A Distributed Object-Oriented Telescope Control System Based on RT-CORBA and ATM

José M. Filgueira
Control Group, GTC Project
Instituto de Astrofísica de Canarias
Email: jmfilgue@iac.es

Abstract

The GTC project is in charge of the construction of an optical-infrared 10-meter class telescope at the ORM observatory in Canary Islands. First light will be by the end 2002. The operating life of the telescope will be over 50 years. The control system of the GTC will be responsible for the management and operation of the telescope, including its instrumentation. This paper presents a first approach to the design of the control system.

1 Introduction

The continuous and rapid development of the technologies related to hardware, software and communications has permitted a greater complexity in control systems. In their turn, the new techniques of active and adaptive optics, the new methods of optimisation of useful observing time and the programmes of continuous evolution to ensure competitive telescopes capable of assimilating future technological advances, present new challenges in the design of control systems.

The life cycle of the control system of the GTC will be subject to a continuous flux of changes brought about by different factors (evolution of the requirements, continuous development of new instruments, correction of faults, etc.). These factors must be taken on board with the minimum impact possible on the availability of the GTC once it enters operation. This will only be possible through the selection and planning of an adequate technological framework that enables these changes to be assimilated throughout the entire life cycle of the telescope. In order to accomplish this, the adoption of open 'de facto' standards like ATM, CORBA and POSIX becomes a key point.

In the past, the technologies, needed in order to achieve the required network bandwidth pushed the adoption of heterogeneous network architectures being based in different technologies: FDDI, Ethernet, etc. This resulted in complex and non-scalable solutions with fixed QoS (Quality of Service). Recently ATM allows the unification of all network communication requirements into one network technology. This results in more flexible, scalable, and easy-to-maintain networks. Additionally, ATM provides other important advantages such as high bandwidth, end-to-end QoS and virtual connections. This makes ATM an appealing solution for real-time LAN based control systems.

CORBA [1] is a distributed object computing middleware standard. It supports the development of flexible and reusable distributed services and applications providing independence of hardware platform, network technology, operating systems and programming languages. Current researches on RT CORBA (e.g. real-time event services and end-to-end QoS guarantees) can provide an effective architecture capable of supporting current control systems while retaining their real-time behaviour. Recently, several vendors have also recognized this issue and have ported their ORB to real-time POSIX compliant OS platforms.

2 Hardware architecture

The hardware architecture of the control system will consist of VME nodes with real-time processing capacity connected directly to the physical devices of the GTC. These connections will be able to use a varied set of control buses (e.g. CAN bus, GPIB, Bitbus). Other higher-level nodes will carry out co-ordination functions and will offer critical services to the remaining nodes (e.g. event dispatching, logging, monitoring, scheduling). Both the VME nodes and the co-ordination units will be connected by means of one or more high-performance ATM switches to form the so-called control network. This architecture will allow the dynamical configuration of traffic so that each node has an adequate bandwidth for its needs.

ATM technology provides a series of important features: wide bandwidth, end-to-end individual QoS, virtual connections and flexible topology. These features will allow the building of a system based on an open standard with integrated communication services (data, images, audio and video). This can lead to a reduction in cost, greater flexibility and applications which are more dynamical. Although Ethernet technology continues to evolve rapidly (Gigabit Ethernet), it is still inadequate for real-time traffic due to its unpredictable delays and the lack of support to guarantee QoS. Moreover, ATM constitutes a scalable technology (possibility of widening bandwidths without modifying the topology of the network) from 1.5 Mbps (DS-1) to 10 Gbps (OC-192c). At present, there are products available from various manufacturers working with speeds starting from 25Mbps, 155 Mbps (OC-3c) and 622 Mbps (OC-12c). This will allow the adoption of a homogeneous and scalable solution for all the data-transmission needs of the GTC. The use of physical fibre-optic interfaces will be equally important, since these will provide protection against many of the possible sources of electromagnetic interference, as well as a large bandwidth.

3 Software architecture

The adopted hierarchical structure will consists of five layers; the service layers (base operating and middleware services) provide the application domain independent infrastructure which is common to all the components of the control system. These layers isolate the application from modifications of the hardware platforms, operating systems and commercial off-the-shelf products (e.g. database management systems). To this infrastructure, the framework layer adds a set of specific domain frameworks, control, planning and scheduling and data processing forming the basis of a domain-specific software architecture. These frameworks will provide adequate 'hot-spots' for adaptation. The component layer adds highest-level reusable binary components grouped into toolkits (e.g. Image viewer). The application layer contains the final applications (mainly built by the composition of the underlying components) and the majority of the user interfaces and interfaces with external systems.

3.1 Distributed architecture

The architecture of the control system will consist of a set of highly integrated systems distributed by means of networks in a hierarchical organization. This hierarchy will be organized by following the client-server model. There will be a number of control points and, therefore, of processors necessary for managing them.

As in other application domains (avionics, tele-communications, multimedia), the control system must guarantee real-time capability with regard to communication networks, as well as operating systems and underlying middleware components, with the aim of satisfying their QoS requirements. Applications in these domains must be flexible and reusable to provide point-to-point QoS guarantees. These flexibility and reusability requirements drive the use of object-oriented middleware like CORBA, MidART [2] ACE (Associative Computing Environment) [3], ILU or DCOM (Distributed Common Object Model). Although some operating systems, networks and protocols at present support real-time scheduling, they do not provide an integrated solution. At present, there are various projects working on the adaptation of this object-oriented middleware to hard real-time systems (e.g. avionics) or systems with restricted latency (e.g. teleconferences).

The processing elements of the control system will use a real-time implementation [4] of the CORBA standard for communications between objects via the network. The adoption of this client-server model, together with the CORBA/IDL (Interface Definition Language) interfaces and C++ and Java programming, will be in accordance with recent developments in programming languages and distributed architecture design.

The basic mechanism consists of providing connections and interoperability among different objects which reside in different processors. This will be achieved by means of the implementation of different distributed services:

- *Object Services* are interfaces for general services that are likely to be used in any program based on distributed objects.
- *Common Facilities* are interfaces for horizontal end-user-oriented facilities applicable to most application domains.
- *Domain Interfaces* are application domain-specific interfaces.
- *Application Interfaces* are non-standardized application-specific interfaces.

The above object frameworks (see previous section) can be implemented as collections of co-operating objects categorized into Application, Domain, Facility, and Service Objects. Each object in the framework will support, or will make use of, some combination of Application, Domain, Common Facility, and Object Service interfaces (e.g. via interface inheritance or client requests).

The possibility of providing server interfaces to other systems will be considered, e.g. access to the Experimental Physics and Industrial Control System (EPICS) [5] via the channel access mechanisms or CDEV [6]. In this case, EPICS devices can be represented as CORBA-compliant device services.

Access from the user interfaces to the distributed services could be effected by means of stand-alone applications or through the Internet Inter-ORB Protocol (IIOP) already incorporated into some browsers. This will permit the execution of user interfaces from any remote platform.

4 Software engineering issues

- *Analysis, design and object-oriented programming:* These techniques replace traditional data-directed methods and functional decomposition (analysis and structured design) by an integrated approach to the analysis, design and implementation based on an object model.
- *Fast prototyping and iterative development:* In a fast-prototyping and iterative-development process [8.24] an initial version of the system is rapidly constructed with an emphasis on the areas of greatest risk.
- *Architecture-directed development:* In an architecture-directed process, the objective is to achieve an architecture which is resilient to changes in requirements, within reasonable limits.
- *Large-scale reuse:* Object-oriented design and architecture-directed development implicitly support reuse; this is more effective when reasonably large components are reused, such as subsystems and class categories. Therefore, the analysis, design, integration and testing of these components will also be reused.
- *Improvement and control of the software process:* Experience with real-world large projects shows that a highly integrated environment is necessary in order

to facilitate and reinforce the control of the management of the process.

- *Software first focus:* The use of standards for open systems will allow the postponement, until the optimum moment in the project cycle, of the choice of technologies (hardware platforms, operating systems, network protocols and topology). This is crucial if it is required to achieve an effective compromise between functions, capabilities, cost and planning in an area where technology can change dramatically over the lifetime of the project.
- *Object-oriented frameworks:* An object-oriented framework [7] is the skeleton of the architecture for a problem in a given domain and provides opportunities for the reuse of designs and code on a large scale, reducing the time and cost needed for building an application.
- *Design patterns:* A design pattern [8] describes a set of co-operating objects united by certain relationships, which are repeatedly encountered in the solution of similar problems. One of the main interests in design partners is the representation of knowledge of design decisions in a domain; therefore, these can be reused.

5 Conclusions

Although the cost of ATM is at present higher than that of other solutions (although not by much in comparison to switched Ethernet), the benefits in terms of simplicity and architecture scalability permit a simpler system and therefore cheaper software. Scalability allows solutions to be found for all needs, avoiding the use of different protocols for each need and the use of gateways that can give rise to bottlenecks in the system.

Moreover, the specification of interfaces will be very important for sustaining and preserving investment in the face of rapid technological change. For this purpose open standards, such as RT POSIX or ATM, as well as CORBA itself, will be used.

References

[1] Object Management Group 1995. ‘*CORBA Specification v.2’*

[2] Mizunuma, I., Shen, C., Takegaki, M. 1996. ‘Middleware for distributed industrial real-time systems on ATM networks’, *17th IEEE Real-Time Systems Symposium*.

[3] Schmidt, D. 1994. ‘The Service Configurator Framework: An extensible architecture for dynamically configuring concurrent, multi-service daemons’, In *Proc. Second International Workshop on Configurable Distributed Systems, IEEE Computer Society,* 190.

[4] Harrison, T. H., Levine, D., Schmidt, D. 1997. ‘The design and performance of a real-time CORBA Object Event Service’. In *Proc. OOSPLA’97 Conference, Atlanta October 1997.*

[5] Dalesio, L. et al. 1993. ‘The Experimental Physics and Industrial Control System Architecture: past, present, and future’, *Proc. ICALEPCS’93.*

[6] Chen, J., Heyes, G., Akers, W., Wu, D., Watson, W. 1995. ‘CDEV: an object-oriented class library for developing device control applications’, *Proc. ICALEPCS’95.*

[7] ‘Leveraging Object-oriented frameworks-a technology primer from Taligent’, *Taligent White Paper*. Available form http://www.taligent.com.

[8] Gamma, E., Helm, R., Johnson, R., Vlissides, J. 1995. *Design Patterns. Element of Reusable Object-oriented Software*. Addison-Wesley.