The VLT Control Software – Status Report

G. RAFFI, ESO-Garching

Introduction

The VLT control software consists of all the software, which will be used to directly control the VLT Observatory and associated instrumentation. This is now in the implementation phase, performed to a large extent directly by ESO staff in the VLT software group, but also by Consortia of Institutes responsible for some ESO instruments and by Contractors who implement some of the telescope subsystems.

The main foundation body for the VLT control software, called VLT common software, has already been implemented at about 90% by now. It is used across all computers of the VLT observatory (telescopes and instruments) and is designed, implemented and maintained by the VLT software group. The group is also responsible for monitoring software developed outside ESO and for later integration into the VLT control software at the VLT Observatory.

Starting with this *Messenger* issue, regular reports on the VLT control software will be given in *The Messenger*. The first contribution, just after this Status Report, is an article on the Real Time display software.

VLT Common Software

The general architecture of the VLT control system consists of a fully distributed system based on a number of workstations and microprocessors, called LCUs (Local Control Units). There will be about 40 workstations and 150 LCUs in the complete VLT and instrumentation configuration. This complies with the VLT control system architecture, as defined for the VLT project by the Electronics Department.

The VLT common software consists of a layer of software over the UNIX operating system, in the case of workstations and on top of the VxWorks operating system, for the LCU microprocessors.

The main Packages in the VLT common software are: LCC (LCU Common software). It works on LCUs over VxWorks and is completed by a set of drivers for the ESO standard cards and a user interface (GUI) on the host workstation. It is the common platform for all the LCUs of the VLT and instruments. A motor library, dealing with the VLT standard control cards is also contained in this software. Test and debugging tools are provided.

CCS (Central Control Software). It is a layer of software built on top of a commercial system (RTAP, Real Time Application Platform by Hewlett-Packard), which provides a real-time database, runnable on several UNIX platforms. It contains also a tool to assist in the implementation of user interfaces (Panel editor).

HOS (High level Operation Software). It consists of a set of high-level tools to provide support for operators and astronomers, also in the preparation phase of observing runs.

Additionally there is a library, which forms what is called the Instrumentation common software and is specific to instrumentation applications.

VLT Common Software Releases

The VLT common software is used in the whole VLT programme, telescope and instruments, by ESO staff, Contractors and Consortia. It is also the basis for the NTT upgrade project.

Therefore, the VLT software group has been supporting for one and a half year a system of releases of this software, which is by now distributed to about 15 sites, both internal and external to ESO and runs on both HP (HP-UX) and SUN (Solaris 2) platforms. The next release will be distributed externally at the end of August 1995.

The VLT common software has a size of about 400,000 lines of code (including code, comments and test procedures), mostly written in C, but C++ is on the increase and Tcl/Tk procedures are also present.

Together with all this goes also a relevant Maintenance activity, supported by a system of SPRs (Software Problem Reports or Change requests). The level of confidence in the capabilities and quality of the VLT common software is quite high at this stage. This results from all the implementations and tests done by the NTT team for the NTT Upgrade project (see also separate section in *The Messenger*) and by other field tests at La Silla, like the one carried out with a CCD prototype in January 1995.

VLT Control Software for Telescopes and Instruments

This is the main part of the VLT software and emphasis is shifting towards it, as more and more design work gets completed. Implementation has started within the VLT software group in several areas, like telescope subsystems, telescope co-ordination, CCDs, ISAAC prototypes.

The first internal milestone of the Telescope Control System software (TCS) will be reached in August, with integration of co-ordination software and subsystems. This is actually a joint venture between the VLT and the NTT upgrade teams, so that most of the software will be the same.

The next major milestone for TCS is then coming at the end of the year, when TCS has to get ready for tests in Milan next year.

For what concerns instrumentation and detector software, the scientific CCD software will reach its first milestone, after the January 1995 prototype tests, in August. Instrumentation software starts to be implemented also in ESO, taking advantage of the INS common software. In particular prototypes for the ISAAC detector software are being developed, while the UVES software is still in the design phase. Collaboration with the NTT team, upgrading now the EMMI software, allows early verification and potential sharing of software.

Other Software for the VLT

Interfaces with the off-line software, which is also essential to run the whole VLT observations cycle and will be developed by the Data Management Division of ESO, are already partly defined and partly still in the definition phase. This includes interfaces to MIDAS, Archive, Scheduler and Catalogues. The VLT software group participates also in the activities of the On Line Data Flow working group.

The software developed outside ESO for the VLT by Consortia and Contractors is not mentioned in detail here.

VLT Control Software Trends

A set of new tools in the CCS software (Database Loader, Extended CCS, Event Handler, Class Browser) support an object-oriented design and implementation of the VLT software (workstation part). The VLT control software projects developed in ESO at the moment make use already of such tools.

A kernel of software, called CCS-lite, has been isolated in the CCS software. It consists of Panel editor, Sequencer, message system, error and logging system and database access to LCU databases.

CCS-lite does not require HP RTAP and therefore does not allow to access the workstation based real-time database. It allows though to easily implement user interfaces to LCU software, providing access to LCU databases and support at the workstation level for passing messages, errors and logs. It represents also an easily portable platform to any users, who will be able to prepare an Observing programme ahead of time on a generic UNIX workstation.

Some developments, like the Real time display, although part of the VLT control software, have been implemented in a fully portable way and can run completely independent of the VLT software (see following article).

Documentation and Standards

A complete set of specifications and user manuals exists for the VLT software and for the VLT common software in particular.

In spite of the large number of documents, there exist obvious "access roads" to documentation. So if you are interested in instrumentation software, you should simply start reading one document: VLT Instrumentation software specification. (Please refer to latest issue of 12/4/95.)

If instead the point of interest is telescope subsystems, the access specification is: VLT software – Telescope Control system functional specification.

These two documents can also be used as references to any other documentation, if deeper understanding is required.

The general requirements, standards and rules for this software have been fixed some time ago (although there is a constant evolution in the project) by the VLT software group.

All documents are available from the VLT Archive or directly via the ESO home page of the World Wide Web (main documents).

Meeting with Consortia and Contractors

A two-day meeting on the VLT control software was held at ESO on June 19–20,1995. Software representatives of about 10 external Consortia and Contractors were present. The meeting was organised around a number of tutorials and demonstrations, as most of the software comes with a set of auxiliary tools and user interfaces, to facilitate development.

For more information on the VLT software please use the ESO home page in the WWW. Should you have additional questions, you can contact by e-mail either graffi@eso.org or other members of the VLT software group.

The VLT Real Time Display Software

T. HERLIN, A. BRIGHTON, P. BIEREICHEL, ESO

Introduction

The VLT will contain a large number of technical CCD's, which will be used for guiding and slit-viewing cameras requiring real-time display facilities. The frequency of image display from these image sources can be several Hz, for example, while performing tracking. For scientific detectors, real-time display capability is equally important. In particular, requirements for infrared detectors have been a driving force in specifying the functionality for a real-time display system.

The Real Time Display (RTD) software was developed in order to support image display in real-time on the VLT. The RTD software provides a tool for users to display video like images from a camera or detector as fast as possible on a workstation or X-terminal. The RTD is implemented as a package providing a widget and library. It is designed to be a building block, adding display capabilities to dedicated VLT applications in areas such as telescope and instrument control.

The intention of RTD is *not* to provide the image processing functionality already existing with image processing packages such as MIDAS. Functional overlapping with image processing systems is kept to a minimum and confined to the area of on-line operations.

Although it is part of the VLT control software, the RTD is an independent software package and does not require any other VLT software components, external packages, such as MIDAS or any commercial products.

Functionality

The core of RTD is an image display widget which supports two image sources: cameras and FITS files. The *camera* is either a technical CCD or scientific detector, which provides the image data in shared memory. The RTD widget is notified via an image event mechanism by the *camera* that a new image is available. See below for more about real-time images.

Figure 1 shows a typical screen layout when working with the RTD widget. The application shown was designed for demonstration purposes and shows an image loaded from a FITS file.

The user can change the magnification of the displayed image, zooming in to get a close-up view of a section of the image or zooming out to get an overall view of a large image. A panning window supports navigation on the image and a zoom window enables the targeting of a single pixel.

The RTD widget supports a basic set of colour scaling algorithms: linear, square-root, logarithmic and histogram equalisation. MIDAS compatible colour maps are supported by the RTD widget as well as the MIDAS intensity transfer tables.

The colours can be manipulated by a colour bar rotating the colour map or changing the slope of the intensity via mouse interaction. The colour distribution can be changed by either manually setting the cut-levels or using automatic cut-level calculation.

The widget supports line graphics so that the user can overlay markers and text on the image. This might be used, for example, to identify interesting areas around a star. Standard line graphic components such as line, circle and text are available and can be drawn interactively on the image. Line graphic attributes such as line width, colour, filling, font, etc. can be set by the user via buttons and menus. In addition to the interactive line graphics, a programmatic interface is available to support overlaying of more complex line graphics, such as star maps taken from a catalogue.

Pixel query operations are supported at various levels:

• get and/or display the value of the pixel directly under the mouse.

• get and/or display a table of pixel values surrounding the pixel directly under the mouse.