ISSN 2039-7941
Anno 2013_Numero 251

rapporti tecnici INGV

Nuova interfaccia WEB della stazione sismica digitale GAIA2

251

Istituto Nazionale di Geofisica e Vulcanologia

ISSN 2039-7941
Anno 2013_Numero 251

rapporti tecnici

INGV

NUOVA INTERFACCIA WEB DELLA STAZIONE SISMICA DIGITALE GAIA2

Leonardo Salvaterra, Sandro Rao

INGV (Istituto Nazionale di Geofisica e Vulcanologia, Centro Nazionale Terremoti)

251

Indice

Introduzione

Il lavoro, di seguito presentato, riguarda sostanzialmente la realizzazione degli sviluppi ipotizzati in un precedente Rapporto Tecnico INGV [Rao e Salvaterra, 2011] sull'interfaccia WEB della stazione sismica GAIA2, nominata WEBMON e ideata all'interno del Laboratorio di Sismologia dell'INGV. La prima versione dell'interfaccia era composta di un'unica pagina WEB caratterizzata da una grafica molto semplice e con la sola possibilità, da parte dell'utente, di visualizzare le informazioni sul funzionamento della stazione. Avendo reputato WEBMON uno strumento molto potente giacché, basandosi su protocolli di tipo WEB, è in sostanza svincolato dal sistema operativo dal quale viene visualizzato, ci siamo resi conto di quanto fosse importante dotarlo della possibilità di poter cambiare i parametri di configurazione e quindi di interagire attivamente con la stazione sismica. Il nuovo sviluppo, perciò, si è concentrato sul miglioramento della grafica per la presentazione delle informazioni e sull'aggiunta di nuovi strumenti per la configurazione della GAIA2 quali il poter sceglier l'intervallo temporale dei dati sismici da scaricare, di effettuare le modifiche al setup della stazione e di verificare la posizione delle masse dei sensori Nanometrics Trillium™ (modelli 120p e 240) con eventuale centratura delle medesime attraverso un collegamento seriale.

Per quanto riguarda lo sviluppo dell'interfaccia grafica ci si è avvalsi delle professionalità del Laboratorio di Grafica e Immagini dell'INGV, cui è stato richiesto uno studio per il nuovo logo GAIA. Il loro lavoro alla fine si è esteso anche alla modifica dell'aspetto generale dell'interfaccia grafica. In parallelo, per gli altri sviluppi, è stato necessario apportare modifiche ai file HTML e agli script CGI componenti il server WEB della GAIA2, scrivere nuovi script CGI e, soprattutto, sviluppare due programmi in linguaggio C per l'elaborazione dei file miniseed (formato di archiviazione dei dati sismici) e del dialogo con i sensori Trillium.

Nel presente rapporto quindi viene descritta in dettaglio la nuova interfaccia WEB della GAIA2 e i software ad essa collegati.

1. Nuova interfaccia WEB

Per la nuova interfaccia grafica della stazione GAIA2 si è fatto riferimento al modello di alcuni router commerciali per uso SOHO (Small Office/Home Office): attraverso la programmazione a frame è stata creata una nuova pagina WEB (index.html) principale divisa in tre parti (fig. 1) con, in alto, l'header composto dal logo, il nome dell'Istituto e del Laboratorio di Sismologia Sperimentale, a sinistra il menù di navigazione e al centro lo spazio per la visualizzazione delle informazioni.

Figura 1. Pagina di avvio.

I vari link del menù richiamano quattro script CGI scritti o modificati in modo parametrico e due pagine HTML statiche: index_old.html, Seisgram.html, Webmon.cgi, Download.cgi e, in cascata, Cut.cgi, Setup.cgi (listati script in Appendice A).

Prima di analizzare il resto dell'interfaccia è bene comprenderne la struttura. In figura 2 è riportato il diagramma a blocchi del funzionamento dell'interfaccia WEB della GAIA2. Come ampiamente trattato in Rao e Salvaterra [2011; capitolo 1] il programma che si occupa di gestire le richieste WEB verso la GAIA2 è *mini_httpd* che è in grado di gestire l'esecuzione di script CGI, dare un livello di protezione e autenticazione di base e fornire l'utilizzo dei metodi GET, HEAD e POST. *Mini_httpd* come prima cosa fornisce la pagina di avvio statica chiamata index.html, da essa tramite i link del menù richiama i vari script CGI menzionati sopra che, interagendo con programmi esterni, generano le relative pagine HTML con i risultati. Gli script sono indicati nello schema dal colore turchese, le pagine HTML statiche o generate dinamicamente in arancione, i server esterni in verde, i programmi in rosa e le risorse hardware interessate in lilla.

I link che "partono" da index.html sono così costruiti:

- **Home**: richiama la pagina index_old.html (caricata all'avvio)
- **Status**: richiama webmon.cgi con il parametro *status*
- **Grafico Tensione – GPS**: richiama webmon.cgi con il parametro *grafico*
- **Coordinate e Mappa**: richiama webmon.cgi con il parametro *mappa*
- **Dati Archiviati**: richiama download.cgi. Nella pagina HTML generata c'è la funzionalità di taglio file chiamando lo script cut.cgi che interagisce col programma *extr_seed* che legge e scrive sulla compact flash di cui è dotata la GAIA2
- **Setup TN e AGDF**: richiama setup.cgi, con il parametro *TN* e *AGDF* rispettivamente, che interagisce con i programmi *snmpget* e *snmpset* che dialogano con il server SNMP
- **Segnali In Tempo Reale**: richiama la pagina seisgram.html che tramite l'applet Seisgram2K visualizza i segnali sismici collegandosi al server SeisComP della GAIA2
- **Comandi**: richiama webmon.cgi con il parametro *comandi*
- **SOH Trillium**: richiama webmon.cgi con il parametro *trillium* che interagisce con il programma *trillium* per il dialogo seriale con i sensori omonimi
- **Cambio Password**: richiama webmon.cgi con il parametro *passwd*

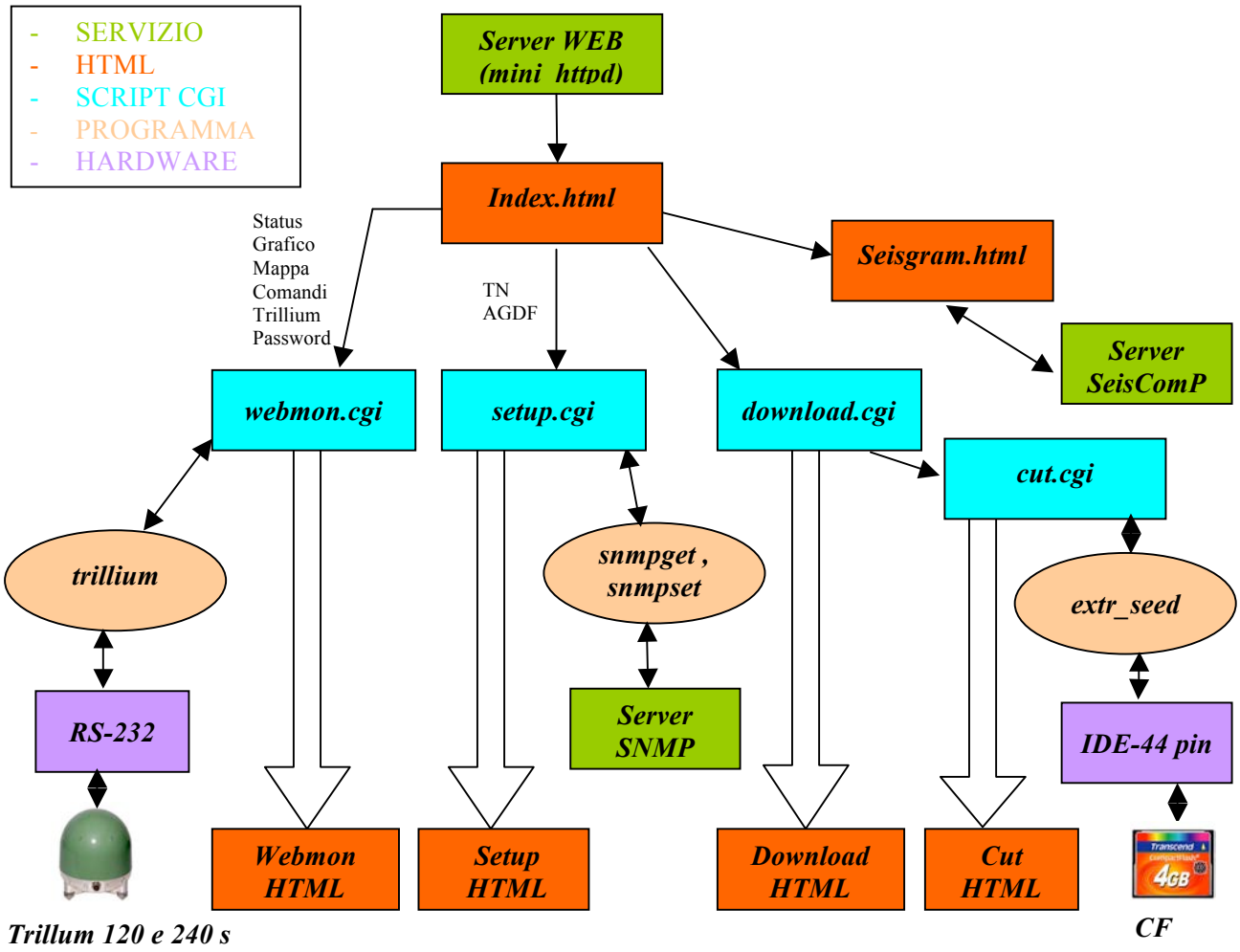


Figura 2. Diagramma a blocchi del server WEB.

Di seguito vengono presentate in dettaglio le varie pagine HTML generate dagli script.

Cliccando sul link "Status" del menù, dopo la richiesta d'inserimento delle credenziali di accesso, vengono visualizzate (fig. 3) le informazioni più importanti relative allo stato di funzionamento della stazione: ora di sistema (ora GMT aggiornata con i dati GPS una volta al giorno), satelliti visibili e agganciati, livello della tensione della batteria, numero progressivo del pacchetto dati sismici, informazioni sulla compact flash in dotazione (dimensioni, occupazione e disponibilità) ed infine la versione del firmware della TN2 che, con le ultime modifiche, è arrivata alla 2.9.2.

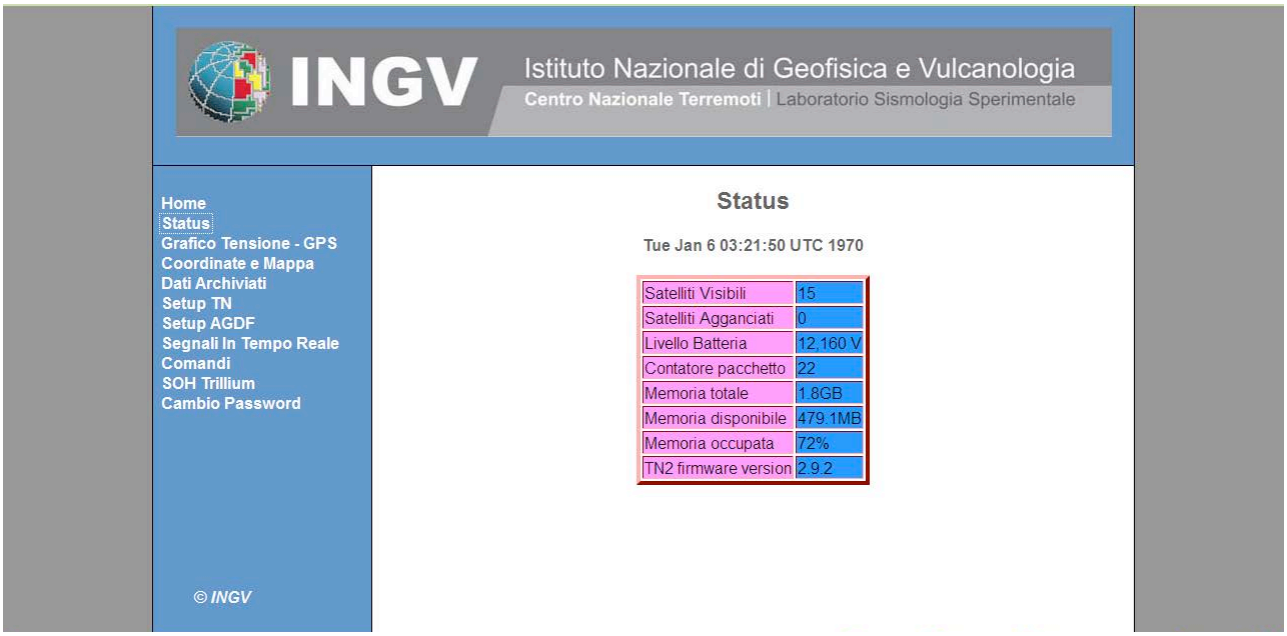


Figura 3. Pagina di status.

Dal link "Grafico Tensione – GPS" si ottiene (fig. 4) l'informazione sulle ultime ventiquattro ore (un valore ogni 15 minuti) sulla tensione di alimentazione (in verde) e del numero di satelliti agganciati (in rosso).

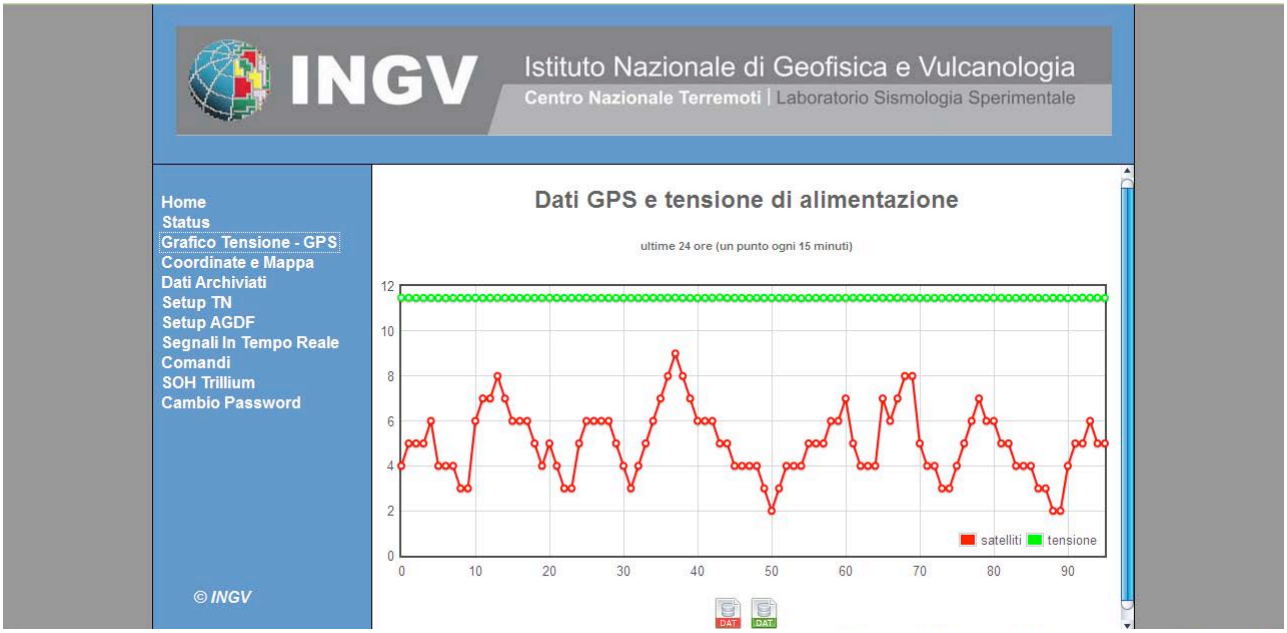


Figura 4. Grafico tensione – GPS.

Sotto il grafico ci sono due ulteriori link che permettono di scaricare i valori di un'intera settimana.

Dal link "Coordinate e Mappa" si ottiene quanto in fig. 5 ossia le ultime coordinate valide (calcolate con un minimo di quattro satelliti agganciati dal ricevitore GPS) e la relativa mappa di Google Maps ™ (chiaramente solo con collegamento internet attivo).

Figura 5. Coordinate e mappa.

Con il link "Dati Archiviati" vengono visualizzati i file dei dati sismici a pieno campionamento, archiviati nella compact flash (fig. 6). L'elenco è di tipo cronologico con la distinzione tra canali velocimetrici (in colore verde) e accelerometrici (in colore rosso).

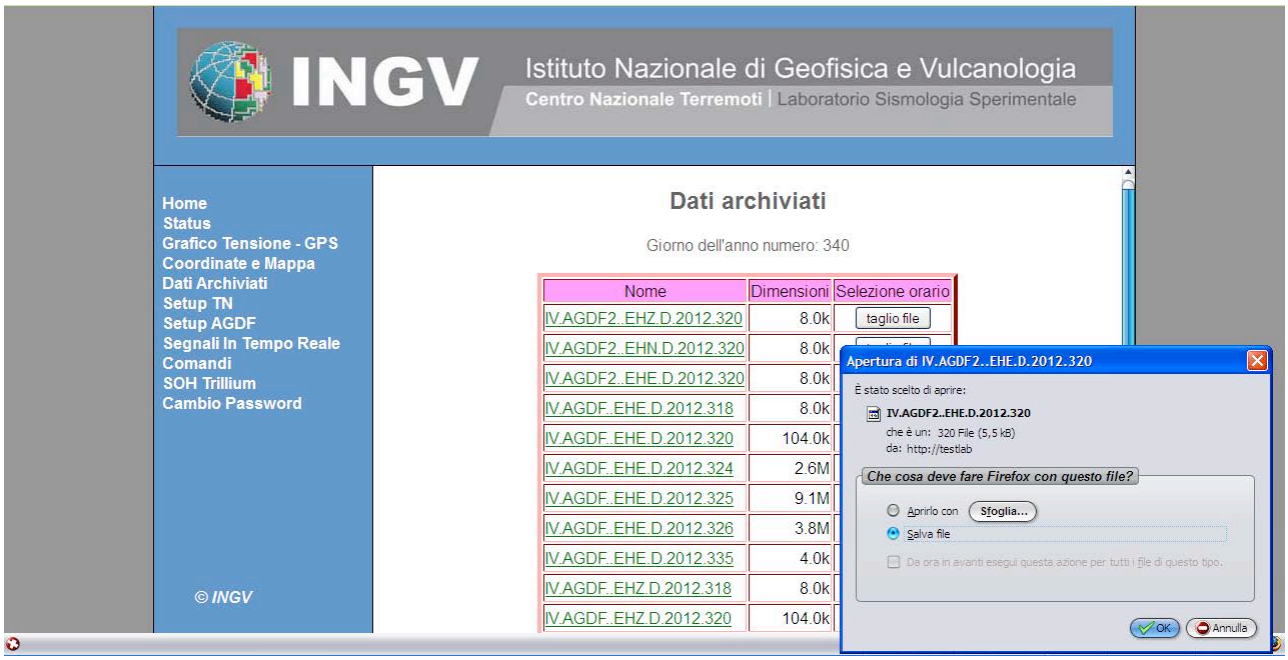


Figura 6. Pagina di download dati archiviati.

Cliccando sul nome di ogni file si può scaricare il file giornaliero, altrimenti cliccando sul pulsante "taglio file" si apre una nuova pagina (fig. 7), creata dallo script cut.cgi, che permette, dopo aver scelto ora e minuti d'inizio e di fine taglio e premuto il pulsante Start, il download del file tagliato.

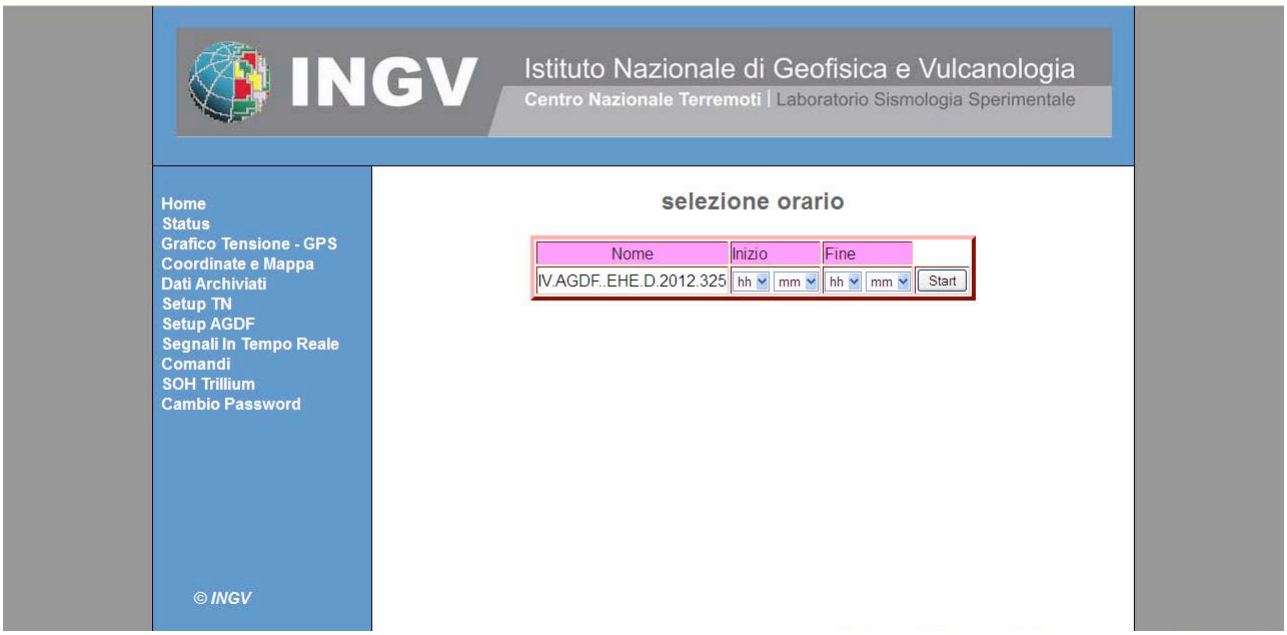


Figura 7. Interfaccia di "taglio" file.

Per far ciò lo script lancia il programma, sviluppato appositamente, *extr_seed*, la cui sintassi è:

extr_seed file_miniseed hhi mmi hhf mmf

dove **file_miniseed** è chiaramente il nome del file da tagliare, **hhi** e **mmi** sono rispettivamente l'ora e i minuti di inizio taglio e **hhf** e **mmf** gli equivalenti di fine taglio. Nel prossimo capitolo sarà analizzato con più dettagli.

Continuando con l'analisi del menù s'incontra la prima grossa novità introdotta, che va a supplire ad una esigenza molto sentita, che è la possibilità di modificare il setup della stazione via WEB rendendo la GAIA2 indipendente dal sistema operativo del computer da cui ci si collega. A tal proposito si ricorda che la stazione sismica GAIA2 è composta principalmente da due schede elettroniche in dialogo fra loro tramite collegamento seriale: quella a più alto livello, TN-2, dotata di sistema operativo Linux e interfaccia ethernet dove risiede il server WEB e l'acquisitore sismico a 24 bit chiamato AGDF2. Gli strumenti per attuare ciò sono dati dal nuovo script setup.cgi e dai programmi *snmpget* e *snmpset*, contenuti nel pacchetto per Linux *net-snmp*, che consentono di dialogare con l'agent SNMP della GAIA2 deputato alla modifica dei settaggi della stazione stessa [Rao et al., 2010].

Col link "SETUP TN" viene richiamato, con il parametro *TN*, lo script setup.cgi che interroga, tramite *snmpget*, l'agent SNMP per i valori riguardanti la configurazione di rete della scheda TN2 e il tipo di supporto per la memorizzazione dei dati. La visualizzazione (fig. 8) è con una tabella di due colonne, delle quali la prima è composta dai nomi dei parametri e la seconda dai relativi valori. Per effettuare le modifiche si cambiano i valori e si clicca sul pulsante "Applica TN". Se vengono variati gli indirizzi IP chiaramente l'interfaccia non risponderà più e sarà necessario perciò aggiornare l'URL a cui ci si collega.

Figura 8. Pagina di setup della TN2.

Per il setup della scheda AGDF2 la pagina generata da setup.cgi (parametro *AGDF*) è più complessa (fig. 9) e cambia leggermente a seconda della presenza o meno della scheda di espansione attiva (figg. 10 e 11).

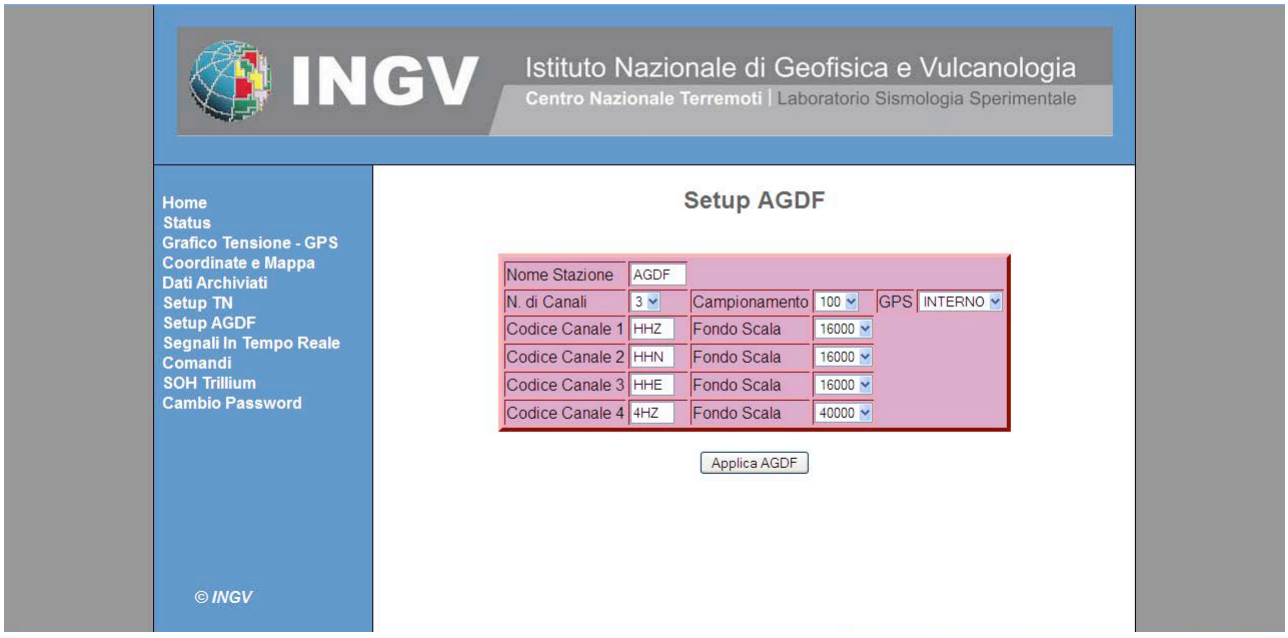


Figura 9. Pagina di setup dell'AGDF2 senza espansione.

Sono presenti tutti i parametri per la conversione A/D dei segnali sismici: nome stazione, numero e codice dei canali, frequenza di campionamento, fondo scala e tipologia del modulo GPS. Anche in questo caso per effettuare modifiche si cambiano i valori dei vari parametri e si applicano le modifiche agendo sul pulsante "Applica AGDF". Un messaggio pop-up comunicherà che la variazione avverrà in circa trenta secondi. Infatti la modifica richiede lo stop dell'applicativo che gira sulla TN2, la sincronizzazione del nuovo setup con l'AGDF2 e il riavvio dell'applicativo.

Con la presenza della scheda AGDFEXT (espansione) che rende la GAIA2 da 4 a 8 canali, compare, nella prima riga della tabella, una casella di testo con l'indicazione di quale scheda è selezionata e un altro pulsante che consente di passare dai parametri dell'AGDFBASE a quelli dell'espansione (ovale rosso nelle figure 10 e 11).

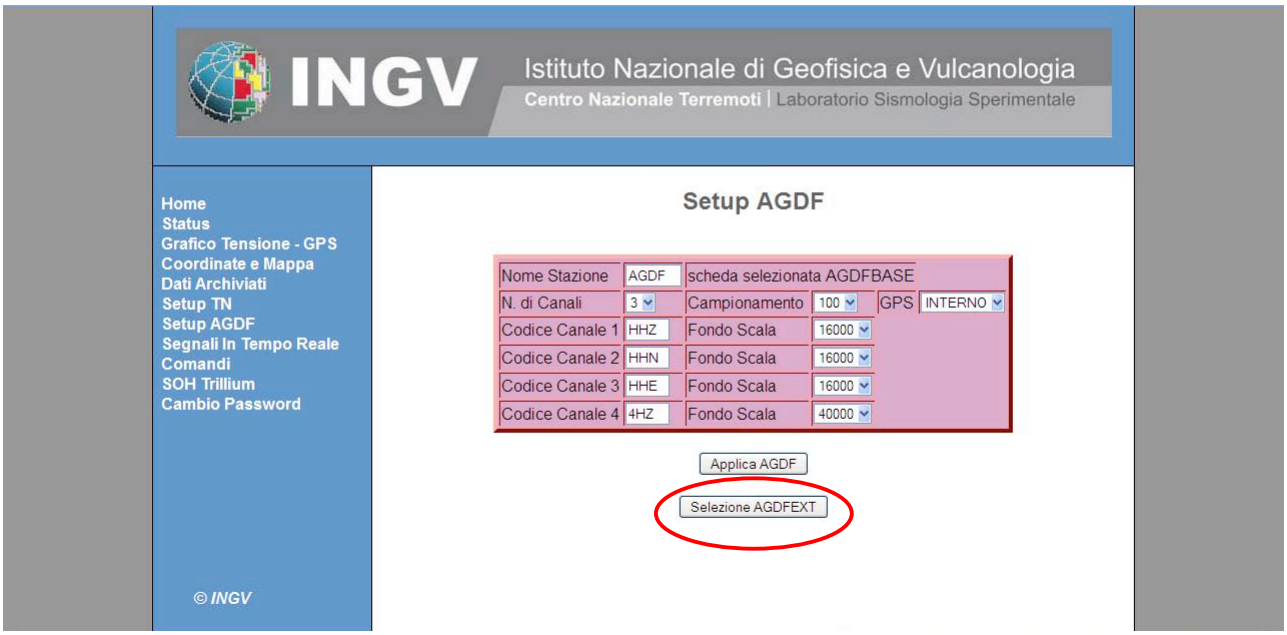


Figura 10. Pagina di setup dell'AGDF2BASE con espansione.

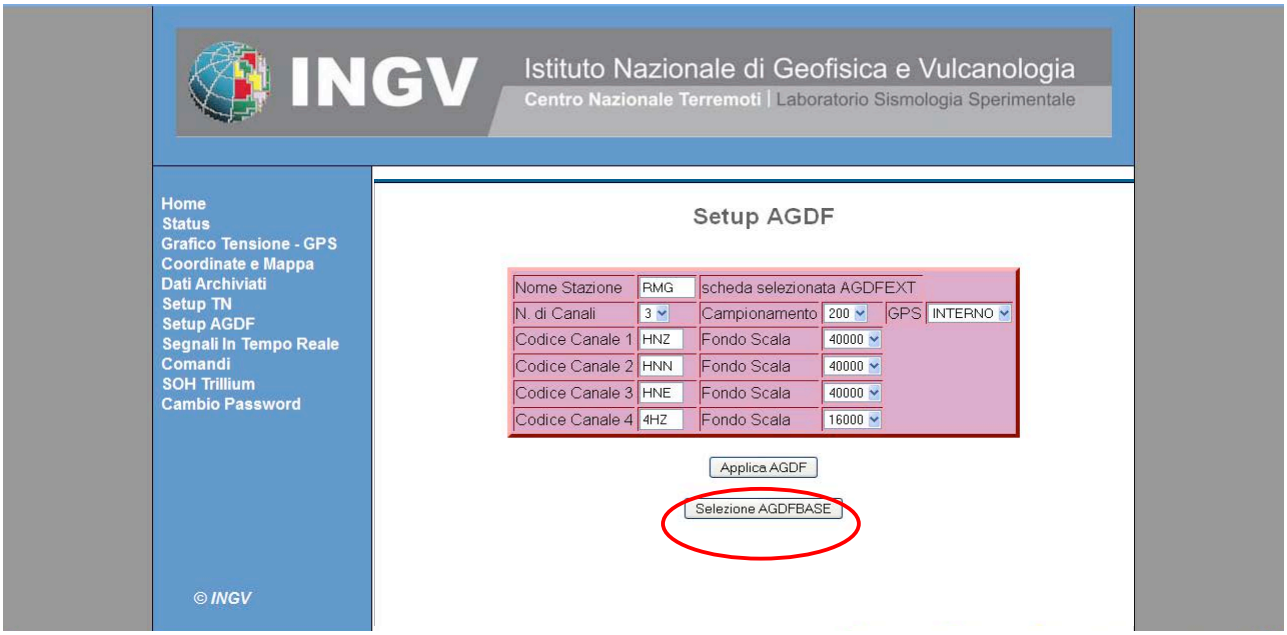


Figura 11. Pagina di setup dell'AGDFEXT (espansione attiva).

Come detto l'interrogazione dell'agent SNMP viene fatta attraverso il programma *snmpget* la cui sintassi è la seguente:

snmpget [options...] <hostname> {<community>} OID [OID]...

Le opzioni utilizzate sono **-v2c** che specifica la versione 2 dell'SNMP, **-O v** (opzione di output) che fa restituire solo il valore della variabile richiesta, **-c tco-comm** che specifica il nome della community SNMP della GAIA2, mentre **OID** (Object IDentifier) è l'identificativo della variabile.
Ad esempio per ricavare l'indirizzo IP della GAIA il comando è:

```
snmpget -v2c -O v -c tco-comm localhost .1.3.6.1.4.1.21949.2.3.1.1.4.1 | sed s/'IpAddress: '//g
```

(il commando linux *sed* viene utilizzato per eliminare la scritta "IpAddress:" restituita dall'agent SNMP)
Per effettuare le modifiche invece, viene utilizzato *snmpset* che ha la seguente sintassi:

```
snmpset [options...] <hostname> {<community>} OID TYPE VALUE [OID TYPE VALUE]...
```

è, come si può notare, molto simile a *snmpget*, ci sono in più i parametri **TYPE** e **VALUE** che, chiaramente, rappresentano il tipo (possibili valori sono i: INTEGER, u: unsigned INTEGER, t: TIMETICKS, a: IPADDRESS, o: OBJID, s: STRING, x: HEX STRING, d: DECIMAL STRING, b: BITS, U: unsigned int64, I: signed int64, F: float, D: double) e il valore da assegnare alla variabile individuata da OID. Ad esempio per assegnare un nuovo indirizzo IP il comando è:

```
snmpset -v2c -O v -c tco-comm localhost .1.3.6.1.4.1.21949.2.3.1.1.4.1 a nuovo_IpAddress
```

Proseguendo con l'analisi del menù, in figura 12 è visualizzato il risultato del click sul link "Segnali In Tempo Reale": viene richiamata la pagina Seigram.html che ha i parametri per l'esecuzione dell'applet java SesiGram2K [Lomax, 2011] che si collega al server SeisComp della GAIA2.

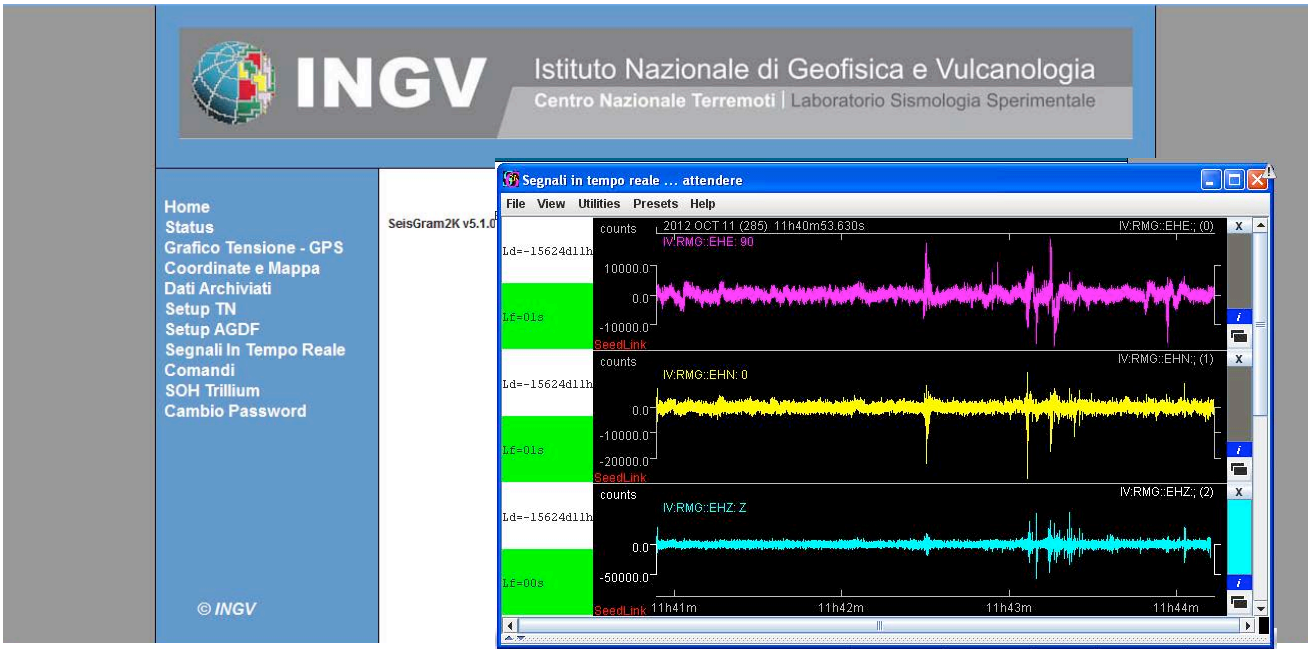


Figura 12. Segnali in tempo reale.

Per quanto riguarda il link "Comandi" il risultato è riportato in fig. 13: c'è la possibilità di visualizzare i processi attivi, di analizzare, visualizzandoli in formato ASCII, i dati in arrivo sulla seriale di comunicazione con l'AGDF2 e di riavviare il sistema operativo della TN2. Per eseguire i comandi viene lanciato lo script command.cgi con parametri *ps*, *serial* o *reboot* rispettivamente.

Figura 13. Comandi per la TN2.

Per il successivo link (SOH TRILLIUM) è stato modificato, come detto, lo script webmon.cgi, ma soprattutto è stato scritto un altro programma, in linguaggio C, per la comunicazione con la seriale dei sensori Trillium: *trillium*. La sintassi è:

```
trillium soh | center
```

dove **soh** comanda a *trillium* di leggere le informazioni dal sensore e **center** di operare la centratura. Questo modulo software verrà analizzato più in dettaglio nel prossimo paragrafo. L'interfaccia WEB, visualizzata in fig. 14, è composta da una tabella con i valori di posizione delle masse del sensore (State Of Health), una leggenda relativa al significato dei colori, la configurazione del periodo e dei segnali e il pulsante per effettuare, in caso di valori critici, la centratura delle masse (in questo caso viene richiamato lo script command.cgi con il parametro *center*). La pagina è dotata di autorefresh ogni venti secondi.

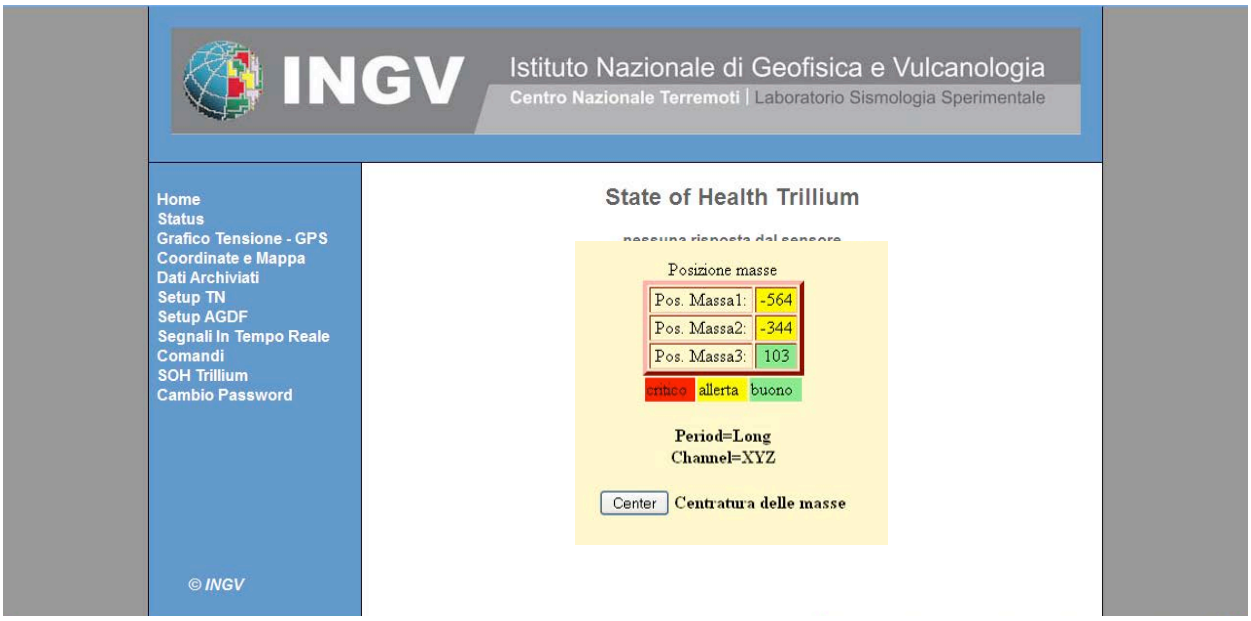


Figura 14. Interfaccia per il SOH dei sensori Trillium.

Infine l'ultimo link, Cambio Password, consente l'ovvia modifica della password di accesso (fig. 15).

Figura 15. Form per il cambio della password.

Il contenuto della directory cgi-bin con gli script è il seguente:
js: directory con javascript per la generazione del grafico;

command.cgi: script CGI per l'esecuzione dei comandi dei tre pulsanti;

download.cgi: script CGI di gestione del download dei file archiviati;

cut.cgi: script CGI per il taglio dei file giornalieri;

passwd.cgi: script CGI per la modifica della password;

sat.sh: script per la memorizzazione, ogni 15 minuti, del numero di satelliti (nel file di testo sat) e dei valori della tensione di alimentazione (nel file power) utilizzati per il grafico;

webmon.cgi: script CGI principale;

extr_seed: programma per il taglio dei file miniseed;

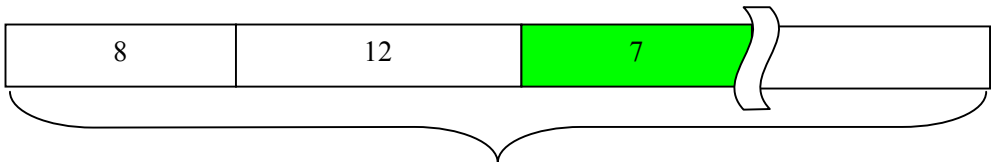
setup.cgi: script CGI per controllare ed effettuare i setup della stazione;

trillium: programma per il dialogo con i sensori Trillium.

2. I programmi extr_seed e trillium

Come anticipato nel precedente capitolo l'aggiunta delle funzionalità ha richiesto lo sviluppo di due programmi in linguaggio C [Schildt, 1998]: *extr_seed* per il taglio dei file miniseed archiviati nella GAIA2 e *trillium* per comunicare con i sensori Trillium.

Per il primo (vedi listato commentato in appendice B) lo sviluppo è partito dall'analisi dei file miniseed archiviati ed in particolare sulla posizione delle informazioni relative all'orario all'interno del file stesso. Il file è composto dalla concatenazione di singoli pacchetti SEED (Standard for the Exchange of Earthquake Data) composti di 512 byte suddivisi in un header di informazioni (48 byte) e il resto di dati [IRIS, 2004]. In figura 16 c'è una descrizione dei primi 30 byte dell'header per i quali l'attenzione va posta sulla posizione dell'informazione della data di inizio registrazione, 7 byte a partire dal ventunesimo.



Descrizione byte	Numero di byte
Sequenza numerica del pacchetto	6
Data Header o indicatore di qualità (“D”|“R”|“Q”)	1
Byte riservato (spazio)	1
Codice di stazione	5
Codice di location	2
Codice di canale	3
Codice di rete	2
Data di inizio	7

Figura 16. Primi 15 byte dell’header dei pacchetti SEED.

Il formato della data è AAAAGGGGHHMMSS dove AAAA sono due byte, in esadecimale, per l’anno, GGGG due byte per il giorno dell’anno (0-365), HH un byte per l’ora, MM un byte per i minuti e SS un byte per i secondi (totale 7 byte). Per lo sviluppo del programma *extr_seed* è stato sufficiente quindi concentrarsi su questi byte, in particolare sulle ore e sui minuti. Infatti, poiché il formato SEED è compresso, nel singolo pacchetto di 512 byte possono entrare più secondi di registrazione, variabilità data sia dal campionamento sia dal tipo di segnale sismico registrato (presenza di eventi o meno ecc.), per cui si è preferito per una semplicità di sviluppo, escludere i secondi dalla ricerca. In definitiva *extr_seed* apre il file miniseed, cerca le ore e i minuti di inizio taglio, passati come parametri, e scrive su un nuovo file i byte letti fino a superare le ore e i minuti di fine taglio, passati sempre come parametri.

Per lo sviluppo del programma *trillium* (codice in appendice C), è stato necessario analizzare l’interfaccia di comunicazione dei sensori Trillium versione 240 e 120p [Nanometrics, 2005]. Hanno a disposizione un terminale a caratteri attraverso una porta seriale a bassa velocità, per mezzo dei pin C (TX) e D (RC) del cavo del sensore, la cui configurazione è 9600 baud di velocità, 8 bit, nessuna parità, 1 bit di stop e controllo di flusso di tipo software. Connettendosi con un computer e un terminale seriale, e premendo i caratteri TX e il tasto <invio>, dopo circa tre secondi appare la scritta Serial Transmit Enabled. Per avere la lista dei comandi disponibili basta scrivere help e <invio> ed il risultato è il seguente:

```
Nanometrics Trillium User Menu (Version 3.30) Program A
*******************************************************************
Help - Repeat this menu (also turns on Serial TX)
Tx - Enable the Serial Transmit Signal
TxOff - Disable the Serial Transmit Signal
Upload - Upload software to the alternate program
Switch - Switch to the alternate program
Default - Set the current program as default
Reboot - Reboot the instrument
GetInfo - Get factory configuration information
ReadFC - Read factory calibration parameters
WriteUC - Write user calibration parameters
ReadUC - Read the user calibration parameters
Soh - Report state-of-health
```

```
ShortPer - Set sensor to short period mode
LongPer - Set sensor to long period mode
SetXYZ - Set sensor to XYZ mode
SetUVW - Set sensor to UVW mode
Center - Center all masses or (u/v/w)
CheckSum - Print checksum value for both program A and B
*****************************************************************
Please type a command and hit return:
```

Per quanto ci riguarda i comandi necessari al controllo e all'operazione di centratura delle masse sono Soh e Center. Scrivendo Soh più <invio> si ottiene:

```
<SOH>
<Manufacture>"Nanometrics, Inc."</Manufacture>
<Product>"Trillium Firmware"</Product>
<Version>3.33</Version>
<Temperature>24.47</Temperature>
<Mass>U=-1.163 V=1.698 W=0.191</Mass>
<Adc>U=-525 V=768 W=87</Adc>
<Modes>Period=Long Channel=XYZ</Modes>
<Positions>U=649 V=-5214 W=-1264</Positions>
<Level>U=335 V=-39 W=-16</Level>
<Range>U=+/-7077 V=+/-7187 W=+/-7647</Range>
</SOH>
```

I valori di nostro interesse sono evidenziati in verde e ci danno una rappresentazione numerica dello stato delle masse dei tre canali in un intervallo di ±1900. Da notare che questo valore è circa 452 volte il valore in Volt dato dalla precedente riga Mass.

Mandando il comando "center" si fa partire la centratura delle masse con il seguente output di esempio:

```
move=0, steps=0, per=3, pos=0, ADC=-551
move=1, steps=84, per=3, total steps=84, ADC=-2
MC=CENTERED, Ch=U, Pos=733, Mass=-0.006V, ADC=-2
move=0, steps=0, per=3, pos=0, ADC=807
move=1, steps=-90, per=3, total steps=-90, ADC=-3
MC=CENTERED, Ch=V, Pos=-5304, Mass=-0.008V, ADC=-3
move=0, steps=0, per=3, pos=0, ADC=90
move=1, steps=-6, per=3, total steps=-6, ADC=-14
MC=CENTERED, Ch=W, Pos=-1270, Mass=-0.032V, ADC=-14
```

che dà un'indicazione dei movimenti dei motori di centraggio. I valori di nostro interesse sono quelli dopo "ADC=".

La nostra esigenza è stata quindi di avere un programma che aprisse la porta seriale della TN2, mandasse in sequenza al sensore i comandi TX e, in base a un parametro, **center** o **soh** e che restituisse in uscita i valori ADC filtrando opportunamente l'output del sensore.

3. Conclusioni

Con le migliorie introdotte dal presente lavoro si è voluto attribuire al server WEB della stazione GAIA un ruolo centrale di massima importanza ed efficacia per la gestione ed il controllo dell'acquisitore sismico che precedentemente l'utente poteva svolgere solo con il pacchetto software Earthlab, sviluppato anch'esso nei nostri laboratori in linguaggio Viual Basic™ e funzionante esclusivamente su piattaforma Windows. Gli utilizzatori di altri sistemi operativi (come ad esempio Linux), chiaramente risultavano penalizzati nello svolgere le operazioni di setup della stazione, potendo sfruttare solo connessioni di tipo telnet o SSH per il collegamento all'hardware. Webmon invece basandosi su protocolli di tipo internet è sostanzialmente indipendente dal sistema operativo dal quale viene invocato. Inoltre alcune funzioni, come ad esempio il centraggio delle masse del sensore erano possibili solo in locale sfruttando un collegamento seriale mentre adesso tale procedura si può eseguire anche da remoto laddove necessario, evitando di recarsi in stazione con conseguente risparmio di tempo e denaro.

Bibliografia

Schildt, H. (1998). La guida completa C++. MC Graw Hill;

IRIS, (2004). Seed Manual v. 2.4,

Lomax, A., (2011). SeisGram2K Seismogram Viewer. http://alomax.free.fr/seisgram/SeisGram2K.html

Nanometrics Inc., (2005). Trillium 240 Seismometer User Guide

Rao, S., Salvaterra, L. and Acerra, C., (2010). *Software per l'installazione e la configurazione della stazione sismica GAIA2*. Rapporti Tecnici INGV, 130.

Rao, S. and Salvaterra, L., (2011). *WebMon: piccola interfaccia web della stazione sismica digitale GAIA2*. Rapporti Tecnici INGV, 212.

Appendice A: listato degli script webmon.cgi, command.cgi, passwd.cgi, download.cgi, cut.cgi e setup.cgi

Webmon.cgi: in neretto sono evidenziati i test del parametro passato dal menu

```
#!/bin/sh
echo Content-type: text/html
echo "<HTML><HEAD><TITLE>WebMon</TITLE></HEAD>"
echo "<font face=\"arial\" color=\"#666666\">"
echo "<BODY bgcolor=\"white\">" #<H1><center>Benvenuto in
WebMon</center>"
echo   "<script   language=\"javascript\"   type=\"text/javascript\"
src=\"js/jquery_002.js\"></script>"
echo   "<script   language=\"javascript\"   type=\"text/javascript\"
src=\"js/jquery.js\"></script>"
. /var/tn-2/1
if [ $1 = "status" ]
then
echo "<H2><center>Status</center></H2><CENTER><b>"
 date
echo "<br><br></center>"
 echo "<CENTER><table border='4' BORDERCOLOR=RED
BORDERCOLORLIGHT=ORANGE BORDERCOLORDARK=RED >"
echo "<tr><td bgcolor=#FF99FF>Satelliti Visibili</td><td
bgcolor=#3399FF>"
echo $visibleSatellites
echo "</td></tr>"
echo "<tr><td bgcolor=#FF99FF>Satelliti Agganciati</td><td
bgcolor=#3399FF>"
echo $trackedSatellites
echo "</td></tr>"
echo "<tr><td bgcolor=#FF99FF>Livello Batteria</td><td
bgcolor=#3399FF>"
echo $(($powerSupply/1000)),$(($powerSupply%1000))
echo " V </td></tr>"
echo "<tr><td bgcolor=#FF99FF>Contatore pacchetto</td><td
bgcolor=#3399FF>"
echo $packetsCounter
echo "</td></tr>"
echo "<tr><td bgcolor=#FF99FF>Memoria totale</td><td
bgcolor=#3399FF>"
echo $(df -h | grep hda2 | awk 'BEGIN{FS=" "}{print $2}')B
echo "</td></tr>"
echo "<tr><td bgcolor=#FF99FF>Memoria disponibile</td><td
bgcolor=#3399FF>"
echo $(df -h | grep hda2 | awk 'BEGIN{FS=" "}{print $4}')B
echo "</td></tr>"
echo "<tr><td bgcolor=#FF99FF>Memoria occupata</td><td
bgcolor=#3399FF>"
echo $(df -h | grep hda2 | awk 'BEGIN{FS=" "}{print $5}')
echo "</td></tr>"
```

```
    echo "<td bgcolor=#FF99FF>TN2 firmware version</td><td
bgcolor=#3399FF>"
    echo $(cat /etc/VERSION)
    echo "</td></tr>"
    echo "</table></CENTER><br>"
    elif [ $1 = "grafico" ]
    then
    echo "<H2><center>Dati GPS e tensione di alimentazione </H2>
</center><H6> <center >ultime 24 ore (un punto ogni 15
minuti)</H6></center>"
    echo "<div id=\"placeholder\" style=\"width: 100%; height: 300px;
position: relative; \"><div class=\"tickLabels\" style=\"font-size:
smaller; color: rgb(84, 84, 84);\"></div></div>"
    echo "<script id=\"source\" language=\"javascript\"
type=\"text/javascript\">"
    echo "\$(function () {"
    echo "for (var i = 0; i <100; i += 0.5)"
    i=0
    echo "var sat = [];"
    for riga in $(cat ./sat)
    do
       echo "sat.push([ $i, $riga]) "
       i=$(( $i + 1 ))
    done
    i=0
    echo "var power = [];"
    for riga in $(cat ./power)
    do
       echo "power.push([ $i, $riga/1000 ]) "
       i=$(( $i + 1 ))
    done
    echo "var options = {"
    echo "lines: { show: true }, legend: {position: \"se\" ,
noColumns:2, backgroundOpacity: 0},"
    echo "points: { show: true }};"
    echo "$.plot(\$(\"#placeholder\"), [{ data: sat , label: \"satelliti
\"},{ data: power , label: \"tensione \"}], options );"
    echo "});"
    echo "</script>"
    elif [ $1 = "mappa" ]
    then
    echo "<H2><center>Coordinate stazione e mappa</center></H2>"
    altitude=$(($altitude / 100))
    latitude=$(echo "scale= 9; $latitude / 3600000" | bc)
    longitude=$(echo "scale=9; $longitude / 3600000" | bc)
    echo "<center> <h3> <b>Lat: $latitude - Lon: $longitude - Alt:
$altitude</center>"
    echo "<CENTER> <iframe width=\"480\" height=\"400\"
frameborder=\"0\" scrolling=\"no\" marginheight=\"0\" marginwidth=\"0\"
src=\"http://maps.google.it/maps?f=q&amp;source=s_q&amp;hl=it&amp;geocod
e=&amp;q=$latitude+$longitude&amp;sll=$latitude,$longitude&amp;sspn=0.00
```

```
9729,0.021951&amp;ie=UTF8&amp;ll=$latitude,$longitude&amp;spn=0.009729,0
.021951&amp;z=16&amp;output=embed\"></iframe><br />"
    elif [ $1 = "comandi" ]
    then
    echo "<CENTER><H2>Invio comandi a GAIA</H2>"
    echo "<table>"
    echo    "<tr><td><input    type=button    value='Processi    attivi'
onclick=location.href='command.cgi?ps'></td><td>  Elenco  dei  processi
attivi</td>"
    echo   "<tr><td><input   type=button   value='Dati   da   Seriale'
onclick=location.href='command.cgi?serial'></td><td>Visualizzazione dati
dalla seriale con l'AGDF</td>"
    echo        "<tr><td><input        type=button        value=Reboot
onclick=location.href='command.cgi?reboot'></td><td> Riavvio del sistema
operativo</td>"
    echo "</table></CENTER><br>"
    elif [ $1 = "passwd" ]
    then
    echo "<CENTER><H2>Cambio password</H2>"
    echo "<form action='passwd.cgi'><br>"
    echo    "<div><center>Nuova    password:    <input    type='password'
name='password' size='10'></div></center><br>"
    echo "<div><center>Riscrivere nuova password: <input type='password'
name='new_password' size='10'></div></center><br>"
    echo "<center><input type=submit Value='Cambio Password'></center>"
    echo "</form>"
    elif [ $1 = "trillium" ]
    then
    echo "<CENTER><H2>State of Health Trillium</H2>"
    mass=$(./trillium soh | egrep 'Adc|Modes')
    if [ "$mass" = "" ]
    then
         echo "<center><b>nessuna risposta dal sensore<center>"
    else
    massM1=$(echo "$mass" | grep Adc | awk 'BEGIN{FS="="}{print $2}' |
awk 'BEGIN{FS=" "}{print $1}')
    massM2=$(echo "$mass" | grep Adc | awk 'BEGIN{FS="="}{print $3}' |
awk 'BEGIN{FS=" "}{print $1}')
    massM3=$(echo "$mass" | grep Adc | awk 'BEGIN{FS="="}{print $4}' |
awk 'BEGIN{FS="<"}{print $1}')
    period=$(echo "$mass" | grep Modes | sed -e 's/ /<br>/g') #| awk
'BEGIN{FS="="}{print $1}')
    if [ "$massM1" -lt -1356 ] || [ "$massM1" -gt 1356 ]
    then
            colorM1="red"
    elif [ "$massM1" -gt -1356 ] && [ "$massM1" -lt -226 ]
    then
            colorM1="yellow"
    elif [ "$massM1" -gt 226 ] && [ "$massM1" -lt 1356 ]
    then
            colorM1="yellow"
```

```
    elif [ "$massM1" -gt -226 ] && [ "$massM1" -lt 226 ]
    then
            colorM1="lightgreen";
    fi
    if [ "$massM2" -lt -1356 ] || [ "$massM2" -gt 1356 ]
    then
            colorM2="red"
    elif [ "$massM2" -gt -1356 ] && [ "$massM2" -lt -226 ]
    then
            colorM2="yellow"
    elif [ "$massM2" -gt 226 ] && [ "$massM2" -lt 1356 ]
    then
            colorM2="yellow"
    elif [ "$massM2" -gt -226 ] && [ "$massM2" -lt 226 ]
    then
            colorM2="lightgreen";
    fi
    if [ "$massM3" -lt -1356 ] || [ "$massM3" -gt 1356 ]
    then
            colorM3="red"
    elif [ "$massM3" -gt -1356 ] && [ "$massM3" -lt -226 ]
    then
            colorM3="yellow"
    elif [ "$massM3" -gt -226 ] && [ "$massM3" -lt 226 ]
    then
            colorM3="lightgreen";
    fi
    echo "Posizione masse<table border='4' BORDERCOLOR=RED
BORDERCOLORLIGHT=ORANGE BORDERCOLORDARK=RED width='145'>"
    echo "<tr><td align='center'>Pos. Massa1:</td><td bgcolor='$colorM1'
align='center'>$massM1</td></tr>"
    echo "<tr><td align='center'>Pos. Massa2:</td><td bgcolor='$colorM2'
align='center'>$massM2</td></tr>"
    echo "<tr><td align='center'>Pos. Massa3:</td><td bgcolor='$colorM3'
align='center'>$massM3</td></tr>"
    echo "</table></CENTER>"
    echo "<center><table BORDERCOLOR=RED BORDERCOLORLIGHT=ORANGE
BORDERCOLORDARK=RED width='145' >"
    echo "<tr><td bgcolor='red'>critico</td><td
bgcolor='yellow'>allerta</td><td bgcolor='lightgreen'>buono</td></tr>"
    echo "</table></CENTER><br>"
    echo "<b><center>$period<br><br>"
    echo "<input type=button value='Center' onclick=\"alert('Attendere,
operazione che può richiedere parecchi secondi');
location.href='command.cgi?center'\"> Centratura delle
masse<br><br><br><br>"
    fi
    echo "<meta http-equiv='refresh' content='20
url=webmon.cgi?trillium'>"
    fi
```

```
echo "</CENTER>"
echo "</BODY></HTML>"
```

Command.cgi

A seconda del parametro passato da webmon.cgi lo script esegue comandi diversi

```
#!/bin/sh
 echo Content-type: text/html
 echo "<HTML>"
 echo "<HEAD><TITLE></TITLE></HEAD>"
 echo "<font face=\"arial\" color=\"#666666\">"
 echo "<BODY bgcolor=\"white\">"
- reboot
if [ $1 = reboot ]
then
       echo " <center>Riavvio del sistema in corso</center>"
        /sbin/reboot
- lettura dati  dalla seriale
elif [ $1 = serial ]
then
 killall -2 master2bn-seed-com1 2> /dev/null
 rm serial.txt
 sleep 1
 cat /dev/ttyS1 > serial.txt &
echo " <H1><center>Dati da seriale</H1></center>"
 pid=$(ps | grep "cat /dev/ttyS1" | awk 'BEGIN{FS=" "}{print $1}')
sleep 5
 kill -9 $pid 2> /dev/null
 echo "<CENTER>"
 echo " <table border='2' width='80%'>"
 echo "<tr><td>"
 echo  $(cat serial.txt)
 echo "</td></tr></table>"
 echo "<B><blink>Attenzione applicativo fermo: si riavvierà entro un
minuto</blink>"
 echo "</CENTER><HR>"
- elenco dei processi attivi
elif [ $1 = ps ]
then
        echo "<PRE>"
        ps | grep "S " | awk 'BEGIN{FS=" "}{print $1}' > ps.txt
        for line in $(cat ps.txt)
        do
                echo $(cat /proc/$line/cmdline 2> /dev/null)
        done
        echo "</PRE><HR>"
- comando di centratura masse dei sensori Trillium
elif [ $1 = center ]
then
        echo "<CENTER><br>centratura masse in corso, attendere il
ricaricamento della pagina</CENTER>"
        echo      "<meta      http-equiv='refresh'      content='10
url=webmon.cgi?trillium'>"
        ./trillium center 2>&1 > /dev/null
fi
echo "</CENTER></BODY></HTML>"
```

Passwd.cgi

```
#!/bin/sh
echo Content-type: text/html
echo "<HTML>"
echo "<font face=\"arial\" color=\"#666666\">"
echo "<HEAD><TITLE>WebMon - New Password</TITLE></HEAD>"
echo "<BODY bgcolor=\"white\"><H1><center>New
Password</H1></center>"
password=$(echo "$QUERY_STRING" | awk 'BEGIN{FS="&"}{print $1}' |
awk 'BEGIN{FS="="}{print $2}')
new_password=$(echo "$QUERY_STRING" | awk 'BEGIN{FS="&"}{print $2}'
| awk 'BEGIN{FS="="}{print $2}')
if [ $password = $new_password ];
then
  /usr/local/sbin/htpasswd -cb /home/www/cgi-bin/.htpasswd GAIA2
$password 2>&1 > /dev/null &
  echo "<br><br><br><H2><CENTER><blink>PASSWORD
CHANGED</blink></CENTER></H2><br><br>"
else echo "no passwd"
fi
echo "<hr><CENTER><B><I><a href='../index.html'>Home</a> | <a
href='webmon.cgi'>Indietro</a>"
echo "<hr><a href=\"mailto:sandro.rao@ingv.it,
leonardo.salvaterra@ingv.it\"><B><I>By S.Rao & L.Salvaterra </a>"
echo "</CENTER></BODY></HTML>"
```

Download.cgi

```
#!/bin/sh
echo "<HTML><HEAD><TITLE>Download data</TITLE></HEAD>"
echo "<font face=\"arial\" color=\"#666666\">"
echo "<BODY bgcolor=\"white\"><center><H2> Dati archiviati </center>
</H2>"
echo "<center>Giorno dell'anno numero:"
date  +%j
echo "</center><br>"
stat=$(cat /seedlink-ridotto/config/seedlink.ini | grep "description
= \"INGV1" | awk 'BEGIN{FS=" "}{print $2}')
anno=$(date | awk 'BEGIN{FS=" "}{print $6}')
rm -f cut.txt
echo "<center><table border='4' BORDERCOLOR=RED
BORDERCOLORLIGHT=ORANGE BORDERCOLORDARK=RED ></center>"
echo "<tr bgcolor=#FF99FF><td><center>Nome</td>"
echo "<td>Dimensioni</td></center>"
echo "<td>Selezione orario</td></tr></center>"

du -ah data/archive/$anno/IV/$stat/H* 2> /dev/null | grep "IV\." >
H.txt
du -ah data/archive/$anno/IV/$stat/E* 2> /dev/null | grep "IV\." >
E.txt
if [ -s H.txt ]
  then
  while read line
  do
        file_name=$(echo $line | awk 'BEGIN{FS=" "}{print $2}')
        echo "<td> <a href=$(echo $line | awk 'BEGIN{FS=" "}{print
$2}') color=\"#FF0000\" title=\"click per scaricare l'intero file\"
>$(echo $line | awk 'BEGIN{FS="/"}{print $7}')
</a></font></center></td>"
```

```
            echo "<td align='right'> $(echo $line | awk 'BEGIN{FS=" "}{print $1}') </td>"
            echo "<td><center><input type=button value='taglio file' onclick=location.href='cut.cgi?$file_name'></td></tr></center>"
      done < H.txt
    else
        du -ah data/archive/$anno/IV/$stat/B* 2> /dev/null | grep "IV\." >  B.txt
         if [ -s B.txt ]
            then
                while read line
                do
                      file_name=$(echo $line | awk 'BEGIN{FS=" "}{print $2}')
                      echo "<td> <a href=$(echo $line | awk 'BEGIN{FS=" "}{print $2}') color=\"#FF0000\" title=\"click per scaricare l'intero file\"    >$(echo    $line    |    awk    'BEGIN{FS="/"}{print    $7}') </a></font></center></td>"
                      echo "<td align='right'> $(echo $line | awk 'BEGIN{FS=" "}{print $1}') </td>"
                      echo "<td><center><input type=button value='taglio file' onclick=location.href='cut.cgi?$file_name'></td></tr></center>"
               done < B.txt
            fi
    fi
    if [ -s E.txt ]
    then
      while read line
     do
            file_name=$(echo $line | awk 'BEGIN{FS=" "}{print $2}')
            echo "<td> <a href=$(echo $line | awk 'BEGIN{FS=" "}{print $2}') color=\"#FF0000\" title=\"click per scaricare l'intero file\" >$(echo      $line      |      awk      'BEGIN{FS="/"}{print      $7}') </a></font></center></td>"
            echo "<td align='right'> $(echo $line | awk 'BEGIN{FS=" "}{print $1}') </td>"
            echo "<td><center><input type=button value='taglio file' onclick=location.href='cut.cgi?$file_name'></td></tr></center>"
     done < E.txt
    else
            du -ah data/archive/$anno/IV/$stat/S* 2> /dev/null | grep "IV\." >  S.txt
            if [ -s S.txt ]
            then
                while read line
                do
                      file_name=$(echo $line | awk 'BEGIN{FS=" "}{print $2}')
                      echo "<td> <a href=$(echo $line | awk 'BEGIN{FS=" "}{print $2}') color=\"#FF0000\" title=\"click per scaricare l'intero file\"    >$(echo    $line    |    awk    'BEGIN{FS="/"}{print    $7}') </a></font></center></td>"
                      echo "<td align='right'> $(echo $line | awk 'BEGIN{FS=" "}{print $1}') </td>"
                      echo "<td><center><input type=button value='taglio file' onclick=location.href='cut.cgi?$file_name'></td></tr></center>"
               done < S.txt
```

```
            fi
    fi
    echo "</tr></table><br></CENTER></BODY></HTML>"
    Cut.cgi
    #!/bin/sh
    echo "<font face=\"arial\" color=\"#666666\">"
    echo       "<HTML><BODY       bgcolor=\"white\"><H2><center>selezione
orario</H2></center>"
    hhi=$(echo  "$QUERY_STRING"  |  awk  'BEGIN{FS="&"}{print  $1}'  |  awk
'BEGIN{FS="="}{print $2}')
    mmi=$(echo  "$QUERY_STRING"  |  awk  'BEGIN{FS="&"}{print  $2}'  |  awk
'BEGIN{FS="="}{print $2}')
    hhf=$(echo  "$QUERY_STRING"  |  awk  'BEGIN{FS="&"}{print  $3}'  |  awk
'BEGIN{FS="="}{print $2}')
    mmf=$(echo  "$QUERY_STRING"  |  awk  'BEGIN{FS="&"}{print  $4}'  |  awk
'BEGIN{FS="="}{print $2}')
    #file=$(echo  "$QUERY_STRING"  |  awk  'BEGIN{FS="&"}{print  $5}'  |  awk
'BEGIN{FS="="}{print $2}')
    if [ -e cut.txt ]
    then
         file=$(cat cut.txt)
         rm cut.txt
    else
         file='';
    fi
    if [ "$file" != "" ]
    then
         if [ "$hhi" -le "$hhf" ]
         then
             if [ "$mmi" -le "$mmf" ]
             then
                 ./extr_seed $file $hhi $mmi $hhf $mmf
                 file2=$(ls $file | grep _cut)
                 mv $file2 ./file_cut/
                 file2=$(ls ./file_cut/)
                 echo      "<center>      <a      href='./file_cut/$file2'
color=\"#FF0000\"   title='click   per   scaricare   il   file'   >$file2
</a></center>"
             else
                 echo       "<center>errore       nella       selezione
dell'intervallo</center><br>"
                 echo     "<meta     http-equiv='refresh'     content='3
url=cut.cgi?$file'>";
           fi
      else
             echo         "<center>errore         nella         selezione
dell'intervallo</center><br><meta     http-equiv='refresh'     content='3
url=cut.cgi?$file'>";
      fi
      elif [ $1 = "" ]
      then
             echo  "<center>some  error</center><meta  http-equiv='refresh'
content='3 url=download.cgi'>";
      else
    rm -f ./file_cut/* 2>&1 > /dev/null
    echo $1"*" > cut.txt
    echo "<form action='cut.cgi'>"
```

```
    echo        "<center><table        border='4'        BORDERCOLOR=RED
BORDERCOLORLIGHT=ORANGE BORDERCOLORDARK=RED ></center>"
    echo "<tr bgcolor=#FF99FF>"
    echo "<td><center>Nome</td>"
    echo "<td>Inizio</td></center>"
    echo "<td>Fine</td></tr></center>"
    echo "<tr><td>$(echo $1 | awk 'BEGIN{FS="/"}{print $7}') </td>"
    echo "<td><select name='hi'>"
    echo "<option value='0'>hh</option>"
    for i in `seq 0 23`;
    do
                echo "<option value='$i'>$i </option>"
    done
    echo "</select>"
    echo "<select name='mi'>"
    echo "<option value='0'>mm</option>"
    for i in `seq 0 59`;
    do
          echo "<option value='$i'>$i </option>"
    done
    echo "</select></td>"
    echo "<td><select name='hf'>"
    echo "<option value='0'>hh</option>"
    for i in `seq 0 23`;
    do
        echo "<option value='$i'>$i </option>"
    done
    echo "</select>"
    echo "<select name='mf'>"
    echo "<option value='0'>mm</option>"
    for i in `seq 0 59`;
    do
         echo "<option value='$i'>$i </option>"
    done
    echo "</select></td>"
    echo "<td><input type='submit' value='Start'></td>"
    echo "</tr></table></CENTER><br>"
    echo "</form>"
    fi
    echo "</BODY></HTML>"
```

Setup.cgi

```
    #!/bin/sh
    . /etc/tn-2/tn-2-config.staz
    DIR_SNMP="/opt/net-snmp/sbin"
    echo "<HTML><HEAD><TITLE>Setup</TITLE></HEAD>"
    echo "<font face=\"arial\" color=\"#666666\">"
    echo "<BODY bgcolor=\"white\"><H2><center>Setup
$1</center></H2><br>"
    if [ "$CONTENT_LENGTH" -ne "" ]
    then
         query=$( head -c $CONTENT_LENGTH )
         submit=$(echo $query | sed "s/.*=//" | awk 'BEGIN{FS="+"}{print
$2}')
         if [ $submit = "TN" ]
         then
```

```
            hostname=$(echo $query | awk 'BEGIN{FS="&"}{print $1}' |
awk 'BEGIN{FS="="}{print $2}' | ./tr '[:lower:]' '[:upper:]' | sed -e
"s/+/ /g")
            tnIpAddress=$(echo $query | awk 'BEGIN{FS="&"}{print $2}' |
awk 'BEGIN{FS="="}{print $2}' )
            tnNetwork=$(echo $query | awk 'BEGIN{FS="&"}{print $3}' |
awk 'BEGIN{FS="="}{print $2}' )
            tnNetMask=$(echo $query | awk 'BEGIN{FS="&"}{print $4}' |
awk 'BEGIN{FS="="}{print $2}' )
            tnBroadcast=$(echo $query | awk 'BEGIN{FS="&"}{print $5}' |
awk 'BEGIN{FS="="}{print $2}' )
            tnGateway=$(echo $query | awk 'BEGIN{FS="&"}{print $6}' |
awk 'BEGIN{FS="="}{print $2}' ) $DIR_SNMP/snmpset -v2c -O v -c tco-comm
localhost  1.3.6.1.4.1.21949.2.3.1.1.4.6 s "$hostname" \
          .1.3.6.1.4.1.21949.2.3.1.1.4.1 a $tnIpAddress \
          .1.3.6.1.4.1.21949.2.3.1.1.4.2 a $tnNetwork \
          .1.3.6.1.4.1.21949.2.3.1.1.4.3 a $tnNetMask \
          .1.3.6.1.4.1.21949.2.3.1.1.4.4 a $tnBroadcast \
          .1.3.6.1.4.1.21949.2.3.1.1.4.5 a $tnGateway 2>&1 > /dev/null
          if [ $? = 0 ]
          then
               /etc/rc.inet2 2>&1 > /dev/null
               echo "<center>Modifiche effettuate</center><br>"
               echo     "<meta     http-equiv='refresh'     content='1
url=setup.cgi?TN'>";
          fi
          elif [ $submit = "AGDF" ]
          then
              station=$(echo $query | awk 'BEGIN{FS="&"}{print $1}' |
awk 'BEGIN{FS="="}{print $2}' | ./tr [:lower:] [:upper:])
              nec=$(echo $query | awk 'BEGIN{FS="&"}{print $2}' | awk
'BEGIN{FS="="}{print $2}' )
              sr=$(echo $query | awk 'BEGIN{FS="&"}{print $3}' | awk
'BEGIN{FS="="}{print $2}' )
              gps=$(echo $query | awk 'BEGIN{FS="&"}{print $4}' | awk
'BEGIN{FS="="}{print $2}' )
              chn11=$(echo $query | awk 'BEGIN{FS="&"}{print $5}' | awk
'BEGIN{FS="="}{print $2}' | ./tr [:lower:] [:upper:])
              fs1=$(echo $query | awk 'BEGIN{FS="&"}{print $6}' | awk
'BEGIN{FS="="}{print $2}' )
              chn21=$(echo $query | awk 'BEGIN{FS="&"}{print $7}' | awk
'BEGIN{FS="="}{print $2}' | ./tr [:lower:] [:upper:] )
              fs2=$(echo $query | awk 'BEGIN{FS="&"}{print $8}' | awk
'BEGIN{FS="="}{print $2}' )
              chn31=$(echo $query | awk 'BEGIN{FS="&"}{print $9}' | awk
'BEGIN{FS="="}{print $2}' | ./tr [:lower:] [:upper:])
              fs3=$(echo $query | awk 'BEGIN{FS="&"}{print $10}' | awk
'BEGIN{FS="="}{print $2}' )
              chn41=$(echo $query | awk 'BEGIN{FS="&"}{print $11}' |
awk 'BEGIN{FS="="}{print $2}' | ./tr [:lower:] [:upper:])
              fs4=$(echo $query | awk 'BEGIN{FS="&"}{print $12}' | awk
'BEGIN{FS="="}{print $2}' )
              $DIR_SNMP/snmpset -v2c -O v -c tco-comm -t 1 localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.2.0 s "     " 2>&1 > /dev/null
              $DIR_SNMP/snmpset -v2c -O v -c tco-comm -t 1 localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.2.0 s $station \
                   .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.3.1 i $nec \
```

```
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.2.1 i $sr \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.4.1 s $chn11 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.8.1 i $fs1 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.5.1 s $chn21 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.9.1 i $fs2 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.6.1 s $chn31 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.10.1 i $fs3 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.7.1 s $chn41 \
                    .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.11.1 i $fs4  2>&1
> /dev/null
               $DIR_SNMP/snmpset -v2c -O v -c tco-comm -t 8 localhost
.1.3.6.1.4.1.21949.2.3.1.1.2.8 i 2 2>&1 > /dev/null #stop
               $DIR_SNMP/snmpset -v2c -O v -c tco-comm -t 20 localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.20.2.0 i 2 2>&1 > /dev/null
               if [ $? = 0 ]
               then
                    echo "<center>Modifiche effettuate </center><br>"
                    echo    "<meta    http-equiv='refresh'    content='3
url=setup.cgi?AGDF'>";
               fi
               $DIR_SNMP/snmpset -v2c -O v -c tco-comm -t 8 localhost
.1.3.6.1.4.1.21949.2.3.1.1.2.7 i 2 2>&1 > /dev/null #star
               else
                     if [ $submit = "AGDFEXT" ]
                     then
                     $DIR_SNMP/snmpset -v2c -O v -c tco-comm localhost
.1.3.6.1.4.1.21949.2.3.1.2.4.0 i 2 2>&1 > /dev/null
                     else
                          $DIR_SNMP/snmpset   -v2c   -O   v   -c   tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.4.0 i 1 2>&1 > /dev/null
                     fi
                     echo "<center><b>Cambio scheda effettuato, attendere
il ricaricamento della pagina</center><br>"
                     echo    "<meta    http-equiv='refresh'    content='2
url=setup.cgi?AGDF'>";
              fi
              else
                  echo "<form name='setup' id='setup' method='post'>"
                  echo    "<center><table    border='4'    bgcolor=#DDAACC
BORDERCOLOR=RED BORDERCOLORLIGHT=ORANGE BORDERCOLORDARK=RED ></center>"
                  echo "<tr>"
                  if [ $1 = "TN" ]
                  then
                 hostname=$($DIR_SNMP/snmpget -v2c -O v -t 2 -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.1.4.6 | sed s/\"//g)
                           echo     "<td>Hostname</td>     <td>     <input
type='text'       name='hostname2'       value='$hostname'       size='12'
maxlength='12'></td></tr>"
            tnIpAddress=$($DIR_SNMP/snmpget   -v2c   -O   v   -c   tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.1.4.1 | sed s/'IpAddress: '//g)
                     echo   "<tr>   <td>IP   Address</td>   <td>   <input
type='text'   name='tnIpAddress'   value='$tnIpAddress'   maxlength='12'
size='12'></td></tr>"
                tnNetwork=$($DIR_SNMP/snmpget  -v2c  -O  v  -c  tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.1.4.2 | sed s/'IpAddress: '//g)
                     echo "<tr> <td>Network</td> <td> <input type='text'
name='tnNetwork' value='$tnNetwork' maxlength='12' size='12'></td></tr>"
```

```
           tnNetMask=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm localhost
.1.3.6.1.4.1.21949.2.3.1.1.4.3 | sed s/'IpAddress: '//g)
                  echo "<tr> <td>Netmask</td> <td> <input type='text'
name='tnNetMask' value='$tnNetMask' maxlength='12' size='12'></td></tr>"
            tnBroadcast=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.1.4.4 | sed s/'IpAddress: '//g)
                  echo "<tr> <td>Broadcast</td> <td> <input type='text'
name='tnBroadcast'          value='$tnBroadcast'          maxlength='12'
size='12'></td></tr>"
            tnGateway=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.1.4.5 | sed s/'IpAddress: '//g)
                  echo "<tr> <td>Gateway</td>  <td><input type='text'
name='tnGateway' value='$tnGateway' maxlength='12' size='12'></td></tr>"
           elif [ $1 = "AGDF" ]
           then
              nome_stazione=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.2.2.0 | sed s/\"//g | sed "s/ //")
            echo "<tr> <td>Nome Stazione</td> <td> <input type='text'
name='nome_stazione'   value='$nome_stazione'   size='5'   maxlength='5'
onclick='rest_soft(this)'></td>"
             if [ $g200AgdfNumber -eq 2 ]
            then
                board=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.4.0)
                case "$board" in
                     1)  board="AGDFBASE"
                         ;;
                     2)  board="AGDFEXT"
                         ;;
                esac
                echo      "<td      colspan='3'>scheda      selezionata
$board</td></tr>"
            fi
            numberEnabledChannels=$($DIR_SNMP/snmpget -v2c -O v -c tco-
comm localhost .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.3.1 | sed s/\"//g)
            SampleRate=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost  .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.2.1)
            echo "<tr> <td>N. di Canali</td>"
            echo "<td><select name='nec'>"
            echo        "<option        value='$numberEnabledChannels'>
$numberEnabledChannels </option>"
           echo "<option value='0'>0 </option>"
           echo "<option value='1'>1 </option>"
           echo "<option value='2'>2 </option>"
           echo "<option value='3'>3 </option>"
           echo "<option value='4'>4 </option>"
           echo "</select></td>"
           echo "<td>Campionamento</td>"
           echo "<td><select name='sr'>"
           echo "<option value='$SampleRate'>$SampleRate</option>"
           echo "<option value='40'>40 </option>"
           echo "<option value='50'>50 </option>"
           echo "<option value='100'>100 </option>"
          echo "<option value='125'>125 </option>"
          echo "<option value='200'>200 </option>"
          echo "<option value='250'>250 </option>"
          echo "<option value='500'>500 </option>"
```

```
          echo "</select></td>"
          gps_type=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.13.0)
               case "$gps_type" in
               0) gps_type="INTERNO"
                   ;;
               1) gps_type="EXT 232"
                   ;;
               2) gps_type="EXT 485"
             esac
        echo "<td> GPS</td>"
        echo "<td><select name='gps'>"
        echo "<option value='$gps_type'>$gps_type</option>"
        echo "<option value='0'>INTERNO</option>"
        echo "<option value='1'>EXT 232</option>"
        echo "<option value='2'>EXT 485</option>"
        echo "</select></td></tr>"
        Channelname11=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.4.1 | sed s/\"//g)
        Channelfs11=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.8.1 | sed s/\"//g)
         echo "<tr> <td>Codice Canale 1</td> <td><input type='text'
name='Channelname11' value='$Channelname11' size='3'
max_length='3'></td>"
          echo "<td>Fondo Scala</td>"
          echo "<td><select name='fs1'>"
          echo "<option value='$Channelfs11'>$Channelfs11</option>"
          echo "<option value='2000'>2000 </option>"
         echo "<option value='2500'>2500 </option>"
         echo "<option value='4000'>4000 </option>"
         echo "<option value='5000'>5000 </option>"
         echo "<option value='8000'>8000 </option>"
         echo "<option value='10000'>10000 </option>"
         echo "<option value='16000'>16000 </option>"
         echo "<option value='20000'>20000 </option>"
         echo "<option value='40000'>40000 </option>"
         echo "</select></td></tr>"
         Channelname21=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.5.1 | sed s/\"//g)
         Channelfs21=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.9.1 | sed s/\"//g)
         echo "<tr> <td>Codice Canale 2</td> <td><input type='text'
name='Channelname21' value='$Channelname21' size='3'
max_length='3'></td>"
         echo "<td>Fondo Scala</td>"
         echo "<td><select name='fs2'>"
         echo "<option value='$Channelfs21'>$Channelfs21</option>"
         echo "<option value='2000'>2000 </option>"
         echo "<option value='2500'>2500 </option>"
         echo "<option value='4000'>4000 </option>"
         echo "<option value='5000'>5000 </option>"
         echo "<option value='8000'>8000 </option>"
         echo "<option value='10000'>10000 </option>"
         echo "<option value='16000'>16000 </option>"
         echo "<option value='20000'>20000 </option>"
         echo "<option value='40000'>40000 </option>"
         echo "</select></td></tr>"
```

```
         Channelname31=$($DIR_SNMP/snmpget   -v2c   -O   v   -c   tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.6.1 | sed s/\"//g)
         Channelfs31=$($DIR_SNMP/snmpget -v2c -O v -c tco-comm localhost
.1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.10.1 | sed s/\"//g)
         echo  "<tr>  <td>Codice  Canale  3</td>   <td><input  type='text'
name='Channelnames31'          value='$Channelname31'          size='3'
max_length='3'></td>"
         echo "<td>Fondo Scala</td>"
         echo "<td><select name='fs3'>"
         echo "<option value='$Channelfs31'>$Channelfs31</option>"
         echo "<option value='2000'>2000 </option>"
         echo "<option value='2500'>2500 </option>"
         echo "<option value='4000'>4000 </option>"
         echo "<option value='5000'>5000 </option>"
         echo "<option value='8000'>8000 </option>"
         echo "<option value='10000'>10000 </option>"
         echo "<option value='16000'>16000 </option>"
         echo "<option value='20000'>20000 </option>"
         echo "<option value='40000'>40000 </option>"
         echo "</select></td></tr>"
            Channelname41=$($DIR_SNMP/snmpget  -v2c  -O  v  -c  tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.7.1 | sed s/\"//g)
            Channelfs41=$($DIR_SNMP/snmpget  -v2c  -O  v  -c  tco-comm
localhost .1.3.6.1.4.1.21949.2.3.1.2.2.1.2.1.11.1 | sed s/\"//g)
            echo "<tr> <td>Codice Canale 4</td>  <td><input type='text'
name='Channelname41'                        value='$Channelname41'size='3'
max_length='3'></td>"
            echo "<td>Fondo Scala</td>"
            echo "<td><select name='fs4'>"
            echo "<option value='$Channelfs41'>$Channelfs41</option>"
            echo "<option value='2000'>2000 </option>"
            echo "<option value='2500'>2500 </option>"
            echo "<option value='4000'>4000 </option>"
            echo "<option value='5000'>5000 </option>"
            echo "<option value='8000'>8000 </option>"
            echo "<option value='10000'>10000 </option>"
            echo "<option value='16000'>16000 </option>"
            echo "<option value='20000'>20000 </option>"
            echo "<option value='40000'>40000 </option>"
            echo "</select></td></tr>"
         fi
         echo "</table></CENTER><br>"
         if [ $1 = "TN" ]
         then
            echo     "<center><input     type='submit'     name='submit'
value='Applica      $1'      onClick=\"alert('il      sistema      verrà
aggiornato')\"></center><br>"
         else
            echo     "<center><input     type='submit'     name='submit'
value='Applica      $1'      onClick=\"alert('Attendere      circa      30
sec.')\"></center><br>"
         fi
         if [ $g200AgdfNumber -eq 2 ]
         then
            case "$board" in
                "AGDFBASE")
```

```
                    echo   "<center><input   type='submit'   name='submit'
value='Selezione AGDFEXT'></center><br>"
                    ;;
                "AGDFEXT")
                    echo   "<center><input   type='submit'   name='submit'
value='Selezione AGDFBASE'></center><br>"
                    ;;
            esac
        fi
        echo "<script type=\"text/javascript\"> "
        echo " function rest_soft(thisfield) {"
        echo " thisfield.value=''; }"
        echo "</script>"
    fi
    echo "</form></CENTER></BODY></HTML>"
```

Appendice B: listato in linguaggio C del programma per il taglio dei file miniseed extr_seed

```
    #include <sys/types.h>
    #include <stdio.h>
    #include <errno.h>
    int main ( int argc, char* argv[] ) {
        int hhi=0, mmi=0, hhf=0, mmf=0;
        unsigned char seed_data[1024];
        char file_name_in[70], file_name_out[70];
        unsigned char inizio[6], fine[6];
        unsigned long int * orario;
        FILE * miniseed_in, * miniseed_out;;
        memset(file_name_in, '\0', 70);
        memset(file_name_out, '\0', 80);
        if (argc<6) {  //controllo numero argomenti passati al programma
           printf("uso: extr_seed file_miniseed hhi mmi hhf mmf \n");
           exit(1);
        }
        if( argc >= 4 ){
           strcpy(file_name_in, argv[1]);
           strcpy(file_name_out, file_name_in);
           sprintf(file_name_out,     "%s.%s.%s_cut",     file_name_out,
argv[2], argv[3]);
        }
        hhi = atoi(argv[2]);
        mmi = atoi(argv[3]);
        hhf = atoi(argv[4]);
        mmf = atoi(argv[5]);
        //apertura file miniseed da tagliare
        if((miniseed_in = fopen(file_name_in, "rb")) == NULL) {
            printf("File in open error");
        }
        //creazione nuovo file miniseed
          if((miniseed_out = fopen(file_name_out, "wb+")) == NULL) {
             printf("File out open error");
             }
         //lettura file miniseed
         fread(seed_data, 1, 1024, miniseed_in);
         if ((int)(seed_data[24]) == 0) {
             fseek(miniseed_in, hhi * 600 * 512, SEEK_SET);
         }
         do { //cerco l'orario di inizio
             fread(seed_data, 1, 1024, miniseed_in);
         }while    (((int)(seed_data[512    +    24])    <    hhi)    ||
((int)(seed_data[512 + 25]) < mmi));
         do {   //scrittura byte nel file di output fino all'orario di
fine
              fwrite(seed_data, 1024, 1, miniseed_out);
              fread(seed_data, 1, 1024, miniseed_in);
         }while    ((((int)(seed_data[512    +    24])    <    hhf)    ||
((int)(seed_data[512 + 25]) < mmf)) && !feof(miniseed_in) ) ;
         if     (!feof(miniseed_in))     fwrite(seed_data,     1024,     1,
miniseed_out);
         fflush(miniseed_out);
         //chiusura file miniseed
         fclose(miniseed_out);
         fclose(miniseed_in);
    }
```

Appendice C: listato in linguaggio C del programma per la comunicazione con i sensori Trillium

```
    #ifdef HAVE_CONFIG_H
    #include <config.h>
    #endif
    #include <string.h>
    #include <unistd.h>
    #include <stdio.h>
    #include <time.h>
    #include "serial_port.h"
    #define DIM 230
    int main (int argc, char *argv[])
    {
      int ris, i=0, status, size=DIM, pd;
      char port_name[11] = "/dev/ttyS4", serial, serial2;
      unsigned char serial3[DIM], comando[10];
      port_name[10] = '\0';
      memset(serial3, '\0', DIM);
      memset(comando, '\0', 10);
      if( argc >= 2 ){
            strcpy(comando,argv[1]);  //salvataggio parametro comando
      }
      else exit(1);
      pd = open_port(port_name, 9600);  //chiamata per apertura porta
      tcflush(pd, TCIFLUSH);
     //spedizione comando TX e attesa risposta sensore
      if ((ris = write(pd, "TX\r", 3))== -1) printf("Errore nella write
TX\n");
      if  ((ris  =  tcdrain(pd))  ==  -1)  printf("Errore  nella  tcdrain
TX\n");
      sleep(4);
      while(1) {
            ris = read(pd, serial3, 24);
            if(ris == 0) {
                  if(i<2) {
                        i++;
                        if  ((ris  =  write(pd,  "TX\r",  3))==  -1)
printf("Errore nella write TX\n");
                        if ((ris = tcdrain(pd)) == -1) printf("Errore
nella tcdrain TX\n");
                          sleep(4);
                  }
                  else {
                     printf("nessuna lettura\n");
                     close(pd);
                     exit(-1);
                  }
            }
            else {
                 if(strstr(serial3, "enabled") == NULL ) {
                 if ((ris = write(pd, "TX\r", 3))== -1) printf("Errore
nella write TX\n");
                 if  ((ris  =  tcdrain(pd))==  -1)  printf("Errore  nella
tcdrain TX\n");;
                 sleep(4);
                 i = 0;
              }
```

```
            else break;
         }
       }
       printf("%s\n",serial3);
       memset(serial3, '\0', DIM);
       i = 0;
       //spedizione comando passato come parametro
       if ((ris = write(pd, comando, 10))== -1) printf("Errore nella
write comando\n");
       if ((ris = write(pd, "\r", 1))== -1) printf("Errore nella write
CR\n");
       tcdrain(pd);
     //lettura output sensore distinta per commando (soh o center)
       if (!memcmp(comando, "center", 6)) {
             sleep(2);
             do {
                     ris = read(pd, &serial, 1);
                     if (ris == -1) Errore_("Errore read 1");
                     if (ris==0) {
                             sleep(1);
                             i++;
                     }
                     else {
                             printf("%c", serial);
                     }
             }while((serial != 'W') && (i < 90));
             ris = read(pd, serial3, 33);    //finisco l'output del
centraggio
             printf("%s\n", serial3);
       }
       else if(!memcmp(comando, "soh", 3))
       {
         sleep(2);
         do {
             ris = read(pd, &serial, 1);
             if (ris == -1) Errore_("Errore read 1");
             if (ris==0) {
                sleep(1);
                i++;
             }
             else {
               ris = read(pd, &serial2, 1);
               if (ris == -1) Errore_("Errore read 1");
               printf("%c%c", serial, serial2);
             }   //</SOH>
        }while(i<2);
      }
      //chiusura comunicazione con sensore
      if ((ris = write(pd, "TXoff\r", 6))== -1) printf("Errore nella
write TX\n");
      tcdrain(pd);
     do {
             ris = read(pd, serial3, 20);
      }while(ris!=0);
      sleep(1);
      fflush(stdout);
      close(pd);
```

```
 exit(0);
}
```

Coordinamento editoriale e impaginazione
Centro Editoriale Nazionale | INGV

Progetto grafico e redazionale
Daniela Riposati | Laboratorio Grafica e Immagini | INGV

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

http://www.ingv.it

Istituto Nazionale di Geofisica e Vulcanologia