VERY LARGE TELESCOPE

Gasgano User's Manual

VLT-PRO-ESO-19000-1932

Issue 4
23/03/2007

Prepared:	M. Klein Gebbinck D. Sforna		
	Name	Date	Signature
Approved:	P. Ballester		
	Name	Date	Signature
Released:	M. Peron		
	Name	Date	Signature

CHANGE RECORD

Issue	Date	Affected Paragraphs(s)	Reason/Initiation/Remarks
1.0	27/9/99	All	Preliminary Draft
1.4	26/2/01	All	Draft PSO Commissioning MAR2001
1.4	26/2/01	All	Revised Version
2	2002	All	Public release
3	04/27/2004	All	General update and addition of Recipe Panel section.
4	23/03/07	3,4,5,6,7	General Update

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. WELCOME TO *GASGANO*

Gasgano is a Data File Organiser developed and maintained by the European Southern Observatory (ESO) to help its user community manage and organize astronomical data observed and produced by VLT compliant telescopes. The tool also supports FITS files which are not generated by those telescopes but with limited functionality.

1.1 Purpose & Scope

This document is the user's manual for the *Gasgano* ESO Data Flow System File Organiser, in short *Gasgano*. It is intended for guiding ESO-external users through its basic usage.

This manual describes *Gasgano* Version 2+, the second public release of the tool. The tool was distributed to internal ESO users for the first time at the end of 1999 and since then it has successfully assisted Paranal Science Operations in their daily operations with the large amount of data generated by the VLT.

The manual is organized as follows: Chapter 2 gives an Overview of the tool, describing its main and most general features and functionalities; Chapter 3 leads the user through a simple session of *Gasgano* with the aim of showing the basic usage of the tool; Chapter 4 takes the user deeper into *Gasgano*, to discover how the tool can be customized to the user's needs; Chapter 5 describes the execution of external programs launched from within the tool; Chapter 6 details the loading and execution of ESO CPL recipes. Chapter 7 provides a quick reference guide to keywords, rules, keyboard-shortcuts, etc. At the end, three Appendixes complete the *Gasgano* documentation: System Requirements (Appendix A), description of configuration parameters and a listing of the .gasganorc file (Appendix B), and information on the Verbose Levels (Appendix C).

This manual does not contain an installation guide. Installation instructions are maintained online on the *Gasgano* web pages ([http://www.eso.org/gasgano)](http://www.eso.org/gasgano). The latest version of this manual will also be available from this web site.

1.2 Intended Audience

This manual is intended for all astronomers making use of ESO VLT compliant telescopes, who after obtaining/receiving their set of data need to process them in an efficient way. Familiarity with the high-level operational concepts used to operate VLT compliant telescopes (e.g. Observation Block) and with the ESO data format (in particular the hierarchical keywords in FITS files) is desirable.

1.3 Reference documents and web pages

[1] GEN-SPE-19400-794/1.1/25-Nov-97 - *Data Interface Control Document*

[2] VLT-SPE-ESO-19000-0842/0.92/14-Aug-97 - *VLT Data Flow System, Architectural Design Document*

[3] VLT-PRO-ESO-19000-1930/1.0/27-Sep-99 – *VLT Data Flow System, **Gasgano** DFS File Organiser Design Document*

[4] For more information on science operations and the VLT Data Flow, go to http://www.eso.org/projects/dfs/

[5] The Common Pipeline Library http://www.eso.org/observing/cpl

1.4 Glossary

Where appropriate, these definitions were adopted from the Data Interface Control Document ([1]). Minor wording changes have been made since *Gasgano* is used at several ESO facilities, not just the VLT.

Acquisition Template (AT): an Observation Block component. An Acquisition Template is used to specify how a target is acquired by the telescope. It may also specify any preliminary instrument configuration steps (e.g. set rotator to specific angle). It can contain parameters for interactive as well as automatic acquisitions. This template may define a different instrument/detector configuration from the templates within the Observation Description. Each science Observation Block contains at most one Acquisition Template.

Calibration Observation Blocks: Observation Blocks used to acquire calibration data.

Common Pipeline Library (CPL): An ESO-DMD developed data reduction recipe encapsulated in a shared object library that can be read into Gasgano and executed.

Data Flow System (DFS): a system that can handle the flow of data and information within certain ESO facilities, in particular, the VLT. It includes subsystems for proposal handling, observation handling, science and engineering data archiving, science data pipeline processing and handling and data quality control [2]. The DFS is being developed by the *ESO Data Management and Operations Division* (DMD).

Data Organizer (DO): software, part of the DFS, used in the on-line environment to support the automatic pipeline data reduction of incoming data.

Data Products: data files delivered to PI/Co-Is as result of the execution of ESO observing programs or setup information submitted to ESO in relation to observing programs. Data products include observation frames, observatory calibrations, meteorological and seeing measurements relevant to primary observations, etc.

Exposure: a synonym for the acquisition of a single data frame, typically resulting in a single FITS file.

Header Keywords: header keywords can be standard FITS (8 character long, e.g. TELESCOP, MJD-OBS), ESO standard (e.g. HIERARCH ESO OBS PROG ID) or *Gasgano* "Meta Keywords". Within *Gasgano* HIERARCH ESO keywords are specified in "dot" notation without the HIERACH ESO part. E.g.: OBS.PROG.ID represents HIERARCH ESO OBS PROG ID.

Master calibration product: a reduced frame used as a master for calibration of raw science and calibration frames.

Observing Run: an approved ESO Program consists of one or more Observing Runs, each of which specifies an independent combination of telescope, instrument, and observing operations mode (i.e. Service Mode or Visitor Mode).

Observing Program ID: see Run ID.

Observation: a coordinated sequence of telescope, instrument, and detector actions that

results in a scientific or technical dataset.

Observation Blocks (OB): the smallest observational unit within the Data Flow System. An Observation Block contains a sequence of high level operations, called templates, which need to be performed sequentially and without interruption in order to ensure the scientific usefulness of an observation. Observation Blocks may contain only one target acquisition. Observation Blocks may contain scheduling and collect all status information of operations as they pass through the Data Flow System from definition, execution, and reduction. Observation Blocks are used in both Visitor and Service Mode to acquire data.

Observation Block ID: the unique identifier of an Observation Block. It is stored in the keyword HIERARCH ESO OBS ID. Valid OB ID examples: 19368, 200083938

Raw Frame: Result of execution of an exposure by the VCS. The execution of an Observation Block may generate one or more Raw Frames. Raw frames are stored as FITS files. Their headers contain all the information relevant for pipeline data reduction, QC, and archiving.

Reduction Block (RB): A Reduction Block contains all the information needed to reduce one or more raw frames: it contains the name of the pipeline recipe to be applied, the list of input frames, the list of associated master calibration files, and the name of the output pipeline products to be generated.

Reduction Block Scheduler (RBS): software, part of the DFS, which operates on Reduction Blocks, obtained from the Data Organizer: it analyzes the RB, checks the presence of input files, and fires the corresponding data reduction recipe.

Reduction recipe: standard procedure for reducing observational data in a standard way. Recipes are implemented for each of the instrument standard templates. Those scripts take as input raw frames and execute them in a particular Data Reduction System (DRS).

Pipeline Product: Result of the execution of a Reduction Block. Pipeline products are stored on disk as FITS files (image or table) or ASCII files (e.g. PAF file).

Run ID: As with Observing Program ID: the ID of an Observing Run. The format is kPP.X-0000(X) where kpp.c-nnnn where k is blank for normal, 1 for key programmes, pp is the period, c the category defined for proposals approved by the ESO OPC and nnnn a sequential number. Valid Run ID examples: 63.A-0352(A), 167.A-4563(B). The Run ID is stored in the HIERARCH ESO OBS PROG ID keyword.

Service Mode (SM): observing operations mode where the astronomer submits a detailed description of their observing program to ESO for later possible execution. Service Mode programs are executed primarily in order of their ESO Observing Programs Committee assigned priority but only when the astronomer specified observing conditions are achieved onsite.

Science Observation Blocks: Observation Blocks used to acquire scientific data.

Template: a high-level data acquisition operation. Templates provide the means to group commonly used procedures into well-defined and standardized units. They can be used to specify a combination of detector, instrument, and telescope configurations and actions.

Templates have input parameters described by a template signature, and produce results that can serve as input to other templates. As an example, an Acquisition Template takes target coordinates and produces through an interactive procedure the precise positions used later, e.g. to place an object on a slit.

Visitor Mode (VM): observing operations mode where the astronomer is present at the telescope while their observing program is being executed**.**

VLT Control System (VCS): the software and hardware tools that are used to control VLT instruments, telescopes, and related hardware. It enables and performs the acquisition of scientific and technical data. Originally designed for the Very Large Telescope, the VCS has also been installed at the La Silla New Technology Telescope (NTT) and 3.6m telescope.

1.5 Abbreviation and Acronyms

AT   Acquisition Template

CPL   Common Pipeline Library

DFS   Data Flow System

ESO   European Southern Observatory

GUI   Graphical User Interface

OB   Observation Block

RB   Reduction Block

SM   Service Mode

UT   (VLT) Unit Telescope

VM   Visitor Mode

VLT   Very Large Telescope

VRB   Virtual Reduction Block

VCS   VLT Control System

1.6 Conventions used in this document

- Tips are highlighted by a "candle bullet":

  ' This is a tip!

- Examples and default values are typed using Courier fonts.

- *Gasgano* graphical user interface labels and key items are typed in **bold**.

- Keystrokes combinations are typed in ***underlined italic***.

1.7   How to get help or report problems with *Gasgano* or this manual

In all of the following situations:

- *Gasgano* bug reports

- Suggestion for improvements to this product or manual

- Technical assistance

Contact the **ESO User Support Group** (usg-help@eso.org)**.**

2. OVERVIEW

Gasgano is a GUI software tool for managing and viewing data files produced by VLT Control System (VCS) and the Data Flow System (DFS). The main purpose for developing it was to provide a user-friendly tool capable of dealing with the large amount of data generated by the VLT and the other ESO astronomical facilities.

As you go through this guide you see that the *Gasgano* distribution includes a series of satellite files that are instrument-dependent (e.g. classification rules). At the time of release, *Gasgano* includes instrument packages for the following (VLT) instruments: FORS1 and FORS2, ISAAC, UVES, VIMOS. As new instruments are released to the community, these will be included in future versions of the *Gasgano* distribution kit.

2.1 Files supported by *Gasgano*

Gasgano understands and supports the specific types of files listed in the following sections, and ignores the others.

2.1.1 Raw Data

All VLT compliant instruments generate data which are stored as FITS files. In addition to the FITS standard keywords, ESO uses a set of primary keywords and a set of hierarchical keywords in its data products. The additional primary keywords, although not belonging to the official FITS standard are widely used by the community. One of them, MJD-OBS is essential for the DFS and *Gasgano* (e.g. to sort files). It represents the modified Julian Date at the start of the observation. Hierarchical keywords are used as means to group keywords that belong to the same logical entity, i.e. domain. An ESO hierarchical keyword starts by convention with "HIERARCH ESO" and is followed by words describing the domain. All ESO official dictionaries are available on-line through the following link:

http://archive.eso.org/Tools/DidRep/DidRepWebQuery

Important hierarchical keywords for *Gasgano* are:

HIERARCH ESO PROG ID = `60.A-9203(A)': corresponds to the unique identifier of the parent Observing Run.

HIERARCH ESO OBS ID =   200103093   : corresponds to the unique identifier of the parent Observation Block

Note that *Gasgano* supports and uses shorter format for hierarchical keywords: The "HIERARCH ESO" prefix is not used and all spaces are replaced by dots. For example: HIERARCH ESO INS FILT1 ID is being shortened to INS.FILT1.ID.

Gasgano is able to understand and load FITS files which do not contain those keywords; however the functionality of the tool is restricted.

2.1.2 Pipeline Products

The VLT instrument pipelines generate for each raw frame and/or each set of raw frames one or more pipeline products of the types described below.

- **FITS images and tables**: These files contain a number of hierarchical keywords which are being used by *Gasgano* e.g.:

HIERARCH ESO PRO CATG = 'MASTER_IMG_FLAT_BADPIX': is used to identify a FITS file as being a pipeline product. The keyword value is also used to classify the data.

The keywords RECi RAWj contain all the information describing the set of input frames (raw and calibration) used to generate the product. They provide means to associate the pipeline products with the corresponding raw frames and master calibration data. Note that pipeline products created before October 2001 do not contain those keywords.

HIERARCH ESO PRO REC1 RAW1 NAME = 'ISAAC.2001-06-09T07:14:46.460.fits'

HIERARCH ESO PRO REC1 RAW2 NAME = 'ISAAC.2001-06-09T07:16:05.599.fits

HIERARCH ESO PRO REC1 CAL1 NAME = 'IS_SSFM_010608A_SK_s03t_216.fits'

HIERARCH ESO PRO REC1 CAL1 CATG = 'MASTER_SP_FLAT'

HIERARCH ESO PRO REC1 CAL1 DATAMD5 = 'dda44920b1dda6fe222fc5a0a5092040'

- **PAF and ASCII files**

Some of the instrument pipelines (e.g. ISAAC) create in addition to the primary pipeline products additional ones which contain for instance Quality Control information such as zero points. Those products are stored on disk either in a Parameter File (PAF) format or in an ASCII format.

2.1.3 Reduction Blocks

Reduction Blocks are entities created and used in particular by the pipeline in its on-line environment (at the VLT). They contain all the information needed to reduce one or more raw frame, i.e. name of the recipe, list of appropriate calibration frames and name of the outputs to be generated. They provide means for associating pipeline products to raw and master calibration data.

2.2 What *Gasgano* does

The functionalities offered by *Gasgano* can be summarized as follows:

- **Data Grouping/Sorting:** *Gasgano* groups files located on a specified set of directories automatically into a tree structure based on the program ID and observation ID of the files. Files are put into the same folder/subfolder if they have some common features: same Observing Run ID/ same Observation Block ID within the same observing run. This holds for all types of files supported by *Gasgano*. Users have the option to change the virtual view by first grouping their data around the directory to which they belong or/and by defining instrument specific groups. The tree structure is refreshed every time

a new file is loaded into the tool. Note that *Gasgano* is capable of differentiating raw files from pipeline products: the former are displayed in blue, whereas the latter are displayed in red (reduced frames) or grey (e.g. reduction blocks). Furthermore, by default, pipeline products are grouped together with the raw and master calibration frames that were used to generate them, making the raw-processed data association very intuitive. Within a folder files are ordered after their alphabetical names. The option exists to order them by the value of the MJD-OBS keyword.

- **Data Classification:** *Gasgano* overcomes the well-known difficulty of understanding the *type* of a data file is (e.g. BIAS, DARK, SCIENCE, etc.). The archive filenames used for ESO data products ensure the file names are unique, but they are unfortunately user-unfriendly. *Gasgano* automatically assigns a classification tag to all FITS files (tables and images) which have been loaded by the tool, by applying a set of keyword based logical rules. These are instrument dependent and can be modified by the user (cf Chapter 4). A set of classification rules for FORS1, FORS2, UVES, ISAAC and VIMOS are part of the *Gasgano* delivery package. Data reduction products are classified using the hierarchical keyword PRO CATG.

- **Data Browsing/Filtering:** *Gasgano* allows the user to select a list of keywords to be displayed in the tree view for all FITS files loaded by the tool. Users might browse these folders in a friendly and effective way, or filter the file set based on logical keywords-based expressions specified on the fly. Filtering and searching functionality permits searching folders for specific keyword values.

- **Data Viewing/Searching:** *Gasgano* provides means for viewing and searching all the relevant information of a data file. Viewing options include displaying FITS images and header keyword information (as a whole or based on a selected set of keywords only, e.g. by using the Filter option). The Search option allows searching keywords/keyword values within a displayed header and FITS extensions are supported.

- **Data Handling:** *Gasgano* allows the user to make mouse selections on the files displayed on the tree. The headers of selected files are visualized in detail (or just in part, depending on the filtering option used). Selected files may additionally be moved, copied, archived, or piped into an external display tool. The Report option provides a useful tool to generate a text report on the selected files. These options (Move, Copy, Archive, Report, etc.) are *Gasgano* pre-defined commands. Furthermore, *Gasgano* may also be used as a front-end application to other UNIX tools, such as instrument pipelines. Once a set of files has been selected, users can send them as direct inputs to external executables and scripts.

2.3 *Gasgano* at a glance

The purpose of this section is to introduce the tool, to familiarize the user with its look and main functions. A more detailed session follows in the next chapter.

Figure 1 shows the main screen of the tool as it appears when started. It consists of two panels: the N**avigation Panel** at the top and the **File Detail Panel** at the bottom. A sliding horizontal bar separates the panels and allows panel re-sizing.

' On the left edge of each sliding bar there are two control buttons, of triangular shape, which allow expanding and collapsing of the panel, both upwards and downwards.
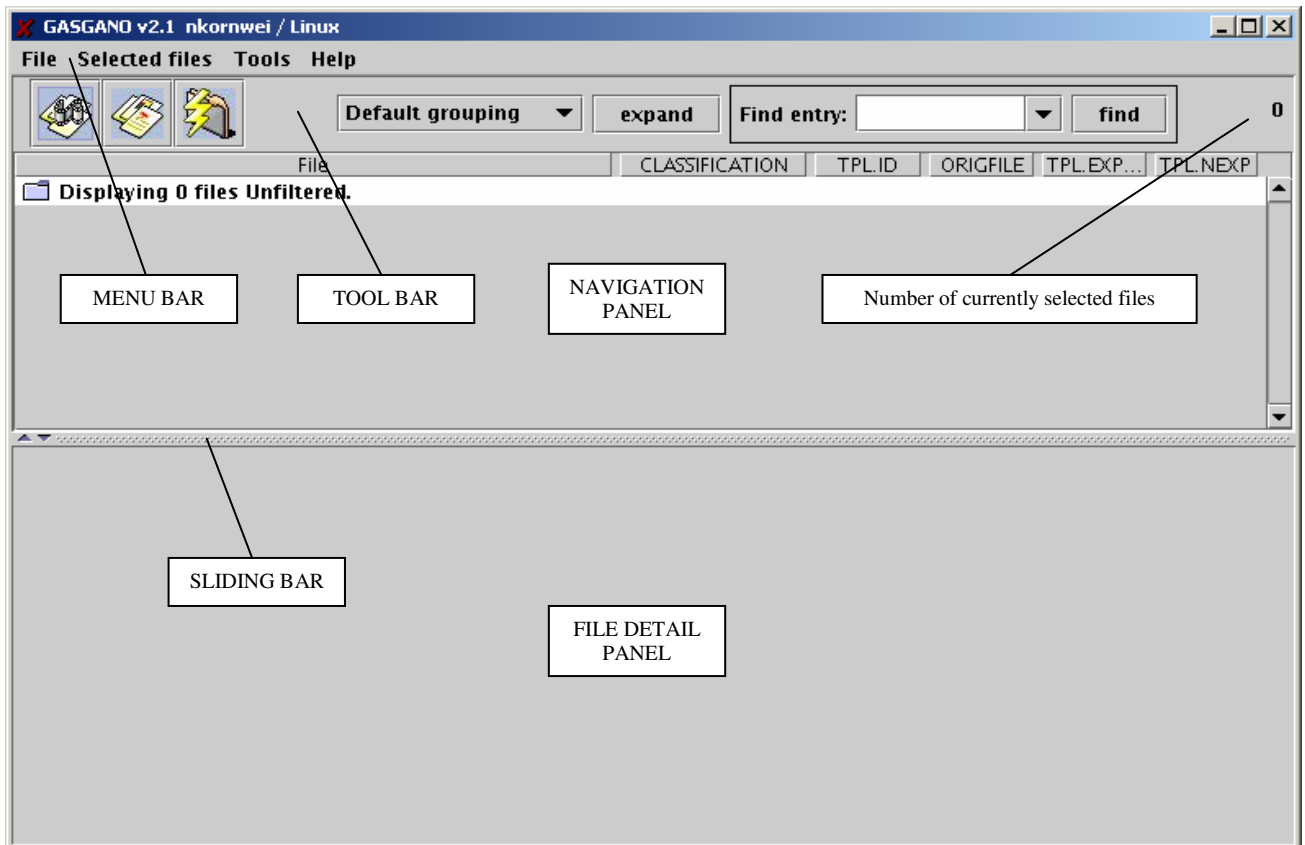

Figure 1: *Gasgano*'**s main panel. The panels are Navigation Panel (top) and the File Detail panel (bottom). Panels are separated by a horizontal sliding bar that can be moved up or down with "click and drag" or can be collapsed using the triangular icon on their left side. The Menu bar contains File, Selected Files, Tools, and Help pull-down Menus. Note: by default the File Association Panel is hidden. It may be displayed if required with the File>Show Frame Assoc menu item.**

Gasgano can also be set to display the File Association Panel. This is useful when running Gasgano on a file set that contains Reduction Block files (*.rb, *.rx, *.ry or *.rz). The File Association Panel is turned on/off through the File->Show Frame Assoc. menu item.

Figure 2 shows the main screen of the tool with the File Association Panel. It consists of three panels: the N**avigation Panel** at the top, the **File Association Panel** in the middle and the **File Detail Panel** at the bottom. Sliding horizontal bars separate the panels and allow panel re-sizing. Note: by default the File Association Panel is hidden. It may be displayed if required with the **File>Show Frame Assoc** menu item.
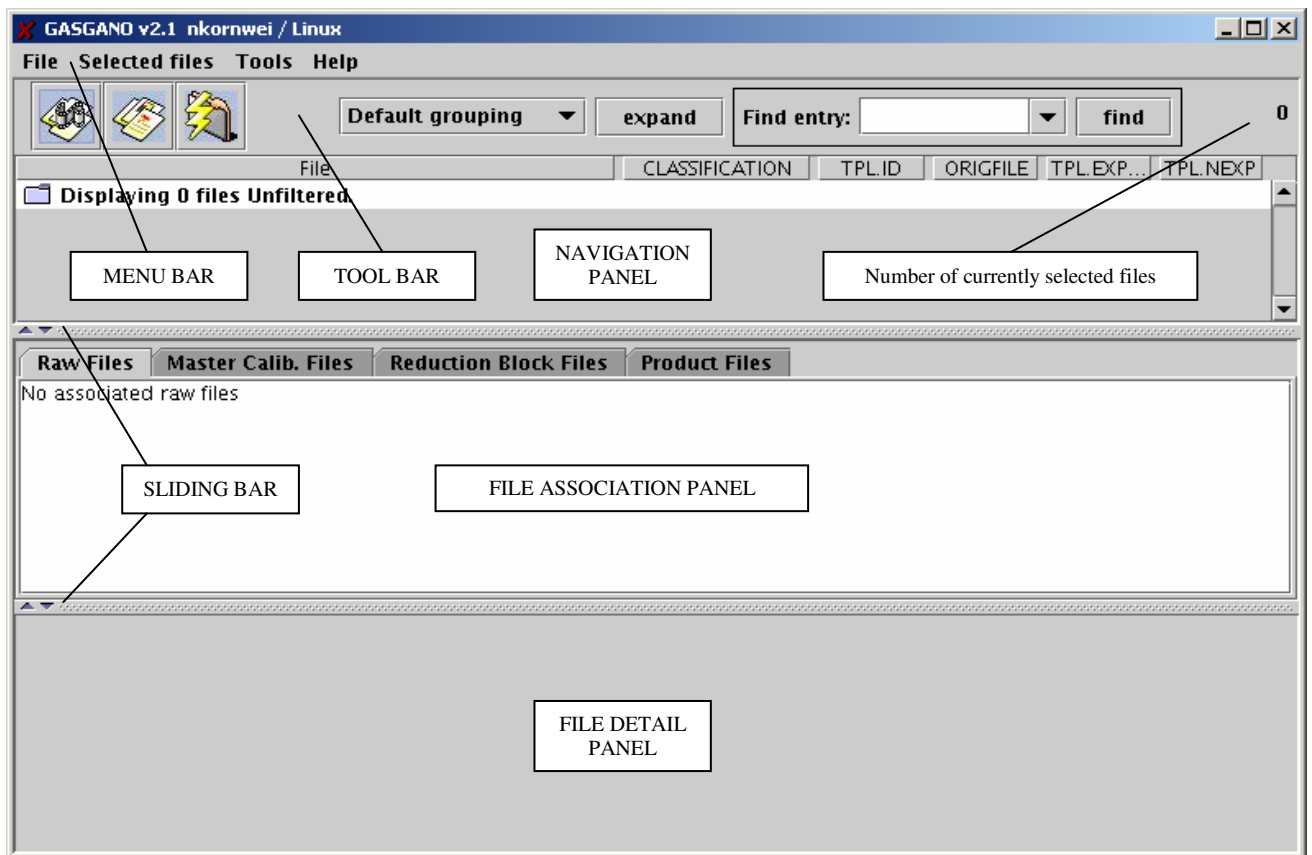
Figure 2: *Gasgano*'s main panel. The three sections are, from top to bottom, the Navigation Panel, the Association Panel, the File Detail panel. Sections are separated by horizontal sliding bars, which can be moved up or down with "click and drag" or can be collapsed using the little triangular icon on their left side. The Menu bar contains File, Selected Files, Tools, and Help pull-down Menus. Note: by default the File Association Panel is hidden. It may be displayed if required with the File>Show Frame Assoc menu item.

I. The **Navigation Panel** can be identified by (from top to bottom) :

    q the **Menu bar** (**File, Selected Files, Tools** and **Help**)

    q the **Toolbar,** showing from left to right:

        § The **External Viewer button** which starts a viewing application (e.g. ds9, skycat, rtd, etc) and displays the selected file,

        § The **Frame Report button**, which creates reports in a tabular format containing a user-defined list of keyword values of the selected file(s).

        § The S**cript Board button**, which launches the script board window, to run scripts on the selected file(s).

        § The **Grouping Menu button,** which changes the grouping of the files in the **Navigation Panel** (e.g. grouping around directory contents or telescope).

§ The **Expand/Collapse button,** which unfolds/folds all the branches of the tree**.**

§ The **Find Entry** field and **find** bottom, which allow the user to search the **Navigation Panel** for any entry typed in.

§ The **File counter:** at the right end of the **Toolbar** a number indicates the total number of files currently selected.

Files loaded by the tool are shown in the **Filesystem** view of the **Navigation Panel**. For each FITS file in the view a set of (user-defined) list of keywords is displayed. The keyword names are displayed at the top of each column. In the example shown in Figure 1, these are: CLASSIFICATION, TPL.ID, ORIGFILE, TPL.EXPNO, and TPL.NEXP.

II. The **File Association Panel** (by default not displayed) displays associations between set of raw frames, pipeline products and the master calibration frames used to process them. The file associations are made through Reduction Block files. If these files are not present in the file set, this panel will be essentially empty. It is organised into four different registers:

  ❑ **Raw Files**,

  ❑ **Master Calibration Files**,

  ❑ **Reduction Blocks,**

  ❑ **Product Files**.

Given a file selected in the Filesystem view of the navigation panel, clicking on one of those registers will result in the corresponding and appropriate type of data being displayed.

III. The **File Detail Panel** shows the keyword blocks of the last file selected in the **Navigation Panel.** This panel can be used to filter the header using a user-defined list of keywords and to search for strings in the header itself. Depending on the type of file selected, the **File Details Panel** displays different details: the header of fits and tfits files, the content of PAF and ASCII files (including reduction blocks).

Once a set of data files has been loaded into the tool (more on this step in Chapter 3), the main *Gasgano* panel resembles Figure 3, below. In the example below the **Navigation Panel** displays 50 files. This can be seen the first row of the panel which indicates how many files is loaded and how they are grouped and/or filtered. In the example, the default grouping mechanism is being used, that is the files are organised in folders, according to their Run ID (e.g. 168.A-0322(B)) and their OB ID (e.g. 100120929). Each folder is identified and shown with a node icon, a folder icon, and a line of text describing the folder (**folder line).** The latter may be, for instance, the combination of the OBS.PROG.ID + INSTRUME keywords, as seen in Figure 3 (e.g. 60.A-9022(B) GIRAFFE).

Raw files are displayed in blue, reduced in red, text files in grey. The icons for each file differ according to the file type.

For each FITS file (image or table) the values of some selected and user definable keywords are displayed. In this example CLASSIFICATION (SKY FLAT), **TPL.ID** (WFI_img_cal_SkyFlat), ORIGFILE, TPL.EXPNO, and TPL.NEXP are displayed.

The file WFI.2003-04-15T10:39:55.363.1.fits is currently selected, as can be seen by the light blue highlight band. On the upper right corner of the window the number 1 gives the total number of selected files.
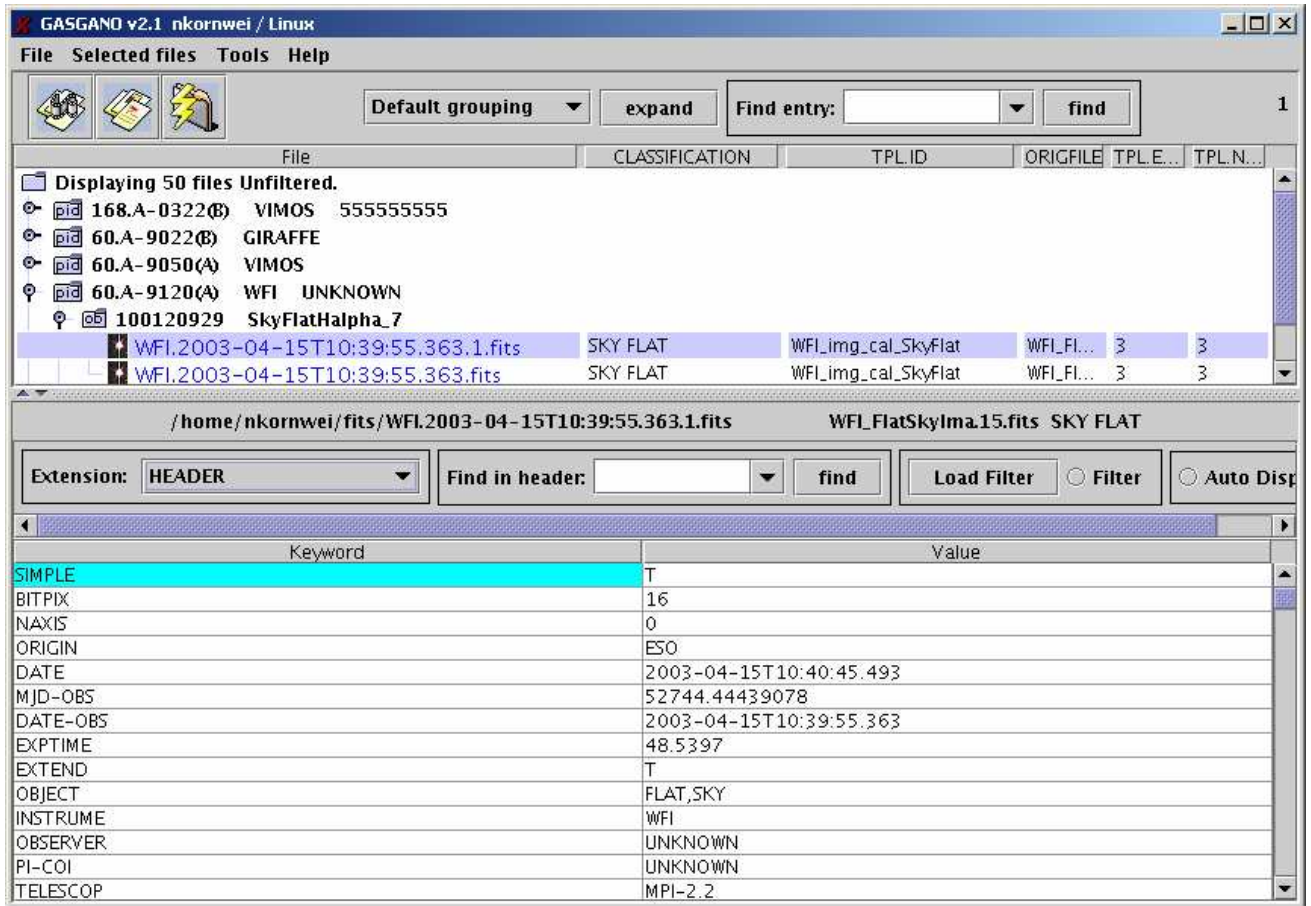
Figure 3: *Gasgano*'s main panel after loading a VLT data set. Refer to the text for a description of this figure.

The lower **File Details** Panel displays the keyword blocks for the selected WFI fits file. The **Extension** combo box permits navigation between the various keyword blocks in the fits file. Currently the primary header (**HEADER**) is selected. The **Find in header** and **Filter** control permit searching and filtering the keywords displayed.

3. A SIMPLE SESSION

This chapter describes a simple session with *Gasgano* and its most fundamental features. If you are a first time user please take some time to read this part.

3.1 Running *Gasgano*

To start the application, at the prompt, enter the following command:

- `gasgano &`

  If *Gasgano* starts correctly, the main screen will appear. If the DISPLAY setting is incorrect or .Xauthority is not correctly set, the application will not start, and messages similar to the following will be displayed on the screen (note you will need to use the –debug option to see

these messages):

```
DISPLAY environment variable not set – exiting
```

or

```
Xlib: connection to "xt313:0.0" refused by server
Xlib: Client is not authorized to connect to Server
```

3.2   Your local directory structure and configuration files

With the default installation, a directory named "gasgano" is created under ${HOME}. This directory contains all the executables and the accessory files. These files should never be modified directly by the user, except:

- o   The configuration file called .gasganorc created in ${HOME} (cf Appendix B for an example).

- o   the additional directories used to store:

    - keyword lists for reports and filtering of headers (defaulted to ${HOME}/gasgano/keywords),

    - classification rules files,

    - rules for file filtering (both defaulted to ${HOME}/gasgano/),

    - and user defined scripts (defaulted ${HOME}/gasgano/scripts).

The user may change the default values by means of the **Preference editor** (see Chapter 4).

If *Gasgano* does not find the .gasganorc configuration file, or if this file is not correct, the following message will be displayed:

```
Failed to load properties file: /diska/astro1/.gasganorc
Creating /diska/astro1/.gasganorc with default values
```

In such case clicking OK will create the .gasganorc file and start the application with default settings.

3.3   Exiting *Gasgano*

To exit *Gasgano*, select **File>Quit** on the Menu bar or type *CTRL+Q*.

3.4   Loading files into *Gasgano*

To load files into the application, click on **File>Add/Remove Files…**. A window appears (see below, Figure 4).

Figure 4: the Edit directory set **window allows adding or removing directories/files to/from the** Navigation Panel. **In this example no files are loaded. Clicking on** Add File **opens a file chooser dialog for browsing and selecting files and directories.**

To add directories and files, click on the **Add File** button. A file chooser dialog appears, as shown in Figure 5.

It is possible to navigate through the file system either double clicking on a folder to change directory or using the first button after the combo box on the upper toolbar to move up one level.

Files can be loaded by a single click on the file name and then pressing the **Add File** button. A set of files can be selected as well, using the Shift-click feature (for consecutive items), or the Ctrl-click (for non-consecutive feature).

A directory can also be loaded (its name should appear in the field **File Name** at the bottom). A set of directories can be selected as well, using the Shift-click feature (for consecutive items), or the Ctrl-click (for non-consecutive feature).

Loading a directory will automatically load all its subdirectories. If some of them are not needed, they have to be selected in the **Edit directory set** panel (cf Figure 4) and removed by clicking on the **Remove Selected Files** button. When removing directories or files from the list of loaded ones, multiple selections are possible. To learn how to do multiple selections, go to Section 3.6.
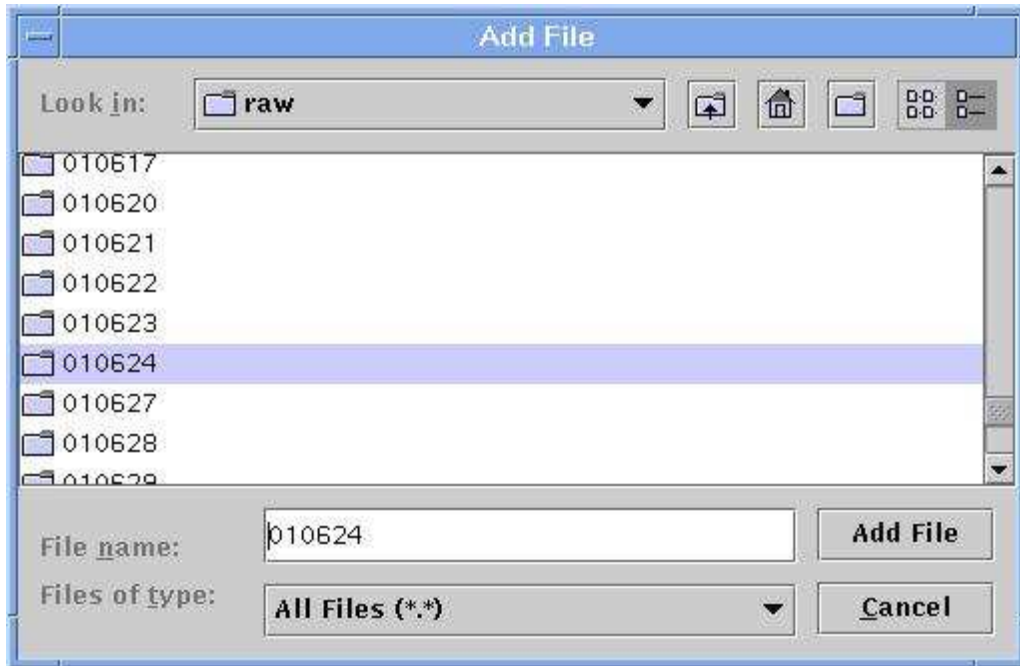
Figure 5: The Add File **Panel allows browsing the file system to select directories and files to be loaded into the application. To select a directory/file, single click on it and then click on the** Add File **button. Multiple selections when adding directories is** NOT **possible.**

Once files are loaded into the tool, the **Navigation Panel** will look like Figure 6.
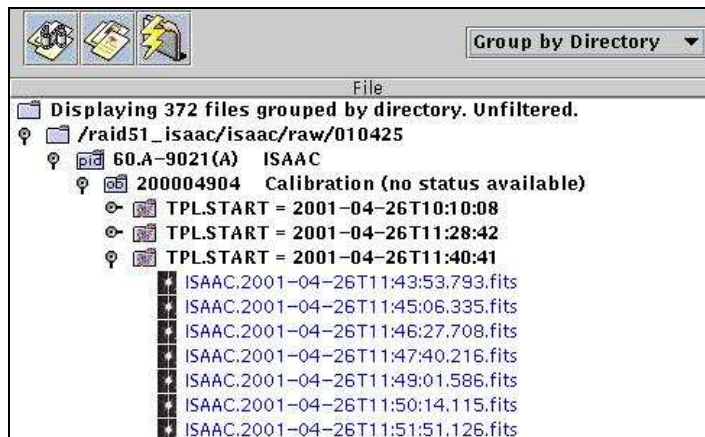
Figure 6: snapshot of the Navigation Panel **when the** Group by Directory **option is selected. In addition to the three levels OBS.PROG.ID (60.A-9021(A)), OBS.ID (200004904), and TPL.START, the path of the files is included at the top: these files are under the directory** /raid51_isaac/isaac/raw/010425**.**

3.5   Refreshing the contents of the folders

It is possible to request a rescanning of the directory contents to update the Navigation Panel (for instance after moving/deleting files) by using the **Refresh option,** under **File>Refresh** in the **Menu** bar. There is also an automatic, periodic, refresh function. The automatic refresh option is useful when the content of the loaded directories changes frequently and can be activated by setting up the refresh time in the **Data** tab of **File>Preferences**.

3.6 Changing the view

The view in the **Navigation Panel** can be changed in various ways, according to the user's preferences or needs.

3.6.1 Grouping files

On the **Toolbar** of the **Navigation Panel**, on the right side of the icon buttons, there is a selector menu (combo box), which sets the grouping mode. Its default value is **Default Grouping**. The other available options are **Group by Directory** and **Group by Telescope**. The default setting displays files grouped according to their OBS.PROG.ID, OBS.ID and any other optional grouping keywords defined by the user. The **Group by Directory** option adds an extra layer on top of the default view and groups files according to their location in the file system. The complete absolute path of each group of files present in the same directory is displayed in the panel, as shown in Figure 5.

The **Group by Telescope** option works in the same way as the **Group by Directory** but adds a top layer which group files according to the keyword TELESCOP (e.g. ESO-VLT-U1).

3.6.2 Expanding and collapsing the view

The first time a data set is loaded into the application, the tree view in the Navigation Panel is collapsed, and only the list of the top folders (e.g. the list of OBS.PROG.ID) is displayed. In order to view the content of the folders below, single click on the folder line (at any position, i.e. on the node, on the folder icon or on the name). Be aware that a second click closes the folder (that is, the double click has no effect, and if you are fast enough no changes in the view are noticeable!!). Clicking through the various layers of the tree will expand/collapse the selected tree node. It is possible to expand or collapse the entire tree at once, by means of the **Expand/Collapse** toggle button on the **Toolbar**.

3.6.3 Editing the list of keywords to be displayed

The list of keywords displayed in the Navigation Panel can be edited/modified using the register in the **File Display & Grouping tab** of **File>Preferences…** . Short names for ESO hierarchical keywords (e.g. TPL.ID) are used.

3.6.4 Resizing the fields

All fields in the **Navigation Panel** can be resized to accommodate the length of the keywords' values. The fields names are displayed in a bar just below the main **Toolbar,** in the order specified in the **"***FileSystem – Displayed Keywords"* item , **File Display & Grouping** tab of **File>Preferences…**. When the mouse pointer is on the separation line between two fields, it changes its shape. Clicking the mouse left button allows grabbing the separation line and resizing the field to the preferred length. Note that with some windows manager the mouse pointer might not change the shape, resizing is however always possible.

3.6.5 Selecting Files

Any operation you may want to do on your file(s), such as viewing the header content, displaying a frame with a viewer, copying/moving, creating reports, etc, requires a selection. Selected files appear highlighted in light blue. Single selection is done by single clicking on the file. Multiple selections are possible in two ways:

After selecting the first file via a single click, the user can:

1) press the *Shift* key while mouse clicking on another file: this selects a block of contiguous files;

2) press the *Ctrl* key while mouse clicking on another file: this adds individual files to the selected list.

In all cases, if files are selected, the action of clicking will deselect all of them except the file on which the user clicks on.

Once a set of files is highlighted, the **Selected files** menu provides the list of actions that may be applied to the files. The **Selected files** menu is also displayed when pressing the right mouse button on a file selection.

In Chapter 6, Table 2 summarises all mouse click options.

3.7   Search tools

It is possible to find entries in the **Navigation Panel** and in the **File Details Panel** by means of the "**Find" tools**. To find an entry in the **Navigation Panel**, type a search string in the field **Find entry** on the upper **Toolbar** and then click on **find**, to the right. The string is searched "as is", i.e. no wildcard characters may be entered. The search is case insensitive. A successful match is highlighted. Pressing **find** again will locate the next match in the Navigator Panel, looping back to the start when the end of the data is reached.

To find an entry in a fits header proceed in the same way but from the **File Details panel**, using the **Find in header** field on the panel's toolbar.

3.8   Displaying Files

Gasgano offers the possibility to define default file viewers, as explained in Section 4.1.2, even though NO viewers are distributed together with the application, however several example scripts are provided showing how to connect Gasgano with ds9, skycat, rtd, etc. The simplest way to display a file is to select it and to click on the leftmost button icon on the **Toolbar** (if you move the mouse pointer over this icon, the tool tip reads "External Viewer"). The selected file is passed to the defined application/script, which displays the file. It is possible to define three types of viewers, which will be automatically started by the tool, according to the type of selection in the **Navigation Panel**:

1) Single tfits table: the tool runs the tfits viewer

2) Single fits selection: the tool runs the fits file viewer

3) Multiple fits file selection: the tool runs a viewer that accepts data cubes as input

It is advisable to use a script which calls the viewer application and to launch the viewer in the background. This permits Gasgano to continue running after launching the viewer.

In addition to the **Toolbar** button, the viewer can be launched to display a selected file from the **Menu** bar**,** choosing **Selected files>Display ...** item**.**

3.9 Copying/Moving Files

Gasgano can move or copy files from one location to another one, as selected by the user. After selection of file(s), click on the **Menu Bar** on **Selected Files>Copy** or **Selected Files>Move.** A window similar to that shown in Figure 4 appears: it allows browsing the file system to select the destination directory. When the location has been selected, click on the **Copy/Move** button. According to the operation here selected, *Gasgano* may behave in different ways:

1) If files are moved to a new directory (i.e. not loaded), the directory is automatically loaded by the tool

2) If files are copied to a new directory the directory is not automatically loaded by the tool

3) If files are copied or moved to a directory that is already loaded by Gasgano, a manual refresh will display the new files (click on **File>Refresh** in the **Menu bar**.)

✐ It is not possible to delete a file from *Gasgano*, but you can move files to a designated "wastebasket" directory in your file system and then remove them manually.

3.10 Creating a tar file

To create a tar file out of a set of selected files click on the **Menu bar**, **Selected Files>Tar….** A window similar to the one shown in Figure 4 appears. Select the destination directory and type the name of the tar file in the field **File name,** at the bottom. Click on the **tar** button. The tool will "freeze" for the time needed to create the tar file (i.e., the time needed by the underlying UNIX workstation to perform the job). This can be quite long if the number of files is large. The syntax of the tar command used by *Gasgano* is:

tar –cbf 20 - <list of parameters>

where <list of parameters> is the list of files selected in *Gasgano*.

3.11 Creating a keyword list file

Keyword lists are ASCII files used in various parts of *Gasgano* for example to show subsets of file headers or print reports.

To create a keyword list file from scratch, click from the **Menu Bar** on **Tools>Edit Keyword Lists….** A new window, called **Keyword List Editor** appears (cf Figure 6). It contains two columns, **Keyword Name** and **Report Label**. At the bottom there are the buttons **Insert Row** and **Delete Row**. On the upper **Menu Bar**, the **File Menu** contains the entries: **Open, New, Save As** and **Quit.**

Add as many rows as needed with the **Insert Row** button. In the first column enter the fits keyword as it appears in the file header, i.e. in uppercase and including the dot notation (but without the HIERARCH ESO part). The **Report Label** column allows the user to specify the label used by the application when creating a report. The syntax and the contents are free.

Once the list is complete, click on **File>Save As…** to save the file.

The file should be saved in the default directory specified in the **Preferences**, under the **Data** register, since this is the default directory where *Gasgano* looks when trying to load a keyword list file (default is ~/gasgano/keywords).

To edit an existing keyword list file, click on **File>Open** in the **Menu bar** of the keyword list editor. The tool opens the usual file system browse window, in the default directory (as specified in the **Preferences**). Browse and/or select the file, then click **Load**. Edit the file as needed and then save. To create a new file: click on **File>New**.

File>Quit quits the editor without saving the file.

′ The keyword list files are standard ASCII files and can be edited with any editor, without having to use the built in editor.

′ Keywords and, in fact, any text in Gasgano that may be selected can be copied into the clip board using Ctrl-C. To paste the text in another text field, press Ctrl-V.

Figure 7: The Keyword List Editor window. The list of keywords has been edited and the file saved to disk.

3.12   Filtering file headers

On the top part of the **File Detail Panel** there is a **Load Filter** button and a **Filter** toggle on/off button. Click on **Load filter** to have the browsing window open on the default directory for keyword lists. Select the file and click on **Load**. Click on the toggle button **Filter** so that it is checked. If a file in the **Navigation Panel** is selected, only the header keywords present in the keyword list file will be displayed. To remove the filter, deselect the toggle button **Filter**. To load another keyword list file, repeat the above procedure.

When the fits header filtering is on, the **Find in header** works only on the currently displayed keywords. You may need here to enlarge the main *Gasgano* panel to see all keywords associated to the current selected file.

3.13   Filtering files

Gasgano offers the possibility to filter files loaded in the **Navigation Panel** on the basis of keyword values. The concept is similar to that of the classification rules. It is possible to set a constraint on a keyword, or a set of keywords by means of AND/OR logical expressions, and to have the tool displaying only those files that fulfil that condition.

To filter files select **File> Filter Files** from the **Menu** bar. The **Filter Builder** dialog will be displayed (cf Figure 8). It consists of an upper panel of existing filters and a lower panel (Expression Builder) for editing or adding filters. The lower window allows editing in the same way as the **Classification rules builder**. Refer to Section 4.2, for a description of the syntax and how to edit these rules. Once OK is pressed, the currently selected filter is applied to the loaded data set. Rules are saved in the ${HOME}/gasgano/config/filter.rul file, unless otherwise specified in the **Preferences**.
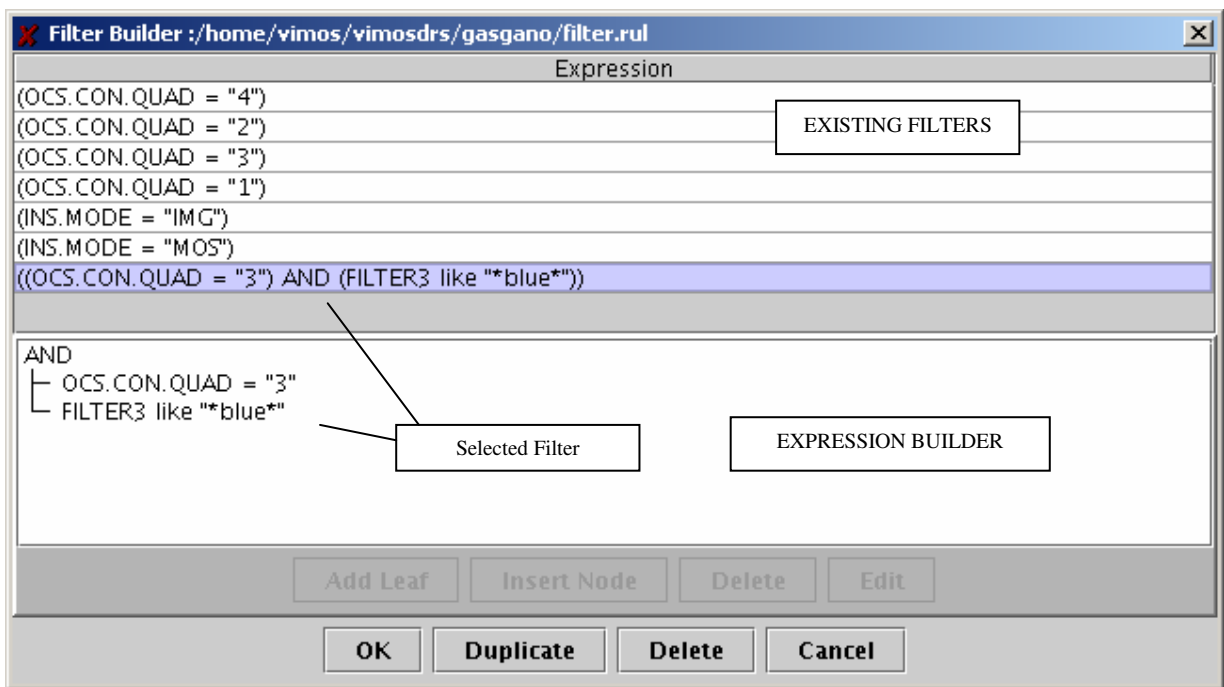
Figure 8: The Filter Builder **window. In this example, rules are saved in the file** filter.rul. **It is possible to configure the file and its location through the** File>Preferences **panel. The upper panel shows a list of rules (Existing Rules), with the selected rule (highlighted in blue) displayed in the lower panel (Expression Builder).**

Once the filter has been applied, the first row of the **Navigation Panel** displays the filter information, for this example:

Displaying X files. Filter: ((OCS.CON.QUAD="3") AND (FILTER3 like "*blue*"))

Where "X" is equal to the number of files matching the filter. To remove the filter: click on

the **Menu bar** the **File>Unfilter files**.

′ When a data set is filtered, the application keeps in memory the files that have been filtered out. Therefore, removing a filter does not require any re-loading of the files from disk.

′ The filter files option is very useful when dealing with a large data set. It is usually recommended to load at first the entire data set and then to apply filters. One application is, for instance, to filter out all calibration files (DPR.CATG!="CALIB") when just a quick look at the scientific frames is needed.

3.14 Creating a report

A report is an ASCII file that displays in a tabular format file header keywords (or a sub-set of them, as specified) for a group of fits and/or tfits files. To create a report a keyword list file is needed. If none is available, create one as explained above. Then:

1) Select fits or tfits files in the **Navigation Panel** ;

2) Click on the **Report** button on the main **Toolbar** (the second icon button from left) or click on **Selected files>Report…;**

3) The **Load Keyword list** window appears. At first use it is empty, while for subsequent iterations the last loaded keyword list is kept in memory;

4) If it is empty, click on the **Load Keyword List** button at the bottom; the browse window opens in the default directories for keyword list files. Browse and select a file.

5) Modify the loaded file if needed, using the **Insert Row,** and **Delete Row** buttons. To add a row above another one, select reference line and right click on the **Insert Row** button. Edit the fields as needed.

6) Click on **OK**.

7) The report is generated and the report window is opened (see Figure 8).

Figure 9: The Report **window. A list of files (previously selected in the** Navigation Panel**) is displayed with their keyword values. If a keyword is not present in the header, this is replaced by a blank space (like in the case of ra and** dec **for this example).**

On the **Menu bar** of the **Report** window the **File** and **Options Menus** are available:

The **File Menu** contains the **Print, Save As…,** and **Quit** entries. **Print** sends the report file to the default printer (as specified in the tab **File Display & Grouping** of **Preferences** submenu**); Save As**… allows saving the file to disk (the usual file chooser window is opened).

The **Options Menu** contains the **Select Fonts** entry. When selected it opens the **Edit fonts** window, which allows selecting a different font type and size from the one selected in the **File Display & Grouping** tab of **Preferences** submenu**.** The selected font is used only for the current report. To save it permanently: edit the entry in the **Preferences.**

' Monospaced fonts are ideal for printing because they maintain the correct alignment of the columns. Be aware that other fonts may not do this.

4. CONFIGURING *GASGANO* PREFERENCES

Gasgano was designed to be flexible and configurable to the greatest possible extent in order to accommodate a wide range of uses. The tool comes with a standard configuration set, which can be tailored to specific user requirements by means of the **Preference Editor**.

4.1 Running the Preference Editor

To access the **Preference Editor**, click on **File>Preferences…**. The **Preference Editor** window will appear. The preferences editor is organised in the following tab panels:

1. **Data:** settings related to the file set, file locations and the file system.

2. **File Display & Grouping:** display preferences and external viewer configuration.

3. **Recipe Configuration:** the set of CPL recipes available

Clicking on any of these tabs activates a configuration panel. All panels have an **OK** and **Cancel** buttons on the lower part. Accepting the preferences via the **OK** button will have the twofold effect of saving them into the .gasganorc configuration file and enforcing them in the current session. The modification of certain settings may require either reloading the data set or restarting the application. In either case you will be warned of any required action.

' The preference file (.gasganorc) is an ASCII text file and can be edited using any ASCII text editor. Refer to Appendix B for the list of preference items and their meaning.

4.1.1 The Data preferences tab:

The **Data** register, shown in Figure 10, contains file system and directory settings: filenames of filter and classification rules, recognised text files extensions and the (default) location of the keyword list and scripts directory. The **Data** tab is divided into three main sections:

1) In the upper part, the **Data Directories and Files** field specifies which directories or files are automatically loaded at start-up. Adding entries to this panel and accepting the preferences with the **OK** button automatically loads the corresponding data sets into memory, if not yet loaded. Selecting a directory file selects all files in that directory. To load a directory:

  q Click on the **Add File** button. A dialog box, which allows navigation through the file system, is opened.

  q Select the desired directory or file and accept the selection by clicking on the **Add File** button (a more detailed description of the **Add File** dialog box can be found in section 3.4).

  q To remove a directory from the loaded list, select it with a single mouse click and then click on the **Remove Selected Files** button. When removing items from the list, multiple selections (*CTRL+click*) are possible**.**
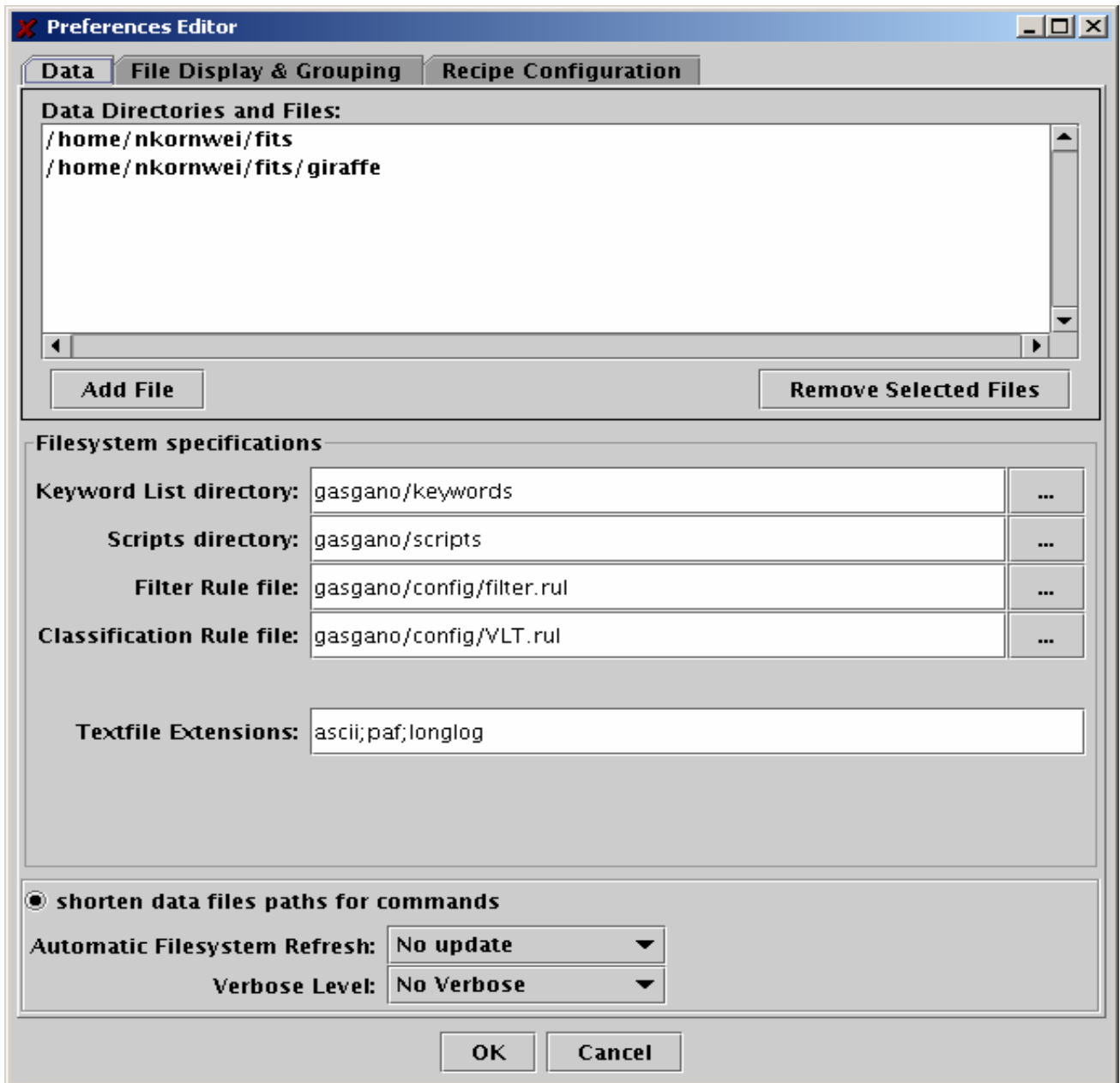
Figure 10: The Data panel under the Preferences Menu

q In the middle section of the register, the **Filesystem** specifications allow the user to define:

   a. The directory for storing user-defined keyword lists, to be used, for example, to create printable reports or to filter the fits header displayed in the **File Detail** panel in the main screen. The default value is ${HOME}/gasgano/keywords;

   b. The directory for storing defined scripts or executables external command (see Chapter 5). The default value is set to ${HOME}/gasgano/scripts, under the account's ${HOME} directory;

   c. the name of the file which contains the rules to filter the files in the **Navigation Panel;**

the default file is filter.rul, located into the account's ${HOME}/gasgano/config directory;

d. The name of the file that contains the classification rules for the raw files. It is defaulted to ${HOME}/gasgano/config/VLT.rul.

e. The recognised text files extensions; it is defaulted to ascii;paf;log;longlog, but as many extensions as desired can be added. Entries should be separated with a semicolon.

Example: in the default case, the tool recognises pippo.ascii as a simple text file. When the file is loaded and selected, its content is displayed in the **File Detail** panel. The file pippo.txt will not be recognised and therefore not loaded (will be ignored) by the tool.

′ When the text file extensions list is modified, a manual refresh of the data set is required (Click on **File>Refresh** in the Menu bar). If an extension is deleted from the list, the change takes effect the next time the tool is started. In other words, the files with this extension are not automatically unloaded by the tool.

′ As already mentioned, *Gasgano* recognises file types based on their file extension. For fits files these extensions are .fits (2-D frame) and .tfits (table). It is not possible to configure the file extensions for fits or tfits files. Therefore, the fits file pippo.rtd will not be recognised by the tool. The user should take care that FITS and TFITS files use the default extensions (.fits and .tfits, respectively).

❏ The bottom section deals with miscellaneous settings:

a. The **shorten data files paths for commands** checkbox specifies whether or not the application should try to determine a common path for all data files supplied to an external command, and execute a 'cd' to that path prior to executing the external command itself. This can be useful to overcome shell limitations in the maximum amount of characters permitted on the command line when calling an external application or script. If the user selects a set of files with no common path, the file names are not shortened.

b. The **Automatic Filesystem Refresh** sets the frequency for a refresh procedure: the tool scans the loaded directories and verifies whether a file has been modified or whether files have been added/removed from a directory in the current data set. The possible settings are: **No update** (default), **1,2,5,30 minutes**. This setting is typically used in the telescope online data flow system.

′ This option should be turned off when you have loaded a large number of files, since the refresh action freezes the application for a certain amount of time, during the refresh.

c. The **Verbose Level** option is used for debugging purposes only and specifies which debug level should be monitored while the application is running. This option should be set to **No Verbose**, which is the default value. For more information about the meaning of the other verbose levels, refer to Appendix C of this manual.

′ Whenever a **Verbose Level** is activated, diagnostic messages are written to stdout. In order to see Gasgano messages on stdout, the application must be run with the "-

debug" option. If this is not used, all messages are redirected to /dev/null. When troubleshooting, your ESO technical contact may ask you to activate one of the verbose levels and report these messages.

′ It is recommended to start *Gasgano* with "gasgano –debug > my_log_file 2>&1 &" if you want to reproduce an error. This command saves all messages from stdout and stderr in the file my_log_file, which can be used for troubleshooting

4.1.2 The File Display & Grouping tab

The **File Display & Grouping** tab, shown in Figure 11, contains settings for the **Navigation Panel** in the **Main Screen**, settings for the printer options and for configuring the file viewers.

The configurable items in this register are:

1) **Filesystem – Display Keywords** editable list: keywords in this list are displayed in the table columns in the **Navigation Panel.** The display follows the ordering in the list. The following keywords may be listed:

   a) All 8 character standard fits and ESO keywords: Example: RA, DEC, EXPTIME, ORIGFILE;

   b) HIERARCH ESO specific keywords, without the HIERARCH ESO prefix. Example: INS.FILT1.ID, TPL.START, INS.GRIS1.NAME;

   c) *Gasgano* meta-keywords, which are defined by the tool only but are available for display purposes: CLASSIFICATION, FILE_PATH, and PIPE_PRODUCT. More on these keywords and their meaning in Section 6.4.

   ′ CLASSIFICATION is, among the meta-keywords, the most useful to display: it represents the classification of a raw science/calibration file done by the tool according to the classification rules, which are fully configurable by the user. More on classification rules and how to configure them in Section 4.2.

To enter a keyword, type its name inside a field in the table. The buttons **Insert Row** and **Delete Row**, allow the user to add more rows (a scroll bar appears on the right side, when there are more rows than those which fit in the field) or to delete a selected row. The keyword names are case sensitive (must be uppercase) and the tool will mark the field as ∗INVALID∗, whenever this rule is not respected. There are no syntax or validity checks, therefore if a wrong or non-existent keyword name is typed in, no error message is displayed. The order of the keywords in the list is the order in which they will be displayed in the **Navigation Panel**.
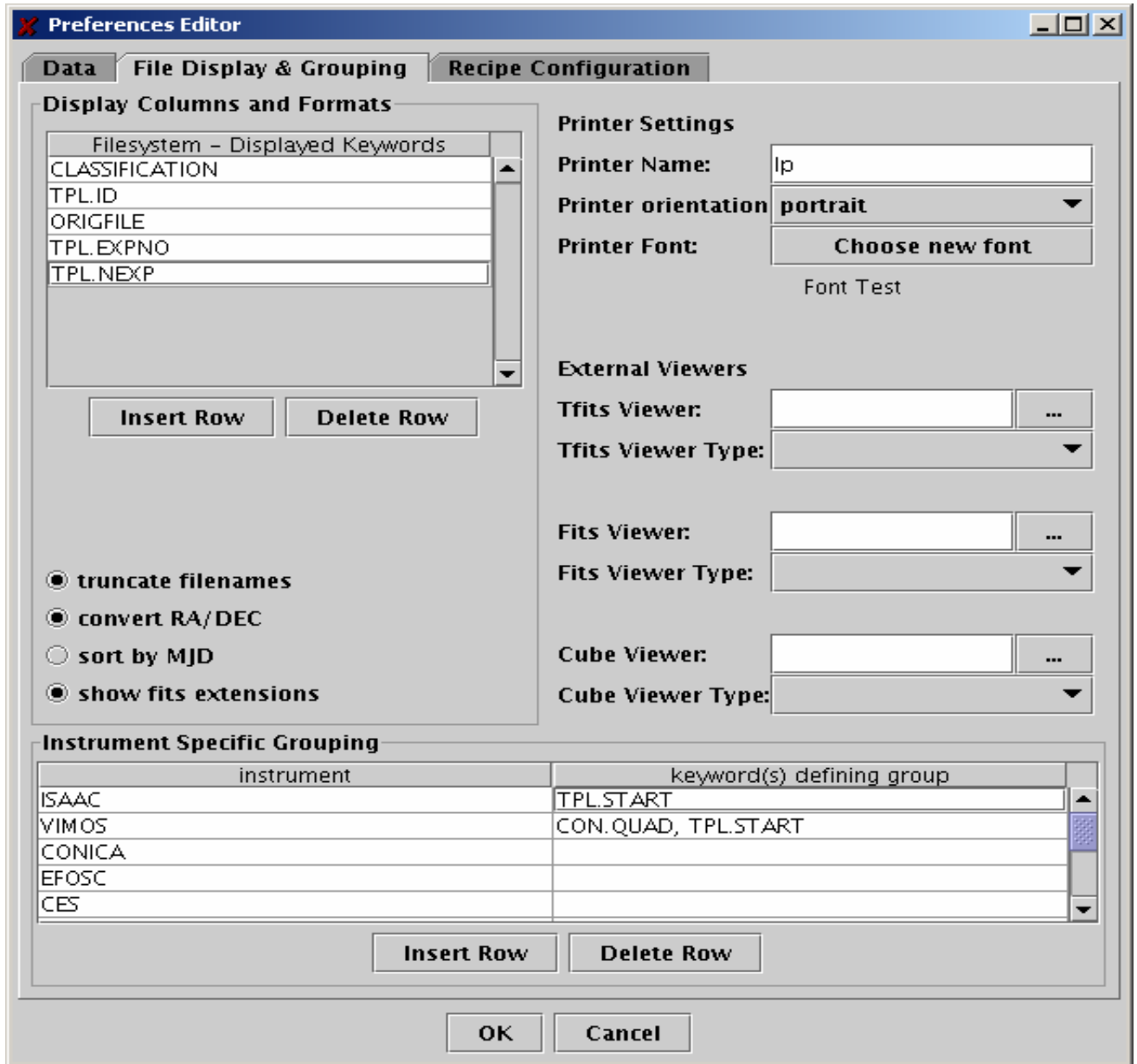
Figure 11: The File Display & Grouping tab in the Preferences Menu

♦ To add a keyword above an existing one in the list, right-click on the **Insert Row** button.

  2) **Printer Settings** situated on the right half side of the panel, include:

    a) **Printer Name**, with no default value. Specify in this field the "Unix" name of your printer; e.g. ppbla51.

    b) **Printer orientation**, default value is portrait. It can be set to Landscape.

    c) **Printer Font,** default value is Monospaced, 8pts. The chosen font can be checked interactively as it is selected (Font Test). To change the font type, size and characteristics, click on the **Choose new font** button.

3) **The External Viewers** setup fields, located below the printer setup fields, allow specifying the UNIX command to launch file viewers. Three different viewers are possibly configurable:

    a) **Tfits Viewer**, launched by the application whenever a single tfits file is selected;

    b) **Fits Viewer,** launched by the application whenever a single fits file is selected;

    c) **Cube Viewer**, launched by the application whenever more than one fits file is selected;

It is responsibility of the user to define calls to external viewers. No viewers are delivered with *Gasgano* and the three fields are empty by default. It is possible to specify a viewer in two ways:

    I. Type the name of the application as you would do from a terminal command line. The path to the application you plan to use should be in the ${PATH}. Example: skycat. The "&" is not allowed in the field and therefore *Gasgano* will freeze while the viewer is running. In order to be able to load one image after the other, by clicking on the filename in the application panel, it is recommended to use method II).

    II. You call an external shell script, which in turns calls the correct application and let it run in the background.

There are a set of example viewer scripts under the ${HOME}/gasgano/bin directory that may be used. These scripts are not viewers; rather they simply provide scripting around possible standard viewers (ds9, skycat, rtd, etc.). The scripts may require some modification to run.

The **View Type** combo box should always be set to **SHELLCOMMAND** unless the **Viewer** is SkyCat. If the **Viewer** is SkyCat, the viewer command can be set to **SKYCAT** and Gasgano will start and connect to SkyCat through a SkyCat specific TCP/IP interface.

4) The **truncate filenames** toggle on/off button allows the user to specify whether the filenames are displayed with their complete path or not in the Navigation Panel view. Default is **on**.

5) The **convert RA/DEC** toggle button allows the user to specify whether the entries for right ascension (RA) and declination (DEC), if present in the **Displayed Keywords** list, should be displayed in sexagesimal or in decimal notation. Default value is **on, i.e**. sexagesimal.

6) The **sort by MJD** toggle button, forces the application to display files sorted by the content of the keyword MJD-OBS. This option is useful when the files have been renamed following a user defined code, which would sort them in a different order than that of the UT time of observation. Default value is **off**.

7) The **show fits extension** toggle button, allows to display the FITS extension headers. Note that this is only possible wit not compressed FITS files.

8) **Instrument Specific Grouping** allows configuration of the tree view in the **Navigation Panel** by adding an additional grouping level to files. The table contains two columns: the Instrument column lists all available instruments and the criteria column - both are editable. The **Instrument** column should contain the name of the instrument as specified by the INSTRUME keyword in the header. The **Criteria** column should contain a list of keywords separated by semi-colons (e.g. OBS.NAME; TPL.START) that will be used to group together all the files having the same value for the specified keywords.

′ For many instruments the use of TPL.START is useful whenever an OB contains several templates. TPL.START is identical for all files belonging to the same template instance, and it is set to the UT starting time of its execution. For FORS1 and FORS2, in case of observation with many different masks, grouping by means of MASK.ID, might prove useful. For VIMOS the OCS.CON.QUAD will group files by quadrant. In addition to (or instead of) grouping, the file set may be filtered, described in section 3.1.3.

4.1.3 The Recipe Configuration tab

The Recipe Configuration tab lists all CPL recipes known to Gasgano. The recipes listed here are displayed as menu items under the **To Recipe** sub-menu of the **Selected Files** menu.
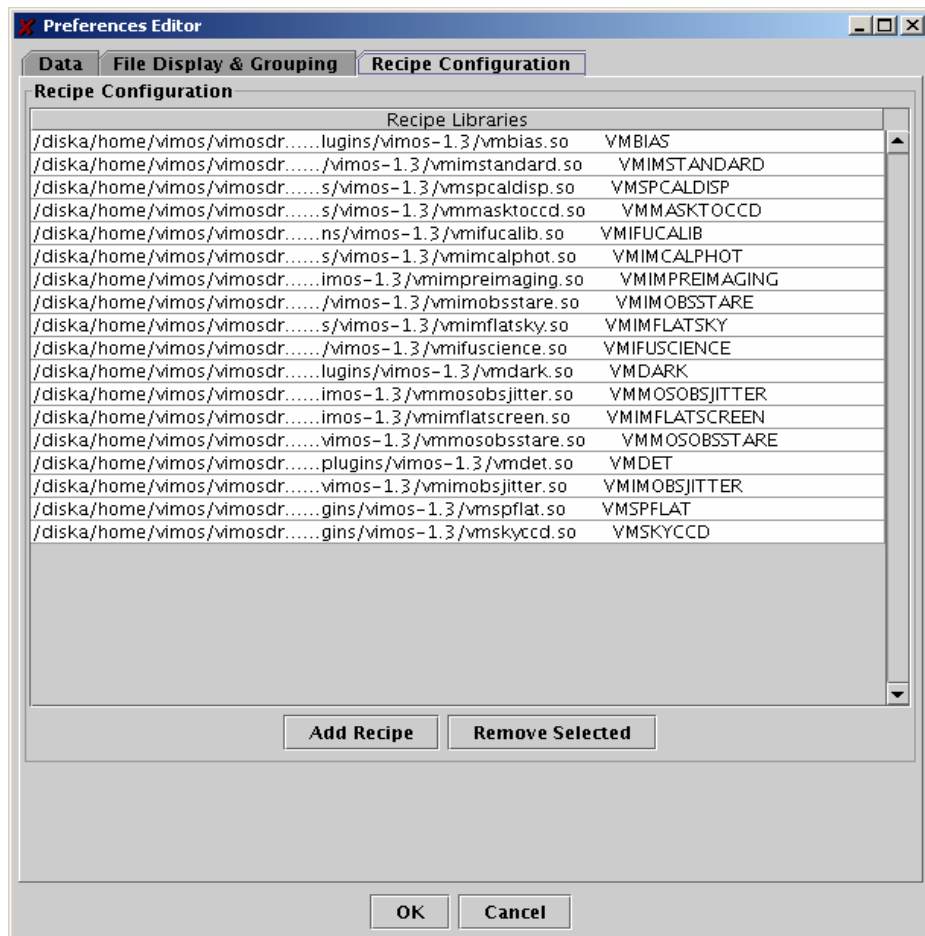
Figure 12: The Configure Recipes Preferences **tab**

The figure above (Figure 12) shows the Recipe Configuration tab with a set of VIMOS recipes selected. The table shows the paths to the CPL recipe libraries and the name of the recipe as it will appear in the menu.

Modification of the set of recipes in this list requires the application to be restarted before the recipe list is reflected in the menu.

1) The **Add Recipe** button displays a file chooser for selecting one or multiple CPL recipe files (*.so dynamic link libraries). Once selected the recipes will be opened to determine if the files are CPL recipes of the supported version. Opening a non-CPL recipe can have unpredictable results and you are warned of a possible application crash before opening the selected *.so files.

Once one or more selected CPL recipes have been opened, the **Library Contents** Panel will be presented, showing the recipe names, version, description, and so forth. See figure 13 below. Pressing **OK** on the **Library Contents** Panel will add the selected CPL recipes to the **Recipe Configuration Panel**.

The **Remove Selected** button removes the highlighted CPL recipes from the **Recipe Configuration** panel.
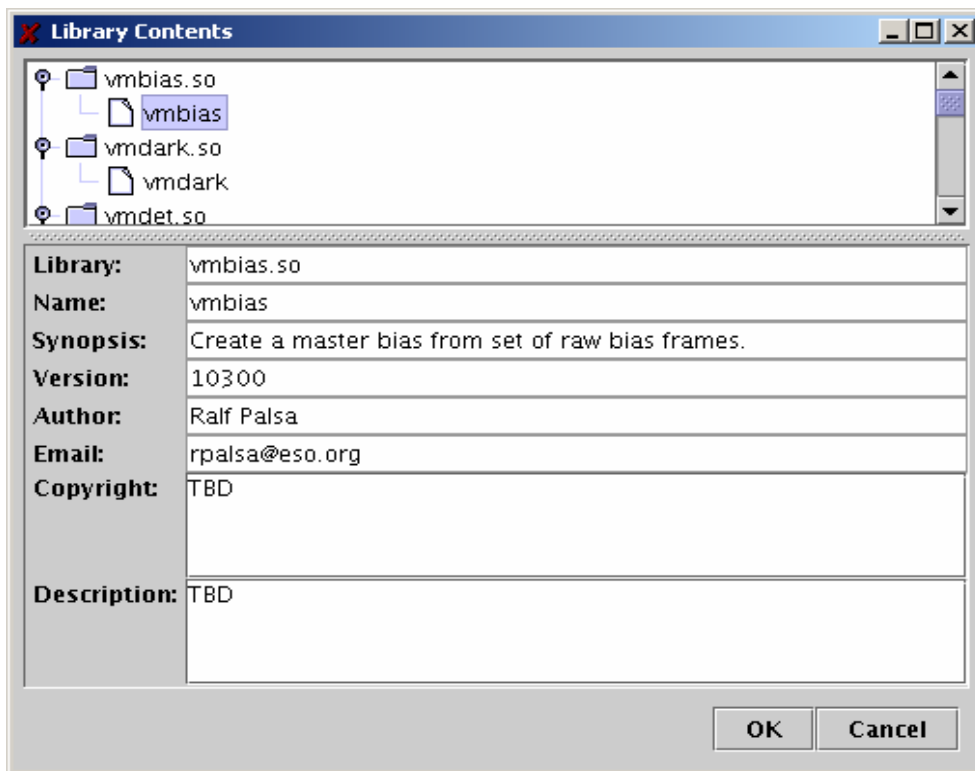
Figure 13: The Library Contents Panel **showing a set of selected CPL recipes with vmbias selected.**

4.2   Editing the classification rules

Gasgano has means to classify files based on keyword expressions. Classification determines

the type of a file based on keyword values stored in the primary header. A files classification is stored in the CLASSIFICATION MetaKeyword and is useful for data organization as well as for identifying the file type to reduction recipes. The Recipe Panel uses the CLASSIFICATION MetaKeyword to identify file types to the CPL recipes.

Outside of recipe execution, classification of files according to user specified rules might be useful for quick look assessment of a data package and for data reduction purposes.

Gasgano classifies files as they are loaded, comparing the value of specific keywords in their headers with the rules specified in the file ${HOME}/gasgano/config/VLT.rul (by default). The location of this file can be changed via the **Data** register of the **Preference Editor** (see subsection 4.1.1). The rules are applied in the order they are written in the classification rule file. If there is any conflict between rules, identical or similar (intersecting) rules, the first matching rule in the file has the precedence. *Gasgano* is distributed to the users with a default classification rules file, which can be edited to meet the user's specific needs. This file contains basic classification rules for all the available VLT instruments and it will be updated every time a new instrument becomes operational (available in the distribution kit). In addition, instrument specific classification rules files are also distributed. Example: ISAAC.rul, UVES.rul, etc.

The use of the default files is encouraged for all first time users. More experienced users who wish to modify the existing rules to create their own sets of rules are encouraged to read sections 4.2.2 and 4.2.3.

If the user foresees working with files produced by only one single instrument, it is advisable to configure *Gasgano* to use the appropriate instrument-specific classification rules file, instead of the generic one. The classification rules supplied with *Gasgano* are similar to those used by the ESO data reduction pipeline, therefore tailored to an automatic data reduction strategy for large amounts of data.

′ If using ESO supplied reduction recipes, the classification rules supplied with the software are designed to match the classifications expected by the recipe. This should be noted when planning to modify the classification rule set.

′ The classification rules file is not in ASCII format and therefore can only be edited within the tool, by means of the classification rules editor. To access the file click on the **Menu Bar Tools>Classification rules…**. The **Classification Rule Builder** window appears, as shown in Figure 14.

The window is divided into two sections, the **Classification Rules Panel** (with the list of existing rules) in the upper part and the **Expression Builder** panel in the lower part. Along the bottom, a set of buttons allows maintenance of the list of rules. A rule selected from the **Classification Rules Panel** is displayed in the **Expression Builder**, where it can be edited. For each rule, an Instrument and a Classification label must be specified. The Instrument label tells the tool to apply the rule to all the files that share the same keyword INSTRUME; the Classification label specifies the classification to assign a file matching that rule (stored in the files CLASSIFICATION MetaKeyword).

The upper panel consists of a table with three columns: The **Instrument** and **Classification**

columns display the content of the above mentioned labels. The **Rule** column displays the classification criteria.

The lower panel allows viewing, creation and editing of classification rules. The rules are graphically rendered in a sort of **RPN (Reverse Polish Notation).** In short, RPN is a way of expressing arithmetic and/or logical expressions avoiding the use of brackets to define priorities for evaluation of operators. Here are some examples (not all of them are examples taken from the supplied classification rules files):

1) <u>The simplest rule</u>. The simplest rule classifies frames using a one keyword classification criteria:

   ISAAC   TWFLAT   TPL.ID="ISAACSW_IMG_CAL_TWFLATS"

   Where ISAAC is the Instrument, TWFLAT is the Classification label and the Rule states that if in the header of a fits file the keyword TPL.ID (observing template identifier) is equal to ISAACSW_IMG_CAL_TWFLATS, then this file must be classified as TWFLAT (twilight flat). In the lower panel the rule will be displayed simply as:

   TPL.ID="ISAACSW_IMG_CAL_TWFLATS"

Figure 14: The Classification Rules Builder Panel. **Some of the rules for the ISAAC instrument are displayed in the upper panel. The lower panel (**Expression Builder Panel**) shows the classification rule for files of the type JITTER_OBJ. This rule is explained in more detail in the following text.**

2) <u>Double condition rule.</u> In this case a double condition on one or two keywords is required to classify a file:

AND
TPL.ID="ISAACLW_IMG_CAL_DARKS"
DET.MODE.NAME="UncorrHighBias"

which classifies an ISAAC file as LW_Uncorrelated_Dark, whenever both conditions are satisfied. Note that AND is referred to as a **Node** and

TPL.ID="ISAACLW_IMG_CAL_DARKS" is called a **Sub Expression.**

OR
ORIGFILE like "dumpRtd*"
ORIGFILE like "*_acq_Image*"

which classifies a file as Acquisition image whether its ORIGFILE (Original filename) keyword contains the string dumpRtd* or the string *_acq_Image*.

3) Triple (multiple) condition rule. The simplest example for this rule is to check if three (or more) conditions are all met or one of three (or more) condition is met:

AND
DPR.CATG="CALIB"
DET.CHIPS=1.0
DPR.TYPE="DARK"

which classifies a UVES file as a DARK_BLUE.

OR
OBS.PROG.ID like "60.A-9023*"
EXPTIME < 60
INS.FILT1.ID != "V_BESS"

which classifies a file as FORS1 frame that was either observed under the engineering standard Run ID or its exposure time is less than 1 minute or its filter is not a V_BESS (this example is not in the distributed version of the classification rules).

4) Complex rules. These rules include combinations of nodes (AND and OR) and multiple sub expressions (conditions on keywords). An example is shown below:

AND
   DPR.CATG="SCIENCE"
   DPR.TYPE="SKY"
   DPR.TECH="MOS"
   OR
      TPL.ID="VIMOS_mos_obs_Stare"
      TPL.ID="VIMOS_mos_obs_jitter"

This example classifies matching files as MOS_SCIENCE frames taken with VIMOS. The corresponding expression in "algebraic" notation would be:

```
(DPR.CATG="SCIENCE")&(DPR.TYPE="SKY")&(DPR.TECH="MOS")&((TPL.ID="V
IMOS_mos_obs_Stare")OR (TPL.ID="VIMOS_mos_obs_jitter")).
```

Following is another example.

```
OR
   AND
        TPL.ID="ISAACSW_img_obs_AutoJitter
        DPR.TYPE="OBJECT"
   AND
        TPL.ID="ISAACSW_img_obs_AutoJitterOffset
        DPR.TYPE="OBJECT"
```

The rule classifies as JITTER_OBJ frames taken with ISAAC. Again, the corresponding expression in "algebraic" notation would be:

```
((TPL.ID="ISAACSW_img_obs_AutoJitter)&(DPR.TYPE="OBJECT")) OR
((TPL.ID="ISAACSW_img_obs_AutoJitterOffset)&(DPR.TYPE="OBJECT")
```

The same classification rule can be written in different ways. In order to aid readability, it is recommended to use the logical expression with the least possible number of constraints. The higher this number, the slower will be the tool in classifying files.

′ It is the responsibility of the user to ensure that there are no contradicting rules. The tool applies the rules for file classification in the order in which they have been specified in the classification rules file. The first matching rule is applied.

′ You may need to use a rule that is a subset of another one. For example:

```
AND
   TPL.ID="ISAACLW_IMG_OBS_AUTOJITTER"
   DPR.TYPE="OBJECT"
```

and

```
AND
   TPL.ID="ISAACLW_IMG_OBS_AUTOJITTER"
   DPR.TYPE="OBJECT"
   SEQ.CUMOFFSETX = 0
```

If this order is maintained in the classification rule file, the second rule will never be applied. Inverting the order of the rules ensures that both will be applied. When one rule is a subset of another rule, always put the most specific or restrictive rule on top.

4.2.1 Changing the order of the rules in the classification rules file

To change the order of the rules in the classification rules file, use the two **Move Up** and **Move Down** buttons at the bottom of the window: first select the rule that has to be moved and then click on the button so many times as many position changes are required.

4.2.2 Editing an existing rule – Advanced users

The easiest way to configure a personal set of classification rules is to modify the rules distributed with the tool: the file VLT.rul contain rules for all instruments of the VLT (commissioned and operative at the time of this release), while ISAAC.rul, FORS12.rul and UVES.rul contain the individual classification rules.

' It is possible to append a classification rule file to another one. Click the **Menu File>Append other rule file** in the **Classification Rule Builder.** The browse window opens. Find and select the file, click on **Append.**

Suppose you want to modify the following rule:

AND
    TPL.ID="ISAACLW_IMG_CAL_DARKS"
    DET.MODE.NAME="UncorrHighBias"
into

OR
    TPL.ID like "*Jitter*"
    TPL.ID notlike "*Generic*"

1) To edit the node AND, double click on it (alternatively single click and then click on the leftmost button EDIT). A window with a toggle button appears from which you can choose between the AND/OR option. Select OR. Confirm with OK.

2) Double click on the first term of the logical expression, and a window appears (see Figure 13). On the top part there are two fields to be edited with a toggle button in between. The first field contains the keyword, as seen in the header (without the HIERARCH.ESO part) of ESO standard fits files. Type in it TPL.ID. The syntax for keywords is case sensitive and requires dots in between strings.

3) Select with the toggle button the field "like"

4) Type in the leftmost field: "*Jitter*"

5) Confirm with OK.

6) Repeat steps 2) –5) to edit the second sub expression.

As a safety feature, whenever an incorrect value is inserted in the field and *OK* is pressed, the content of the field is reset to the previous value.
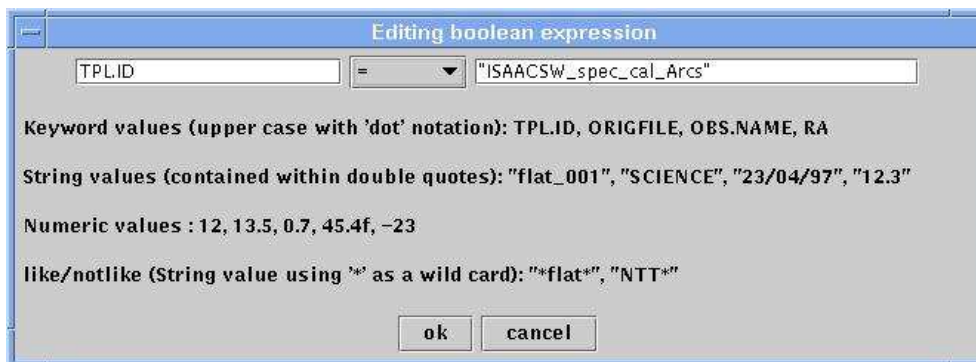
Figure 15: Editing Boolean expression window. Syntax and examples are explained in the text below the editable fields.

In the **Editing Boolean expression** window, there are some basic examples on how to type the rules.

′ Ensure all strings are enclosed in double quotes.

′ Use '*' for a wild-card, but only with Like and Not Like.

′ Do not enclose numeric values in quotes.

It is possible to add a node after a sub expression, simply selecting the sub expression and clicking on the **Insert Node** button. Vice versa, it is possible to add a sub expression under a node, clicking on the node and then on the **Add Leaf** button. If an operation is not allowed (like adding a node under another node) the corresponding button is will be disabled.

Once the rule is modified, you must submit it to the file using either the **Add Rule to table** or **Replace Selected Rule** button.

Before adding or replacing the rule, be sure of checking that the **Instrument** field is correct.

′ If you add a rule before changing its classification, or, in other words, you have two rules with the same name, you can select one and delete it via the **Delete Selected Rule** button.

′ What happens if there are two identical rules or two rules with the same name? Not much, really! The second rule is just redundant information, since the tool classifies the files using the rules in the order they are specified in the classification rule file. The second instance of the rule will therefore never be applied.

4.2.3 Creating a new rule – Advanced users

If the application is started with no classification rules file (you will be notified through a warning pop-up on start), the **Classification Rule Builder** window will be empty. To create a new rule, follow this step by step sequence:

1) Fill in the **Instrument** field. E.g.: ISAAC

2) Fill in the **Classification** field. E.g.: SCIENCE_imaging

3) Click on the rule editor panel, the KEYWORD="value" string appear. Either

   a) Click on the **Edit** button and edit the rule (*double click* on the string is also possible).

   or

   b) Click on the **Insert Node** button.

   Case (a) applies when one single constraint rule needs to be edited, whereas case (b) allows adding a node with two sub expressions.

   To edit and modify the nodes and/or the sub expressions proceed as described in the previous section.

4) Click on the **Add Rule to table** button.

Repeat steps 1)-4) until all the rules have been added.

4.2.4 Closing the Classification Rule Builder window

Once modifications are complete selecting **Dismiss** under the **File** menu will automatically save all changes in the rule file. The location of the rules file is stored in the **Preference, Data** register.

Once the window has been closed, the changes can be applied to the file set in *Gasgano*. This is done by selecting **Reclassify** under the **File** menu of the *Gasgano* main window.

4.3 The .gasganorc **file**

The .gasganorc file, if not present, is created in the $HOME directory when the tool is launched. The first version of this file contains default settings, which the user can modify using the **Preferences** panel, as explained in the previous sections.

Every time the set up of the tool is modified, by means of the **Preference** panel, the corresponding setting is saved into the .gasganorc file. This is a standard ASCII file, containing a list of entries, with the format:

```
KEYWORD = value
```

E.g. `CLASSRULE_FILE=/my_gasgano_path/config/UVES.rul` specifies the name of the classification rule file.

The list of all the possible entries in the .gasganorc file and their meaning is given in Appendix B.

It is recommended to modify the preferences from within the tool, instead of editing manually the .gasganorc file.

5. RUNNING EXTERNAL EXECUTABLES

This chapter covers the various features available in *Gasgano* to pass selected file as parameters to external programs.

This chapter is intended for users who are familiar with the basic use of *Gasgano* and wish to exploit the tool further.

5.1 Running external executables

A set of selected files can be sent as command line parameters to a user selected executable. The files selected in the **Navigation Panel** are passed as command line parameters, one file per parameter, with the full path name. On demand, the full path may be dropped and the external command executed after a `cd' into the common directory of all selected files (This requires that all selected files are stored in the same directory.). In order to drop the path, activate "shorten data files paths for commands" button in the **File>Preferences Data** register.

There are two ways to send selected files to a customized program:

1. Select one data set and use the option **Run** from the **Selected Files** menu in the **Menu bar**. A browse window appears. Locate the script, select it and click on **Run** (or double click on the script name to execute). Click **Ok** in the confirmation window.

2. Use the **Script Board**. This option is recommended when instrument dependent external commands have to be executed frequently.

5.2 The Script Board

The Script Board is an additional window, which shows a set executable files selected from a user-definable directory. The **Script Board** is activated by clicking on the **Script Board** button (the third from the left) on the **Toolbar,** or by selecting **Script Board** from the **Tools** menu, or by typing Ctrl-s when the mouse pointer is on the main *Gasgano*'s window. The window can be left open while using *Gasgano*.
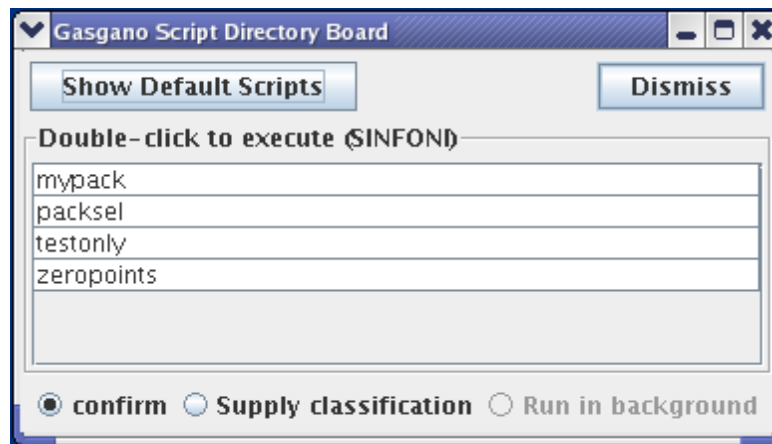
Figure 16: The Script Board **showing four scripts**

To execute a command one has to double-click on the desired script in the **Script Board**.

The **confirm** option will trigger a pop-up window in which one additional confirmation needs to be given before being able to run the specified executable.

On the Script Board the **Supply classification** option will pass the CLASSIFICATION Meta-keyword along with each file name. The sequence passed to the script is of the kind:

```
Filename1 CLASSIFICATION1 Filename2 CLASSIFICATION2 …
```

The **Run in background** option is disabled in the present version of *Gasgano*.

The **Script Board** automatically looks for the executables in a directory (the **common script** directory. This directory cannot be a link to another directory) specified using the **Preferences Editor** in the Data register. When the **Script** Board is open and a data set (one or more files) is selected, *Gasgano* checks if all files share the same INSTRUME keyword. If this is the case, the **Script Board** looks into an instrument specific directory, specified as a subdirectory of the script directory named after the INSTRUME keyword itself. A typical example situation looks as follows:

```
<Gasgano-script-dir>/files common to all instruments (if any)

<Gasgano-script-dir>/ISAAC/files for isaac....

<Gasgano-script-dir>/FORS1/files for fors1....

<Gasgano-script-dir>/EMMI/1.57/files for EMMI release 1.57....,
```

where <Gasgano-script-dir> is the **common script** directory set in the **Preferences**.

If the selected files do not share a common INSTRUME keyword or the instrument specific directory does not exist, *Gasgano* shows the files in the <Gasgano-script-dir> directory and the **Show Default Scripts** button is hidden.

The **Show Default Scripts** button on the top left corner of the **Script Board** clears the instrument specific script list and reloads the view with the scripts in the **common script** directory. In this case the label **Show Default Scripts** changes in **Show** INSTRUME **Scripts**, and clicking the button reloads the instrument specific scripts.

5.3   Passing a directory path to an executable

It is possible to pass the path of a directory (loaded into the application) on the file system to an executable. To do this:

1) Select the **Group by directory** view on the **Tool bar.** In the **Navigation** Panel a collapsed list of folders with the directories names (paths) should be visible;

2) Click and highlight the desired directory.

3) Double click on the desired script in the **Script Board.** If the **confirm** option on the board is active the confirm window should give the message:

```
/{gas-script-dir/../your_script /path_of_the _selected directory
```

5.4   Example scripts

Gasgano can run executables, in the sense that it passes a list of parameters (the files currently selected in the **Navigation Panel**) to an external application. The executable may be anything (a shell script, a C program, etc) that can be run from the command line. It is responsibility of the user to create executables that handle the list of parameters (file names and/or classification string) in a correct way.

In all the following examples of scripts, substitute mypath with the appropriate path to the executable in your file system.

Examples:

1. <u>Parameters passed to an executable</u>: when files are selected and passed to an executable *Gasgano* may sends three different outputs. Consider the script that saves the list of parameters sent by *Gasgano* to the file /tmp/output.txt

```
##!/bin/sh
#
# Example shell script to be used with Gasgano:
# Print output - save to output.txt the list of parameters
# sent to an executable.
#-----------------------------------------------------------
echo $* > /tmp/output.txt
```

Assume that the following files are selected in the **Navigation Panel**:

```
ISAAC.2001-06-08T22:28:22.108.fits
ISAAC.2001-06-08T22:28:47.361.fits
ISAAC.2001-06-08T22:29:12.282.fits
ISAAC.2001-06-08T22:29:38.666.fits
ISAAC.2001-06-08T22:30:03.865.fits
ISAAC.2001-06-08T22:30:28.985.fits
ISAAC.2001-06-08T22:30:54.343.fits
```

1) Per default files are passed as a list of names with full path separated by a space:

```
/diska/test/ISAAC.2001-06-08T22:28:22.108.fits
/diska/test/ISAAC.2001-06-08T22:28:47.361.fits
/diska/test/ISAAC.2001-06-08T22:29:12.282.fits
/diska/test/ISAAC.2001-06-08T22:29:38.666.fits
/diska/test/ISAAC.2001-06-08T22:30:03.865.fits
/diska/test/ISAAC.2001-06-08T22:30:28.985.fits
/diska/test/ISAAC.2001-06-08T22:30:54.343.fits
```

Even if the files are shown in the **Navigation Panel** without the path (the option **truncate filenames** in the **File Display & Grouping** tab of **Preferences** is turned on); the path is passed to the executable.

2) In case the **Supply Classification** toggle button on the **Script Board** window is turned on, each file and its classification separated by a space are passed as a list:

```
/diska/test/ISAAC.2001-06-08T22:28:22.108.fits TWFLAT
/diska/test/ISAAC.2001-06-08T22:28:47.361.fits TWFLAT
```

```
/diska/test/ISAAC.2001-06-08T22:29:12.282.fits TWFLAT
/diska/test/ISAAC.2001-06-08T22:29:38.666.fits TWFLAT
/diska/test/ISAAC.2001-06-08T22:30:03.865.fits TWFLAT
/diska/test/ISAAC.2001-06-08T22:30:28.985.fits TWFLAT
/diska/test/ISAAC.2001-06-08T22:30:54.343.fits TWFLAT
```

3) In case the **Supply Classification** toggle button on the **Script Board** window is turned off and the **shorten data files paths for command** toggle button in **Preferences, Data** register is set to off, files are passed as a list separated by spaces:

```
ISAAC.2001-06-08T22:28:22.108.fits ISAAC.2001-06-
08T22:28:47.361.fits ISAAC.2001-06-08T22:29:12.282.fits
ISAAC.2001-06-08T22:29:38.666.fits ISAAC.2001-06-
08T22:30:03.865.fits ISAAC.2001-06-08T22:30:28.985.fits
ISAAC.2001-06-08T22:30:54.343.fits
```

2. <u>A simple parser for the list of parameters</u>: this shell script expects as input a list of file names from *Gasgano* (it assumes the option **Supply Classification in the** Script is turned off). The file should be saved in the **common script** directory (e.g. call it `crea_list_of_frames`). When executing, select the files in the **Navigation Panel** and double click on **crea_list_of_frames** in the Script Board. If the **Script Board** is positioned on an instrument specific script directory, use the **Show Default Sctipts** button.

```
#!/bin/sh
#
# Example shell script to be used with Gasgano:
# crea_list_of_frames.sh - creates a file framelist.ascii to
# be used with eclipse recipes
#-------------------------------------------------------------
# move to the working directory
GAS_TMP_DIR=/mypath/Gasgano
cd $GAS_TMP_DIR

#-------> Clean old stuff

if [ -s ${GAS_TMP_DIR}/framelist.ascii ]
then
        rm ${GAS_TMP_DIR}/framelist.ascii
fi

#-------> Format input for eclipse routine

list=`echo $*`

for i in $list
do
        printf "%s\n" $i >> framelist.ascii
done
```

6.   RUNNING CPL RECIPES

This section details the use and execution of ESO CPL recipes within *Gasgano*.

CPL recipes are ESO developed ISO-C data reduction libraries that may be loaded and executed by *Gasgano*.   For   more   information   on   CPL,   please   see http://www.eso.org/observing/cpl/.

Once configured in *Gasgano*, CPL recipes (here after simply recipes) may be loaded using the **To Recipe** menu under the **Selected Files** menu. Loaded recipes are displayed in a **Recipe Panel**.

6.1   The Recipe Panel

The **Recipe Panel** is a window launched from *Gasgano* that permits configuring and execution of a recipe.

The panel provides controls for setting input parameters, input frame sets, and product settings. In addition, the recipe may be executed and the products and log messages reviewed.

A **Request Pool** in the **Recipe Panel** allows you to build a series of requests containing (possibly) different sets input frames and parameters and add them to the Request Pool. Requests in the pool can then be selected and executed as a batch.

6.2   Opening the Recipe Panel

To load a recipe and open the Recipe Panel, select the recipe from the set of recipes displayed under the **To Recipe** menu of **Selected Files** of the **Recipe Panel**. If there are files selected in the **Navigation Panel**, these will be placed in the recipe **Input Frames** panel. If no files were currently selected the **Input Frames** panel will be empty.

The figure 17, below shows an example of the **To Recipe** menu and explains the action of the various menu items.

File **Selected files** Tools

Display...
To Recipe   ▶   vmimflatsky v10300
To Recipe Request Pool ▶   **Load Recipe**   ▶   vmbias
Report...   vmdark
Move...   vmdet
Copy...   vmifucalib
Tar...   vmifuscience
Run...   vmimcalphot
                                                                                     vmimflatscreen
                                                                                     vmimflatsky
Open Recipe Panels are listed here;   vmimobsjitter
this example shows one open   vmimobsstare
Recipe Panel (vmdet v10300).   vmimpreimaging
                                                                                     vmimstandard
                                                                                     vmmasktoccd
                                                                                     vmmosobsjitter
                                                                                     vmmosobsstare
All available recipes are listed here.   vmskyccd
                                                                                     vmspcaldisp
                                                                                     vmspflat
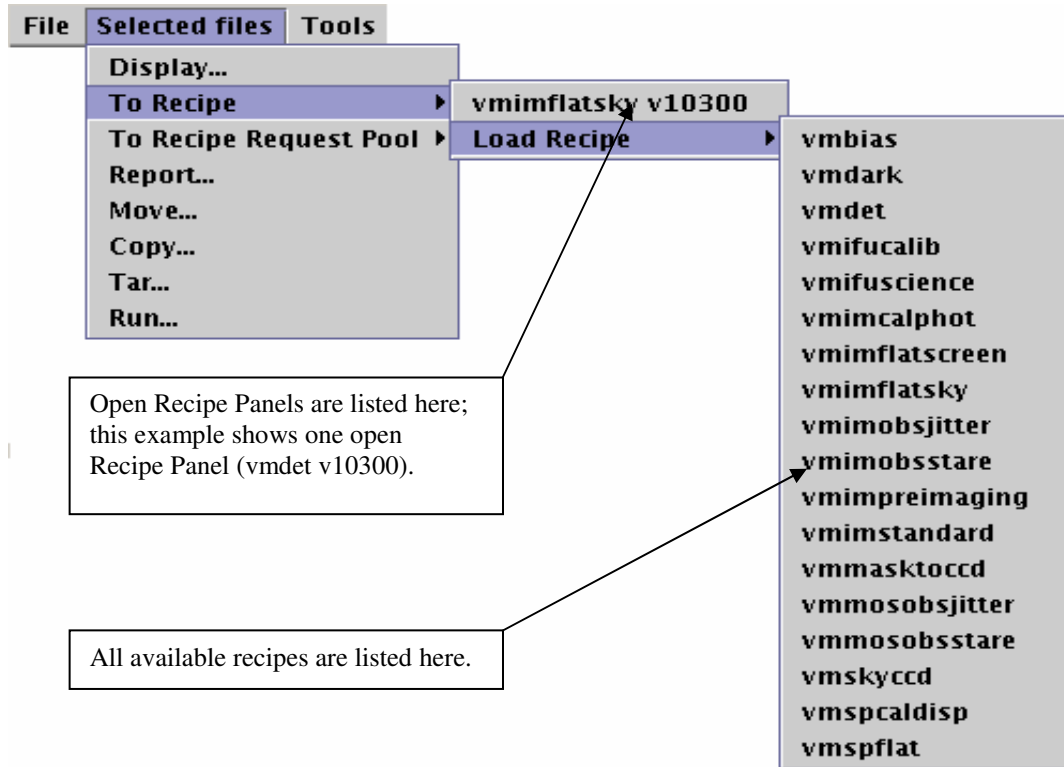
Figure 17: The To Recipe **menu, showing one open recipe (vmdet v10300) and the full set of recipes under the Load Recipe menu.**

The figure 18, below, shows an example Recipe Panel with the vmmosobstare recipe loaded. The key sections of the Recipe Panel have been labelled.
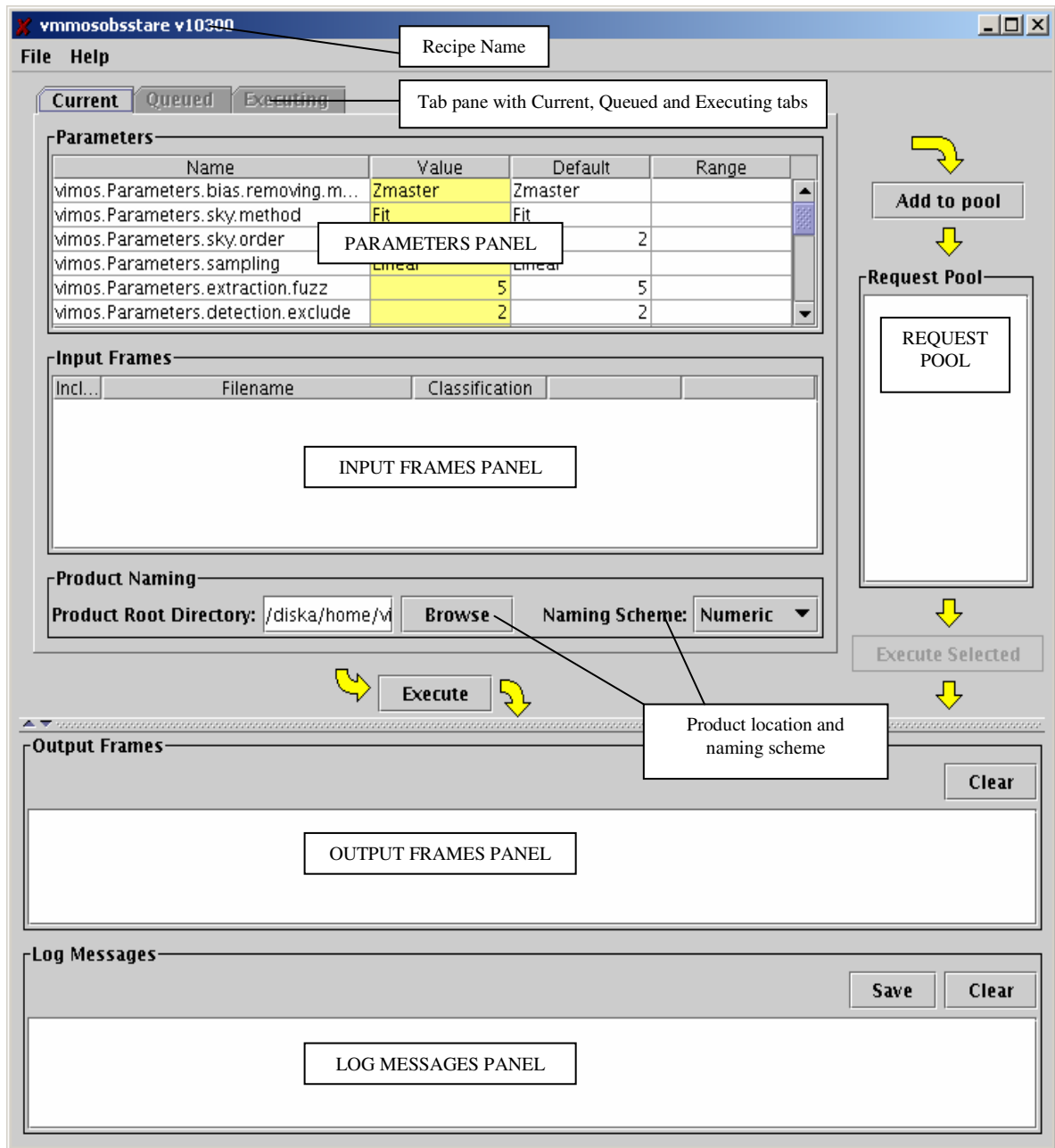
Figure 18: The Recipe Panel with the recipe vmmosobsstare loaded.

The **Recipe Panel** displays a tab pane with three different tabs: **Current**, **Queued** and **Executing**. The tabs display the various attributes of a recipe as follows:

- The **Current** tab is the base recipe settings and input frames. It is the data contained in this tab that is passed to the recipe when the **Execute Button** is pressed. The contents of the **Current** tab may be saved and loaded to/from a file through the **File** menu **Save Current Settings** and **Load Current Settings** menu items.

- The **Queued** tab displays a selected request from the **Request Pool**. When a request in the **Request Pool** is selected, it is loaded into the **Queued** tab and displayed.

' To switch from the Queued tab to the Current tab, simply press **Current** on the tab pane. The opposite is also true. Note that if the Request Pool is empty the Queued tab will be disabled.

- The **Executing** tab is automatically displayed when a recipe is executing. It shows the recipe parameters and input frames in a panel that does not permit editing. Once execution completes, the **Executing** tab will be replaced by the **Current** tab.

6.3   Parameters Panel

The **Parameters** panel contains the set of user settable recipe input parameters. Cells in the yellow column labelled "Value" maybe edited as required. The **Default** column contains the default value as specified by the recipe. Tools tips for cells in the Parameters panel provide a short description of the input panel.

For details on the meaning and impact of a parameter on recipe execution, please consult the recipe documentation.

6.4   Input Frames Panel

The **Input Frames** panel contains the set of input frames that will be used by the recipe. An input frame consists of a file and a classification. The type and number of input frames required by a recipe is described in the recipe documentation.

The classification is taken from the file classification as it appears in Gasgano. Modification to the file classification can be made directly by clicking and editing the yellow **Classification** cell of input frame.

The Input Frames panel provides **Locate** and **Display** buttons for locating an input frame within the *Gasgano* **Navigation Window**, and sending the input frame to an external viewer.

6.5   Product Naming Panel

The **Product Naming** panel contains settings for the product files generated by the recipe. The settings are: the **product root directory**, which determines the location of the product file, and the **naming scheme**. The **product root directory** is by default set to the current working directory. If in the .gasganorc file a property PRODUCT_DIR is defined and set to a valid directory, then its value is used as default for **product root directory** instead of the current working directory**.**

The naming scheme is a simple means to rename the product files. Available naming schemes are:

- **Numeric:** This scheme appends a four-digit number to the end of all product files, starting at 0000 and increasing. If a file of the desired name already exists in the product directory, the index is increased to the next number. This scheme insures that product files resulting from multiple runs of the recipe do

not over-write each other.

- **Overwrite:** This scheme performs no product file renaming and thus multiple runs of a recipe will result in overwriting the product file set on each run.

6.6   Output Frames Panel

The Output Frames panel contains the full set of products for each run of the recipe.

As with the Input Frames panel, the Output Frames panel provides **Locate** and **Display** buttons for locating a frame within the *Gasgano* **Navigation Window**, and sending the frame to an external viewer.

The Clear button will empty the list of product frames displayed. The product files are not actually deleted from the file system. The **Recipe Panel** (or in fact the *Gasgano* application) provide no means of removing files from the file system.

✐   Frames displayed in the Output Frames panel may be added to other Recipe Panels using *Drag and Drop* (see below).

6.7   Log Message Panel

All messages generated by the recipe are logged in the **Log Message** panel. In addition, prior to executing the recipe, the full set of input parameters and input frames are written to the **Log Message** panel.

As the execution of a recipe proceeds, new messages will be logged to the window indicating the progress of the recipe.

Upon completion of the recipe execution, the set of products will be listed along with the final status of the recipe execution. If any error messages were logged during execution, the status will be FAILED.

The **Log Message** panel may be cleared using the **Clear** button, alternatively the logs may be saved to a text file using the **Save** button.

✐   The full logs of each recipe execution are saved automatically in the designated product directory. These log files are named as follows: <recipe>_<execution date>.log. For example the log file: *vmimobsstare_2004-03-02_13:45:48.log* would contain the results of the execution of the vmimobsstare recipe from March 2nd 2004, 13:45:48.

6.8   Adding frames to the Recipe Panel

Files may be added to the input frames panel in a number of ways:

- *Drag and Drop* from the **Navigation Window**: by selecting one or more files in the Navigation Window and pressing (and holding) the left mouse button you can drag the selected files from the Navigation Window to the Recipe Panel. Once the mouse cursor is over to the **Input Frames** panel the mouse button is released. The selected files will be added to the input frames list along with their classification.

- Through the **Selected Files** menu: by selecting one or more files in the Navigation Window and then clicking the **To Recipe** item under **Selected Files** menu, the set of open Recipe Panels will be displayed, along with the option to open a new recipe. By selecting one of the existing Recipe Panels the selected files will be added to the input frame set of the selected Recipe Panel. Recipe Panels are listed under the To Recipe menu with the same names as appears in the title bar of the Recipe Panel. In Figure 18, above, this would be: "vmimflatsky v10300".

' The **Selected Files** menu can be displayed by pressing the _right_ mouse button on the Navigation Window once a file selection is made. The popup menu displayed is identical to the Selected Files menu in the menu bar.

' When multiple **Recipe Panels** are open, frames may be taken from one Recipe Panel and added to another using _Drag and Drop_. Again, select the desired input or product frame from one **Recipe Panel**, click and hold the left mouse button and drag the selection into the **Input Frames** panel of another **Recipe Panel**.

6.9   Executing Recipes

Recipes can be executed in one of two ways through the Recipe Panel:

1. Using the **Execute** button – execute the recipe using current parameters and input frames.

2. Using the **Request Pool** – build a pool of possibly different recipe execution requests and execute a selection of these requests in batch.

The **Execute** button and **Request Pool** are detailed in the following sections.

6.10   Using the Execute Button

The **Execute** button executes the recipe using the current parameters and input frames. The parameters and input frames will be displayed on the **Executing** tab during the execution and the Recipe Panel can not be used until the recipe execution completes. All logs and products are added to the **Output Frames** and **Log Messages** panels during execution.

The typical steps in using the Execute button are as follows:

1. Open the required Recipe Panel

2. Add the set of input frames.

3. Set the required parameters

4. Set the product directory

5. Press the **Execute** button

6. Review logs and products

7. Repeat if required, modifying parameter, input frame, and product settings.

6.11 Using the Request Pool

The **Request Pool** offers a means to build and edit a set of recipe execution requests, add the requests to a pool, and execute some or all of the pool in batch mode.

The **Request Pool** and its main controls are shown in the figure below.
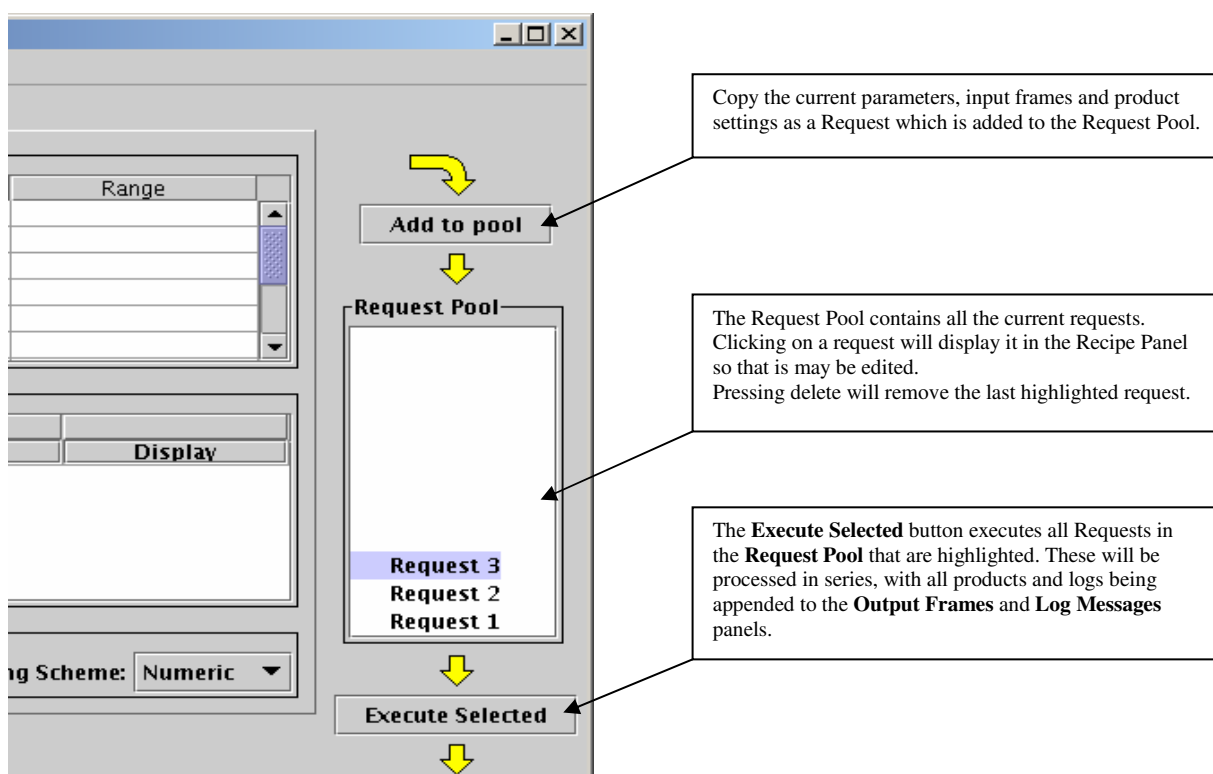
Figure 19: The Request Pool **showing three requests, with** Request 3 **selected.**

6.12 Adding Frames directly to the Request Pool

In addition to adding frames to the Input Frames panel of a Recipe Panel, frames may also be added directly to the **Request Pool**. When this is performed, the parameters and input frames of Recipe Panel are combined with the frames selected in the **Navigation Window** and the resulting combination is added as a new request in the **Request Pool**.

Frames may be added to the Request Pool in a number of ways:

- *Drag and Drop* from the **Navigation Window**: by selecting one or more files in the Navigation Window and pressing (and holding) the left mouse button you can drag the selected files from the Navigation Window to the Recipe Panel. Once the mouse cursor is over to the **Request Pool** panel the mouse button is released. The selected files will be combined with the current input frame set and parameter settings to create a new request in the **Request Pool**.

- Through the **Selected Files** menu: Frames may be added from the *Gasgano* **Navigation Window** directly into the **Request Pool** of an open **Recipe Panel** through

the **To Recipe Request Pool** menu under the **Selected Files** menu.

′ The **Selected Files** menu can be displayed by pressing the <u>right</u> mouse button on the Navigation Window once a file selection is made. The popup menu displayed is identical to the Selected Files menu in the menu bar.

• When multiple **Recipe Panels** are open, frames may be taken from one Recipe Panel and added to another using *Drag and Drop*. Again, select the desired input or product frame from one **Recipe Panel**, click and hold the left mouse button and drag the selection into the **Request Pool** of another **Recipe Panel**.

6.13   Using the Execute Selected button

Once a set of requests has been built and added to the **Request Pool** they may be selected and executed using the **Execute Selected** button.

All products and logs are appended to the **Output Frames** and **Log Messages** panel.

′ It is advisable to use the **Numeric** Naming Scheme (on the **Product Naming** panel) so that products from the multiple request execution do not overwrite each other.

The typical steps in using the **Request Pool** are as follows:

1. Open the required Recipe Panel

2. Add the set of base Input Frames and edit Parameters as required

3. Press Add to Pool button to copy the input frames and parameters to the pool

4. Edit the input frames and parameters

5. Repeat steps 3 and 4 as required

6. Select one or more of the requests in the **Request Pool**

7. Press **Execute Selected** button

8. Review logs and products

6.14   Saving and Loading Settings

The settings of the **Current** tab may be saved to (an XML) file using the **Load Current Settings** and **Save Current Settings** menu item under the **File** menu of the Recipe Panel.

When saving the settings, the recipe, input frames, parameters, and product settings are saved (at user discretion) and may be reloaded later.

7. QUICK REFERENCE GUIDE

7.1 Summary of Inputs

Table 1: Keyboard shortcuts

SHORTCUT	ACTION
CTRL+q	Quit *Gasgano*
CTRL+r	Open keyword list editor
CTRL+s	Toggle open/close the Script Board
CTRL+c	Copy selected text to the Clipboard
CTRL+v	Paste content of the Clipboard to text input field

Table 2: Navigation Panel Mouse actions

BUTTON	CONTEXT	ACTION
Left click	Navigation panel, on nodes, folder icons, folder lines	Expand or collapse the corresponding item, showing or hiding its sub-components
Left click	Navigation panel, on files	Select the file, highlight the file and display it in the **File Detail Panel**
Right click	Navigation panel, on file	Displays the **Selected Files** popup menu
Ctrl-Left click	Navigation panel, on files	Multiple selection of non contiguous files
Shift-Left click	Navigation panel, on files	Multiple selection of contiguous files selection (first file must be already selected)

7.2 Keywords

Gasgano uses keywords for expressions, classification, and display and report generation. There are no keyword-dictionaries or defined keyword-sets in *Gasgano;* therefore entries of non-existent or bad keyword names will not be detected. For example a FITS file report requiring the PIPPO keyword to be printed will not produce an error. The column in the resulting report will be left blank.

ESO Hierarchical keywords can be expressed in both full (e.g.: HIERARCH ESO TEL AIRM START) and dot (e.g.: TEL.AIRM.START) notation. While *Gasgano* supports both forms of notation for data entry, typically the dot notation is used for display purposes.

7.3   Meta Keywords

Gasgano defines internally a number of keywords that are used within the application but are not read from a FITS file header. These keywords are called Meta Keywords.

The Meta Keywords currently in use in *Gasgano* are the following:

♦ CLASSIFICATION: a keyword representing the FITS file Classification, as defined by means of the classification rules.

♦ FILE_PATH: a keyword representing the full path of the data file on disk.

♦ PIPE_PRODUCT: a keyword indicating if the file is a raw frame or pipeline product. A raw frame will have a PIPE_PRODUCT value of "F" while a pipeline product file will have a PIPE_PRODUCT value of "T".

7.4   Keyword Expressions

This section discusses the use of keyword expressions in *Gasgano*. Keyword expressions are used to classify FITS data files and filter the data file set.

Keyword expressions return a Boolean value. They are a combination of Operands and Operators.

Operands have three different types: character strings, numbers and keywords.

♦ Keywords are represented as the names of the keyword and return either a Float or a String value.

♦ Strings are represented as characters enclosed in double quotes (e.g. "SKY"). A String cannot return a Float value.

♦ Floats are numeric values, (12, 0.1, 45.67, etc.). Floats can not return a String value.

Supported Operators are listed in Table 3 (where LHS = Left Hand Side; RHS = Right Hand Side).

Table 3: Keyword expression operators

OPERATOR	DESCRIPTION
AND	Boolean operator which evaluates to **true** if and only if all of its (two or more) sub-expressions evaluate to true.
OR	Boolean operator that evaluates to **true** if one of its two or more sub-expressions evaluates to true.
=	Returns **true** if the LHS equals the RHS.
!=	Returns **false** if the LHS equals the RHS.
>	Returns true if the LHS is greater than the RHS.
>=	Returns true if the LHS is greater or equal to the RHS.
<	Returns true if the LHS is less than the RHS.
<=	Returns true if the LHS is less than or equal to the RHS.
Like	Performs a string comparison where the RHS must be a string operand. The RHS may contain '*' characters which are treated as wild cards when matching with the LHS. **Note: Only the "*" wild card is valid, there are no other wild card operators available with the Like operator.**
NotLike	As for Like but the opposite result is returned upon evaluation.

7.5 Keyword Expression Classes

A keyword expression is a tree structure with AND and OR operators as nodes and comparison operators (=, >, <, Like, etc) as leaves of the tree structure. Evaluation of the expression involves evaluation of each of its leaves and nodes in a depth first traversal until a result for the root node is determined.

The nodes of the tree are either AND or OR operators containing two or more branches. The leaves of the nodes must be comparison operators (=, !=, <, etc.). Each comparison operator contains a RHS (Right Hand Side value) and LHS (Left Hand Side value) and an operator.

Comparison operators may be evaluated as Float objects or String objects depending on the values of operands. A comparison will be evaluated using Float objects only if all operands return a valid Float value. Failing this, the comparison is tried using String objects. If comparison using Floats or Strings is not possible, the expression will fail to evaluate.

Figure 20 shows a couple of simple expressions and their string representations.
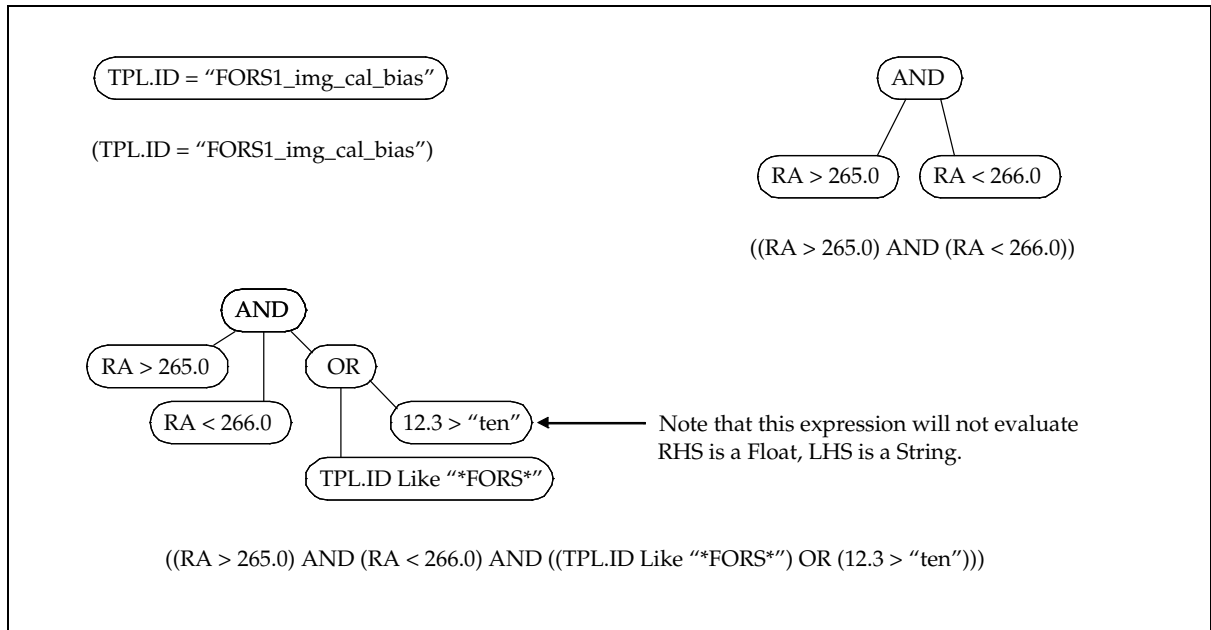
Figure 20: Expression Tree Examples

APPENDIX A – SYSTEM REQUIREMENTS

System definition and requirements

Software requirements: supported platforms are Linux RedHat 9, Scientific Linux, HP-UX 11.0, as well as Solaris 5.8. For viewing fits files, image display software such as skycat or ds9 is needed.

External Viewer: *Gasgano* comes with some example scripts to assist the user in connecting *Gasgano* with an external viewer. The example scripts, located under ${HOME}/gasgano/bin, are:

- § load_ds9: run ds9 as the FITS viewer

- § load_rtd: run rtd to display FITS viewer

The scripts may require some editing to run. In order to test and modify the scripts, they may be run passing the paths to a FITS file as the first command line argument, for example:

```
${HOME}/gasgano/bin/load_ds9 /data1/vimos/vmbias.fits
```

CPL Recipes: To run **CPL recipes** within *Gasgano* the CPL core libraries must be installed and be available in the library path when running *Gasgano*. The LD_LIBRARY_PATH environment variable must include the directory paths of all CPL core libraries.

To ensure that the required libraries are in the library path the following command can be run:

```
ldd ${HOME}/gasgano/lib/libjavacpl.so
```

The output of the command should indicate that all required libraries **except the libjava.so** are located. If any libraries **other than libjava.so** are lists as "not found", the LD_LIBRARY_PATH environment variable needs to be corrected.

The following is an example of correct output for the ldd command, indicating that all required CPL core libraries have been located:

```
libltdl.so.3 => /usr/lib/libltdl.so.3

libcplui.so.0 => /home/theuser/cpl/lib/libcplui.so.0

libjava.so => not found

libc.so.6 => /lib/tls/libc.so.6

libdl.so.2 => /lib/libdl.so.2

libcplcore.so.0=>/home/theuser/cpl/lib/libcplcore.so.0

libqfits.so => /home/theuser/cpl/qfits/lib/libqfits.so

libcext.so.0 => /home/theuser/cpl/lib/libcext.so.0

libm.so.6 => /lib/tls/libm.so.6

libnsl.so.1 => /lib/libnsl.so.1
```

```
/lib/ld-linux.so.2 => /lib/ld-linux.so.2
```

The following is an example of typical output when the CPL core libraries were not located:

```
libltdl.so.3 => /usr/lib/libltdl.so.3

libcplui.so.0 => not found

libjava.so => not found

libc.so.6 => /lib/tls/libc.so.6

libdl.so.2 => /lib/libdl.so.2

/lib/ld-linux.so.2 => /lib/ld-linux.so.2
```

Hardware Requirements: *Gasgano* requires at least 100 Mb of RAM (depending upon data set size and usage). It is advisable to run it on machines with not less than 512MB RAM. If visualised on a remote X11 terminal, make sure that at least 48 MBs of memory is available.

Processor speed becomes an issue only when accessing large numbers of compressed fits files.

Downloading *Gasgano*

Gasgano can be downloaded from the dedicated Web page, located at:

http://www.eso.org/gasgano.

Installation instructions are also posted on this web page.

Contact the User Support Group (usg-help@eso.org) if you need assistance.

APPENDIX B - .GASGANORC DEFAULT SETTINGS

Printout of a default .gasganorc, note that not all properties are relevant to this version of Gasgano. Key properties are explained in the table below.

```
# Gasgano Application Properties
# Date: Tue Jun 08 15:16:55 MEST 2004
AUTO_DISPLAY=false
CLASSRULE_FILE=gasgano/config/VLT.rul
COMPONENT_FONT=Dialog:14
CUBE_TO_VIEWER=
DATA_FILES=
DBOBS_DISPLAY_COLUMNS=CLASSIFICATION:RA:DEC:TPL.EXPNO:TPL.NEXP
DBOBS_DISPLAY_COLUMNS_SIZE=0:0:0:0:0
DBOBS_INSTRUMENT=All
DBOBS_INSTRUMENTS=ISAAC;FORS1;FORS2;UVES;VIMOS;CONICA;EFOSC;CES;SUSI;SOFI;EMMI;WFI;VINCI;MIDI
DBOBS_PERIOD=66
DBOBS_REQUIRED=false
DBOBS_SERVER_ID=
DBOBS_SERVER_URL=
DBOBS_USERID=
DB_SERVER_URL=
DB_USER_NAME=qc
DISPLAY_COLUMNS=CLASSIFICATION:TPL.ID:ORIGFILE:TPL.EXPNO:TPL.NEXP
DISPLAY_COLUMNS_SIZE=0:0:0:0:0
FILTER_FILE=gasgano/config/filter.rul
FITS_TO_VIEWER=
FRAME_BOUNDS=8,100,920,650
INSTRUMENT_GROUPING=ISAAC=;FORS1=;FORS2=;UVES=;VIMOS=;CONICA=;EFOSC=;CES=;SUSI=;SOFI=;EMMI=;WFI=;VI
NCI=;MIDI=;
KEYWORDLIST_DIR=gasgano/keywords
MENU_CUSTOM1=
MENU_CUSTOM1_LBL=Custom Menu1
MENU_CUSTOM2=
MENU_CUSTOM2_LBL=Custom Menu2
MENU_CUSTOM3=
MENU_CUSTOM3_LBL=Custom Menu3
MENU_CUSTOM4=
MENU_CUSTOM4_LBL=Custom Menu4
MENU_RETRIEVE=dpclient -renameschema ARCFILE -outpath
MJD_SORTING=false
OBSTATUS_MAPPING=P:Partially defined D:Defined Q:queued for scheduling I:Initiated S:Started p:Paused during execution A:Aborted
X:Executed R:Reduced C:Completed L:released K:Cancelled
OB_STATUS_FROM_DB_REQUIRED=false
PRINTER_NAME=lp
PRINT_FONT=Monospaced:16
PRINT_ORIENTATION=P
RADEC_CONVERSION=true
SCRIPTS_DIR=gasgano/scripts
SHORTEN_FILES_PATH=true
SHORT_FILENAME=true
SHOW_EXTENSIONS=true
TEXTFILE_EXTENSIONS=ascii;paf;longlog;log
TFITS_TO_VIEWER=
UPDATE_TIMER=0
VERBOSE_LEVEL=0
```

The following table lists the key properties found in the .gasganorc file along with their meaning and location.

Property Name	Preference Edit Location	Meaning
AUTO_DISPLAY	File Detail Panel for fits file: **Auto Display** checkbox	Send files to external view automatically when selected in the **Navigation Panel**
CLASSRULE_FILE	Preferences, Data tab:	Location of the classification rules

	Classification Rule file	
COMPONENT_FONT	Edit .gasganorc file directly	Java font and size for display of file entries in the **Navigation Panel**
CUBE_TO_VIEWER	Preferences, File Display & Grouping tab: **Cube Viewer**	Path to external viewer of cubes (display of multiple fits files as cube)
CUBE_VIEWER_TYPE	Preferences, File Display & Grouping tab: **Cube Viewer Type**	Type of viewer specified in CUBE_TO_VIEWER
DATA_FILES	Preferences, Data tab: **Data Directories and Files**	List of all data files and directories of data files to be read in on start of application.
DISPLAY_COLUMNS	Preferences, File Display & Grouping tab: **File System – Displayed Keywords Data**	List of keywords to be displayed in the table columns of the **Navigation Panel**
FILTER_FILE	Preferences, Data tab: **Filter Rule file**	Location of file of filters used to filter the data set (**File->Filter Files**)
FITS_TO_VIEWER	Preferences, File Display & Grouping tab: **Fits Viewer**	Path to external viewer of fits files
FITS_VIEWER_TYPE	Preferences, File Display & Grouping tab: **Fits Viewer Type**	Type of viewer specified in FITS_TO_VIEWER
INSTRUMENT_GROUPING	Preferences, File Display & Grouping tab: **Instrument Specific Grouping**	List of instrument and keyword(s) couples for defining keyword based file grouping on a per instrument basis
KEYWORDLIST_DIR	Preferences, Data tab: **Keyword List directory**	Default directory location for loading keyword lists
MJD_SORTING	Preferences, File Display & Grouping tab: **sort by MJD**	Turn on or off file display sorting by MJD keyword. If off file sorting is alphabetical (*true* or *false*)
PRINTER_NAME	Preferences, File Display & Grouping tab: **Printer Name**	The name of the default printer used when printing a text report
PRINTER_FONT	Preferences, File Display & Grouping tab: **Printer Font**	Java font and size used printing text reports
PRINT_ORIENTATION	Preferences, File Display & Grouping tab: **Printer orientation**	Print text reports as Landscape or Portrait (*P* or *L*)
RADEC_CONVERSION	Preferences, File Display & Grouping tab: **convert RA/DEC**	Display RA and DEC keyword values as sexagesimal or decimal notation (*true* for sexagesimal *false* for decimal notation)
RECIPE_SET	Preferences, Recipe Configuration	List of CPL recipe library paths and recipe name loaded and made available in *Gasgano*
SCRIPTS_DIR	Preferences, Data tab: **Scripts directory**	Default location of scripts presented in the **Script Board**. This directory may contain sub-directories with instrument names containing instrument specific scripts
SHORTEN_FILE_PATHS	Preferences, Data tab: **shorten data file paths for commands**	Short path names to selected files, if they have a command path. See RUNNING EXTERNAL EXECUTABLES for details. (*true* or *false*)
SHORT_FILENAME	Preferences, File Display &	Display files in the Navigation Panel

	Grouping tab: **truncate filenames**	with full path (*false*) or file name alone (*true*)
TEXTFILE_EXTENSIONS	Preferences, Data tab: **Textfile Extensions**	List of file extensions indicating text files that are to be read by *Gasgano*
TFITS_TO_VIEWER	Preferences, File Display & Grouping tab: **Tfits Viewer**	Path to external viewer of tfits files
TFITS_VIEWER_TYPE	Preferences, File Display & Grouping tab: **Tfits Viewer Type**	Type of viewer specified in TFITS_TO_VIEWER
UPDATE_TIMER	Preferences, Data tab: **Automatic Filesystem Refresh**	Time (in minutes) between refresh of the data file set to synchronize the Gasgano data file set with the file system. *0* means no auto refresh
VERBOSE_LEVEL	Preferences, Data tab: **Verbose Level**	Controls debug output to stdout. *0* indicates not output to stdout. If non-zero, *Gasgano* should be run with "-debug" command line parameter

APPENDIX C – VERBOSE LEVELS

The verbose level can be set in the **Data** register of the **Preferences Menu**. The verbose facility is to be used <u>for debugging purposes only</u>, upon request from ESO Support personnel and is in no way meant to be complete, user-friendly, or understandable for the users. The default value of this option should be set to **No Verbose** during normal use of the tool.

When a verbose level is set, Gasgano should be run with the "-debug" command line parameter. If this parameter is not used all output is redirected to /dev/null.

The possible values and their description are:

DataModelChange: changes like addition or removal of OBComponents and VRB to the application memory are displayed

DataModelEvent: notification to the Navigation Panel tree of addition/removal of components

ExternalCommands: not used

BaseName: verbose information on how the basename for an OBComponent is determined

Timing: activates collection of statistics for header read, classification and overall file read times.