

LINC-NIRVANA observation preparation software: a flexible approach

Alexey Pavlov^a, Jan Trowitzsch^a, Wolfgang Gässler^a and Jürgen Berwein^a

^aMPIA – Max-Planck-Institut für Astronomie, Heidelberg, Germany

ABSTRACT

LINC-NIRVANA (LN) is a near-infrared imaging interferometer for the Large Binocular Telescope (LBT). It is expected to have unprecedented imaging performance in the near-infrared both in terms of angular resolution and limiting magnitude, thanks to the large collecting area of the two LBT mirrors (8.4m) and by means of Multi-Conjugated Adaptive Optics (MCAO) and a Fringe and Flexure Tracker System (FFTS). For such a complex instrument the process of observations has to be carefully prepared, considering the constraints imposed by features of the instrument and scientific objectives. This paper addresses the design of the LN Observation Preparation Software (LOPS), the main goal of which is to provide the observer with a tool to create valid observation program for LINC-NIRVANA. The current status of LOPS with its key components is described and critical aspects are addressed.

Keywords: LINC-NIRVANA, observation preparation, observation planning and execution, plug-in system

1. INTRODUCTION

Astronomical observation with interferometers is different to conventional imaging or spectroscopy. It not only needs advanced data reduction algorithms to extract the information from the data, but also needs well-prepared observations, taking into account the constraints imposed by scientific objectives as well as features of the instrument.

LINC-NIRVANA (The **L**BT **I**Nterferometric **C**amera and **N**ear-**IR**/Visible **A**daptive **I**Nterferometer for **A**stronomy)¹ is a near-infrared imaging interferometer for the **L**arge **B**inocular **T**elescope (LBT). It will combine the two 8.4 m primary mirrors of the LBT in so-called "Fizeau" mode. The beams collected by the two telescope units are corrected by two adaptive optics systems, then co-phased in real time and combined inside a cryogenic camera, interfering in the focal plane. It will combine the two 8.4 m primary mirrors of the LBT in so-called "Fizeau" mode. The beams collected by the two telescope units are corrected by two adaptive optics systems, then co-phased in real time and combined inside a cryogenic camera, interfering in the focal plane. LINC-NIRVANA will operate at wavelengths between 0.6 and 2.4 microns, using state-of-the-art detector arrays. It is expected to have unprecedented imaging performance in the near-infrared, both in terms of angular resolution and limiting magnitude, thanks to the interferometric mode and to the large collecting area of the two mirrors. The LINC-NIRVANA instrument will deliver the sensitivity of a 12 m telescope and the spatial resolution of a 23 m telescope, over a field $10'' \times 10''$.

Observing with LINC-NIRVANA depends on the earth's rotation to present different position angles of the objects. Therefore there is considerably less freedom in the preparation and scheduling process. There may be the need for a particular position angle on a particular source first, then move to another position angle on another source for a while, then return. The aim is to create a consistent software that guides a scientific project through all stages of the LINC-NIRVANA proposal and observation program preparation. It should support the observer in advance to get an idea what kind of observation can be executed and what quality of data could be expected under given conditions. The focus of this work is to present the concept and design for the LINC-NIRVANA observation preparation software (LOPS) and its included tools and components.

Send correspondence to A. Pavlov: E-mail: pavlov@mpia.de, Telephone: +49(0)6221 528 329, Address: Max-Planck-Institut für Astronomie, Königstuhl 17, 69117 Heidelberg, Germany

2. SYSTEM OVERVIEW

The overall software structure of the LINC-NIRVANA instrument is divided in three layers. The top layer is the observation layer which provides the observation preparation. The middle layer is the system layer that contains the instrument control interface, the command flow and the sub-system optimization. The bottom layer is the sub-system layer and includes all sub-system dependent services, the device drivers, and it has a direct connection to the underlying hardware layer which is beyond the scope of this paper*. Here we are interested in the content of the two other layers, and in particular focusing on the tools an astronomer/observer will be mostly dealing with.

2.1 LN Observation Preparation Software

The **Linc-Nirvana Observation Preparation Software (LOPS)** is a tool that is meant to support an astronomer (observer) during the complex process of preparing the observations for LINC-NIRVANA. The LOPS consists of several software tools to define the overall data structure of an observation program. Executional relevant data like constraints and division into multiple observation blocks, depending on the amount of parallactic angles needed for an observation, will be defined there. Usually this task is done in advance before starting the actual nightly observation procedure.

The observer sets up all observations to achieve a scientific goal in an **Observation Program (OP)**. It contains all the information associated with one "observing program" including multiple targets. An **Observation Target Unit (OTU)** that contains all the information associated with one target is introduced. This means that for each target there is a separate OTU. Each OTU is a self contained entity and possess a target, predefined observing constraints, scientific objective, multiple instrument configurations including multiple calibrations and exposure times. Astronomers specify their programs in terms of OTUs, defining a target with the number of parallactic angles. Exposures taken at different angles are needed to obtain a better uv-coverage; an aperture synthesis image reconstruction procedure can provide reconstructed high-resolution images from LBT raw data with a spatial-resolution which reaches that of a 22.8 m single-dish telescope. By using OTUs "single" observations (**Observation Blocks**) are grouped together allowing the sharing of certain parameters among the elements of the OTU. **Observation Blocks (OBs)** present different position angles for a given object. It is used as a central concept for the data flow from the scientific oriented LOPS packages to the instrument oriented ICS package (to be more correct between LOPS and OSS (see Section 2). Each OB is composed by a number of instrument-mode-specific procedures. In its turn each procedure contains a number of parameters which are defined by the user. This is done either via directly typing their values or by using a special tool/editor component of LOPS. The procedures are grouped in a procedure sequence table.

2.2 LOPS External Entities and Observation Support Software

LOPS should provide an astronomer with easy access to any useful information about targets, possible reference stars (catalogs) as well as instrument configurations. Thus, a connection to corresponding external entities needs to be provided. Figure 1 depicts a simple context diagram of the LOPS with its relationships to the external systems and actors. Target and reference star information is available via a database/catalog interface. Furthermore, existing observation programs are accessible from a special repository. Each observation preparation carried out by the observer needs to be based on a defined instrument configuration. The existing instrument configurations are supported within LOPS.

Observer This is an astronomer who would like to create a proposal and has to provide an observation program with the related observation information.

Target/Reference Star Databases The system takes advantage of access to external data bases (remote or local) in order to present the observers with helpful information about their targets, reference stars, or other such information.

*A complete description of the overall LINC-NIRVANA software structure can be found in²

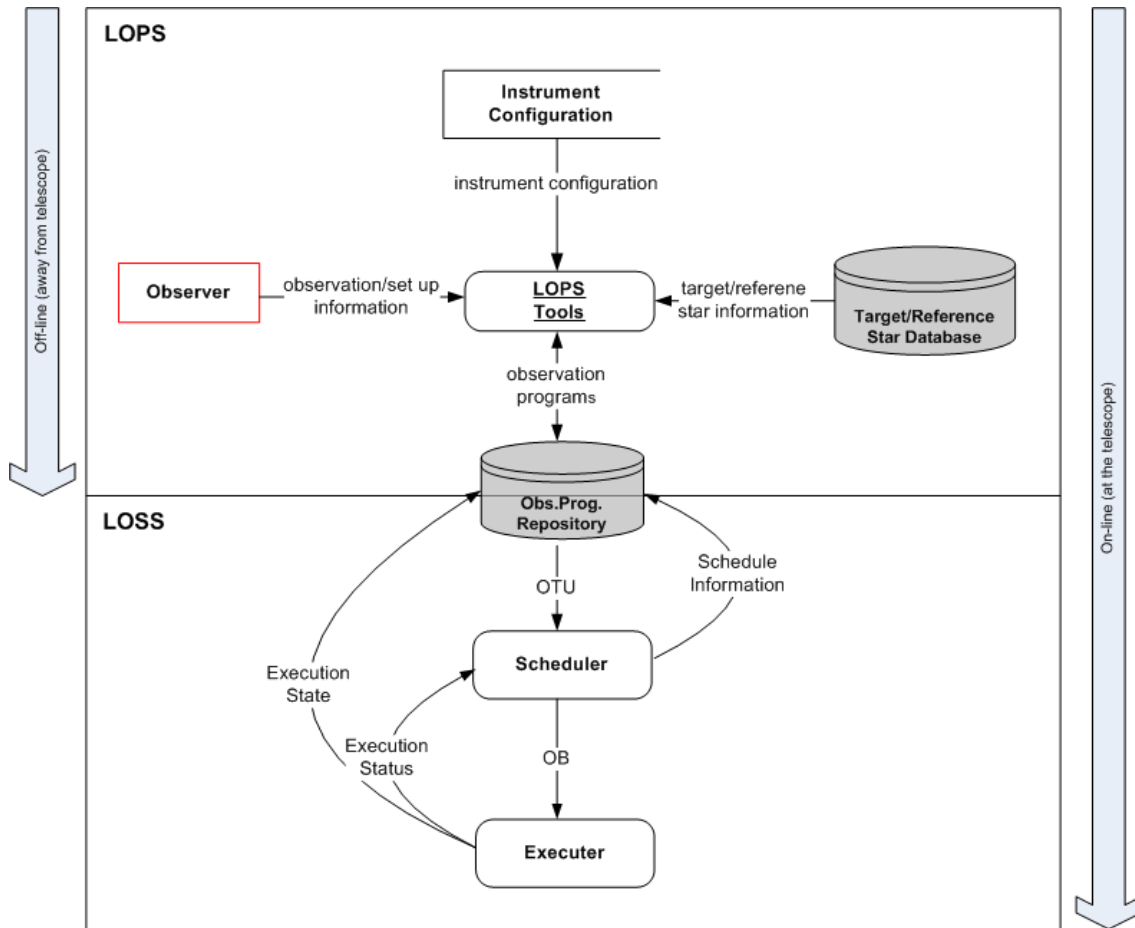


Figure 1. On the top panel a simple context diagram shows the relationships between LOPS and external entities. The arrows represent the information received or generated by LOPS. The closed and opened boxes represent an actor and data store respectively. On the bottom panel shows the data flow from the LOPS through the Observation Program Repository (data storage) to the Observation Support Software.

Instrument Configurations The base configuration of the LN instrument must be provided to complete an observation program. This information together with the observation one will be fed into the hierarchical-structured observation program.

Observation Program Repository (OPR) The efficiency of LN will depend strongly on the availability of various observation programs with different constraints. It might be not possible to receive all images for one program in one night. The observations can spread over several nights or run due to the fact of availability of parallactic angles, atmospheric conditions or instrument problems. Therefore, it is reasonable to have a repository of available observation programs (located at the telescope) to keep track of the completion of the programs. The OPR feeds the observation support software with available programs for the following scheduling and execution. The specific feature of the LN scheduler will be to scan the OPR and to re-optimize the schedule during the observation and to react to current conditions.³

LN Observation Support Software (LOSS) The observation support software consists of a set of software tools to proceed the scheduling and execution of observation blocks. The LOSS is placed between the scientific oriented software, the "observation program software", and the instrument oriented software, the so called

”instrument control software”. In the framework of the LOSS the data structure from the LOPS are processed, evaluated and sent as instrument level commands to the ICS.

The flow of the data from the LOPS to the LOSS including the scheduling unit and execution unit is described in Figure 1. After the observer has defined the entire OP, the data is stored in a data storage unit (repository). This unit is a central unit accessible from either the LOPS or the LOSS for data exchange and process logging. When the observation process has started, either in advance for a prescheduling or during the observation time, one or more OTUs are loaded from the data storage to create a schedule for the included OBs. The defined schedule will be stored again in the data storage to eventually edit the schedule for future use. If the schedule is ready for execution, each OB is send to the execution process when an OB is next to be executed. During this process, information about the execution status is collected and fed back to the scheduling process to do a rescheduling if necessary. A detailed algorithm of the scheduling process is discussed in.³

2.3 Functional Specification

Several functional requirements for LOPS can be specified. These requirements have been identified on the basis of the extended use case diagram for LOPS that is shown in Figure 2. It depicts the use cases that lead to the creation of a LN observation program. Each use case represents a specific sequence of events that occur when the user interacts with the system. The user (observer) is represented by a single actor. The actor initiates the creating the Observation Program (“Obs.Program”) use case.

The main functions of the LOPS are summarized in the following:

- to build an observation program with various elements of the observation program (observation program is a full set of observations the observer sets up to achieve his scientific goal)
- to allow automatic derivation of certain elements in the observation program;
- to allow automatic propagation of the certain items among elements of the observation program;
- to allow interactive work with astronomical catalogs (remote or local);
- to allow the performance estimation and the investigation of the parameter space for justification of the scientific objectivities;
- to enforce feasibility and consistency checks of the observation program as well as verification of certain elements of the observation program;
- to allow submission of the observation program to an observation program repository for authorized users;
- to allow retrieving and modifying of the existing observation program for authorized users;
- to allow the observatory staff(support astronomer) to retrieve and modify some of components of the observation program;
- to allow on-line (at the telescope) and off-line (away from the telescope) work.

2.4 LOPS Design

The LOPS design is arranged in a layered structure. It is depicted in Figure 3. The fundamentals for LOPS are concentrated in a set of components that implement basic features comprising such as *Logging Management*, *Data Management*, *Error Handling*, *Preference Management* and etc.

The GUI is build on top of these basic features. It implements a module plugin system that provides a common way to add specific tools/modules to the LOPS. Constructing a flexible plug-in system the development of the data management is a very important part. In the framework of the LOPS Data Manager package integrates the representation of the astronomical science objects and OTU as well as OB and procedure configuration data. In allows not only to a safe way to incorporate new modules/tools into the LOPS but offers a transparent way

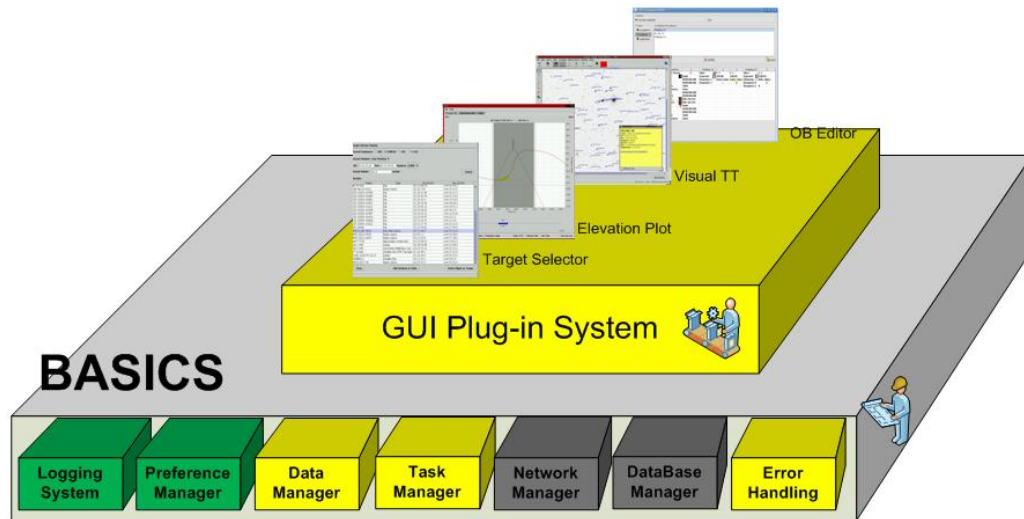


Figure 3. Design of LOPS - layer oriented structure

to upgrade/update the configuration of the observation procedures which keeps an interface to the instrument control software agile (in 2.4.2 we discuss it in detail).

The tools included in LOPS are described briefly in the following. Additionally the Observation Program Editor and Observation Block Editor are discussed in more detail because are key components of the LOPS. It is worth to mention here that the development of the LOPS is based on extensive reusing the components from the existing observation preparation softwares and libraries such as SEA/APT^{4†} and JSky.⁵

Observation Program Editor The Observation Program Editor is the main graphical user interface for of LOPS. It contains a navigator area on the left side which shows the observation program components in a hierarchical structure and the corresponding editor tools for each observation component on the right side.

Target Star Selector The Target Star Selector is a catalog navigator that is used to search and extract of the objects (targets and reference stars) from available remote (or local) catalogs and images.

Elevation Plot The Elevation Plot provides a panel for displaying a plot of the elevation, airmass and parallactic angles for a given target. We've extended it with the features to define interactively a number of OBs and the suitable time slot for each of them. It also can provide the "perfect" point-spread function (PSF) and the modulation transfer function (MTF) of the ideal LBT interferometer.

Visual Target Tuner The Visual Target Tuner of the LOPS provides a graphical view of the observation and allows interactively to modify elements of it. Together with the Target Star Selector it enables an image of the field of the target to be overlaid with the field-of-views of the MCAO and available reference stars.

Observation Block Editor The Observation Block Editor is used to create an Observation Block (OB) and manage its procedures.

Offset Pattern Editor The Offset Pattern Editor will be used to construct a nod-, dither- mosaic pattern.

[†]Granted to MPIA due to Software Usage Agreement between NASA and MPIA.

Performance Estimator The Performance Estimator is one of the important component of the OPS that should provide the observer with an idea what he/she can expect under given conditions.

2.4.1 OPE

The OPE represents a single observation program and contains all components which allow user to interact with the current observation program (see also fig. 4):

- **Module** allows the astronomer to view/edit the current selected item in the OP tree structure.
- **ModuleSettings** is an interface that must be implemented by any object which wishes to contain a Module. In other words this interface must be used for any LOPS tool that is intended to be plugged into the LOPS framework.
- **TreeNavigation** is a tree object that displays the contents of an observation program **ObsProgram**. Each node in the **TreeNavigation** is a navigation component which implement the component listener so that it is notified when it's needed.
- **ProgramNavigation** is a container for navigation objects. It provides access to the navigation component tree.

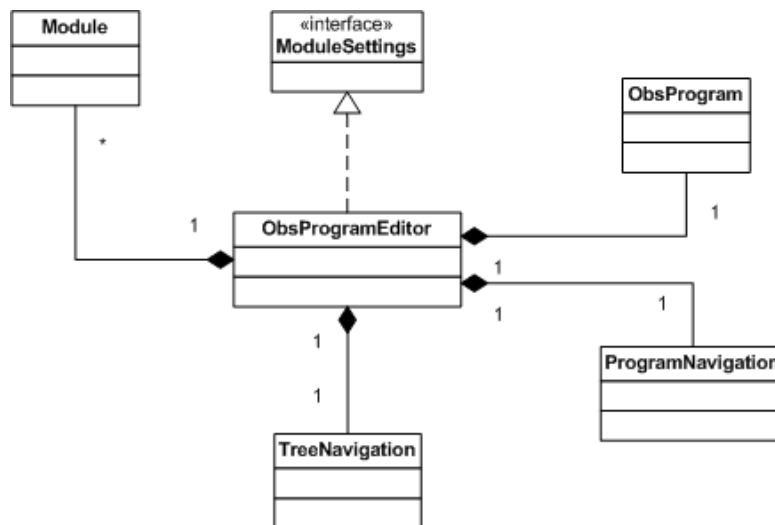


Figure 4. *OP Editor - main components to allow interaction with observation program via plug-in modules*

One of the very useful feature is that the plug-in system provides a mechanism to retain module instances and reuse it whenever possible instead of creating a new one each time. It is important for performance issue since the creation of modules is expensive.

2.4.2 OBE

The **Observation Block Editor (OBE)** is used to create an Observation Block (OB) and manage its procedures. The procedure that a typical science OB contains are:

- An acquisition procedure, describing how the target acquisition needs to be performed. It includes instrument setup, the exposure time the acquisition image should have and the position angle of the target on the sky.
- At least one science procedures, describing filter and exposure parameters, for example which integration time (DIT and NDIT) each exposure should have and which nodding/dithering pattern should be used.

The development of the OBE in the framework of LOPS follows the concept of the Observation Block used in the P2PP Tool⁶ (ESO). Observation Block is used to describe observing sequences in the terms of pre-defined procedures (scripts). Each of these procedures includes a set of parameters and is defined in a procedure signature file. In our design needed meta data that is important for dynamically displaying and handling the procedure in the LOPS is also included in this file. Such a file is based on a XML Resources read/write interface syntax implemented by the 'basic' *DataManager* of LOPS (see above). The supported node types (keywords) for these files are:

- **scalar** → simple node, only one value (string) comparable with pair-key concept
- **structure** → complex node, including more than one scalar node
- **array** → complex node, including more than one scalar and/or structure node

Several node entries are mandatory at the different levels of each signature file. At the top level of each signature file some nodes are required for important global data such as procedure type, date of creation, version, etc.:

- **Name** → Name of the procedure that is specified in the signature file
- **Instrument** → Used instrument configuration (name)
 - For example: considering an instrument configuration file `inst_linc_nirvana.conf` than a signature based on this configuration needs to include an entry: `<Instrument> linc_nirvana </Instrument>`
- **Mode** → Operational mode for the instrument
- **Version** → Version of the procedure signature
- **Date** → Date when procedure signature was created or latest modification
- **ProcType** → Type of procedure (acquisition, science, calibration)

The dependencies between the main OBE components are shown in Figure 5. They can be assigned to three different parts: *GUI*, *Table/Model*, and *Procedure Signatures*.

The *GUI* part comprises all classes implementing the main view (`OBEditorView`), its sub panels, like the `ProcedureView` panel and the `ProcedureTable` which is based on the `ProcedureModel`, and the user interactions. It implements the module plug-in system as well as the needed action handlers and listeners. Furthermore, the selection of procedures based on their type and the displaying of the existing pre-defined procedures from the signatures repository directory is managed. The action buttons for adding, copying and deleting a selected procedure from the procedure sequence table are included and handled.

The *Table/Model* part comprises the classes responsible for the procedure sequence table as well as for the underlying model. It implements the needed listeners for the user interaction with the table. By using a the `RendererFactory` and the `EditorFactory` the displaying and the editing of procedures and their parameters is managed. The `ProcedureModel` class extends the standard `AbstractTableModel` class and implements the underlying procedure model for the `ProcedureTable`. The `RendererFactory` class manages to render the value for each cell in the table based on the parameter type. For example it manages that a *Boolean* parameter is displayed as a check box centered in the table cell. The `EditorFactory` class provides the appropriate editors for the editing of procedure parameters. When the user types or clicks on the cell selected in the table, the appropriate editor is selected like for example a combo box for selecting a value from a list of pre-defined values.

The *Procedure Signature* part comprises all classes managing the parsing of the procedure signature files as well as the handling of the included meta data for setting up the displaying for the procedures and their parameters. The `Parameter` class is the base class that stores information for a parameter of a procedure. It contains getter and setter methods for mandatory signature file entries for a parameter. The meta data for a

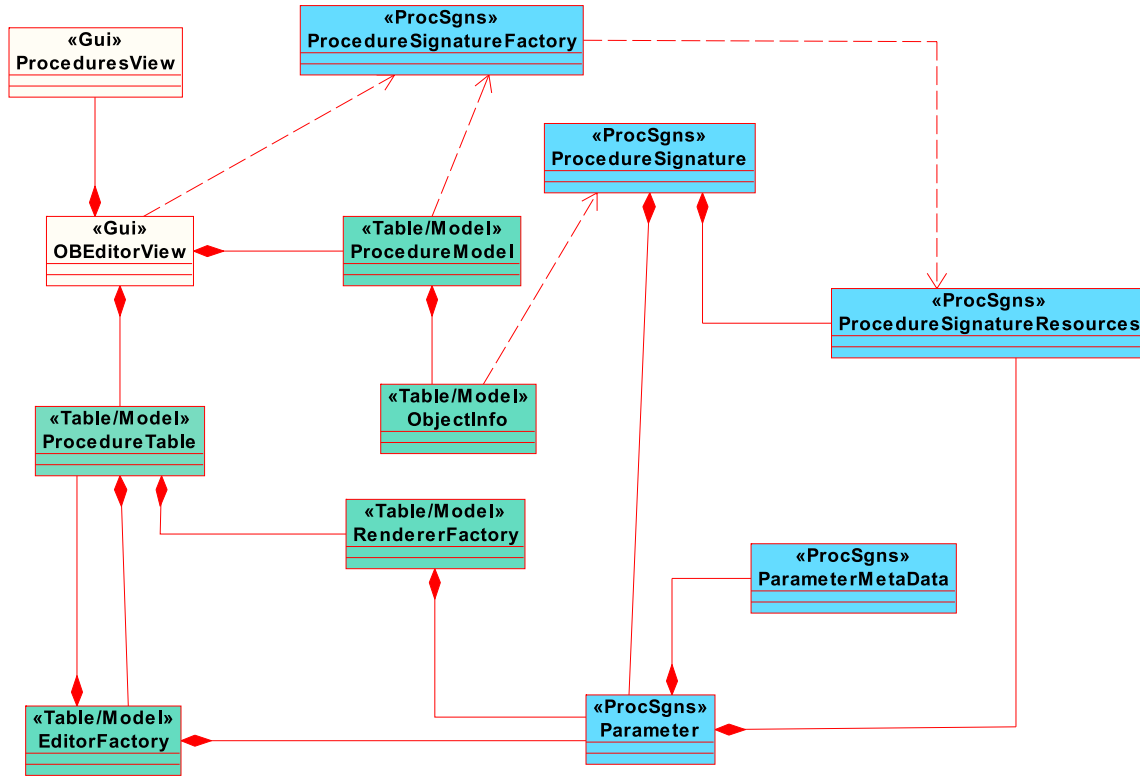


Figure 5. OB Editor - dependencies between the main components

Parameter is handled based on the related **Resources** object by the **ParameterMetadata**. The data from the procedure signature files is managed via the **ProcedureSignatureResources** class using the basic *DataManager* interface of LOPS. Based on the **ProcedureSignatureResources** the **ProcedureSignature** class implements the procedure signature representation which handles the list of the procedure's parameters.

The OBE allows the user to select specific procedures according to its type (acquisition or science). The corresponding procedures are listed in a frame on the right side of the OBE main GUI framework (see Fig. 6). A procedure can be selected and then added to the OBE's procedure sequence table. The OBE displays the parameters of each added procedure in the form of a two-column table entry, where the 1st column includes the parameter names (labels) and the 2nd column allows to set-up and modify the parameter values. The verification of the corresponding value range, format, and data type itself is provided: entering of valid values should be only possible. The help (tool tip) is shown if needed. A connection to existing other tools within LOPS for certain parameter types is provided. For example for a target parameter the *Target Star Selector* can be called. Additionally a procedure tree providing an overview of the procedure sequence is included in the OBE. This tree is interconnected with the sequence table which means that the user can navigate to a certain column (or parameter entry) via the corresponding tree node.

3. CONCLUSION AND OUTLOOK

The LOPS as presented is intended to be used off- or online for preparation and planning of observations by the astronomers. Astronomical observations with interferometers depend much more on proper preparation and planning than conventional observations. Therefore, to gather the full set of data for successful reduction of the observed data for the LINC-NIRVANA instrument we develop the observation preparation software – LOPS. The underlying framework of the LOPS is based on a plug-in system. This framework is designed not only to provide a common way to add specific tools/modules to the LOPS, making it extendable and easy to maintain, but also allows to keep the configuration of the observation procedures agile. Features to edit and validate consistency of

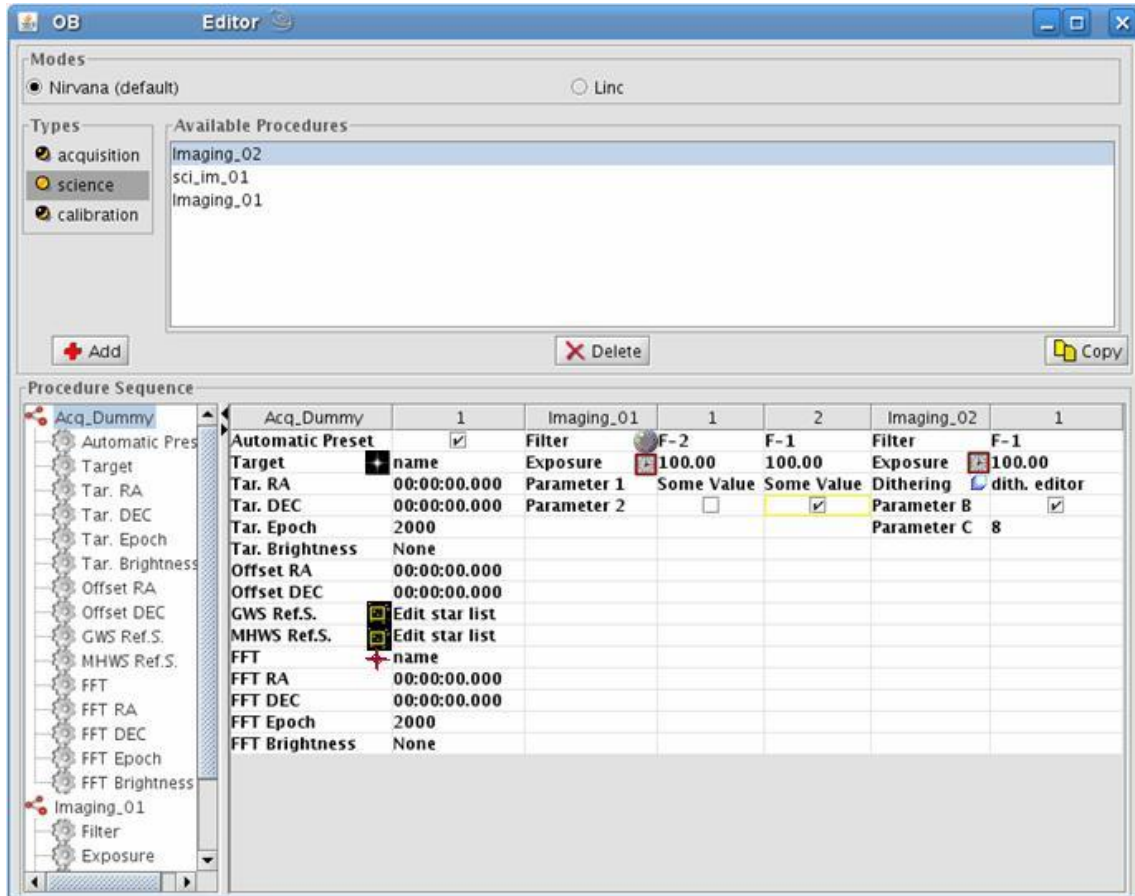


Figure 6. LOPS - OB Editor Overview

the OP are implemented within the software by developing the key components of the LOPS: OPE and OBE. With the next step we will add the plug-in modules, such as Target Selector, Performance Estimator, etc. to the LOPS.

REFERENCES

- [1] Herbst, T. and et al., "Beyond the fringe: an update on the construction of LINC-NIRVANA, a Fizeau imaging interferometer for the LBT," *SPIE* **6268-72** (May 2006).
- [2] Kittmann, F., Gässler, W., Briegel, F., and Berwein, J., "LINC-NIRVANA Instrument Control Software," in [*Astronomical Data Analysis Software and Systems XVI*], Shaw, R. A., Hill, F., and Bell, D. J., eds., *Astronomical Society of the Pacific Conference Series* **376**, 661+ (Oct. 2007).
- [3] Berwein, J., Pavlov, A., Briegel, F., Gaessler, W., and Storz, C., "Reactive scheduling for LINC-NIRVANA," in [*Observatory Operations: Strategies, Processes, and Systems. Edited by Silva, David R.; Doxsey, Rodger E.. Proceedings of the SPIE, Volume 6270, pp. 627010 (2006).*], Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference **6270** (July 2006).
- [4] Roman, A. J., Douglas, R., Downes, R., Krueger, A., and Karla, P., "Astronomer's Proposal Tool: the first two years of operation," in [*Optimizing Scientific Return for Astronomy through Information Technologies*], Quinn, P. J. and Bridger, A., eds., *SPIE* **5493**, 351-358 (2004).
- [5] Albrecht, M., Ochsenbein, F., A., B., Fernique, P., Dolensky, M., and Wicenc, A., "JSky: Building a Library of reusable Java Components for Astronomy," *ADASS*, <http://archive.eso.org/JSky> (1999).

- [6] Comeron, F. and Silva, D., "Phase 2 Proposal Preparation Tool (P2PP)," *User Manual: Doc. No VLT-MAN-ESO-19200-1644* **Issue 7** (Dec. 2005).