Callisto data transfer and preparation

Callisto Data Transfer and Preparation

– version 1.1 –

Christina Pöpper, poepperc@student.ethz.ch

July 16, 2004

1 Overview

This document describes the structure and scripts for the data transfer and preparation of the Callisto spectrometer. Figure 1 illustrates the relationship between the scripts executing the transfer and preparation. The schedule of script execution is managed by a UNIX cron job, but the scripts can be called individually on demand. All scripts are implemented in Perl (indicated by *.pl*). The whole system is located on the server `pandora.ethz.ch`.

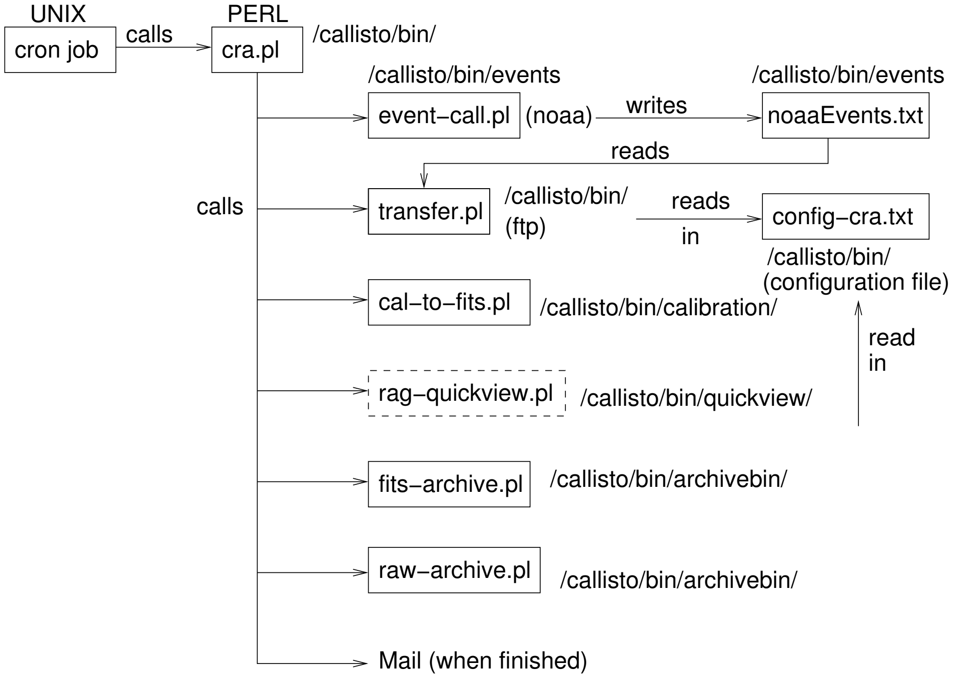

Figure 1: Overview of the system.

2 State of the System (July 16, 2004)

The following files and scripts exist. They can be found in the directory `/ftp/pub/rag/callisto/bin/` and its subdirectories.

cra.pl: The 'heart' of the system. It sets necessary configurations (such as environmental variables) and starts the whole process by calling the scripts.

event-call.pl: Script for extracting the event times from the NOAA[1] website used for the selection of files to transfer.

transfer.pl: Script for the data transfer from the remote host in Bleien to the local host at ETH Zurich.

cal-to-fits.pl: Script for the conversion from raw to fits files.

raw-archive.pl: Script for the archiving of raw data files.

fits-archive.pl: Script for the archiving of fits files.

subroutines.pl: 'Helper' script containing subroutines that are used by several cra scripts (reading in the configuration file, log file processes).

config_cra.txt: A simple text file containing the configuration information and default parameters used by the scripts (such as host name, directories, etc.).

initManually.txt: Contains the environmental variables to be set for calling the scripts manually from the command line (only for copying and avoiding tedious typing). These environmental variables may of course be set in the configuration file of the shell (such as `.bashrc` for the bash).

The remaining scripts from figure 1 are still to be written or completed, as implied by the dotted boxes.

[1] NOAA - National Oceanic and Atmospheric Administration, see www.noaa.com and www.sec.noaa.gov/ftpdir/indices/events/ for solar events.

3 Design

3.1 Directory Structure

`/ftp/pub/rag/callisto`, indicated by `$CALLISTO` in the following, is the root directory for CRA, the Callisto Remote Access tool. The following subdirectories can be found:

$CALLISTO/bin and its subdirectories contain the executable programs and scripts, including idl[2] scripts.

$CALLISTO/docu contains documentation material for the callisto remote access (CRA), such as this file. Note: All files `callisto.*` originating from the creation of this document must not be readable by non-rag members, because they contain password information. Keep to the access rights accordingly!

$CALLISTO/frequency-programs contains the configuration files for certain frequencies, which are needed by the data recording itself and the calibration.

$CALLISTO/misc contains the log files written by the cra scripts during data transfer and processing as well as future miscellaneous data.

$CALLISTO/observations contains the archived fits files. They are stored in subdirectories labeled by the date (such as `2004/07/03/`).

$CALLISTO/raw-archive contains the archived raw data files as well as the log and configuration files written during data recording. They are stored in subdirectories labeled by the date (such as `2004/07/03/`).

$CALLISTO/temp and its subdirectories contain temporal data; in particular the recently transferred data in the directory `$CALLISTO/temp/raw`.

3.2 Launching

The general design is shown in figure 1. Usually, `cra.pl` is called by the cron job once per day and then executes all the scripts necessary for the process of data transfer and calibration.

If certain scripts (such as `transfer.pl` and `raw-archive.pl`) are called manually directly from the command line, two environmental variables have to be set (which is done by `cra.pl` during automatic execution). The two variables are as follows (and can be found in `initManually.txt`):

[2] Interactive Date Language, see section 3.6

```
setenv CRA_BIN_DIR /ftp/pub/rag/callisto/bin/
setenv CRA_SUBROUTINES_SCRIPT /ftp/pub/rag/callisto/bin/subroutines.pl
```

If the directory structure changes or `subroutines.pl` is moved these two variables have to be adapted in `cra.pl` and on manual setting.

3.3 Cron Job

Automation of the callisto processes is achieved by a cron job. Listing the crontab for user *callisto* on the server *pandora* shows (`crontab -l` for listing; `crontab -e` for editing):

```
# Crontab for cra - callisto remote access tool
# NOTE: cra.pl should be executed after midnight -
#       otherwise the noaa list may still be lacking
30 02 * * * /ftp/pub/hedc/fs/data3/rag/callisto/bin/cra.pl
```

That means that the cra process is launched every day at 2:30 am.

Standard output during the program execution launched by a cron job is sent as a mail to the user this cron tab belongs to. Hence, all (error) messages are sent to `callisto@pandora.ethz.ch`. For convenience, all mails to user callisto are forwarded. This is done by storing a file with the name `.forward` (note the dot! - a hidden file) in callisto's home directory. This file looks like:

```
\callisto                      # a copy remains on callisto's account
monstein@astro.phys.ethz.ch    # include new recipients by storing
meyerh@astro.phys.ethz.ch      #         the address in a new line
```

3.4 Configuration File - config_cra.txt

Purpose: `config_cra.txt` is a simple text file containing the configuration information and default parameters that are used by the scripts (such as host name, directories, etc.).

Directory: `$CALLISTO/bin`

Format: Lines starting with # are comments and are ignored. Each parameter is listed one per line, followed by its value separated by a space. A new parameter can be added by writing its name and value into a new line. Empty lines may be used for structuring.

Sample File: The file can be found in figure 2.

Note: Some remarks:

- There is no restriction on the length of a parameter or parameter value, except for there must not be a line break within a word.

```
# for ftp: transfer.pl
host pisces
timeout 800
user_id xxxxxx
password xxxxxx
ftp

# directory on the remote host
remote_dir .

# directories/files on the local host used by transfer.pl
cra_dir /ftp/pub/rag/callisto
raw_dir /ftp/pub/rag/callisto/temp/raw
log_file /ftp/pub/rag/callisto/misc/cra.log
fits_dir /ftp/pub/rag/callisto/temp/fits
archraw_dir /ftp/pub/rag/callisto/raw_archive
archfits_dir /ftp/pub/rag/callisto/observations

# noaa base url needed by event-call.pl
noaa_url http://www.sec.noaa.gov/ftpdir/indices/events
# events extracted from noaa: event-call.pl & transfer.pl
noaa_dest_file events/noaaEvents.txt

# paths to used programs
w3c /usr/local/w3c-libwww/w3c-libwww
idl /usr/local/rsi/idl/bin/idl

# mail recipients
mail monstein,meyerh@astro.phys.ethz.ch

# cal-to-fits section, used by the idl scripts
# values may not contain whitespaces!
rawDataExt raw
freqProgDir /ftp/pub/rag/callisto/frequency-programs
freqProgPre frq
freqProgExt cfg
calibTableDir /ftp/pub/rag/callisto/bin/calibration
calibTableExt .txt
calibTablePre calib_
obsInfo.origin Bleien_Observatory
obsInfo.telescope Broadband_Spectrometer
obsInfo.instrument Callisto_V1.0
obsInfo.object Sun
```

Figure 2: The configuration file config_cra.pl

- The variable `host` may be any symbolic or IP name, such as `pisces`, `pisces.ethz.ch`, or `129.132.63.196`.
- `timeout` is the maximum time in seconds waited until the server answers while establishing the ftp connection. Default is 120 seconds.

3.5 Mails

As already mentioned, mails are sent during the execution of the cra scripts. To summarize it:

Mail on termination : Sent by `cra.pl` on finishing. The recipients of this mail are set in the configuration file.

Mail on failure/error : Sent by root on pandora. Caused by the cron job, when a scripts tries to print an error message to the standard output. The recipients, except for callisto itself, are specified in the `.forward` file in callisto's home directory. See also section 3.3.

3.6 Used Programs

For the execution of the perl scripts the following additional programs are necessary (additionally to PERL itself, of course :-)). The paths to these programs are set in the configuration file.

idl - Interactive Data Language: a general purpose scientific computing package, sold by the Research Systems Inc. (RSI)[3]. Needed by `cal-to-fits.pl` for the calibration.

w3c - W3C Libwww: A client side web API used for transfering files over the web (developed by W3C[4]), needed by `event-call.pl` for getting the Noaa data.

3.7 Background

We continue using *ftp* (file transfer protocol) for the file transfer from the host to the local server, as it was done for *phoenix*. One reason is the simpler and faster development, because parts of the implementation for *phoenix* could be reused. The second reason is that the alternative propositions (such as *scp*, secure copy, or *ssh*, secure shell) would not bring many advantages or reduce the drawbacks of ftp significantly.

[3] http://www.rsinc.com

[4] World Wide Web Consortium, see http://www.w3.org/Library/ for the library

ssh is nothing more than a program for remote login and for executing commands on the remote machine, providing encrypted communication. The data transfer itself is not solved by ssh.

scp copies files between hosts. It uses ssh for data transfer, uses the same authentication and provides the same security as ssh. The raw data to transfer in our case is public anyway and the password for the login on the host remains the security breach. The danger of missusage directly from within the building is much more critical than the insecurity through the wire between host and server.

3.8 Comment

It might be reasonable to modify the program priorities for the complex and time-consuming scripts during their execution (such as for the calibration scripts and `raw-archive.pl` with its zipping). There are two possibilities to achieve this:

- Change the priority of user callisto globally for all processes [need be done by root using 'usermod' probably]. This may, however, be an overkill, as many scripts running on callisto are not that time-consuming.
- Modify the program priorities of certain scripts directly during the call using 'nice'. This could be included in `cra.pl` and would look like `system ''/usr/bin/nice -n x $binDir/script.pl''`, `x` substituted by the priority adjustment, such as 5.

For the time being we decided to delay this decision, because pandora does not seem to run on full capacity.

4 The Scripts

4.1 Cra.pl

Purpose: `cra.pl` launches the default cra process as shown in figure 1 with date transfer, calibration, and archiving.

Directory: `$CALLISTO/bin`

Options: If `cra.pl` is called without options it launches the default execution.

--time [interval] : specification of a time interval for the raw data choice. [interval]=”start end”, where both match ’yyyymmdd_hhmmss’.

Note:

- Having executed all steps of the data transfer, calibration, and archiving, `cra.pl` sends a mail informing about the finished execution. Its recipients are set in the configuration file.
- Moreover, a particularity of `cra.pl` is the following: the raw data (`*.raw`) are fetched directly. As there occurs an access violation when fetching the currently written log file, ’get’ instead of ’fetch’ is used for the log files (`LOG*.txt`). They are removed from the host automatically after a certain number of days.
- The files not fetched remain on the host. They are deleted after a certain number of days (at the moment: 7 days), as also launched by `cra.pl`.

4.2 Event-call.pl

Purpose: `event-call.pl` extracts the relevant solar events for a certain day from the NOAA website using a w3c program and stores them in the text file `noaaEvents.txt` in the same directory.

Directory: `$CALLISTO/bin/events`

Options: `event-call.pl` may be called with two options, the date and whether a new log file is to be created.

--date [date|’today’] : the date for which the time intervals are to be extracted. It has the form *yyyymmdd* or ’today’ for today’s date.
Default: date of yesterday (because cra starts after midnight and transfers the raw files of the day before).

--newlog : if set, a new log file `log_file` (`$CALLISTO/ misc/cra.log`) will be created and the current one is appended to the old one (`$cra.log.old`).
Default: no new log.

Tool: `w3c-libwww` is used, located at `/usr/local/w3c-libwww/w3c-libwww`.

Output: `noaaEvents.txt` has a form as shown in figure 3:

```
#200040629
0018   ////   0019   RSP   III/1
0255   0301   0310   XRA   C1.2
0549   0605   0621   XRA   C2.5
1104   ////   1104   RSP   III/1
```

Figure 3: Sample form of noaaEvents.txt

The first line contains the date, then the events follow one per line. The first and the third column are of special interest for `transfer.pl`, because they contain the start and end time of an event. The second column contains the peak time of this event (`////` if not specified), the forth and fifth column specify the burst type and frequency/intensity.

4.3 Transfer.pl

Purpose: `transfer.pl` is responsible for the data transfer from the callisto server to the local server. It is implemented by FTP (file transfer protocol), allowing for temporal selection with the help of data from NOAA.

Directory: `$CALLISTO/bin`

Options: `transfer.pl` is called by the `cra.pl` main script, but can also be called from the command line on demand. Information during execution is written into the log file `log_file` (`$CALLISTO/misc/cra.log`). The options for `transfer.pl` are:

-h Print information to the script.

-i Ping host: Checks connection to host.

-v Verbose: Set verbose mode and print progress information (on the command line). Even if not set, the log file will be written.

-f [file] Fetch [file]: Transfer file(s) from remote to local host, deleting every transfered file immediately; accepts wildcards.

-g [file] Get [file]: Transfers file(s) from remote to local host; accepts wildcards.

-t [interval|'noaa'] Only composed with **-g** or **-f**: specification of a time interval. [interval] = "start end", where start and end both are of the form *yyyymmdd_hhmmss*.

['noaa']: select the files to transfer by the NOAA observed events stored in `noaaEvents.txt`.

-d [**file**|**number**] Delete [file]: Removes file(s) from the remote host; accepts wildscards.
Delete [number]: If a number is given instead of a file name, all raw files whose age correspond to this number of days, are deleted.

-p [**file**] Put [file]: Transfers file(s) from local to remote host; accepts wildcards.

-r [**cmd**] Remote [cmd]: Execute command on remote host.
Accepted commands:

`rm [file]`	removes file(s) on remote host
`ls [file]`	list directory / file information on remote host

-s [**parameter**] [**value**] Set one or several cra parameters.
Accepted variables:

`host`	host to connect to
`user_id`	userid
`password`	password
`raw_dir`	local directory for raw data

In addition to executing the actions the log file will be written (its location is set in `config_cra.pl`).

Sample Use: Called either from the command line or from another script.
From the command line:

```
> ./transfer.pl -i
```
to check the connection using ping

```
> ./transfer.pl -s "raw_dir ./tmp/text" -g "*.txt"
```
to set the raw directory and get all .txt files

```
> ./transfer.pl -g "*.raw" -t "20040603_080000 20040603_093000"
```
to get all raw files in the specified time interval

From the `cra.pl` script:

```
system "./transfer.pl -f '*.raw' -t 'noaa'"
```
to fetch all raw files using the noaa event list.

Note: Some things more to say...

- On every execution of `transfer.pl`, new log information is appended to the log file `$CALLISTO/misc/cra.log`.
- The raw data files are assumed to stick to the file name format `*yyyymmdd_hhmmss*.*`, where `*` is a sequence of any characters with the restriction that the first `*` contains no `1` or `2`, which represent the beginning of the year date.
- If the option `-t 'noaa'` is set, but the file `noaaEvents.txt` does only contain the date, an error at noaa is assumed and hence all

raw files are transfered. If `noaaEvents.txt` does not exist at all, `event-call.pl` is called first to write this file.

4.4 Cal-to-fits.pl

Purpose: `cal-to-fits.pl` converts the raw files into calibrated fits files using a number of idl scripts.

Directory: `$CALLISTO/bin/calibration`

Options: `cal-to-fits.pl` may be called with or without options. If no options are set the default values are taken as specified below.

--help Print information to the script.

--rawdir [**dir**] Set the rawdir to [dir], containing the raw date files to convert.
Default: specification in the configuration file

--fitsdir [**dir**] Sets the fits directory to [dir].
Default: specification in the configuration file

--file [**file**] Alternatively to rawdir: specify the raw files to convert in the current working directory.
Default: specification in the configuration file

--(no)verbose print progress information to the standard output
Default: no verbose (the log file will be written nevertheless)

Note: All `cal-to-fits.pl` does is writing an instruction file for idl. It then calls idl with this file name as a parameter. The conversion itself is implemented in idl, see section 5.

Sample Use: >`./cal_to_fits.pl --file 'B*.raw' --fitsdir '.'`
to convert all files matching the pattern 'B*.raw' in the current working directory and store the resulting fits files in '.', as well.

4.5 Fits-archive.pl

Purpose: `fits-archive.pl` archives the fits files. It zips the fits files and moves them from a temporary directory to the appropriate subdirectory of `archfits_dir`, which is specified in the configuration file `config_cra.txt`. It appends progress information to the log file `log_file` (`$CALLISTO/misc/cra.log`).

Directory: `$CALLISTO/bin/archive`

Options: `fits-archive.pl` may be called with or without options. If no options are set the default values are taken as specified below.

--help Print information to the script.

--archivedir [**dir**] archive root directory
Default: specification in the configuration file

--fitsdir [**dir**] directory where the files to archive are located
Default: specification in the configuration file

--(no)zip does (not) compress .fit files before moving
Default: zip

--(no)verbose print progress information to the standard output
Default: no verbose (the log file will be written nevertheless)

4.6 Raw-archive.pl

Purpose: `raw-archive.pl` archives the raw data files. It moves the complete set of raw files (including LOG*.TXT and *.CFG) from `raw_dir` to the appropriate subdirectory of `archraw_dir`, which are both specified in the configuration file `config_cra.txt`. It appends progress information to the log file `log_file` (`$CALLISTO/misc/cra.log`).

Directory: `$CALLISTO/bin/archive`

Options: `raw-archive.pl` may be called with or without options. If no options are set the default values are taken as specified below.

--help Print information to the script.

--rawdir [**dir**] directory of raw data files
Default: specification in the configuration file

--archroot [**dir**] root archive directory
Default: specification in the configuration file

--archdir [**dir**] archive directory relative to archroot
Default: actual day

--(no)verbose print progress information to the standard output
Default: no verbose (the log file will be written nevertheless)

--(no)zip zip the raw data files
Default: zip

4.7 Subroutines.pl

Purpose: `subroutines.pl` provides subroutines that are used by the cra scripts such as reading in the parameters in the configuration file and executing log file related actions (open, close, write).

Directory: `$CALLISTO/bin`

Subroutines: `subroutines.pl` provides the following subroutines (among others):

readParameters(): reads in the parameters from the configuration file and returns them in a hash %parameters.

openLog(. . .), **writeLog(. . .)**, and **closeLog(. . .)** for log file related actions

getDate(. . .) for extracting the date from a file name.

5 From RAW to FITS in IDL

The process of converting `.raw` files to `.fit` files is implemented in IDL. The IDL scripts can be found in **$CALLISTO/bin/calibration/idl**, having the ending `.pro` for IDL. The following description gives an overview over the active scripts found in this directory (which is illustrated by figure 4). [**calprocess.pro** and **calpusher__define.pro** are not active (i.e. not in use) at the moment.]

calcalib__define.pro : The 'heart' of the system: responsible for the calibration part. So far, it performs a dummy calibration, but is predetermined for using a calibration table later on. Change calibration details here! Particularly redo the addition and substraction for interpolation and polarization here (after the calibration has been implemented).

calconfig__define.pro : Reads in the parameters from the configuration file `config_cra.txt`.

calfreqprog__define.pro : Reads the frequency programs in the `$CALLISTO/frequency_programs` directory.

calmeasurement__define.pro : Reads raw files, writes fit files. Creates fit file names. Add focus code mappings to alphanumeric strings here!

cal_to_fits.pro : Master script, launches everything.

extractdatetime.pro : Helper script to extract all kinds of information from a header. Used by calmeasurement.

str_pad.pro : Helper script for string processing. Used by extractdatetime.pro and calcalib__define.pro.

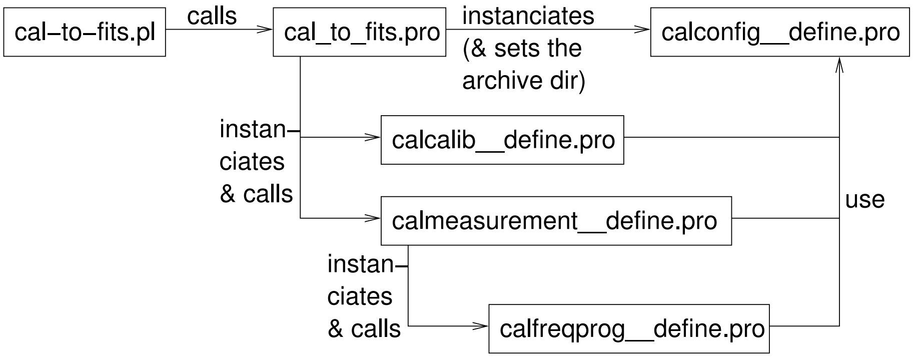

Figure 4: Overview of the main idl scripts for the file conversion.

6 Moving the Callisto Software

This system has been developed keeping in mind the portability of the software. The goal was to be able to easily transfer it to another callisto system. For making it run wherever note the following points:

- Copy the bin directories **$CALLISTO/bin** recursively.
- Make sure you keep to the directory structure as described in section 3.1.
- Adjust the configuration file `config_cra.txt`.
- If the necessary additional programs do not exist yet, install them, see section 3.6.
- Make sure that the configuration files for the needed frequencies exist in **$CALLISTO/frequency-programs**. Otherwise the calibration process will fail.
- Create/Modify the `.forward` file in the home directory of the user (i.e. of callisto) to set the forward mail addresses in case of failure.
- Verify all access rights!

Good Luck!

7 Open Issues

- Elaboration of the raw-to-fits conversion. So far, there exists only a dummy calibration.
- Make all IDL scripts write all process information to the configuration file and then remove the **1>/dev/null** (when calling idl) term that ignores print commands to the standard output (otherwise the cron job would send a mail every time).
- One further suggestion: Copy `cra.pl`. Then this copy can be adapted (comment out etc.) for manual calls if something went wrong without worrying about restoring the original file that is called by the cron job.

Of minor importance are:

- Delete raw data after half a year automatically
- Long Options for `transfer.pl`: `--get` instead of `-g` for easier use?

8 Remark

This document and the described system originate from a semester thesis, which was supervised by Christian Monstein, monstein@astro.phys.ethz.ch, and Hansueli Meyer, meyerh@astro.phys.ethz.ch, in the *Radio Astronomy and Plasma Physics Group* under professor Arnold Benz, Institute of Astronomy, ETH Zurich.