Università degli Studi di Ferrara

Dottorato di ricerca in Matematica e Informatica

Ciclo XXI

Coordinatore: Prof. Luisa Zanghirati

Studio e realizzazione di un sistema di gestione fault tolerance applicato ad una piattaforma di calcolo distribuito a livello geografico

Settore Scientifico Disciplinare INF/01

Dottorando:
Dott. Veronesi Paolo

Tutore:
Prof. Luppi Eleonora

Anni 2006/2008

Università degli Studi di Ferrara

Dottorato di ricerca in Matematica e Informatica

Ciclo XXI

Coordinatore: Prof. Luisa Zanghirati

Studio e realizzazione di un sistema di gestione fault tolerance applicato ad una piattaforma di calcolo distribuito a livello geografico

Settore Scientifico Disciplinare INF/01

Dottorando:
Dott. Veronesi Paolo

Tutore:
Prof. Luppi Eleonora

Anni 2006/2008

Indice

Introduzione

L'ultimo decennio ha visto un aumento considerevole della potenza di calcolo e delle performance di rete, dovuti principalmente al miglioramento della velocità dei componenti hardware e a software sempre più sofisticato. Parallelamente a questo sviluppo tecnologico sono emerse sfide, in ambito scientifico, ingegneristico ed economico, che non possono essere affrontate con la generazione attuale di super computer, data la complessità computazionale che comportano. Inoltre, la ricerca è sempre più condotta nell'ambito di team e progetti internazionali e richiede la collaborazione tra più discipline, nonché la condivisione di risorse (informatiche, ma anche di banche dati, strumenti e software) che sono distribuite geograficamente su scala internazionale.

Condividere dunque, le risorse di calcolo di diversi istituti, enti e università, anziché creare ex-novo grandi centri di calcolo, non è più solo auspicabile, ma è diventata una vera e propria necessità in vista delle prossime generazioni di esplorazioni scientifiche. Gli esperimenti nella Fisica delle Alte Energie, così come gli studi genomici, biomedici e astrofisici, sono per esempio caratterizzati da enormi quantità di dati e da una comunità di ricercatori internazionali. Sebbene le infrastrutture nazionali siano fondamentali nel fornire connettività locale e risorse ai ricercatori, questi necessitano di rapporti sempre più stretti per collaborare a livello globale. Questo è un requisito sentito prima di tutto a livello Europeo dove la costituzione di una *European Research Area (ERA)* supera le attuali limitazioni nazionali degli stati membri, ed è stata riconosciuta come una delle priorità principali della Unione Europea.

Negli anni recenti una serie di progetti di breve durata, come il progetto EGEE [1] e DEISA [2], hanno sviluppato e fornito l'infrastruttura e i servizi per cercare di rispondere a questo tipo di necessità tramite una piattaforma Grid per il calcolo distribuito. Questi progetti hanno dimostrato di poter cogliere la sfida della

realizzazione di infrastruttura pan-europea al servizio della ricerca, soddisfacendo le necessita di migliaia di utenti in una grande varietà di discipline scientifiche. Il progetto EGEE, nella sua ultima fase, si pone come obiettivo quello di consolidare la propria infrastruttura per favorirne la sostenibilità a medio e lungo termine e di soddisfare le crescenti esigenze di una Grid consolidata in grado di fornire servizi a livelli di affidabilità crescente. Questo requisito apre le frontiere per nuove ricerche nel campo delle tecnologia informatica applicata, che si propongono di studiare e mettere a punto meccanismi di alta affidabilità su piattaforme altamente distribuite.

High Availability e *business critical services* sono due termini che tradizionalmente implicano costi e impegni significativi in tutti gli ambiti, non escluso quello informatico. L'affidabilità di un sistema complesso è funzione di quella dei suoi singoli componenti, sono quindi diversi gli aspetti da prendere in considerazione. Questa tesi elabora il concetto di Service Level Agreement applicandolo in ambito Grid, e propone una serie di metodologie per l'affidabilità dei servizi Grid, mettendone a confronto e illustrandone i relativi risultati sperimentali. L'attività di tesi è svolta negli ambiti dei progetti europei EGEE-I (INFSO-RI-031688), EGEE-II (INFSO-RI-031688), ed EGEE-III (INFSO-RI-222667), in particolare dell'attività SA1 (Service Activity 1), che tra i suoi compiti principali si prefigge di gestire l'infrastruttura Grid di produzione europea basata sul middleware gLite [3], e del progetto speciale INFNGrid [4].

La tesi è così strutturata: nel *primo capitolo* viene introdotto il paradigma del Grid Computing, le necessità che tale tecnologia è chiamata a soddisfare, gli aspetti architetturali, i servizi offerti e i principi di evoluzione e standardizzazione di questa tecnologia.

Nel *Secondo Capitolo* viene presentato il progetto EGEE [1], esempio di grande progetto infrastrutturale Grid nell'ambito del quale è stata svolta l'attività di tesi. Viene descritta l'evoluzione che l'infrastruttura ha avuto nelle diverse fasi del progetto e del suo utilizzo da parte delle comunità scientifiche. Viene infine presentato il progetto EGI_DS [5] che ha lo scopo di pianificare il futuro dell'infrastruttura EGEE per garantire la sua sostenibilità a medio e lungo termine.

Nel *Terzo Capitolo* sono trattati in dettaglio la complessità, i componenti e i principali servizi offerti dal middleware gLite, sul quale si basa l'infrastruttura EGEE. Lo studio architetturale dei servizi offerti dai centri di calcolo che fanno

parte dell'infrastruttura di produzione, è propedeutico per la comprensione delle scelte implementative ed applicative proposte per le tecniche di alta affidabilità illustrate nel capitolo seguente.

Nel *Quarto Capitolo* vengono descritti i temi riguardanti l'affidabilità dei servizi informatici e descritte alcune tecniche generali per ottenere servizi altamente affidabili. Viene definito quindi il concetto di Service Level Agreement, descritta l'architettura proposta dall'Open Grid Forum per implementare la negoziazione di SLA nell'ambito dei sistemi di calcolo distribuiti e presentato, come esempio di SLA, quello utilizzato nell'ambito del progetto EGEE.

Nel *Quinto Capitolo* vengono descritte le soluzioni per l'alta affidabilità elaborate e adottate nell'infrastruttura di produzione europea per diversi servizi critici utilizzati in ambito Grid. Viene analizzato l'impatto che questi contributi hanno avuto nel migliorare sia la disponibilità dei servizi del middleware gLite, che l'affidabilità dei singoli centri di calcolo, utilizzando le metriche di SLA definite dal progetto EGEE. Sono infine introdotte alcune linee guida per l'evoluzione degli attuali sistemi di controllo e supporto di Grid, al fine di agevolare ulteriormente il passaggio ad un modello organizzativo completamente distribuito come previsto dal progetto EGI_DS, e al tempo stesso migliorare l'affidabilità dell'infrastruttura Grid.

Capitolo 1

Grid Computing: architettura e servizi

"*I think there is a world market for maybe five computers.*"
Thomas J. Watson, IBM Chairman

1.1 La sfida del calcolo distribuito

É praticamente impossibile al giorno d'oggi fare ricerca scientifica senza l'ausilio di computer, sono allo studio problemi così complicati che non è più sufficiente solo una lavagna. Spesso un singolo computer, un cluster di computer o anche un super computer, non offrono la necessaria potenza di calcolo per le sfide scientifiche attuali. L'evoluzione tecnologica nel mondo dell'informatica è molto veloce, ma sebbene la potenza dei processori continui a raddoppiare ogni diciotto mesi, fenomeno a cui spesso ci si riferisce con il termine 'Legge di Moore'[1], ancora non raggiungono la potenza di calcolo che la scienza in certi ambiti di ricerca richiede loro.

Come risultato, si ha sempre più spesso a che fare con situazioni nelle quali diventa molto difficile, costoso, e qualche volta impossibile raggiungere obiettivi con l'attuale tecnologia. Si provi a considerare uno scenario in cui un numero consistente di persone, distribuite geograficamente, a diverso titolo voglia condividere delle risorse utili all'esecuzione di un certo lavoro.

Il concetto chiave è rappresentato dalla capacità di condividere risorse distribuite,

[1] La prima legge di Moore è tratta da un'osservazione empirica di Gordon Moore, cofondatore di Intel, pubblicata nel 1965: '*le prestazioni dei processori, e il numero di transistor ad esso relativo, raddoppiano ogni 18 mesi.*'

appartenenti a domini amministrativi distinti ed eterogenei, attraverso interfacce comuni ed interoperabili, in grado di astrarre dalle peculiarità implementative e funzionali delle medesime. Esempi di risorse distribuite di cui le scienze richiedono la condivisione sono i dispositivi per il calcolo e lo stoccaggio di dati, gli archivi digitali, i dati e la strumentazione.

È importante sottolineare che con il termine *condivisione* non si vuole indicare un semplice scambio, ma un accesso, che può anche essere diretto, all'hardware, al software, ai dati e ad altre risorse.

L'applicazione di questi requisiti richiede meccanismi per esprimere politiche, per identificare univocamente utilizzatori e risorse (*authentication*), per caratterizzare le proprietà funzionali e di stato delle risorse, per distribuire il carico di lavoro in modo trasparente all'utente, per determinare se una data operazione è permessa (*authorization*), etc. Per loro natura, gli aspetti riguardanti questo tipo di relazioni possono variare nel tempo, in termini delle risorse coinvolte, sul tipo di accesso a queste e sulle persone a cui è permesso accedere. Occorre quindi uno strumento che in un dato momento le elenchi e descriva.

Quello descritto non è uno scenario virtuale, ma che trova numerose corrispondenze nella realtà: una di queste è rappresentata dalla comunità di migliaia di fisici appartenenti a centinaia di centri di ricerca e università di tutto il mondo impegnati nella progettazione, sviluppo, produzione e analisi dei dati prodotti da quattro grandi detector costruiti presso il CERN (Centro Europeo Ricerca Nucleare) [6] di Ginevra. Questo esempio fornisce una risposta ad una delle due grandi domande che continuano ad animare il dibattito intorno al termine al termine 'Grid': *esiste una tipologia di problemi ben definita per i quali sono necessarie tecnologie Grid?*

1.2 Grid Computing

Il termine Grid è senza dubbio molto inflazionato ed è arrivato ad abbracciare, almeno nella percezione popolare, gran parte dell'informatica, dall'intelligenza artificiale alla rete.

Analogamente al Web [7], Grid in origine è nato esclusivamente nell'ambito scientifico per favorire il calcolo distribuito intensivo e l'accesso flessibile a una grande mole di dati, ma negli ultimi tempi sta ricevendo particolari attenzioni anche nel

campo industriale, commerciale e finanziario. Attualmente la tecnologia Grid si trova ancora in una fase di transizione, dall'utilizzo a fini accademici e di ricerca a più ampie applicazioni nel mondo imprenditoriale. Per esempio, all'interno del Sesto programma quadro di ricerca e sviluppo europeo (FP6), è stato finanziato il progetto BEinGRID[2], composto da 75 partner che danno vita a 18 diversi esperimenti industriali (denominati Business Experiments) che hanno lo scopo di verificare la bontà delle soluzioni basate su GRID in ambito industriale. Obiettivi del progetto sono la diffusione delle tecnologie GRID in Europa ed una validazione di innovativi modelli di business basati sulle nuove applicazioni. Nell'ambito del progetto europeo EGEE [1], oltre alle applicazioni scientifiche, si cerca di estendere i servizi dell'infrastruttura all'industria attraverso la definizione di un programma denominato EGEE Business Associates (EBA) [9] al fine di rendere l'infrastruttura di calcolo distribuito della griglia più accessibile, più efficace e più sicura in un contesto industriale. Nel mondo accademico, così come in quello commerciale, ci sono molteplici definizioni di Grid Computing, ma tutte si concentrano sulla necessità di virtualizzare un insieme distribuito di risorse in modo da ottenere un insieme di risorse di calcolo e di stoccaggio dati scalabile e flessibile, che possano essere utilizzate in maniera efficiente. Un altro problema fondamentale del Grid Computing riguarda la promozione di standard che permettano l'interoperabilità di sistemi eterogenei e l'integrazione modulare dei sistemi esistenti in un sistema più complesso.

Il termine *Grid*, in ambito informatico, è apparso a metà degli anni '90 per descrivere un'infrastruttura di calcolo distribuita per la ricerca e l'ingegneria [10]. Già nel 1969 Len Kleinrock prevedeva uno scenario nel quale il servizio informatico, come già quello elettrico e telefonico, sarebbero stati disponibili in tutte le case e gli uffici. Nel 2000, Foster, Kesselman e Tuecke, ridefinivano la Grid [11] ponendo l'accento su aspetti di condivisione e utilizzo, prendendo atto che il Grid Computing era fortemente legato alla condivisione coordinata di risorse finalizzata alla risoluzione di problemi in *Virtual Organizzations* dinamiche e internazionali. Negli ultimi anni c'è stata una notevole evoluzione delle reti di calcolatori e di molte altre tecnologie che sono migliorate in termini di performance, funzionalità e affidabilità.

Tra le tante definizioni del termine, consideriamo le seguenti tre per completezza nella descrizione degli aspetti caratteristici:

[2]Business Experiments in GRID [8]

- Grid è un insieme geograficamente distribuito di risorse di calcolo, archiviazione e di rete, che fornisce un utilizzo e un accesso ai dati semplificato ad alte prestazioni e qualità di servizio.

- Grid permette la condivisione di applicazioni, risorse e dati in un ambiente aperto e eterogeneo da parte di organizzazioni virtuali. Questa condivisione deve essere necessariamente controllata, cioè i fornitori di risorse e gli utilizzatori devono prima di tutto definire chiaramente e senza ambiguità cosa viene condiviso, chi è autorizzato a condividere e le condizioni che ne regolamentano l'accesso.

- Grid promuove la definizione di interfacce standard per i servizi che devono interoperare per creare un'infrastruttura generale distribuita che soddisfi le esigenze degli utenti e metta a disposizione loro degli strumenti per utilizzarla.

1.3 L'architettura Grid

Sebbene il concetto di Grid sia recente, c'è un considerevole consenso tra gli sviluppatori della tecnologia Grid su come l'architettura dovrebbe essere strutturata [11]. Con '*Architettura Grid*' si intende identificare i componenti fondamentali, descriverne gli scopi e il funzionamento, e specificare come queste componenti interagiscono tra di loro. Questa architettura viene spesso descritta in termini di strati (*layer*), ognuno dei quali è caratterizzato da specifiche funzioni.

In generale, i livelli superiori sono finalizzati alle applicazioni, mentre i livelli inferiori hanno come obiettivo la gestione dell'infrastruttura fisica, come illustrato nella figura 1.1 :

- *Network layer*: assicura la connettività tra le risorse appartenenti alla Grid.

- *Resources layer*: è composto da tutte le risorse, hardware o software, che fanno parte della Grid, come calcolatori, sistemi di stoccaggio dati, cataloghi elettronici, sensori atmosferici, telescopi e tutti gli strumenti che possono essere connessi direttamente alla rete, file, archivi elettronici, etc.

- *Middleware layer*: questo strato di software, così denominato perché si trova nel mezzo tra il sistema operativo, che permette ad un normale calcolatore di

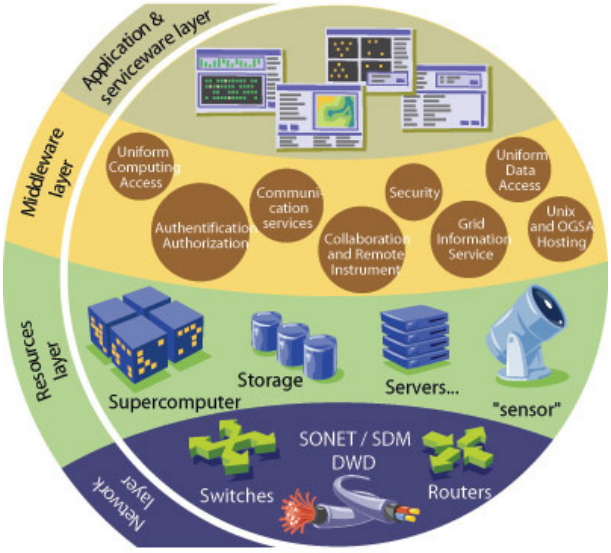


Figura 1.1: Architettura multi layer di Grid

funzionare, e i software applicativi specifici, ha come obiettivo quello di organizzare ed integrare i differenti elementi (server, storage, network, etc.) in un ambiente unico e omogeneo. Lo sviluppo del Middleware è al momento al centro dei diversi progetti di ricerca di Grid nel mondo. Alcune implementazioni hanno superato la fase prototipale e sono correttamente utilizzate in diverse Grid di produzione su scala geografica.

- *Application and Service layer*: include tutte le differenti applicazioni (scientifiche, ingegneristiche, industriali, economiche), i portali e i servizi che permettono l'adattamento delle applicazioni alle funzionalità della Grid. Questo è il livello direttamente accessibile dagli utenti Grid, a cui appartengono alcuni servizi Grid chiamati *serviceware*: un insieme di funzioni generali di amministrazione della Grid usate per esempio per misurare l'utilizzo delle risorse da parte di un utente o di un gruppo di utenti, il costo di questo utilizzo (riferendoci ad un modello di Grid commerciale) e in generale per tenere traccia di chi fornisce le risorse e chi le sta utilizzando. Questi aspetti sono di grande importanza quando si condividono risorse tra istituti, enti, collaborazioni e paesi diversi. Il *serviceware* si trova nello strato più alto perché raccoglie servizi con

i quali gli utenti devono interagire, mentre il *middleware* è uno strato nascosto di cui gli utenti finali non devono preoccuparsi.

1.3.1 Architettura e Servizi

Vi sono altri modi per descrivere l'architettura a strati della Grid (figura 1.2). Per esempio, gli sviluppatori preferiscono racchiudere sotto il termine **Fabric** l'insieme delle infrastrutture fisiche della Grid (calcolatori, rete, etc.), mentre il *Middleware* viene distinto in due tipologie:

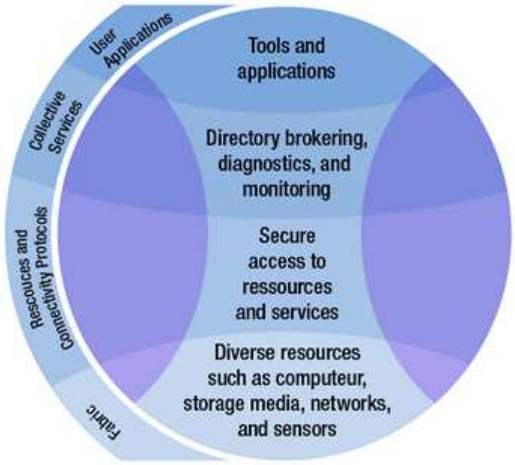


Figura 1.2: Schema dei servizi offerti per ogni livello logico in Grid.

- **Resources and Connectivity Protocols**: indica il middleware che si occupa di tutte le transazioni di rete tra le risorse della Grid. É importante evidenziare che la rete utilizzata è Internet, la stessa usata dal Web a da una miriade di altri servizi. Le risorse che fanno parte di Grid devono riconoscere il traffico per loro rilevante e filtrare il resto. Questo è possibile grazie ai *protocolli di comunicazione*, che permettono alle risorse di dialogare tra loro, rendendo possibile lo scambio di dati, e ai *protocolli di autenticazione* che forniscono un meccanismo sicuro per verificare l'identità sia degli utenti che delle risorse stesse.

- **Collective Services**: Anche questi servizi sono basati su protocolli. *Protocolli di informazione*, che servono per raccogliere dati sulla struttura e sullo stato

delle risorse, e *protocolli di amministrazione*, che negoziano l'accesso alle risorse in modo tale da uniformarlo utilizzando degli standard.

Anche in questo schema, lo strato superiore è quello applicativo (*User Applications*). Per utilizzare la Grid occorre quindi che una applicazione scritta per essere eseguita su un singolo calcolatore, o comunque in un ambiente non Grid, debba essere adattata per utilizzare correttamente i servizi offerti dagli strati inferiori e usare gli opportuni protocolli. Così come è avvenuto l'adattamento di molte applicazioni per l'utilizzo del WWW (*webifying of applications*), anche per l'utilizzo di Grid si rendono necessari degli adattamenti (*gridifying of applications*).

1.4 Requisiti per il Grid Computing

Si possono quindi individuare alcuni aspetti caratterizzanti che distinguono il Grid Computing da altri modelli di calcolo.
Grid è un sistema nel quale:

- sono condivise risorse che non sono soggette ad un controllo centralizzato.
  Una Grid integra e coordina risorse non omogenee e utenti che vivono in diversi paesi, come per esempio PC desktop e computer centrali, unità amministrative diverse nella stessa compagnia o in differenti istituti. La Grid si deve occupare di mascherare i differenti aspetti quali la sicurezza, le condizioni di utilizzo, di pagamento delle singole risorse che vi appartengono.
  Se così non fosse, avremmo a che fare con diversi sistemi computazionali locali, non con la Grid.

- sono usati protocolli e interfacce standard, aperte, e generali.
  Una Grid è costruita partendo da protocolli e interfacce generiche (multi-purpose) che si occupano di regolare aspetti fondamentali come l'autenticazione, l'autorizzazione, la selezione delle risorse (*resource discovery*) e il loro accesso. É molto importante che questi protocolli ed interfacce siano standard e aperte. Se così non fosse, avremmo a che fare con sistemi applicativi specifici (*Application specific system*).

- sono forniti informazioni sulla qualità del servizio.
  Una Grid permette alle risorse che la costituiscono di essere usate in modo

coordinato, fornendo informazioni qualitative relative per esempio ai tempi di risposta, al throughput, alla disponibilità, alla sicurezza, all'allocazione di risorse multiple per incontrare le esigenze degli utilizzatori, in modo tale che l'utilità del sistema complessivo sia significativamente maggiore della somma delle sue parti.

Ovviamente questi aspetti suggeriscono anche altre discussioni sul significato di 'controllo centralizzato', 'protocolli standard e aperti' e 'qualità del servizio'. Alcuni dubbi è ambiguità sono comunque destinati a venire risolti solo con l'evoluzione e l'utilizzo effettivo che verrà fatto della Grid.

1.4.1 Sistemi non Grid

Consideriamo ora alcuni sistemi di calcolo che, in accordo con le caratteristiche sopra descritte, non possono essere qualificati come sistemi Grid.

- Cluster management system come quelli commerciali forniti da Sun (Sun Grid Engine) [12], dalla Platform (Load Sharing Facility) [13] o dalla Altair Engineering (Portable Batch System) [14] possono, quando installati su computer paralleli o in una farm di calcolatori, fornire precise garanzie sulla qualità del servizio e costituiscono una potente sistema di calcolo. Questi sistemi non possono però essere considerati delle Grid, perché il controllo e l'amministrazione delle risorse che li compongono è centralizzato: vi è una conoscenza completa dello stato del sistema e delle richieste degli utilizzatori, nonché un controllo diretto delle singole risorse che lo compongono.

- Su scala differente, anche il WWW non può essere considerato un sistema Grid: è senz'altro aperto, permette l'accesso a risorse distribuite attraverso protocolli *general purpose*, ma non permette l'uso coordinato delle risorse stesse ne fornisce informazioni adeguate sulla qualità del servizio.

- Scheduler su reti locali differenti; sistemi di calcolo distribuito che sfruttano i normali PC desktop quando non utilizzati dai rispettivi proprietari (come Condor [15], Entropia [16] e United Devices [17]); sistemi P2P (Peer to Peer), come Gnutella [18] e BitTorrent [19], che supportano la condivisione di file tra i partecipanti; installazioni di Storage Resource Broker, che permettono

accesso distribuito a risorse di archiviazione. Questi sono esempi di sistemi nei quali i protocolli utilizzati, sebbene spesso molto efficienti, sono a volte troppo specializzati per essere considerati standard aperti.

1.5 Principi per l'evoluzione di Grid

Una delle aree di maggior interesse e che riveste grande importanza, non solo in campo informatico, è quella che riguarda gli standard. Perché la visione della Grid si realizzi, sono necessari protocolli ed interfacce che non solo devono essere generali e aperti, ma anche standard. Saranno questi che permetteranno di stabilire delle politiche di condivisione delle risorse in modo dinamico, altrimenti si otterranno solo Grid incompatibili, ovvero sistemi distribuiti che non possono interagire tra loro.

Negli ultimi anni il concetto di *Service Oriented Architecture* è emerso come il meccanismo standard di progettazione per sistemi di Grid. Una architettura orientata ai servizi *(SOA)* è una forma di sistema distribuito che divide le entità coinvolte fra richiedenti di servizi e fornitori di servizi, che comunicano con lo scambio di messaggi attraverso un meccanismo di trasporto condiviso. La definizione di un componente software che rappresenta una servizio può essere fatta attraverso la forma dei messaggi da esso scambiati. Questo significa che la progettazione della composizione interna di un agente che svolge un determinato servizio può essere rimandata al momento dell'implementazione. Questo scenario si adatta all'architettura di Grid, dove l'indipendenza dalle piattaforme utilizzate, da differenti linguaggi in differenti sistemi operativi è di fondamentale importanza, ma introduce la necessità di gestire lo scambio dei messaggi attraverso internet, dove l'affidabilità e la velocità non possono essere garantite.

L' *Open Grid Forum (OGF)* [20], ed il suo omologo Europeo *OGF-Europe* [21], rappresentano una comunità di utenti, sviluppatori e finanziatori che cercano di spingere l'adozione del Grid computing nel mondo della ricerca e dell'industria. Gli scopi principali sono quelli di definire le specifiche che portino all'adozione diffusa di software standard ed interoperabile e di costruire una comunità internazionale per incentivare lo scambio di idee, esperienze e requisiti. Una delle maggiori conseguenze dell'attività del OGF è la definizione del ***Open Grid Service Architecture (OGSA)*** [22] definito dall'OGSA Working Group.

L'OGSA ha scelto la tecnologia dei Web services[3] come tecnologia di base: i Grid Services sono un'estensione dei Web services, e vengono creati aggiungendo a questi ultimi nuove caratteristiche.

L'insieme di specifiche OGSA propone una associazione fra il paradigma di Grid e una architettura orientata ai servizi definendo un insieme di funzionalità di base e indicazioni che definiscono i concetti chiave del sistema Grid. OGSA definisce sei famiglie di servizi che assieme rendono funzionale un sistema Grid: *Execution Management, Resource Management, Security Management, Self-Management e Information Services.*

Il servizio *Execution Management* ha il compito di creare e gestire unità di elaborazione. Queste attività richiedono la capacità di trovare risorse che soddisfino i requisiti dei processi di computazione e di selezionare la risorsa destinazione in funzione di determinati criteri specificati. Una volta che la risorsa è stata identificata, è necessario che le applicazioni vengano opportunamente configurate per creare un ambiente di lavoro adeguato. A questo punto, l'esecuzione delle unità lavorative può avere inizio, e durante questa fase fino al completamento è necessario che siano monitorate. Le capacità richieste sono state raggruppate in tre principali categorie:

1. servizi di risorsa che modellano l'esecuzione, lo storage e la gestione risorse;
2. la gestione dei job e i servizi di monitoring;
3. servizi di selezione delle risorse che decidono collettivamente dove eseguire una unità di lavoro.

Il *Data Service* riguarda il movimento e la gestione di un insieme di differenti risorse dati, come possono essere file, flat files, data stream e database. Inoltre vengono considerate derivazioni, come dati risultanti da interrogazioni asincrone a database e trasformazioni di dati iniziali. I tipi di attività che devono essere supportate da un servizio di data includono l'accesso remoto, le operazioni di staging, di replicazione, di federazione di insiemi di dati e di gestione e generazione di metadata. Un altra importante funzionalità che questa classe di servizi deve gestire è la

[3]Nel documento che ne descrive l'architettura, la W3C descrive un Web service così: *Un Web service è un componente software capace di fare interagire diversi computer attraverso la rete. Possiede un'interfaccia descritta secondo il WSDL che specifica anche i modi attraverso cui esso può interagire con altri sistemi. L'interazione si basa sullo scambio di messaggi effettuato tramite il protocollo di accesso SOAP e quello di trasporto HTTP.*

trasparenza di accesso, ovvero deve essere fornita una visione astratta dei dati sorgenti che nasconda le peculiarità dovute a differenti protocolli, interfacce, dispositivi e collocazione delle risorse di storage.

Il *Resource Management Service* fornisce differenti attività di gestione, identificate dal corrispondente livello di astrazione al quale si applicano. Il livello più basso è chiamato *resource* e si riferisce alla risorsa fisica e logica. Le risorse sono gestite direttamente attraverso la loro interfaccia nativa di controllo e le attività principali riguardano il monitoring, la configurazione e il discovery. Queste attività inoltre fanno affidamento sulla descrizione data dal modello di informazione, che definisce le proprietà della risorsa, le operazioni, gli eventi e la loro relazione. Il livello intermedio è definito come *infrastructure* e fornisce la gestione di base delle risorse. A questo livello, i differenti approcci alla gestione delle risorse sono astratti e fanno affidamento su un meccanismo standard. Le differenze nel modello informativo sono integrate in una soluzione comune. Il livello più alto è chiamato *OGSA function* e si riferisce alla tipiche attività di gestione di un sistema distribuito.

Il *Security Service* riguarda l'applicazione delle policy di sicurezza all'interno di una organizzazione virtuale. Le policy di sicurezza sono indicazioni che riguardano le entità coinvolte in un sistema di Grid e che specificano restrizioni, in funzione di determinati attributi associati, sull'utilizzo delle risorse. Le regole devono poter essere espresse in termini delle entità coinvolte, risorse e alle caratteristiche dell'ambiente. Queste policy possono riguardare differenti aspetti, come autorizzazione, autenticazione, fiducia, mapping dell'identità, delega e livello di sicurezza. Le capacità funzionali e i corrispondenti servizi di sicurezza sono:

- autenticazione, cioè la verifica della prova di una identità dichiarata;
- il mapping dell'identità, che è la capacità di trasformare una identità da un certo dominino ad un altro;
- l'autorizzazione, che è la risoluzione di una decisione basata su policy precedentemente definite;
- audit e secure logging, che riguarda la produzione di tracce degli eventi inerenti la security;
- privacy, che è la classificazione delle informazioni personali attraverso le policy.

Il *Self Management Service* riguarda l'automazione di compiti quale la configurazione, l'ottimizzazione e la manutenzione necessari per mantenere il servizio Grid funzionale così come definito dai Service Level Agreement (SLA). Queste tipologie di servizi hanno il compito di monitorare il carico e l'utilizzo delle risorse e lo stato dei servizi. Basati su dati di monitoring, i servizi di gestione devono analizzare le condizioni per soddisfare quelle definite dai differenti SLA. Se questo non avviene, i servizi di gestione devono agire sulle priorità degli eventi o aggiungere nuove risorse.

I servizi *Information Service* forniscono i meccanismi usati dagli altri servizi di Grid per conoscere dinamicamente lo stato degli eventi usati per il monitoring, discovery e logging. Questi servizi devono fornire la possibilità di accedere e manipolare le informazioni riguardanti risorse, applicazioni e servizi di Grid in modo efficiente. L'informazione deve essere resa disponibile per essere consumata, sia dal produttore originale o attraverso un intermediario che agisca per conto del produttore. Uno o più consumatori desiderano ottenere l'informazione da uno o più produttori, o uno o più produttori desiderano inviare l'informazione ad uno o più consumatori. I produttori e i consumatori possono essere disaccoppiati e possono non avere nozione gli uni degli altri.

L'OGSA-WG produce differenti tipi di documenti. Ci sono le specifiche, che descrivono precisi requisiti tecnici, includendo anche interfacce e protocolli di un certo componente software, e i profili, che definiscono un insieme di specifiche e come un dato insieme di specifiche debba essere implementato e utilizzato. L'***Open Grid Service Infrastructure (OGSI)*** [23], anch'esso frutto del lavoro del Open Grid Forum, è un documento di specifiche che definisce in modo dettagliato le modalità con cui i servizi Grid con stato (*stateful Grid services*) vengono creati, utilizzati e distrutti in modo coerente con quanto previsto dall'OGSA. Un aspetto molto importante da tenere presente è che parallelamente alla costruzione dell'OGSI, l'architettura dei Web services si è evoluta; questa evoluzione si è manifestata nell'uscita del WSDL 2.0. Per fare in modo che l'OGSI potesse beneficiare dell'evoluzione dell'architettura dei Web service, sono state prodotte delle nuove specifiche [24]: WS-Resource Framework [25] e WS-Notification [26]. Queste due famiglie di specifiche mantengono tutte le caratteristiche e le potenzialità dell'OGSI, adottano una nuova terminologia, ma hanno il vantaggio di separare in più sottogruppi funzioni tra loro diverse che invece si ritrovavano descritte in un unico documento; questa

separazione di compiti migliora decisamente la flessibilità nell'implementazione delle specifiche.

1.6 Grid per necessità differenti

L'obiettivo di una Grid, paragonabile come concetto a quello che oggi rappresenta Internet e il Web, appare ancora lontano dal realizzarsi, ma iniziano ad essere numerosi gli esempi di Grid diverse che troviamo oggi: alcune pubbliche, altre private, nazionali ed internazionali, generiche e altre dedicate a specifici problemi.

- Grid Nazionali
  Questo tipo di Grid è costituita da una collaborazione nazionale in supporto all'analisi e allo studio di complessi problemi scientifici e ingegneristici. L'idea è quella di raggruppare le risorse di calcolo disponibili a livello nazionale in modo da avere a disposizione una potenza di calcolo strategica da utilizzare per problemi di larga scala anche in tempo di crisi, per esempio per simulare scenari in seguito a disastri ambientali, terremoti, attacchi terroristici.

- Grid Private
  Chiamate spesso anche Local-Grid e/o Intra-Grid, possono essere utili in molte istituzioni (ospedali, enti di ricerca). Collegano risorse su scala geografica limitata, hanno un controllo centrale e in molti casi nascono dalla necessità di disporre di una maggiore potenza di calcolo in modo economico collegando risorse già presenti.

- Progetti Grid
  I progetti Grid vengono creati per soddisfare le esigenze di numerose istituti di ricerca e di gruppi distribuiti, hanno comunemente obiettivi a breve e medio termine (collaborazioni scientifiche, progetti ingegneristici). Un progetto viene tipicamente costruito ad hoc e le risorse dei partecipanti al progetto vengono condivise per un certo periodo di tempo e utilizzate per raggiungere uno scopo specifico. Il progetto LCG (*LHC Computing Grid*) è un esempio di questa tipologia di Grid per esperimenti nella Fisica delle Alte Energie (*High Energy Physics*).

- Grid @home

  Queste Grid sono per chiunque voglia donare un po' della potenza di calcolo del proprio computer collegato ad Internet. La maggiore motivazione delle persone coinvolte è generalmente rappresenta dalla curiosità verso lo scopo del progetto, dove la possibilità (molto remota) di un ritorno economico o di fama sono solo un dettaglio. Per esempio il progetto SETI@home prevede un viaggio gratis a Puerto Rico, a visitare il radio telescopio utilizzato, per il proprietario del computer che analizzerà per primo un messaggio proveniente dallo spazio attribuito ad una forma di vita intelligente.

- Peer-to-Peer Grid

  La tecnologia P2P dipende dalla volontà delle persone di condividere dati presenti nei loro computer. Il nome suggerisce che non vi è un controllo centralizzato, e l'idea è quella di dare qualcosa in cambio di quello che si riceve.

- Grid commerciali

  Questa tipologia prevede la condivisione di risorse su base commerciale e non sulla base di donazioni o mutui interessi. Compagnie e organizzazioni usano risorse di terzi e i proprietari di queste sono pagati "a consumo" per la potenza di calcolo e la capacità di archiviazione che mettono a disposizione.

Capitolo 2

Il Progetto EGEE

"*There is no reason for any individual to have a computer in their home.*"
Ken H. Olsen, Founder of Digital Equipment Corporation

2.1 EGEE: Enabling Grids for E-Science in Europe

EGEE[1] è un esempio di grande progetto infrastrutturale Grid. Finanziato inizialmente dalla Commissione Europea nel 2004, strutturato in tre fasi (EGEE-I/II/III), ha l'obiettivo di fornire ai ricercatori una infrastruttura Grid di produzione in grado di soddisfare i requisiti delle applicazioni delle varie comunità scientifiche: Scienze della Terra, Fisica delle Alte Energie, biomedica, astrofisica, chimica computazionale, etc.

La prima fase del progetto (2004/2006) ha avuto come obiettivo principale quello di integrare le griglie nazionali, regionali e tematiche di calcolo e di dati per creare un'infrastruttura europea abilitata alla tecnologia Grid. Durante la seconda fase (2006/2008), è proseguita la crescita dell'infrastruttura e sono state gettate le basi per fornire un servizio continuo in grado di garantire la qualità dell'infrastruttura di produzione. Parte importante e integrale dell'attuale infrastruttura sono le risorse del progetto *Worldwide LHC Computing Grid (WLCG)* [27] il cui scopo è la messa a punto dell'infrastruttura di calcolo per gli esperimenti di fisica *LHC (Large Hadron Collider)* [28]. L'ammontare complessivo delle risorse, la gamma di funzionalità fornite e l'uniformità dei metodi di accesso alle risorse messi in campo dall'infrastruttura EGEE non sono neppure comparabili ai classici servizi erogati dai singoli centri di calcolo che sono parte di EGEE. In EGEE molteplici istituti di

ricerca, centri di calcolo e università si sono federati allo scopo di formare una unica piattaforma per le scienze che richiedono calcoli intensivi (*e-Science*).

Oggi il progetto è giunto alla sua terza fase (2008/2010) e quella esistente è la più grande infrastruttura Grid multi disciplinare del mondo, cui partecipano più di 140 istituti e consiste approssimativamente di 260 centri di calcolo in 50 Nazioni, fornendo ai suoi 10000 utenti accesso a 80000 CPU e 20 PB di storage. Visto il numero crescente delle comunità scientifiche coinvolte, l'aumento delle applicazioni provenienti dai più disparati settori, dalla geologia alla chimica computazionale, che vengono integrate nell'infrastruttura, e la promozione dell'infrastruttura stessa attraverso attività come l'EGEE Business Forum [9], questi numeri sono destinati ad aumentare nei prossimi anni

Lo scopo della terza fase è quello di:

- espandere e ottimizzare l'attuale infrastruttura di Grid Europea con un supporto continuo alla stessa, e con un aumento delle comunità scientifiche e delle risorse coinvolte.

- agevolare la migrazione da un modello organizzativo basato su un progetto di ricerca ad un nuovo modello sostenibile, basato sulla federazione delle Grid multi disciplinari nazionali, denominate *National Grid Initiative (NGI)* [29].

Questa infrastruttura è a disposizione degli scienziati di tutto il mondo indipendentemente dalla loro posizione geografica, in modo continuativo, ed è basata sul servizio di connettività di rete offerto dal progetto GEANT [30].

L'infrastruttura europea è basata sul middleware sviluppato nel contesto del progetto EGEE e denominato gLite [3], la cui prima versione è stata rilasciata nel Marzo 2005 e che comprende i seguenti servizi principali: Workload Management, Data Management, Information and Monitoring, Virtual Organizzation Membership Service (VOMS), User Interface, che verranno descritti nel capitolo successivo. Il middleware gLite è basato su una architettura orientata ai servizi con lo scopo di facilitarne l'interoperabilità e l'adattamento a standard futuri.

2.2 L'infrastruttura EGEE

L'infrastruttura Grid di EGEE consiste in un insieme di servizi offerti dal middleware gLite installati su risorse di calcolo e archiviazione distribuite geograficamente, e da

un insieme di servizi e strutture di supporto utilizzate per gestirli. L'organizzazione dell'infrastruttura è organizzata in tre parti:

- ***L'infrastruttura di Produzione*** è la parte di infrastruttura che supporta tutte le comunità scientifiche, composta dalle risorse di calcolo e archiviazione di circa 260 centri geograficamente distribuiti. È usata quotidianamente da diverse migliaia di ricercatori raggruppati in più di 200 Virtual Organizzations. L'infrastruttura è gestita e controllata per garantirne la stabilità e la continuità operativa dei servizi offerti.

- Il ***Pre Production Service (PPS)*** è pensato per fornire accesso ai servizi Grid in via di sviluppo agli utenti interessati, in modo che nuove funzionalità possano essere testate e valutate. Sul PPS viene estesa l'attività di certificazione del middleware, utile nella valutazione delle procedure di installazione, aggiornamento e configurazione da applicare poi nell'ambiente di produzione.

- ***EGEE Network Operation Center (ENOC)*** si occupa del coordinamento e gestione delle procedure relative alla rete tra EGEE e i fornitori di rete (GEANT2 e NRENs).

Nei paragrafi seguenti viene valutata l'evoluzione dell'infrastruttura di produzione e del suo utilizzo nell'arco delle diverse fasi del progetto EGEE. L'analisi si basa su un insieme di metriche definite dal progetto stesso [31] utilizzate per misurare qualitativamente e quantitativamente lo stato dell'infrastruttura ed il suo utilizzo. Occorre precisare che la definizione stessa delle metriche viene concordata da tutti i partner del progetto e soggetta a periodiche verifiche e revisioni in modo tale da utilizzare le metriche più adatte con le quali valutare i progressi dell'infrastruttura.

2.2.1 Evoluzione delle risorse dell'infrastruttura

All'interno del progetto EGEE, la *Service Activity 1 (SA1)* è quell'attività il cui obiettivo principale è di gestire l'infrastruttura di produzione EGEE fornendo un alta qualità di servizio al gruppo delle applicazioni. Le procedure e le applicazioni utilizzate sono costantemente valutate e riviste per poterle rendere sostenibili in vista della transizione al nuovo modello organizzativo, discusso in seguito. Questa transizione prenderà quindi il via dalle strutture presenti, che hanno caratterizzato

tutte le fasi del progetto EGEE: i *Regional Operations Center* (**ROC**), che forniscono supporto di primo livello nell'installazione e nell'amministrazione dei siti, e che attraverso turni settimanali, iniziati nel Novembre del 2004, si occupano di monitorare lo stato dei servizi generali della Grid. Il gruppo si occupa inoltre di definire le procedure per la creazione, amministrazione e supporto di nuove VO. Ad oggi sono circa un centinaio le Virtual Organizzation definite[1] che hanno accesso all'infrastruttura.

L'infrastruttura di produzione di EGEE è in continua crescita ed espansione, in tutti i suoi aspetti: nel numero dei centri di calcolo che fanno parte dell'infrastruttura, nella potenza di calcolo disponibile, nelle capacità di stoccaggio, ma ancora più significativamente nell'utilizzo che ne viene fatto.

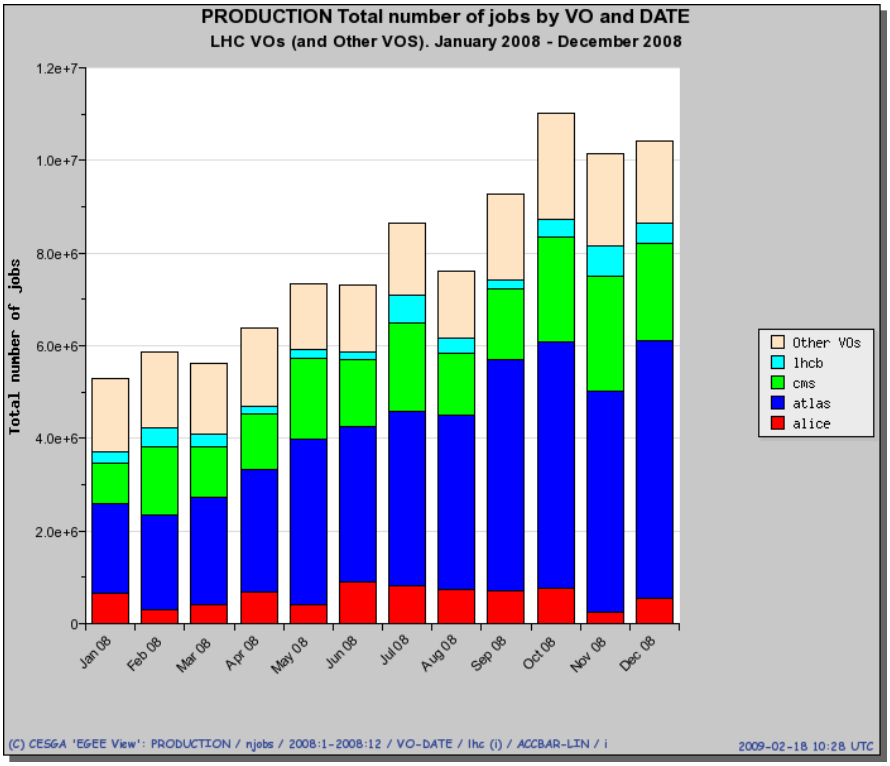


Figura 2.1: Numero totale di Job per VO, aggregati mensilmente, nel 2008. Fonte: [33]

Se nel Settembre 2007 si registrava una media di 100000 job alla settimana, che già rappresentava un aumento del 100% rispetto al Gennaio 2007 [34], nel Ottobre

[1]Per un elenco aggiornato delle VO definite in EGEE si veda la sezione VO del *CIC Operation Portal* [32]

2008 sono stati eseguiti più di undici milioni di job con una media superiore a 350000 job al giorno, come illustrato in figura 2.1).
La figura 2.2 mostra il continuo aumento del numero dei centri di calcolo, che fanno parte dell'infrastruttura, e il costante aumento del numero degli *slot* disponibili per le VO. Per lo storico dei dati è stata usata come fonte l'applicazione Gstat [35] che fornisce dati e grafici recuperando le informazioni del sistema informativo di Grid, descritto nella sezione 3.3.1.

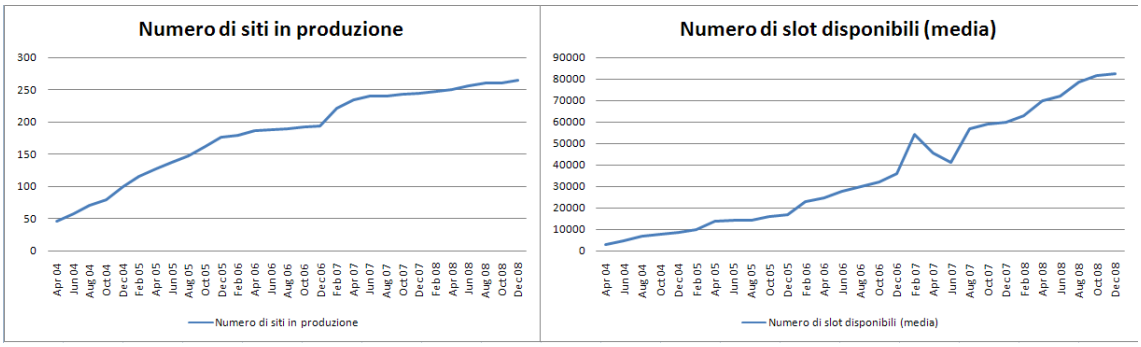


Figura 2.2: Evoluzione del numero di siti e di slot nell'infrastruttura EGEE

Occorre sottolineare che con il termine slot viene spesso indicato, impropriamente, il numero di CPU o altre volte il numero di core disponibili: impropriamente perché al momento non è possibile stabilire con certezza se un sito pubblica il numero di CPU, core o slot, poiché il sistema informativo Grid non è in grado di rappresentare adeguatamente le configurazioni dello scheduler locale ad un sito che fa parte dell'infrastuttura.
Di seguito si assumeranno le seguenti convenzioni:

- Con numero di CPU si indicherà il numero di CPU fisiche;
- Il numero di Core indicherà il numero di CPU logiche, che generalmente corrisponde al numero di CPU riconosciute dal sistema operativo. Per esempio, nel caso di server dual CPU quad core, il numero totale di core è otto.
- Il numero di Slot indicherà il numero totale di job che possono essere in esecuzione contemporaneamente, questo valore può essere anche superiore al numero di core disponibili su una macchina: questa tecnica, denominata *overbooking*, è utilizzata per sfruttare al massimo i cicli di clock di una CPU che sono inutilizzati durate le fasi di Input/Output.

Il numero di slot (o di CPU o core) è una proprietà dinamica nel tempo e da solo non è in grado di rappresentare totalmente la potenza di calcolo disponibile in un sito o nell'intera infrastruttura, per le attuali limitazioni del middleware sopra citate.
La capacità di stoccaggio dati disponibile ed utilizzata, ha avuto un incremento meno evidente rispetto a quello osservato per la capacità di calcolo. Tale incremento è previsto essere comunque continuo e particolarmente significativo nel momento in cui il progetto LHC inizierà la presa dati. Risulta essere molto complesso recuperare dati dettagliati riguardo alla capacità disponibile ed usata; diverse attività sono in corso in questo campo, soprattutto nell'ambito WLCG [36] [37] . Per esempio il sistema informativo non permette di distinguere tra capacità di stoccaggio su disco o su nastro magnetico. Nelle figure 2.3 e 2.4 viene mostrato l'andamento della capacità di stoccaggio disponibile ed utilizzata, aggregato per regione, per il periodo Gennaio/Ottobre 2007 così come evidenziato in [34].

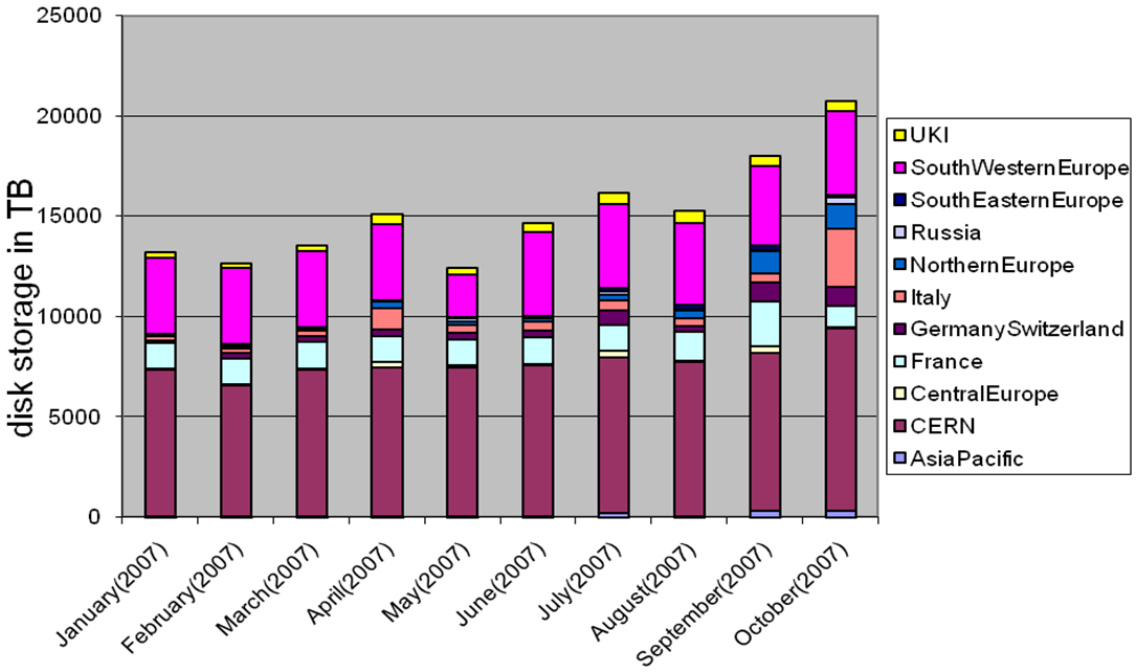


Figura 2.3: Evoluzione della capacità di stoccaggio disponibile. Fonte [34]

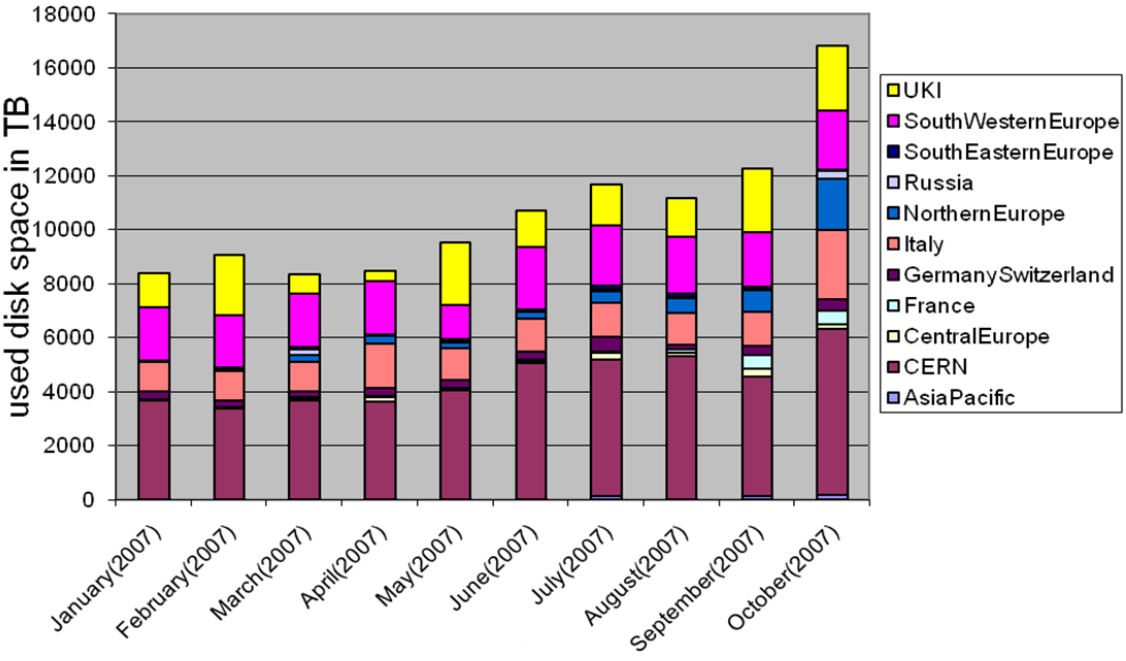


Figura 2.4: Evoluzione della capacità di stoccaggio utilizzata. Fonte [34]

2.3 Il sistema di supporto in EGEE

Un sistema di supporto efficiente per gli utenti Grid rappresenta una sfida importante vista la natura distribuita del sistema. Inoltre la varietà delle applicazioni e delle comunità scientifiche coinvolte aggiunge complessità a questa sfida. Le richieste di supporto arrivano da diverse tipologie di utenti: dagli utenti novizi agli utilizzatori di specifiche applicazioni, dagli amministratori di sistema, dai responsabili del controllo dell'infrastruttura. Attraverso il sistema denominato *Global Grid User Support (GGUS)* [38], il progetto EGEE fornisce un portale di accesso unico nel quale gli utenti possono richiedere e trovare assistenza per le loro quotidiane attività. Tramite GGUS, gli esperti e il personale di supporto possono quindi tenere traccia dei problemi che vengono aperti. Vanno evidenzianti due importanti gruppi di utenti che si occupano di segnalare problemi aprendo ticket seguendo poi la loro evoluzione:

1. il gruppo *CIC-on-Duty (COD)*. Fanno parte di questo gruppo esperti delegati da ciascun ROC che in turni organizzati su base settimanale si occupano

delle cosiddette Grid Operations, ovvero mediante l'utilizzo di diversi sistemi di monitoring, provvedono a diagnosticare le anomalie evidenziate nell'intera infrastruttura Grid e a segnalarle, aprendo ticket che verranno assegnati all'unità di supporto più appropriata.

2. ROC-on-Duty. Analogamente ai COD, si tratta di un gruppo di esperti che si prende carico delle Operations e delle problematiche relative al proprio ROC, fornendo in particolare assistenza agli amministratori di sistema dei siti che fanno parte della relativa regione.

Questo modello può essere definito come un modello di supporto regionale con un coordinamento centrale, come illustrato in figura 2.5.

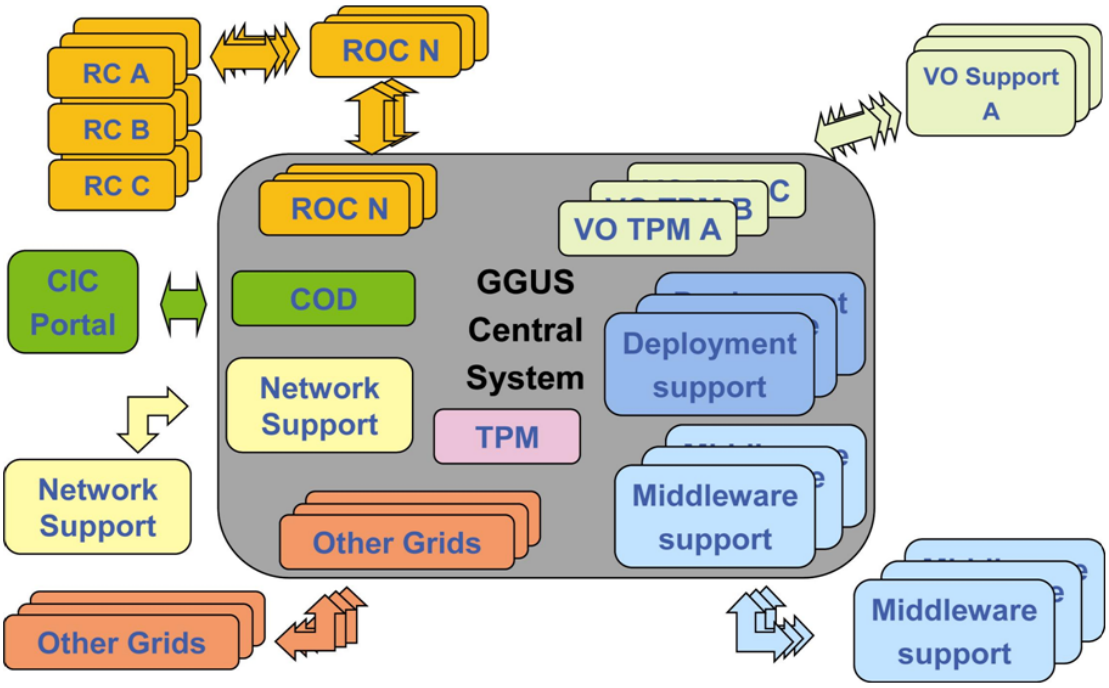


Figura 2.5: Modello del sistema di Supporto in EGEE. Fonte [39]

In accordo con questo modello, gli utenti possono scegliere di sottomettere le loro richieste nel portale GGUS o, in quello del loro ROC di appartenenza, oppure utilizzando il sistema di supporto che la loro VO di appartenenza gli ha messo a disposizione. La richiesta sottomessa genera un cosiddetto *ticket* che viene preso in carico dal sistema, smistato e assegnato ai *gruppi di supporto* più appropriati. Per

gruppo di supporto si intende un insieme di esperti specializzati nel fornire supporto a specifiche problematiche. Per esempio, nel caso un problema generico riguardi una particolare VO, il ticket verrà assegnato al gruppo di supporto di quella VO. Nel caso poi venga evidenziato che il problema affligge una specifica applicazione di quella VO, il ticket verrà assegnato al gruppo che si occupa di supportare quella applicazione. Se invece viene sollevato un problema in uno dei siti partecipanti all'infrastruttura, il ticket viene assegnato ai sistemisti di quel sito. Una dettagliata descrizione del modello di supporto è disponibile in [40].
Il grafico in figura 2.6 riporta il numero di ticket assegnati alle principali unità di supporto di GGUS nel periodo Maggio/Agosto 2008.

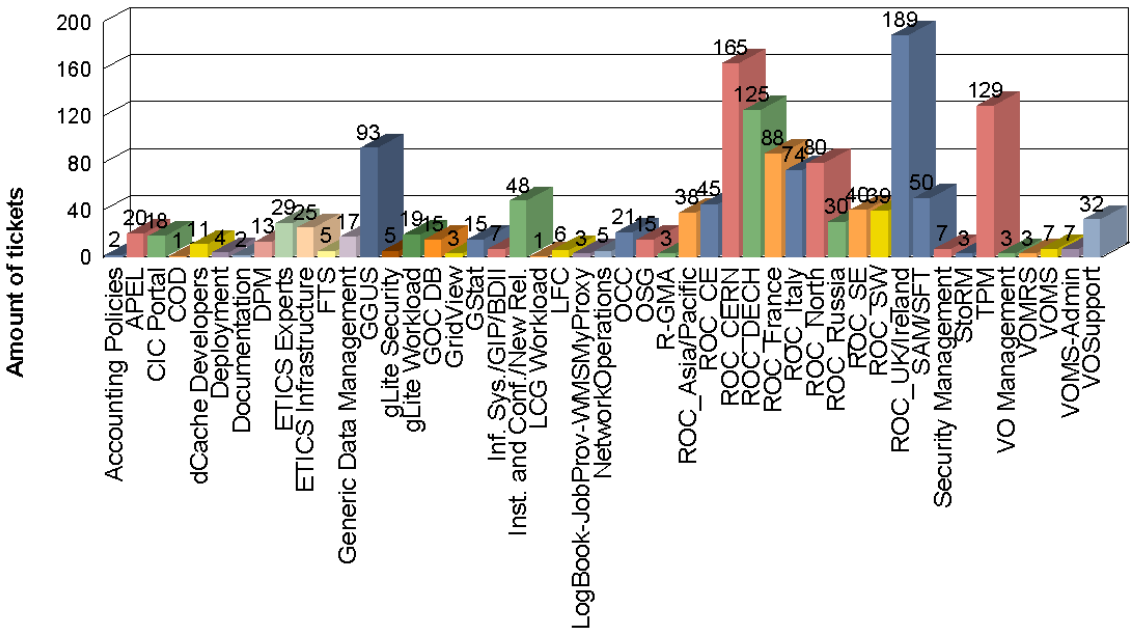


Figura 2.6: Numero di ticket per unità di supporto in GGUS, Maggio/Agosto 2008. Fonte [39]

I ticket relativi a problemi che riguardano i siti Grid sono aggregati e assegnati ai gruppi di supporto dei ROC di cui fanno parte. In figura 2.7, per lo stesso periodo, è riportato il numero di ticket sottomessi per VO.

Tra le diverse metriche prese in considerazione, si valutano anche:

- il numero di ticket in stato 'UNSOLVED'. Rientrano in questa categoria i ticket relativi ad un baco di una applicazione o di uno dei servizi del middleware

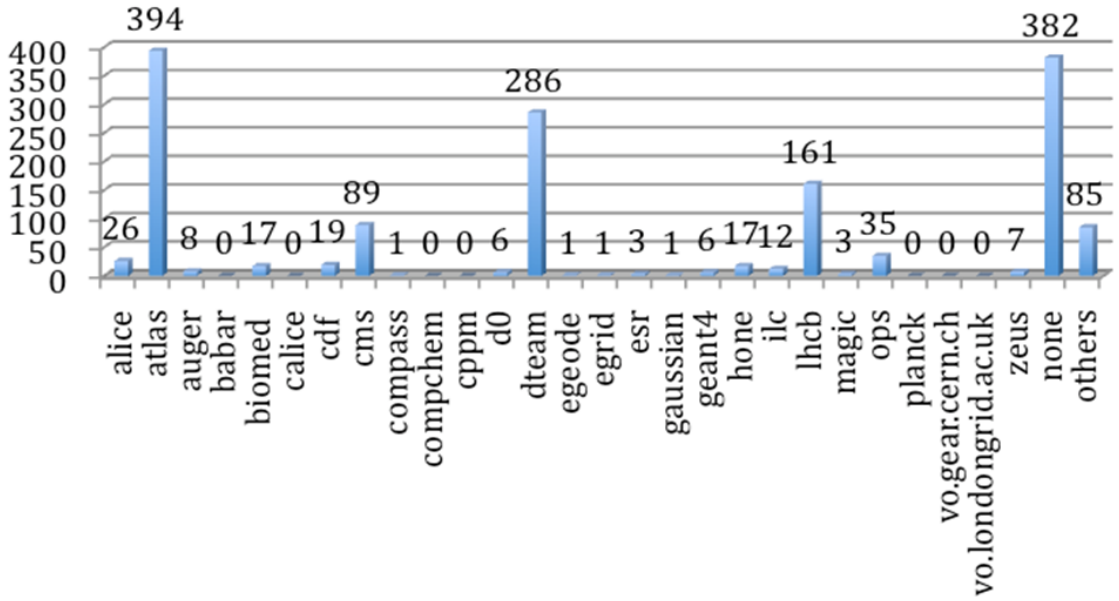


Figura 2.7: Numero di ticket per VO, Maggio/Agosto 2008. Fonte [39]

e rimangono in questo stato fino a che il baco non viene risolto e la soluzione resa disponibile in produzione.

- i tempi che intercorrono tra quando il ticket viene aperto e quando viene preso in carico da un gruppo di supporto (*assign time*) e in generale il tempo necessario affinché il ticket venga risolto (*solution time*).

- il numero di ticket assegnati al gruppo di supporto sbagliato. La complessità dell'infrastruttura e delle applicazioni utilizzate incide sulla difficoltà di fare diagnosi precise sulla natura di un problema, con conseguente possibilità di assegnazione del ticket ad una unità di supporto non appropriata.

Tra le procedure previste dal complesso sistema di supporto vi è anche quella della valutazione dei ticket che presentano metriche anomale, come lunghi periodi di mancato aggiornamento del ticket stesso: questi casi sono sollevati e discussi settimanalmente a livello di progetto e di regione, in modo da intraprendere le azioni più opportune alla loro soluzione.

2.4 Futura organizzazione di EGI

L'architettura Grid descritta fino a questo punto è una realtà che si è consolidata attraverso una serie di progetti iniziati fin dal 2001 con il progetto *European DataGrid (EDG)* [41] e proseguita con le diverse fasi del progetto EGEE. Parallelamente a questi progetti, altre attività e progetti Grid a livello nazionale hanno contribuito all'infrastruttura Grid europea oggi disponibile. Inoltre numerose discipline scientifiche, oltre a quelle legate alla fisica delle particelle, la utilizzano e contribuiscono al suo sviluppo. Tutti questi progetti hanno generalmente in comune tre aspetti:

- le infrastrutture di calcolo sono finanziate a livello nazionale e dai partner dei progetti;
- una parte dei contributi proviene dalla Commissione Europea;
- un'altra parte di contributi proviene dai singoli Stati.

La ricerca scientifica non si ferma ai confini nazionali. C'è una richiesta pressante da parte della Comunità Europea affinché l'infrastruttura per la e-Science europea raggiunga una sostenibilità a lungo termine, indipendente da cicli di finanziamento che essa può fornire, dando vita ad una **European Grid Initiative (EGI)** che la sostenga. Per questo motivo EGI è stata pensata per essere costruita su un numero limitato di partner nazionali, che nascono come aggregazione di tutte le iniziative nate in un singolo paese, e vanno sotto il nome di **National Grid Initiatives (NGI)**. Nonostante debbano essere fatti tutti gli sforzi possibili per assicurare la continuità operativa dei servizi a disposizione degli utenti europei, EGI non è pensata essere una semplice continuazione di EGEE o di altri progetti Grid infrastrutturali, dove gli accordi diretti tra i fornitori di risorse e gli enti di ricerca definiscono l'utilizzo e l'allocazione delle risorse. Il concetto di EGI si basa sulle singole NGI che possono rappresentare sia le locali comunità scientifiche che i fornitori delle risorse.

Facendo riferimento alla figura 2.8, i componenti base di EGI saranno quindi le NGI, che avranno le seguenti caratteristiche:

- ogni NGI deve essere l'unica entità riconosciuta a livello nazionale, rappresentando l'unico punto di riferimento per tutte le istituzioni e le comunità nazionali all'infrastruttura Grid;

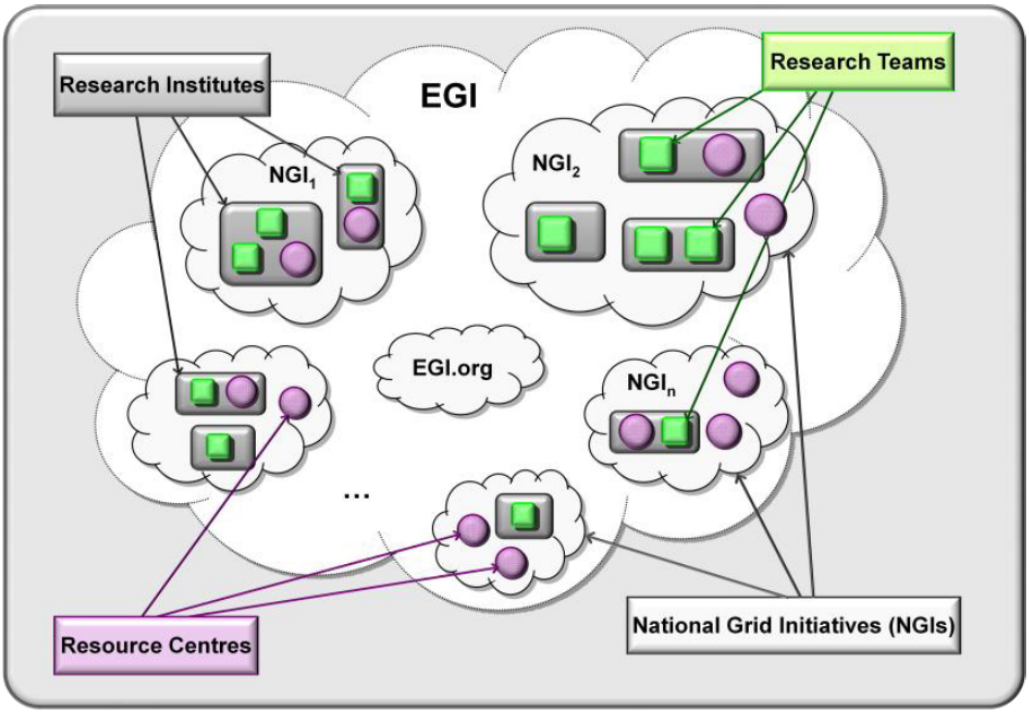


Figura 2.8: Schema del modello previsto da EGI. Fonte [42]

- ogni NGI dovrà assicurare la gestione dell'infrastruttura nazionale rispettando accordi sulla qualità di servizio e l'integrazione con le altre NGI;
- ogni NGI deve fornire servizi generali alle applicazione delle proprie comunità, promuovendone l'uso anche a nuove discipline scientifiche;

Il mantenimento, il supporto, l'interoperabilità e lo sviluppo del middleware non sarà una attività interna di EGI poiché le esperienze necessarie non potranno essere presenti nelle singole NGI. Per garantire la continuità dell'infrastruttura Grid in Europa, il partner naturale di EGI saranno i cosiddetti *Middleware Consortia* che forniranno i diversi componenti del middleware: gLite [3], ARC [43], UNICORE [44]. I compiti principali di questi consorzi saranno di:

- mantenere, supportare e promuovere l'interoperabilità del middleware in produzione nell'infrastruttura Grid;
- provvedere allo sviluppo di nuove funzionalità quando richieste dalle comunità scientifiche, dalle VO e dai gruppi che si occupano della gestione dell'infrastruttura.

EGI non promuoverà lo sviluppo di nuovo middleware, ma di una distribuzione unificata di componenti certificati: *EGI Unified Middleware Distribution (UMD)*.

2.5 Esempi di applicazioni in Grid

Varie sono le applicazioni che sono state completamente adattate all'ambiente distribuito di Grid e che utilizzano quotidianamente l'infrastruttura europea. Tali applicazioni sono raggruppabili per area a seconda delle discipline scientifiche di riferimento, come illustrato nella figura 2.9.

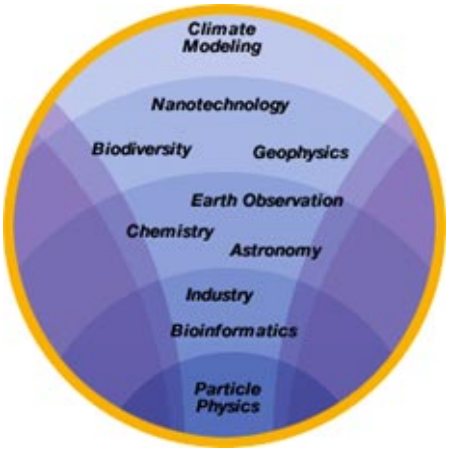


Figura 2.9: Area delle applicazioni in EGEE

Nei paragrafi seguenti sono descritte le tipologie delle principali applicazioni. Un quadro più completo può essere ricostruito mediante il database delle applicazioni [45] mantenuto dal progetto EGEE-III.

2.5.1 Fisica delle particelle

La Fisica delle particelle rappresenta la disciplina scientifica che attualmente apporta la gran parte delle risorse disponibili sull'infrastruttura europea e che ne fa il maggior utilizzo. Per esempio i quattro esperimenti LHC usano Grid per eseguire programmi di simulazione e per analizzare i dati generati dai raggi cosmici, e in futuro per svolgere programmi per la ricostruzione e l'analisi dei dati.

Inoltre, gli esperimenti LHC svolgono periodicamente i cosiddetti Data Challenge, cioè periodi prolungati di tempo nei quali testano il workflow completo del loro sistema di produzione, analisi e trasferimento dei dati. Un esempio di questo

tipo di test è quello fornito dall'esperimento LHCb (*Large Hadron Collider beauty*) che nel Data Challenge del 2006 [46] ha generato circa 700 milioni di eventi usando risorse distribuite in 120 centri di calcolo, raggiungendo punte di 10000 job eseguiti contemporaneamente, in 475 giorni. Ogni esperimento LHC opera in un ambiente multi-grid, a seconda delle rispettive storie e mandati: i due sperimenti principali, ATLAS e CMS, hanno una parte significativa della collaborazione negli Stati Uniti, per un totale di circa 200 enti di ricerca. Entrambi usano anche l'infrastruttura di *Open Science Grid (OSG)* [47]. Quest'ultima non è basata sul middleware gLite, ma utilizza *Virtual Data Toolkit (VDT)* [48], una distribuzione di pacchetti basata su Condor-G [15], Globus e altri componenti Grid[2], come per esempio il VOMS, descritto nella sezione 3.2.1. L'esperienza maturata da questa collaborazione è molto importante nello sviluppo del middleware gLite: da un lato sono da considerare gli stimoli derivanti dalla competizione di prodotti basati su soluzioni differenti, dall'altro non bisogna dimenticare di affrontare le problematiche di interoperabilità derivanti dall'uso di middleware diversi da parte delle comunità scientifiche.

Anche altri esperimenti HEP[3] usano l'infrastruttura EGEE, come per esempio l'esperimento CDF [50][4], D0 [51], ZEUS [52] e BaBar [53][5]. Sebbene le applicazioni di questi esperimenti non fossero state pensate per essere eseguite in ambiente Grid, le rispettive comunità scientifiche, dopo una valutazione dei servizi offerti, hanno adattato il loro software per essere eseguito anche in Grid e sono riusciti a trarne profitto nelle loro produzioni. I risultati ottenuti confermano la bontà dei servizi generali offerti dal middleware gLite.

2.5.2 Biomedicina

La Biomedicina è una delle prime discipline che si è impegnata nello sviluppo di applicazioni orientate a Grid. Al momento sono diverse decine le applicazioni che vengono regolarmente eseguite sull'infrastruttura di EGEE, e spaziano dall'analisi tomografiche a quelle genomiche. Fra i progetti principali vi è Mammogrid, che ha l'obiettivo di creare un database europeo di mammografie che possono essere usate per l'analisi da un vasto numero di applicazioni mediche e rappresenta una delle

[2] Per un elenco aggiornato delle componenti di VDT si veda [49].
[3] High-Energy Physics
[4] CDF: Collider Detector at Fermilab
[5] the B and B-bar experiment

prime collaborazioni mediche all'interno dell'Unione Europea. Altre applicazioni biomediche, nei campi degli studi del genoma e nella scoperta di nuovi farmaci, sono in fase di sviluppo e test per essere eseguite in ambiente Grid.

2.5.3 Scienze della Terra

Varie sono le discipline che stanno sviluppando applicazioni Grid in questa area: la Geoscienza, l'Idrologia, l'Osservazione della Terra, la climatologia. Ognuno di questi campi ha diverse applicazione in fase di test in EGEE e altre già in produzione. In generale sono tutti esperimenti legati ad osservazioni satellitari che prevedono l'inizio della presa dati a breve. Le simulazioni che vengono prodotte oggi tramite EGEE hanno una grande importanza per lo sviluppo degli esperimenti stessi, e la partecipazione attiva di queste comunità rappresenta la prova più significativa che la strada intrapresa con lo sviluppo di tecnologie Grid in generale, e del middleware gLite in particolare, potrà essere la risposta alle esigenze di calcolo e archiviazione più diverse.

2.5.4 Altri ambiti scientifici

Tra le altre comunità scientifiche che utilizzano Grid ed il middleware gLite per la loro produzione vale la pena citare i seguenti progetti:

- ARGO/YBJ
  L'esperimento è realizzato da una collaborazione italo-cinese, che vede coinvolte l'Istituto Nazionale di Fisica Nucleare (INFN) [54] e alcune Università per la parte italiana e IHEP di Pechino, insieme a numerose università cinesi. In particolare, l'INFN ha finanziato la realizzazione di ARGO/YBJ, il cui apparato è stato realizzato in Tibet. L'esperimento è volto all'individuazione e allo studio delle sorgenti di radiazione gamma, nonché allo studio della radiazione cosmica diffusa e dei lampi di emissione. I dati dal Tibet vengono trasferiti in Italia, presso il CNAF [55], su supporti magnetici a causa delle difficoltà di interconnessione con il Tibet. Da qui, attraverso servizi basati sul middleware gLite, vengono messi a disposizione e analizzati dagli istituti che partecipano all'esperimento.

- MAGIC

Le applicazioni di questo progetto simulano i venti originati da raggi cosmici ad alta energia nell'atmosfera terrestre. Queste simulazioni sono necessarie per analizzare i dati prodotti dal telescopio MAGIC [56] nelle isole Canarie. Le prime produzioni su EGEE sono iniziate nel Marzo 2005.

- ESA Planck
  La missione ESA Planck, grazie al satellite lanciato nel 2007, ha lo scopo di fare uno scanning completo del cielo e misurare la radiazione cosmica di fondo nel dominio delle microonde, con delle implicazioni cosmologiche di grande importanza in quanto si misureranno con grande precisione un insieme di parametri che definiscono il nostro Universo. Su EGEE sono state eseguite simulazioni attraverso il software utilizzato durante la missione.

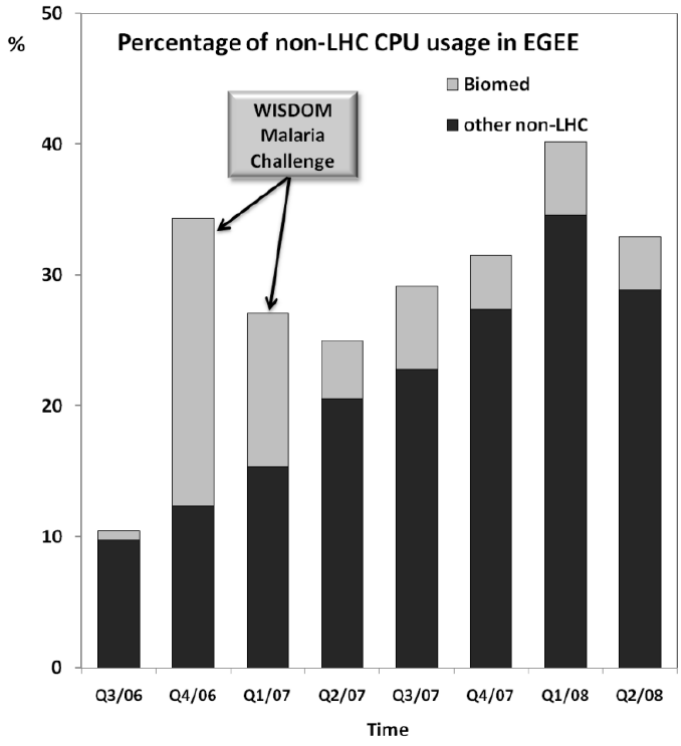


Figura 2.10: Percentuale di utilizzo di CPU in EGEE da parte delle VO non LHC. Fonte: [42]

Per riassumere, negli ambienti scientifici, sempre più comunità hanno iniziato a considerare le tecnologie Grid come un utile strumento per raggiungere i loro obiettivi. Come evidenziato in figura 2.10, sebbene la disciplina della fisica delle particelle e

le enormi necessità degli esperimenti LHC abbiano promosso l'adozione del calcolo basata su Grid e ne siano i principali utilizzatori, l'utilizzo di Grid è cresciuto anche nelle altre discipline scientifiche, passando da una frazione del 10% nel 2006 a oltre il 30% a metà del 2008.

Più risorse di calcolo significa avere l'opportunità di sperimentare algoritmi di analisi sempre più complessi. Poter accedere alle risorse di stoccaggio dati distribuite in tutto il mondo significa aumentare la complessità della propria analisi. Possiamo dire che il contributo più significativo che le tecnologie Grid offrono alle comunità scientifiche medie e piccole è la possibilità di fare parte di collaborazioni internazionali altrimenti fuori dalla loro portata.

Mentre l'applicabilità del modello di calcolo Grid pare dunque ormai consolidata in varie discipline scientifiche, l'uso di Grid si deve ancora affermare in campo industriale. In questo ambito attualmente, le tecnologie Grid sono utilizzate prevalentemente nel settore economico finanziario con l'esecuzione di simulazioni Montecarlo e complesse analisi statistiche, ma non mancano applicazioni pilota anche in altri ambiti, come dimostrano il progetto BEinGRID [8] e l'attività svolta nell'ambito dell'EGEE Business Forum [9]. Tuttavia, standard aperti e condivisione di risorse e informazioni sono visti come concetti antagonisti alla competizione e privacy (requisiti base nel mondo industriale) e la tecnologia Grid deve ancora raggiungere un livello tale da consentire un veloce apprendimento e adattamento delle applicazioni esistenti.

Capitolo 3

Il middleware gLite

"*640 kilobytes is all the memory you will ever need*"
Bill Gates.

Un singolo calcolatore è formato da diverse componenti, come il processore, il disco fisso, il sistema operativo, le periferiche di input/output. Il concetto di Grid è quello di creare un ambiente simile a questo, in presenza di componenti eterogenee e distribuite geograficamente. Il **middleware Grid** è il software che si posiziona tra il sistema operativo e le applicazioni e che permette un accesso sicuro e omogeneo alle risorse, a prescindere dalle loro specificità implementative. Il middleware è costituito da diverse componenti, come librerie e servizi, che contribuiscono a rendere l'infrastruttura Grid disponibile agli utenti. I servizi Grid sono elementi costitutivi dell'infrastruttura di Grid. Alcuni di questi vengono detti '*centrali*' essendo condivisi da tutti i gruppi di utenti ed essendo necessari per il funzionamento dell'intera infrastruttura. Altri gruppi di servizi sono invece gestiti a livello di sito e assolvono la funzione di rendere le risorse di tale sito accessibili via Grid.

Il Grid middleware è organizzato in componenti, ognuna delle quali si occupa di determinate funzioni per fornire agli utenti una interfaccia unica e standard alle risorse dei siti. Una *risorsa Grid* è un componente dell'infrastruttura che fornisce determinati servizi con regole e modalità di accesso definite da chi la amministra. Risorse tipiche in ambiente Grid possono essere quelle di calcolo e di storage. Gli accessi possono essere gestiti a livello della risorsa stessa o da una terza componente che fa da mediazione tra l'utilizzatore e la risorsa stessa, proteggendola da accessi non autorizzati.

3.1 Il middleware gLite

Il progetto EGEE ha sviluppato un gruppo di componenti che costituiscono il middleware denominato gLite [3]. Di seguito vengono descritte nel dettaglio le componenti principali più interessanti dal punto di vista del load balancing e failover, sulle quali si basa il lavoro di tesi. Una descrizione accurata delle funzionalità di gLite è disponibile in [57].

Il middleware gLite fornisce un insieme di servizi integrati che permettono l'esecuzione sicura di job e la gestione di dati in un ambiente Grid distribuito. La figura 3.1 mostra le componenti principali che verranno descritte in questo capitolo.

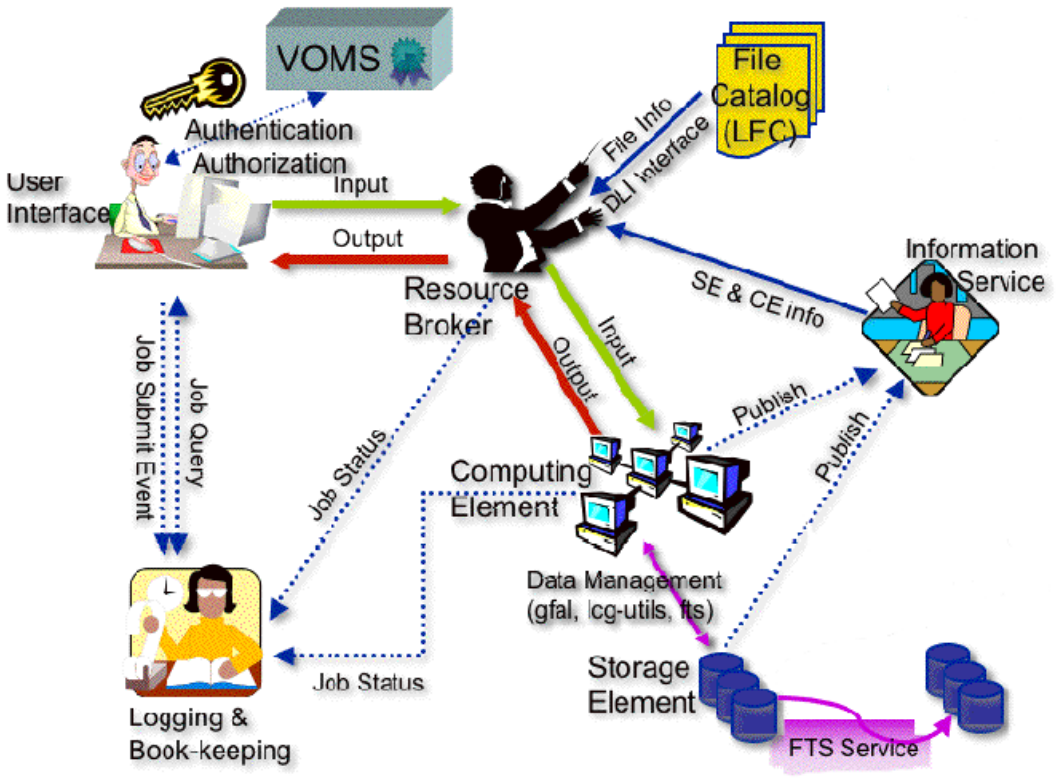


Figura 3.1: I componenti principali del middleware gLite.

3.1.1 User Interface

Il punto di accesso al middleware gLite è la cosiddetta ***User Interface (UI)*** che fornisce comandi e API[1] per la sottomissione sicura di job, per la gestione dei dati e

[1] Application Program Interface

per l'interrogazione del sistema informativo. Tipicamente una UI non è altro che un insieme di applicativi client installati su una singola macchina alla quale gli utenti possono accedere con il loro account personale. Le operazioni Grid che possono essere eseguite da una UI sono varie, per esempio:

1. l'esecuzione delle operazioni di autenticazione e autorizzazione necessarie per l'accesso alle risorse Grid, utilizzando il servizio *VOMS*;

2. la consultazione del sistema informativo per recuperare informazioni sulle risorse, interagendo con il servizio top-level BDII;

3. la gestione dei job: è possibile determinare una lista delle risorse disponibili all'esecuzione di un dato job, mandarlo in esecuzione, cancellarlo, determinarne lo stato in un dato momento, recuperare l'output e le informazioni di log di un job già terminato. Queste operazioni coinvolgono l'utilizzo dei servizi di *Logging and Book-keeping* e del *Workload Management System*.

4. la copia, la replica e la cancellazione di file dalle risorse di storage della Grid, attraverso i servizi offerti dagli *Storage Element*, dai cataloghi dei file (*Logical File Catalog*) e del servizio per il trasferimento dati (*File Transfer Service*).

3.2 La sicurezza in Grid

La sicurezza è un requisito fondamentale per l'infrastruttura di gestione dei dati in Grid. Ogni livello dell'architettura deve essere in grado di soddisfare i requisiti di sicurezza richiesti, una singola omissione comprometterebbe l'intero sistema. I meccanismi di sicurezza devono essere costruiti in funzione dei differenti contesti applicativi, ad esempio i dati appartenenti a comunità mediche risultano estremamente sensibili e devono essere condivisi rispettando le norme a tutela della privacy, i dati provenienti da comunità finanziare devono soddisfare le politiche di accesso precedentemente stabilite, mentre nel contesto di fisica delle alte energie l'enfasi è sull'accesso alle risorse di storage che deve avvenire secondo gli accordi fra le organizzazioni virtuali. Il sistema deve essere quindi in grado di recepire delle policy che definiscono le politiche di utilizzo delle risorse e dei dati condivisi, e i servizi devono fornire i meccanismi per garantire che tali politiche siano rispettate. La sicurezza

deve essere completamente integrata per prima cosa nei meccanismi di comunicazione utilizzati, per garantire l'autenticazione dei vari servizi coinvolti nelle operazioni. Il secondo aspetto è l'autenticazione degli utenti che richiedono le operazioni sui dati. L'***autenticazione*** deve avvenire attraverso meccanismi standard come *Grid Security Infrastructure (GSI)* [58]. Questa infrastruttura di sicurezza collega insieme tre componenti:

1. *certificati X.509*: è uno standard ISO[2] e IETF[3] che collega credenziali a chiave pubblica ad un identità. I certificati sono rilasciati da un insieme di ben definite *Certification Authorities (CA)*. Le credenziali sono divise in due parti: la parte pubblica nel certificato (da condividere), e la parte privata che deve essere tenuta segreta.

2. *Public Key Infrastructure (PKI)*: insieme di standard che definisce come i certificati e le CA debbano lavorare insieme per consentire, per esempio, la mutua autenticazione agli utenti e alle risorse;

3. *Generic Security Services Application Program Interface (GSS-API)*: standard IETF che definisce una interfaccia unificata a meccanismi di autenticazione eterogenei come Kerberos, i certificati X.509, etc.

Una volta autenticato, l'utente deve essere autorizzato a eseguire l'operazione richiesta. In un contesto complesso come quello di Grid si possono avere differenti forme di valutazione dell'autorizzazione. Per esempio può essere autorizzata la generica operazione sul servizio oppure la specifica richiesta composta dall'operazione e dai dati in ingresso. Questo implica che il meccanismo generale di sicurezza in Grid sia sufficientemente flessibile per prevedere come dati di input del processo decisionale anche i dati richiesti. Il processo di autorizzazione, inoltre, deve essere in grado di considerare i ruoli, che differenziano ad esempio utenti con i diritti di amministrazione da quelli con i diritti semplici di utilizzo delle risorse.

3.2.1 Autenticazione e autorizzazione in gLite

All'interno del progetto European Data Grid (EDG) [41] è stato sviluppato il servizio **Virtual Organization Membership Service (VOMS)** [61], un sistema di

[2] International Organization for Standardization [59]
[3] Internet Engineering Task Force [60]

gestione degli attributi di autorizzazione in ambienti di collaborazioni distribuite, che ora è parte del middleware gLite. Il servizio VOMS permette di formulare asserzioni che coinvolgono gli utenti utilizzabili per la valutazione delle politiche di autorizzazione. Il VOMS gestisce e definisci gruppi, ruoli e proprietà dei membri e associa questi attributi a richiesta, fornisce un database di ruoli e proprietà e un insieme di applicazioni per l'accesso e la gestione del DB e la generazione di credenziali contenenti gli attributi.

Come definito da OGSA, in Grid la gestione dell'identità deve essere effettuata in modo da essere portata in diversi contesti. L'esecuzione di un job in Grid richiede l'interazione fra un insieme di servizi per la designazione delle risorse e la creazione dell'ambiente necessario, ma al livello più basso ha come conseguenza l'esecuzione di un processo su un certo elaboratore. Diventa quindi necessario garantire che l'utente locale con il quale il processo viene eseguito rispetti le esigenze di accesso alle risorse determinate dalla sua identità di Grid. Sistemi come **LCMAPS**[4] attraverso la valutazione dell'identità del soggetto e degli attributi associati determinano per conto di quale utente locale il job dovrà essere eseguito. Il numero elevato di utenti utilizzatori delle risorse e la dinamicità delle VO ha introdotto la necessità di un meccanismo che eviti la definizione di un utente locale specifico per ogni soggetto. Con il concetto di *pool account* si identifica un insieme limitato di utenti locali associato ad un determinato gruppo e sottogruppo di una certa VO. Quando un utente appartenente a quel gruppo richiede l'esecuzione di un job, viene associato ad un utente locale preso dallo specifico pool account. L'associazione identità di Grid-utente locale, definita per la VO dell'utente, ha un tempo limitato: richieste successive potranno essere eseguite da utenti locali differenti.

Questo meccanismo implementa il cosiddetto *single sign-on*, che consiste nella possibilità di autenticarsi una volta sola e creare una *credenziale proxy*, cioè un certificato temporaneo, con un tempo di vita molto più limitato del certificato originario, che accompagna ogni singola richiesta dell'utente stesso. Il meccanismo della *delega* permette poi ad una risorsa remota o ad un servizio di utilizzare le informazioni di security dell'utente, ovvero il suo certificato proxy, in nome dell'utente stesso. Questo meccanismo è molto importante in quelle operazioni basate su un workflow che coinvolge molteplici risorse e servizi.

[4]Local Credential Mapping Service [62]

3.3 Il sistema informativo

Altri protocolli di base che si trovano nel Resource layer di ogni middleware sono quelli che permettono l'inserimento di informazioni sulle risorse nel sistema informativo. Un ***Grid Information Service*** permette la registrazione delle risorse e dei servizi Grid disponibili, in modo che queste possano essere annunciate in Grid e conseguentemente utilizzate. Attraverso il sistema informativo, gli elementi di informazione sulle risorse disponibili in Grid e il loro stato possono essere utilizzati in modo da decidere dove e come un dato job potrebbe e dovrebbe essere eseguito *(Resource Discovery)*. Uno degli aspetti chiave di un sistema informativo è la definizione di uno ***schema*** completo e estensibile che possa descrivere completamente e in modo univoco le risorse, i servizi e le loro proprietà. La presenza di uno schema universale permette di descrivere le risorse di calcolo e storage e i servizi in modo omogeneo e trasparente. Questa è una delle condizioni necessarie che permette l'interoperabilità fra infrastrutture Grid differenti. Per questo motivo i progetti Grid europei e statunitensi hanno dato vita ad un gruppo di lavoro OGF denominato ***Grid Laboratory for a Uniform Enviroment (GLUE)*** [63], che si occupa della standardizzazione di questo schema. Per ogni tipologia di sistema o di servizio il modello definisce l'insieme degli attributi che lo caratterizzano in merito alle operazioni di controllo dello stato e di discovery della risorsa. Attraverso il sistema informativo vengono pubblicate tutte le informazioni significative, definite dallo schema GLUE, relative ai servizi e alle risorse presenti, ad esempio, in un sito di Grid. I servizi di coordinamento ed esecuzione dei job interagiscono con il sistema informativo per determinare il grafo di esecuzione di un processo.

3.3.1 gLite Grid Information System

Nel middleware gLite il sistema informativo è basato sul Globus Monitoring and Discovery Service (MDS) [64] e implementato dal servizio BDII[5]. Il servizio BDII consiste in uno database LDAP[6] che viene aggiornato da un processo esterno (*provider*) che si occupa di recuperare le informazioni dalle risorse sorgenti in conformità

[5] Berkeley Database Information Index [65]
[6] Lightweight Directory Access Protocol [66]

del GLUE schema [63] [67], unirle e creare un file in formato LDIF[7] con il quale viene aggiornato il database.

Come evidenziato in figura 3.2, il servizio BDII è organizzato in una struttura gerarchica su tre livelli:

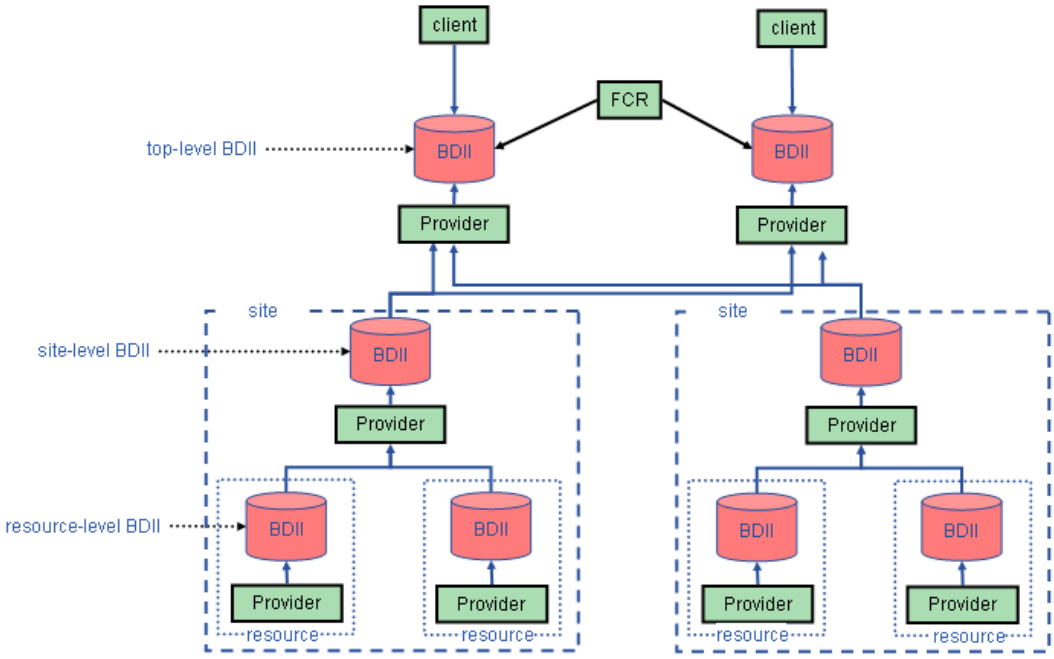


Figura 3.2: La struttura gerarchica del sistema informativo basato sul servizio BDII. Fonte [65]

1. Resource-level BDII: è generalmente installato sulla risorsa stessa e fornisce tutte le informazioni sul servizio che la risorsa offre;

2. Site-level BDII: ogni sito ha un site level BDII che interroga i BDII delle risorse che compongono il sito stesso, in modo da aggregare le varie informazioni del sito;

3. Top-level BDII: questo servizio fornisce un aggregato di tutte le informazioni che recupera interrogando i site level BDII e contiene quindi le informazioni su tutti i servizi e le risorse disponibili in Grid.

Le differenze tra Resource-level, Site-level e Top-level BDII sono quindi legate al dominio di riferimento e alle informazioni che contengono, e non alla natura del

[7]LDAP Data Interchange Format [68]

software. Essendo sia il Top-level BDII che il Site-level BDII a loro volta dei servizi Grid, devono anche loro essere pubblicati nel sistema informativo.

Nel progetto EGEE è stato poi definito un meccanismo denominato ***Freedom of Choice for Resources (FCR)*** [69], utilizzato al livello del Top-level BDII. Interrogando il top-level BDII, viene ottenuta una lista dei servizi disponibili per una data VO. A questo punto un VO manager può creare una black list di servizi sulla base dei risultati dei SAM test, descritti nella sezione 4.5.2, in modo da mascherare la presenza nel sistema informativo per la sua VO di quei servizi che falliscono determinati test. Il Top-level BDII quindi scarica la lista che elenca tutti i siti che compongono la Grid in base a quanto registrato nel *Grid Operation Center Database (GOCDB)* [70], a cui applica le modifiche contenute nelle black list eventualmente definite a livello di VO.

3.4 Workload Management System

Il ***Workload Management System*** è uno dei Collective Service dell'architettura Grid descritta nella sezione 1.3.1, e rappresenta uno dei servizi più importanti di Grid. È composto da un insieme di componenti del middleware che si occupano di selezionare le risorse che soddisfano i vincoli specificati dall'utente, di distribuire il carico sulle risorse candidate in base ai criteri specificati dall'utente e/o in base a politiche interne di bilanciamento del carico, permettere agli utenti di supervisionare lo stato di task computazionali, comunemente chiamati job, durante il loro ciclo di vita.

3.4.1 Funzionalità del WMS

Il componente principale del WMS è il ***Workload Manager (WM)***, che ha il compito di accettare e soddisfare le richieste per la gestione dei job, espresse attraverso il *Job Description Language (JDL) [71]* e la cui architettura, insieme alle interazioni con altri servizi, è mostrata in figura 3.3.

Le componenti principali del WM sono le seguenti:

- **Match Maker**: il WM può adottare diverse politiche di scheduling. Consideriamo da un lato la politica di scheduling definita 'veloce', la quale prevede che per un dato job siano selezionate la risorse richieste più vicine al job stesso,

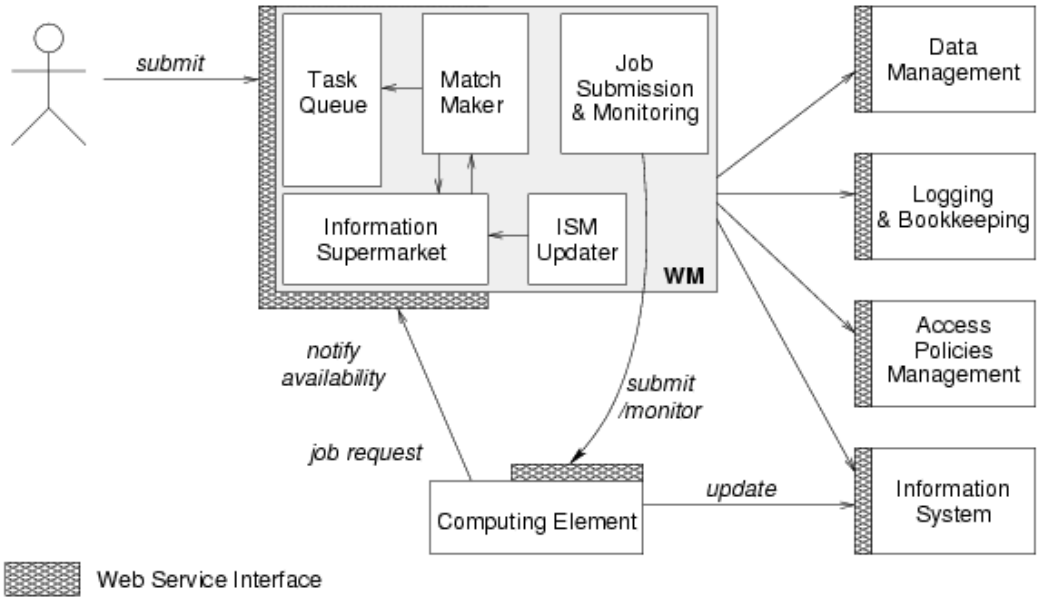


Figura 3.3: Schema dell'architettura del Workload Manager in gLite e interazioni con gli altri componenti del middleware. Fonte [72]

e una volta identificate, il job sia trasferito ad esse, venendo quindi inserito in uno scheduler locale. Dal lato opposto, consideriamo la politica di scheduling 'lenta' che prevede che i job siano trattenuti dal WM fino a che non diventa disponibile una risorsa: a questo punto viene selezionato il job più adatto ad essere eseguito su quella risorsa, viene trasferito e quindi eseguito immediatamente. Esistono poi diverse politiche di scheduling intermedie alle due descritte. A livello di match-making la differenza fondamentale tra i due tipi di scheduling è che il primo implica un confronto del job con molte risorse, mentre nel secondo caso il confronto è tra una risorsa e molti job. Il WM permette di utilizzare contemporaneamente diverse politiche, attraverso una serie di plugin che, valutati prima di tutto i vincoli espressi dal job, possono adottare politiche diverse considerando anche lo stato generale della Grid attraverso le metriche e le statistiche che avranno definito.

- **Information Supermarket (ISM)**: è la componente del WM in cui è mantenuta una collezione di informazioni relative alle risorse e al loro uso. L'ISM è una sorta di cache interna che permette di rendere il WM indipendente dalle prestazioni e dalla disponibilità del sistema informativo che utilizza come

sorgente.

- **Task Queue**: il WM accoda internamente le richieste di sottomissione ricevute, mentre le risorse non sono ancora disponibili, invece di rifiutarle. Queste richieste vengono riprocessate periodicamente, non appena una notifica di risorsa disponibile appare nel ISM.

Oltre al WM, un'altra componente fondamentale del WMS è il ***Logging and Bookkeeping Service (L&B)***. Questo servizio traccia i job in termine di eventi, come ad esempio stato di sottomissione, determinazione di un CE, inizio dell'esecuzione. Gli eventi sono passati ad una componente fisicamente vicina al L&B, denominata *locallogger* per evitare ogni problema di rete. Queste componenti conservano gli eventi in un disco locale e hanno solo la responsabilità di consegnarli in seguito. La destinazione di un evento è uno dei *bookkeeping servers* assegnato in maniera statica ad un job appena viene sottomesso. Il server processa l'inizio degli eventi per dare una panoramica dello stato del job e può contenere anche informazioni relative a vari attributi, come per esempio il CE di destinazione, il codice con cui il job è terminato, etc. Un utente può eventualmente registrarsi per avere le notifiche su un particolare cambiamento di stato del job che ha sottomesso.

3.5 Computing Element e Worker Node

Il Computing Element (CE) è il servizio che interfaccia le risorse di calcolo locali con l'infrastruttura Grid. La sua funzione principale è quella della gestione dei job locali, ma deve fornire diverse altre funzioni come, per esempio, fornire le informazioni riguardanti le caratteristiche delle risorse di calcolo offerte e il loro stato.

Il CE può lavorare secondo due modalità differenti:

1. *push mode*: prevede che il job sia inviato ad un CE per essere eseguito. Quando un job viene inviato a un CE, esso viene accettato solo se ci sono risorse che soddisfano i vincoli specificati dall'utente

2. *pull mode*: prevede che sia il CE a chiedere al WMS dei job da eseguire. Nella richiesta devono essere incluse le caratteristiche delle risorse, in modo tale che il WMS possa selezionare il job più adatto. Esistono diverse tecniche

di scheduling per il pull mode, che possono essere analizzate per determinare quali di esse forniscono le migliori performance in situazioni differenti. Le possibili tecniche sono:

- Il CE richiede un job da tutti i WMS conosciuti. Se due o più WMS offrono dei job, solo il primo che arriva è accettato dal CE, mentre gli altri vengono rifiutati.
- Il CE richiede un job da un solo WMS. Se il WMS contattato non ha job disponibili viene notificato un altro WMS. Inoltre, il CE espone un'interfaccia Web service e può essere usato da un generico client.

Un job inviato ad un CE viene preso in carico dal *local scheduler (LRMS)* per la sua esecuzione nei nodi che compongono la farm: i ***Worker Node (WN)***. I Worker Node di gLite hanno gli stessi comandi e librerie installati sulla UI, con l'aggiunta dei comandi di gestione dei job caratteristici del sistema LRMS adottato dal sito.

3.6 La gestione dei dati in Grid

Le problematiche di gestione dei dati in Grid sono conseguenze della struttura distribuita ed eterogenea dell'ambiente di esecuzione, delle diverse tipologie di applicazioni e dei diversi requisiti provenienti dai contesti applicativi. Il documento OGSA Data Architecture [73] descrive le interfacce che virtualizzano servizi e risorse legate al sistema di gestione dati, come per esempio il trasferimento, la gestione dello storage, l'accesso, la federazione e la replicazione di dati. Descrive inoltre le interfacce per la localizzazione dei dati, lo staging di file da sistemi di archiviazione di massa, il caching e la replicazione.

In un ambiente fortemente distribuito ed eterogeneo come Grid è di fondamentale importanza che siano forniti i meccanismi per associare ad ogni entità e risorsa un identificativo univoco e condiviso. Questo permette di identificare le risorse con le quali il processo di elaborazione interagisce, di mantenere informazioni significative sulle operazioni effettuate, di identificare un dato condiviso, di associare un dato ad una o più istanze fisiche su sistemi di storage o ancora di registrare la provenienza dei dati e degli utenti richiedenti. Il sistema di gestione dei nomi deve essere strutturato in modo da garantire la possibilità ad utenti e applicazioni di definire nomi

dipendenti dal contesto e dal contenuto del dato, ma allo stesso tempo deve gestire l'associazione univoca di un nome con una specifica risorsa.

3.6.1 Data management in gLite

L'architettura dei servizi di gestione di dati proposta nel progetto EGEE assume che, al livello più basso, il concetto di dato sia assimilabile a quello di file. Questa assunzione è conseguenza dei casi d'uso reali provenienti da applicazioni di fisica delle alte energie e di scienze biomediche, le due principali comunità coinvolte nel progetto. I componenti di gLite seguono una architettura che facilita l'interoperabilità e soddisfa le raccomandazioni definite da OGSA. In questo modo l'architettura non è vincolata a specifiche implementazioni, ma propone un insieme di servizi interoperabili in grado di eseguire le richieste dell'utente in diversi contesti e su diverse risorse.

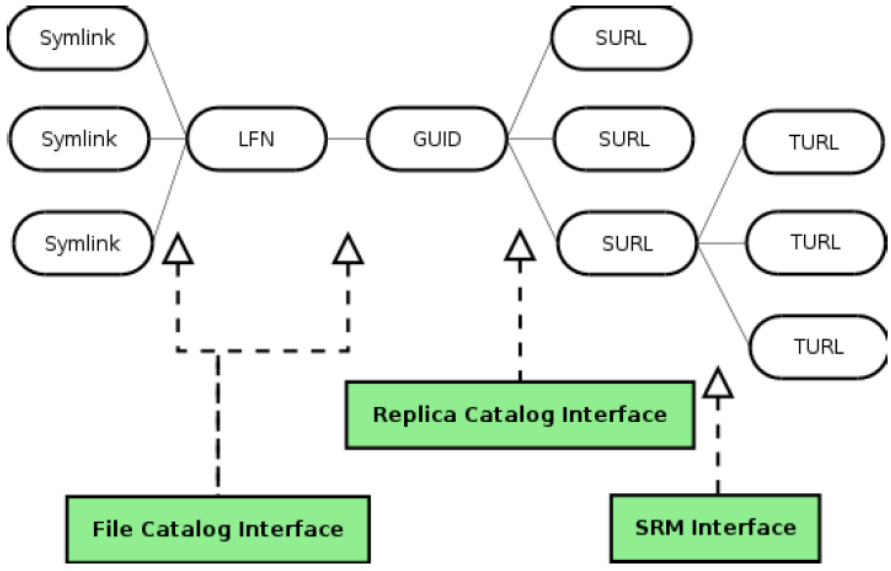


Figura 3.4: Schema del naming dei file in gLite. Fonte [72]

In figura 3.4 sono riportate le varie entità componenti il sistema dei nomi in EGEE:

- il *Logical File Name (LFN)* rappresenta l'identificatore logico, scelto da un utente, di un file. È unico e può essere modificato in accordo con le richieste

degli utenti, è basato su un namespace gerarchico simile ai convenzionali file system. Ogni VO ha il suo namespace di riferimento;

- il *Global Unique Identifier (GUID)* è un identificatore logico immutabile, basato sullo standard UUID [8], che garantisce la sua unicità per costruzione, associato con un LFN. Identifica un puntatore immutabile ad un determinato file;

- i *Logical Symlinks* sono link simbolici ad un certo LFN. Rispecchiano il concetto di link nella struttura logica del namespace, ogni LFN può avere associati diversi Symlink;

- il *Site URL (SURL)* identifica una specifica istanza fisica di un file all'interno di una servizio di storage. Un file può avere diverse istanze, quindi un singolo GUID può essere associato a diversi SURL. La struttura del Site URL in EGEE è quella di un SRM name, compatibile quindi con l'interfaccia SRM.

- il *Transport URL (TURL)* è un URL che può essere utilizzato per il trasferimento di un file attraverso protocolli standard.

Le categorie costituenti i servizi di data management in EGEE sono: storage, cataloghi e trasferimento.

3.6.2 Storage Element

L'insieme dei servizi necessari per l'accesso ai file e alla gestione dei dati in EGEE è identificato con il termine Storage Element. I principali servizi forniti da un SE sono:

1. *Storage back-end.* I siti di Grid sono generalmente strutturati in farm di calcolo composte da hardware e software eterogeneo per architettura e tecnologia utilizzata. Le applicazione devono poter accedere i dati in maniera indipendente dalla struttura dello storage.

2. Servizio *Storage Resource Manager (SRM)* sulla risorsa di storage. L'interfaccia SRM permette di fornire una meccanismo comune di interazione con lo storage nascondendo le peculiarità dei singoli sistemi.

[8] Universally Unique Identifier [74]

3. *Servizi di trasferimento*: è necessario che siano disponibili diversi protocolli di trasferimento utilizzabili per muovere i file.

4. *Servizio di accesso ai dati* con un interfaccia POSIX-like [75]. Devono essere forniti sistemi di accesso ai file in grado di offrire agli utenti e alle applicazioni un interfaccia standard che virtualizzi i reali protocolli utilizzati.

5. *Servizio di sicurezza e logging.* I servizi coinvolti nell'accesso e condivisione dei dati debbono essere completamente integrati nei processi di autenticazione e autorizzazione degli utenti e dei servizi con i quali interagiscono.

Il componente SE nell'architettura EGEE è il responsabile delle principali operazioni di gestione dello storage e delle interazioni con i servizi di trasferimento e accesso. Deve essere in grado di fornire differenti qualità di servizio in funzione delle organizzazioni virtuali. In accordo con i principali servizi forniti, EGEE definisce che un componente SE abbia almeno tre interfacce: l'interfaccia SRM per permettere agli utenti di gestire lo spazio e i file, per preparare i dati e allocare lo spazio, una seconda interfaccia per fornire un meccanismo di accesso POSIX-like e una terza per il meccanismo di trasporto.

In particolare le funzionalità che un SE deve fornire all'utente attraverso un servizio SRM sono:

1. la gestione dello spazio e dei file secondo le politiche definite dall'organizzazione virtuale di appartenenza;

2. l'organizzazione del sistema di storage sottostante;

3. la gestione dello spazio disponibile attraverso meccanismi di quota, pinning, space reservation;

4. l'utilizzo di protocolli di accesso standard, POSIX-like;

5. la capacità di trasferimento di file fra diversi SE.

La maggior parte delle capacità essenziali stabilite da EGEE per un componente SE rientra nei compiti di un servizio di Storage Resource Manager (SRM) così come definito da OGSA. Con l'evoluzione dell'infrastruttura di Grid e con l'esperienza guadagnata dai progetti precedenti, anche in EGEE è considerata di primaria

importanza la necessità di avere sistemi per la gestione dello spazio e dei file sulle risorse di storage.

3.6.3 I cataloghi e il servizio di trasferimento dati

I cataloghi contengono informazioni legate all'associazione di un nome logico per un file (LFN) utilizzato dagli utenti per indicare un file di Grid, e le rispettive istanze presenti nei vari sistemi SRM, attraverso i SURL. Il servizio ***Logical File Catalog (LFC)***, sviluppato nel progetto DataGrid e in seguito adottato [76] da EGEE, è strutturato per contenere informazioni legate ai permessi di accesso a file e directory.

Il servizio di trasferimento dei dati ha il duplice compito di interagire con i servizi SRM sorgente e destinazione e di gestire il processo di trasferimento attraverso il protocollo gridtfp. In gLite il servizio che si occupa del trasferimento dei dati è il ***File Transfer Service (FTS)*** [77]. Il servizio FTS è in grado di interagire con i servizi SRM per creare le condizioni necessarie al trasferimento, ad esempio assegnando al file sorgente un lifetime sufficientemente ampio da consentire il trasferimento con successo, oppure riservando lo spazio necessario sul servizio SRM destinazione per garantire che non ci siano problemi di occupazione di spazio a trasferimento in corso. Inoltre deve interagire con i sistemi SRM per verificare che il file sia stato copiato con successo. Per le operazioni di trasferimento il sistema FTS è in grado di sfruttare le funzionalità avanzate fornite da gridftp. Oltre all'ottimizzazione del trasferimento, è incaricato di reiterare le operazioni di fallimento e implementare semplici meccanismi di rollback in caso in interruzione.

3.7 Il sistema di configurazione del middleware gLite

Yet Another Installation Manager (YAIM) [78] è una applicazione per la configurazione automatizzata di software in ambiente di Grid. Yaim è pensato per la configurazione di un servizio su una singola macchina, è realizzato per facilitare la configurazione di specifici servizi di Grid. Yaim definisce dei profili per le diverse tipologie di servizio da installare, ogni profilo è caratterizzato da un insieme di script da eseguire per la configurazione del servizio. Yaim è composto da una serie di script

bash e ha una struttura modulare, composta dal core dell'applicazione e un insieme di script legati al componente che deve essere configurato, che sono forniti in RPM specifici.

Per quanto riguarda l'installazione dei servizi offerti dal Tier-1 del CNAF, Yaim è utilizzato in abbinamento con il tool ***Quattor*** [79]: Yaim si occupa della configurazione del middleware, mentre Quattor gestisce l'installazione del sistema operativo e del middleware e la configurazione di diversi tool e componenti, middleware escluso. Quattor è un sistema di amministrazione progettato per fornire un toolkit modulare, portabile e efficiente per la configurazione, l'installazione e la gestione automatica di cluster e farm che utilizzano sistemi basati su Unix, come Linux o Solaris. Il funzionamento di Quattor è basato sulla definizione di template che esprimono lo stato, in termini di software installato e configurato, di una determinata tipologia di macchina, che corrisponde ad una particolare funzionalità di Grid. Quattor è in grado di associare ad un insieme di nodi il template definito dall'amministratore, e gestisce l'applicazione del modello su ogni macchina corrispondente, installando e configurando il software necessario e mantenendo lo stato della macchina coerente con quanto definito.

Capitolo 4

High availability, tecniche e implicazioni

"*Data expands to fill the space available for storage*"
Parkinson's Law of Data.

4.1 Alcune definizioni

High Availability computing e *business critical services* sono termini implicitamente legati a costi e sforzi perché un servizio sia disponibile, quando richiesto, bilanciando costi e benefici. Nel mondo dell'IT (Information technology) la necessità di continuità operativa dei sistemi e dei servizi informatici è direttamente proporzionale all'importanza che questi sistemi hanno nel ciclo produttivo. La ***continuità operativa*** rappresenta l'abilità da parte di un sistema di fornire un servizio ai propri utenti senza interruzioni ed è un sottoinsieme del concetto di High Availability. Questo obiettivo è molto difficile da raggiungere perché i componenti del sistema, siano hardware o software, non sono esenti da problemi o necessità gestionali, mentre un sistema che deve fornire continuità di servizio deve essere *fault tolerant*. Spesso la fault tolerance è una proprietà disponibile per i componenti hardware, ma raramente per quelle software. Per fare in modo che gli utenti non si accorgano dei fallimenti, occorre quindi utilizzare meccanismi di protezione aggiuntivi.

Quanto un sistema debba essere disponibile dipende da quale servizio il sistema deve fornire. Per esempio, per un *Internet Service Provider (ISP)*, High Availability può significare offrire ai propri utenti la possibilità 24 ore su 24 di poter accedere a Internet, mentre per altre imprese potrebbe voler dire garantire una disponibilità dalle

otto del mattino alle otto di sera nei giorni lavorativi. In altri casi, l'indisponibilità di un servizio per qualche secondo potrebbe essere insignificante o disastrosa.

4.1.1 Availability

L'***availability*** [80] di un sistema rappresenta la percentuale di tempo nel quale il sistema opera correttamente all'interno del periodo temporale nel quale è previsto che debba funzionare. Per esempio, se un servizio deve essere operativo per otto ore al giorno, allora l'availability è misurata come percentuale di quelle otto ore.

L'availability si può quindi misurare (formula 4.1) come il rapporto tra il tempo nel quale il servizio è disponibile (***uptime***) e il tempo totale nel quale era previsto che dovesse esserlo, rappresentato dalla somma di disponibilità e indisponibilità (***downtime***):

$$Availability = \frac{Uptime}{Uptime + Downtime} \tag{4.1}$$

La stessa misura può essere espressa in valore assoluto (239 ore su 240 nel mese scorso) o più comunemente in percentuale (99.6% nel mese scorso).

Nel caso siano conosciuti i valori medi tra due fallimenti (*mean time between failures (**MTBF**)*) e il tempo medio necessario per ripristinare il servizio (*mean time to repair (**MTTR**)*), è possibile stimare (formula 4.2) la disponibilità di un servizio e capire su quali grandezze è più conveniente concentrarsi per aumentarla:

$$Availability = \frac{MTBF}{MTBF + MTTR} \tag{4.2}$$

4.1.2 Reliability

La IEEE[1] definisce la ***Reliability*** come l'abilità di un sistema o di un suo componente di svolgere la funzione richiesta sotto certe condizioni e in uno specifico intervallo di temporale.

A differenza dell'availability quindi, nel calcolo della reliability sono esclusi i periodi di indisponibilità pianificati. Possiamo quindi utilizzare le formule 4.3 e 4.4:

[1] Institute of Electrical and Electronics Engineers

$$Reliability = \frac{Availability}{Scheduled\ Availability} \tag{4.3}$$

$$Reliability = \frac{Availability}{1 - Scheduled\ Downtime - Unknown} \tag{4.4}$$

4.1.3 Serviceability

La ***serviceability*** [80] esprime l'abilità del personale tecnico nell'installare, configurare e monitorare i servizi, identificare le eccezioni e fallimenti, analizzare le cause, isolare i problemi, gestire hardware e software in modo tale da risolvere eventuali problemi e rimettere in funzione il servizio. Può quindi esprimere la misura di quanto facilmente un sistema può essere ripristinato o riparato. Per esempio un server con componenti modulari (come dischi o alimentatori) e rimovibili a caldo (*hot swappable*) avrà un alto valore di serviceability. Questa misura può essere espressa (formula 4.5) come l'inverso del tempo totale di gestione (***Mainteinance***) più quello dovuto a fallimenti (***Mainteinance for failures***) sulla vita totale del sistema (***Service Life Time***):

$$Serviceability = \frac{Service\ Life\ Time}{Mainteinance + Mainteinance\ for\ failures} \tag{4.5}$$

4.1.4 Disaster Recovery

In letteratura il concetto di *Disaster Recovery* è forse ancora più incerto da definire rispetto all'Availability. Nel campo delle tecnologie informatiche per ***Disaster Recovery*** [80] si intende l'insieme di misure tecnologiche e i processi organizzativi atti a ripristinare sistemi, dati e infrastrutture necessarie all'erogazione di un dato servizio a fronte di gravi emergenze.

La prolungata indisponibilità di un dato servizio, inseguito ad un 'disastro', può rendere necessario l'utilizzo di una strategia di ripristino del sistema utilizzando anche siti alternativi rispetto a quello primario che solitamente fornisce il servizio. In pratica i sistemi e i dati considerati critici vengono ridondati in un 'sito secondario' o '*sito di Disaster Recovery*' per far si che, in caso di un disastro tale da rendere inutilizzabili i sistemi informativi del sito primario per un prolungato periodo di

tempo, sia possibile attivare le funzionalità dal sito secondario in tempi brevi e minimizzando la perdita di dati. Chiaramente quanto più stringenti saranno i livelli di continuità richiesti, tanto più alti saranno i costi di implementazione di una possibile soluzione. In particolare, i livelli di servizio sono usualmente definiti dai due parametri:

- *Recovery Time Objective (RTO)* rappresenta il tempo necessario affinché il servizio sia nuovamente disponibile;
- *Recovery Point Objective (RPO)* rappresenta il punto temporale dal quale i dati verranno ripristinati, infatti in caso di disastro può capitare che una parte di questi venga persa.

In particolare, per quanto riguarda la gestione dei dati, sono possibili varie procedure di Disaster Recovery, come per esempio le seguenti due:

- **La replica sincrona** garantisce la consistenza dei dati presenti sui due siti poiché considera ultimata una transazione solo se i dati sono stati scritti sia sulla postazione locale che su quella remota. In caso di evento disastroso sulla sede principale, le operazioni sul sito di Disaster Recovery possono essere riavviate molto rapidamente (basso RTO e RPO praticamente nullo). La replica sincrona è limitata dalla incapacità dell'applicazione di gestire l'impatto del ritardo di propagazione sulle prestazioni: tale vincolo può essere sia fisico che tecnologico.
- **La replica asincrona** prevede che prima avvenga la transazione dai dati sulla risorsa principale (*master*) e successivamente che i dati vengano copiati anche sulla risorsa secondaria (slave). Questa tecnica è caratterizzata da un basso RTO e un RPO legato alla frequenza con la quale avvengono le copie dalla risorsa master a quella slave.

4.2 Alcune tecniche utilizzate

Questa sezione illustra alcune delle metodologie più utilizzate nelle tecnologie informatiche nel campo dell'High Availability. Le tecniche per l'implementazione dell'HA di un sistema possono essere applicate a più livelli, come illustrato nella tabella 4.1.

Questa tesi si focalizza sulle tecniche a livello applicativo che possono essere ideate in modo particolarmente mirato ai servizi che compongono una Grid computazionale. Infatti diamo per scontato che le tecniche ai livelli sottostanti, comuni in genere a tutti i sistemi informatici, siano state implementate.

Tabella 4.1: Tecniche di Fault Tolerance per categoria di componenti

Componente	**Tecnica di Fault Protection**
Applicazioni, Middleware, Sistema Operativo	Ridondanza di sistemi, replicazione dei dati, clustering delle risorse, bilanciamento del carico, Tecniche di Failover geografico
Hardware	Ridondanza di componenti, Contratti di manutenzione appropriati, Componenti sostituibili a caldo (dischi e alimentatori *hot swappable*)
Ambiente fisico	UPS, generatore di corrente, sistema di refrigerazione ridondato, contratti di manutenzione adeguati

Tutti gli eventi che avvengono ai primi due livelli, Hardware e Ambiente fisico, devono essere adeguatamente monitorati. Alcuni di questi eventi, infatti, possono dare il via alle procedure di Fault Protection adottate dal livello superiore, altri potrebbero invece pregiudicarne l'esecuzione: in questi casi sarà necessario ricorrere a piani di Disaster Recovery affinché sia minimizzato il tempo di indisponibilità.

Il grafico in figura 4.1 descrive la probabilità che un evento si manifesti in funzione del danno che causa. Le tipologie di danni sono state suddivise in tre gruppi:

1. gli eventi che si manifestano più frequentemente sono quelli che statisticamente causano danni minori; per molti di questi è possibile adottare tecniche affinché il sistema nel complesso sia considerato disponibile da parte dei suoi utilizzatori anche se alcune sue componenti diventano temporaneamente inutilizzabili. Molte delle tecniche applicabili in questi casi si prestano per essere eseguite in automatico senza interruzione di servizio.

2. non sempre è possibile applicare tecniche di recovery automatiche senza interruzione di servizio. Per esempio, nel caso di dati corrotti, il servizio potrebbe interrompersi o manifestare un comportamento anomalo: in questi casi potrebbe essere necessario ripristinare i dati danneggiati utilizzando copie precedentemente archiviate.

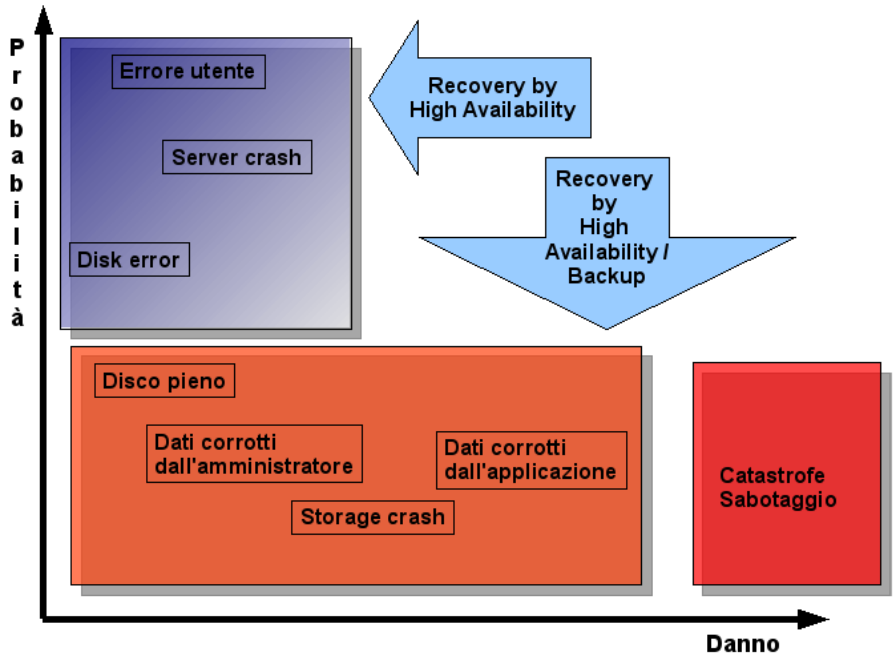


Figura 4.1: Grafico probabilità dell'evento in funzione del danno causato

3. per gestire l'impatto di eventi catastrofici occorre dotarsi di un piano di Disaster Recovery che descriva nel dettaglio le procedure da seguire per minimizzare l'indisponibilità dei sistemi e ripristinarli in tempi ragionevoli.

4.2.1 Round Robin DNS e Load Balancing

Il Domain Name System (DNS) [81] è un servizio utilizzato per la risoluzione di nomi di host in indirizzi IP e viceversa. Il servizio è realizzato tramite un database distribuito, costituito dai server DNS. L'operazione di convertire un nome in un indirizzo è detta risoluzione DNS; convertire un indirizzo IP in nome è detto risoluzione inversa. In figura 4.2 viene descritto il funzionamento della tecnica denominata *Round Robin DNS (RRDNS)*. Nel DNS è possibile attribuire più nomi allo stesso indirizzo IP (o viceversa) per rappresentare diversi servizi o funzioni forniti da uno stesso host (o più host che erogano lo stesso servizio).

Questa flessibilità risulta utile poiché facendo corrispondere più indirizzi IP a un nome, nel esempio *WebApp*, il DNS risolverà ogni richiesta da parte dei client: l'ordine con il quale gli indirizzi IP della lista vengono presentati è alla base del termine round robin. Ad ogni risposta, la sequenza di indirizzi IP della lista viene

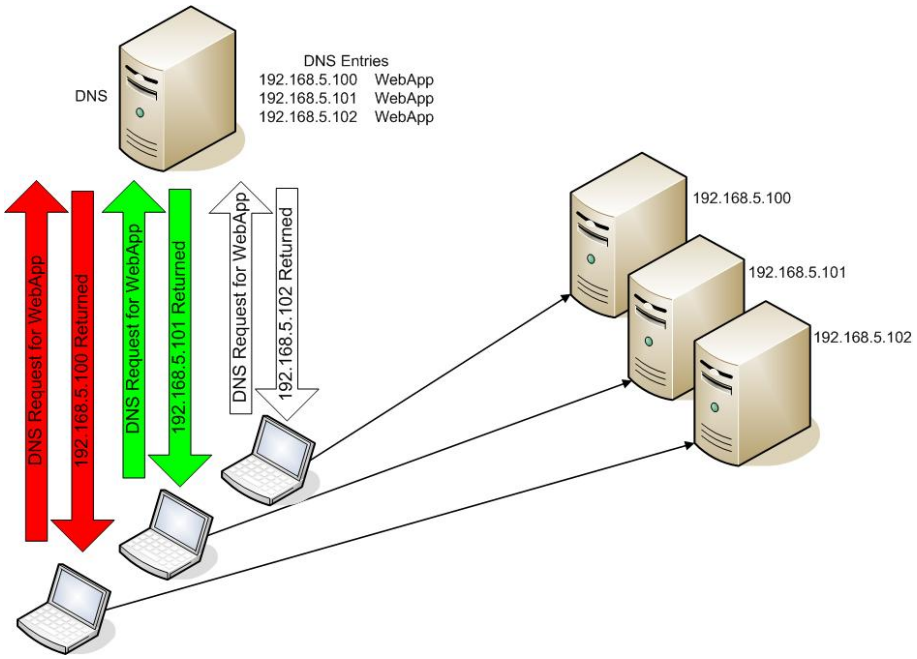


Figura 4.2: Esempio di funzionamento e di applicazione del Round Robin DNS

permutata. Poiché generalmente i client che hanno fatto la richiesta al DNS sono implementati in modo da utilizzare il primo indirizzo IP della lista per connettersi ad un dato servizio, in richieste consecutive i client eseguiranno tentativi di connessione utilizzando IP diversi, distribuendo in tal modo le richieste tra gli indirizzi IP disponibili; tuttavia è necessario assicurarsi che i diversi server siano sempre allineati, ovvero offrano esattamente lo stesso servizio ai client.

In realtà non esiste una procedura standard per decidere quale indirizzo verrà utilizzato, ma dipende dall'implementazione delle applicazioni. Alcune di esse tentano di riordinare la lista dando la priorità agli indirizzi appartenenti alle reti più 'vicine', altre invece utilizzano un approccio per tentativi: utilizzano uno alla volta gli indirizzi disponibili della lista nell'ordine in cui vengono presentati, passando all'IP successivo nel caso la loro richiesta non venga soddisfatta dopo un timeout di n secondi.

Questa tecnica è molto semplice, ma presenta diversi difetti, come ad esempio le problematiche legate alla gestione della cache sia da parte del server DNS che da parte dei client. Questi potrebbero salvare l'indirizzo IP risolto dal DNS allo scopo di riutilizzarlo per periodi di tempo più o meno brevi.

Inoltre, il RRDNS non sa se il servizio corrispondente ad uno degli indirizzi sia

realmente disponibile: in caso di indisponibilità, il DNS continuerà comunque a fornire l'IP ad ogni richiesta e i client tenteranno di connettersi al sistema indisponibile. Dunque RRDNS non è la scelta migliore come soluzione al problema del bilanciamento del carico poiché si limita ad alternare l'ordine degli indirizzi IP offerti ai client senza considerare aspetti quali la distribuzione geografica, l'eventuale stato di congestione della rete, il carico dei server, etc.

Spesso questa tecnica è utilizzata con l'aggiunta di un arbiter o un sistema di monitoring esterno, che si occupa di inserire o rimuovere gli indirizzi nel DNS a seconda delle politiche adottate.

- Nel caso in cui le richieste al servizio siano uniformemente distribuite, che il loro impatto sui server sia pressoché omogeneo e che i server stessi siano equivalenti, per semplicità può essere sufficiente avere un'applicazione che aggiunga, o rimuova a seconda dei casi, un indirizzo IP dalla lista controllando unicamente che il servizio sottostante sia disponibile o meno;

- In altri casi, come in quello rappresentato in figura 4.3 dove per esempio le richieste non sono tutte simili, ma producono un impatto diverso sui server che le ricevono o i server stessi non sono equivalenti dal punto di vista hardware, allora si rende necessario l'utilizzo di un *arbiter* che per ogni server prende in considerazione e valuta diversi parametri (come la disponibilità, il carico, lo spazio disco disponibile, i tempi di risposta, etc.) e ritorna un valore che verrà confrontato con quelli degli altri server: a questo punto l'arbiter inserirà nella lista del DNS, per esempio, soltanto gli indirizzi IP dei due server migliori, su un pool di quattro.

La maggiore affidabilità deriva dal fatto che l'indisponibilità di uno dei server non compromette la fornitura del servizio nel suo complesso. I sistemi di load balancing in genere sono integrati con sistemi di monitoraggio del cluster al fine di escludere automaticamente dal cluster i server non disponibili. In questo modo eventuali eventi di fallimento vengono così resi trasparenti agli utenti. Viene da sé che affinché l'architettura complessiva del servizio sia in High Availability, il sistema di load balancing deve basarsi su un cluster in HA.

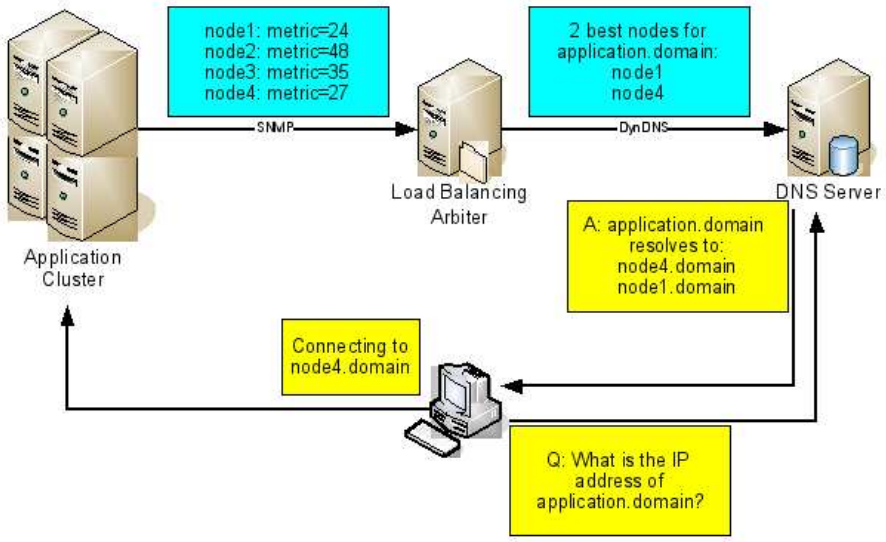


Figura 4.3: Esempio della tecnica di DNS Load Balancing con arbiter

4.2.2 La virtualizzazione delle risorse

La virtualizzazione consiste nella creazione di una versione virtuale di una risorsa normalmente fornita fisicamente ed è una tecnologia il cui uso si sta sempre più diffondendo nei centri di calcolo anche come metodologia di consolidamento dei server. Qualunque risorsa hardware o software, quali il sistema operativo, la memoria, lo spazio disco, etc, può essere virtualizzata. La virtualizzazione può essere effettuata sia a livello software che a livello hardware.

La virtualizzazione del sistema operativo è uno degli esempi più comuni. In modalità software, la virtualizzazione richiede un sistema operativo sul quale eseguire il software di virtualizzazione, il cui compito è quello di creare un layer che astragga dall'hardware e dal software sottostante e sul quale vanno in esecuzione le macchine virtuali.

L'architettura x86 non supporta nativamente la virtualizzazione, ma recentemente sono state introdotte estensioni all'insieme di istruzioni della CPU, sia per processori Intel che per AMD, che hanno permesso di sormontare vari problemi tecnici legati alla virtualizzazione. Varie sono le tecniche implementative disponibili a seconda dello scopo che ci si prefigge. Gli aspetti più rilevanti da considerare sono le prestazioni e la flessibilità necessarie nel proprio scenario applicativo. La flessibilità nella virtualizzazione rappresenta il grado di dipendenza della risorsa virtualizzata

dall'hardware e del sistema operativo, ovvero dalla macchina ospitante.

L'indipendenza viene fornita dal cosiddetto '*abstraction layer*' ed è inversamente proporzionale alle prestazioni della risorsa virtuale risultante. In altri termini, un abstraction layer complesso offre grande flessibilità, ma prestazioni più basse. Generalmente si distinguono tre tipi di virtualizzazione:

1. **Full virtualization (o native virtualization)**: emula completamente l'hardware e rende possibile ospitare sistemi operativi senza richiedere alcuna modifica. In questo caso, il codice binario è eseguito direttamente oppure viene tradotto o adattato per l'architettura del computer. La traduzione viene applicata laddove il codice di partenza non possa essere usato perché le istruzioni che esso contiene non sono virtualizzabili. Invece il codice a livello utente è sempre eseguito direttamente dal processore. Il sistema operativo contenuto nella macchina virtuale è indipendente dal fatto di trovarsi su un vero hardware perché l'abstraction layer fornisce periferiche e componenti hardware (cpu, ram, dischi, etc.) per ogni ambiente virtuale. Le prestazioni sono minori di quelle che si riscontrano in una ambiente non virtualizzato a causa dell'elaborazione aggiuntiva necessaria per la traduzione del codice.

2. **Operating system-level virtualization (o Single Kernel Image (SKI))**: si basa su un'esecuzione 'leggera' del sistema operativo in cui il sistema operativo 'master' si duplica in memoria. Il sistema operativo ospite esegue esattamente lo stesso sistema operativo del suo sistema principale: la differenza è che il kernel non viene rimandato in esecuzione.

3. **Para virtualizzazione**: prevede l'uso di un sistema operativo modificato (in genere i cambiamenti sono limitati al kernel e ad alcune librerie). Il sistema viene modificato perché alcune funzioni non sono compatibili con la virtualizzazione (come le chiamate di sistema, l'amministrazione della memoria, la gestione dell'orologio, etc.). Sistemi non modificabili, come la varie versioni di Microsoft Windows, non possono funzionare usando la para virtualizzazione. Le prestazioni sono simili a quelle ottenibili con un sistema operativo in funzione direttamente sull'hardware. La macchina virtuale non simula l'hardware in toto, ma offre specifiche API il cui supporto richiede però la modifica del sistema operativo ospite.

In questo lavoro di tesi la virtualizzazione è stata prevalentemente utilizzata come tecnica di ottimizzazione e consolidamento delle risorse. Tra le diverse tecnologie di virtualizzazioni disponibili, la scelta è ricaduta sul software Xen [82].

Xen

La para virtualizzazione è la tecnica adottata da **Xen**, un prodotto Open Source rilasciato sotto licenza GPL per piattaforma x86 e compatibili sviluppato presso il Computer Laboratory dell'Università di Cambridge. Xen non mira a creare un'emulazione dell'hardware di un generico computer x86, ma piuttosto di regolare e controllare l'accesso alle risorse fisiche della macchina reale da parte delle varie istanze virtuali. Questo tipo di approccio consente di contenere il decadimento delle prestazioni rispetto all'esecuzione non-virtualizzata, poiché le istruzioni provenienti dalle macchine virtuali vengono eseguite quasi tutte direttamente sul processore, evitando dunque l'intervento di un sistema operativo aggiuntivo che si ponga tra la macchina virtuale e le risorse fisiche. Tuttavia questo approccio richiede un adattamento del sistema operativo destinato a girare sulla macchina virtuale (*guest*) al fine di renderlo compatibile con Xen. Senza questo adattamento, alcune chiamate di sistema del kernel non sarebbero possibili. L'aspetto vantaggioso è che le applicazioni non necessitano ricompilazione, in quanto i kernel adattati a Xen espongono la stessa *Application Binary Interface (ABI)*. Intel ha contribuito al progetto Xen introducendo il supporto per la sua tecnologia VT-X denominata Vanderpool [83]: se eseguito su sistemi dotati di questa caratteristica, Xen permette di creare delle macchine virtuali che consentono ai sistemi operativi ivi installati di andare in esecuzione senza modifiche. Una tecnologia simile, denominata AMD-V [84], è stata sviluppata da AMD al fine di ottenere lo stesso risultato su processori AMD.

Ambiti applicativi

La tecnologia Xen è stata scelta perchè ben supportata dalle distribuzioni di sistema operativo basate su RedHat ES [85], come Scientific Linux [86], largamente utilizzate in ambito EGEE; uno dei vantaggi della virtualizzazione, a fronte di una leggera perdita prestazionale rispetto alla macchina fisica, che da recenti evidenze sperimentali è stato visto essere nell'ordine del 5% [87], è la completa indipendenza dall'hardware installato. In altre parole, le macchine virtuali sono simulate con hardware standard

e ciò le rende totalmente estranee al reale hardware della macchina su cui viene eseguito il software di virtualizzazione.

Questa caratteristica agevola le operazioni che richiedono interventi di natura sistemistica a livello hardware: per esempio, è possibile gestire interventi che richiederebbero lo spegnimento di un server, come upgrade o sostituzione dei componenti hardware, sul quale sono in esecuzione macchine virtuali, semplicemente migrando le macchine virtuali su un altro server, anche se con caratteristiche hardware diverse. Tale migrazione, sotto certe condizioni, può essere fatta 'a caldo', cioè mentre le macchine virtuali sono in esecuzione, in maniera totalmente trasparente agli utenti e senza interruzioni di servizio.

Un altro vantaggio importante della virtualizzazione è quello di poter snellire drasticamente le procedure di backup e disaster recovery: mantenendo una copia giornaliera delle immagini virtualizzate delle risorse, è possibile un rapido e semplice ripristino in caso di un qualsiasi malfunzionamento.

Infine, la virtualizzazione permette di concentrare i servizi gestiti su un numero inferiore di server e quindi di abbattere il costo dell'infrastruttura hardware sia in fase di acquisto che in fase di gestione e manutenzione, mantenendo al tempo stesso i servizi tra loro indipendenti utilizzando macchine virtuali distinte.

4.3 Service Level Agreement

Il ***Service Level Agreement (SLA)*** è un documento che assume la connotazione di un contratto e ha l'obiettivo di definire una serie di vincoli inerenti l'erogazione dei servizi (qualità del servizio offerto, parametri di valutazione dei risultati, verifica dei risultati, etc.) e di pattuirne i relativi prezzi richiesti dal fornitore. Il contratto di SLA rappresenta, pertanto, uno strumento di definizione e valutazione delle esigenze dell'organizzazione in termini di servizi richiesti e conseguentemente di valorizzazione degli stessi da parte del fornitore che li offre.

Per la definizione del SLA si rende necessaria una analisi dei servizi esistenti, con particolare riguardo alla loro gestione, organizzazione e parametrizzazione: ciò permette di definirne le caratteristiche, i livelli di servizio e di qualità esistenti, le eventuali lacune da colmare, le difformità fra i vari servizi. Nel SLA sono individuabili:

- regole e parametri relativi alla funzionalità operativa dei servizi (orario di erogazione, uptime, manutenzione pianificata e straordinaria, gestione di anomalie ecc.);
- regole e parametri relativi all'utilizzo dei servizi (attivazione, monitoraggio, helpdesk, comunicazioni ecc.).

Tabella 4.2: Livelli di Availability

Availability(%)	Downtime year	Downtime month	Downtime week
90%	36.5 days	72 hours	16.8 hours
95%	18.25 days	36 hours	8.4 hours
98%	7.30 days	14.4 hours	3.36 hours
99%	3.65 days	7.20 hours	1.68 hours
99.5%	1.83 days	3.60 hours	50.4 min
99.9% ('three nines')	8.76 hours	43.2 min	10.1 min
...			
99.9999% ('six nines')	31.5 s	2.59 s	0.605 s

Parlando di availability, è stato precedentemente spiegato nella sezione 4.1.1, che un modo comune di misurarla è utilizzando una percentuale che identifica il periodo di disponibilità di un dato servizio in un determinato arco di tempo. Questo tipo di notazione è utilizzata nella definizione di SLA: nella tabella 4.2 vengono riportati per le SLA più comuni i corrispondenti periodi di indisponibilità ammessi per anno, mese, settimana, considerando una continuità di servizio di 24 ore giornaliere sull'intero arco settimanale (24x7).

4.4 Specifiche per Agreement via Web service

Nell'Open Grid Forum è attivo un gruppo di lavoro, denominato *Grid Resource Allocation and Agreement Protocol (GRAAP)*, che ha rilasciato la specifica di un protocollo Web service [88] con lo scopo di standardizzare la terminologia, i concetti, le strutture dati e le operazioni necessarie per il controllo e la gestione di Agreement in un ambiente di calcolo distribuito. L'Agreement viene stipulato tra due parti, che possono essere per esempio rappresentate da un centro di calcolo che fornisce dei servizi e una comunità scientifica (nel progetto EGEE si tratterebbe di un Agreement tra sito e VO), o da un centro di calcolo e il Resource Operation Center a cui afferisce (sito e ROC), etc. La specifica contiene le seguenti indicazioni:

- una grammatica XML per definire un agreement;
- un protocollo per generare un agreement da un template;
- un insieme di operazioni per la gestione del ciclo di vita di un agreement.

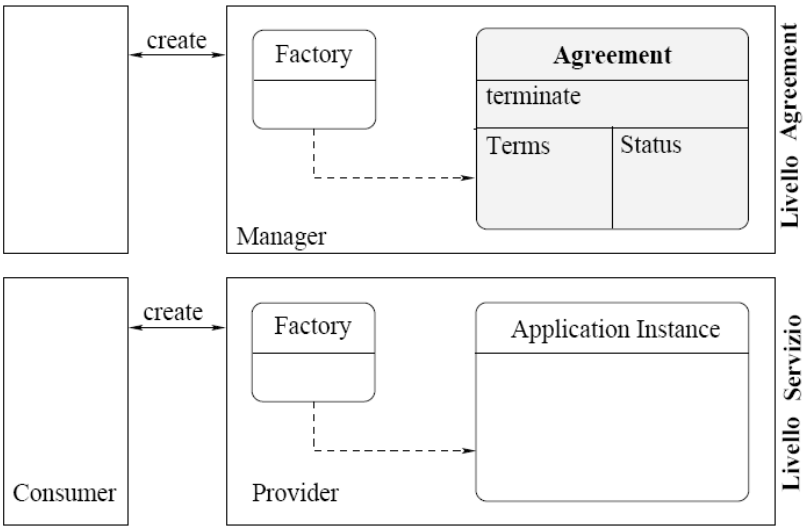


Figura 4.4: Modello a layer dell'architettura WS-Agreement proposta da OGF. Fonte [88]

L'architettura di WS-Agreement è organizzata su due livelli, come illustrato in figura 4.4, in modo da separare la parte relativa all'agreement da quella relativa al servizio. In questo modo si costruisce un'architettura indipendente dal servizio che può essere applicata ad ogni tipo di dominio applicativo. Il modello comprende una parte relativa al consumer del servizio e una relativa al provider, le due parti sono ulteriormente divise in due livelli:

- livello servizio: rappresenta il livello specifico del servizio che si vuole offrire;
- livello agreement: rappresenta il livello indipendente dall'applicazione che implementa WS-Agreement. Consiste di un interfaccia basata su Web service che serve per creare, monitorare e terminare agreement. Inoltre questo livello espone le informazioni sulla tipologia dei servizi disponibili e le relative SLA offerte.

La creazione di un agreement può avvenire da zero o da un agreement precompilato, chiamato *agreement template*, in modo tale da permettere al consumer di

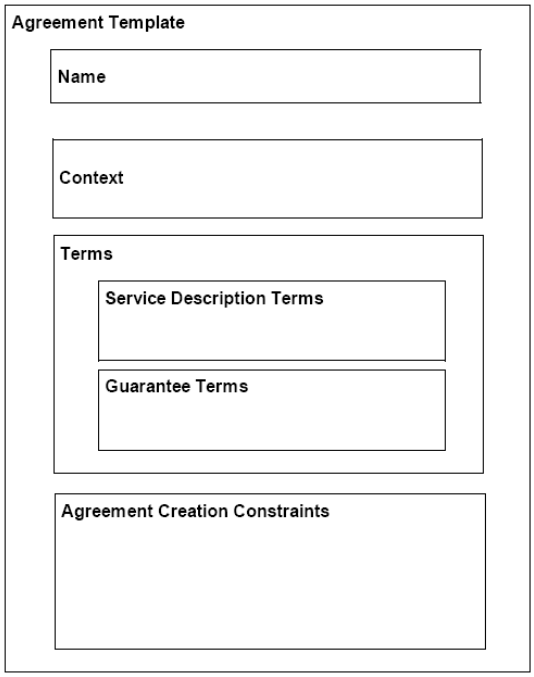


Figura 4.5: Struttura di un Agreement template secondo il modello WS-Agreement. Fonte [88]

scegliere una delle combinazioni di servizio e garanzia che offre il provider. Il provider compila dei template e li deposita in una collezione, il consumer accede ai template, ne sceglie uno e lo compila seguendo le indicazioni fornite dal provider.

Con riferimento alla figura 4.5, un agreement template è composto dalle seguenti sezioni:

- *Name*: contiene il nome opzionale dell'agreement;
- *Context*: contiene il contesto in cui si specificano i meta dati dell'agreement, come per esempio i contraenti e la validità temporale;
- *Terms*: contiene due sezioni organizzate con una struttura ricorsiva, basata su operatori logici, che permette di stillare delle combinazioni di accordi. La sezione *service description terms (SDT)* specifica le operazioni del servizio che saranno eseguite se la conformità all'accordo sarà verificata. La definizione delle operazioni in questa sezione e la loro corrispondenza ad un servizio reso disponibile dal fornitore può essere espressa tramite riferimenti formali in qualsiasi linguaggio accettato, oppure tramite riferimenti non formali ma deri-

vanti dalle proprietà o dalla descrizione del servizio. La specifica non definisce il linguaggio per descrivere i riferimenti finali al servizio, né il meccanismo per tracciare un servizio in base all'identificazione delle sue proprietà; essa fornisce solo la rappresentazione astratta che può essere implementata con elementi standard o specifici del dominio. La sezione *guarantee terms (GT)* definisce la modalità di fornitura delle operazioni del servizio. In questa sezione sono riportati i *Service Level Objectives (SLO)* (ad esempio il tempo medio di risposta per i servizi descritti nelle sezioni SDTs), le condizioni che devono essere rispettate perché un dato SLO possa essere fornito (ad esempio, il tasso di richiesta deve essere inferiore ad una soglia predefinita) e un valore che rappresenta l'importanza di raggiungere l'obiettivo descritto dal SLO nell'accordo (ad esempio, un valore alto implica che il fornitore usi più risorse per eseguire le operazioni descritte nella sezione SDT, nel caso in cui le risorse disponibili non possano soddisfare il SLO definito).

- *Constraint*: si presenta solo nella struttura di un template e ha lo scopo di imporre vincoli ai valori accettati per riempire i campi dei termini presentati sopra. Per esempio, il valore accettato per il numero di cpu usate per soddisfare i SLO (e.g. tempo medio di risposta) definiti per la fornitura delle operazioni del servizio, non può avere valore più di sette. Offerte di accordi che richiedono un numero di cpu superiore a sette non saranno accettate. Questa sezione si usa per verificare la conformità di una richiesta per la creazione di un accordo con un dato template.

Gli stati attraverso cui può passare un Agreement durante il suo ciclo di vita possono essere visualizzati attraverso il diagramma di figura 4.6. In esso si evince come **Pending**, **Observed** e **Rejected** sono gli unici possibili stati iniziali di un Agreement, a cui si arriva dallo stato di transizione **OfferReceived**, non esposto. Si nota inoltre, come dagli stati **PendingAndTerminating** e **ObservedAndTerminating** si può ritornare nei rispettivi stati di partenza, se la richiesta di terminazione fatta dall'iniziatore dell'accordo viene respinta.

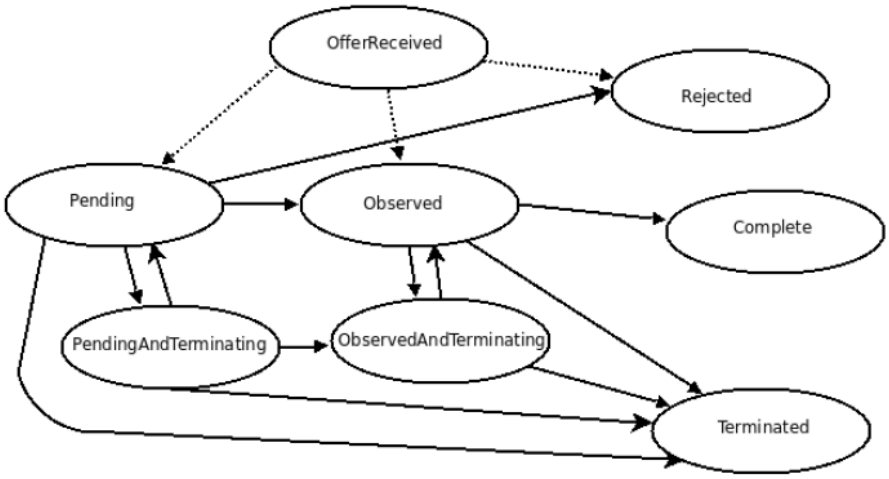


Figura 4.6: Diagramma degli stati di un Agreement secondo il modello WS-Agreement. Fonte [88]

4.5 Il Service Level Agreement di EGEE

L'infrastruttura di calcolo Grid del progetto EGEE è composta da un insieme di risorse, applicazioni e servizi disponibili per le comunità scientifiche Europe e di altre Nazioni. Il SLA definito nel progetto ha lo scopo di specificare i vincoli ai *ROC* e ai *siti*[2] per assicurare una infrastruttura di calcolo distribuito affidabile e disponibile. Non sono prese in considerazioni le relazioni che possono esistere tra VO e siti: questi accordi potranno essere dettagliati in specifiche SLA tra siti e VO. Sebbene nella definizione della SLA non siano seguite le specifiche proposte dall'architettura precedentemente descritta, possiamo individuare una serie di elementi che per loro natura potrebbero essere facilmente definiti in termini di WS-Agreement. Nel SLA di EGEE [89] vengono definite:

- Le parti contraenti, le loro responsabilità e i requisiti per fare parte dell'infrastruttura.
- La durata del contratto. Il contratto vale per tutto il tempo in cui un sito fa parte dell'infrastruttura.
- Le procedure per modificare una o più parti del contratto stesso.

[2] Centri di calcolo che fanno parte dell'infrastruttura

- I servizi, le metriche e gli algoritmi per calcolare l'Availability di un sito;

4.5.1 Le parti contraenti: responsabilità e requisiti

I ROC

Come è stato precedentemente detto, EGEE è suddivisa in federazioni regionali, ognuna delle quali è identificata da un ROC[3]. Le principali responsabilità dei ROC previste dal SLA sono:

- fornire un sistema di supporto, sul quale registrare gli amministratori di sistema dei siti che fanno parte del ROC, e mediante il quale vengano gestite le operazioni stesse di supporto generiche e specialistiche per i problemi che gli amministratori dei siti non riescano a risolvere.
- assicurare che i ticket vengano aggiornati e seguirne l'evoluzione fino alla risoluzione del problema, rispondere entro quattro ore ai ticket aperti dagli amministratori dei siti[4].
- eseguire le procedure di gestione e controllo dell'infrastruttura concordate a livello di progetto [91].

I siti

I centri di calcolo mettono a disposizione risorse di calcolo e/o di archiviazione all'infrastruttura mediante i servizi offerti dal middleware gLite. Per fare parte dell'infrastruttura, un sito deve almeno essere costituito da:

- un site-BDII, discusso nella sezione 3.3.1;
- un Computing Element (mediante il quale accedere a risorse di calcolo per almeno otto slot, ovvero in grado di poter eseguire otto job contemporaneamente) o uno Storage Element (con capacità di almeno 1 TB);
- un *sistema di accounting*[5] dell'uso delle risorse.

Tra le principali responsabilità definite nel SLA per i siti ci sono:

[3] La lista dei ROC è disponibile in [90]
[4] Per le metriche di gestione dei ticket si fa riferimento ai dati del sistema GGUS
[5] I siti del ROC italiano usano il sistema DGAS [92]

- l'adesione alle procedure di gestione del sito concordate a livello di progetto [91];
- l'aggiornamento delle informazioni sul sito stesso contenute nel GOCDB;
- l'adesione alle norme e alle procedure riguardo alla sicurezza informatica del sito [93];
- l'impegno a tenere i servizi del middleware gLite aggiornati alle versioni più recenti;
- l'obbligo di rispondere ai ticket entro quattro ore.

4.5.2 Il calcolo dell'Availability

In EGEE vengono utilizzate diverse applicazioni per il controllo e la gestione dell'infrastruttura Grid. Per valutare l'Availability dei siti, ci si avvale dei risultati dei test effettuati attraverso SAM (*Service Availability Monitoring* [94]), un sistema centralizzato composto da un insieme di test sottomessi a intervalli regolari, un database nel quale vengono archiviati i risultati e una interfaccia Web per visualizzarli. Questi risultati vengono elaborati dall'applicazione Gridview [95] in modo tale da recuperare i dati necessari alla verifica delle metriche definite nella SLA. Per il calcolo dell'Availability totale di un sito, vengono utilizzate le seguenti definizioni:

- ***istanza di un servizio***: rappresenta la singola risorsa in Grid, come per esempio un particolare Computing Element o Storage Element in un sito;
- ***servizio***: è un insieme di istanze di servizi della stessa tipologia. Per esempio, un insieme di istanze di Computing Element di un sito è da considerarsi come il servizio Computing Element di quel sito. Un servizio può quindi essere composto da una o più istanze.
- ***sito***: è rappresentato da un insieme di servizi, come Computing Element, Storage Element, Storage Resource Manager, LCG File Catalog, File Transfer Service.
- ***stato***: lo stato dell'istanza del servizio, del servizio o di un sito è lo stato di quella entità in un dato momento, in accordo con i risultati dei SAM test. Gli stati possibili sono quelli riportati nella tabella 4.3.

Tabella 4.3: Definizione dei possibili stati per le istanze, i servizi e i siti

Stato	**Significato**
UP	L'istanza, il servizio o il sito hanno passato i SAM test con successo
DOWN	L'istanza, il servizio o il sito hanno fallito i SAM test
SCHEDULED DOWN	L'istanza, il servizio o il sito sono in manutenzione programmata (informazione contenuta nel GOCDB)
UNKNOWN	I SAM test per l'istanza, il servizio o il sito non sono disponibili
DEGRADED	Una o più istanze, ma non tutte, componenti un singolo servizio hanno fallito i SAM test

In un dato momento, un sito sarà considerato 'UP' se lo saranno tutti i servizi di cui è composto. Un servizio è considerato 'UP' se lo sono tutte le singole istanze che lo compongono. Nel caso un servizio sia in stato 'DEGRADED', cioè quando una o più istanze, ma non tutte, componenti un singolo servizio hanno fallito i SAM test, il servizio nel suo insieme sarà considerato 'UP'. La SLA di EGEE prevede che un sito debba essere:

1. **AVAILABLE** almeno per il 70% del tempo su base mensile;

2. **RELIABLE**, definita dalla formula 4.6, almeno per il 75% del tempo su base mensile;

$$Reliability = \frac{Availability}{Availability - Unscheduled\ Downtime} \tag{4.6}$$

I periodi di indisponibilità di una istanza, servizio o dell'intero sito devono essere inseriti nel GOCDB: i downtime pianificati incidono quindi negativamente sull'Availability, ma non sulla Reliability.

Poiché i SAM test per essere eseguiti necessitano di credenziali, in conformità di quanto discusso nella sezione 3.2.1 sull'autenticazione e l'autorizzazione, occorre tenere presente che i test sono sempre relativi a queste credenziali, in particolare sono sempre relativi alla VO utilizzata per richiedere l'autorizzazione all'uso delle risorse. Può quindi capitare il caso in cui, per esempio, su uno Storage Element

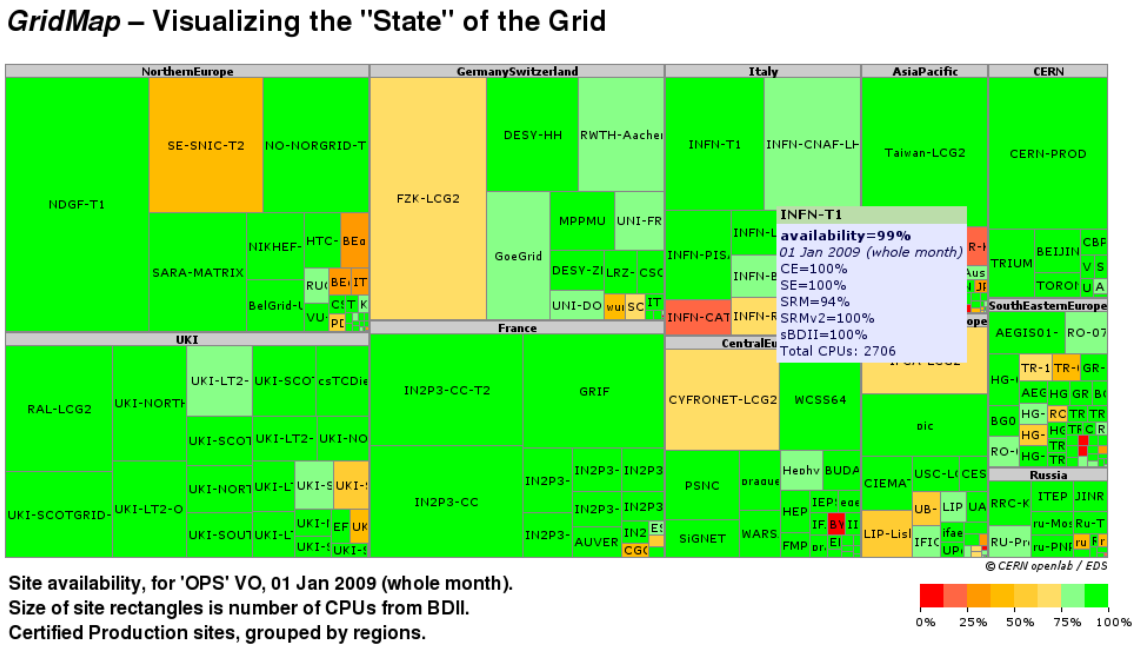


Figura 4.7: Availability dei siti componenti l'infrastruttura EGEE nel mese di Gennaio 2009. Fonte [96]

venga eseguito con successo il test utilizzando le credenziali della VO OPS[6], mentre lo stesso test fallisca se si usano credenziali con una VO diversa. SAM e Gridview permettono di calcolare anche le Availability dei siti per VO diverse da OPS: ogni VO può selezionare quali test di SAM debbano essere considerati per il calcolo. Questi risultati possono essere quindi utilizzati nel caso di SLA tra VO e siti. In questa tesi, ove non diversamente specificato, si considererà sempre l'Availability relativa alla VO OPS.

Availability e Reliability su base oraria, giornaliera, settimanale e mensile possono essere visualizzate attraverso l'applicazione GridMap [96]: l'applicazione raggruppa i siti, rappresentati da rettangoli di dimensioni proporzionali al numero di slot disponibili, nei rispettivi ROC di appartenenza. Il colore rappresenta il grado di Availability totale dei siti, per ogni sito è possibile evidenziare l'Availability di ogni servizio: in figura 4.7 è riportata la visualizzazione dell'Availability del mese di Gennaio 2009, con il dettaglio dell'Availability dei servizi offerti dal sito INFN-T1.

[6]La VO OPS deve essere supportata in tutti i siti e su tutti i servizi, è la VO utilizzata per eseguire i test sui quali si calcola l'Availability definita nel SLA di EGEE.

Capitolo 5

Tecniche di High Availability per servizi Grid

"That's not a bug, that's a feature!"
The canonical first parry in a debate about a purported bug.

All'aumentare del grado di complessità e della scala della rete, dei sistemi di calcolo e di archiviazione e dell'infrastruttura informatica in generale, anche le attività più semplici di controllo possono diventare problematiche. Ad esempio, per sapere se un determinato gruppo di servizi è regolarmente in funzione, o se il livello di carico è tanto alto da pregiudicarne il normale funzionamento, non è certo efficiente controllare su tutti gli host che compongono il servizio lo stato del sistema autonomamente e in modo pro attivo. Sono dunque necessari sistemi di monitoraggio più sofisticati in grado di fornire una '*istantanea*' dello stato dei servizi e in generale delle risorse, mantenendo uno storico e fornendo strumenti di correlazione tra cause ed effetti, importanti per esempio quando un servizio è costituito da diverse componenti.

Dunque in Grid, i sistemi di monitoring uniti ad un sistema di notifica (o allarmistica), sono necessari per consentire la rilevazione tempestiva di anomalie e per intraprendere le opportune azioni correttive, eventualmente anche in modalità automatica ove possibile, indagando nel contempo sulle cause. Non tutti i servizi presenti nel middleware gLite sono però 'disegnati' in modo tale da permettere che l'applicazione di tecniche di High Availability abbia come risultato una gestione dei fallimenti delle istanze dei servizi trasparente agli utenti. Per alcuni servizi, data la loro semplicità, ciò è possibile (per esempio nel caso del servizio BDII). Altre applicazioni sviluppate recentemente, come FTS e StoRM, hanno una architettura

modulare che permette l'applicazione di diverse tecniche di fault tolerance ai singoli componenti del servizio: generalmente questi servizi sono gestibili in maniera completamente trasparente agli utenti. Il servizio VOMS può essere configurato in modo tale da prevedere un intrinseco meccanismo di failover: in caso di più istanze di servizi VOMS disponibili per una data VO, se l'istanza primaria non dovesse essere disponibile, si ottiene un messaggio di errore, ma la richiesta viene automaticamente reindirizzata ad eventuali istanze secondarie. Il WMS è invece un esempio di servizio nel quale la gestione dei fallimenti al momento non è trasparente agli utenti per motivi architetturali del servizio stesso. Nel seguito di questo capitolo vengono illustrate le strategie di fault tolerance applicate a diversi servizi del middleware gLite.

In questa tesi è stato preso in considerazione Nagios [97], uno degli strumenti di monitoraggio e allarmistica open source più diffusi, conosciuto in ambito aziendale ed utilizzato in molte delle più grandi società ed organizzazioni in tutto il mondo. In particolare se ne presenterà l'architettura, il funzionamento, e l'applicazione in Grid per le attività di controllo, allarmistica e recovery automatico di numerosi servizi del middleware gLite.

5.1 Un sistema di monitoring e allarmistica: Nagios

Nagios funziona in ambiente Linux/Unix e permette di integrare diversi sistemi per la notifica degli eventi: email, messaggistica istantanea, SMS. L'architettura di Nagios è composta da varie componenti.

1. Il nucleo principale ha il compito di eseguire i controlli che sono stati definiti con una certa frequenza temporale, di notificare il cambiamento di stato di un servizio e, in caso di fallimento, di eseguire le procedure di *escalation* e di *recovery* eventualmente predefinite, attraverso l'esecuzione di azioni predisposte (*event handler*), come per esempio la rimozione di un IP dal DNS.

2. Una interfaccia Web in grado di visualizzare lo stato degli host, dei servizi controllati e del sistema Nagios in generale. Inoltre è possibile visualizzare la schedula dei controlli, inserire downtime (e quindi sospendere temporaneamente un controllo) o forzare l'esecuzione immediata di un check. Vista la

sensibilità delle informazioni accessibili e manipolabili tramite questa interfaccia, è possibile limitarne l'accesso ad un ristretto gruppo di utenti locali mediate l'uso del certificato digitale personale.

3. Un insieme di *plugin*, cioè applicazioni e script, che permettono da linea di comando di controllare lo stato di un servizio. Nagios propone un insieme di plugin generici, ma è molto semplice crearne di nuovi utilizzando un qualsiasi linguaggio di programmazione.

Tabella 5.1: Corrispondenza tra i valori di ritorno dei plugin di Nagios e i corrispondenti stati dei servizi e degli host

Valore di ritorno del plugin	**Stato del Servizio**	**Stato dell'host**
0	OK	UP
1	WARNING	UNREACHABLE
2	CRITICAL	DOWN/UNREACHABLE
3	UNKNOWN	DOWN/UNREACHABLE

Nagios determina lo stato di un host o di un servizio valutando il valore di ritorno del plugin eseguito: nella tabella 5.1 sono mostrati i valori con i corrispondenti stati per i servizi e gli host.

Oltre al valore di ritorno (obbligatorio), i plugin possono avere come risultato altri due elementi:

- una stringa testuale, utilizzata generalmente per riportare dettagli sul risultato del controllo, come per esempio un messaggio di errore;
- una stringa numerica, che può essere per esempio utilizzata per riportare i tempi di risposta o esecuzione del plugin al fine di rilevare anche problemi di congestione o carico elevato del servizio.

Nagios permette di definire delle dipendenze tra servizi e host. Questa proprietà è molto utile in particolare per definire due tipi di relazioni:

- *host-host*: questo tipo di relazione permette di descrivere le dipendenze tra i server e gli apparati di rete a cui sono connessi. In questo modo vengono rappresentate in Nagios le connessioni di rete così come si presentano nella

realtà e di conseguenza si riescono meglio a correlare tra loro diversi eventi. Per esempio, nel caso di rottura di uno switch, Nagios rileverà l'irraggiungibilità di tutti i server collegati, ma notificherà solo l'indisponibilità dello switch, in quanto causa del problema. Inoltre tutti i controlli dei server coinvolti verranno di conseguenza interrotti.

- *servizio-servizio*: questo tipo di relazione ha il pregio di aggiungere una logica per definire correlazioni causa/effetto. Si pensi per esempio ad un sito Web il cui funzionamento si basa su un database, come nei moderni CMS[1]. Nel caso in cui il database diventasse indisponibili, i contenuti del sito Web non sarebbero più fruibili, mentre l'applicazione Web potrebbe o meno essere operativa. Configurando opportunamente una relazione tra il servizio database e il server Web, Nagios sospenderà i controlli sullo stato del server Web, fino a quando il database non viene ripristinato, notificando solo l'indisponibilità di quest'ultimo.

Entrambe queste relazioni sono utilizzate per ottimizzare i controlli e le notifiche per i servizi gLite.

5.1.1 Controlli attivi, passivi e l'integrazione con un database

Nagios permette di definire due modalità di controllo:

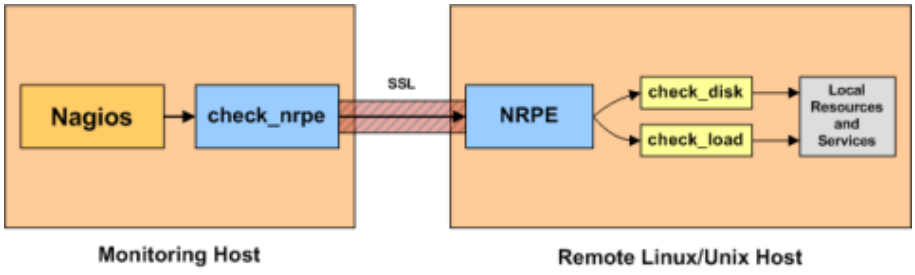


Figura 5.1: Schema di funzionamento dei controlli attivi locali con NRPE.

- I **controlli attivi** sono eseguiti attivamente da Nagios e possono essere remoti o locali. *I controlli remoti* sono eseguiti da Nagios, tramite plugin, senza l'ausilio di altre applicazioni. Rientrano in questa categoria i controlli sulle porte

[1] Content management system

di un host per verificare se il servizio che utilizza quella porta di comunicazione sia in esecuzione e risponda alle richieste. Per esempio il controllo di un server Web potrebbe essere fatto con il plugin di Nagios che esegue richieste sulla porta su cui opera il servizio, di default la porta 80. Invece, come illustrato in figura 5.1, se si volesse controllare il carico della cpu di un server o lo stato del suo disco, sarebbe necessario eseguire gli opportuni plugin localmente alla macchina che si vuole controllare. Questi check sono possibili utilizzando l'applicazione *NRPE*: questo demone riceve le richieste, criptate con protocollo ssl, dal server Nagios, le esegue localmente, e ritorna il risultato.

- I **controlli passivi** non vengono eseguiti direttamente da Nagios. Come illustrato in figura 5.2, la logica è inversa rispetto a NRPE. Sul server Nagios viene installata l'applicazione *NSCA* che rimane in attesa dei risultati dei controlli eseguiti da terze parti. Per questi controlli non sono necessari script sul modello dei plugin di Nagios: solo la sintassi con la quale si invia il risultato a Nagios deve essere corretta affinché il risultato sia valutato. Sul server Nagios è possibile definire un tempo entro il quale il risultato del controllo passivo è atteso. Nel caso il risultato non arrivi, Nagios può essere impostato per eseguire un controllo attivo, oppure assegnare uno stato al servizio, per esempio 'UNKNOWN'. Questo tipo di controlli hanno il pregio di lasciare la massima libertà nella loro definizione e alleggerire il carico del sistema Nagios. Sono utilizzati, per esempio, in presenza di un arbiter:

  1. l'arbiter colleziona le metriche e dei server componenti un cluster;
  2. in base alla logica implementata al suo interno, assegna uno stato a ciascun server;
  3. comunica lo stato a Nagios che, a sua volta, potrà inviare allarmi, tentare recovery del servizio o intraprendere qualunque altra azione che sia stata predefinita.

  Questo scenario è stato applicato per esempio nel controllo dei servizi di WMS e L&B, descritti successivamente.

Tutti i risultati dei controlli di Nagios sono raccolti in una serie di file di log. Può risultare molto utile averli a disposizione in un database: questo potrebbe essere

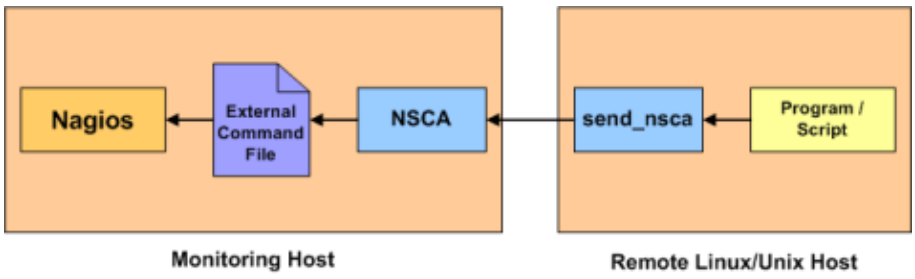


Figura 5.2: Schema di funzionamento dei controlli passivi con NSCA.

utilizzato per fare correlazioni, visualizzazioni e report diversi da quelli definiti dalla logica di Nagios. Per questo scopo è stato utilizzato un modulo, *NDOUtils*, che ha lo scopo di inserire in un database MySQL, in maniera asincrona, le informazioni dei controlli presenti nei log di Nagios.

Le sezioni che seguono illustrano vari esempi di applicazione che sono stati ideati per implementare soluzioni di High Availability per vari servizi Grid ad elevata criticità.

5.2 Soluzioni High Availability per il servizio BDII

Come il servizio DNS è una delle componenti funzionali fondamentali in Internet, il sistema informativo è uno dei servizi più critici per l'infrastruttura Grid. Il sistema informativo, come evidenziato nella sezione 3.3.1, è basato sul BDII. L'architettura Grid prevede che le singole risorse pubblichino le proprie caratteristiche, i siti aggreghino le informazioni delle risorse ospitate e pubblichino questo aggregato. Infine il top-level BDII interroga i BDII dei siti, aggregando a sua volta le informazioni che sono così messe a disposizione degli utenti e dei servizi che ne fanno richiesta.

Questa architettura implica quindi la necessità da parte degli utenti e delle risorse di conoscere quale sia l'hostname del top-level BDII da interrogare. Tipicamente, questa informazione non è soggetta a frequenti cambiamenti nel tempo e fa parte di quell'insieme di configurazioni di default che utenti e site manager non hanno motivo di modificare nella loro attività quotidiana.

A tutti i livelli (risorsa, sito, core), il servizio BDII, ad ogni interrogazione, deve fornire informazioni complete e aggiornate con tempi di latenza contenuti:

- La completezza delle informazioni è essenziale per la corretta descrizione dei

servizi offerti. Per esempio, se il sistema informativo di uno Storage Element pubblica l'informazione che la VO *ATLAS* è da esso supportata, ma non fornisce indicazioni su quale sia il punto di accesso per tale VO, la risorsa sarà di fatto inutilizzabile.

- Le informazioni pubblicate devono rispecchiare il più possibile lo stato attuale della risorsa stessa. Prendendo sempre come riferimento uno Storage Element, l'informazione riguardo allo spazio disponibile deve essere aggiornata di frequente, per evitare, per esempio, che ad un certo punto i trasferimenti falliscano perché lo spazio su disco è esaurito, mentre il sistema informativo riporta la presenza di spazio residuo.

- Il BDII è utilizzato da altri servizi e deve rispondere alle richieste evitando che queste vadano in timeout, nel qual caso le informazioni recuperate dai client potrebbero essere incomplete (o invalide) e la risorsa (o il sito) potrebbero venire considerati indisponibili o comunque si potrebbero manifestare problemi nel loro utilizzo.

L'architettura del servizio BDII e i dati del suo utilizzo in ambiente di produzione indicano che il servizio, per funzionare correttamente, necessita di risorse adeguate in termini di potenza di cpu e quantità di memoria, mentre al contrario lo spazio disco utilizzato è minimo. L'adeguatezza delle risorse è direttamente proporzionale alla quantità di informazioni contenute e al numero di richieste che devono essere soddisfatte. Per tale motivo, i problemi più frequenti si presentano nei BDII di siti che pubblicano molte informazioni e che sono interrogati più frequentemente, in particolare quindi nelle istanze BDII dei siti di grandi dimensioni e nel top-level BDII. Quest'ultimo contiene le informazioni su tutti i siti e sui servizi dell'infrastruttura e viene potenzialmente interrogato per ogni operazione.

Nella infrastruttura di produzione EGEE, per convenzione, ogni Grid regionale mette a disposizione un servizio top-BDII: ovvero l'infrastruttura è stata partizionata in domini, in modo che ogni dominio Grid corrisponda ad un proprio top-level BDII. Questo implica che tale istanza deve contenere tutte le informazioni dell'infrastruttura EGEE, ma viene utilizzata solo da una frazione delle risorse e degli utenti, ovvero quelli della regione corrispondente (anche se nulla vieta che essa possa ricevere richieste provenienti da altri domini). Tutti i top-level BDII, afferendo alle

medesime sorgenti di informazioni (i BDII di tutti i siti in produzione), contengono gli stessi dati, con eventuali piccole differenze di sincronizzazione dell'ordine di qualche minuto, essendo le operazioni di aggiornamento dei BDII asincrone.

Le raccomandazioni per la gestione di questo servizio prevedono:

1. Un adeguato sistema di monitoraggio: utilizzo della cpu, della memoria e i tempi di risposta del servizio sono le grandezze tipiche da tenere sotto controllo;

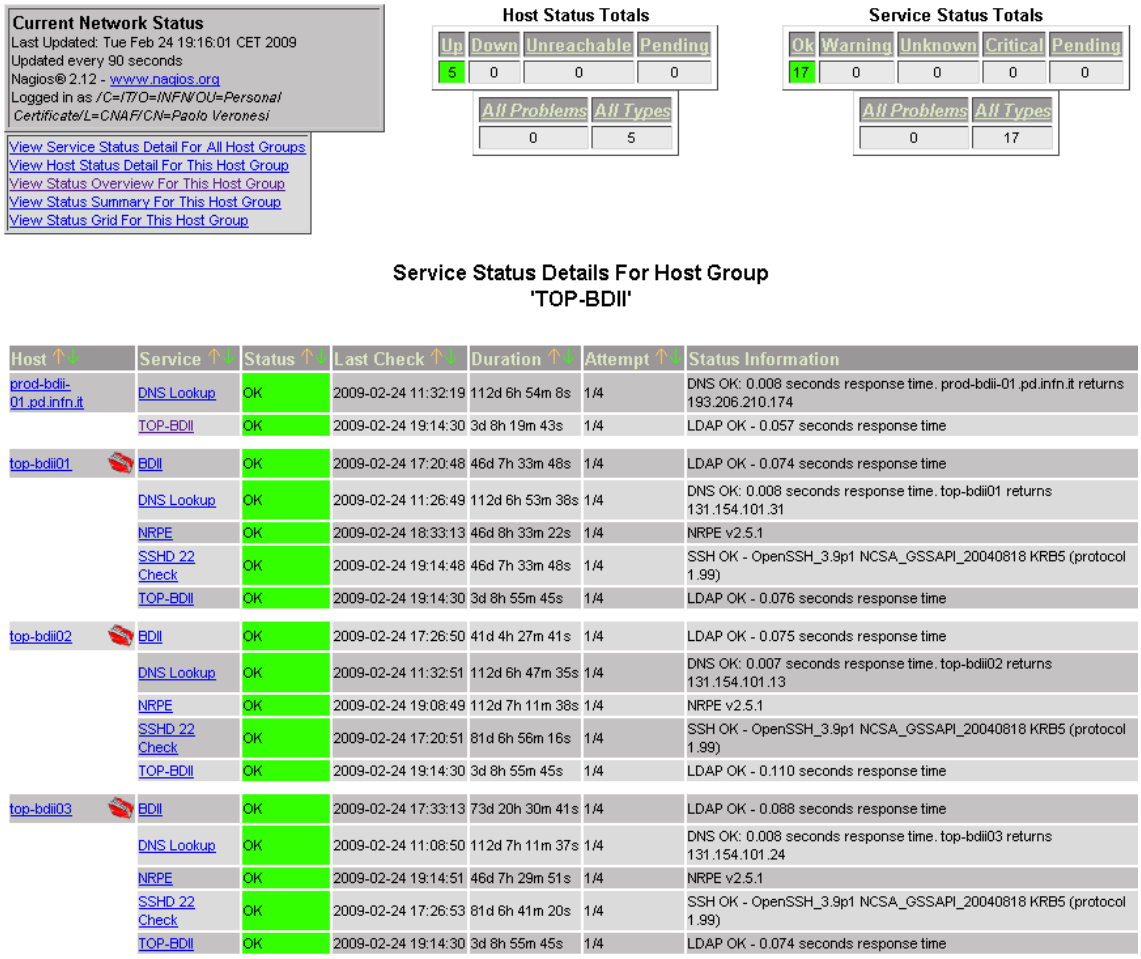
2. Un efficiente sistema di allarmistica: eventuali anomalie o malfunzionamenti del servizio pregiudicano la disponibilità della singola risorsa, di un intero sito o di tutta la Grid di produzione. In caso di problemi, è essenziale agire tempestivamente per ripristinarne la funzionalità.

5.2.1 Configurazione del servizio top-level BDII della regione italiana

Il top-level BDII per l'infrastruttura Grid italiana è un servizio composto da cinque server, geograficamente distribuiti (tre al CNAF, uno alla sezione INFN di Ferrara e uno a Padova). Questi server hanno la stessa configurazione e sono parte di un cluster in round robin DNS. Il sistema di monitoring e allarmistica Nagios è responsabile del loro controllo ed è configurato per escludere automaticamente dal DNS i server che dovessero fallire i controlli abilitati, e per reinserirli nel caso il check indichi un ripristino del servizio. Un server viene escluso nel caso in cui non sia più raggiungibile (DOWN) o nel caso in cui il check del servizio BDII ritorni uno stato CRITICAL.

In figura 5.3 sono riportati i controlli fatti su server che fanno parte del cluster. I controlli sono stati implementati in modo da essere eseguiti ogni 5 minuti, ma per evitare falsi positivi, imputabili a problemi transitori (per esempio un problema temporaneo di rete), l'esclusione di un server non è immediata. Infatti, in caso di un problema alla prima verifica, lo stato viene impostato come CRITICAL SOFT1. A questo punto vengono eseguiti altri tre controlli ravvicinati (uno ogni minuto). Se confermano il problema, lo stato della risorsa passa a CRITICAL HARD e vengono eseguiti due interventi: la notifica via email del problema e la rimozione dell'indirizzo IP della risorsa dal DNS. In questo caso non viene effettuata nessuna notifica

via SMS, poiché il servizio top-level BDII, composto dai server rimanenti, risulta comunque nel suo complesso operativo.



Current Network Status
Last Updated: Tue Feb 24 19:16:01 CET 2009
Updated every 90 seconds
Nagios® 2.12 - www.nagios.org
Logged in as */C=IT/O=INFN/OU=Personal Certificate/L=CNAF/CN=Paolo Veronesi*

View Service Status Detail For All Host Groups
View Host Status Detail For This Host Group
View Status Overview For This Host Group
View Status Summary For This Host Group
View Status Grid For This Host Group

Host Status Totals

Up	Down	Unreachable	Pending
5	0	0	0

All Problems	All Types
0	5

Service Status Totals

Ok	Warning	Unknown	Critical	Pending
17	0	0	0	0

All Problems	All Types
0	17

Service Status Details For Host Group 'TOP-BDII'

Host	Service	Status	Last Check	Duration	Attempt	Status Information
prod-bdii-01.pd.infn.it	DNS Lookup	OK	2009-02-24 11:32:19	112d 6h 54m 8s	1/4	DNS OK: 0.008 seconds response time. prod-bdii-01.pd.infn.it returns 193.206.210.174
	TOP-BDII	OK	2009-02-24 19:14:30	3d 8h 19m 43s	1/4	LDAP OK - 0.057 seconds response time
top-bdii01	BDII	OK	2009-02-24 17:20:48	46d 7h 33m 48s	1/4	LDAP OK - 0.074 seconds response time
	DNS Lookup	OK	2009-02-24 11:26:49	112d 6h 53m 38s	1/4	DNS OK: 0.008 seconds response time. top-bdii01 returns 131.154.101.31
	NRPE	OK	2009-02-24 18:33:13	46d 8h 33m 22s	1/4	NRPE v2.5.1
	SSHD 22 Check	OK	2009-02-24 19:14:48	46d 7h 33m 48s	1/4	SSH OK - OpenSSH_3.9p1 NCSA_GSSAPI_20040818 KRB5 (protocol 1.99)
	TOP-BDII	OK	2009-02-24 19:14:30	3d 8h 55m 45s	1/4	LDAP OK - 0.076 seconds response time
top-bdii02	BDII	OK	2009-02-24 17:26:50	41d 4h 27m 41s	1/4	LDAP OK - 0.075 seconds response time
	DNS Lookup	OK	2009-02-24 11:32:51	112d 6h 47m 35s	1/4	DNS OK: 0.007 seconds response time. top-bdii02 returns 131.154.101.13
	NRPE	OK	2009-02-24 19:08:49	112d 7h 11m 38s	1/4	NRPE v2.5.1
	SSHD 22 Check	OK	2009-02-24 17:20:51	81d 6h 56m 16s	1/4	SSH OK - OpenSSH_3.9p1 NCSA_GSSAPI_20040818 KRB5 (protocol 1.99)
	TOP-BDII	OK	2009-02-24 19:14:30	3d 8h 55m 45s	1/4	LDAP OK - 0.110 seconds response time
top-bdii03	BDII	OK	2009-02-24 17:33:13	73d 20h 30m 41s	1/4	LDAP OK - 0.088 seconds response time
	DNS Lookup	OK	2009-02-24 11:08:50	112d 7h 11m 37s	1/4	DNS OK: 0.008 seconds response time. top-bdii03 returns 131.154.101.24
	NRPE	OK	2009-02-24 19:14:51	46d 7h 29m 51s	1/4	NRPE v2.5.1
	SSHD 22 Check	OK	2009-02-24 17:26:53	81d 6h 41m 20s	1/4	SSH OK - OpenSSH_3.9p1 NCSA_GSSAPI_20040818 KRB5 (protocol 1.99)
	TOP-BDII	OK	2009-02-24 19:14:30	3d 8h 55m 45s	1/4	LDAP OK - 0.074 seconds response time

Figura 5.3: Elenco dei controlli attivi per i server del cluster top-level BDII

Il DNS server è stato configurato per propagare le modifiche riguardo alla risorsa top-level BDII agli altri DNS server ogni minuto. Nel caso peggiore (uno dei cinque server riscontra un problema subito dopo un check), trascorrono circa 10 minuti prima che la sua esclusione sia propagata ovunque. L'intervallo di tempo è così ripartito: cinque minuti di attesa per il controllo che evidenzierà per la prima volta il problema, quattro minuti per passare dallo stato CRITICAL SOFT a HARD, e infine un minuto affinché la rimozione dal DNS sia propagata. Nel migliore dei casi, cioè se fallisce un attimo prima di essere controllato, occorreranno invece circa cinque minuti.

In questo scenario quindi, per un periodo di tempo che va da cinque a dieci minuti, una query su cinque al servizio top-level BDII andrebbe a contattare il server indisponibile e fallirebbe. Potenzialmente è possibile aumentare la frequenza dei controlli portando l'intervallo compreso tra uno e due minuti, ma vanno considerati anche eventuali effetti collaterali che questa configurazione potrebbe comportare. A causa di malfunzionamenti del middleware può capitare che il servizio BDII sia instabile in certe condizioni. Per esempio i check di Nagios potrebbero determinare uno stato CRITICAL ogni tre/quattro ore con un periodo di persistenza di qualche decina di minuti. Come precedentemente illustrato, in questi casi Nagios aggiorna il DNS rimuovendo il server che fallisce (e reinserendolo dopo qualche minuto). Se i check fossero più frequenti e l'aggiornamento del DNS immediato (invece che effettuato in seguito ad una conferma del problema), potrebbe verificarsi che in un dato intervallo di tempo tutti i server del cluster siano esclusi dal DNS. In questo caso la conseguenza sarebbe la totale indisponibilità del servizio, un evento estremamente più grave rispetto alla presenza di un sistema informativo transitoriamente instabile.

State	Type / Reason	Time	% Total Time	% Known Time
OK	Unscheduled	55d 15h 51m 39s	94.340%	94.340%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**55d 15h 51m 39s**	**94.340%**	**94.340%**
WARNING	Unscheduled	3d 8h 8m 21s	5.660%	5.660%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**3d 8h 8m 21s**	**5.660%**	**5.660%**
UNKNOWN	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
CRITICAL	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
Undetermined	Nagios Not Running	0d 0h 0m 0s	0.000%	
	Insufficient Data	0d 0h 0m 0s	0.000%	
	Total	0d 0h 0m 0s	0.000%	
All	Total	59d 0h 0m 0s	100.000%	100.000%

Figura 5.4: Availability del servizio top-level BDII nei primi due mesi del 2009.

La figura 5.4 mostra lo storico dell'availability del servizio top-level BDII italiano per i primi due mesi dell'anno 2009:

- Lo stato 'OK' indica il periodo di tempo in cui tutti i server del cluster sono operativi contemporaneamente.
- Lo stato 'WARNING' viene riportato quando uno o più server (ma non tutti) vengono rimossi dal DNS per problemi di funzionalità.
- Lo stato 'CRITICAL' segnala il periodo di tempo nel quale il cluster è vuoto o contiene al massimo una istanza.

Nel periodo considerato quindi, dal punto di vista degli utilizzatori, il servizio top-level BDII ha avuto una disponibilità effettiva del 100% data dalla somma delle percentuali corrispondenti agli stati 'OK' e 'WARNING'. Possiamo concludere che l'evidenza sperimentale dimostra l'efficacia delle politiche di controllo e ripristino sopra riportate.

5.3 Meccanismo di High Availability per VOMS

VOMS,discusso nella sezione 3.2.1, è un servizio composto da una interfaccia Web, un Web service di frontend e un database per le funzioni di backend (sono supportati sia MySQL che Oracle).

- Attraverso una interfaccia Web, un utente con regolare certificato personale X509, può fare richiesta di appartenenza ad una determinata Virtual Organization. La sua richiesta viene quindi inoltrata via email al relativo VO Manager che si occupa di approvarla o meno;
- Il Web service è interrogato dai servizi Grid per verificare che le credenziali presentate dall'utente siano sufficienti per permettere di acquisire l'autorizzazione necessaria a compiere l'operazione richiesta. Per esempio, se un utente cerca di scrivere su uno Storage Element in un'area riservata alla VO ATLAS, sarà il VOMS server di tale VO ad essere contattato per verificare che l'utente sia effettivamente membro dei quella VO.

Da questo esempio risulta evidente che il numero di operazioni sul servizio VOMS che coinvolgono l'interfaccia Web (come la richiesta di registrazione di un membro alla VO, l'ammissione o eliminazione di un utente), siano meno frequenti e meno

critiche di quelle che coinvolgono il Web service (contattato ad esempio per tutte le operazioni nelle quali va verificata l'appartenenza di un utente alla VO). Infatti, nel caso di indisponibilità dell'interfaccia Web, un utente non potrà fare richieste di appartenenza alla VO e il VO manager non potrà compiere le relative operazioni, ma tutto ciò non comporta disservizi per tutti gli altri utenti già registrati.

Nel caso invece di problemi al Web service o al database sottostante, falliranno le richieste di tutti gli utenti della VO, con la conseguenza che la procedura di autorizzazione sarà totalmente compromessa.

L'architettura di autenticazione e autorizzazione VOMS prevede un intrinseco meccanismo di failover. È possibile definire una lista di VOMS server per ogni VO, in modo tale che se il primo della lista risulta indisponibile, viene contattato il successivo, e così via.

5.3.1 Configurazione

Il servizio VOMS della Grid italiana supporta una ventina di VO nazionali e internazionali. Queste VO sono distribuite su due server distinti, ognuno dei quali ha una interfaccia Web e un database server MySQL, entrambi indipendenti. Per ogni VO definita, viene creato il rispettivo Web service contattabile su una determinata porta di comunicazione. Allo scopo di rendere il servizio indipendente da eventuali problemi di rete, i server primari sono stati affiancati da repliche geograficamente distribuite. Le repliche non hanno il servizio Web, il loro database quindi non viene modificato, ma solo sincronizzato con quello delle istanze primarie. Queste operazioni di sincronizzazione avvengono in automatico con una data frequenza configurabile.

Nel caso in cui i VOMS server primari diventino indisponibili, le richiesta da parte delle risorse Grid saranno processate da quelli secondari. Non sarà però possibile da parte degli utenti richiedere la registrazione alla VO e da parte dei VO manager di aggiungere o eliminare un utente (tali operazioni sono abilitate solo sulle istanze primarie).

Nagios è quindi configurato in modo da eseguire controlli diversificati sulle istanze primarie e secondarie. Oggetto del controllo sono: l'interfaccia Web dei VOMS server primari, e i Web service e il backend MySQL sia dei primari che dei secondari. Quando un problema viene rilevato su una di queste componenti, viene eseguito in

automatico un tentativo di riavvio della componente in oggetto. Se il tentativo non dovesse dare esito positivo, il problema viene notificato via email. La notifica via SMS avviene solo se sia il servizio VOMS primario che quello secondario risultano indisponibili per una determinata VO simultaneamente.

State	Type / Reason	Time	% Total Time	% Known Time
OK	Unscheduled	58d 13h 40m 0s	99.270%	99.270%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**58d 13h 40m 0s**	**99.270%**	**99.270%**
WARNING	Unscheduled	0d 10h 20m 0s	0.730%	0.730%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 10h 20m 0s**	**0.730%**	**0.730%**
UNKNOWN	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
CRITICAL	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
Undetermined	Nagios Not Running	0d 0h 0m 0s	0.000%	
	Insufficient Data	0d 0h 0m 0s	0.000%	
	Total	0d 0h 0m 0s	0.000%	
All	Total	59d 0h 0m 0s	100.000%	100.000%

Figura 5.5: Availability del servizio VOMS nei primi due mesi del 2009.

La figura 5.5 riporta l'andamento dell'availability del servizio VOMS nei primi due mesi dell'anno 2009:

- Lo stato 'OK' significa che tutti i server VOMS erano operativi;
- Lo stato 'WARNING' che uno dei VOMS server era indisponibile per almeno una VO;
- Lo stato 'CRITICAL' si manifesta quando sia il servizio VOMS primario che quello secondario non sono disponibili per almeno una VO.

Dal punto di vista degli utilizzatori quindi, il servizio VOMS è stato sempre disponibile nell'arco di tempo considerato.

5.4 Meccanismo di High Availability per FTS

FTS, introdotto nella sezione 3.6.3, è un servizio che permette di gestire task di trasferimento file tra siti geograficamente distribuiti. Questo servizio è presente al CERN e in altri sette centri di calcolo denominati Tier1 (tra cui il CNAF). Il servizio FTS nel suo insieme prevede una distribuzione dei compiti tra i server FTS della federazione così creata:

- Il server FTS del CERN è responsabile della gestione dei trasferimenti che implicano attività sia di importazione che di esportazione dati.
- I server FTS dei Tier1 sono responsabili della gestione dei trasferimenti che prevedono l'importazione dei dati dagli altri centri Tier1 e l'importazione/esportazione di dati verso i centri di calcolo loro associati (Tier2).

L'architettura del servizio prevede vari componenti: un Web service basato sull'applicazione Tomcat [98], il servizio BDII che pubblica le informazioni riguardo al servizio FTS stesso, un agente per ogni VO abilitata all'uso del servizio, e un agente per ogni canale di trasferimento configurato. Completa questa architettura il backend, basato su piattaforma Oracle.

5.4.1 Configurazione

Il frontend del servizio è composto da tre server sui quali sono distribuiti gli agenti delle VO e quelli dei canali. Su ogni server inoltre è configurato un BDII della risorsa e un Web service. I tre server compongono un cluster accessibile via round robin DNS, schematizzato in figura 5.6.

Nagios è configurato per tenere sotto controllo tutti e tre i server del cluster. I casi di indisponibilità che si possono verificare sono i seguenti:

- Indisponibilità di uno dei servizi BDII e/o Tomcat: essendo i tre server parte di un cluster in round robin DNS, una query su tre al sistema informativo di FTS o al Web service fallisce. In questo caso, come già spiegato, il sistema Nagios esclude dal DNS il server che fallisce il check e lo reinserisce successivamente quando il servizio torna disponibile. Essendo considerato un servizio essenziale, si è deciso di implementare una politica di allarmistica tale per cui la notifica

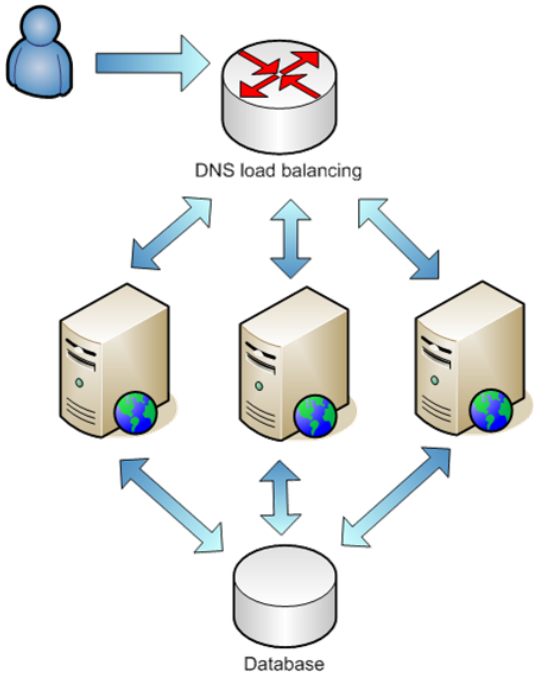


Figura 5.6: Schema della configurazione del servizio FTS

di un problema via SMS avviene già se due server vengono rimossi dal DNS, quindi quando il servizio nel suo complesso risulta essere ancora disponibile agli utenti.

- Indisponibilità di un agente: in questo caso tutte le operazioni che interessano un dato agente non sono più disponibili. Se si tratta di un agente di VO, allora quella VO non potrà più utilizzare il servizio FTS. Se si tratta invece di un agente di canale, non sarà possibile trasferire file sul canale relativo. Questo tipo di problematiche comporta una indisponibilità parziale del servizio FTS. Viene dunque fatto un tentativo di riavvio dell'agente e, in caso di esito negativo, viene lanciata la procedura di notificata via SMS.

Il backend è basato su Oracle ed è anch'esso configurato in modo da implementare una soluzione in alta affidabilità [99]. Fanno parte del cluster Oracle tre server, sui quali sono abilitati i controlli e le notifiche via SMS nel caso in cui l'intero cluster non sia disponibile.

A titolo di esempio, è riportata in figura 5.7 l'availability del servizio BDII del cluster composto dai frontend di FTS per i primi due mesi dell'anno 2009.

State	Type / Reason	Time	% Total Time	% Known Time
OK	Unscheduled	61d 10h 1m 35s	99.994%	99.994%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**61d 10h 1m 35s**	**99.994%**	**99.994%**
WARNING	Unscheduled	0d 0h 5m 0s	0.006%	0.006%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 5m 0s**	**0.006%**	**0.006%**
UNKNOWN	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
CRITICAL	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
Undetermined	Nagios Not Running	0d 0h 0m 0s	0.000%	
	Insufficient Data	0d 0h 0m 0s	0.000%	
	Total	0d 0h 0m 0s	0.000%	
All	Total	61d 10h 6m 35s	100.000%	100.000%

Figura 5.7: Availability del servizio BDII del cluster FTS nei primi due mesi del 2009.

- Lo stato 'OK' significa che il servizio BDII è operativo su tutti i server.
- Lo stato 'WARNING' indica che il servizio BDII è indisponibile su uno o più istanze server del cluster, fino ad un massimo di N-1, dove N indica il numero dei componenti di cluster.
- Lo stato 'CRITICAL' indica che il servizio BDII è indisponibile su un numero di istanze del cluster maggiore o uguale ad N-1.

Dal punto di vista degli utilizzatori quindi, il servizio FTS ha avuto una disponibilità effettiva del 100%, visto che la somma dei periodi in cui il servizio era in stato 'OK' o 'WARNING' coprono la totalità dell'intervallo di tempo preso in considerazione.

5.5 Meccanismo di High Availability per LFC

LFC, introdotto nella sezione 3.6.3, si occupa di gestire le corrispondenze tra LFN, GUID, SURL e TURL e, a seconda delle sue configurazioni, può svolgere il ruolo di catalogo globale di una data VO o locale di un dato sito. È un servizio abbastanza critico e problematico da configurare in alta affidabilità; l'architettura prevede infatti

che per ogni VO esista un solo catalogo globale, mentre in un determinato sito per una data VO, può esistere un solo catalogo locale. LFC è composto da uno o più frontend e da un database come backend (MySQL o Oracle).

5.5.1 Configurazione

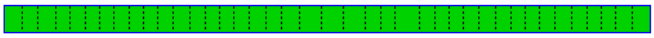
La grid italiana offre il servizio di catalogo LFC Globale ad una ventina di VO (nazionali e internazionali). Frontend e backend (basato su MySQL) del servizio insistono sullo stesso server. La scelta dell'utilizzo di MySQL come backend deriva dal fatto che quanto il servizio è stato installato, non era supportata la possibilità di utilizzare Oracle. Per i limiti di MySQL descritti in seguito, è stata pianificata la procedura per passare a Oracle, operazione che verrà concordata con tutte le VO coinvolte visto l'inevitabile indisponibilità che questa operazione comporterà nell'erogazione del servizio.

Il sistema di monitoring e allarmistica è stato progettato per controllare il servizio BDII della risorsa, i demoni del servizio LFC coinvolti e il database MySQL. Le politiche di ripristino e allarmistica sono state implementate in modo che in caso di fallimento viene provato il riavvio automatico del processo che fallisce. Se questo intervento non è sufficiente per risolvere il problema, viene inviata una notifica tramite SMS.

Purtroppo non ci sono alternative di backup per istanze di LFC che usano MySQL come database e il catalogo non è più utilizzabile fino a che non si interviene. La difficoltà maggiore nell'adottare una soluzione come quella praticata per il servizio VOMS, basata su istanza primaria e secondaria in sola lettura, riguarda la sincronizzazione del database MySQL che intercorre tra le diverse istanze LFC. A differenza del servizio VOMS, sul quale le operazioni di modifica del contenuto sono molto meno frequenti rispetto a quelle di lettura, le operazioni di aggiornamento su un catalogo sono comuni. Inoltre, il database in oggetto ha tipicamente dimensioni maggiori rispetto a quello di un server VOMS (che contiene sostanzialmente un elenco di utenti). Per esempio, il database dei server VOMS italiano ha una grandezza dell'ordine di tre MB per VO, mentre quello del catalogo LFC è di circa dieci GB. Ciò nonostante, configurazioni di servizi LFC in alta affidabilità adottano diverse soluzioni tecniche per il backend. Per esempio, la tecnologia Oracle supporta strumenti più sofisticati per la sincronizzazione, come la tecnologia Oracle Streams

[100], che rendono possibile la presenza di copie secondarie di un database. Questa tecnica è stata adottata per il catalogo della VO LHCb [101]: il catalogo globale primario della VO è ospitato al CERN, ed è l'unico sul quale sono permesse operazioni di scrittura, mentre una replica in sola lettura viene tenuta sincronizzata al CNAF.

La figura 5.8 riporta l'andamento dell'availability del server LFC della regione italiana nei primi due mesi dell'anno 2009.



State	Type / Reason	Time	% Total Time	% Known Time
UP	Unscheduled	59d 0h 0m 0s	100.000%	100.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**59d 0h 0m 0s**	**100.000%**	**100.000%**
DOWN	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
UNREACHABLE	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
Undetermined	Nagios Not Running	0d 0h 0m 0s	0.000%	
	Insufficient Data	0d 0h 0m 0s	0.000%	
	Total	0d 0h 0m 0s	0.000%	
All	Total	59d 0h 0m 0s	100.000%	100.000%

State Breakdowns For Host Services:

Service	% Time OK	% Time Warning	% Time Unknown	% Time Critical	% Time Undetermined
BACKUP-7937	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
BACKUP-7938	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
DNS Lookup	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
GRIS	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
LFCDAEMON	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
LFCDLI	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
NRPE	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
NTP	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
SSHD 22 Check	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
mysql	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
Average	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%

Figura 5.8: Availability del servizio LFC del ROC italiano nei primi due mesi del 2009.

- Lo stato 'OK' indica che tutti i servizi erano operativi.
- Lo stato 'WARNING' indica che per un servizio è stato registrato un carico superiore alla soglia predefinita.
- Lo stato 'CRITICAL' indica un servizio nel quale sono falliti i controlli.

Dal punto di vista degli utilizzatori quindi, il servizio LFC ha avuto una disponibilità effettiva del 100% nel periodo considerato.

5.6 Il servizio StoRM

StoRM [102] è un esempio di Grid Storage Resource Manager per sistemi di storage basati su disco. Implementa una interfaccia SRM, versione 2.2, ha una architettura modulare, supporta diversi protocolli per il trasferimento dati e diversi tipi di file system. StoRM, come anche il servizio FTS, è un'applicazione recente, nella quale gli aspetti di alta affidabilità e le problematiche di Grid sono state tenute in considerazione fin dagli sviluppi iniziali.

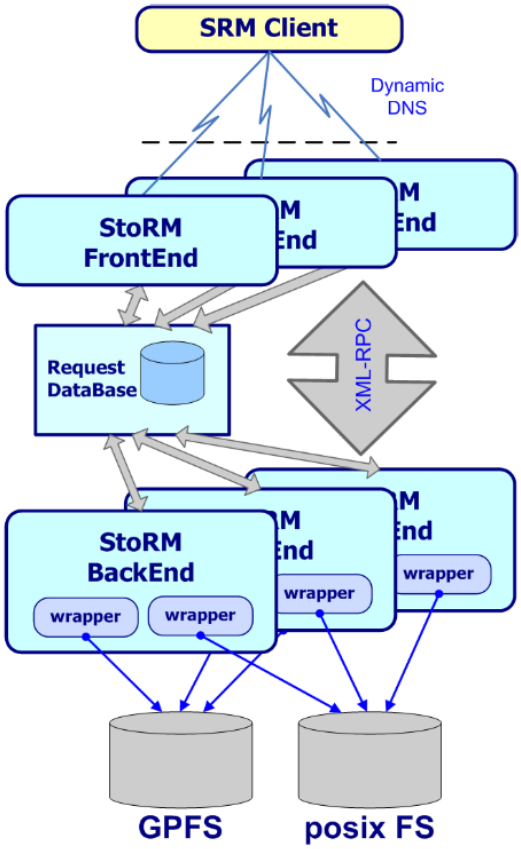


Figura 5.9: L'architettura modulare del sistema StoRM. Fonte [102].

Sebbene sia possibile implementare il servizio StoRM su di un singolo server, soluzione adottata in siti con quantità limitate di spazio disco, è raccomandata una configurazione modulare per siti con grandi risorse di stoccaggio dati. L'architettura prevede l'organizzazione delle diverse componenti come entità separate, in grado di comunicare attraverso protocolli di rete, e permette di realizzare configurazioni con più istanze dei servizi di frontend e backend in funzione delle esigenze provenienti dai

contesti applicativi. Come si può vedere in figura 5.9, il sistema può assumere una configurazione con più istanze dei servizi backend o frontend. Il database rimane l'unico componente condiviso, che può essere installato su un host dedicato.

La configurazione di una installazione di StoRM in alta affidabilità prevede le seguenti componenti:

- Più servizi frontend su host diversi incaricati della gestione delle richieste SRM. Attraverso una gestione dinamica del server DNS si possono implementare politiche di bilanciamento del carico fornendo agli utenti la visione di un unico servizio.

- Il database deve essere presente in un unica istanza, per le operazioni di replicazione si possono utilizzare le funzionalità avanzate fornire dai servizi DBMS.

- Possono esistere più istanze del servizio backend per garantire le caratteristiche di affidabilità e efficienza del servizio in funzione delle richieste degli utenti.

5.6.1 Configurazione

StoRM è la soluzione adottata al CNAF come sistema di Storage per dati su disco, utilizzando il file system distribuito GPFS [103]. La configurazione in alta affidabilità, illustrata in figura **??**, prevede diverse istanze di frontend, di server GridFTP per il trasferimento dei dati, e un backend che contiene anche il database.

Il sistema di monitoring e allarmistica è configurato nel seguente modo:

- I server configurati come frontend di StoRM sono in round robin a livello di DNS. Il sistema Nagios esclude dal DNS il server che fallisce il check e lo reinserisce successivamente quando il servizio torna disponibile.

- I server GridFTP sono anch'essi in round robin DNS. In questo caso i parametri tipici da controllare per prendere decisioni sono il carico dei server e lo stato del servizio GridFTP. Nel caso il numero totale dei server disponibili in un dato momento scenda al di sotto di un soglia configurabile, viene inviata una notifica via SMS.

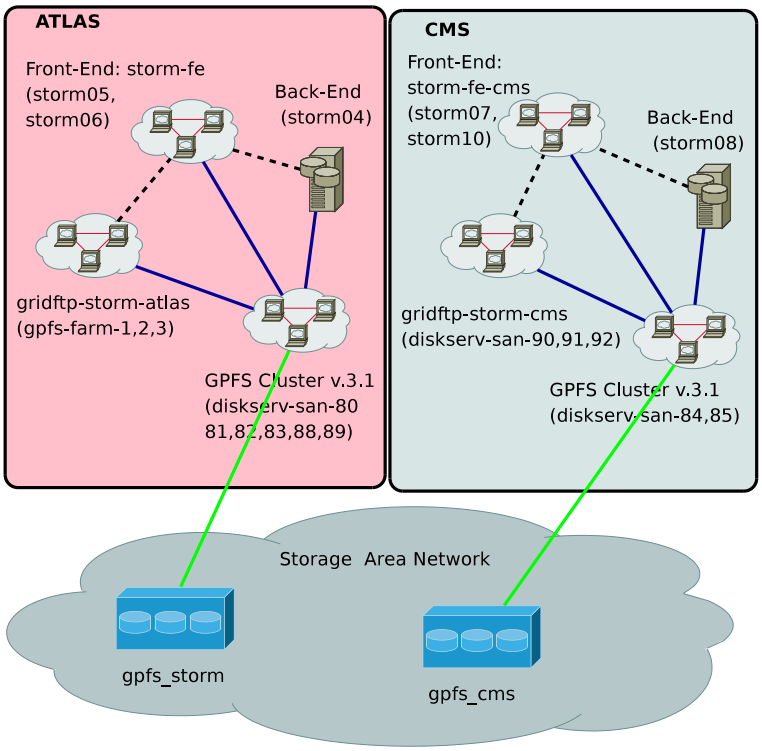
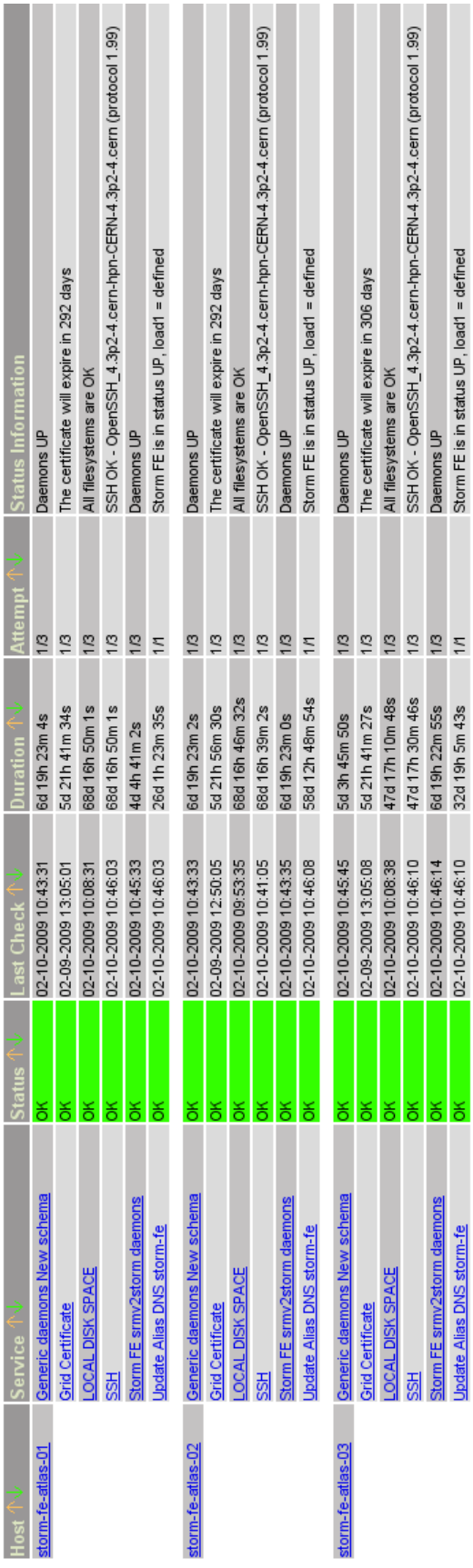


Figura 5.10: Configurazione del servizio StoRM al CNAF per le VO ATLAS, CMS.

- La scelta di avere una singola istanza di backend che ospita il database per più frontend deriva dal fatto che il backend non rappresenta un collo di bottiglia in termini di prestazioni, come evidenziato in [104], con le configurazioni adottate al CNAF. Sui server di backend sono attivi controlli sulle performance e sullo stato del servizio MySQL con notifiche via SMS in caso di problemi che ne compromettano la disponibilità.

In figura 5.11 sono mostrati i controlli che vengono effettuati sui server componenti i cluster di frontend di StoRM per la VO ATLAS. Poiché questi controlli sono passati dallo stato di test a quello di produzione recentemente, non è ancora possibile valutarne l'efficacia in termini di disponibilità globale del servizio: per una valutazione completa è infatti necessario correlare le disponibilità di tutti i componenti coinvolti: il cluster di frontend, il backend e il cluster di server GridFTP.



Host	Service	Status	Last Check	Duration	Attempt	Status Information
storm-fe-atlas-01	Generic daemons New schema	OK	02-10-2009 10:43:31	6d 19h 23m 4s	1/3	Daemons UP
	Grid Certificate	OK	02-09-2009 13:05:01	5d 21h 41m 34s	1/3	The certificate will expire in 292 days
	LOCAL DISK SPACE	OK	02-10-2009 10:08:31	68d 16h 50m 1s	1/3	All filesystems are OK
	SSH	OK	02-10-2009 10:46:03	68d 16h 50m 1s	1/3	SSH OK - OpenSSH_4.3p2-4.cern-hpn-CERN-4.3p2-4.cern (protocol 1.99)
	Storm FE srmv2storm daemons	OK	02-10-2009 10:45:33	4d 4h 41m 2s	1/3	Daemons UP
	Update Alias DNS storm-fe	OK	02-10-2009 10:46:03	26d 1h 23m 35s	1/1	Storm FE is in status UP, load1 = defined
storm-fe-atlas-02	Generic daemons New schema	OK	02-10-2009 10:43:33	6d 19h 23m 2s	1/3	Daemons UP
	Grid Certificate	OK	02-09-2009 12:50:05	5d 21h 56m 30s	1/3	The certificate will expire in 292 days
	LOCAL DISK SPACE	OK	02-10-2009 09:53:35	68d 16h 46m 32s	1/3	All filesystems are OK
	SSH	OK	02-10-2009 10:41:05	68d 16h 39m 2s	1/3	SSH OK - OpenSSH_4.3p2-4.cern-hpn-CERN-4.3p2-4.cern (protocol 1.99)
	Storm FE srmv2storm daemons	OK	02-10-2009 10:43:35	6d 19h 23m 0s	1/3	Daemons UP
	Update Alias DNS storm-fe	OK	02-10-2009 10:46:08	58d 12h 48m 54s	1/1	Storm FE is in status UP, load1 = defined
storm-fe-atlas-03	Generic daemons New schema	OK	02-10-2009 10:45:45	5d 3h 45m 50s	1/3	Daemons UP
	Grid Certificate	OK	02-09-2009 13:05:08	5d 21h 41m 27s	1/3	The certificate will expire in 306 days
	LOCAL DISK SPACE	OK	02-10-2009 10:08:38	47d 17h 10m 48s	1/3	All filesystems are OK
	SSH	OK	02-10-2009 10:46:10	47d 17h 30m 46s	1/3	SSH OK - OpenSSH_4.3p2-4.cern-hpn-CERN-4.3p2-4.cern (protocol 1.99)
	Storm FE srmv2storm daemons	OK	02-10-2009 10:46:14	6d 19h 22m 55s	1/3	Daemons UP
	Update Alias DNS storm-fe	OK	02-10-2009 10:46:10	32d 19h 5m 43s	1/1	Storm FE is in status UP, load1 = defined

Figura 5.11: Elenco dei controlli di Nagios sui frontend dell'istanza StoRM per la VO ATLAS.

5.7 Setup del servizio WMS del ROC italiano

Come illustrato nella sezione 3.4, il WMS è uno dei servizi Grid centrali più importanti. Questo servizio deve essere efficiente per minimizzare le latenze tra la sottomissione di un job e la sua effettiva esecuzione e prevedere dei meccanismi di recovery dei job per aumentare il rapporto tra successo e fallimento. Le attività interne del WMS, come il resource discovery, il brokering e il recovery dei job, devono essere il più trasparenti possibili agli utenti. La complessità di queste operazioni, la crescita delle risorse che fanno parte dell'infrastruttura Grid e il continuo aumento dell'utilizzo della Grid stessa, concorrono ad aumentare proporzionalmente le difficoltà di gestione di questo servizio. Inoltre, come precedentemente detto, non tutti i servizi offerti dal middleware gLite siano architetturalmente predisposti per permettere l'applicazione di tecniche di fault tolerance in modo tale che eventuali fallimenti siano gestiti in maniera trasparente agli utilizzatori, e il WMS è uno di questi.

Per controllarlo adeguatamente è stato sviluppato dal CNAF una applicazione denominata WMSMonitor [105].

Durante il ciclo di vita di un job, dalla sua sottomissione al recupero dell'output, sono diversi gli stati e le componenti coinvolte dal punto di vista del WMS e, conseguentemente, le possibili cause di fallimento. Il WMSMonitor è stato realizzato per rispondere alle esigenze di diverse categorie di utenti [106]:

- gli sviluppatori dei servizi Grid sono interessati ad applicazioni che controllano i loro servizi per migliorarne la qualità;
- gli utenti esperti che sottomettono grandi quantità di job, sono interessati per esempio a testare la scalabilità dei servizi della propria VO;
- i VO manager sono interessati ad ottenere statistiche aggregate sull'attività della propria VO;
- i gestori del servizio WMS vogliono rilevare tempestivamente situazioni problematiche e analizzare le cause dei fallimenti.

L'architettura di WMSMonitor, illustrata in figura 5.12, è caratterizzata dai seguenti componenti:

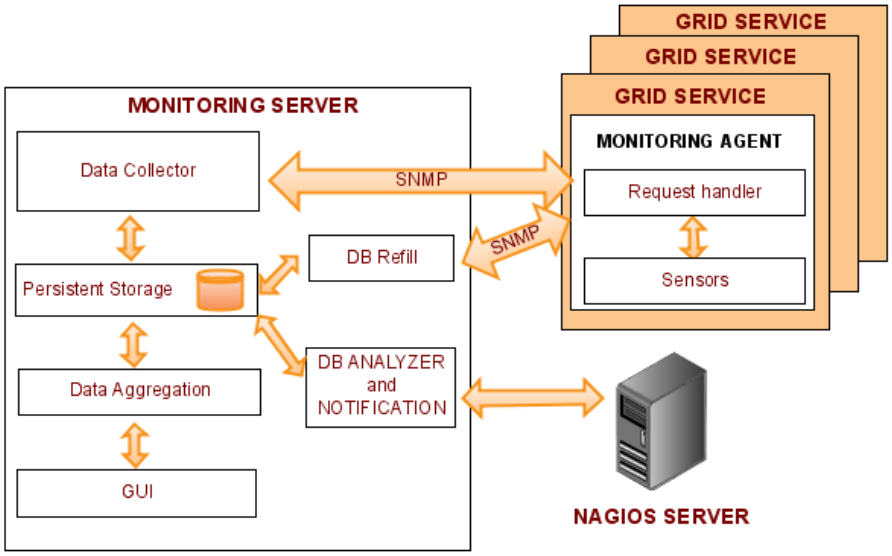


Figura 5.12: Schema dell'architettura del WMSMonitor. Fonte [105]

- il *Monitoring agent* è il componente Web service installato sui WMS e sui L&B, composto da un insieme di sensori che raccolgono diverse metriche definite a seconda del servizio, e le pubblica utilizzando il protocollo SNMP[2];
- *una base dati persistente* nella quale vengono inserite le metriche raccolte interrogando ciascun Monitoring agent;
- Un *aggregatore di dati* che si occupa di estrarre le informazioni dalla base dati e di elaborarle mettendole a disposizione dell'interfaccia grafica;
- L'*interfaccia grafica* dove, attraverso diverse view predefinite, è possibile visualizzare grafici e metriche delle componenti coinvolte.

Recentemente a questa architettura è stato aggiunto anche Nagios, che concorre da un lato a fornire al tool WMSMonitor risultati di un insieme di check eseguiti, in modo da incrementare le metriche utili alla valutazione dello stato generale del servizio WMS; dall'altro riceve, via NSCA, lo stato complessivo del servizio WMS elaborato dal WMSMonitor, e ne notifica eventualmente l'indisponibilità.

Al CNAF sono disponibili diverse istanze del servizio WMS e L&B: alcune di queste sono dedicate a singole VO, altre sono istanze di test e sviluppo. Sebbene

[2]Simple Network Management Protocol [107]

Service	% Time OK	% Time Warning	% Time Unknown	% Time Critical	% Time Undetermined
ALICE-WMS	91.480% (91.480%)	8.421% (8.421%)	0.064% (0.064%)	0.035% (0.035%)	0.000%
ATLAS-WMS	98.202% (98.202%)	1.776% (1.776%)	0.022% (0.022%)	0.000% (0.000%)	0.000%
CDF-WMS	75.085% (75.085%)	0.000% (0.000%)	24.907% (24.907%)	0.007% (0.007%)	0.000%
CMS-ANALISI-WMS	87.327% (87.327%)	12.620% (12.620%)	0.017% (0.017%)	0.036% (0.036%)	0.000%
CMS-PROD-WMS	97.190% (97.190%)	2.633% (2.633%)	0.164% (0.164%)	0.014% (0.014%)	0.000%
DEVEL-WMS	41.976% (41.976%)	14.538% (14.538%)	1.883% (1.883%)	41.603% (41.603%)	0.000%
LHCB-WMS	99.887% (99.887%)	0.000% (0.000%)	0.007% (0.007%)	0.105% (0.105%)	0.000%
MULTI-WMS	88.182% (88.182%)	11.818% (11.818%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
NRPE	100.000% (100.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000% (0.000%)	0.000%
SAM-WMS	72.582% (72.582%)	0.000% (0.000%)	27.372% (27.372%)	0.046% (0.046%)	0.000%
Average	85.191% (85.191%)	5.180% (5.180%)	5.444% (5.444%)	4.185% (4.185%)	0.000%

Figura 5.13: Availability del servizio WMS del CNAF aggregato per cluster per i primi due mesi del 2009

indipendenti l'una dalle altre, le istanze del WMS sono state raggruppate in cluster logici in modo tale da poter definire la disponibilità del servizio WMS come la disponibilità data da quello del cluster. In figura 5.13 viene mostrata l'availability del servizio WMS disponibile al CNAF aggregato per cluster.

- I cluster ALICE-WMS, ATLAS-WMS, CDF-WMS e LHCB-WMS raggruppano le istanze di WMS e L&B dedicate rispettivamente alle VO ALICE, ATLAS, CDF e LHCB.
- Il cluster CMS-ANALISI-WMS raggruppa le istanze di WMS e L&B utilizzate dalla VO CMS per le attività di analisi dei dati, mentre il cluster CMS-PROD-WMS raggruppa quelle utilizzate per la produzione Montecarlo dei dati della VO CMS.
- Il cluster MULTI-WMS raggruppa le istanze di WMS e L&B general purpose che supportano più VO.
- Il cluster SAM-WMS raggruppa le istanze di WMS e L&B utilizzate per la sottomissione dei SAM test a livello europeo;
- Il cluster DEVEL-WMS raggruppa infine le istanze di test e sviluppo di WMS e L&B, non utilizzate in produzione.

Il fallimento di una singola istanza, pregiudica l'esecuzione di parte o tutti i job che sono stati sottomessi a quella istanza; questo fatto non è assolutamente trasparente all'utente, che però ha la possibilità di risottomettere i job ad una istanza diversa funzionante: la notifica SMS viene infatti inviata quando tutte le istanze di un singolo cluster sono contemporaneamente indisponibili.

- Lo stato 'OK' indica che tutte le istanze del cluster sono operative.
- Lo stato 'WARNING' indica che almeno una istanza del servizio WMS del dato cluster era indisponibile.
- Lo stato 'UNKNOWN' indica il periodo di tempo in cui Nagios non ha ricevuto i dati via NSCA dal WMSMonitor nei tempi previsti.
- Lo stato 'CRITICAL' indica che tutte le istanze del servizio WMS di un dato cluster erano contemporaneamente indisponibili.

5.8 Disponibilità dell'infrastruttura Grid Europea

La misura continua e sistematica della Availability e Reliability su base mensile dei siti e dei servizi Grid centrali è uno strumento utile per verificare il livello di affidabilità, la robustezza e il livello di maturità dell'infrastruttura nel suo complesso. Dall'anno 2008, nell'ambito del progetto EGEE è partita l'iniziativa di migliorare il livello di qualità fornito dalle Grid nazionali e regionali, e dai relativi siti. Gli algoritmi utilizzati per le stime sono quelli illustrati nella sezione 4.5. A livello italiano sono state individuate due linee di azione con l'obiettivo di migliorare il livello di disponibilità delle singole istanze dei servizi, sia locali che centrali, e in generale dei siti.

In primo luogo sono state dettagliate con maggior precisione le procedure per descrivere le operazioni da seguire in caso di aggiornamenti del middleware gLite. In particolare sono stati sviluppati e divulgati esempi e raccomandazioni sulle procedure da seguire nella gestione dei servizi e delle risorse. Sono stati inoltre configurati diversi ambiente di test per sperimentare le procedure di installazione e configurazione dei servizi Grid in caso di aggiornamento del software. Lo scopo è quello di svolgere attività di certificazione prima che il middleware entri in ambiente di produzione. L'adozione di procedure comuni e strumenti omogenei permette inoltre di offrire un supporto migliore e più efficiente nel caso si presentino problemi di malfunzionamento dei servizi. In secondo luogo sono state studiate e applicate tecniche sempre più raffinate di controllo e recovery automatico. Tramite una migliore definizione dei parametri da valutare, una accurata analisi delle problematiche più comuni e una più approfondita conoscenza delle architetture dei servizi, è stato

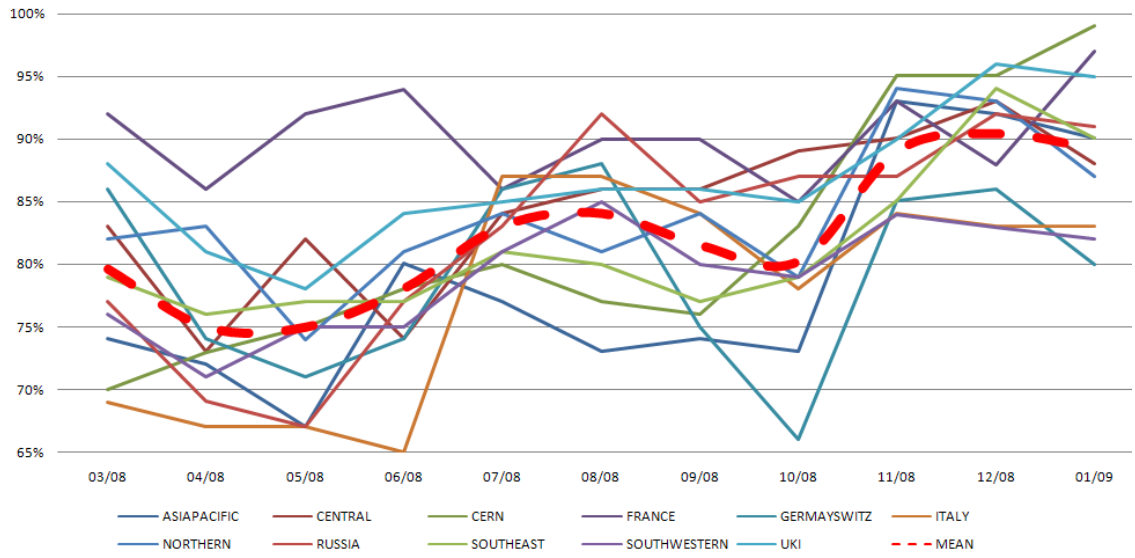


Figura 5.14: Availability della Grid di produzione europea.

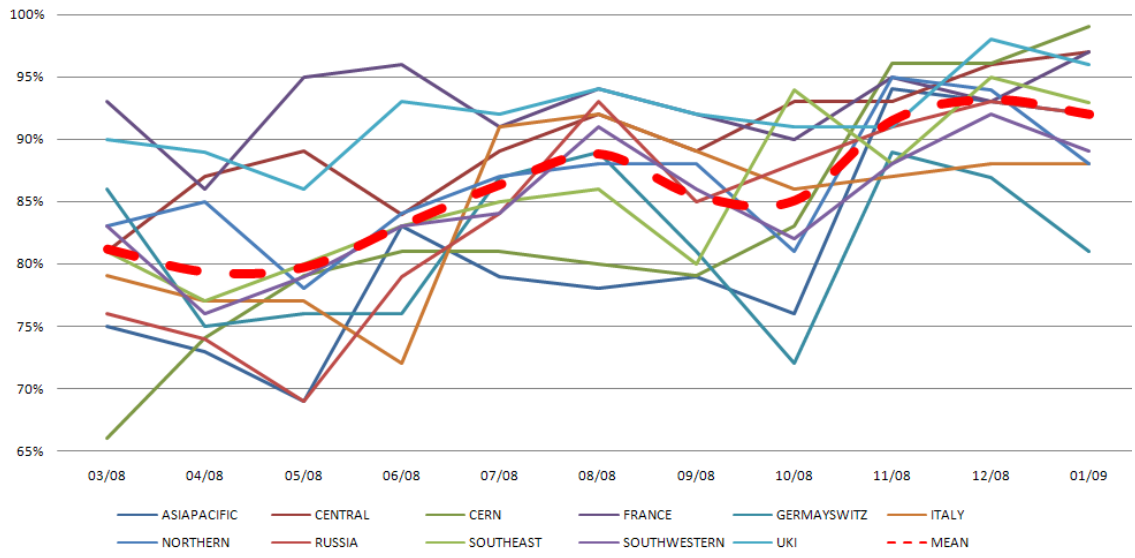


Figura 5.15: Reliability della Grid di produzione europea.

possibile migliorare l'infrastruttura di monitoring di basso livello. I risultati sono stati la riduzione del numero di casi di indisponibilità dei servizi offerti a livello più alto, e il veloce ripristino delle funzionalità in caso di fallimento.

Le figure 5.14 e 5.15 mostrano l'andamento dell'Availability e della Reliability dei siti dell'infrastruttura Grid europea nei mesi che vanno da Marzo 2008 a Gennaio 2009. Le statistiche sono aggregate per Grid nazionale/regionale e mostrano che mediamente il SLA di EGEE, che prevede una Availability maggiore del 70% o una Reliability maggiore del 75%, è stato rispettato. In particolare si sottolinea una evoluzione positiva degli andamenti delle due misure con una diminuzione delle differenze qualitative tra le varie regioni. Storicamente queste differenze hanno toccato delle punte massime del 30%, mentre negli ultimi mesi si sono consolidate al di sotto del 20%. Nell'analisi bisogna tenere in considerazione che l'aggregazione proposta si basa sulla media aritmetica delle prestazioni dei siti che fanno parte di ciascuna regione. In altre parole al momento non vengono calcolate medie pesate in funzione della quantità di risorse disponibili in un sito.

Questo scenario penalizza le regioni caratterizzate da un alto numero di siti, come quello italiana. In [108] è stato calcolato che normalizzando i dati di Availability con il numero di cpu offerte, Availability e Reliability sono mediamente superiori del 10% rispetto ai valori attualmente riscontrati. Questo miglioramento è probabilmente dovuto al fatto che grandi centri di calcolo hanno tipicamente una infrastruttura di monitoraggio più sofisticata e maggiori risorse umane dedicate alla gestione del sito. L'approccio più corretto per calcolare medie pesate è quello di prendere in considerazione sia l'ammontare delle risorse di calcolo che la capacità di stoccaggio dei dati dei siti.

Nelle figure 5.16 e 5.17, sono evidenziati i progressi fatti dalla regione italiana. I siti sono aggregati in due gruppi: quelli che hanno il ruolo di Tier1 e Tier2 nell'ambito del progetto WLCG (per quantità di risorse di calcolo e stoccaggio dati rappresentano circa il 65% dell'infrastruttura), e gli altri centri denominati Tier3. Il miglioramento generale dei siti, indipendentemente dalla loro dimensione, trova dirette corrispondenze temporali, in particolare dal mese di Giugno in poi, con lo sviluppo dei piani di lavoro precedentemente descritti e la promozione delle procedure e degli strumenti elaborati.

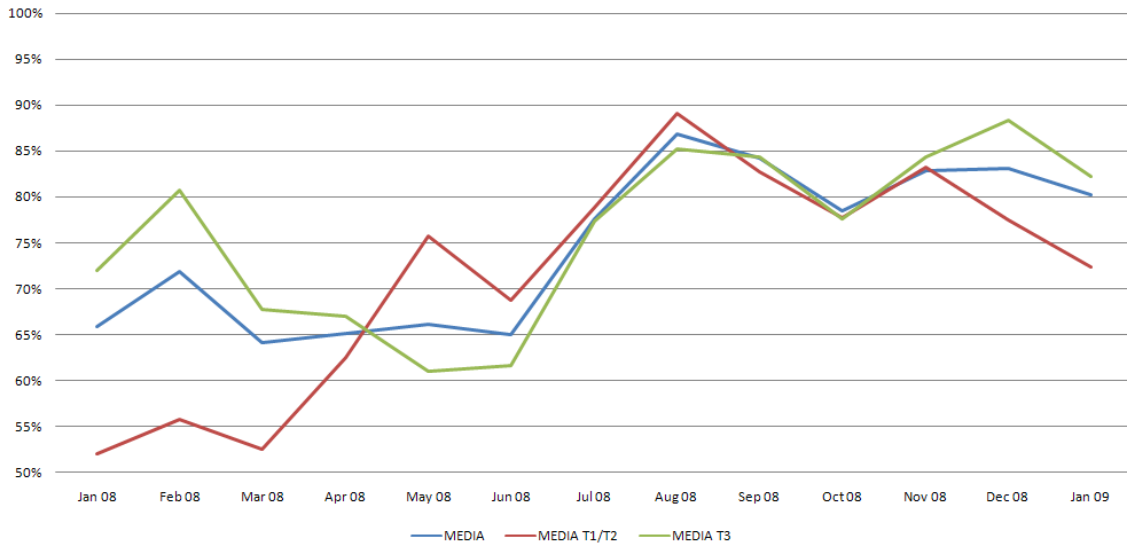


Figura 5.16: Availability della Grid di produzione italiana.

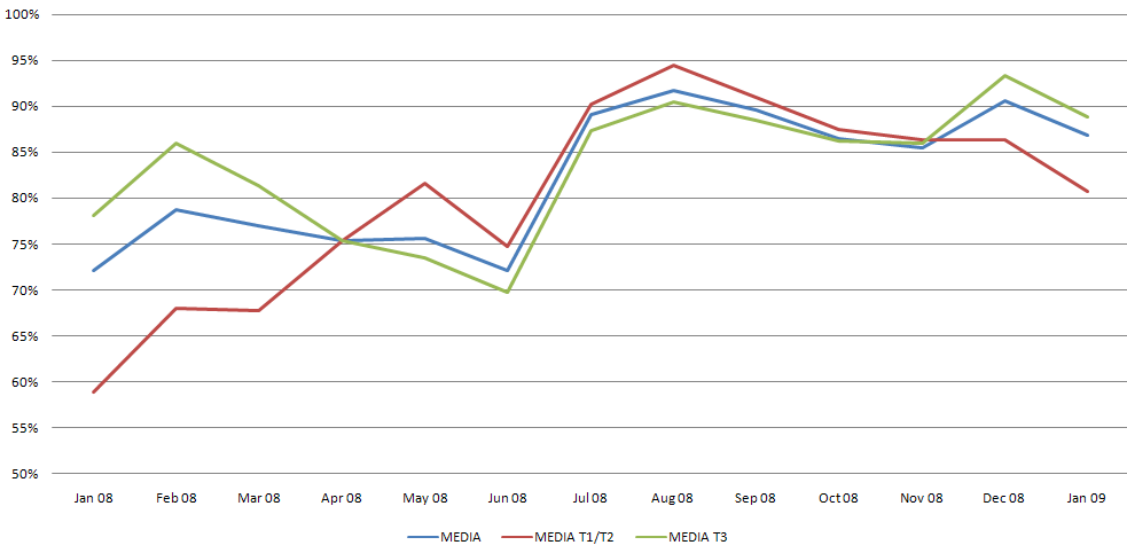


Figura 5.17: Reliability della Grid di produzione italiana.

Conclusioni

Il paradigma di Grid nasce per offrire la condivisione di risorse eterogenee e geograficamente distribuite in modo coordinato e trasparente fra comunità di utenti con differenti finalità e scenari d'uso. L'evoluzione delle infrastrutture di Grid, dal concetto iniziale di Computing e Data Grid verso architetture orientate ai sevizi, ha portato allo sviluppo di applicazioni avanzate in grado di sfruttare al meglio queste capacità di coordinazione e condivisione. L'esperimento *Large Hadron Collinder (LHC)* del CERN è un esempio di applicazione che richiede una infrastruttura di Grid per la condivisione di grandi quantità di dati fra un numero elevato di centri internazionali. L'aumentare della complessità delle applicazioni e la crescente necessità di una infrastruttura sempre più affidabile hanno però evidenziato come una delle principali limitazioni delle infrastrutture di Grid esistenti sia la gestione e la disponibilità delle risorse condivise.

In questa tesi è stato presentato il lavoro di studio dell'architettura dei principali servizi offerti dal middleware gLite e le relative problematiche di gestione che sono subentrate al fine di aumentarne la robustezza. Sono state inoltre illustrate le problematiche della negoziazione in Grid in modalità standard di Service Level Agreement e la relativa architettura di servizi. La tesi propone varie politiche e metodologie di High Availability per servizi Grid centrali e di sito, che sono state sperimentate e messe in produzione con successo nell'ambito dell'infrastruttura di Grid italiana. In particolare, sono stati sviluppati e adottati meccanismi di High Availability con lo scopo di migliorare la disponibilità dei singoli servizi, e di conseguenza le potenzialità dell'infrastruttura nel suo complesso. Questi meccanismi sono stati integrati nel sistema di controllo e allarmistica Nagios con lo scopo di automatizzarli, avere notifiche puntuali dei problemi, correlare gli eventi dei diversi componenti di un servizio al fine di comprenderne meglio le criticità e di conseguenza studiare soluzioni per risolverle. Le soluzioni adottate e la continua attività di

controllo dell'infrastruttura hanno contribuito ad un sensibile miglioramento della disponibilità dei siti e dei servizi nella regione italiana, permettendo di soddisfare i vincoli di disponibilità definiti nel Service Level Agreement di EGEE.

Le applicazioni di controllo utilizzate e le tecniche di High Availability e Recovery automatico definite rappresentano le basi sulle quali investire per raggiungere l'obiettivo della sostenibilità a lungo termine. Lo scopo è quello di agevolare la transizione ad una infrastruttura sostenibile, basata su un nuovo scenario europeo costituito da un insieme di Grid nazionali (*National Grid Initiatives*) pienamente integrate e interoperabili su scala internazionale. La disponibilità generale dell'infrastruttura EGEE ha visto nell'ultimo anno un apprezzabile miglioramento, caratterizzato da piccoli, ma importanti progressi nella qualità della gestione delle risorse, anche se non costanti nel tempo. Per consolidare e migliorare i risultati ottenuti è necessario dunque concentrare l'attenzione sui diversi aspetti che concorrono alla generale affidabilità dell'infrastruttura. In primo luogo è importante disporre di strumenti di supporto per l'analisi nel dettaglio delle problematiche derivanti dalla indisponibilità dei servizi in seguito ad aggiornamenti del middleware, in particolare tramite la definizione di ambienti di testing adeguati dove certificare sia gli aggiornamenti software dei diversi componenti, che le procedure per applicare questi aggiornamenti, considerando le eventuali contromisure da prendere per minimizzarne l'indisponibilità e assicurarne la continuità operativa.

In secondo luogo serve promuovere l'utilizzo di strumenti di controllo di basso livello, condividendo le esperienze delle soluzioni adottate nei vari centri di calcolo, in modo da rilevare tempestivamente le anomalie e avere un insieme comune di procedure e controlli standard.

Infine è necessario migliorare l'accuratezza con la quale si calcolano parametri quali l'availability e reliability, per esempio normalizzando le metriche riguardanti la disponibilità di tutte le tipologie di risorse di un sito, come la potenza di calcolo effettivamente disponibile e le risorse di stoccaggio dati.

In questo modo si possono ottenere informazioni più precise sul reale andamento delle disponibilità delle risorse offerte dall'infrastruttura nel suo complesso. È inoltre importante disporre di strumenti che permettano di stabilire SLA in modo più semplice e automatico tra i vari attori coinvolti in Grid (resource provider, VO, il centro operativo della Grid, etc), e che ne permettano il loro monitoraggio. Il

modello di gestione e controllo fin qui utilizzato si è consolidato ed è maturato nel tempo riuscendo ad adattarsi bene al continuo aumento delle risorse e dei servizi dell'infrastruttura.

Questo modello però è prevalentemente centralizzato, sia per quello che riguarda gli strumenti utilizzati, che per la responsabilità delle attività quotidiane di gestione. È importante adattare meccanismi di alta affidabilità al fine di ridurre gli sforzi necessari alla gestione dell'infrastruttura e agevolarne la transizione verso il modello organizzativo regionale basato sulle NGI. Occorre dunque ridistribuire le responsabilità delle attività di controllo, privilegiando le soluzioni di monitoring a livello di sito e automatizzandole il più possibile. In questo ambito è in fase di sperimentazione una soluzione basata su Nagios contenente diversi plugin sviluppati ad hoc per il controllo dei servizi Grid. Questo nuovo strumento può essere integrato con installazioni eventualmente preesistenti nei siti e facilmente esteso con plugin aggiuntivi. Questo strumento, opportunamente integrato con le tecniche di alta affidabilità descritte in questa tesi ha l'ambizione di migliorare ulteriormente la qualità complessiva dell'infrastruttura di Grid europea.

Glossario

A

ABI Application Binary Interface;
AFS Andrew File System;
ALICE A Large Ion Collider Experiment;
AMGA ARDA Metadata Grid Application;
AMS Advanced Multi-threaded Server;
APEL Accounting Processor for Event Logs;
API Application Program Interface;
ARC Advance Resource Connector;
ATLAS A Toroidal LHC ApparatuS;
AUP Acceptable Use Policy;

B

BaBar B and B-bar experiment;
BDII Berkeley Database Information Index;
BOGUS Babar Object-oriented Geant-4-based Unified Simulation;

C

CA Certification Authority;
CDF Collider Detector at Fermilab;
CE Computing Element;
CERN European Laboratory for Particle Physics;
CIC Core Infrastructure Center;
CM Computer Model;
CMS Compact Muon Solenoid;
CNAF INFNs National Center for Telematics and Informatics;
COD CIC-On-Duty;
CP Charge Parity;
CPU Central Process Unit;
CVS Concurrent Version System;

D

DANTE Delivery of Advanced Network Technology to Europe;
DEISA Distributed European Infrastructure for Supercomputing Applications;
DGAS Distributed Accounting System;
DNS Domain Name System;
DoW Description of Work;

E

EC European Commission;
EDG European DataGrid;
EGEE Enabling Grids for E-Science in Europe;
EGI European Grid Initiative;
EGI_DS European Grid Initiative Design Study;
ENOC EGEE Network Operations Center;
ERA European Research Area;
ERI European Research Infrastructure;
ESM Experiment Software Manager;
EU European Union;

F

FCR Freedom of Choice for Resources;
FTP File Transfer Protocol;
FTS File Transfer Service;

G

GGUS Global Grid User Support;
GIIS Grid Index Information Server;
GLUE Grid Laboratory for a Uniform Environment;
GOC Grid Operation Center;
GOCDB GOC Database;
GPFS General Parallel File System;
GRAM Globus Resource Allocation Manager;
GridICE a distributed monitoring tool designed for Grid Systems;
GRIS Grid Resource Information Service;
GSI Grid Security Infrastructure;
GT Guarantee Term;
GUI Graphical User Interface;
GUID Grid Unique ID;

H

HEP High Energy Physics;
HTTP Hyper Text Transfer Protocol;

I

ICMP Internet Control Message Protocol;
ID Identifier;
IEEE Institute of Electrical and Electronics Engineers;
IETF Internet Engineering Task Force;
IN2P3 Institut Nacional de Physique Nuclèaire et de Physique des Particules;
INFN Istituto Nazionale di Fisica Nucleare;
IP Internet Protocol;
IS Information Service;
ISO International Standard Organization;
ISP Internet Service Provider;

J

JA Job Adapter;
JC Job Controller;
JCS Job Control Service;
JDL Job Description Language;

K

kSI2K kilo SPCEint2000 unit;

L

LAN Local Area Network;
LB Logging and Bookkeeping Service;
LCAS Local Centre Authorization Service;
LCG LHC Computing Grid;
LCMAPS Local Credential Mapping Service;
LDAP Lightweight Directory Access Protocol;
LFN Logical File Name;
LHC Large Hadron Collider;
LHCb Large Hadron Collider beauty experiment;
LM Log Monitor;
LSF Load Sharing Facility;

M

MAC Media Access Control;
MB Match-Maker Broker;
MDS Monitoring and Discovery Service;
MOOSE Monolithic Object Oriented Simulation Executable;
MoU Memorandum of understanding;

N

NFS Network File System;
NGI National Grid Initiative;
NIKHEF National Institute for Nuclear Physics and High Energy Physics in the Netherlands;
NRPE Nagios Remote Plugin Executor;
NS Network Server;
NTP Network Time Protocol;

O

OGF Open Grid Forum;
OGSA Open Gris Service Architecture;
OGSI Open Grid Service Infrastructure;
OSG Open Science Grid;

P

PBS Portable Batch System;
PERL Pratical Extraction and Report Language;
PEP-II Positron Electron Project II;
PFN Physical File name;
PID Process IDentifier;
PKI Public Key Infrastructure;

R

RA Registration Authority;
RAL Rutherford Appleton Laboratory;
RAM Random Access Memory;
RB Resource Broker;
RC Resource Center;
ROC Regional Operation Center;
ROOT An Object-Oriented Data Analysis Framework;
RPC Remote Procedure Call;
RPC Resistive Plate Chamber;
RPM RedHat Package Manager;

S

SAM Service Availability Monitoring;
SDK Software Development Kit;
SDT Service Description Term;
SE Storage Element;
SFT Site Functional Test;
SKI Single Kernel Image;
SLA Service Level Agreement;
SLAC Stanford Linear Accelerator Center;
SLD Service Level Description;
SLO Service Level Objectives;
SNMP Simple Network Management Protocol;
SP Simulation Production;
SRM Storage Resource Manager;
SSH Secure SHell;

T

TCP Transmission Control Protocol;

U

UDP User Datagram Protocol;
UI User Interface;
UMD Unified Middleware Distribution;
UNICORE Uniform Interface to Computing Resources;
URL Universal Resource Locator;
UUID Universally Unique Identifier;

V

VDT Virtual Data Toolkit;
VO Virtual Organization;
VOMS Virtual Organization Management Team;

W

WAN Wide Area Network;
WLCG Worldwide LHC Computing Grid;
WM Workload Manager;
WMS Workload Management System;
WN Worker Node;

X

XROOTD eXtended Root Daemon;

Elenco delle tabelle

Elenco delle figure

Bibliografia

[1] EGEE. Egee project. Website, 2008. http://www.eu-egee.org/.

[2] DEISA Project. Distributed european infrastructure for supercomputing applications. Website, 2008. http://www.deisa.eu/.

[3] gLite. glite. Website, 2009. http://glite.web.cern.ch/glite/.

[4] INFN. Infn grid project. Website, 2008. http://grid.infn.it/.

[5] European Grid Initiative. European grid initiative. Website, 2008. http://www.eu.egi.eu/.

[6] CERN. Centro europeo per la ricerca nucleare - cern. Website, 2008. http://public.web.cern.ch/public/.

[7] Wikipedia the free encyclopedia. World wide web definition. Website, 2008. http://en.wikipedia.org/wiki/World_Wide_Web.

[8] Business experiments in grid. Website, 2006. http://www.beingrid.eu/.

[9] EGEE Project. Egee business associates. Website, 2008. http://business.eu-egee.org/index.php?id=120.

[10] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 2004.

[11] I. Foster, C. Kesselman, and S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, 15(3):200, 2001.

[12] SUN. Sun grid engine. Website, 2007. http://gridengine.sunsource.net/.

[13] Platform. Load sharing facility - lsf. Website, 2007. http://www.platform.com/Products/platform-lsf.

[14] Altair Engineering. Portable batch system - pbs. Website, 2009. http://www.pbsgridworks.com/.

[15] Condor Project. Condor project. Website, 2009. http://www.cs.wisc.edu/condor.

[16] Wikipedia the free encyclopedia. Entropia, inc. Website, 2000. http://en.wikipedia.org/wiki/Entropia,_Inc._(company).

[17] Wikipedia the free encyclopedia. United devices. Website, 2000. http://en.wikipedia.org/wiki/United_Devices.

[18] Wikipedia the free encyclopedia. Gnutella. Website, 2008. http://en.wikipedia.org/wiki/Gnutella.

[19] Inc BitTorrent. Bittorrent, inc. Website, 2009. http://www.bittorrent.com/.

[20] OGF. Open grid forum. Website, 2008. http://www.ogf.org.

[21] OGF-Europe. Ogf-europe. Website, 2008. http://www.ogfeurope.eu.

[22] I. Foster, C. Kesselman, and S. Tuecke. The Open Grid Services Architecture. *The Grid2: Blueprint for a New Computing Infrastructure*, pages 215–257, 2004.

[23] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, and P. Vanderbilt. Open Grid Services Infrastructure (OGSI). In *Global Grid Forum*, 2003.

[24] K. Czajkowski, D. Ferguson, I. Foster, J. Frey, S. Graham, T. Maguire, D. Snelling, and S. Tuecke. From Open Grid Services Infrastructure to WS-Resource Framework: Refactoring & Evolution. In *Global Grid Forum Draft Recommendation, May*, 2004.

[25] K. Czajkowski, D.F. Ferguson, I. Foster, J. Frey, S. Graham, I. Sedukhin, D. Snelling, S. Tuecke, and W. Vambenepe. The WS-Resource Framework

Version 1.0. In *Global Grid Forum, available at http://www. globus. org/wsrf*, 2004.

[26] M. Humphrey, G. Wasson, J. Gawor, J. Bester, S. Lang, I. Foster, S. Pickles, M. Mc Keown, K. Jackson, J. Boverhof, et al. State and events for web services: a comparison of five WS-resource framework and WS-notification implementations. In *High Performance Distributed Computing, 2005. HPDC-14. Proceedings. 14th IEEE International Symposium on*, pages 3–13, 2005.

[27] WLCG Project. Worldwide lhc computing grid (wlcg). Website, 2008. `http://lcg.web.cern.ch/LCG/`.

[28] CERN. Large hadron collider (lhc). Website, 2008. `http://public.web.cern.ch/Public/en/LHC/LHC-en.html`.

[29] EGEE Project. National grid initiatives - ngi. Website, 2009. `http://collaborating.eu-egee.org/index.php?id=534`.

[30] GEANT Project. Geant project. Website, 2009. `http://www.geant.net/`.

[31] EGEE Project. Msa1.1: Operations metrics defined. Website, 2008. `https://edms.cern.ch/document/723928`.

[32] EGEE Project. Cic operation portal. Website, 2008. `https://cic.gridops.org/`.

[33] EGEE Project. Egee accounting portal. Website, 2008. `http://www3.egee.cesga.es/gridsite/accounting/CESGA/egee_view.php`.

[34] EGEE Project. Assessment of the production grid infrastructure service status. Website, 2008. `https://edms.cern.ch/document/726263`.

[35] EGEE Project. gstat monitor. Website, 2008. `http://goc.grid.sinica.edu.tw/gstat/`.

[36] WLCG Management Board. Requirements to report WLCG installed capacity. Technical report, 2009. `https://twiki.cern.ch/twiki/pub/LCG/WLCGCommonComputingReadinessChallenges/installedcapactiyreportingrequirements-v3-1-2.pdf`.

[37] S. Burke, G. Cowan, F. Donno, L. Field, J. Jensen, M. Jouvin, M. M. Coelho Dos Santos, L. Magnoni, P. Millar, J. Shih, J. Templon, S. Traylen, R. Trompert, C. Schiaua, J. V. Eldik, R. Zappi, B. Holzman, and B. Bockelman. Usage of Glue Schema v1.3 for WLCG Installed Resource Capacity information. Technical report, 2009. https://twiki.cern.ch/twiki/pub/LCG/WLCGCommonComputingReadinessChallenges/WLCG_GlueSchemaUsage-1.8.pdf.

[38] EGEE Project. Global grid user support portal. Website, 2008. www.ggus.org/.

[39] EGEE Project. Assessment of the status of user support. Website, 2008. https://edms.cern.ch/document/951913/1.

[40] EGEE Project. Global grid user support (ggus) plan. Website, 2008. https://edms.cern.ch/document/931935.

[41] EDG Project. European datagrid project. Website, 2004. http://eu-datagrid.web.cern.ch/eu-datagrid/.

[42] European Grid Initiative. Egi blueprint. Website, 2008. http://www.eu.egi.eu/documents/other/egi-blueprint.

[43] NorduGrid. Nordugrid middleware, the advanced resource connector. Website, 2008. http://www.nordugrid.org/middleware/.

[44] UNICORE. Uniform interface to computing resources. Website, 2008. http://www.unicore.eu/.

[45] EGEE Project. Egee - regional applications registry. Website, 2009. https://na4rs.marie.hellasgrid.gr/na4/index.php.

[46] R. Nandakumar, S. Gomez, M. Adinolfi, R. Bernet, J. Blouw, D. Bortolotti, A. Carbone, B. M'Charek, D. Perego, A. Pickford, et al. The LHCb Computing Data Challenge DC06. 2007.

[47] OSG Consortium. Open science grid. Website, 2009. http://www.opensciencegrid.org/.

[48] OSG Consortium. Virtual data toolkit. Website, 2009. http://vdt.cs.wisc.edu/.

[49] VDT. What is in vdt 1.10.1. Website, 2008. http://vdt.cs.wisc.edu/releases/1.10.1/contents.html.

[50] Cdf. Website, 2009. http://www-cdf.fnal.gov/.

[51] D0. Website, 2009. http://www-d0.fnal.gov/.

[52] Zeus. Website, 2009. http://www-zeus.desy.de/.

[53] Babar. Website, 2009. http://www.slac.stanford.edu/BFROOT/.

[54] Infn - istituto nazionale di fisica nucleare. Website, 2009. http://www.infn.it.

[55] CNAF. Centro nazionale per la ricerca e sviluppo nelle tecnologie informatiche e telematiche. Website, 2008. http://www.cnaf.infn.it/.

[56] Magic. Website, 2009. http://magic.mppmu.mpg.de/.

[57] S. Burke, S. Campana, A.D. Peris, F. Donno, P.M. Lorenzo, R. Santinelli, and A. Sciaba. glite user guide. Website, 2008. https://edms.cern.ch/document/722398.

[58] Globus Project. Gsi: Grid security infrastructure. Website, 2008. http://www.globus.org/security/overview.html.

[59] International organization for standardization. Website, 2008. http://www.iso.org.

[60] The internet engineering task force. Website, 2008. http://www.ietf.org/.

[61] R. Alfieri, R. Cecchini, V. Ciaschini, L. Dell'Agnello, A. Frohner, A. Gianoli, K. Lorentey, and F. Spataro. VOMS, an authorization system for virtual organizations. *Lecture notes in computer science*, pages 33–40.

[62] Site authorisation and enforcement services: Lcas, lcmaps, and glexec. Website, 2008. http://www.nikhef.nl/grid/lcaslcmaps/.

[63] S. Andreozzi, S. Burke, L. Field, S. Fisher, B. K'onya, M. Mambelli, JM Schopf, M. Viljoen, and A. Wilson. Glue schema specification version 1.3 draft 1, 2006.

[64] The Globus Toolkit. Globus monitoring and discovery service. Website, 2008. http://www.globus.org/toolkit/mds/.

[65] EGEE Project. Berkeley database information index v5. Website, 2008. https://twiki.cern.ch/twiki/bin/view/EGEE/BDII.

[66] K. Zeilenga. RFC 4510 Lightweight Directory Access Protocol: Technical Specification Road Map, 2006.

[67] S. Burke, S. Andreozzi, and L. Field. Experiences with the GLUE information schema in the LCG/EGEE production Grid. In *Journal of Physics: Conference Series*, volume 119, page 062019. Institute of Physics Publishing, 2008.

[68] G. Good. The LDAP Data Interchange Format (LDIF)-Technical Specification, 2000.

[69] EGEE Project. Freedom of choice for resources. Website, 2008. https://cic.gridops.org/index.php?section=vo&page=freedomofchoice.

[70] EGEE project. Goc database. Website, 2008. http://www.ukiroc.eu/content/view/115/235/.

[71] EGEE Project. Job description language. Website, 2005. https://edms.cern.ch/document/590869.

[72] EGEE Project. Egee middleware architecture. Website, 2005. https://edms.cern.ch/document/594698.

[73] D. Berry, A. Luniewski, and M. Antonioletti. OGSA Data Architecture. *OGF, September*, 2007.

[74] P. Leach, M. Mealling, and R. Salz. RFC 4122: a Universally Unique IDentifier (UUID) URN Namespace. *Retrieved from http://www. ietf. org/rfc/rfc4122. txt*, 2005.

[75] IEEE. Ieee posix® certification authority. Website, 2006. http://standards.ieee.org/regauth/posix/.

[76] J.P. Baud, J. Casey, S. Lemaitre, C. Nicholson, D. Smith, and G. Stewart. LCG Data Management: From EDG to EGEE. In *UK eScience All Hands Meeting Proceedings, Nottingham, UK*, 2005.

[77] EGEE Project. File transfer service. Website, 2008. `https://twiki.cern.ch/twiki/bin/view/EGEE/FTS`.

[78] EGEE Project. Yaim. Website, 2008. `http://www.yaim.info/`.

[79] CERN. Quattor. Website, 2005. `http://quattor.web.cern.ch/quattor/`.

[80] K. Schmidt. *High Availability and Disaster Recovery: Concepts, Design, Implementation*. Springer-Verlag New York, Inc. Secaucus, NJ, USA, 2006.

[81] P. Mockapetris. RFC 1035—Domain names—implementation and specification, November 1987. *URL http://www. ietf. org/rfc/rfc1035. txt*.

[82] Citrix. Xen.org. Website, 2008. `http://www.xen.org/`.

[83] Intel Corporation. Intel virtualization technology. Website, 2008. `http://www.intel.com/technology/itj/2006/v10i3/1-hardware/6-vt-x-vt-i-solutions.htm`.

[84] Advanced Micro Devices Inc. Amd virtualization. Website, 2008. `http://www.amd.com/us-en/0,,3715_15781,00.html`.

[85] Red Hat Inc. Red hat. Website, 2009. `http://www.redhat.com/`.

[86] Scientific Linux. Scientific linux. Website, 2009. `https://www.scientificlinux.org/`.

[87] A. Chierici. A comparison between xen and kvm. `http://indico.cern.ch/contributionDisplay.py?contribId=93&sessionId=60&confId=35523`, 2009.

[88] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, and M. Xu. Web Services Agreement Specification (WS-Agreement) - GFD.107. Global Grid Forum (GGF), 2007.

[89] EGEE Project. Service level agreement between rocs and sites. Website, 2008. https://edms.cern.ch/document/860386.

[90] EGEE Project. List of egee rocs. Website, 2008. https://egee-sa1.web.cern.ch/egee-sa1/.

[91] EGEE Project. Egee sa1 operational manula. Website, 2008. https://twiki.cern.ch/twiki/bin/view/EGEE/EGEEROperationalProcedures.

[92] The distributed grid accounting system (dgas). Website, 2008. http://www.to.infn.it/grid/accounting/main.html.

[93] EGEE Project. Security and availability policy for lcg. Website, 2008. https://edms.cern.ch/document/428008.

[94] LCG Project. Sam (service availability monitoring). Website, 2008. https://twiki.cern.ch/twiki/bin/view/LCG/SAMOverview.

[95] N. Guangbao, M. Jie, and L. Bo. GridView: A Dynamic and Visual Grid Monitoring System. In *The 7th Int'l Conf. High Performance Computing and Grid in Asia Pacific Region, Omiya Sonic City, Tokyo Area, Japan*, 2004.

[96] CERN Openlab. Gridmap – visualizing the state of the grid. Website, 2008. http://gridmap.cern.ch/gm/.

[97] Nagios. Nagios: Enterprise class open source monitoring. Website, 2008. http://www.nagios.org/.

[98] Apache. Tomcat. Website, 2008. http://tomcat.apache.org.

[99] Oracle. Oracle real application clusters administration and deployment guide. Website, 2009. http://www.sysdba.de/oracle-dokumentation/11.1/rac.111/b28254/toc.htm.

[100] Oracle. Oracle streams administration guide. Website, 2008. http://download-uk.oracle.com/docs/cd/B14117_01/server.101/b10785/toc.htm.

[101] F. Bonifazi, A. Carbone, E.D. Perez, A. D'Apice, B. Martelli, et al. LHCb experience with LFC replication. In *Journal of Physics: Conference Series*, volume 119, page 042005. Institute of Physics Publishing, 2008.

[102] StoRM Project. Storm. Website, 2008. `http://storm.forge.cnaf.infn.it/home`.

[103] IBM. General parallel file system. Website, 2008. `www.ibm.com/systems/clusters/software/gpfs.html`.

[104] Carbone A., Dell'Agnello L., Forti A., Ghiselli A., Lanciotti E., Magnoni L., Mazzucato M., Santinelli R., Sapunenko V., Vagnoni V., and Zappi R. Performance studies of the StoRM Storage Resource Manager. In *In Proceedings of the 3rd IEEE International Conference on e-Science and Grid computing (eScience2007)*, Bangalore, India, Feb 2007.

[105] CNAF. Wmsmonitor project. Website, 2008. `https://twiki.cnaf.infn.it/cgi-bin/twiki/view/WMSMonitor/WebHome`.

[106] D. Cesini, D. Dongiovanni, E. Fattibene, and T. Ferrari. WMSMonitor: A monitoring tool for workload and job lifecycle in Grids. In *Grid Computing, 2008 9th IEEE/ACM International Conference on*, pages 209–216, 2008.

[107] JD Case, MS Fedor, ML Schoffstall, and C. Davin. Simple Network Management Protocol (RFC 1157). *DDN Network Information Center, SRI International, May*, 1990.

[108] EGEE Project. Assessment of the infrastructure reliability. Website, 2008. `https://edms.cern.ch/document/975596`.