r t Anno 2009_Numero 93

# rapporti tecnici INGV

# Progettazione e realizzazione di un computer-cluster per l'analisi dati GPS con i software GAMIT e QOCA

93

Istituto Nazionale di Geofisica e Vulcanologia

Anno 2009_Numero 93

rapporti tecnici INGV

# PROGETTAZIONE E REALIZZAZIONE DI UN COMPUTER-CLUSTER PER L'ANALISI DATI GPS CON I SOFTWARE GAMIT E QOCA

Enrico Serpelloni[1], Paolo Perfetti[2], Adriano Cavaliere[3]

[1]**INGV** (Istituto Nazionale di Geofisica e Vulcanologia, Centro Nazionale Terremoti)
[2]**MeTA~LAbS**
[3]**INGV** (Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Bologna)

93

## Indice

## Introduzione

Negli ultimi 5 anni si è assistito ad un rapido aumento del numero di reti di stazioni GPS continue (CGPS) attive sul territorio Italiano e, più in generale, nell'area Mediterranea. Se da un lato lo sviluppo delle reti CGPS per lo studio dei fenomeni geofisici (terremoti, vulcani, variazioni del livello del mare, ecc…) è ancora legato a particolari programmi di ricerca nei diversi paesi del bacino Mediterraneo, dall'altro un po' in tutta Europa, ma anche in alcune aree del continente Africano, si è assistito alla nascita di reti CGPS realizzate per scopi diversi da quelli geofisici (cartografici, topografici, catastali o per la navigazione). Se da una parte le reti CGPS realizzate con criteri "geofisici" [es., Anzidei & Esposito, 2003] forniscono un dato generalmente più affidabile, in termini di stabilità delle monumentazioni, qualità del dati e continuità temporale delle osservazioni, dall'altra le reti CGPS regionali di tipo "non-geofisico", nonostante una distribuzione ovviamente disomogenea, hanno dimostrato di fornire comunque informazioni utili alla stima dei campi di velocità e di deformazione crostale [es., D'Agostino et al., 2008], e di integrarsi il più delle colte con altre reti di tipo "geofisico" esistenti. Al fine di migliorare la risoluzione spaziale del segnale tettonico misurabile da una rete GPS, la scelta di realizzare un *computer cluster* per l'analisi dati GPS è stata presa al fine di garantire un rapido, ed il più possibile automatico, processamento di tutti i dati a disposizione per l'area Euro-Mediterranea ed Africana.

I software comunemente utilizzati in ambito scientifico per l'analisi dei dati GPS sono il GAMIT/GLOBK il BERNESE ed il GIPSY. Al di là delle differenze legate agli algoritmi di calcolo dei tre software in questione, e dei vantaggi o svantaggi di uno e dell'altro approccio di cui necessitano, una corretta progettazione della dotazione hardware e software è il passaggio fondamentale per la creazione di un moderno ed efficiente centro di analisi dati GPS finalizzato alla razionalizzazione delle risorse e dei costi.

Dato il numero molto elevato di stazioni CGPS oggi potenzialmente disponibili (diverse centinaia per la sola area Mediterranea), una procedura che analizzi simultaneamente tutte le stazioni è difficilmente praticabile. Nonostante recenti sviluppi di nuovi algoritmi [Blewitt, 2008] rendano effettivamente possibile un'analisi simultanea di "mega-reti", anche a scala globale, la disponibilità di calcolo su sistemi multi-processore risulta comunque fondamentale. Nel caso specifico in cui il software utilizzato per l'analisi dei dati si basi su soluzioni di rete (*network solutions*), come il BERNESE ed il GAMIT, riveste fondamentale importanza lo sfruttamento ottimale delle risorse computazionali, e soprattutto la possibilità di sfruttare appieno le potenzialità sia dei più recenti computer multi-processore che dei nuovi processori ad architettura *multi-core*. Nessuno dei software indicati precedentemente è implementato per il calcolo parallelo, di conseguenza, lo sfruttamento delle architetture multi-processore o *multi-core* deve passare necessariamente per altre vie. Una di queste è quella del calcolo distribuito (*distributed-processing*), in cui, ad esempio, diversi nodi di calcolo (che possono essere diverse macchine, diversi processori, o diversi *core* di processori) analizzano reti CGPS diverse, o diversi giorni della stessa rete CGPS. Se da una parte il mercato offre numerose soluzioni commerciali per la realizzazione di procedure di calcolo distribuito (Microsoft Windows Compute Cluster Server 2003, Sun Cluster, NEC ExpressCluster; IBM Parallel Sysplex, per citarne alcuni), dall'altra la disponibilità di software *open source* per questo tipo di scopi è oggi completa e ben integrata nei sistemi operativi UNIX based.

In questo rapporto tecnico viene descritta la procedura seguita per la realizzazione di un nuovo server per l'analisi dei dati GPS presso la Sede INGV di Bologna basato su un *computer cluster*, utilizzando software *Open Source*, in ambiente GNU/Linux.

## 1. Reti di stazioni GPS permanenti da analizzare

La Figura 1 mostra la distribuzione delle stazioni GPS continue (CGPS) attive nell'area Mediterranea, i cui dati RINEX (Receiver INdependent EXchange format) sono scaricati, archiviati ed analizzati giornalmente presso la Sede INGV di Bologna, attraverso il software GAMIT [Herring et al., 2006].

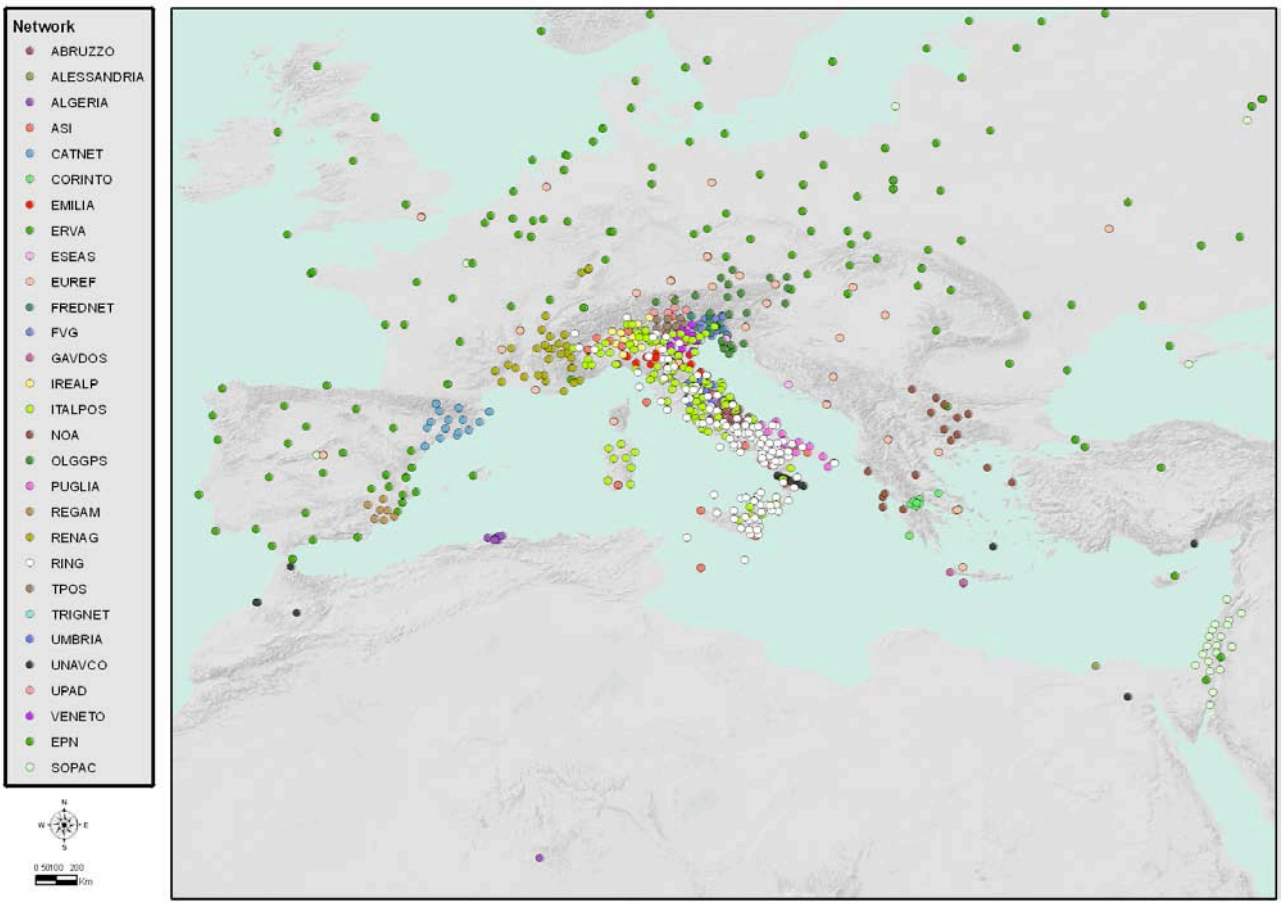


**Figura 1** La figura mostra la distribuzione delle reti di stazioni CGPS i cui dati sono archiviati ed analizzati presso la Sede INGV di Bologna.

Attualmente sono presenti in archivio i dati RINEX di circa 780 stazioni appartenenti a diverse reti CGPS operative nell'area Euro-Mediterranea ed Africana a partire dal 1993. I dati vengono scaricati ed archiviati quotidianamente con procedure automatiche. Oltre alla rete Rete Intergrata Nazione GPS (RING) dell'INGV [Selvaggi et al., 2006], si tratta di reti CGPS i cui dati sono liberamente disponibili via internet, oppure i cui dati sono resi disponibili tramite convenzioni o particolari accordi. Tale numero di reti CGPS è destinato, in ogni caso, ad aumentare nel tempo.

Vista la mole di dati da analizzare, si è deciso di progettare e sviluppare un nuovo centro di analisi dati GPS, basato sul software GAMIT, che sfrutti appieno le nuove architetture dei processori *multi-core*, per un utilizzo efficiente delle risorse di calcolo disponibili e future.

## 2. Realizzazione del *computer cluster*

Un *computer cluster*, o più semplicemente un *cluster* (dall'inglese *grappolo*), è un insieme di computer connessi tramite una rete telematica [http://it.wikipedia.org/wiki/Computer_cluster]. Lo scopo di un *cluster* è quello di distribuire una elaborazione molto complessa o molto onerosa tra i vari computer componenti il *cluster*. In sostanza un problema che richiede molte elaborazioni per essere risolto viene scomposto in sottoproblemi separati i quali vengono risolti in parallelo. I vantaggi dell'utilizzo di questo sistema sono: 1) l'economicità, infatti, questi sistemi sono fino a 15 volte più economici dei tradizionali supercalcolatori rispetto ai quali, a parità di prestazione, permettono un notevole risparmio sui componenti hardware; 2) la scalabilità, dal momento che le risorse sono distribuite;
3) la facilità di aggiornamento e manutenzione; 4) l'incremento della capacità e velocità di calcolo grazie allo sfruttamento di più unità di calcolo, di un'architettura più potente e maggiore disponibilità di memoria; 5) lo sfruttamento della cooperazione per risolvere problemi complessi; 6) l'affidabilità, in quanto il sistema continua a funzionare anche in caso di guasti a parti di esso, ovviamente con prestazioni inferiori.

Esistono tre tipi di *cluster*: 1) *Fail-over*; 2) *Load balancing* e 3) *High Performance Computing*, con i primi due che sono probabilmente i più diffusi. Nei *cluster Fail-over* il funzionamento delle macchine è continuamente monitorato, e quando uno dei due *host* smette di funzionare l'altra macchina subentra. Lo scopo è garantire un servizio continuativo. Nei *cluster* con *load balancing* le richieste di lavoro sono inviate alla macchina con meno carico. Nei *cluster HPC* i computer sono configurati per fornire prestazioni estremamente elevate. Le macchine suddividono i processi di un *job* su più macchine, al fine di guadagnare in prestazioni. La peculiarità saliente è che i processi sono parallelizzati e che le *routines* che possono girare separatamente saranno distribuite su macchine differenti invece di aspettare di essere eseguite una dopo l'altra. GLI *HPC* sono diffusi specialmente tra centri di elaborazione dati.

Per ottenere un sistema di computer operanti come un *cluster* è necessario: 1) un sistema operativo in grado di far funzionare i computer come *cluster* (per esempio GNU/Linux, utilizzando con Kernel OpenMosix), 2) hardware di rete ad elevate prestazioni, 3) un algoritmo parallelizzabile.

Come prima fase, nella creazione del nuovo *cluster* per l'analisi dati GPS si è proceduto alla valutazione della tipologia di *computer-cluster* da configurare, sulla base della tipologia di dati da analizzare, del software a disposizione, e delle risorse hardware già presenti, o disponibili presso il centro di calcolo della Sezione INGV di Bologna. In pratica il flusso di dati è caratterizzato da una fase di input da *banche dati* esterne, con archiviazione dei dati, ed analisi delle osservazioni, ed una fase di archiviazione e pubblicazione dei risultati. I dati RINEX vengono analizzati su base giornaliera per diverse reti, e successivamente combinati, sempre giorno per giorno, utilizzando il software QOCA, usato anche per la fase di post-processamento e la stima delle velocità. Il software a disposizione per l'analisi ed il post-processamento [si veda anche Serpelloni et al., 2006], non è parallelizzabile, di conseguenza l'obiettivo del nostro *computer-cluster* non è quello di suddividere gli oneri di calcolo di un singolo *job* su più nodi, bensì quello di distribuire ai nodi del *cluster* diversi *job*, indipendenti l'uno dell'altro; ossia analizzare diversi giorni o lo stesso giorno di diverse reti sui nodi disponibili, ponendosi comunque nella categoria dei *cluster HPC*.

Un *cluster HPC* è composto da due tipi di nodi: *master* (primari) e *slave* (secondari), e normalmente da un solo nodo *master* e da molti nodi *slaves*. Il nodo *master* è il *server* del *cluster* ed il suo compito è quello di coordinare i nodi *slave*, suddividendo il peso computazionale ed i *job* da eseguire. Per questo il nodo *master* richiede alcuni demoni di servizio quali DHCPD, NFS, PBS_SERVER, e PBS_SCHEDULER, e deve inoltre rendere possibili sessioni interattive e accettare la sottomissione di processi. I nodi *slave* invece attendono di ricevere istruzioni (ad esempio via ssh/rsh) dal nodo *master*. Il loro unico compito è di elaborare le istruzioni ricevute, quindi non dovrebbero avere installati servizi inutili.

È importante evidenziare che alla base della filosofia di realizzazione del *cluster* c'è stata la scelta di utilizzare software *libero* o *open-source* se disponible, o altrimenti gratuito.

## 2.1 Analisi preliminare e descrizione dell'hardware a disposizione

Le risorse critiche per una macchina dedicata al calcolo, ed in particolare all'analisi di dati GPS provenienti da reti di stazioni permanenti sono nell'ordine: 1) potenza di calcolo, 2) spazio di *storage*, 3) velocità della rete.

L'hardware a disposizione per la creazione del *cluster* consiste in 3 elaboratori multiprocessore/multi-core (Tabella 1 e Figura 2). Per sfruttare al meglio queste risorse l'idea è stata quella di utilizzare un *batch-manager*, ossia un sistema in grado di suddividere i lavori di analisi tra i differenti nodi "parallelizzando", di fatto, l'esecuzione dei processi su più *core* e massimizzando quindi la quantità di lavoro fatto in un intervallo di tempo definito (o *throughput* dei dati). Purtroppo, questa non è una caratteristica nativa nel software utilizzato per l'elaborazione dei dati GPS e quindi si è resa necessaria l'installazione di un *resource-manager* che implementa tutte le funzioni necessarie allo scopo. A tale fine è stato scelto il software TORQUE ovvero un programma *open source* che permette di gestire la suddivisione e l'esecuzione dei *job* sui nodi di calcolo nei *computer cluster*.

La seconda risorsa critica, lo spazio su disco riservato ai dati di ingresso ed ai risultati delle analisi, può essere condivisa tra gli elaboratori creando un unico archivio successivamente esportato via rete locale. Questo permette non soltanto di risparmiare risorse che altrimenti verrebbero impegnate in operazioni di inutile copia, ma anche di centralizzare le operazioni di *backup* e la sicurezza dei dati. Nella configurazione dello *storage*, infatti, si è tenuto conto non solo della quantità di spazio disponibile ma anche della tolleranza ai guasti, garantendo la consistenza dei dati anche in caso di rottura di uno dei dischi mediante l'implementazione di un sistema RAID.

Per quanto riguarda il terzo fattore di criticità invece, al fine di ottimizzare la velocità della rete si è posta particolare attenzione alla configurazione del server *Network File System* (NFS), ovvero del protocollo di condivisione in rete di directory e file con gli altri *device*. Tale configurazione non ha rappresentato, tuttavia, un vantaggio molto significativo; infatti, visti i differenti ordini di grandezza tra il tempo trascorso a reperire i dati necessari all'analisi e quello deputato al puro calcolo, le prestazioni del server NFS non hanno rappresentato un fattore critico.

Infine, per aumentare l'usabilità del sistema e la sua facilità di manutenzione, si sono resi necessari alcuni accorgimenti, quali l'installazione di server X, DNS (*Domain Name System)*, ntpd (*Network Time Protocol daemon*) e rsyncd (per la sincronizzazione ed il back up su altri *device*) e la creazione di alcuni script di controllo.

**1 Nodo principale**
HOST: spritz.bo.ingv.it
CPU: 2x Intel Xeon E5410 @ 2.33GHz (QuadCore)
RAM: 8GB DDR2-667 ECC
STORAGE: 3ware Inc 9650SE PCI-E - SATA-II RAID Controller
8x Maxtor STM3500320AS 465.76 GB

**2 Nodi di calcolo**
HOST: aperol.spritz-cluster.bo.ingv.it
CPU: 2x Intel Xeon CPU 3.20GHz
RAM: 4GB DDR2
STORAGE: 2x Maxtor 6L250R0 250GB
1x Seagate ST3500641AS 500GB

HOST: campari.spritz-cluster.bo.ingv.it
CPU: 2x Intel Xeon CPU 3.20GHz
RAM: 2GB DDR2
STORAGE: 1x Maxtor STM3250820A 250GB
1x Maxtor 6L250R0 250GB
1x Seagate ST3500641AS 500GB

**3 Rete**
SWITCH GIGABIT: D-Link DGS-1008D

**Tabella 1** Configurazione dell'hardware a disposizione.

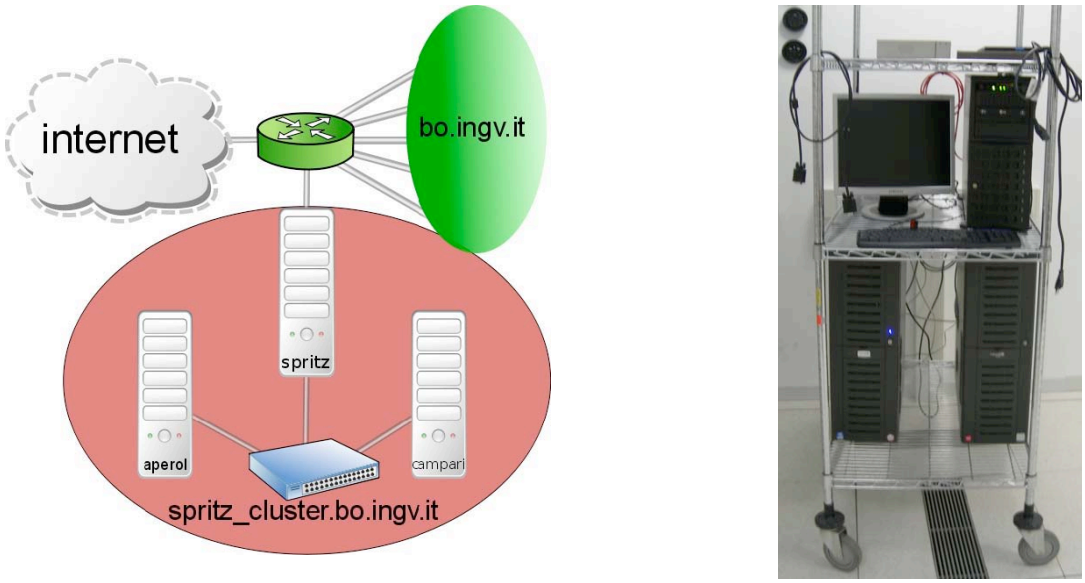


**Figura 2** Schema del *cluster* e foto dei tre computer multi-processore, collegati via rete Gbit, che lo costituiscono.

## 3. Nodo principale (*master*)

In questo paragrafo viene descritta la configurazione del nodo principale (host: spritz.bo.ingv.it), in termini di *storage*, sistema operativo e servizi.

### 3.1 Configurazione dello *storage*

La configurazione hardware del nodo principale prevede un controller RAID (*Redundant Array of Independent Disks*) *3ware* 9650SE/8ML SATA II ed 8 dischi Maxtor STM3500320AS da 465.76 GB. Questa scheda RAID mette a disposizione differenti schemi di ridondanza e concatenazione (RAID *levels* 0, 1, 5, 6, 10, 50; per una introduzione rapida ai sistemi RAID si veda la pagina: http://it.wikipedia.org/wiki/RAID) che permettono all'utente di scegliere il compromesso migliore tra sicurezza/prestazioni/spazio disponibile, a seconda delle esigenze specifiche. Considerati i risultati dell'analisi preliminare, descritta nel paragrafo 2, ed avendo a disposizione 8 dischi identici la scelta è stata quella di privilegiare la quantità di spazio utilizzabile senza rinunciare ad un livello di sicurezza accettabile, garantendo una certa tolleranza ai guasti (*Fault Tolerance), ovvero* di non subire interruzioni di servizio anche in presenza di guasti Questo è stato possibile configurando due unità logiche di quattro dischi ciascuna con un livello di RAID5 (4 dischi: 3 per i dati ed 1 per la parità distribuita ~ 1.5TB) e successivamente concatenandole in un RAID0 (2*1.5TB =~ 3TB).

Si sarebbe potuta ottenere una configurazione con la medesima tolleranza ai guasti e prestazioni in lettura migliori configurando un RAID1, ma ciò avrebbe comportato notevole perdita di spazio utile (circa 2 TB anziché 3TB, con una diminuzione del 33%); trattandosi in questo caso di analisi che richiedono pochi dati per ogni singolo processo (o *job*), a fronte di in un tempo di calcolo molto elevato ed una quantità di dati raccolti imponente si è preferito quindi un RAID5, trascurando coscientemente una perdita di prestazioni comunque ininfluente sul lavoro complessivo da svolgere. Per le stesse considerazioni riguardo allo spazio di *storage* utile si è scelto di non implementare un livello di RAID6 anche se questo avrebbe aumentato la resistenza ai guasti (2 dischi).

| Mount point | Fs | Dimensione | Utilizzo |
|---|---|---|---|
| / | ext3 | 966 MB | Root directory |
| /boot | ext3 | 130 MB | File necessari al boot loader |
| /tmp | ext3 | 2.0 GB | Dati volatili e temporanei per il funzionamento del s.o. |
| /usr | ext3 | 10 GB | Programmi, librerie e dati statici necessari al s.o. |
| /var | ext3 | 10 GB | Dati di dimensione variabile, log e spool necessari al s.o. |
| /home | ext3 | 10 GB | Dati privati e configurazioni degli utenti |
| /opt | ext3 | 20 GB | Sorgenti e programmi necessari all'analisi dei dati GPS |
| /data | xfs | 2.0 TB | Storage dedicato ai dati e alle osservazioni GPS |

**Tabella 2** Partizioni e loro utilizzo.

### 3.2 Sistema Operativo

Il sistema operativo (SO) prescelto per l'installazione sui tre nodi del *cluster* è stato *Gentoo GNU/Linux 2008.0*, decisione effettuata in parte a priori al fine di mantenere l'omogeneità con altre macchine da calcolo presenti in istituto. Tale scelta si è rivelata ottima per le note caratteristiche di

flessibilità e configurabilità del software *open source* in genere e di questa distribuzione in particolare. Inoltre, grazie al sistema di pacchettizzazione su cui si basa Gentoo (il cosiddetto *portage*) è stato possibile effettuare una istallazione ed una configurazione perfettamente modellata sulle esigenze di questo progetto. Per esempio si sono potuti risolvere i problemi dovuti alla richiesta di compilatori e librerie differenti da parte di programmi quali GAMIT/GLOBK e QOCA rimanendo all'interno della gestione di routine del sistema; garantendo in questo modo un'estrema facilità nella manutenzione e una comprovata resistenza ai futuri aggiornamenti.

```
spritz ~ # equery list gcc
[ Searching for package 'gcc' in all categories among: ]
* installed packages
[I--] [ -] sys-devel/gcc-3.4.6-r2 (3.4)
[I--] [ -] sys-devel/gcc-4.1.2 (4.1)
[I--] [ ~] sys-devel/gcc-4.2.4 (4.2)
```

**Figura 3** Esempio di convivenza tra differenti versioni dello stesso programma.

Durante l'installazione si è proceduto alla suddivisione dello spazio disponibile nelle 8 partizioni (+ 1 per lo *swap*) necessarie (Tabella 2). Per garantire una notevole flessibilità nella gestione dello spazio, sia per usi futuri sia per far fronte a necessità impreviste, a livello più basso lo spazio è gestito tramite *Logical Volume Manager* (LVM). LVM è un metodo di allocazione dello spazio del disco fisso in volumi logici che possono essere facilmente ridimensionati (al contrario delle tradizionali partizioni fisiche), e garantisce un'ulteriore livello di astrazione dai dischi che permette di allocare dinamicamente lo spazio dando così la possibilità di allargare/restringere le partizioni senza riformattare i dischi e, in alcuni casi, addirittura senza smontare le partizioni stesse.

Per le partizioni di sistema è stato utilizzato il *filesystem ext3*, presente di default in Linux e maggiormente supportato degli altri disponibili, mentre per la partizione di *storage* si è preferito XFS. Questa scelta è giustificata principalmente da 3 motivi:

1) la grande scalabilità di XFS, che permette partizioni e file di dimensione fino a 8 exabytes;

2) la possibilità, in un sistema multi-processore o *multi-core* come quello a disposizione, di utilizzare più processi concorrenti per la realizzazione di operazioni contemporanee sul filesystem. Questo, sommato ad una opportuna configurazione del RAID sottostante e di XFS stesso, permette di sfruttare maggiormente i vantaggi di avere un *filesystem* distribuito su diversi *device* fisici (maggior banda passante per le operazioni su disco);

3) la possibilità di estendere il *filesystem* (grazie anche alla presenza di LVM) senza dover riformattare e neppure disattivare (smontare) la partizione (*online resizing*).

## 3.3 Configurazione dei servizi

Prima di installare gli strumenti di sviluppo, è stato necessario configurare alcuni servizi accessori indispensabili per il funzionamento del *cluster*, che sono qui elencati:

1) NFS (*Network File System*): Data l'omogeneità delle architetture degli elaboratori a disposizione (Tabella 1) non solo è stato possibile condividere i dati evitando inutili (e potenzialmente addirittura dannose) copie locali dei dati, ma anche i programmi, compilati una sola volta sul nodo principale e resi poi disponibili ai nodi secondari via rete, realizzando così un punto unico di manutenzione ed aggiornamento. Per renderlo possibile è stato necessario, sfruttando alcuni *link* simbolici e opportuni parametri di compilazione, ottenere un'accurata separazione tra i file eseguibili (uguali per tutti i nodi) e i file di configurazione e dati temporanei (potenzialmente dipendenti dal singolo elaboratore). Le condivisioni esportate sono quindi due: `/data`, contenente i dati per l'analisi, e `/opt` per i programmi. Per massimizzare le prestazioni di NFS inoltre, sono state aumentate le dimensioni dei *buffer* in lettura e scrittura configurate di default, diminuendo il numero di richieste e sfruttando maggiormente la rete al GB;

2) RSYNCD: mette a disposizione dei *client* del *cluster* l'albero del sistema di pacchettizzazione aggiornato (*portage*), incrementandone notevolmente le performance di aggiornamento ed evitando di sovraccaricare inutilmente sia la connessione ad internet sia i server esterni utilizzati;

3) X e XDMCP: alcuni dei programmi installati rendono disponibili dei *tool* grafici di semplice utilizzo per la gestione/monitoraggio dei loro compiti. Per sfruttare questi strumenti e per usufruire di tutte le funzionalità di GAMIT, e dei programmi di post-processamento, è stato necessario installare un server X-window e le librerie necessarie al suo corretto funzionamento. Da remoto, l'utilizzo di questi programmi è possibile in differenti maniere: lanciando il *client* ssh con opportuni parametri (-X) oppure, grazie ad un sistema di *login* grafico come XDM, esportando un'intera sessione X su un server locale in esecuzione sul *client* da cui si sta lavorando. Per esempio, per Windows esiste *Xming*, un server X *open source* che supporta queste funzionalità e che in fase di testing si è rivelato sufficientemente curato e semplice. Utilizzando questa seconda opzione, infatti, è sufficiente specificare l'indirizzo del server ed servizio a cui di vuole accedere per trovarsi di fronte ad un *login* grafico del tutto identico a quello che si presenterebbe all'utente locale ed operare sul server con tutta la comodità data da un ambiente X. In Figura 4 è mostrato un esempio di desktop remoto da *Xming*;

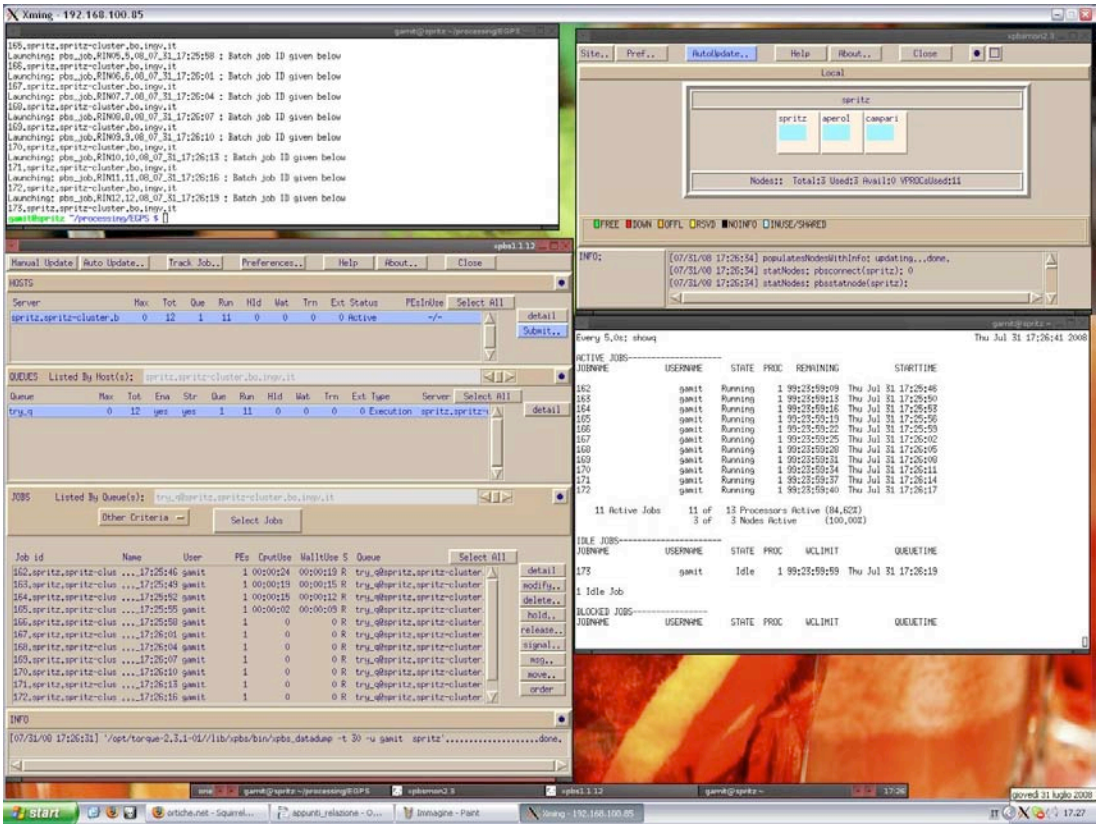

**Figura 4** *Screen-shot* del desktop remoto da *Xming* sul nodo principale (*spritz*).

4) POSTFIX: per la corretta manutenzione e comunicazione, sia dei risultati dei lavori di analisi, sia delle operazioni e controlli periodici , si è reso necessario installare un sistema di posta che provvedesse all'invio delle e-mail attraverso un server SMTP. Il recapito delle e-mail locali è disabilitato e tutte le comunicazioni rivolte agli utenti UNIX locali sono inoltrate ad indirizzi configurabili nel file `/etc/mail/aliases`;

5) NTPD: si tratta di un server dedicato al mantenimento dell'ora corretta sul sistema su cui si trova installato ed (eventualmente) i suoi *client*. Sul nodo principale si occupa anche di fornire il servizio NTP (*Network Time Protocol*) ai nodi di calcolo, evitando problemi dovuti alla mancata sincronizzazione dei diversi elaboratori;

6) BIND e NAT: per il funzionamento di TORQUE è necessaria una corretta risoluzione dei nomi DNS e dei rispettivi indirizzi (*reverse* DNS). Per mantenere la gestione del *cluster* semplice ed indipendente dal resto della rete locale si è preferito ricorrere quindi all'installazione di un server DNS locale, delegato solo alla risoluzione della sottorete *spritz-cluster.bo.ingv.it*. Il servizio di NAT (*Network Address Translation*) da invece la possibilità ai nodi del *cluster* di accedere ai servizi di rete esterni al *cluster*, potenzialmente necessario per il reperimento di dati e/o aggiornamenti software;

7) `/etc/env.d/99local`: diverse soluzioni specifiche adottate che richiedono la modifica di alcune variabili globali e del comportamento del sistema devono necessariamente sopravvivere al riavvio del sistema ed essere disponibili per tutte le sessioni aperte dagli utenti. Queste modifiche sono state apportate nel file `/etc/env.d/99local`, incluso nella procedura di inizializzazione dell'ambiente in cui vengono eseguite le *shell* degli utenti (e quindi tutti i programmi da lì eseguiti). Anche questa scelta è giustificata dalla facilità di manutenzione e dalla coerenza con quando previsto dalla distribuzione installata.

## 3.4 Programmi accessori per l'analisi dei dati

Per la post-elaborazione dei dati e la successiva rappresentazione grafica, dei risultati delle analisi sono necessari non inclusi della suite GAMIT/GLOBK, quali Generic Mapping Tools (GMT) e QOCA (Quasi Observation Combination Analysis):

1) GMT: è una collezione di circa 60 programmi per la manipolazione di dati geografici e cartesiani e per produrre immagini EPS (Encapsulated PostScript File) rappresentanti dai semplici grafici x-y a complesse prospettive 3D. La versione installata è la 4.1.1, la più aggiornata presente nell'albero dei *portage*, e viene utilizzata per la rappresentazione grafica dei risultati delle analisi;

2) QOCA: questo programma, sviluppato da Danan Dong al Jet Propulsion Laboratory, permette di combinare e compensare vari tipi di soluzioni (SINEX, GAMIT, GIPSY, ecc…), e di eseguire l'analisi delle serie temporali [es., Serpelloni et al., 2006]. Di questo programma sono disponibili i soli file eseguibili, compilati a 32 bit. Per permettere il funzionamento della suite di programmi che fanno parte di QOCA quindi, nelle fasi iniziali dell'installazione di Gentoo è stato scelto di configurare un ambiente *multilib* (come anticipato nel paragrafo 2.2), un ambiente cioè che gestisca librerie compilate per diverse architetture, quali la x86 a 32 bit e la amd64 a 64 bit (il nome dell'architettura rimane adm64 anche se i processori sono in realtà Intel). Le librerie necessarie sono state quindi installate nelle cartelle `/lib32` e `/usr/lib/gcc/x86_64-pc-linux-gnu/3.4.6/32/` e la variabile d'ambiente utilizzata dal *linker*, LDPATH, è stata modificata per comprendere le precedenti directory.

## 3.5 Installazione e configurazione della suite GAMIT/GLOBK

Il lavoro principale di calcolo svolto dal *cluster* è l'analisi dei dati GPS attraverso la suite di programmi GAMIT/GLOBK sviluppata dal Department of Earth Atmospheric and Planetary Sciences del MIT. Lo scopo di questo tipo di installazione è quello di "parallelizzare" il più possibile diverse elaborazioni, sfruttando le potenzialità dell'architettura multi-processore/*multi-core* disponile, caratteristica non presente di default in GAMIT e QOCA, cosi come in tutti gli altri software scientifici per l'analisi dei dati GPS. La versione disponibile al momento della configurazione era la 10.33 (ora è installata la versione più recente, la 10.34) ed è stata scaricata ed installata all'interno della directory `/opt`. Per facilitarne la manutenzione e l'aggiornamento sono stati creati alcuni *link* alla cartella della versione corrente, garantendo così la consistenza dei percorsi esportati in variabili di sistema come PATH e LDPATH. In questa maniera, ad ogni aggiornamento del software sarà sufficiente modificare gli opportuni collegamenti per passare alla nuova versione senza rischiare incoerenze tra i diversi eseguibili.

Per la corretta compilazione della suite è necessaria una versione del compilatore GCC, appartenente al ramo 4.2.x, non presente e non configurato come predefinito nella distribuzione 2008.0 di Gentoo e quindi installato oltre alla versione disponibile di default con il sistema (la 4.1.x). Questo è stato possibile tramite gli SLOT utilizzati dai *portage*, che permettono di avere differenti versioni di uno stesso programma senza causare conflitti: alla conclusione dei lavori saranno presenti addirittura 3 differenti pacchetti di GCC senza problemi di sorta. Il file `/opt/gg/libraries/Makefile.common`, che serve per l'installazione dei codici sorgente del GAMIT, è stato modificato per utilizzare la versione corretta dei compilatori (GCC e GFORTRAN) con le migliori opzioni di ottimizzazione e sono stati modificati alcuni parametri delle reti in analisi (come il numero massimo di siti, ecc…). Purtroppo, nella pacchettizzazione di GAMIT e nella sua procedura di configurazione/compilazione non è prevista una maniera comoda di conservare queste modifiche attraverso differenti versioni e si rende quindi necessario effettuare le medesime modifiche ad ogni ricompilazione della suite.

L'utente dedicato ai lavori di analisi e per cui sono stati preparati l'ambiente ed i permessi sui dati ed eseguibili è l'utente *gamit*: nella home si troverà infatti un *link* all'installazione della suite GAMIT ed i programmi e gli *script* accessori installati sono stati configurati per rimanere coerenti con i permessi assegnati. Infine, per le ragioni esposte precedentemente dal paragrafo 3.3, la directory `/opt` (e quindi l'installazione di GAMIT/GLOBK, e tutti gli altri programmi installati) è esportata in sola lettura ai *client* del *cluster* permettendo di avere un unica versione coerente e di facile manutenzione della suite e di tutti i programmi accessori disponibile per tutti i nodi di calcolo.

## 4. Nodi secondari (*slaves*)

L'installazione dei due nodi di calcolo del *cluster* è avvenuta in maniera parallela e seguendo molte delle operazioni eseguite per il nodo principale. Il sistema operativo installato è lo stesso, Gentoo *GNU/Linux* 2008.0, e grazie a questo alcune modifiche apportate all'ambiente del SO (come il file `/etc/env.d/99local`) sono state direttamente esportate da *spritz*; per altri servizi invece, come ad esempio NFS, NTPD ed altri, sono state create configurazioni apposite cercando di rendere i nodi più indipendenti possibile dalla rete esterna a quella locale tra le macchine di *spritz_cluster*.

### 4.1 Network File System (NFS)

I due nodi di calcolo sono configurati per montare in fase di avvio le due partizioni esportate dal nodo principale: `/data` contenente i dati ed i risultati delle analisi, e `/opt` contenente il software necessario all'elaborazione. I percorsi su cui vengono configurate questi due *filesystem* di rete sono i medesimi di SPRITZ, semplificando la gestione delle configurazioni e dei percorsi (sia degli eseguibili che dei dati) sia nella fase di installazione sia nella fase di calcolo vero e proprio.

La partizione di *storage* è scrivibile anche dai *client*, caratteristica necessaria visto che i risultati delle analisi vengono scritti nelle stesse cartelle da dove vengono reperiti i dati, mentre la partizione del software è in sola lettura, evitando così inutili problemi e anticipando alla fase d'installazione l'emergere di eventuali problemi.

### 4.2 NTPD, DNS, RSYNCD

Per questi servizi è stato configurato il nodo principale come server; con questa soluzione si risparmiano inutili accessi ad Internet anche per operazioni che potenzialmente possono generare molto traffico di rete (e quindi rallentamenti) con la sincronizzazione dell'albero dei *portage*. Inoltre, il nodo principale agisce come server NAT e DNS per i propri *client*, dando come vantaggio l'indipendenza dell'ambiente di rete interno al *cluster* da quello esterno e viceversa; grazie a questo sarà possibile aggiungere altri nodi senza dover modificare alcunché nell'infrastruttura esterna ed allo stesso tempo sarebbe possibile spostare l'intero *cluster* in un altro ambiente apportando solo poche modifiche ad alcune configurazioni del solo nodo principale.

### 4.3 *Storage*

Su entrambi gli elaboratori *client* sono presenti 3 dischi fissi: 2 da 250GB configurati in RAID software per l'integrità del sistema, ed uno da circa 500GB precedentemente utilizzato per la memorizzazione dei dati dedicati all'analisi. Si è pensato, per utilizzare al meglio le risorse ed innalzare il livello di *Failure Tolerance* del sistema, di utilizzare quindi questi due dischi (e parte dei RAID software inutilizzato) per mantenere una copia dei dati presenti nello *storage* sul nodo principale, divisa tra i due *client*. Per l'aggiornamento del back-up, eseguito settimanalmente, sono stati utilizzati degli *script* automatizzati e periodici, basati su RSYNC.

## 5. *Portable Batch System* (PBS)

Con *Portable Batch System* (PBS) si indica un sistema *software* il cui compito è quello di gestire delle code di *job* (o *batch* appunto) allocando e sfruttando al meglio le risorse computazionali rese disponibili sul *cluster* in cui è installato e configurato. L'interazione con un sistema di questo tipo avviene tramite degli *script*; sarà poi compito del *resource manager*, in accordo con le politiche definite, occuparsi di reperire le risorse necessarie, creare l'ambiente opportuno, eseguire il *job* e restituirne il risultato o gestirne gli eventuali errori a *run-time*.

Data la flessibilità di soluzioni di questo tipo, i *Portable Batch System* sono spesso utilizzati per gestire grandi *pool* di risorse o per aumentare il parallelismo di programmi non multiprocesso/*multithread* ma suddividibili in elaborazioni parziali ed indipendenti.

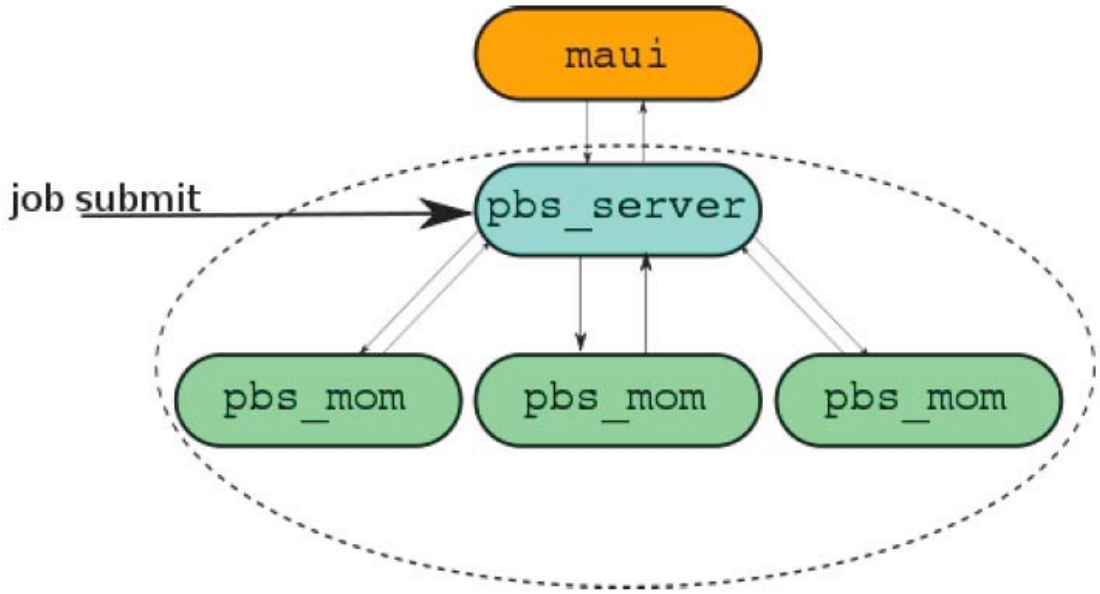


**Figura 5** Schema di funzionamento del sistema PBS.

### 5.1 *Resource manager* – TORQUE

TORQUE è un *resource manager open source* che si occupa di gestire l'esecuzione dei *job* assegnati su un insieme di nodi di calcolo, assegnando priorità e risorse in base a quanto disponibile e alle *policy* stabilite dallo *scheduler* (nel nostro caso MAUI, di cui si parlerà successivamente).
Per le finalità progettuali del *cluster*, ossia l'analisi di dati provenienti da reti di stazioni GPS, le caratteristiche offerte da TORQUE si sono rivelate esaustive ed addirittura sovradimensionate rispetto alle necessità. Ad esempio, molte caratteristiche di questo software sono sviluppate per la gestione del lato concorrenziale dei job in esecuzione (differenti operatori che necessitano di potenza di calcolo per i propri fini indipendenti), come la possibilità di definire diverse code con diverse caratteristiche e risorse assegnate. Nel nostro caso, invece, questo aspetto è del tutto marginale data la natura collaborativa delle analisi da effettuarsi (differenti *job* contribuiscono ad un unico risultato) e verranno quindi privilegiate le prestazioni e la minimizzazione del tempo necessario per lo svolgimento dei *job*.

### 5.2 Installazione

In fase di istallazione si è scelto di non ricorrere alla pacchettizzazione offerta dal sistema operativo, bensì di scaricare i sorgenti del software direttamente dal sito degli sviluppatori e compilandolo a parte; questa decisione è dovuta in parte alla differenza di versioni disponibili, in parte alla necessità di rendere accessibili gli eseguibili anche ai nodi di calcolo sfruttando le condivisioni di rete già predisposte.
Specificando delle directory differenti da quelle di default per i dati variabili (configurazioni, *log*, *spool*), è stato possibile condividere l'installazione mantenendo l'indipendenza dei singoli nodi: gli eseguibili (uguali

per tutti) sono installati in `/opt/torque`, mentre gli altri dati sono memorizzati in `/var/torque` (cartella non esportata ma opportunamente creata su ogni elaboratore).

### 5.3 Configurazione del nodo *master* (*spritz*)

Terminata la fase di installazione e modificato l'ambiente per includere le cartelle contenenti gli eseguibili, si è proceduto alla fase di configurazione necessaria affinché il *resource manager* conosca quali sono le risorse effettivamente disponibili ed allocabili: più in generale la directory `/var/torque/server_priv/` che contiene tutti i file necessari al corretto funzionamento di pbs_server (ovvero il *resource manager* del *cluster);* in particolare il file `nodes` che contiene la lista dei nomi dei *client* e le risorse presenti per ogni elaboratore, nel nostro caso il solo numero di *core* dedicati al calcolo. Dato il tipo di analisi da effettuare, si è deciso di assegnare ogni singolo core ad un *job* differente. Infatti, essendo i programmi utilizzati in grado di utilizzare un solo core per volta, sarebbe stato uno spreco assegnarvi un intero processore (quindi 2 o 4 *core*) con il risultato di avere i rimanenti *core* del processore scarichi da ogni tipo calcolo utile. Come convenzione, il numero di *core* disponibili per calcolatore consiste nel numero totale di *core* presenti diminuito di uno, assicurando così la disponibilità costante di almeno un *core* dedicato alla normale gestione del sistema operativo.

Le altre risorse non sono state specificate essendo la potenza di calcolo l'unico fattore limitante nel nostro contesto (lo spazio su disco e la memoria non sono infatti critici per la singola elaborazione).

### 5.4 Configurazione dei nodi di calcolo (*aperol* e *campari*)

Tutti gli elaboratori coinvolti nel *cluster* vengono intesi cone nodi di calcolo. Nel nostro caso l'insieme di tali nodi di calcolo comprende anche il nodo principale a cui è stato affidato anche parte del calcolo in virtù dell'esiguo carico associato al compito di coordinamento dei *job* e delle risorse. Il programma che si occupa di comunicare con pbs_server si chiama pbs_mom e viene eseguito come demone su ogni nodo; la sua configurazione è specificata in un file locale contenuto nella cartella `/var/torque/mom_priv` e descrive principalmente:

1) quante informazioni conservare sull'esecuzione dei *job* (*logging*);
2) l'intervallo di tempo con cui controllare la presenza di nuovi *job*;
3) l'intervallo di aggiornamento dello stato del nodo;
4) le cartelle remote montate localmente (per utilizzare il comando *cp* anziché *scp*);
5) l'indirizzo del server a cui notificare lo stato e da cui reperire i *job*.

Oltre alla configurazione specifica si è resa necessaria anche la creazione dell'utente *gamit* sui nodi secondari. Tale condizione ha garantito la corretta gestione dei permessi sia per la condivisione di rete sia per l'accesso ai *job* (sottoposti a TORQUE); questi infatti sono eseguiti con lo stesso utente che li crea sul *Submission Node,* corrispondenti, nel nostro caso, al nodo principale.

## 6. Lo *scheduler* – MAUI

Punto critico nell'efficienza di un sistema PBS è senz'altro lo *scheduler*, il programma cioè che si occupa, sulla base di politiche (*policy*) definite a priori, di riservare ed assegnare le risorse ai job secondo la disponibilità. Dato il carattere collaborativi e parallelo dei *job*, non si rendono necessarie particolari politiche di *scheduling* che almeno in questa prima fase dell'utilizzo sono state semplificate nella tecnica FIFO (*First In First Out*, ossia il primo arrivato è il primo servito). Malgrado questo semplice meccanismo di *scheduling* fosse già presente di default in TORQUE si è scelto di installare comunque MAUI per il maggior supporto disponibile e per le opportunità di sviluppo ed espansione che offre.

Interlocutore unico di MAUI è il demone pbs_server in esecuzione sul nodo principale: lo *scheduler* infatti si occupa soltanto di interrogare il server riguardo ai *job* in attesa e alle risorse disponibili istruendolo successivamente sulle azioni da intraprendere. Questo permette di eliminare la comunicazione diretta con i nodi di calcolo, semplificando l'infrastruttura e centralizzando le procedure di controllo (anche se, ovviamente dato che la comunicazione avviene via rete, lo *scheduler* e pbs_server potrebbero essere in esecuzione su elaboratori differenti).

## 6.1 Installazione

Anche per MAUI, al fine di ottenere la versione più aggiornata e non modificare le directory di default, si è scelto di scaricare i sorgenti direttamente dal sito ed installarlo al di fuori dei *portage* di Gentoo Linux. In questo caso però, viene meno la necessità di condividere gli eseguibili con il nodi di calcolo, completamente indipendenti dal lavoro di *scheduling* effettuato da MAUI; anche se la cartella d'installazione rimane comunque all'interno di `/opt`. Come opzione supplementare si è inoltre specificata la posizione nel *filesytem* di TORQUE (`/opt/torque`), anch'essa differente da quella predefinita nel pacchetto fornito dalla distribuzione.

## 6.2 Configurazione

Vista la semplicità del contesto in cui opera MAUI (niente concorrenza, risorse standard, politica FIFO), anche la sua configurazione risulta agevole. È stato sufficiente specificare il tipo di *resource manager* (nel nostro caso PBS, ma MAUI ne supporta differenti), l'indirizzo del nodo su cui è in esecuzione pbs_server e le politiche di allocazione per i *job*. Per questa ultima si è scelto di affidare i nuovi *job* sempre al nodo con minor utilizzo di risorse percentuali al momento dello *scheduling*, ottimizzando così il parallelismo nell'elaborazione delle analisi. Infine, si è definito un apposito utente di sistema per l'esecuzione dello *scheduler*, chiamato *pbs*, rispettando cosi il principio dei minimi privilegi e dando al programma solo la possibilità di eseguire le operazioni necessarie al suo corretto funzionamento, evitando inutili problemi di sicurezza/integrità del sistema.

## 7. Script e personalizzazioni

Per la gestione del sistema da parte del personale deputato al lavoro di analisi, è stato creato e configurato opportunamente l'utente UNIX *gamit*: sia le variabili d'ambiente sia i permessi sono stati modificati per ridurre al minimo la necessità di utilizzare l'utente *root*. Per l'esecuzione di compiti periodici è utilizzabile *cron*, il demone apposito e con supporto multi-utente: sarà infatti sufficiente eseguire dalla normale *shell* dell'utente il comando *crontab -e* per aggiungere/modifcare i processi e la frequenza con cui verranno eseguiti. Per la sottomissione dei *batch* al *resource manager* è stato creato uno script, *sh_aperitivi*, fortemente basato su *sh_PBS_glred*, uno dei programmi messi a disposizione dalla suite GAMIT/GLOBK. Inoltre, tutto il software installato è stato integrato con il sistema operativo, garantendo omogeneità nelle routine di gestione: per esempio per tutti i demoni (per cui non erano presenti di default) sono stati creati gli script di start e stop in `/etc/init.d`, rispettando le convenzioni della distribuzione. Oltretutto, grazie a questi script, dopo un normale riavvio andato a buon fine, tutto l'ambiente si troverà già predisposto nella situazione ottimale per svolgere il lavoro di analisi, non richiedendo ulteriori interventi manuali per l'avvio di programmi e servizi particolari.

## Bibliografia


Anzidei, M. e Esposito, A., (2003). *Linee guida per la identificazione di siti idonei alla realizzazione di stazioni GPS permanenti e non permanenti*, Rapporti Tecnici dell'Istituto Nazionale di Geofisica e Vulcanologià.

Blewitt, G., (2008). *Fixed point theorems of GPS carrier phase ambiguity resolution and their application to massive network processing: Ambizap*, J. Geophys. Res., 113, B12410, doi:10.1029/2008JB005736.

D'Agostino, N., Mantenuto, S., D'Anastasio, E., Avallone, A., Barchi, M., Collettini, C., Radicioni, F., Stoppini, A., and Fastellini, G., (2008). *Contemporary crustal extension in the Umbria-Marche Apennines from regional CGPS networks and comparison between geodetic and seismic deformation*, Tectonophysics, doi: 10.1016/j.tecto.2008.09.033.

Herring, T.A., King, R.W. and McClusky, S., (2006). *Documentation for GAMIT GPS Analysis Software, version 10.3*, Department of Earth, Atmospheric, and Planetary Sciences, Massachusetts Institute of Technology.

Selvaggi, G., Mattia M., Avallone A., D'Agostino N., Anzidei M., Cantarero M., Cardinale V., Castagnozzi A., Casula G., Cecere G., Cogliano R., Criscuoli F., D'Ambrosio C., D'Anastasio E., DE Martino P., DEL Mese S., Devoti R., Falco L., Galvani A., Giovani L., Hunstad I., Massucci A., Minichiello F., Memmolo A., Migliari F., Moschillo R., Obrizzo F., Pietrantonio G., Pignone M., Pulvirenti M., Rossi M., Riguzzi F., Serpelloni E., Tammaro U. & Zarrilli L. (2006), *La Rete Integrata Nazionale GPS (RING) dell'INGV: un'infrastruttura aperta per la ricerca scientifica*, *X Conferenza ASITA, Bolzano, Atti Vol. II, 1749-1754*.

Serpelloni, E., Casula, G., Galvani, A., Anzidei, M. and Baldi, P., (2006). *Data analysis of permanent GPS networks in Italy and surrounding regions: application of a distributed processing approach*, Annals of Geophysics, 49 (4-5), 1073-1104.

## Risorse web

### Concetti generali

COMPUTER CLUSTER: http://it.wikipedia.org/wiki/Computer_cluster

LVM: http://it.wikipedia.org/wiki/Gestore_logico_dei_volumi

RAID: http://it.wikipedia.org/wiki/RAID

XFS: http://it.wikipedia.org/wiki/XFS_(file_system)

### Software e tecnologie utilizzate

GCC: http://gcc.gnu.org/

GMT: http://www.soest.hawaii.edu/gmt

MAUI: http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php

QOCA: http://gipsy.jpl.nasa.gov/qoca

TORQUE: http://www.clusterresources.com/pages/products/torque-resource-manager.php

XDM: http://it.wikipedia.org/wiki/XDM_(programma)

**Coordinamento editoriale e impaginazione**
Centro Editoriale Nazionale | INGV

**Progetto grafico e redazionale**
Laboratorio Grafica e Immagini | INGV Roma


Via di Vigna Murata, 605
00143 Roma
Tel. +39 06518601 Fax +39 065041181

**http://www.ingv.it**

Istituto Nazionale di Geofisica e Vulcanologia