Control system of the VLT Interferometer

Massimo Verola

European Southern Observatory (ESO)
Karl-Schwarzschild-Strasse 2, 85748 Garching bei München, Germany
e-mail: mverola@eso.org, ESO WWW: http://www.eso.org

ABSTRACT

During an interferometric observation the successful coordination of all the complex and heterogeneous devices of the VLT Interferometer[1,2] (VLTI) depends on the effectiveness and reliability of the control system, which carries out the ultimate system integration.

The VLTI Control System (VLTICS) is designed to satisfy both specific technical requirements and general operational constraints. It profits by the valuable experience gained during the development of the control system for the VLT 8-m Unit Telescopes, inheriting the same principles and standards. Since the VLTICS is a direct client of the TCS, the homogeneous architecture simplifies the interface definition and the data communication protocol. Therefore we can focus on the identification of those functions that are peculiar to VLTI and work out the most challenging requirements.

The VLTICS architecture is implemented and integrated within the framework of the VLT Observatory control environment. It relies on two hardware platforms: real-time VME control nodes and UNIX-based workstations. The VME nodes provide the interface to the controlled devices and run the data acquisition and digital control algorithms, while the workstations are used for coordinating and monitoring tasks, on-line database facility and graphical user interface. The standard control network is optical fiber Ethernet. ATM LAN links support image data transfer between the detectors and the workstation. In order to achieve the performance required by the fringe control loop, we have adopted the reflective memory technology, which offers a zero-latency deterministic data link with no software protocol overhead.

This paper describes the basic requirements and the overall architecture of the VLTICS, the software design method and the interfaces towards external systems and software clients. It draws also a parallel between VLTICS and TCS with respect to the main functions provided.

Keywords: control system, software, real-time control, control loop, fringe tracking, telescope, interferometer.

1. SCOPE OF VLTICS

The VLTICS provides the functions to control all the heterogeneous devices, which cooperate to deliver coherently, combined stellar beams to the VLTI Instruments. The VLTICS is directly responsible for the control of the elements that are used exclusively for interferometry: auxiliary telescopes, delay lines and other VLTI support equipment. It controls indirectly the Unit Telescopes through the external command interface provided by their Telescope Control System (TCS)[3]. The VLTICS interacts with its clients by means of a graphical user interface, designed for a human operator, and through a command interface, suitable for an observing run driven by a VLTI Instrument. The VLTICS relies on some other external systems for the following services:

- information about weather, seeing, clouds, seismic and other relevant environment variables, collected by the Astronomical Site Monitor
- information about the site, buildings, lights inside the laboratories and various support systems, provided by the Building Management System
- astronomical catalogs, observation scheduling tools, data reduction software, archive system, available from the Data Flow System.

Table 1 lists the elements controlled and interfaced by the VLTICS.

Direct Control	Indirect Control	External Interfaces
♦ Auxiliary telescopes ♦ Delay lines ♦ VLTI subsystems *VLTI UT Coude'* *Transfer Optics* *Fringe Sensor Unit* *Internal Alignment Units* Image Alignment Unit Pupil Alignment Unit Fringe Phase Alignment Unit	Unit Telescopes (through TCS) **VLTICS Clients** ♦ VLTI Operator ♦ VLTI Instruments *AMBER* *MIDI* *PRIMA* *Test Instrument*	Telescope Control System Astronomical Site Monitor Building Management System Data Flow System

Table 1. VLTICS operational environment: controlled elements, clients and external interfaces.

2. VLT CONTROL ENVIRONMENT

Since the early phase of the VLT Project, ESO has undertaken a major effort in defining and supporting standard hardware and software platforms[4], which constitute the foundation of the VLTICS design architecture.

The VLTICS architecture is multi-platform, distributed, network-based (see Fig. 1). Local Control Units (LCU), which consist of a VME backplane equipped with CPU and interface boards, perform real-time control of remote subsystems. UNIX-based workstations (WS) run high-level coordinating and monitoring tasks and the graphical user interface. Ethernet and ATM network links provide the communication layer among all the computing nodes. The Ethernet is suitable for exchanging control data. They are essentially command and reply messages between the WS and the LCUs, real-time values acquired by the LCU and transferred periodically into the WS on-line database, event and alarm notifications. ATM links are reserved for carrying bulky data, like images acquired by a detector and sent to the WS for early inspection, analysis and processing. A dedicated optical fiber Time Bus conveys the UTC absolute time, which is distributed by a GPS-based Time Reference System (TRS). The accuracy is better than 10 μs, including the maximum data transmission delay.

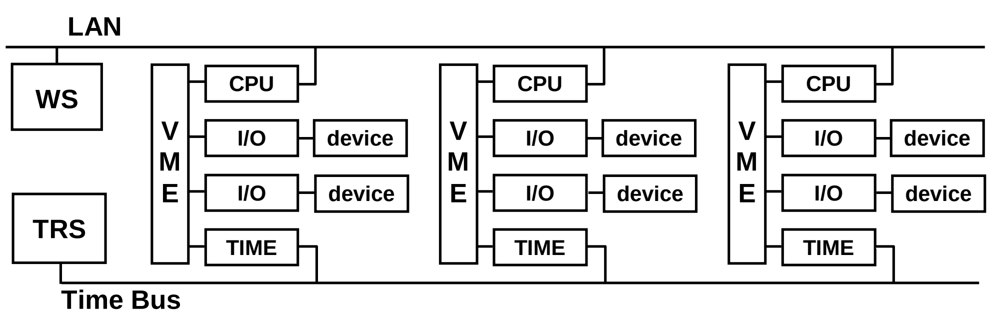

Figure 1. VLT control hardware basic architecture.

The control software is essentially constituted by a set of processes, running on LCU or WS. They interact exchanging messages (commands and replies) and sharing a common on-line database. The application software is based on the services provided by the VLT Central Common Software, which consists of a collection of libraries, servers and utilities, available both on WS (CCS)[5] and LCU (LCC)[6]. CCS is built on top of a commercially available software package, RTAP/Plus by Hewlett Packard, which provides inter-process communication and on-line database services. On the LCU platform, where RTAP/Plus is not available, the database is organized by a specific LCC software module, which offers compatible services. The on-line database has a hierarchical structure and represents always an updated snapshot of the system status. Each application can access a stored value by its name, independently from the physical location. CCS and LCC packages encompass all the common services needed to develop control software applications: on-line database, message system, log and error system, event and alarm system, time services, data sampling, software drivers, resource access control and booking system, command handling and syntax checking, simulation support, graphical user interface toolkit. The inter-process communication between computing nodes is based on ARPA/Berkeley sockets on top of TCP, while UNIX message queues are used for local data exchange (between processes running in the same CPU). Fig. 2 summarizes the control software structure.

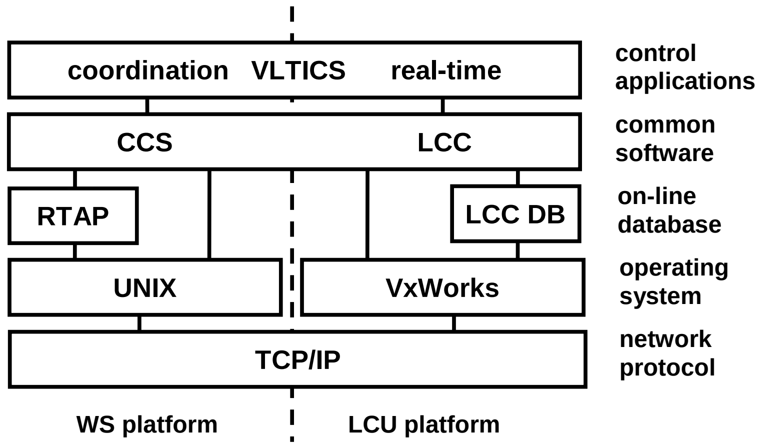

Figure 2. VLT control software: hierarchy of layers.

3. SOFTWARE DESIGN

The design method follows the classical "waterfall" model[7,8], where each phase generates the inputs for the next one (see Fig. 3). Their scheduling may overlap and there may exist some feedback from successive phases for corrections and further refinements. The products of every phase are carefully reviewed and documented. A clear test procedure is defined for each unit (HW assembly, software driver, software module) and often implemented as an automatic task.

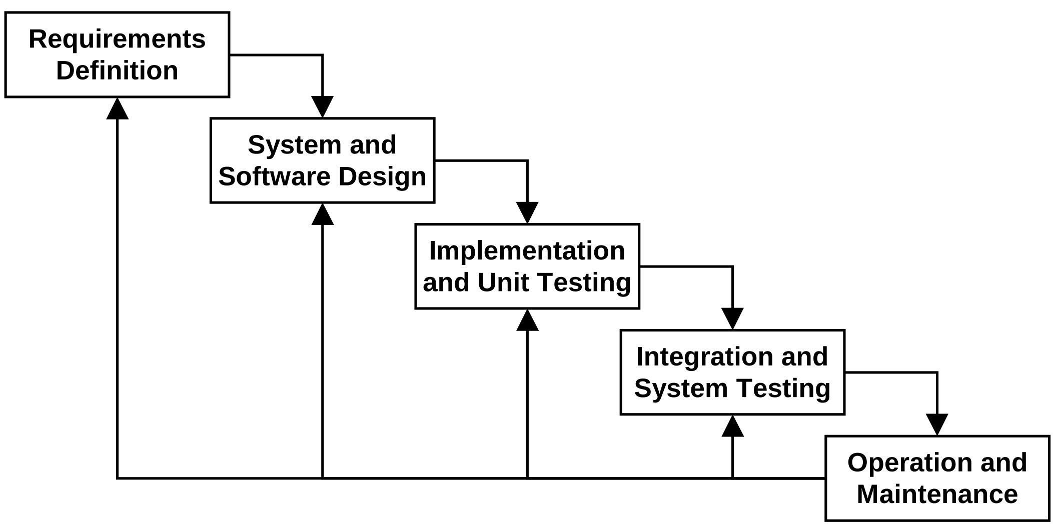

Figure 3. Software design method.

The software is organized in packages and modules. A package constitutes the first level of subdivision and groups all the functions related to the control of a main system component. A package is composed by a set of cooperating modules, which realize the actual control of specific items or carry out well-identified tasks. The module is the basic unit for the software configuration management. It can be implemented as a library or can correspond to one or more tightly coupled running processes. A control process responds to a set of common commands and a set of specific commands, occupies at any instant in time a well-defined state, and can react to events attached to values stored in the on-line database. The real-time software is written in C-language to guarantee a fast response during task spawning, function call and variable allocation. The object-oriented design[9] eases the development of event-driven control applications running on the workstations, which are coded in C++.

We have organized the breakdown of the VLTICS into four packages:

Interferometer Supervisor Software: provides the functions to operate the whole VLTI as a single entity, running global procedure like system start-up and shut-down, configuration setting for the telescope and delay line array, presetting the interferometer and executing the observation run. It represents the higher level within the VLTICS hierarchy. It is functionally located on top of the other packages and includes the user and the command interface towards the VLTICS clients.
Auxiliary Telescope Control Software: provides the functions to operate each 1.8-m telescope, both at subsystem level (low level control of each device) and at system level (coordinating tasks like presetting, field acquisition, tracking, autoguiding, field stabilization).
Delay Line Control Software: provides the functions to operate the array of all delay lines. It coordinates the operation of the delay lines and controls each of them, both at subsystem level and at system level (executing tasks like presetting, fringe searching, blind tracking, fringe tracking).
VLTI Subsystems Control Software: control software for other VLTI support devices.

Fig. 4 shows the VLTICS packages, their main interactions and the external interfaces.

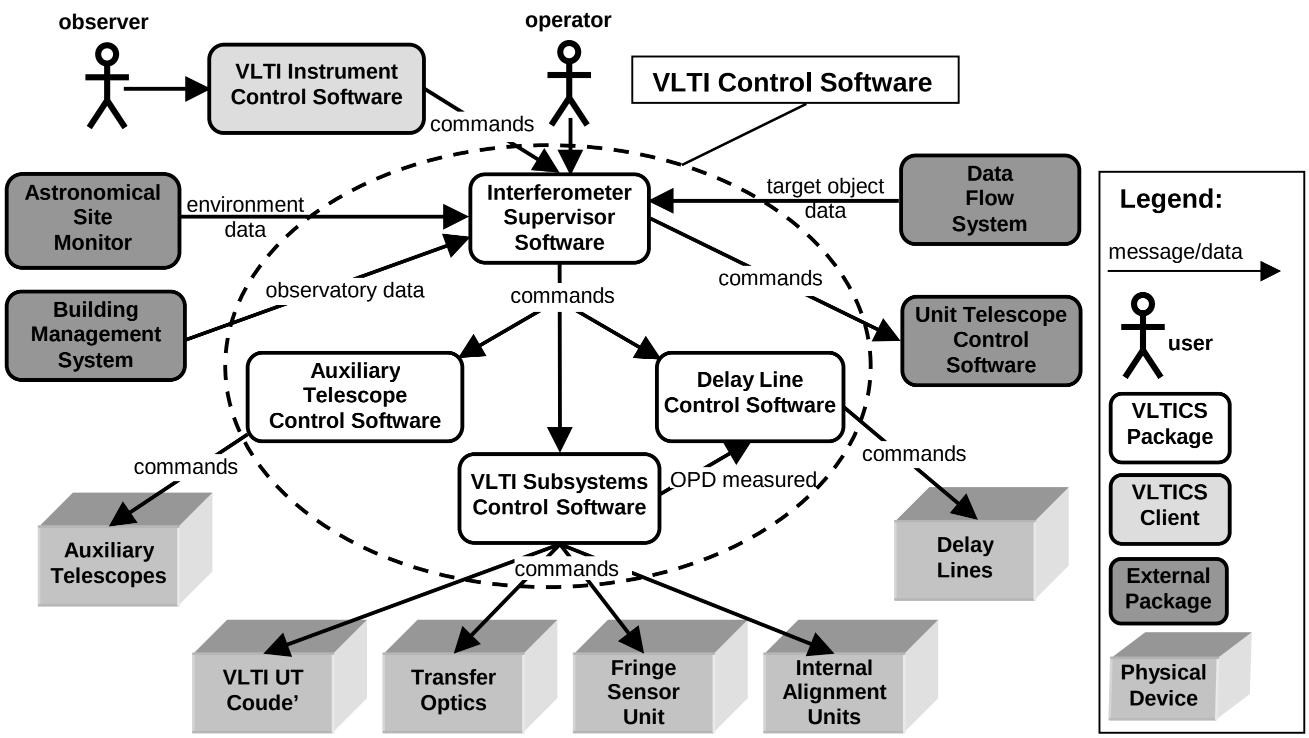

Figure 4. Top-level context diagram: VLTICS packages and interfaces.

The software module definition (see Fig. 5) exemplifies the next step in the refinement of the design process. The Interferometer Supervisor Software includes high-level functions to execute the observation, to monitor the VLTI status during daytime (stand-by), to run maintenance procedures (calibration, alignment, troubleshooting), to interface the VLTI Instrument and the human operator. It runs entirely on the workstation platform. The Auxiliary Telescope Control Software is created by adapting the TCS to the 1.8-m telescope. The software breakdown and most of the control functions remain preserved. The main differences occur in the low-level interfaces for data acquisition and control loop implementation. Two LCUs realize the delay line real-time control: the first one drives the coarse translation stage and the piezo element, the second one interfaces the Variable Curvature Mirror (VCM). The Optical Path Difference (OPD) Controller closes the fringe control loop between the Fringe Sensor Unit and the delay line (refer to section 5). The workstation software distributes all the necessary commands to initialize and position the delay lines, to assign a specific trajectory to each delay line, to start tracking and searching for fringe detection, to monitor and modify some tracking parameters (offset position, additional velocity, timeouts). Finally the VLTI Subsystems Control Software interfaces all the remaining sensors and actuators that are needed to carry the light beam from the telescope to the beam combiner in the interferometric laboratory.

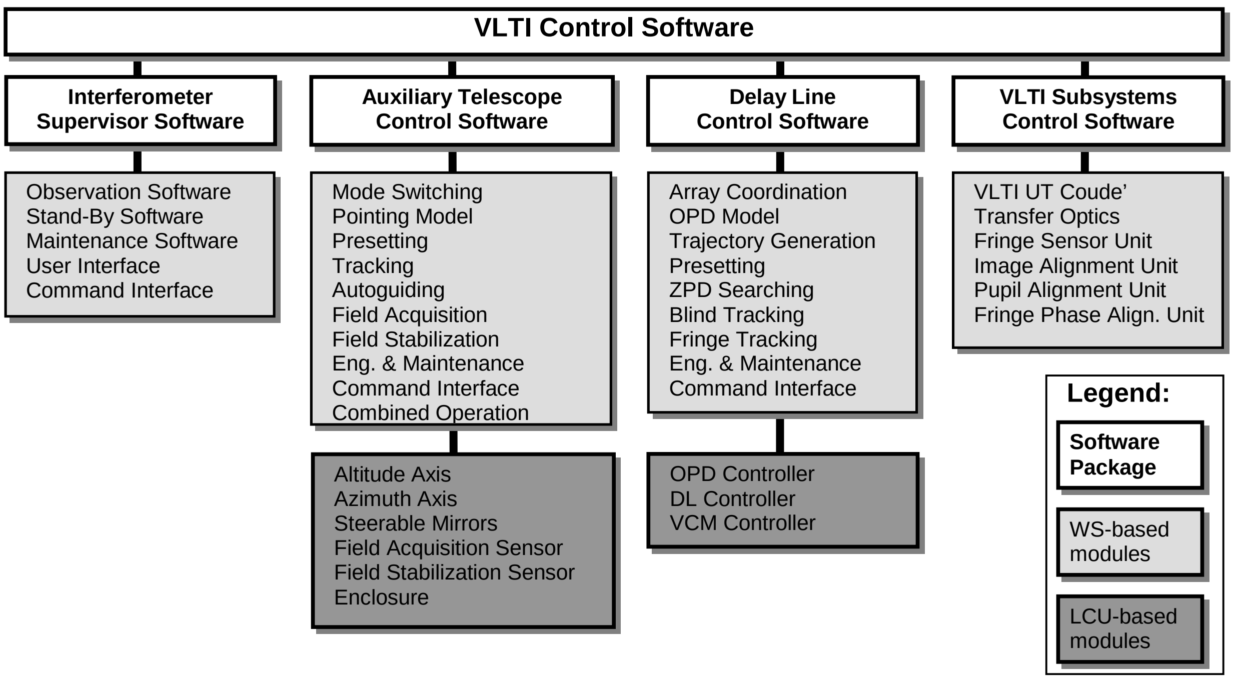

Figure 5. VLTICS software modules.

4. HARDWARE ARCHITECTURE

Fig. 6 shows the control hardware configuration. Each main system has a dedicated LAN that is connected to the VLTI backbone LAN by means of routers in order to isolate the local data traffic. The Ethernet LANs are actually implemented by using optical fibers, which connect each node to an Ethernet hub (star topology). The VLTI backbone LAN is realized by a network of ATM switches and is linked to the backbone LAN of the Unit Telescopes.

Four LCUs, a coordinating WS and two LANs realize the control hardware of each auxiliary telescope. The control LAN is dedicated to control data and connects all the computing nodes, while the video LAN carries image data and links the LCU interfacing the CCDs for data acquisition to the local WS, and to the VLTI Image WS through the router. The delay line LAN gathers the delay line and VCM Controllers, the OPD Controller and the Fringe Sensor Unit LCU. The reflective memory network ties together the LCUs participating in the fringe control loop.

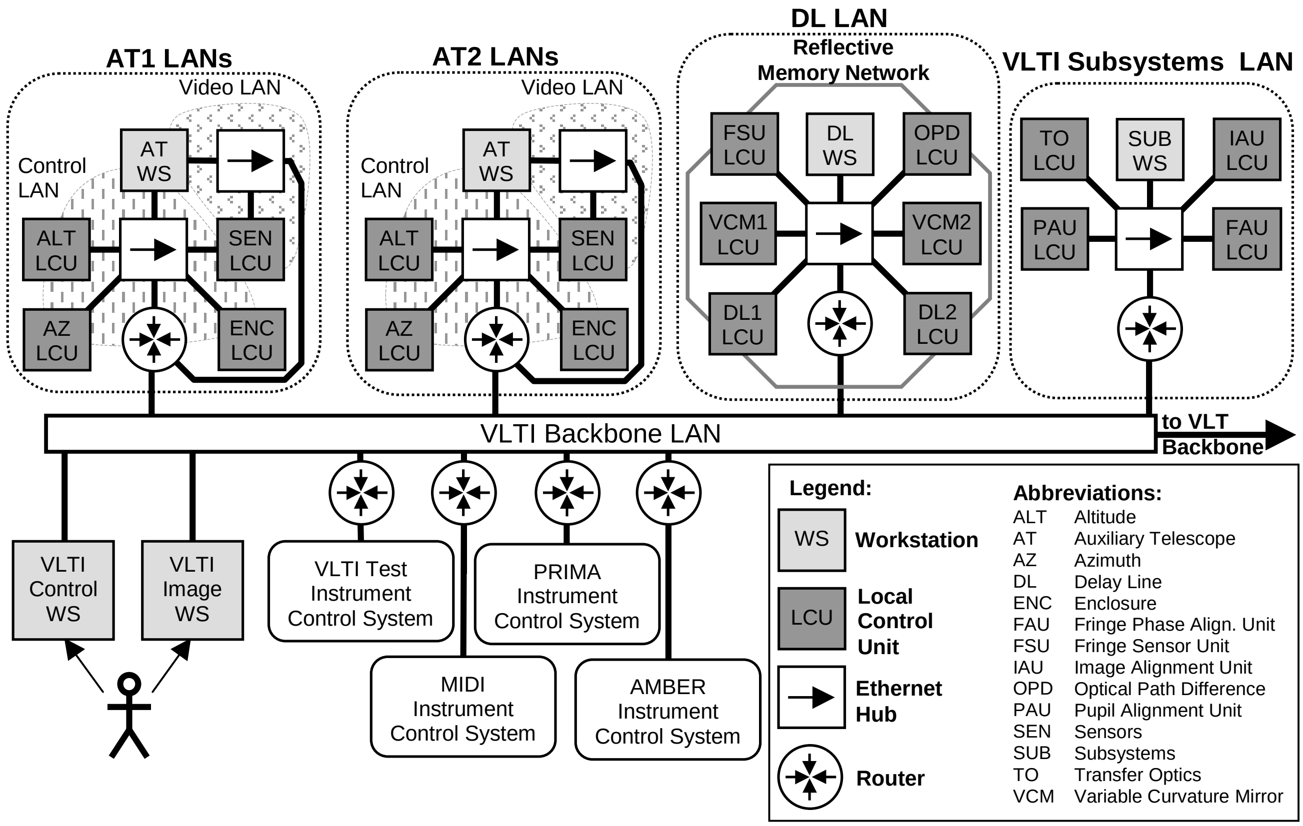

Figure 6. VLTICS hardware architecture.

5. FRINGE CONTROL LOOP

During the observation run the Delay Line (DL) can track according to two possible control modes: blind tracking and fringe tracking. In the former case the DL follows a theoretical trajectory (blind tracking trajectory), which is computed in advance by the DL WS, sent through the LAN to the DL LCU, and stored locally. In the latter case the DL motion depends also on a real-time measurement signal, which is provided by the Fringe Sensor Unit (FSU), an electro-optical device capable to detect the Optical Path Difference (OPD) between pairs of stellar beams. The fringe tracking mode is further divided into coherence tracking, where the instantaneous OPD is measured and controlled with a precision within the coherence length of the observed light beam, and phase tracking, where the OPD precision is in the order of a fraction of the wavelength. Fig. 7 shows the DL controller state transition diagram. The drawing illustrates the sequence of commands and events necessary to bring the DL from the initial idle state to the fringe tracking mode or to the blind tracking mode, if the fringes cannot be detected during the observation run.

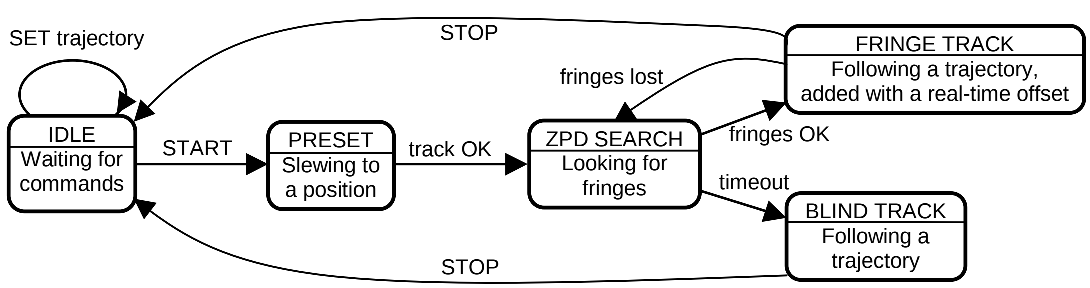

Figure 7. Delay Line Controller state transition diagram.

The blind tracking trajectory is generated according to the specified target coordinates [α,δ], the required observing time, the telescope array configuration and the position of the reference DL. A software module, the OPD Model, takes into account geometrical misalignments and systematic errors in order to improve the DL tracking accuracy. The OPD measurements (one for each tracking DL with respect the reference DL) performed by the FSU are collected by another LCU, the OPD Controller. This one filters the signals, applies optimal control algorithms and distributes the real-time position offsets to each tracking DL over a high-speed, zero-latency dedicated communication link, based on the Reflective Memory technology. The offset value is used by the DL Controller to drive the fine positioning stage (piezo element), whose displacement will be off-loaded by the coarse positioning stage (carriage). Fig. 8 shows the control loop scheme.

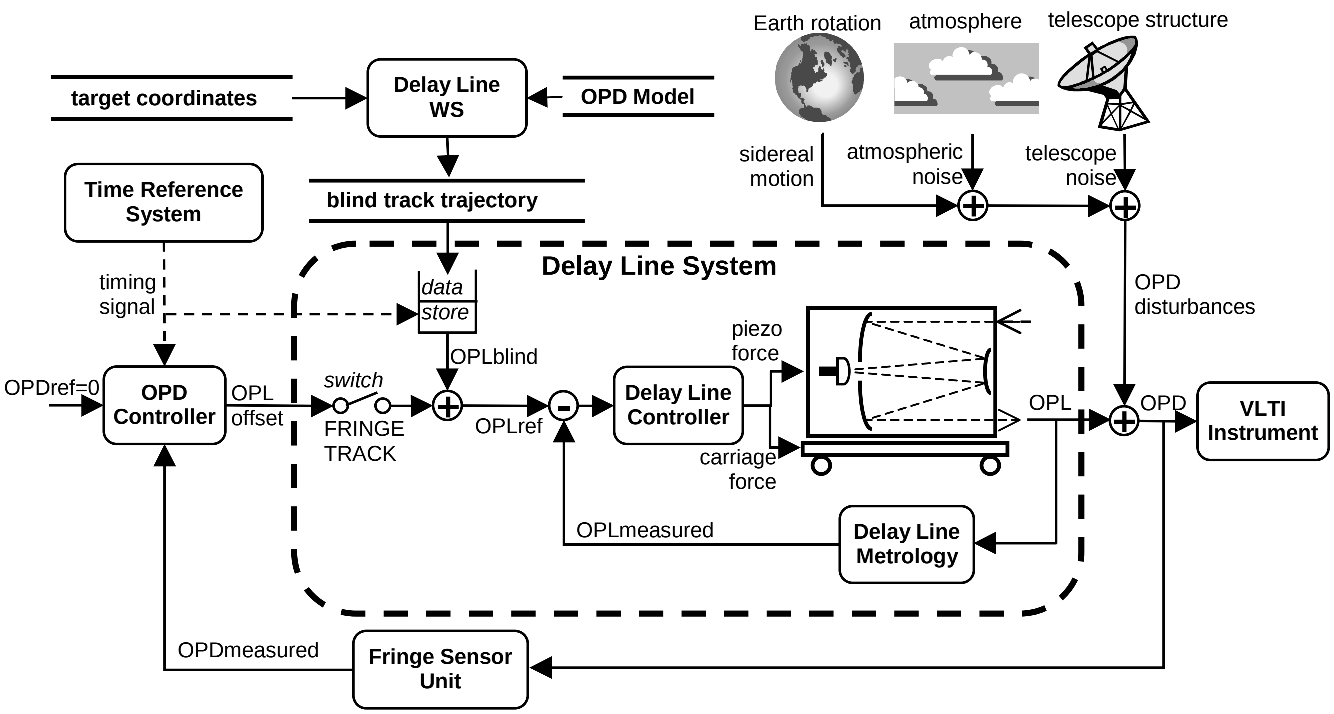

Figure 8. Fringe control loop scheme.

6. PARALLEL BETWEEN VLTICS AND TCS

We have identified a certain degree of similarity between the main tasks that have to be performed by the TCS and the ones realized by the VLTICS during observation. Table 2 relates analogous functions.

TCS	VLTICS
pointing model: software model for analyzing and enhancing the telescope pointing accuracy	**OPD model**: software model for analyzing and predicting the DL position for accurate optical path length equalization
telescope preset: move the telescope to point a new object according to its astronomical coordinates	**delay line array preset**: position the DLs where the theoretical zero OPD condition among the interferometric arms is achieved
field acquisition: acquire and center the object image in the field of view	**ZPD searching**: search for detection of interference fringes between pairs of light beams and lock them
tracking: move the telescope's axes to follow the apparent motion of the object in the sky	**blind tracking**: move the delay lines to compensate the variable OPD due to the apparent motion of the object in the sky
autoguiding: correct at low frequency the tracking errors by driving the altitude and azimuth axes	**coherence tracking**: correct the OPL introduced by the DL with an accuracy within the coherence length of the light beam
field stabilization: correct at higher frequency the tracking errors by modifying the tip/tilt angle of a telescope's mirror	**phase tracking**: correct the OPL introduced by the delay line with an accuracy within the wavelength of the light beam

Table 2. Parallel between TCS and VLTICS functions.

7. VLTI OPERATION

An interferometric observation on a specified object, the so-called target object, usually requires also the identification of a set of other objects, the calibration objects, which are used for calibration purposes. During the observation run the system switches alternately between the target object and one of the associated calibration objects. Each single observation, either on the target or on a calibration object, requires to preset the telescopes and the delay lines, to acquire the object and to take one or more exposures. The sequence of the single observations and the telescope array configuration has to be defined by the observer. The observer might also want to schedule several times the same sequence within an observing night to get a satisfying uv coverage. An interactive software tool, the VLTI Science Model, will help the observer during the detailed observing program definition. It acts mainly as a decision support tool through simulation of the VLTI behavior in various operational configurations and in relation to the selected target object characteristics.

Each single observation has its own time window reserved in the scheduling, corresponding to the required integration time. A single observation is considered over when the requested integration time has been completed. An integration time may correspond to one or several exposures. When the VLTI is in phase tracking, the data acquisition may be performed by means of a single exposure, which lasts as long as the integration time (integration time = one exposure time). When the VLTI is working in coherence or blind tracking, the integration time may be the incremental sum of the times of many short exposures (snapshots). The observation steps can be executed under the direct control of the observer, who interacts with the VLTI Instrument Control System, or in service mode by using the facilities provided by the VLT Scheduling Tools, which are also responsible for the advanced booking of all the required resources. The data generated during the observation are stored in the VLTI Science Archive. They will be retrieved and processed in a later stage by using the Data Reduction Software.

Fig. 9 shows graphically the operational scenario by means of the interaction diagram among the various systems. The connecting lines are tagged with sequence numbers, which define the chronological order of the actions.

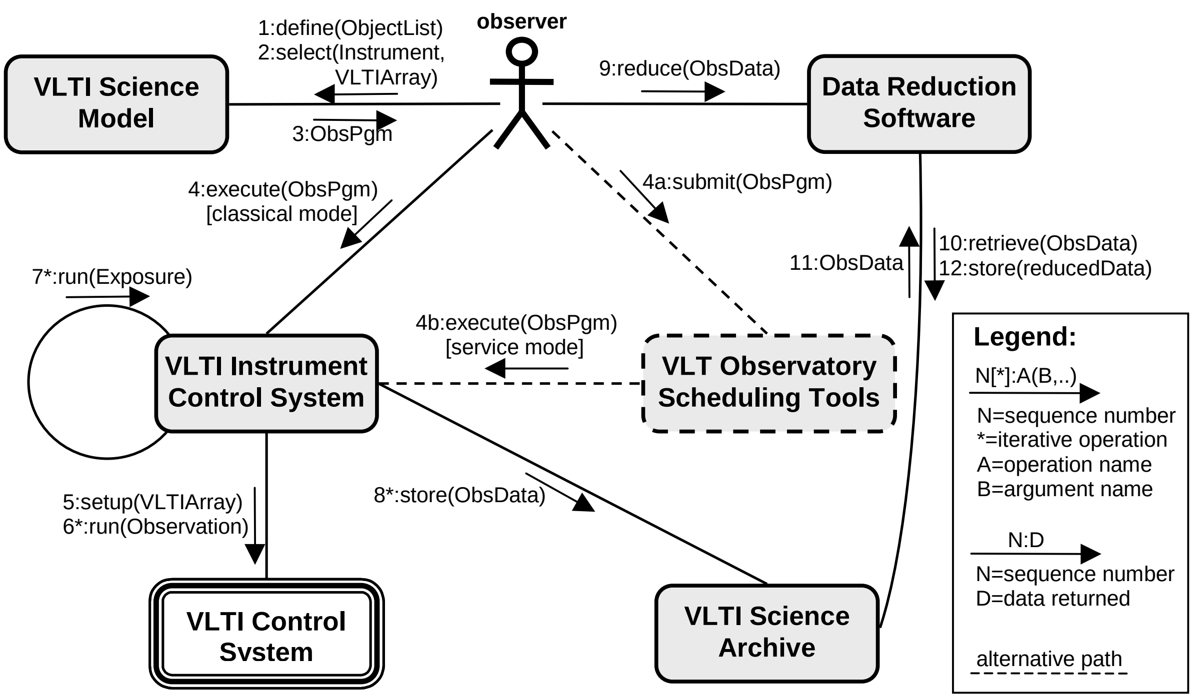

Figure 9. VLTI operation: interaction diagram.

8. CONCLUSIONS

The general design of the VLTICS architecture has been completed and the specific software requirements have been defined. The external interfaces have been clearly identified and the role of the VLTICS within the overall VLTI operation has been analyzed.

The need of a fast data communication link, dedicated to the most critical control loop, the fringe tracking loop, has been carefully investigated and a solution based on a reflective memory network has been selected and successfully tested.

The detailed design phase has started. A prototype of the Fringe Control Loop is currently under development to achieve an early assessment of the actual performance required by the loop and to minimize the impact during the integration phase.

9. ACRONYMS

ATM	Asynchronous Transfer Mode	RTAP	Real Time Application Platform
CCD	Charge Coupled Device	TCP	Transmission Control Protocol
DB	Database	TCS	Telescope Control System
DL	Delay Line	TRS	Time Reference System
ESO	European Southern Observatory	UTC	Universal Time Coordinated
FSU	Fringe Sensor Unit	VCM	Variable Curvature Mirror
I/O	Input/Output	VME	Versa Module Eurocard
LAN	Local Area Network	VLT	Very Large Telescope
LCC	LCU Common Software	VLTI	Very Large Telescope Interferometer
LCU	Local Control Unit	VLTICS	VLTI Control System
OPD	Optical Path Difference	WS	Workstation
OPL	Optical Path Length	ZPD	Zero Path Difference

10. ACKNOWLEDGEMENTS

I wish to thank all colleagues in the VLTI Group and in particular B. Koehler, who initiated me to the principles of astronomical interferometry and explained carefully and patiently the overall design of the VLTI and the purpose of the many subsystems.

I like to mention the helpful cooperation with the VLT Software Group, which represents the main reference for the VLT control standards and the software design at ESO.

11. REFERENCES

1. O. von der Lühe, F. Derie, B. Koehler, S. Lévêque, F. Paresce, M. Verola, "Interferometry with the ESO Very Large Telescope", Proc. of SPIE, vol. 2871, pp. 498-503, 1996.
2. O. von der Lühe, D. Bonaccini, F. Derie, B. Koehler, S. Lévêque, E. Manil, A. Michel, M. Verola, "A New Plan for the VLTI", The ESO Messenger, no. 87, pp. 8-14, 1997.
3. K. Wirenstrand, "VLT Telescope Control Software - an overview", Proceedings of SPIE, vol. 2479, pp. 129-139, 1995.
4. G. Raffi, K. Wirenstrand, "The VLT control software in its final test phase", Proc. of SPIE, vol. 2871, pp. 996-1004, 1996.
5. B. Gilli, "Workstation Environment for VLT", Proc. of SPIE, vol. 2199, pp. 1026-1033, 1994.
6. B. Gustafsson, "VLT Local Control Unit Real Time Environment", Proc. of SPIE, vol. 2199, pp. 1014-1025, 1994.
7. G. Filippi, "Software Engineering for ESO's VLT Project", Nuclear Instruments & Methods in Physics Research, A 352, pp. 386-389, 1994.
8. I. Sommerville, Software Engineering, Fifth Edition, Addison Wesley Publishing Company, 1995.
9. G. Booch, Object-Oriented Analysis and Design with Applications, Second Edition, The Benjamin/Cummings Publishing Company, 1994.