VLT Telescope Control Software installation and commissioning.

K. Wirenstrand, G. Chiozzi, R. Karban

European Southern Observatory
Karl-Schwarzschild-Strasse 2, 85748 Garching, Germany
e-mail: kwirenst@eso.org, gchiozzi@eso.org, rkarban@eso.org

ABSTRACT

Out of the four VLT Unit Telescopes, three have had first light and of those, one is in full scientific operation. So the VLT style Telescope Control Software is in regular operation on three out of four originally foreseen telescopes, and is being installed and tested on the fourth. In fact, it is actually installed and successfully operating on a whole family of ESO telescopes: the NTT, the 3.6m La Silla, the Seeing monitor telescopes on Paranal and La Silla. That means that this software is installed and running on a total of eight telescopes of different kinds, for the moment. The three Auxiliary telescopes of the VLT interferometer and the VLT Survey Telescope, now in the development phase, will join the family.

Another member of the family is the Simulation Telescope, called "Control Model", at ESO headquarters in Garching. Although it cannot look at the sky (it is pure electronics and software: no mechanics and no optics) it has been, and still is, of great value. It can be reconfigured to emulate any of the actual telescopes and it is used for off-line testing of new software releases and to analyse and fix problem reports and change requests submitted from observation sites, without disturbing operation. It is also used to test instrumentation software and its interfaces with the TCS before the actual integration at the telescope.

This paper presents the actual status of the Telescope Control Software on the VLTs and on the other telescopes. The main focus is in the characteristics that make the TCS architecture portable on very different mechanical and optical configurations. Then the paper concentrates on the strategy used throughout the projects for integration and testing of the modular components. Particular attention is given to the installation and commissioning phases for the VLT telescopes.

Keywords: software, real-time control, testing

1. INTRODUCTION

When the development of the VLT Telescope Control Software started, it was, of course, only intended to be used for the VLT 8m telescopes (Unit Telescopes). However, in an early phase, it was decided to use the NTT on La Silla for the very first installation; the main idea being to reduce test time on the VLT later on, but at the same time also to make a necessary upgrade of the NTT. When installation and test on the VLT started, just after the upgraded NTT was operational again, the work was of course much easier than it would have been, but it still required much effort to get it going. For generel information about the VLT project, see ref. [12].

Following the success at the NTT, and the first good results for the VLT, also the upgrade project for the ESO 3.6m jumped on the VLT standard line. When it was time to discuss the seeing monitor telescope for Paranal, which is an upgraded copy of what the La Silla one was at that time, it was decided to completely re-do the software, and make both software and control electronics VLT standard. Substantial parts of the VLT telescope software is used for this small (35 cm) alt-alt telescope; for more information about the seeing monitor, see ref [8].

Now momentum is gained, and more telescopes are being developed; it is natural that the control system for the three VLTI 1.8m Auxiliary Telescopes is completely VLT standard, and that it will re-use most of the existing software; see ref [5]. New parts are of course being developed along the same line. In fact, some parts of the Telescope Control Software will be used also in parts of the VLTI control software that are not directly doing telescope control, since the structure of it invites to re-use for other applica-

tions.
Very recently, two completely new projects have started that will use the VLT TCS standards where applicable; this goes for hardware, software and procedures. They will be developed outside ESO, but the telescopes will be installed on Paranal and maintained there by ESO, so obviously standardisation is very important.
The main important factors, that allows the porting of the software to this variety of other telescopes (equatorial, alt-alt, small, big, old, new) are discussed in the following sections, 4 and 5. Also very important for the development of software, in particular for testing and verification, is the use of the test setup in Garching, called the Control Model, described in section 6.

2. LOOKING BACK or "THE BIG BANG"

When the specification of the Telescope Control Software for the VLT started, nobody thought of putting this software on any other telescope than the VLT unit telescopes. Then, at an early stage of the development, it was decided to upgrade the NTT telescope with the VLT software, long before it was to be installed on any VLT telescope. The purpose was twofold:

- an upgrade of the NTT was to be done anyway, mainly for hardware reasons but also for functionality upgrade
- the NTT was an excellent test-bench for the VLT; it is an alt-az telescope, it has a distributed control system which in many aspects is similar to that of the VLT telescopes (because many of the ideas and standards for the VLT came from the original installation of the NTT!)

After some initial tests of individual subsystems, which was much facilitated by the properties of the control system that will be discussed later, from mid 1996 the complete NTT system was upgraded and tested for almost a year; the "Big Bang" period. Much effort was put into testing, re-writing and re-designing of software and, not less important, of handling procedures. In the end, quite a stable system was given back to the eagerly waiting and curious astronomers. The system now has a much improved usability. The upgrade allowed installation of the modern observation handling and data flow systems. An additional advantage is the strong similarity with the 3.6m telescope, after the upgrade of that one. A considerable manpower effort was put into the project, but we believe that this very much increased the quality of the control system of the NTT, and in particular, it is absolutely clear that this saved a LOT of telescope test time on the VLT. And that was the main purpose of the exercise!
Just after NTT was in operation again, telescope software tests started on Paranal, in August 1997. The testing was a bottom-up approach, with several subsystems being tested in parallel; adapters, M2 unit and M1 cell were tested in a separate maintenance building before being installed on the telescope. When the altitude and azimuth axes were complete and could be moved, i.e. all hydraulics, interlocks and electronics was in place and tested, but before any mirror was installed, we mounted an 8 inch Celestron on the centerpiece, put a guiding CCD on it and made pointing and tracking tests, in March 1998. At that point, it was a comfort knowing that there was no expensive mirror in the telescope! We did not yet have the confidence in the system that we have now (or, at least, some mechanics and optics people didn’t!). With this piggy-back setup, the system could be tested almost up to the point of performance required for First Light, which was achieved May 25-26 1998.
During this period of installation testing on the first VLT telescope, some of the design properties of the software really started to pay off. The parallel and independent subsystem testing, the hierarchical structure of functionality and other issues did greatly, we believe, facilitate the testing.

3. ACTUAL STATUS

3.1 Installations

The first of the four VLT telescopes (ANTU) is in full scientific operation since 1.4.1999. The second (KUEYEN) and the third one (MELIPAL) saw first light 1.3.1999 and 26.1.2000 resp., and are now in the commissioning phase. The fourth VLT telescope (YEPUN) is in the integration phase, and telescope control tests have started. The names of the telescopes are in the mapuche language, a native Chilean language, and their meaning is

- ANTU: The Sun
- KUEYEN: The Moon
- MELIPAL: The Southern Cross
- YEPUN: Sirius

A substantial part of the telescope control software integration and commissioning work for the VLT Unit Telescopes is now done, and the software for these telescopes is entering a maintenance phase. This is also the case for the ESO 3.6m telescope that was upgraded to VLT standard during 1998, as well as for the two ESO Seeing Monitor telescopes, parts of the Astronomical Site Monitors for Paranal and La Silla, which use the core parts of the VLT TCS software. The NTT, the very first telescope to use this software, has recently been upgraded with the latest release of basic VLT software
But the development continues for other telescopes! The VLT TCS software is being used and extended for the VLTI Auxiliary Telescopes and for other parts of the VLTI software, developed in-house, such as the Siderostat which is a "tracking periscope" to be used to test the VLTI delay lines. It will also be used as the base for the VST, VLT Survey Telescope, see ref [7], which is being developed by the Observatory of Capodimonte in Naples, Italy, as well as for the british VISTA telescope, also going to Paranal. The VST and VISTA projects have just started.
These different users of the VLT TCS software represent telescopes of different kind and in different state of development and operation. Different versions are used (where necessary) and new functionality is being introduced (where necessary), and at the same time the original installations are becoming more and more stable and reluctant to changes. But still there is one VLT First Light to go, and there are many new and complicated instruments coming to Paranal the next few years, and all this means more work to maintain the telescope control system.

3.2 Technical status

The foreseen functionality of the TCS is in place and it is performance verified, with a few exceptions: as mentioned above, we have e.g. not yet verified that the Field Stabilisation performs to spec also when running in parallel with chopping. In "normal" cases, however, tracking and Field Stabilisation is verified, see ref [2].
Pointing is good for all practical purposes, stable and reliable, and when operationally needed, a quick re-calibration can be done completely automatic; see ref [2] and [4]. This is sometimes necessary after operations on the hydraulics pads. The rms value for pointing is usually slightly above specification (1 arcsec rms over the complete operational area), but in practice this is good enough.
From the point of view of installation and commissioning of the software on VLT telescopes, we have now reached a point of "serial production". The second VLT was much easier and faster than the first, and number three was "off the shelf" or "Plug&Play". We just installed the workstations, built the TCS, installed it, and it just worked. Of course, all the work of fully commissioning the telescope needs time and systematic work, but that's a different story.
The documentation, in particular the design documentation, needs to be updated. For long periods of time, work was done under pressure and with tight deadlines, in particular on Paranal. This made it difficult to keep design documentation aligned with real implementation. We are aligning now, a posteriori, while making design work for other projects, and we are trying a new development methodology that would alleviate the problem; see ref [5].
The one area where upgrades and improvements are done all the time is for test and maintenance tools; not only, and not mainly, for software maintenance but for electronics, hydraulics etc. A very general and very useful tool is a web-based data presentation system, which retrieves and presents data from the nightly archived operation logs.The operational data available includes telescope and instruments data, as well as meteorological and seeing data. The product is presented in ref [3].

4. PORTABILITY ASPECTS

It was an initial design goal to have the software easily changeable and configurable for the four Unit Telescopes, with their possible small differences. Also, it was clear that a good design should allow re-use of individual components on different parts, and for different subsystems of the telescope. Another important aspect is due to the multitude of equipment and subsystems. Telescopes and subsystems, such as adapters, mirror cells etc., would be produced, tested, installed and commissioned, over a long period of time, with possible modifications being implemented on the way, and with periods of time with many units being tested and integrated in parallel. To make all this testing easier, it is important that subsystems can be tested as independent as possible, and as realistic as possible.

Based on these general design goals, the main aspects of the software design were considered to be Modularity, Distribution and Version control:

- **Distribution** simplifies independent testing of different parts of the telescope.
- **Modularity** simplifies exchange, modification and test of individual building blocks.
- **Version control** is necessary to keep track of versions and changes

4.1 Distribution

The concept of Distribution applies to both control hardware and software. The hardware part of it is fairly obvious: each subsystem has its own control electronics, located on or near to the unit. There are several such subsystems per VLT telescope. This has been described in e.g.ref [6] and ref [10].

The software distribution principle has been to have the level of software in a local control unit as high as possible, yet strictly related to the subsystem. All software that is needed to run the functionality of a certain subsystem should run in the local unit. The only exceptions are functions that coordinate operations of two or more subsystems, like the tracking of the telescope, or that cannot easily be run on the local unit, like e.g. User Interfaces.

As an example of this distribution principle, let's look at the functionality of tracking. The software for tracking is like many other modules split in two parts: a coordinating part running on the workstation, and a local part running on LCUs. The workstation part is coordinating the tracking of the complete telescope, and is the interface to other modules; presetting, User interface etc.

It distributes sky coordinates and other data to the tracking LCUs, and collects data from these. The LCU part of tracking runs in the same version in each tracking axis LCU, i.e. altitude, azimuth, adapters and rotators. Figure 1is an illustration of the structure of the tracking software.

The LCU part inputs mean (RA,dec) and outputs completely calculated, corrected and compensated altitude, azimuth and field rotation angles, the same set irrespective of in which LCU it is running. Though the tracking of the telescope is a coordinating task that is not performed in a local control unit, the tracking of a single axis IS a local control task, and all that is needed for this local task is running in the LCU. That includes the complete coordinate conversion. This means that conversion from sky coordinates to alt/az coordinates is done in 3 LCUs in parallel at any one time when tracking: altitude, azimuth and one of the three adapters. But it is done in different computers and in parallel, so the CPU load doesn’t add up, and having it distributed saves a lot on LAN traffic and removes real-time requirements on the LAN, compared to having to keep continuously changing coordinates updated over the network.

To actually drive a tracking axis, further modules are needed: the axis position servo, encoder module etc. This is described in the following section

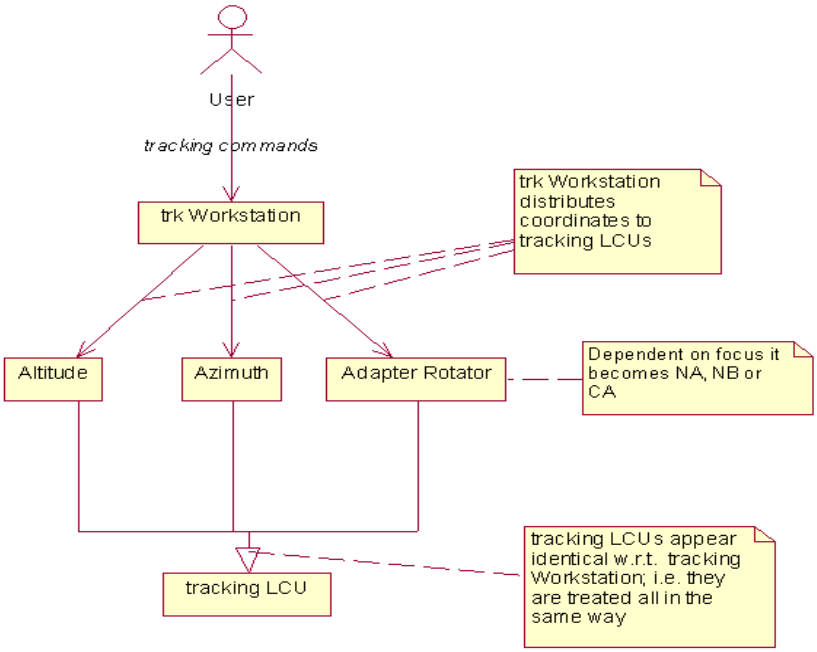

Figure 1. Layout of tracking software

4.2 Modularity

Most software developers are of the opinion that they produce "modular software". It all depends what you mean!
The VLT TCS software modularity is based on making modules that are related to physical parts: motors, axes, probe, adapter, etc.,or to functions related to physical parts: tracking, servo, etc.
To illustrate how the modularity or building-block structuring is done, again let's look at the functionality of tracking a telescope axis, including all that is necessary to drive the axis. A standard tracking axis consists of the following building blocks:

1. input of sky coordinates
2. conversion of sky coordinates to axes coordinates
3. output of axes coordinates to position servo
4. the position servo loop
5. encoder handling
6. output to amplifiers
7. handling various I/O signals, like interlocks, limit switches etc.
8. accurate time keeping

Figure 2 shows the blocks with their relations; each block being one or two software modules.

An important aspect of the modularity is to keep the interfaces between blocks general enough, but well related to the functionality at hand. These interfaces are of course much harder to change than anything internal to a block.
The time keeping is based on the distribution of accurate absolute time, interfaced in the local controller with the TIM board; see ref [11].

The tracking module ’trk’ handles the items 1.-3., in a generic enough way, that it can easily be used for any telescope, independent of mounting, hardware used, encoder etc.; the ’trk’ module will e.g. be used for the small XY-table which carries the guide detector for the Auxiliary Telescopes. The ’trk’ module passes its output data to a function with well-defined interface that is then responsible to do whatever necessary to pass the information to the next level, the position controller. The actual implementation of the function is supplied by the position controller module of the device that is tracking. This function is responsible for converting alt-az coordinates to xy-positions in the case of the XY-table, to phi-theta for the seeing monitor etc.
All encoder functionalities are handled in separate modules; reading, initialisation, calibration etc. Start-up functions and the servo loop communicates with the encoder via standardised interfaces which allow for a relatively easy change of encoder behaviour (as long as it sticks to the interface).

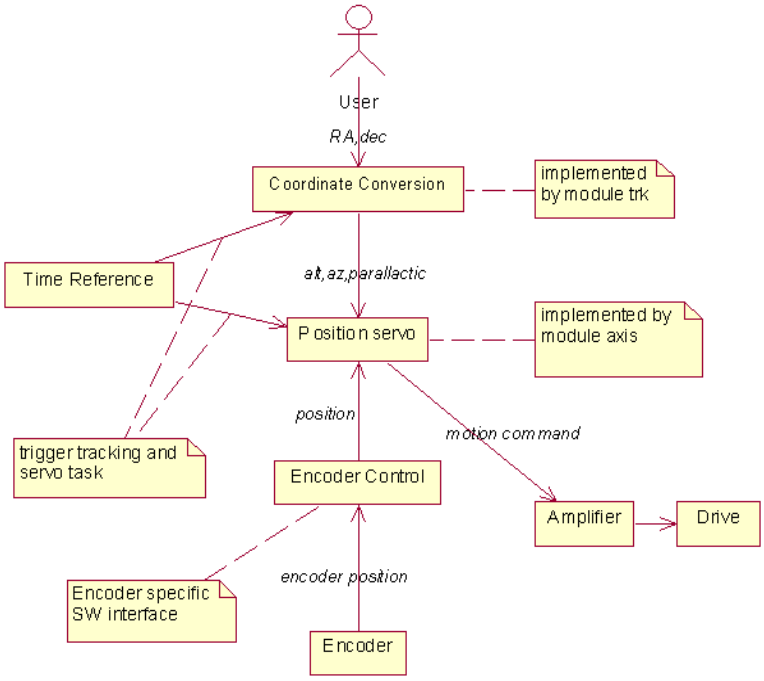

Figure 2. Standard tracking axis LCU software layout

There are a number of software properties in addition to the split up in modules that contribute to modularisation:

- configuration data are in the database, rather than in code. This makes it easy to re-configure at software installation time, or even in some cases at run-time. This applies to data like scale factors, limit values, availability flags etc.
- configuration data is archived, and thus under version control.
- the database builder is working in an object oriented way, which greatly simplifies re-use of data structures
- Where differences in code is unavoidable, a C++ subclasses scheme is used, implementing the differences in separate sources in a separate module, but inheriting all similarities. This is used e.g. for the adapter part of autoguiding: the NTT adapter is completely different compared to the VLT ones, so there is a an NTT subclass in the module handling those differences. The same technique is used for the Field Stabilisation module, which consists of a few subclasses that implement the differences to autoguiding.

4.3 Version control

The VLT standard software is used for several telescopes and instruments, developed inside and outside ESO. The same is true for the TCS software. Without a strict version control system it would be impossible to maintain this in a reasonable way, to keep track of (small) differences and to be in control at any one time of what is actually running on a particular system. The main components of the version control system are:

- the archive. Software modules are archived (the complete history of versions!). Anyone can make a read-only copy of a module, but only one user at a time can check out the module for modification.
- the Software Change Control Board. Proposals for changes, be it bug fixes or modifications, are discussed and decided upon in the SCCB before implementation changes are done. The written proposals with the added comments and decisions are archived, to keep a history of modifications.
- the tcsBUILD module. This is a module, which is archived like all other modules, in which is defined which module versions and configuration data that are used for each telescope. This allows to build a complete TCS for any specific telescope directly from the archive and on an off-line machine, and then just copy a few files to the target machine to have a system up and running.

For more details, see ref [9].
The rules of the game concerning version control are formal and correct, but flexible enough to allow a quick reaction to a problem on a running telescope/instrument, and a smooth cooperation between sites (Garching/Paranal). We were using a system of changing "ownership" of modules from time to time between Garching and Paranal.

5. TESTABILITY ASPECTS

For a software system as complicated as the VLT Telescope Control System, it is necessary to design for testability. One of the most important aspects in this sense is the modularity, i.e. the possibility to test separate parts realistically and independently of other parts. The distributed nature of the control system, both in hardware and software, as described e.g. in ref. [10], was an absolutely necessary condition for the structured testing. The philosophy of letting each electro-mechanical unit have its own controller, with a software level as high as possible, has allowed important independency in testing on each hierarchical level. Thus it is possible to test adapters with all their software and all their modes in an assembly hall, using only its Local Control Unit (LCU). Since also the tracking software is part of the LCU software, also the highest level functions can be tested with the normal software, i.e. without having to go to special test setup. Also the main axes of the telescope can be tested and tuned, also the tracking mode, completely independent of the rest of the telescope, and independent of coordinating software, since also these LCUs contain the coordinates
calculation and tracking software.
Structured testing is also made easier through the use of the command system to trigger actions. The various layers in the hierarchy have their own commands, some of which are pure test and debug commands, and others are the ones that are actually used from higher levels. A lower level function doesn’t see the difference if a particular command is manually entered or if it is coming from a higher level software function.
All communication between instrument control software and the telescope takes place in one module only (tif). This module supplies a run-time command interface, database classes to be included in instrument control databases, and data access functions. These interfaces hides all TCS internals for the instrument, and for the TCS itself it doesn’t add any complication whatsoever, since it is just using the commands that are anyway used internally, and no special logic is introduced in TCS for this interface (except of course in the "tif" module itself). As an additional support for interface testing of instrument software, there is a TCS simulation module which can be installed on an instrument control workstation "behind" the "tif" module, to simulate telescope behaviour.

6. THE "CONTROL MODEL"

An important part of testing of VLT software is done on the Simulation Telescope (Unit Telescope 0) at ESO headquarters in Germany.; see also ref [6]. This Simulation Telescope, usually called The VLT Control Model, is a setup with most of the computer hardware of a real telescope: workstations for instruments and telescope, most of the Local Control Units (LCUs) and LAN equipment. To simulate the main telescope axes, there are small test motors and encoders connected to the corresponding LCUs. CCD cameras for guiding and for Image Analysis are real, and they are fed with light ("star light") to allow realistic testing of guiding and Active Optics. For most parts of the telescope, however, there is no telescope hardware connected; there are e.g. no mirrors and no telescope except for the small motors! For the not available parts, LCUs and/or workstation software is run in simulation mode. The simulation can be done on different levels; on the lowest level, "only" the I/O signals are simulated, but on higher level complete functionality can be, or has to be, simulated.
All the VLT Telescope Control Software is installed, and the telescope can be run just like any other VLT telescope. Since the model includes also instrument control computers, complete "observations" can be done, with the instrument itself running in simulation mode.
The Control Model is an important test-bench both for basic VLT software and for telescope and instrument applications. When new releases of the VLT basic software are prepared, intensive testing and verification is done on the Control Model. The model is particularly useful for debugging, without losing expensive telescope time, and for testing interfaces between telescope and instruments. Instruments that have been carefully tested on the Control Model are integrated with "its" telescope control system on Paranal with very few problems.

7. PROBLEMS AND CHANGES

The previous chapters might for the inexperienced have given the impression that it was easy to develop the VLT TCS software and make it run on real telescopes. This would be the wrong interpretation! We are describing a very complicated system, with many possibilities to make mistakes (we have tried a few of them), with a long history and with many people involved. What we try to say is that we believe we have made a system that is easier to use than to understand; one that is perhaps not easy to use but easier than it would have been, had we not taken great care.
Of course we had many difficult problems to solve (and still have), but it is correct to say that since the start of installation on Paranal, we have not had any major, severe problem that e.g. would have made a complete redesign necessary. We underestimated the complexity in some cases, and we made a few minor conceptual changes, but nothing very dramatic. The handling of adapters and guide probes, with all the logic involved for different cases, was more complex than we thought, and quite some time has been spent on that. When looking back, we can also say that in a few cases the guide lines for implementation were not clear enough; some subsystems didn’t behave as expected in "standard" cases, and surgery had to be applied to correct for that. We also realise now that the policy for re-use could be better defined and better documented. This is becoming clear when outside cooperators are doing the implementation.
We had some problems with the interface between instruments and telescopes due to a slightly different way of implementation. This difference has historical reasons, call it "tradition" or "culture", but the problem is solved and the interface well defined and well working.

8. CONCLUSIONS

The VLT telescopes have already produced results exceeding our expectations, both from a scientific and from a technical point of view. The Telescope Control Software has done its contribution to this, within time and resource schedules. It is a complicated system, and much effort has been done by many people. The system is working well, not only on its originally intended target telescopes, but on a variety of different configurations. New projects are now being developed using this system, so development continues of the software, at the same time as the VLT telescopes one by one are applying a much more "conservative" and strictly organised upgrade policy.
We have described some of the properties of the software we believe have been important factors for reaching the present status. A number of problems and shortcomings have been identified, and they are gradually being corrected.

9. ABBREVIATIONS

3.6m	ESO 3.6m telescope, La Silla
alt	telescope altitude axis
az	telescope azimuth axis
ESO	European Southern Observatory
LAN	Local Area Network
LCU	Local Control Unit
M1,M2, etc.	telescope primary mirror, secondary, etc.
NTT	New Technology Telescope, ESO, La Silla
TCS	Telescope Control Software
VLT	Very Large Telescope
VLTI	VLT Interferometer
VST	VLT Survey telescope

10. ACKNOWLEDGEMENTS

The authors wish to thank all colleagues in the Control Software Group of ESO, and any other ESO staff, who have contributed to the concepts, ideas and software reported in this paper.

The VLT telescope control system is using software packages for coordinate calculations, and for pointing analysis, written by P.T. Wallace, Rutherford Appleton Lab. These packages have been used successfully also on the NTT and other ESO telescopes.

11. REFERENCES

1. G. Chiozzi, R. Karban, P. Dhoux, K. Wirenstrand, “Field Stabilisation with the VLT”
   These Proceedings
2. A. Wallander, J. Spyromilio, K. Wirenstrand, “Commissioning VLT unit telescopes: methods and results"
   Conference 4004 of this Symposium
3. M. Kiekebusch, J. Pavlich "Automatic report generator"
   These Proceedings
4. I. Muñoz "The VLT Pointing Module"
   These Proceedings
5. G. Chiozzi, P. Duhoux, R. Karban "VLTI Auxiliary telescopes: a full Object Oriented approach"
   These proceedings
6. G.Chiozzi, K.Wirenstrand, M. Ravensbergen, B.Gilli (1997) "Integration tests of the VLT Telescope"
   Proceedings of SPIE, vol. 3112
7. D. Mancini et.al "VST project: design overview"
   Conference 4004 of this Symposium
8. S. Sandrock et.al, "VLT astronomical site monitor: control, automation and data flow"
   These proceeding
9. G. Raffi, G. Filippi, "VLT software engineering and management"
   These proceedings

10. K. Wirenstrand, "VLT telescope control software: an overview"
Proceedings of SPIE, vol. 2479, pp. 129-139

11. M. Ravensbergen, "Time reference system of the VLT"
Proceedings of SPIE, vol. 2479, pp. 169-174

12. "The VLT White Book"
http://www.eso.org/outreach/info-events/ut1fl/whitebook/