The VLT control software development and installation

G. Raffi
European Southern Observatory (ESO),
Karl-Schwarzschild-Str. 2
D-85748 Garching, Germany
E-mail: graffi@eso.org

Abstract

The Very Large Telescope (VLT) control software installation is on-going at the new Paranal Observatory of ESO in Chile. The whole process of commissioning will take up to year 2000, when all the four VLT 8 meter telescopes will be completed.

The VLT control software approach has been to use at the basis standard commercial and public domain products (Rtap by HP, Tcl/Tk). A very comprehensive software layer, called VLT common software was added on top, before any specific control software was written. This has been an in house development, provided by a team of about 20 people.

The VLT control software amounts now to about one million lines of code. The common part (VLT common software) is distributed in 2 releases per year to Contractors and Consortia of Institutes associated to the VLT project and they are requested to use it in their VLT developments. The advantage of this approach is clear in terms of uniformity and maintainability of software. This has raised though a number of needs, like backwards compatibility and automatic regression testing (test scripts amount to almost another million lines).

A novel approach of the VLT software in general, is that the VLT control software is well integrated into an end-to-end Data Flow concept. The whole is enabling automatic execution of observations from proposal preparation to archiving and is suitable for service observing.

A relevant feature of the VLT software at this stage is that it has been validated (at about 90%) by re-engineering the New Technology Telescope (NTT) at the La Silla Observatory in Chile. This was done by an independent ESO team and the NTT is now back in service operation, providing at any new release plenty of useful feedback. This gives us the confidence that, above the obvious differences of hardware between telescopes, the VLT software will work properly also at the VLT.

1 Introduction

The VLT project consists of four telescopes each with a primary mirror of 8.2 m diameter, capable of working in parallel, equivalent to a single telescope of 16 m diameter. It is under installation at ESO's new observatory site at Paranal, a 2600 m high mountain top in the Chilean Atacama desert. At the moment of writing, while the four domes have already been erected, also the activity of commissioning the control software has started.

First light on the first unit telescope is scheduled for mid 1998, but the entire commissioning period for the four telescopes and the auxiliary telescopes will extend up to year 2000. At that moment the VLT will be the largest optical telescope in the world. The VLT will also work in interferometric mode, in combination with three movable 1.8 m telescopes. The instrumentation programs, involving control software based on the same components and standards, will be developed in parallel and will obviously continue to require dedicated efforts for a much longer time.

The size of the VLT control software, including telescope control software, is so far about 1 million lines of code (including comments, double so much if test scripts are added) and might become about 1.5 million lines when the full instrumentation complement is ready.

2 The VLT control system

The VLT control system is based on a distributed layout (about 50 workstations and 160 VME based microprocessors) with computers interconnected by various LANs intermixing Ethernet and ATM technologies (Fig.1).

With such a complex layout we found it fundamental not only to try out interface cards and routers in a stand-alone mode, but to build a mock-up where software can be installed and tested in simulation. It is a VLT control system without the hardware interfaces to the telescope and we call it VLT control model. It allows to test software integration and performances in advance. We give quite some importance to this, having seen that most of the problems with our previous releases were software configuration and interface issues, much before any hardware was concerned. We do this also in an effort to cut down installation times to hours rather than weeks.

3 The VLT common software

The main initial idea of the VLT was to have a well defined and comprehensive common layer of software for the whole observatory, main telescopes, interferometry and instruments. This we implemented and call VLT common software.

It makes all the VLT control software behave the same and gives to it a coherent design structure. It allows also to the different user interfaces to have similar look and feel, as they are all based on a common tool, called Panel editor.

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 Local Control Unit (LCU) microprocessors. It provides mainly common services, like an architecture

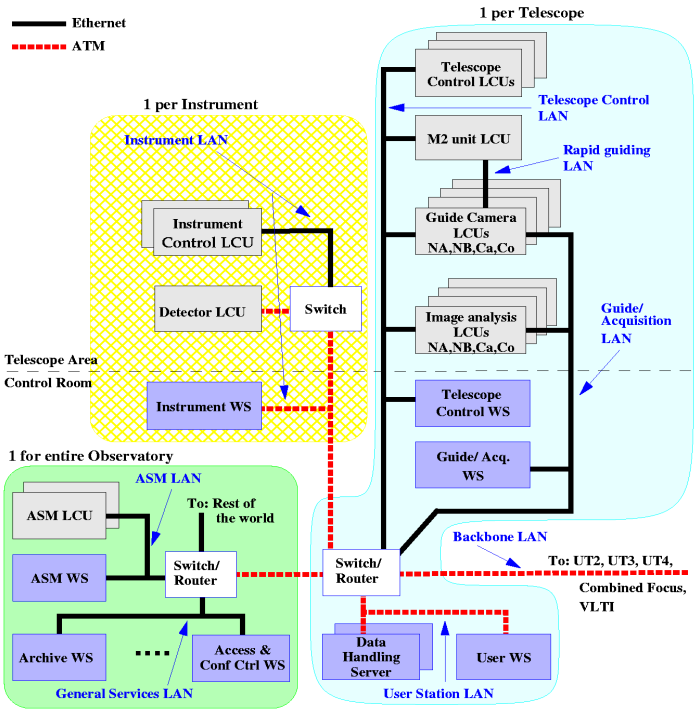

Figure 1. **Logical lay-out of the VLT LANs.**

independent message system, a real-time database for all telescope and instrument parameters, error and logging systems and a large number of utilities and tools. The inner mechanisms of the common software on the workstation side are provided by a commercial software product from Hewlett-Packard, called RTAP. A more extensive description of the VLT common software was given at the ICALEPCS '95 [Ref.1]. The smooth introduction into the project of an Object oriented design and implementation after its start-up phase was also presented at that conference [Ref.2].

The original idea of a thin layer of software common to all Observatory application software, has evolved in the course of time. It moved forward to include more and more high level applications, like the entire CCD control software used both for scientific and technical CCDs (e.g. guide cameras) and other general purpose software used by instrumentation (e.g. Real-Time Display software and INS common software).(Fig.2).

The VLT common software is portable on two platforms and all software has been tested both on SUN and HP Unix (Posix) workstations, while the microprocessors based on VxWorks do now support also PowerPCs.

4 Software development approach

An important choice in our design was to base it as much as possible on existing standards and products (public domain like Tcl/Tk or commercial like HP Rtap). We found it is important to resist as much as possible to the idea of re-inventing something better and slightly different, to spare expensive developments with a limited life-time.

The VLT common software is distributed in 2 releases per year to Contractors and Consortia of Institutesassociated to the VLT project and they are requested to use it in their VLT developments. The advantage of this approach is clear in terms of uniformity and maintainability of software. Bi-annual releases

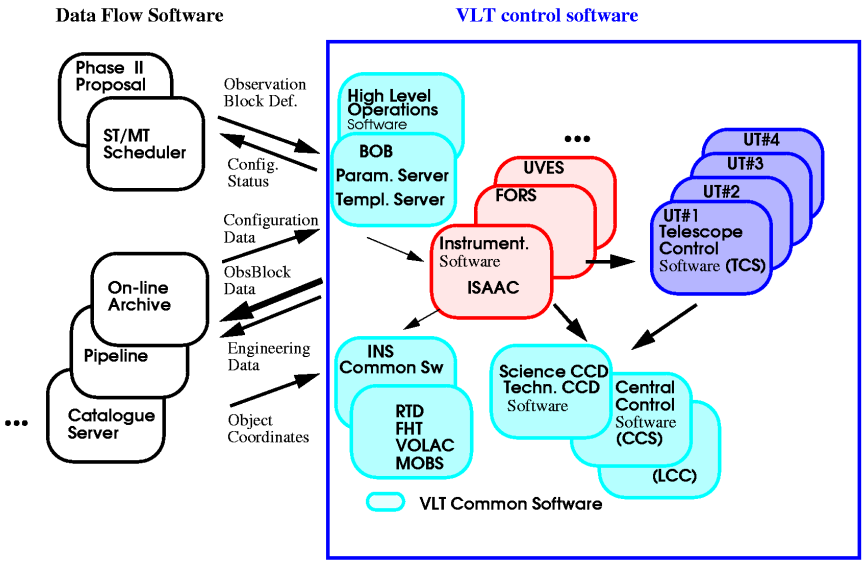

Figure 2 The VLT control software in the Data Flow context

implement in a natural way an incremental development approach. This allows to concentrate on the main features and leave the rest for a future release. It allows also to have frequent deadlines, on which one has to be strict, but they provide developers with a good feeling and feedback about the development process.

The languages used are C (lower layers) and C++ (and Object Oriented concepts) on the workstation side, while the code running on the microprocessors is written exclusively in C. The User Interfaces on the workstations are built using the VLT Panel Editor, which is based on Tcl/Tk.

5 Software test approach

The need to test software in its real computer environment (VLT control model) was already mentioned.

In the context of testing a considerable effort has been done during the last year in submitting most modules of the VLT common software to automatic regression tests, which can get repeated frequently and systematically. This was a goal and a necessity at the same time, as frequent releases become otherwise impossible to test (or alternatively would be of poor and non repetitive quality).

Important to note is also that test coverage has also to be monitored and improved from release to release. Problems which have been reported and fixed have to be picked up one by one and reflected in an equivalent test procedure. So test procedures get also developed incrementally during development and maintenance.

Additionally code has been exposed to tools like Purify and Quantify. This has made the base platform of the common software stable and reliable, but of course new features and higher level layers have been added as well, so that problems discovered nowadays tend to be more in the upper layers.

One constraint we have with the VLT is that non all the control software is done by the same team. Software quality and test coverage to be expected are then different and obviously worse than if one team only had developed everything. The final word on this will come only after the full integration at the observatory, which has still to come. So far we have noticed that Consortia of Institutes, developing instruments and software for the VLT, being typically smaller than ESO, show a lot of interest in following certain standards, particularly when embedded into existing software provided by ESO and are flexible in adapting to changes. Otherwise, differently than industrial contractors, they are less used to work with appropriate documentation and review levels, as done by the internal team of ESO.

6 Integration with data flow software

A novelty of the VLT software is the high level of integration achieved between the VLT control software and the Data Flow software, providing an end-to-end concept to the process of observing [Ref.3]. This goes from proposal preparation via the use of Templates, to computer optimized scheduling according to observation needs.

Several optional scenarios are then prepared for a given night and the final choice among them has to do with atmospheric conditions etc. The VLT control software receives observations sequenced in so-called Observation Blocks in such a way that they can be executed by a skilled operator, while the presence of the astronomer who prepared the programme is not required. The VLT control software acquires then data in the form of images or spectra, which are stored away immediately and then archived. Quality checks are done all along this process and a pipeline reduction is automatically done to work out important results. The data are ready then for packaging and shipment to the requesting astronomer, while other Observation Blocks are being executed (see again Fig.2).

7 Full validation of software

The past period has not been one of pure development only. It has been fully used by ESO also to try out the whole VLT control software on the New Technology Telescope (NTT) at La Silla by re-commissioning the entire control system and software in particular. This work was the responsibility of a different ESO team who did work in good collaboration with the VLT development group, providing a lot of useful feedback [Ref.4]. Not always such a lucky situation can happen in a big project, but it indicates at least how fundamental early field tests with hardware are. It shows also that in a complex system, some kind of simulation of the whole computer network needs to be part of the software test environment.

8 Conclusions

The VLT control software is ready for commissioning at Paranal and a number of conclusions and lessons have been learned from its development so far.

The VLT control software is based on standards and public domain products. It enforces a homogeneous design and outlook across all the project. Looking in retrospective design could have been more effective if it had been driven more by scenarios and examples than by conventional functional specifications. The transition to OO design and coding happened in a smoother fashion than we had envisaged. It created a certain redundancy with existing software, but with the advantage of allowing a transition path to a system with an expected longer life-time. The advantage of reusability has been already shown (e.g. 2 previous telescopes being upgraded with the same scheme).

The VLT control software uses an incremental development approach based on Releases. This was a very fortunate choice and allowed to break the project into reachable and controllable segments, with well defined deadlines. It is now intended to make releases less frequent during the integration and commissioning phases at Paranal and stabilize them by only fixing problems.

Testing is a strong point of the VLT control software, but still we are working on it, because it is crucial for the final result. Automatic regression tests, with a good level of coverage, are in routine use at every release. Still experience with a real case, like the NTT telescope, showed that also testing with a simulated environment (the VLT control model) was needed to shorten installation time. We have learned also that testing costs a lot of effort and time, even when automated.

Although the system is all based on the same common software, integration of software produced and not yet delivered by different teams, is going to happen in the next months. We have put into account some interface problems here. The system of configuration control and software problem reporting we have in place since the beginning, should help us in this phase of the project. The experience gained by the NTT team in "getting things to work" in the field shall be our background strength.

Next year the first VLT telescope will be operational and its software too.

People who wish to know more about the VLT project can access the VLT software documentation and general VLT documentation via a web browser, starting from ESO's home-page (URL: http://www.eso.org). This includes access to relevant specifications and user manuals.

Acknowledgements

The author wishes to thank all colleagues in the VLT Software Group and NTT team of ESO, who are the real authors of most of the Control Software mentioned and described here.

References

[1] G. Raffi - Status of ESO Very Large Telescope control software - Proc.ICALEPCS, pp.162-166, Chicago, 1995.

[2] G. Chiozzi - An object-oriented event-driven architecture for the VLT Telescope control software - Proc.ICALEPCS, pp.121-127, Chicago, 1995.

[3] M. Albrecht, M. Peron et al. - VLT Data Flow System: the NTT prototype experience - Proc.SPIE/TCS, San Diego ‘97 (not yet available).

[4] A. Wallander and J .Spyromilio - NTT project: a field test of the VLT software and hardware - Proc.SPIE/TCS, San Diego ‘97 (not yet available)