddate']);
$type = $_GET['type'];
$norm = $_GET['norm'];
$stdate = $stda[2]."-".$stda[1]."-".$stda[0];
$enddate = $endda[2]."-".$endda[1]."-".$endda[0];

//var check
$error=false;
if (!isset($nodes)){
$error=true;
echo "Select one or more node/s.<BR>";
}
if (!isset($metrics)){
        $error=true;
        echo "Select one or more metric/s.<BR>";
}
if (!(ereg ("([0-9]{4})-([0-9]{2})-([0-9]{2})", $stdate, $regs))) {
$error=true;
echo "Enter the start date in 'DD-MM-YYYY' format.<BR>";
}
if (!(ereg ("([0-9]{4})-([0-9]{2})-([0-9]{2})", $enddate, $regs))) {
$error=true;
echo "Enter the end date in 'DD-MM-YYYY' format.<BR>";
}

if ($stdate > $enddate) {
$error=true;
echo "Start date must be previous to end date.<BR>";
}
if ($error) {
die("Check the field...");
}
//mysql connection
$connection = mysql_connect("localhost", "root", "") or die("Connection refused: " .
 mysql_error());
mysql_select_db("monitoring") or die("Error on DB selection");

if ($type=="average") {
$lnodes="";
```

```
foreach ($nodes as $node) {
$lnodes=$lnodes.$node.";";
}
$lnodes = substr($lnodes, 0,-1);
foreach ($metrics as $metric){
   if ($norm=='norm') {
$query = "SELECT name, norm_type FROM metrics WHERE (value = '$metric')";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
         $mname=$row['name'];
        $norm_type=$row['norm_type'];
}
mysql_free_result($result);
    if ($norm_type != NULL) {
echo "<IMG SRC=\"graph-average.php?norm=$norm_type&nodes=$lnodes&
stdate=$stdate&enddate=$enddate&type=$type&metric=".$metric."_daily\"><BR><BR>";
    }else{
echo "The metric ".$mname." don't support normalized view<br>";
}
   }else{
echo "<IMG SRC=\"graph-average.php?norm=$norm&nodes=$lnodes&
stdate=$stdate&enddate=$enddate&type=$type&metric=".$metric."_daily\"><BR><BR>";
   }
}
}else{
foreach ($nodes as $node) {
foreach ($metrics as $metric){
echo "<IMG SRC=\"graph.php?node=$node&stdate=$stdate&enddate=$enddate&
type=$type&metric=".$metric."_daily\"><BR><BR>";
}
}
}
mysql_close($connection);
?>
</BODY>
</HTML>
```

B.3 Il file graph.php

```
<?php
if (isset($_GET['node'])) {
  $node = $_GET['node'];
}
if (isset($_GET['metric'])) {
  $metric = $_GET['metric'];
}
if (isset($_GET['stdate'])) {
  $stdate = $_GET['stdate'];
}
if (isset($_GET['enddate'])) {
  $enddate = $_GET['enddate'];
}

include ("../jpgraph/src/jpgraph.php");
include ("../jpgraph/src/jpgraph_bar.php");

$nulls=array();

//mysql connection
$connection = mysql_connect("localhost", "root", "") or die("Connection refused: " .
 mysql_error());
mysql_select_db("monitoring") or die("Error on DB selection");
$i=0;
$query = "SELECT value, dt FROM $metric WHERE (node = '$node') AND (TO_DAYS(dt) >=
 TO_DAYS('$stdate')) AND (TO_DAYS(dt) <= TO_DAYS('$enddate')) ORDER BY dt";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
$i++;
if ($row['value']==NULL){
$nulls[]=$i;
}
        $databary[] = $row['value'];
```

```
$dates=explode("-",$row['dt']);
$day[] = $dates[2];
if (!isset($tstdate)){
$tstdate=$dates[2]."/".$dates[1]."/".$dates[0];
}
}
$tenddate=$dates[2]."/".$dates[1]."/".$dates[0];
//end
$met=substr($metric,0,-6);
$query = "SELECT name, ui, scale FROM metrics WHERE value = '$met'";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
        $met_name=$row['name'];
        $ui=$row['ui'];
        $scale=$row['scale'];
}
if (max($databary)>$scale) {
        $scale=max($databary);
}
//graph declaration
$graph = new Graph(740,265,'auto');
$graph->SetScale("textlin",0,$scale);
$graph->xaxis->SetTickLabels($day);
$graph->xaxis->SetTextLabelInterval((count($day)/30)+1);
$graph->title->Set($met_name." for ".$node);
$graph->subtitle->Set("From ".$tstdate." to ".$tenddate);
$graph->xaxis->SetTitle("Days");
$graph->yaxis->SetTitle($ui);
$graph->yaxis->SetTitleMargin(60);

$graph->title->SetFont(FF_FONT1,FS_BOLD);
//end

$b1 = new BarPlot($databary);

//bar setting
$b1->SetColor("blue");
$b1->SetFillColor("cadetblue3");
$b1->SetWidth(0.6);
//end

$graph->Add($b1);

//drow nodes down
$temp=0;
foreach ($nulls as $null){
$band[$temp] = new PlotBand(VERTICAL,BAND_DIAGCROSS,($null-1),$null,'red');
$band[$temp]->ShowFrame(false);
$graph->AddBand($band[$temp]);
$temp++;
}
//end

$graph->Stroke();

mysql_free_result($result);
mysql_close($connection);
?>
```

B.4 Il file graph-average.php

```
<?php
  $nodes = explode(";",$_GET['nodes']);
  $metric = $_GET['metric'];
  $stdate = $_GET['stdate'];
  $enddate = $_GET['enddate'];
  $norm = $_GET['norm'];

include ("../jpgraph/src/jpgraph.php");
include ("../jpgraph/src/jpgraph_bar.php");

$nulls=array();
```

```
//mysql connection
$connection = mysql_connect("localhost", "root", "") or die("Connection refused: " .
 mysql_error());
mysql_select_db("monitoring") or die("Error on DB selection");
$first = true;
$totnproc= array ();
foreach ($nodes as $node){
$query = "SELECT nproc FROM nodes WHERE (fqdn = '$node')";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
$nproc=$row['nproc'];
}
$i=0;
$query = "SELECT value, dt FROM $metric WHERE (node = '$node') AND (TO_DAYS(dt) >=
TO_DAYS('$stdate')) AND (TO_DAYS(dt) <= TO_DAYS('$enddate')) ORDER BY dt";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
if ($first){
if ($row['value']==NULL){
$data[$i]=0;
$count[$i]=0;
$totnode[$i]=0;
$totnproc[$i]=0;
}else{
$data[$i]=$row['value'];
$count[$i]=1;
$totnode[$i]=1;
$totnproc[$i]=$nproc;
}
$dates=explode("-",$row['dt']);
         $day[] = $dates[2];
}else{
if ($row['value']!=NULL){
$data[$i]=$data[$i]+$row['value'];
                $count[$i]++;
$totnode[$i]++;
$totnproc[$i]=$totnproc[$i] + $nproc;
}
}
if (!isset($tstdate)){
                $tstdate=$dates[2]."/".$dates[1]."/".$dates[0];
        }
$i++;
}
$first = false;
}
$tenddate=$dates[2]."/".$dates[1]."/".$dates[0];
$met=substr($metric,0,-6);
$query = "SELECT name, ui, scale, mtype FROM metrics WHERE value = '$met'";
$result = mysql_query($query) or die("Query fallita: " . mysql_error() );
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
$met_name=$row['name'];
$ui=$row['ui'];
$scale=$row['scale'];
$mtype=$row['mtype'];
}
$nulls=array();
for ($i=0;$i<count($data);$i++) {
if ($count[$i] == 0){
$databary[]=0;
$nulls[]=$i;
}else{
if ($mtype == 'sum'){
if ($norm == 'cpu') {
$databary[]=$data[$i]/$totnproc[$i];
}elseif ($norm == 'node'){
$databary[]=$data[$i]/$totnode[$i];
}else{
$databary[]=$data[$i];
}
}else{
$databary[]=$data[$i]/$count[$i];
}
```

```
}
}
if ($norm != 'notnorm') {
$scale=1;
}
if (max($databary)>$scale) {
$scale=max($databary);
}
//end

//graph declaration
$graph = new Graph(740,265,'auto');
if (($norm != 'notnorm') && ($mtype == 'sum')) {
$graph->title->Set("Normalized ".$met_name." on ".$norm." for ".count($nodes)."
 nodes.");
}else{
$graph->title->Set($mtype." of ".$met_name." for ".count($nodes)." nodes.");
}
$graph->SetScale("textlin",0,$scale);
$graph->xaxis->SetTickLabels($day);
$graph->xaxis->SetTextLabelInterval((count($day)/30)+1);
$graph->subtitle->Set("From ".$tstdate." to ".$tenddate);
$graph->xaxis->SetTitle("Days");
$graph->yaxis->SetTitle($ui);
$graph->yaxis->SetTitleMargin(60);
$graph->title->SetFont(FF_FONT1,FS_BOLD);
//end

$b1 = new BarPlot($databary);

//bar setting
$b1->SetColor("blue");
$b1->SetFillColor("cadetblue3");
$b1->SetWidth(0.6);
//end

$graph->Add($b1);

//drow nodes down
$temp=0;
foreach ($nulls as $null){
$band[$temp] = new PlotBand(VERTICAL,BAND_DIAGCROSS,($null-1),$null,'red');
$band[$temp]->ShowFrame(false);
$graph->AddBand($band[$temp]);
$temp++;
}
//end

$graph->Stroke();

mysql_free_result($result);
mysql_close($connection);
?>
```

Bibliografia

1. Andrew S.Tanenbaum, "Distribuited Operating System", Prentice Hall Inc, 1995
2. Ian Foster, Carl Kesselman, "The Grid: Blueprint for New Computing Infrastructure", Morgan Kauffman, 1999
3. Grid computing: "Un prezioso al leato informatico per la ricerca"
   http://www.torinoscienza.it/dossier/apri?obj_id=2601
4. Ian Foster, Carl Kesselman, Steve Tuecke The Anatomy of the Grid Enabling Scalable Virtual Organizations
   http://www.globus.org/research/papers/anatomy.pdf
5. EDG Europ ean DataGrid http://www.eu-datagrid.org
6. EDG Europ ean DataGrid Software Distribution
   http://datagrid.in2p3.fr/distribution
7. LCG Pro ject Homepage:
   http://lcg.web.cern.ch/LCG/
8. F.Gagliardi, B.Jones, M.Reale, S.Burke, Europ ean DataGrid Project, Experiences of deploying a large scale Testbed for e-Science application
   http://hep-proj-grid-tutorials.web.cern.ch/hep-proj-grid-tutorials/presentations/Perf2002Paper.pdf
9. Globus
   http://www.globus.org
   http://www.globus.org/security
10. Introduzione al Globus Toolkit
    http://www.ce.unipr.it/didattica/siselab/Tesine/Globus/
11. INFN Pro duction Grid for Scientific Applications
    http://grid-it.cnaf.infn.it
12. EDG LCFGng FAQ
    http://datagrid.in2p3.fr/distribution/datagrid/wp4/edg-lcfg/documentation/faq.pdf
13. A.Ferraro, A.Caltroni, E.Ferro, Job Submission Tutorial, 29 Settembre 2003
    http://grid-it.cnaf.infn.it/fileadmin/users/job-submission/job_submission.pdf

14. A.Caltroni, A Grid Computing Dictionary, 01 Aprile 2004

15. Andrea Caltroni, Andrea Ferraro, Enrico Ferro, INFN-GRID personal certificates HowTo
http://grid-it.cnaf.infn.it/fileadmin/users/certificates/certificates.pdf

16. CVS User's Guide:
http://grid-deployment.web.cern.ch/grid-deployment/documentation/

17. INFN
http://www.infn.it

18. INFN-GRID
http://www.infn.it/grid

19. INFN-CNAF
http://www.cnaf.infn.it

20. SSH: SSH, The Secure Shell: The Definitive Guide di Daniel J. Barrett, Richard Silverman; Paperback ulterioni informazioni URL:
http://it.wikipedia.org/wiki/SSH

21. PBS URL:
http://www.openpbs.org/

22. MAINFRAME URL:
http://it.wikipedia.org/wiki/Mainframe

23. CALCOLATORI VETTORIALI URL:
http://it.wikipedia.org/wiki/Mainframe

24. TRANSISTOR URL:
http://it.wikipedia.org/wiki/Transistor

25. CLUSTER URL: Distributed Systems: Principles and Paradigms di Andrew S. Tanenbaum, Maarten van Steen

26. MPI: Distributed Systems: Principles and Paradigms di Andrew S. Tanenbaum, Maarten van Steen

27. PVM : Distributed Systems: Principles and Paradigms di Andrew S. Tanenbaum, Maarten van Steen

28. HPF : Distributed Systems: Principles and Paradigms di Andrew S. Tanenbaum, Maarten van Steen

29. TASK FARM : Distributed Systems: Principles and Paradigms di Andrew S. Tanenbaum, Maarten van Steen

30. PIPE LINE : Distributed Systems: Principles and Paradigms di Andrew S. Tanenbaum, Maarten van Steen

31. GLOBAL GRID FORUM URL: http://www.gridforum.org

32. API URL:
http://it.wikipedia.org/wiki/API

33. I-WAY URL:
http://gridcafe.web.

34. UNIX: UNIX for Dummies di John R. Levine, Margaret Levine Young; Paperback ulterioni info URL:
http://it.wikipedia.org/wiki/Unix

35. SETI@HOME URL:
http://setiathome.ssl.berkeley.edu/

36. DAVID ANDERSON URL:
http://setiathome.ssl.berkeley.edu/project.html

37. GLOBUS URL:
http://globus.org

38. GTPL URL:
http://www.unix.globus.org/toolkit/license.html

39. SDK URL:
http://www.google.com

40. EDG URL:
http://eu-datagrid.web.cern.ch/eu-datagrid/

41. EGEE URL:
http://cern.ch/egee

42. CERN URL:
http://cern.ch

43. LHC URL:
http://lhc-new-homepage.web.cern.ch/lhc-new-homepage/

44. PETABYTE URL:
http://it.wikipedia.org/wiki/Byte

45. LCG URL:
http://lcg.web.cern.ch/LCG/

46. INFN URL:
http://infn.it

47. GARR URL:
http://garr.it

48. VO URL:
http://grid-it.cnaf.infn.it

49. LDAP: LDAP Directories Explained: An Introduction and Analysis di Brian Arkills

50. VOMS URL:
http://www.egrid.it/doc/sysadm/voms

51. GSI URL:
http://www-unix.globus.org/toolkit/docs/

52. NASA URL:
http://nasa.gov

53. X.509 URL:
http://www.ietf.org/html.charters/pkix- charter.html

54. SSL Network Security with OpenSSL di John Viega uletriori info:
http://openssl.org

55. XML Learning XML, Second Edition di Erik T. Ray ulterioni info URL:
http://it.wikipedia.org/wiki/XML

56. UDP A TCP/UDP Network Protocols Handbook di Technologies Javvin Technologies

57. HTTP: The Definitive Guide di David Gourley, Brian Totty

58. JDL URL:
http://www.google.com

59. FTP: TCP/IP Bible (Bible) di Rod Scrimger, Paul LaSalle, Mridula Parihar, Meeta Gupta ulterioni informazioni URL:
http://it.wikipedia.org/wiki/FTP

60. IP: TCP/IP Bible (Bible) di Rod Scrimger, Paul LaSalle, Mridula Parihar, Meeta Gupta ulterioni informazioni URL:
http://it.wikipedia.org/wiki/Internet_Protocol

61. PXE URL:
http://www.kegel.com/linux/pxe.html

62. DAG URL:
http://grid-it.cnaf.infn.it/

63. PEM URL:
http://www.openssl.org/docs/

64. NAT URL:
http://it.wikipedia.org/wiki/NAT

65. FIREWALL : Firewall di HENNING MANKELL ulteriori info URL:
http://it.wikipedia.org/wiki/Firewall

66. DHCP : The DHCP Handbook (2nd Edition) di by Ralph Droms, Ted Lemon; Paperback ulterioni informazioni URL:
http://it.wikipedia.org/wiki/DHCP

67. TFTP URL:
http://www.faqs.org/rfcs/rfc1350.html

68. XINETD URL:
http://www.xinetd.org/

69. SYSLINUX URL:
http://syslinux.zytor.com/

70. IPTBLES: Linux iptables Pocket Reference di Gregor N. Purdy ulterioni info URL:
http://www.netfilter.org/

71. BIND: DNS and BIND, Fourth Edition di by Paul Albitz, Cricket Liu; Paperback ulterioni info URL:
http://it.wikipedia.org/wiki/DNS

72. CGI : CGI Programming with Perl di Gunther Birznieks, et al; Paperback ulterioni info URL:
http://it.wikipedia.org/wiki/Cgi

73. NFS : Managing NFS and NIS, 2nd Edition di Hal Stern, et al; Paperback ulterioni informazioni URL:
http://it.wikipedia.org/wiki/File_system

74. TIM BRAY URL:
http://www.textuality.com/

75. CSV URL:
http://www.google.com

76. HTML HTML & XHTML: The Complete Reference di Thomas Powell ulteriori info URL:
http://www.w3c.org

77. YAIM URL:
http://lcg.web.cern.ch/LCG/

78. APT Debian GNU/Linux Bible di Steve Hunger ulteriori info URL:
www.apt-get.org/

79. "L'implementazione ed il deployment di una GRID", Luciano Gaido (INFN-Sezione di Torino)

80. "GRID COMPUTING: DA DOVE VIENE E CHE COSA MANCA PERCHÉ DIVENTI UNA REALTÁ?", Mauro Migliardi

81. "LE VIRTUAL ORGANIZATION NELLA GRID DI LHC", Fabio Spataro

82. "Virtual Organizations e Security", Vincenzo Ciaschini, INFNCNAF 14/2/2002

83. "Installazione e configurazione di un sito Grid di sviluppo", Daniele Leorsini

84. "IMPLEMENTAZIONE DI UN GRID RESOURCE BROKER DEDICATO ALLA VO BABAR E DISTRIBUZIONE DEL SOFTWARE DELL'ESPERIMENTO", Valerio Giacomelli