Preparation for VLT software commissioning at Paranal.

Eric Allaert and Gianni Raffi[a]

European Southern Observatory (ESO)
Karl-Schwarzschildstr. 2, 85748 Garching, Germany

ABSTRACT

The Very Large Telescope (VLT) software commissioning has started since a while, earlier than any VLT subsystems were ready for integration. This was possible thanks to the New Technology Telescope (NTT) upgrade (reported in a separate paper, Ref. 3), which shares most of the software with the VLT. The integration tests with the main VLT structure do also represent another fundamental milestone in the software commissioning process (see also separate paper, Ref. 4).

The whole control software is based on a very distributed computer architecture. The final layout of the computers (workstations and microprocessors), networking devices and underlying concepts have been tested both at the NTT and on the so called VLT computer control model, a relevant off-line subset of the computer equipment to be used in the VLT control room and telescope area for one unit telescope.

The VLT common software, including a real-time database, is the stable core of the whole VLT control software. This comprises also high level applications, like the Real-Time Display (RTD), the Panel Editor and the CCD software to be used for technical CCDs. It is distributed with a policy of regular releases, subject to automatic regression tests and is used by VLT Consortia and Contractors too. New modules have been added to insert the VLT control software in the Data Flow context, interfacing it in particular to Scheduler and Archive.

The VLT software support team will soon start regular operation at the VLT Paranal site, providing continuity between the integration activities of the various subsystems. They will be the front-end of the commissioning effort, based also on background support provided over links from the European headquarters.

Keywords: control software, commissioning, network, computer architecture, software management, progress report

1. INTRODUCTION

The VLT project consists of four telescopes each with a primary mirror of 8 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. The dome enclosures of the first telescope are in fact in their final acceptance phase as far as control system and software is concerned. Soon after the m1m3 subsystem, the m2 subsystem, the adapters and the main structure with the main alt/az axes will be integrated, including their control systems.

The past period has not been one of pure development for what concerns the control software. On the opposite, this period has been fully used by ESO 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 reports it in a separate paper at this Conference (see Ref.3).

First light on the first unit telescope is scheduled for early 1998, but the entire commissioning period for the four telescopes and the auxiliary telescopes used in the interferometric laboratory will extend well after the year 2000. 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.

---

a.For further information, Email: eallaert@eso.org, graffi@eso.org

2. DISTRIBUTED ARCHITECTURE

The logical lay-out of the Paranal Telescope Area LANs and computer equipment is illustrated in Fig. 1. Remark that for reasons of simplicity only the LANs of one Unit Telescope and one instrument are shown in this figure. It clearly illustrates the distributed nature of the VLT computer architecture.

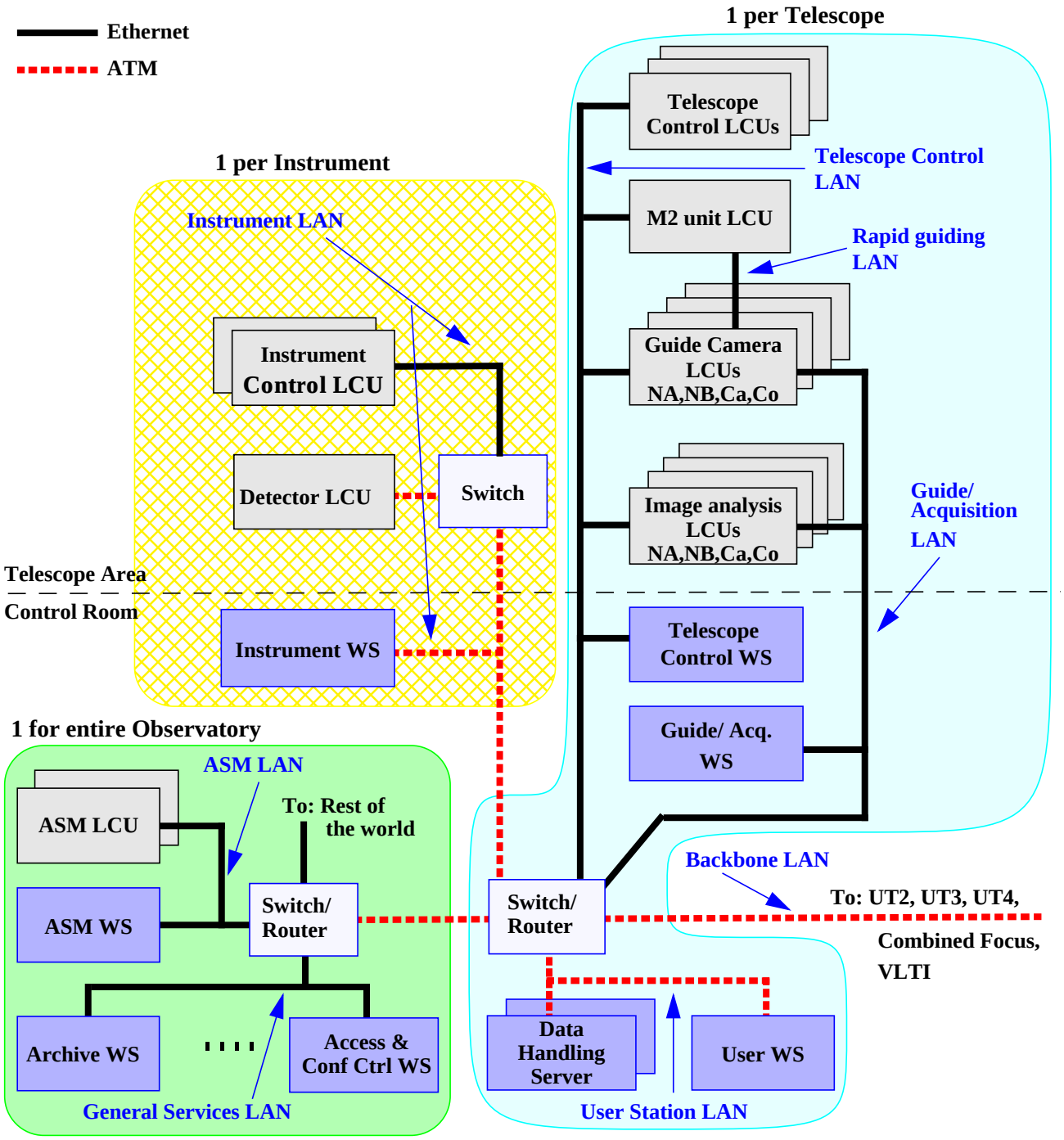

Figure 1. Logical lay-out of the VLT LANs. The label on top of each shaded block indicates how many times it needs to be repeated to complete the VLT.

Experience has been collected on this architecture with the NTT (see Ref.3), with the exception of Local Control Units (LCUs) having two LAN connections, which proved not to be required on the NTT, and ATM links for the new FIERA CCD controllers, which are not yet available. Additionally the fact of having several telescopes introduces the need for an ATM backbone on the VLT. All this to say that additional testing was required to try out the technology implied in the concept of Fig. 1, namely associated ATM switches, routers and related performances.

Still we do not think these stand-alone tests are complete enough. For that reason we are in the phase of building up the so-called *VLT control model*, consisting of a set of LCUs and workstations - a somewhat simplified (but not over-simplified) implementation of the above scheme. This is being realized in the Garching offices with the main purpose of testing the Telescope control software integration in a realistic set-up, even if hardware is mostly missing (but CCDs and motor/encoder mock-ups are available) (see also Ref.4). For the purpose of the architecture of the VLT control system, the VLT control model will allow to test the exact network and communication equipment configuration as planned for Paranal. It will also allow to test the User Station concepts of the Paranal control room, including the fact that access to instruments will be allowed from any user station.

Another aspect for which the VLT control model has now become important is testing of VLT common software. This is described in better detail in Sect. 4. It is important to stress here the role that the high level software (telescope and instrument control) can play on such a model, not just for its own integration tests, but also as a powerful mean to validate the lower layers of software in a realistic environment. We hope that testing VLT common software on the model, will help to eradicate those software problems which are very difficult to spot in the stand-alone VLT common software tests, no matter how automatic or complete they are from their own point of view.

3. THE VLT COMMON SOFTWARE CONCEPT

The main foundation body for the VLT control software is called the *VLT common software*. It 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 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 is given in 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). Nowadays one can find structures, guidelines and recipes to write control software and examples including already the so-called standard commands, within the scope of the VLT common software.

The VLT common software is used across all on-line 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 its later integration into the VLT control software at the VLT Observatory.

4. RELEASES AND SOFTWARE TESTING

The VLT common software is used in the whole VLT programme, telescope and instruments, by ESO staff, Contractors and Consortia. It is distributed via a system of regular releases since more than two years and has been developed mostly internally at ESO.

The size of the VLT common software was 807 Klines of source code in the November 1996 release (including comment lines) and has then increased slightly in the May 1997 release. If telescope control software is included, the size of about 1 million lines of code is reached, to which test software should be added (a comparable amount of lines of test code). 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 on its turn is based on the *VLT Sequencer* - an interpreter based on Tcl/Tk, extended with all the necessary commands to interface to the VLT real-time database and message system (see Ref. 1).

The testing of all this code is of course a fundamental concern, even before it is released internally. This aspect has received

considerable attention since the early phases of development, as everybody was convinced of the importance and benefits of quality control, containing a.o. repeatable test procedures. Although the Sequencer was not conceived primarily for this purpose, the test procedures for the VLT software are largely based on it. The Tcl/Tk scripting language with the Sequencer extensions provides an excellent tool to glue different VLT software components together, or in other words, to interface these components with each other, and exercise them. This has led to the development of automatic regression test procedures, which cover the majority of the VLT common software, leading to automatic comparisons of test results with previously stored references. In spite of the non-negligible resources it requires, especially during initial implementation, we believe that only so we can produce releases which are properly tested.

The quality of the VLT common software has also benefited from the adherence to POSIX standards, and the policy to develop, port and test all common software on two distinct platforms, namely HP-UX and Solaris. Although this policy of portability to two distinct computer architectures was initially instigated to be really vendor independent, a clear side-effect is that during porting bugs can be revealed which normally remain hidden on one particular platform due to (sometimes undocumented) behaviour of compilers, linkers, libraries etc.

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. 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. E.g. applications could misinterpret protocols or formats of the messages they receive, while the message system itself worked flawlessly.

Experience with the NTT has indeed demonstrated that the current level of software module regression tests is not realistic enough to sort out all software integration problems in a complex environment with several workstations and some twenty LCUs. We hope now that an adequate level of testing for VLT can be achieved on what we call the *VLT control model*, a VLT-like computer network without real telescope hardware on which most control software can be tested. More on this subject is explained in the paper on “Integration tests of the VLT telescope control system”, presented separately at this Conference (Ref. 4).

5. INTEGRATION WITH THE 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, from proposal preparation to data archiving and data reduction. This integration has not only been planned but also achieved, although in a prototype form at the NTT. A separate paper (see Ref.5) at this Conference explains the VLT Data Flow software, which is under the responsibility of the DMD Division of ESO.

5.1 The view of the VLT control software

When the VLT Instrumentation software concepts were conceived as part of the VLT control software, it was stipulated that the Instrument software engines should basically deal with a single exposure only, one at a time. The parameters needed to define any such exposure are stored in *set-up files*, which constitute the most direct interface to operate the VLT. The control software for all instruments needs to include graphical user interfaces to edit, display and save these set-up parameters. It also needs to have a set of standard commands e.g. to execute set-ups and start an exposure.

With the addition of the VLT Sequencer, as an external process on top of this, sequences of commands can be collected into Sequencer scripts; this scheme can be used to obtain more complex scenarios than just the single exposure mentioned above. Indeed, one can think of *standard sequences* for sets of frequently used commands or particular modes of operation.

5.2 The high level view: observation blocks

An essential role in the Data Flow is played by *observation blocks* (see also Ref. 5). An observation block is the smallest observational unit for the VLT. It is a rather complex entity, containing all information necessary to execute sequentially and without interruption a set of correlated exposures, involving a single target, i.e. a single telescope preset.

Each observation block contains one or more *templates*. A template is a script, dealing with the set-up and execution of one or more exposures of a particular kind, e.g. calibration or dark current exposures. It is of course from the VLT control soft-

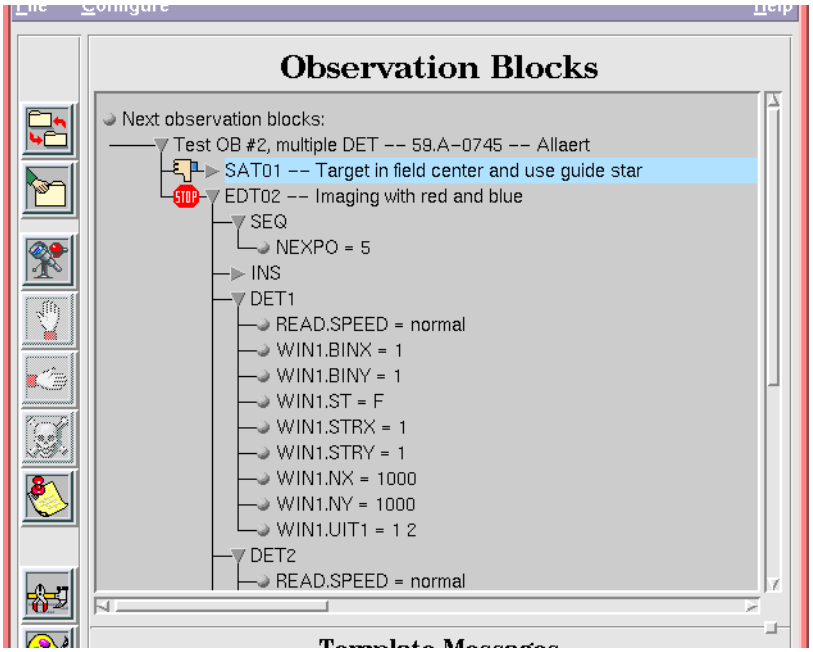

Figure 2. An example observation block loaded into Bob. It is (partially) expanded, and some flags have been set.

ware's point of view nothing else than a Sequencer script. The exact parameters for a template, e.g. the exposure time, is part of the information contained in the observation block, as are additional requirements for scheduling and data reduction. These parameters are defined by the astronomers during Phase II Proposal Preparation, while building their observation blocks with the tools provided for that purpose.

5.3 The VLT Control software integrated into the Data Flow world

Based on the rules listed in Sect. 5.1, which were incorporated into the VLT Instrument Software Specification, the control software for several instruments was already in different stages of development by the time the Data Flow concept of observation blocks was finalized. In order not to impact these projects, a new interface layer between the Data Flow software and the VLT control software was developed as part of the VLT common software. It consists on the VLT control software's side mainly of a single process, called *Broker for Observation Blocks* (or short and affectionately *Bob*). This process requests and receives from the short-term scheduler the observation block to execute next. It then breaks this observation block down into its constituent templates, and tries to execute them one by one. The templates being nothing else than a particular kind of Sequencer scripts with accompanying parameters, the VLT control software knows how to handle it from there on.

Bob has been implemented using the Sequencer (or Tcl/Tk) scripting language, partially with the help of the Panel Editor. It constitutes one of the main graphical user interfaces of the instrument operators at the telescope, showing the high level view of operation. With the help of this user interface, observation blocks can be loaded, executed, skipped, paused, aborted, etc. During execution the status of the individual templates is graphically indicated, and a log of their communication with the low level control software can be displayed. A partial screen dump of Bob can be seen in Fig. 2.

It is important to realize that much effort and emphasis needs to go into the development of templates for every instrument. These templates belong to the instrument just like any other piece of its control software. Although astronomers cannot edit these templates themselves, they can of course set the related parameters while they are building observation blocks as explained in Sect. 5.2.

One of Bob's tasks is to return the status of the currently executing observation block. It does so at the level of the individual templates: it will inform the Data Flow world if a template was e.g. started, paused, aborted or finished. This way the data reduction pipeline and quality control processes do not have to wait until the entire observation block is finished before starting

their analysis on the astronomical data, while the scheduler can monitor the progress of an observation block and estimate when the next one will be requested.

Trivial type and range checks on template parameters can be exercised as soon as they are entered during the preparation of observation blocks. But it is of course required to have a more complete check involving the combinations of parameters and the "real" instrument software and database, yielding a better confidence in the correctness of observation blocks. For that purpose Bob can also run in non-graphical mode. Together with the instrument software in simulation mode this is the best verification one can obtain without really executing the observation blocks at the VLT itself.

Although Bob was initially designed to cooperate with the short-term scheduler, it was quickly realized that a more off-line form of operation could be very useful. For that reason Bob can now also read in observation block descriptions from files, without having to communicate with the Data Flow world. This is particularly appreciated by the engineers, who during the later phases of the development process need a high level tool to exercise their software. While focusing on their control software job, they are often not really interested in the Data Flow, but still want to use the same tools as the ones available at the telescope. This off-line mode of operation is of course also very practical during the implementation and testing of templates.

6. STATUS REPORT ON CONTRACTED SOFTWARE

Several parts of the VLT software will eventually be contracted out, if all the instruments to be developed by Consortia of European Institutes are taken into account. For the main telescopes, still excluding the VLT interferometer, three subsystems have been contracted to Industry, including hardware and software. These are the telescope enclosures, the primary plus tertiary mirror subsystem (m1m3) and the secondary mirror subsystem (m2).

We had in this area definitely some overhead and difficulties due to the approach chosen. On the one hand the VLT is a very homogeneous and tightly connected control system. This has no doubt advantages in maintenance later on, but has forced ESO to distribute its VLT common software to Contractors. While this has given surprisingly few problems (thanks to the application of standards, quality control and mainly previous validation on NTT), it has given though a very rigid frame of software and rules to industry, interested more in producing a product (where software represents a small fraction of the total value) in given short times than in learning and using the ESO VLT software. While working with Industry and Software Houses remains possible, it is surely recommendable in a situation like the one of ESO, to directly allocate the implementation of the Control software for subsystems either to Software Houses working under direct ESO control or to software consultants working in house. The latter is obviously by far the most flexible arrangement.

Collaboration with Consortia has proven easier in the above sense. Consortia of Institutes, which typically are smaller than ESO, showed normally a lot of interest in following certain standards, particularly when embedded into existing software provided by ESO. Otherwise, differently than Industry, a certain resistance was encountered in applying to a full extent the documentation and review levels used at ESO. The same will presumably apply later to test procedures, for what one can foresee today. There were also problems sometimes in the areas when no precise definitions existed yet, e.g because ESO had not yet needed a concept or had not yet worked out the details on an in-house implementation. This was somehow inevitable, as certain concepts at the interface with Data Flow software, which have an impact on instruments, could only be defined very late in the life of the VLT project.

At this stage of the development, with so many different applications developed using the VLT control software, it is definitely easier and more cost effective to develop new applications in house, possibly with the help of software consultants when not enough resources will be available. We plan to proceed in this way for the VLT interferometer.

7. VLT SOFTWARE SUPPORT TEAM

The success of the VLT software commissioning at Paranal, a very remote and hostile site, must also depend a lot on the team ESO will have on site. Earlier experiences at the La Silla observatory had already demonstrated the importance of a proper training of the team that will finally be responsible for the software maintenance and operation, well in advance of the actual installation of this software. Given the complexity of the VLT control software, the need for training staff well in advance was considered all the more important.

At present a team of six Chilean staff are being trained with a program foreseeing that they take part during two years to the life of the development team. Their work is not only of support, but also of participation to the development effort, although

with emphasis on the need of gaining a good overview of the whole control software. In spite of the fact that these staff have only been at 50% of their time at ESO's headquarters in Germany and the other 50% of the time in the Santiago offices, they have been capable of integrating quickly with the rest of the VLT software group and to contribute to the control software. Also, during their stay in Santiago they form a unit which is independent from the other La Silla and Santiago teams; together with the time difference between Germany and Chile this creates a good opportunity to train their independence and self-reliance.

At this stage, when the first ones are soon starting to work in shifts at Paranal, we are sure that their experience and knowledge of our way to work, will enable them to be an effective front-end to the Observatory needs and to solve immediately the most urgent problems at Paranal. Still the rest of the support will be provided from Europe, which will rely for this on a high bandwidth dedicated satellite link.

8. CONCLUSIONS

The VLT software is ready for commissioning at Paranal. It was explained in the paper how testing VLT common software, using the VLT control model and using all the available feedback from NTT are all necessary components to build up the confidence we have to be ready. Obviously commissioning with real VLT hardware has still mostly to come. We do believe however that the system of releases and procedures which is in place will give us a solid frame to proceed.

Integration and commissioning in a very uniform system like the VLT, where instruments are also included, will still not be easy. In particular the fact that parts of the telescopes and many instruments are developed outside ESO, might give surprises in the expected behaviour and integration problems. Still we do believe that the big initial effort in creating a tight and homogeneous system, must pay off in terms of maintainability of those about 160 LCUs and 50 workstations, which will eventually form the VLT control system. We actually do not know of a better way of maintaining so many systems, than to keep them all the same as much as possible via a comprehensive VLT Common software. This we have done, distributed for 2.5 years now and we think it is a valuable experience to recommend anyone embarking on a similar project. We have seen later that this gave us also the advantage of being able to absorb modifications due to the introduction of the Data Flow end-to-end concepts, in a way mostly transparent to previous developments and with a limited impact on instrumentation software.

People who want 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 authors wish to thank all colleagues in the Software Group and NTT team of ESO, who are really the authors of most of the Control Software mentioned and described here. Their engagement has made the VLT common software and NTT upgrade possible and will now be essential during VLT commissioning.

REFERENCES

1. E. Allaert and G. Raffi, "The VLT control software - local and remote support for operations by astronomers and operators", *ASP Conf. Series* **87**, pp. 192-195, 1996.
2. G. Raffi and K. Wirenstrand, "The VLT control software in its final test phase", *Proc. SPIE* **2871**, pp. 996-1004, 1996.
3. A. Wallander and J.Spyromilio, "NTT project: a field test of the VLT software and hardware", *These proceedings.*
4. G. Chiozzi, K. Wirenstrand et al., "Integration tests of the VLT Telescope control system", *These proceedings.*
5. M. Albrecht, M. Peron et al., "VLT Data Flow System: the NTT prototype experience", *These proceedings.*