Mem. S.A.It. Vol. 75, 282

Memorie della

Control software for the SRT. I

F. Palagi

INAF - Istituto di Radioastronomia, Sez. di Firenze, Largo E. Fermi 5, I-50125 Firenze

Abstract. The control software for the SRT is being developed according to the Unified Software Development Process. The first activity in this process is capturing the system requirements and build a model that will be the basis for all the following workflows: analysis, design and implementation. This contribution presents the current status of the system requirement capturing and modeling.

1. Introduction

The SRT team is structured in Integrated Activity Groups (GAI). The Software GAI is commissioned to develop a software system for the control and operation of the SRT. The primary objective is to get the telescope ready for its qualification and acceptance tests.

Other objectives are *i*) Field System integration for VLBI observations, *ii*) data management, and *iii*) definition of a standard interface to the SRT system.

Another requirement for the system is expansibility, so that future requirements and instrumentation can easily be integrated.

The development process is based on the Unified Software Development Process (USDP; Jacobson et al. 1999) and on the Unified Modeling Language (UML)[1],[2], as implemented by the CASE *Poseidon for UML*, by Gentleware AG, Hamburg, DE[3].

The SRT control system will be named **NURAGHE**, after the ancient stone buildings in Sardinia. We hope our software system will be as modular and long-lasting as nuraghes are.

This contribution is a progress report and will illustrate the present state of the system requirements definition and modeling.

2. Requirements description

According to the USDP the first activity is to collect all the system requirements. The relevant information was obtained from the SRT design documentation, from interviews with future users and all other people involved in the SRT development, generally referred to as *stakeholders*. Previous experience gained in other similar projects was also a valuable source of information. Several meetings with the other SRT GAIs gave the necessary input to this step. Another source of information was the recently published report of the *Science with the SRT* Working Group (Brand et al. 2005).

All the requirements will be described in a report of the SRT GAI 06 Memo Series (hereafter referred to as RDD, *in prep.*).

[1] `http://www-128.ibm.com/developer-works/rational/library/769.html`

[2] `http://www.uml.org/`

[3] `http://www.gentleware.com/index.php?id=home`

2.1. Objective

The main objective of the system can be synthesized as follows:

NURAGHE provides the facilities to complete the observation of a radio source. An observation consists of a measurement of either the total power or the spectral distribution of the incoming radiation, using one of the available receivers and one of the available detectors.

The telescope efficiency is regularly monitored in terms of pointing accuracy and antenna gain.

2.2. Facilities and results

A set of facilities and activities are implied by the above description.

- Given the observing frequency, NURAGHE selects the receiver, detects the focus where it is installed and sets up the wave path accordingly.
- If required, the active surface sub-system is activated.
- Once the detector is selected, NURAGHE is able to start the appropriate Control and Data Acquisition Programs (DAPs), which include the control of the telescope and measurement of the radiation power.
- The system provides pointing correction and antenna gain measurements.

2.3. The use-case diagram

The objective of the system can be sketched in a use-case diagram, which shows the external *Actors* (represented by a schematic human), and the possible uses of the system (Fig. 1). Actors are entities to whom the system provides any result of value, such as performing a task or returning some information.

For instance Manager is a *kind of* Operator that uses NURAGHE for maintenance, while Operator has no privilege to access the maintenance use-case.

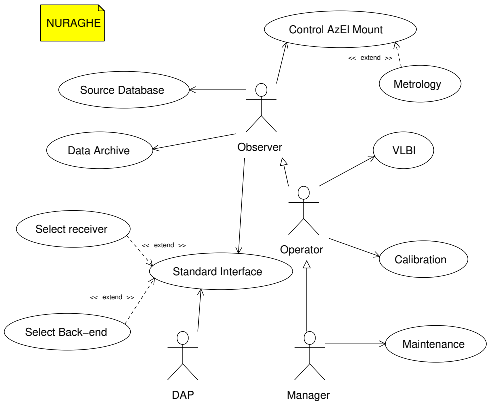

Fig. 1. The overall use-case diagram of NURAGHE. DAP is Data Acquisition Program.

3. The business model

Once the main objective is defined it is useful to analyse the context in which NURAGHE is going to operate. In other words, we must say which are the system boundaries. From RDD we identify at least four entities which are outside the scope of NURAGHE, although NURAGHE has some interaction with them:

Field System: Originally the VLBI data acquisition software. It includes, however, some components that may become part of NURAGHE, namely the antenna monitoring programs.

DAP: Each detector is expected to come with its own control program.

Metrology: The metrology system will provide the information to control the position of the secondary mirror.

ACU: The Antenna Control Unit will be provided by the manufacturer, and will control the primary mirror movement, including the cable-wrap system.

4. The System Requirement Model

Any software model consists of several views of the underlying system. The most important aspects to be modeled are the structural (static) view and the behavioural (dynamic) view of the system.

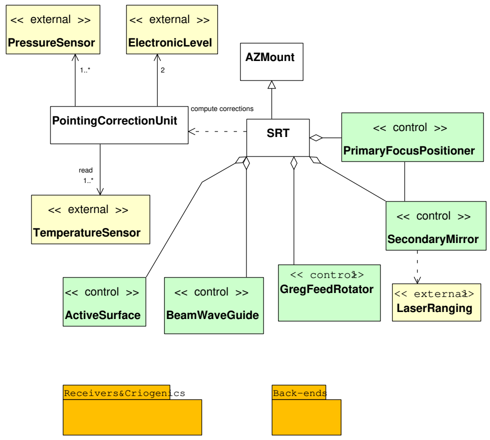

Fig. 2. The overall class diagram of NURAGHE.

Class, Use-Case and Collaboration diagrams mostly describe the static view while Activity, Sequence and State diagrams model the dynamic aspects of the system.

Class diagrams show the system static structure, use-case diagrams show its functionality, while collaboration diagrams show which classes of the static model cooperate in the realization of a specific use-case.

Activity diagrams convey information similar to the collaboration diagrams, adding the time evolution of the collaboration. Sequence diagrams show the time sequence of messages exchanged between objects in the system. State diagrams show how a class reacts to events and external stimuli.

The static view of NURAGHE at a very high level of abstraction is shown in Fig. 2 in the form of a UML class diagram.

Each class represents a component in the system. The diagram shows that the SRT *is an AZMount*. This means that it inherits all the characteristics of an AZMount. The SRT *depends on* the Pointing Correction Unit to get the appropriate pointing corrections. Again, the SRT *is an aggregation* of five control classes, the Primary Focus Positioner, the Secondary Mirror Positioner, the Gregorian Feed Rotator, the Beam Wave Guide and the Active Surface. Classes marked as *external* are external devices whose control is outside the scope of NURAGHE. Finally the diagram shows that there are two packages dealing with the control of backends and of Receivers & Cryogenics, respectively.

RDD describes each component in the system in greater details including:

general description: What is the component?

functional requirements: Which are the services or facilities provided by the components?

informative requirements: Which informative quantities does the component hold?

constraints: Which conditions or restrictions must be met by the component?

For each of these items we give an example; for the first three we refer to the primary mirror component, for the last one to the BWG.

4.1. Primary mirror: general description

We present here an excerpt from the RDD, describing the Primary Mirror control.

> The movements of the primary servo-system will be controlled by an *Antenna Control Unit*, ACU, which receives commands via a communication interface, most probably an ethernet interface.
> ACU operation. It is suggested to include the operation modes defined for the new Medicina and Noto ACUs, in order to have an *operation modes subsystem* common to all three telescopes. In particular the *time-tagged program track* is requested.
> The primary servo-system will move also the cable-wrap which must be synchronised with the telescope movements. Due to its mass and dimensions the cable-wrap cannot be passively dragged by the azimuth drive.

4.2. Primary mirror: functional requirements

Some of the services provided by the primary mirror component are:

1. Move to a predefined (Az, El) position (*preset*).

2. Track an equatorial position (Ra, Dec).
3. Set the telescope to a position at a defined time. *time tagged program track*.
4. Scan at a defined velocity.
5. Cross scan on a point.
6. Set to the stow position

4.3. Primary mirror: informative requirements

The status of the primary mirror contains the following quantities:

1. Current position.
2. Commanded position.
3. Applied pointing correction.
4. Az and El prelimits and final limits.
5. Motion status including type of move, Az rail branch, alarms.

4.4. BWG: constraints

The BWG can be set only when the following conditions are met:

– The Gregorian Feed Rotator is in *open* position.
– The Secondary Mirror is in *ON* position.
– The Primary Focus Support is in *stow* position.

These constraints are shown in the activity diagram of Fig. 3.

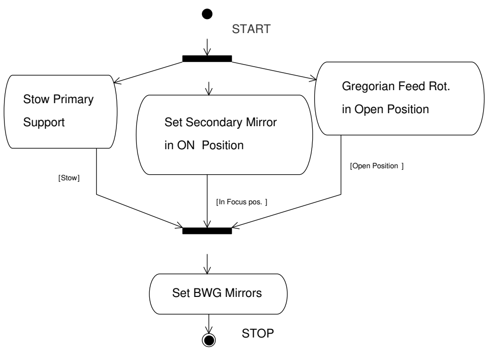

Fig. 3. The conditions to be met for the operation of the BWG

5. Behaviour of the system

At the current stage of development the behaviour of the system is not yet modeled so no diagram will be shown. We can only say that each use-case will be detailed through one or more scenarios which consider the main event flow as well as the exceptional event flows (e.g. error conditions).

For the most complex classes, state diagrams will help in modeling their behaviour against events and stimuli.

6. Preliminary solutions

This section lists decisions that are taken at a very early stage of development.

The ALMA Common Software (ACS), will be adopted as the main support layer, which is based on CORBA, a Distributed Object managment specification. The reason for this choice is described by Migoni (these Proceedings).

For a system of the dimensions of NURAGHE it is highly recommended to adopt a CASE for its development, to help communicating among the various contributors. A second and equally important advantage is that the project documentation and implementation are kept aligned, as they are essential parts of the development process.

Poseidon for UML was chosen as it is a powerful, yet cheap, CASE-tool for software modeling from the requirements specification up to the implementation phase. Once the model is built, this case is able to generate the skeletal structure of the C++ or Java code implementing the classes in the system. Reverse engineering is available only for Java.

7. Conclusion

The present stage of the development process of NURAGHE, the control system for the SRT, has been presented with emphasis to its requirements specification and modeling.

References

Brand, J., Caselli, P., Felli, M., Mack, K.-H., Poppi, S., Possenti, A., Prandoni, I., &

Tarchi, A. (Eds.) 2005, ‘The Sardinia Radio Telescope (SRT). Science and technical requirements”, IRA Internal Report 371/05
Jacobson, I., Booch, G., & Rumbaugh, J. 1999, The Unified Software Development Process (Addison-Wesley, Boston).