The DELPHI Experiment Control System

C. Gaspar
CERN, European Organization for Nuclear Research
CH-1211 Geneva 23, Swizerland

J. J. Schwarz
INSA L3I. B502 IF
20 av A. Einstein
69621 Villeurbanne Cedex, France

Abstract

DELPHI (DEtector with Lepton, Photon and Hadron Identification) is one of the four experiments built for the LEP (Large Electron-Positron) collider at CERN, the European Organization for Particle Physics.

The DELPHI detector is composed of 20 subdetectors, which were built by different teams of laboratories of the DELPHI collaboration (around 500 scientists from 50 laboratories all over the world).

The Control system of the experiment has to assure the correct and continuous operation of the experiment until around the year 2000 and it has to allow for an easy reconfiguration in order to cope with the replacement and upgrades of the different parts of the experiment during its live time.

The Control of a physics experiment involves multiple domains with different requirements (some parts have to work in real-time, some are safety critical, others have to be user-friendly). Usually these domains are engineered separately. In order to increase the operating efficiency and the reliability and maintainability of the system DELPHI took a global approach in the design of the complete experiment control. This approach allowed for the interconnection of the various domains and led to a high degree of automation and an homogeneous interface to the full control system.

This paper describes the strategy adopted in order to handle the different and sometimes contradictory requirements of the different parts of the system keeping thus an overall system prespective.

1 Introduction

DELPHI[1] consists of a central cylindrical section and two end-caps. The overall length and the diameter are over 10 meters and the total weight is 2500 tons.

The electron-positron collisions take place inside the vacuum pipe in the centre of DELPHI and the products of the annihilations fly radially outwards. The products of the annihilations are "tracked" by several layers of detectors and read out via some 200,000 electronic channels. A typical event requires about 1 million bits of information. Figure 1 shows a cut view of the DELPHI detector.

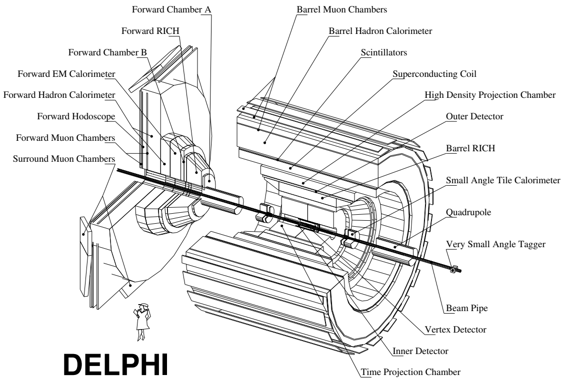

Figure 1. The DELPHI Detector.

The main aim of the experiment is the verification of the theory known as the "Standard Model".

The DELPHI experiment took 10 years of preparation and started collecting data in 1989. It has to be up and running 8 months/year (24h a day) until around the year 2000. During it's life-time it undergoes continuous changes. In 20 years people, detectors, electronics and computers are replaced or upgraded for various different reasons. The control system of the experiment has to allow for upgrades and reconfiguration.

In order to assure a high degree of efficiency and reliability of the full experiment the control system should be automated to the highest possible degree. When complete automation is not possible any control and monitoring tools should be easy-to-use and self-explanatory in order to be effectively operated by the shift crew of 3 operators (normaly phisicists) permanently present.

2 Online System Engeneering

The Online System of the experiment is divided into five main areas :

- The Slow Controls System (SC)[2]

  Controls and monitors slowly moving technical parameters and settings, like temperatures and high voltages, and writes them onto a database.

- The Data Acquisition System (DAS)[3]

  Reads event data from the 20 sub-detectors composing DELPHI and writes it onto tape.

  In order to provide a high degree of independence to the individual sub-detectors the DAS system has been split into 20 autonomous partitions. These partitions are normally combined to form a full detector but they can also work in stand-alone mode for test and calibration purposes.

- The Trigger System[4]

  Provides the DAS system the information on whether or not the event is interesting and should be written to tape.

- The LEP Communication System[5]

  Controls the exchange of data between the LEP control system and DELPHI.

- The Quality Checking System (QC)[6]

  Provides automatic and human interfaced tools for checking the quality of the data being written on tape

The design and engineering of these areas involved various different constraints and requirements ranging from real time behaviour in the DAS system to very strict safety aspects in the SC system. These different requirements should not prevent the achievement of an uniform and homogeneous control system.

An homogeneous system is easier to design, to develop and to maintain and also to operate and automate.

In previous experiments the control of the different areas was normally designed separately by different experts, using different methodologies and tools resulting on a set of dedicated control systems.

DELPHI decided to take a common approach to the full "experiment control" system.

The Online control system is characterized by a highly distributed architecture, it is composed of around 40 workstations interconnected by a Local Area Network (LAN). Each workstation controlling a different part of the experiment (either a sub-detector or a central part : DAS, SC, etc.)

The full system was engeneered according to the following principles

- Maintainability

  The life time of a large physics experiment (from its prototype phase) is of the order of 20 years. The original design of the system should take this fact into account and allow for easy replacements and reconfiguration of the experiment. In order to allow for un easy upgrade and maintenance of the system, the software should be constructed in modules with well defined interfaces.

- Scalability

  The system should be able to grow, the introduction of new detectors should be foreseen. The same software modules should be used by all detectors, and they should provide hooks for detector specific software, in order to allow for the insertion of new detectors.

- Portability

  The software used in the experiment to control a detector should be portable so that it can be used in the testing phase of new detectors sometimes in completely different environments, like a different laboratory in the other side of the world.

In order to cope with the complexity of the system a new concept for the coding of the control logic has been developped. In this concept - SMI [10](State Management Interface) the experiment is described in terms of objects wich may appear in well defined states and on which a set of actions can be performed. The SMI objects are normaly grouped in order to form domains.

The various areas of DELPHI have been mapped into SMI domains: sub-detector domain, DAS domain, SC domain, TRIGGER domain, etc.

SMI objects may correspond directly to an entity of the experiment (i.e. a computer-controlled device) or an abstraction describing a part of the experiment. The interaction between objects can best be understood in the example of Figure 2.

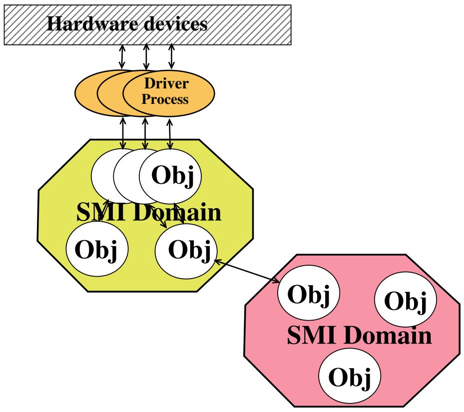

Figure 2. SMI example

The interaction between objects is specified using a formal language, the State Manager Language - SML. The main characteristics of this language are :

- Finite State Logic

  Objects are described as finite state machines. The only attribute of an object is its state. Commands sent to an object trigger actions that can bring about a change in its state.

- Sequencing

  An action on an abstract object is specified by a sequence of instructions, mainly consisting on commands sent to other objects and logical tests on states of other objects. Actions on concrete objects are sent off as messages to the Driver Control Processes.

- Asynchrounous

  Several actions may proceed in parallel: a command sent by object-A to object-B does not suspend the instruction sequence of object-A. Only a test by object-A on the state of object-B suspends the instruction sequence of object-A if object-B is still in transition.

- AI-like rules

  Each object can specify logical conditions based on states of other objects. These when satisfied will trigger an action on the local object. This provides the mechanism for an object to respond to unsolicited state change of its environment.

```
!- Example of SML code

object : DETECTOR_CONTROL
  state : READY
    action : START_RUN
      do MOUNT TAPE
      if TAPE not in_state MOUNTED
        do MOUNT_ERROR ERROR_OBJ
        terminate_action/state=ERROR
      endif
      do START READOUT_CONTROLER
      if READOUT_CONTROLER in_state RUNNING
        terminate_action/state=RUN_IN_PROGRESS
      ...
  state : RUN_IN_PROGRESS
    when TAPE in_state FILE_FULL
      do PAUSE_RUN
    when READOUT_CONTROLER in_state ERROR
      do ABORT_RUN
    action : ABORT_RUN
      ...

object : READOUT_CONTROLER/driver
  state : READY
    action : START
    ...
  state : RUNNING
    action : PAUSE
    action : ABORT
    ...
```

The SMI mechanism allows an easy reconfiguration of the system: changes in the hardware can be easily integrated by modifying or replacing driver processes and logical modifications by changing the SMI code. The decoupling between the actual actions on the hardware (done by the Driver Processes) and the control logic (residing in the SMI objects) made the evolution of the system from its first test phase up to today's complexity a very smooth process.

By using coherently the same mechanism throughout the Online System the design and implementation of the system could benefit from well defined interfaces to all its components. The full online system is controlled through this mechanism, using about 1000 SMI objects in 50 different domains.

2.1 Automation

A high level of automation of the system is very important in order to avoid human mistakes and to speed up standard procedures.

Using the SMI mechanism the creation of a top level domain - BIG BROTHER - containing the logic allowing the interconection of the underlying domains

(LEP, DAS, SC, etc.) was a very easy task.

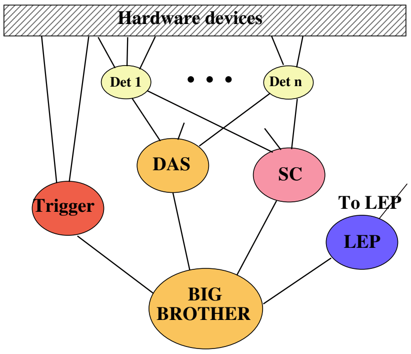

Figure 3. Big Brother Control

Under normal running conditions BIG BROTHER pilots the system with minimal operator intervention as shown in Figure 3. In other test and setup periods the operator becomes the top-level object, using the user-interface he can send commands to any SMI domain.

3 Control and Monitoring Requirements

For the monitoring and control of the different domains the operators dispose of a set of workstations displaying the necessary information and allowing the control of the different parts of the experiment, as shown if Figure 4.

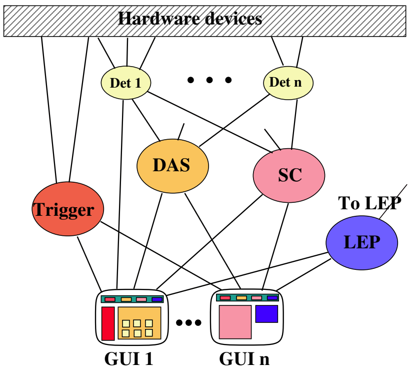

Figure 4. User Control and Monitoring.

In order to accomplish the task of controlling efficiently the experiment a standard graphical user interface was necessary. The user interface is meant to control and monitor all parts of the DELPHI Online System and it was designed according to the following requirements :

- The interface should be able to access any information necessary for monitoring purposes and to send commands to all parts of the system, even though the online system is completely distributed.
- The information displayed should be consistent over all interfaces running at a certain moment.
- The displays as well as the online system components should be able to move freely from one machine to another.
- The interface should be decoupled from the online system so that changes introduced to one do not imply changing the other.
- The interface should be accessible by all members of the collaboration either in the DELPHI control room, in their office at CERN or at their home laboratorys anywhere in the world.
- The interface should provide a standard look and feel and it should be possible to hide or display information as needed.
- The interface should be configurable, so that the same system can be used for all the control posts in DELPHI, by sub-detectors for their test and calibration activities as well as by collaboration members for monitoring and general information.
- The interface should provide context sensitive online help in order to be easy to use and self-explanatory.

The first five requirments were achived by intruducing a communication package- DIM- Distributed Information Management system[7]. The later three were satisfied by DUI[9], the Delphi User Interface system.

4 Communication System

DIM was designed In order to make all the information produced in the Online System available wherever and whenever it is needed for monitoring, display, processing or control purposes. DIM's aim is to be the communication layer used by all processes in the Online System, including SMI.

The DIM system was designed and implemented according to the following additional characteristics :

- Efficiency

  The communication mechanism of DIM was chosen having in mind the asynchrounous character of SMI objects and the speed in reacting to changes or error conditions in the system. The solution we thought the best is for clients to declare interest in a service provided by a server only once (at startup), and get updates at regular time intervals or when the conditions change. I.e. an asynchronous communication mechanism allowing for task parallelism and multiple destination updates.

Distributed applications are often based on Remote Procedure Calls (RPC) [8]. DIM's mechanism - interrupt like - as opposed to RPC's polling approach involves twice less messages sent over the network, i.e. is faster and saves in network bandwidth. It has also the advantages of allowing parallelism (since the client does not have to wait for the server reply and so can be busy with other tasks) and of allowing multiple clients to receive updates in parallel.

- Transparency

  At run time no matter where a process runs, it is able to communicate with any other process in the system independently of where the processes are located. Processes can move freely from one machine to another and all communications are automatically reestablished. (this feature also allows for machine load balancing).

  At coding time the user is not concerned with machine boundaries, the communication system provides a location-transparent interface.

- Reliability and Rubustness

  In an environment with many processes, processors and networks, it often happens that a process, a processor or a network link breakes down. The loss of one of these items should not perturbate the rest of the application. DIM provides an automatic recovery from crash situations or the migration of processes.

DIM uses a publish/subscribe mechanism. Any process in the Online System can publish (Server) information and any interface (or any other process) can subscribe (Client) to this information. A unit of information is called a "Service". A Name Server keeps track of all the Servers and Services available in the system.

Servers "publish" their Services by registering to the Name Server (Normally once at startup).

Clients "subscribe" to Services by asking the Name Server which Server provides the Service and then contacting directly the Server. Client's Services will be kept up-to-date in an event driven mode or at regular time intervals. Clients can also send commands to servers.

Whenever one of the processes (a server or even the name server) in the system crashes or dies all processes connected to it will be notified and will reconnect as soon as it comes back to life. This feature not only allows an easy recovery from any crash situations but also an easy migration of servers from one machine to another. Figure 5 shows an example of the usage of the DIM system.

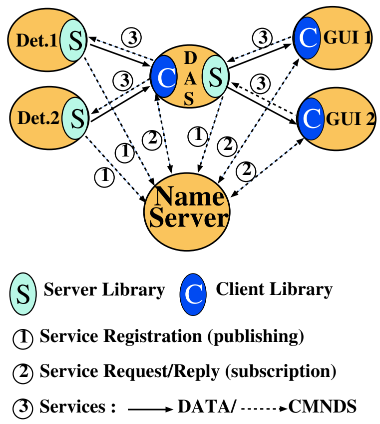

Figure 5. DIM example.

The DIM system is used by SMI in order to communicate ammong objects. Due to the transparency of the DIM mechanism the machine where each object is running is completely indifferent, and the machine load can be easily balanced by moving object code from one machine to another.

DIM is responsible for most of the communications inside the DELPHI Online System, it is used by SMI in order to transfer object states and commands, by the user interfaces in order to access SMI or any other necessary information and by many other processes for monitoring or processing activities. In the DELPHI environment it makes currently available around 15000 Services provided by 300 Servers.

5 User Interface

The DUI System provides the operator with a MOTIF interface to the full Online System. DUI provides a set of pre-defined displayable blocks (widget trees), each of which displaying and/or controling a part of the experiment. These blocks can be combined to form a user-interface that suits the needs of a particular user/operator.

Figure 6 shows an example of a possible DUI configuration: the Data Acquisition control used by the operator on shift. Detailed information on the other domains of the experiment can be obtained by clicking on the buttons of the upper row.

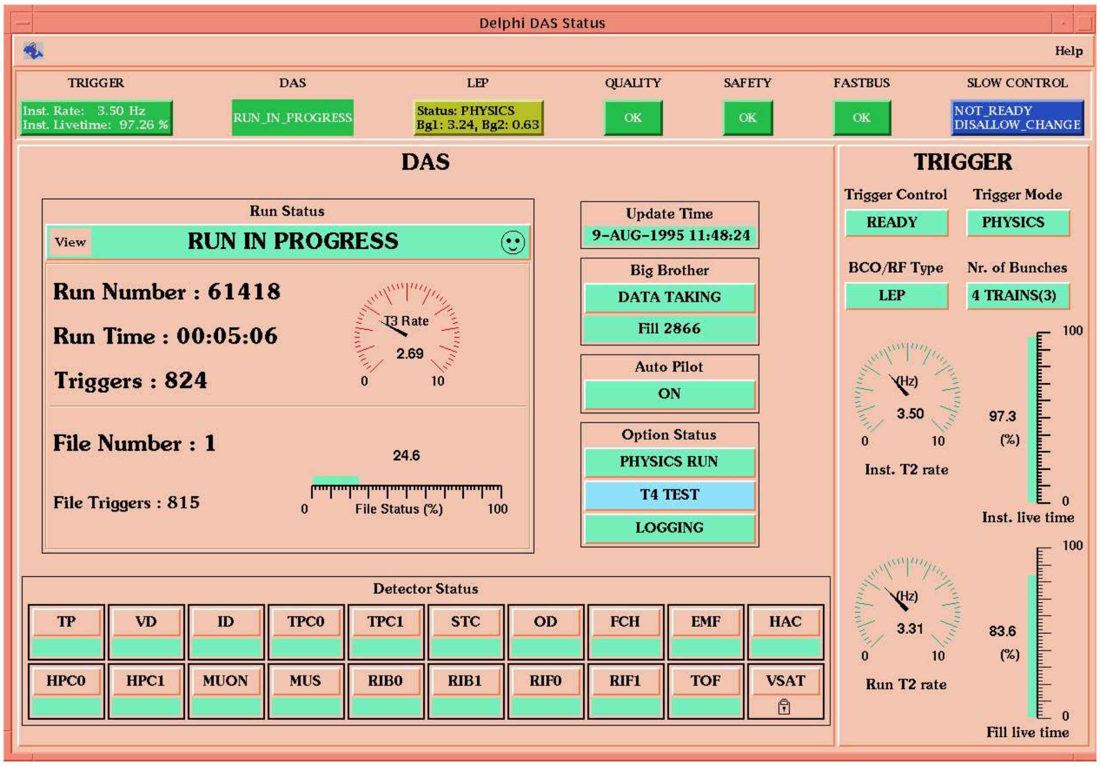

Figure 6. DUI configured for DAS control.

DUI uses DIM in order to get all the necessary information and to send commands to all parts of the system.

The DIM mechanism is perfectly suited for the MOTIF environment, DIM allows clients to specify a routine to be executed when a service is updated, this routine can be used to update the MOTIF display thus making sure all interfaces are coherently kept up-to-date.

6 Conclusions

The Control and monitoring System of a large physics experiment involves constraints of operating efficiency, automation and reliability.

In order to keep the risc of human mistakes to a minimum and to improve efficiency in reacting to changing conditions the system should be as much as possible automated. For the same reasons but when complete automation is impossible an easy-to-use and self-explanatory user interface is of great use specialy when the system has to be operated by non experts.

The possibility of describing the experiment in terms of objects, using SMI, makes it possible to automate DELPHI operations to a maximum.

The availability of a configurable uniform user-interface allowing access to all parts of the system simplifies greatly the task of controlling and monitoring the system whenever complete automation is not possible.

These two charateristics of the DELPHI control system have been reached by the introduction of a communication layer, providing efficient, transparent and reliable inter-process and inter-processor communications.

By fullfilling such characteristics a communication system can greatly improve the performance of the complete system. It provides a decoupling layer between software modules, that makes coding, maintenance and upgrade of the system easier and improves efficiency and reliability at running time.

The use of the State Management Interface combined with a standard user-friendly graphical human interface gives us the possibility to control homogeneously all the different parts of the experiment and to manage the various and unavoidable future upgrades.

References

[1] DELPHI Collaboration, Aarnio, P. et al.,"The DELPHI Detector at LEP" in **Nuclear Instruments and Methods in Physics Research A303** (1991) pp.233–276

[2] Adye, T. et al., "The Slow Controls of the DELPHI Experiment at LEP", in **Proceedings of the International Conference on Computing in High Energy Physics '92** (Annecy, France, September 1992)

[3] Adye, T. et al., "Architecture and Performance of the DELPHI Data Acquisition and Control System", in **Proceedings of the International Conference on Computing in High Energy Physics '91** (Tsukuba, Japan, March 1991)

[4] Fuster, J.A. et al., "Architecture and Performance of the DELPHI Trigger system", in **Proceedings of the IEEE 1992 Nuclear Science Symposium** (Orlando, Florida, October 1992)

[5] Donszelmann, M. and Gaspar, C., "The DELPHI distributed system for exchanging LEP machine related information", in **Proceedings of the International Conference on Accelarator and Large Experimental Physics Control Systems** (Berlin, Germany, October 1993)

[6] Augustinus, A. et al., "The DELPHI Quality Checking system", Poster presentation at the **International Conference on Computing in High Energy Physics '94** (San Francisco, USA, April 1994)

[7] Gaspar, C. and Donszelmann, M., "DIM - A Distributed information management system for the DELPHI experiment at CERN", in **Proceedings of the 8th Conference on Real-Time Computer applications in Nuclear, Particle and Plasma Physics** (Vancouver, Canada, June 1993)

[8] Birrell, A. D. and Nelson, B. J. "Implementing Remote Procedure Call" in **ACM Trans. Comp. Syst.** (1984) Vol. 2, No. 1, pp. 39-59

[9] Dönszelmann, M. and Gaspar, C. "A Configurable MOTIF Interface for the Delphi Experiment at LEP" in **Proceedings of the Second Anual International MOTIF Users Meeting** (Washington DC, December 1992) pp.156–162

[10] Barlow, J. et al., "Run Control in MODEL: The State Manager", in **IEEE trans. nucl. sci. 36** (1989) pp.1549–1553