THE GALILEO PROJECT. WORKSTATION SOFTWARE SYSTEM AND USER INTERFACE

A. Balestra, P. Marcucci, M. Pucillo, C. Vuerli
Trieste Astronomical Observatory

Introduction

The Galileo National Telescope (TNG) [1] is an italian 3.5m optical telescope to be located in the Canary islands. Its main elements have been derived from those of the ESO NTT with some modifications. In the following we will point out the features of the Telescope Control System (TCS) with special concern both on general issues like software standards, portability and modularity, and on particular ones like close telescope-instruments integration, remote control and fault tolerance.

The GALILEO TCS has been designed keeping in mind three main points.

The technology used, based on state-of-the-art components, is subject to a quick evolution and makes it a hard task to foresee future developments and changes. This means that both the hardware and software solutions must be found among those which will guarantee an adequate support for the future, i.e. proprietary systems as well as closed ones should be avoided.

The second point concerns the lifetime of such a project. Either the time needed to build the telescope and its operative lifetime must be considered. A control system able to solve just the current problems will not be adequate, since the evolution of astronomical instruments and, even more unpredictable, the evolution of astronomers' wants cannot be easily foreseen.

Last but not least, the needs of those people that will work on the system to develop instruments, high level software modules and the like, must not be disregarded.

As a result the TCS has been designed, and is now being developed, using only stable standard systems, either hardware and software. A modular, flexible structure has been choosen for the overall architecture of the system, which has been maintained as homogeneous as possible, keeping the number of different components as low as allowed by current technology. Besides, a set of software interfaces and program modules has been built in order to hide all the low level details, allowing programmers to operate in an user-friendly environment.

TCS architecture

The TCS architecture is distributed on a communication network which connects two physical layers: the upper level is formed by UNIX workstations which are dedicated to interface the operators to the telescope as well as to the instruments [2]. At the lower level a set of local processors, based on the VME bus standard, operate directly on the telescope subsystems and on the instruments [3].

A further, logical, level contains the tasks, or groups of cooperating tasks, which run inside the local processors and control a particular function of the subsystem. At the same level are placed the processes running inside the workstations.

The TCS components of the first two levels are called systems, while those belonging to the third one are called units. Both systems and units carry on their activity through the use of a set of parameters, and are controlled through a set of commands.

In order to assist in hiding this structure, every reference to a component of the TCS should be made by a symbolic name. The term 'component' refers in this context to a system connected to the network, to an unit in a system, or to a parameter or a command.

To meet these requirements a standardized format for the names of all the components of TCS must be defined. This naming convention has been designed so that to reproduce the organization of TCS: each name is built up by one to three fields, specifying the full path needed to reach the desired component inside the structure of TCS. The first field will contain the name of the system, the second one the name of the unit and the third the name of the item, either a parameter or a command. The layout of the full name will thus be as follows:

<system>_<unit>_<item>

This convention applies also to the names of the files containing the definitions needed by the software to operate correctly. The Workstation Software System (WSS) is thus able to address each component of TCS, should it belong to the local system or to a remote one, and lends itself naturally to remote control and remote observing.

A central facility has been also provided in order to manage the conversion from symbolic names to physical addresses. The Telescope Data Base (TDB) contains all the definitions concerning the components of TCS, and is complemented by a library of routines which allow WSS processes to access the data in it and to resolve the name-address translation.

It is based on the UNIX Shared Memory capabilities, so that it can be accessed by all the processes of the WSS. It is actually composed by three segments.

The first one (segment 0) is the static segment and contains some housekeeping informations followed by the static section of the database. This part is initialized by WSINIT process from the definition files; it cannot be overwritten by other processes.

The second one (segment 1) is the dynamic counterpart of the static segment, it contains changing status values and can be accessed by all other processes both in read and write mode.

The third one (segment 2) is allocated for some special internal data structures; actually, the segment contains four microcommands queues associated to each connected VME, a number of telemetry log buffers, where the TMVER process puts all the received telemetry data, which are periodically written by WSINIT on a logfile, and finally a number of ring buffers used to perform parameters statistics.

The Workstation Software System and the User Interface

The WSS is built up by a set of seven processes, each dedicated to a unique task, whose layout and mutual interactions have been described in more detail in [2], together with the off-line utilities and the User Interface. Here we wish to put in evidence some of the characteristics of the WSS:

Fully table driven software modules:

all the data needed by WSS processes are contained in special definition files, which can be created and maintained off-line with dedicated editors. Interactive panels and graphic panels are treated in the very same way, allowing system programmers to build telescope and instrument interfaces in a very user-friendly environment.

Message oriented, Internet based communication protocols:

the use of standard communication protocols and the message oriented architecture devised for the WSS, make it robust against common failures, and open to new transport technologies.

The communication mechanism is based on two UNIX tools: SysV messages and BSD sockets. The former are restricted to the use on a single computer, while the latter uses protocols (e.g. TCP/IP) to communicate on computer networks. At the system startup each unit opens its own message queue identified by the PID of the unit. Then it can sends messages to local or remote processes using related PIDs as key, as obtained from the TDB, or read messages sent to its own queue taking up an associated action. This mechanism, with the internetworking capabilities of modern workstations, allows a precise localization of each component of the TCS using both TCP/IP and UDP network protocols. It is also a solid base to build a fault tolerant system using UDP broadcast messages as "workstations heartbeats", and TCP/IP timeout to monitor VMEs activity.

OSF/MOTIF based Graphical User Interface:

OSF/MOTIF is the widely accepted de facto standard in the field of window managers. Its use guarantees the complete compatibility of WSS with future enhancements.

Replicated software structure all-over the workstation systems (textual symmetry):

the software structure of WSS is exactly replicated in each workstation of the TCS. This includes the replica of the Telescope Data Base and of the definition files. The different role of each workstation is defined in a configuration file. With a careful customization of the definitions inside this file it is possible to build a system with high resilience to failures, where each workstation is capable of carrying on the tasks of any of its companion workstations. Moreover one or more workstations can be defined to be remote operating consoles, which are not allowed to access directly the TCS but can obtain informations through an intermediate master system, which in turn has a complete control over their activity.

Conclusions

After a few tests on the functionalities of WSS, whose development is still in an intermediate phase (the last version is the 1.0), the system seems to perform reasonably well. However more tests are to be carried on, mainly about the capability of the software to face a sustained rate of telemetry transmission from a large number of VME systems, and to perform contemporarily complex display operations like graphic animations and interactive panel upgrades.

Even more interesting, and conclusive, will be the tests on a laboratory CCD camera, which will be started soon in order to verify the performance of the system, and of the programmers as well, when building a true interactive interface.

Acknowledgments

We wish to acknowledge the contribution of the colleagues of the GALILEO Project, and the useful discussions held with the colleagues of the ESO/TDE division.

References

[1] C. Barbieri et al.
The Galileo National Telescope
ESO conference on Progress in Telescope and Instrumentation Technologies, 1992

[2] A. Balestra, P. Marcucci, F. Pasian, M. Pucillo, R. Smareglia, C. Vuerli
GALILEO Project. Workstation Software System Architecture Design Document
TNG Technical Report n. 9, 1991

[3] A. Baruffolo et al.
Data acquisition and control informatics for the TNG
ESO conference on Progress in Telescope and Instrumentation Technologies, 1992