Chapter 9

Control system

9.1 Introduction

The LHC control system will be an extension of the infrastructure already in use today for the LHC Injector Chain and the Technical Services. Nevertheless, the LHC will present unique challenges for the control system. The hardware is of unprecedented complexity and requires a precise and unambiguous description. Failures could lead to long cryogenic recovery times and long magnetic conditioning cycles. Operational efficiency will depend on both the technical services and the accelerator hardware control systems. The high stored energies and beam powers will require rigorous operational procedures, sophisticated diagnostics and real-time automation. The challenging design parameters and the large dynamic effects will only be mastered by a flexible control strategy, adapted to changing circumstances as the knowledge of the LHC improves.

9.2 Architecture

The LHC control system architecture is largely based on standard components employed worldwide for the control of accelerators and used during recent years for the CERN injectors (PS, SPS) and the LEP machine. However, there are two major changes in the LHC era:

- The consistent use of object-oriented technologies and design principles for the control of beam-related systems and,
- The wide use of industrial controls solutions for the supervision of complete subsystems of the LHC.

9.2.1 Overall architecture

As shown in figure 9.1, the LHC control system has three hierarchical layers of equipment communicating through the CERN Technical Network, a flat Gigabit Ethernet network using the TCP-IP protocol (section 9.2.2). Starting from the bottom of figure 9.1:

- ***At the equipment level***, the various actuators, sensors and measurement devices are interfaced to the control system through three different types of front-end computers:

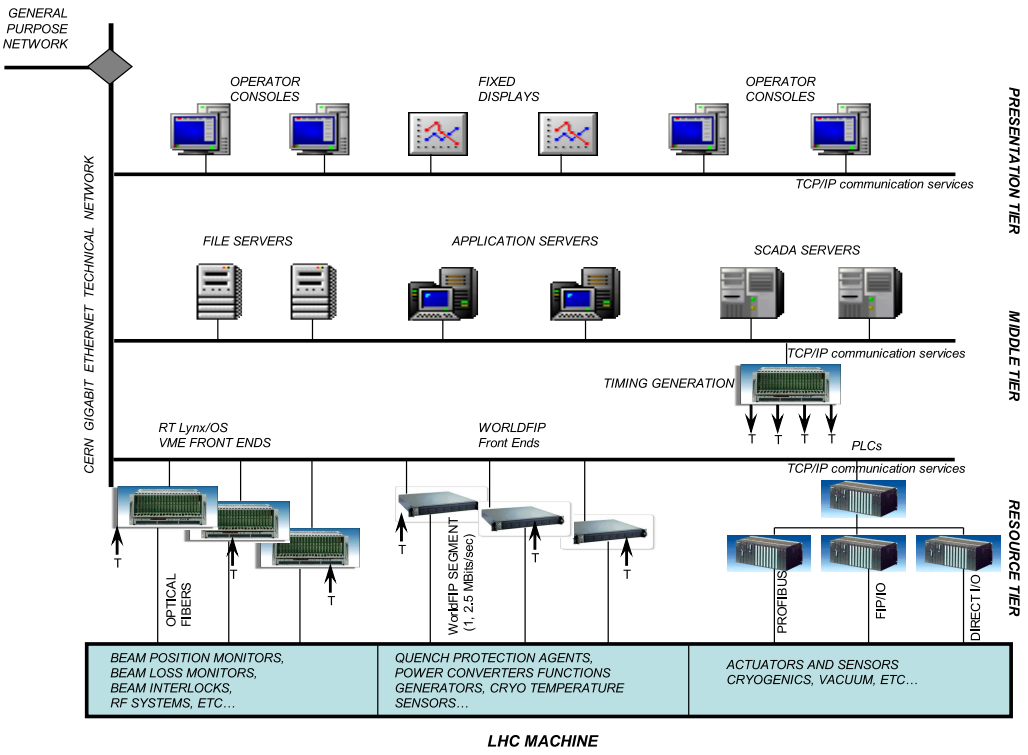

Figure 9.1: Control system architecture.

- VME computers dealing with high performance acquisitions and real-time processing; these employ a large variety of I/O modules. Typically, the LHC beam instrumentation and the LHC beam interlock systems use VME front-ends.
- PC based gateways interfacing systems where a large quantity of identical equipment is controlled through fieldbuses, such as the LHC power converters and the LHC Quench Protection System.
- Programmable Logic Controllers (PLCs) driving various sorts of industrial actuators and sensors for systems such as the LHC Cryogenics systems or the LHC vacuum system.

- ***At the heart of the control system***, powerful UNIX servers host the operational files and run the LHC applications:
  - Application servers hosting the software required to operate the LHC beams and running the Supervisory Control and Data Acquisition (SCADA) systems.
  - Data servers containing the LHC layout and the controls configuration, as well as all the machine settings needed to operate the machine or to diagnose machine behaviour.
  - Central timing, which provides the cycling information for the whole complex of machines involved in the production of the LHC beam, and the timestamp reference.

Processes running in these servers will communicate with the equipment access machines using various mechanisms and protocols such as the Common Object Request Broker Architecture (CORBA) and TCP-Modbus.

- ***At the control room level***, consoles running the Graphical User Interfaces (GUI) will allow machine operators to control and optimise the LHC beams and to supervise the state of key industrial systems. Dedicated fixed displays will also provide real-time summaries of key machine parameters.

9.2.2 Network

The control system for LHC relies on the CERN Technical Network. This network uses IP addresses of the form *172.18.xxx.xxx*. The address prefix prevents any direct communication between the technical network and the Internet. The infrastructure is interconnected with the General Purpose Network so that communications from within the Organization are still possible; however, the technical network can be completely isolated if required.

The technical network is a highly sub-netted, routed network that implements a high performance redundant backbone, reduces parasitic traffic, and keeps the overall structure maintainable. As shown in figure 9.2, the basic distribution is based on a redundant Gigabit Ethernet backbone using fibre optical distribution. The core consists of two redundant backbone routers located in the CERN computer centre and in the Prévessin control room, where most of the fibres terminate. These two central routers are interconnected in a redundant way to a series of regional routers located in the following buildings:

- The computer centre (building 513) for central services like Oracle databases and main file servers.
- The Meyrin control room (building 354) for the control of accelerators located on Meyrin site.
- The technical control room (building 212) for the supervision of the 2 CERN sites.
- The Prévessin control room (building 874) for the SPS control and other technical services located on the Prévessin site.
- Each LHC pit (SR building) for the LHC machine and technical services control.

In each technical building there are one or more "star points" configured to respect the maximum cable length of 100 m for 100-BaseT connections between the end-node and a high performance Fast-Ethernet switch. This switch usually has a 100 Mbps uplink to the closest regional router that may be upgraded to Gigabit if needed. Close to regional routers, 1 Gbit/s Ethernet may easily be made available if needed.

Active equipment in the network architecture is installed with a double power supply connected to two independent power sources (normal and secured), in order to ensure proper connectivity even in case of a power-cut on either 220 V distribution.

2008 JINST 3 S08001

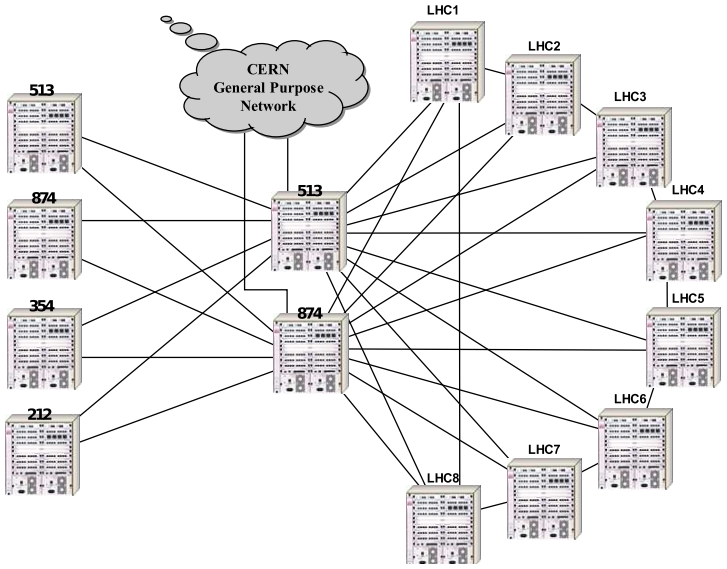

Figure 9.2: The CERN Technical network.

Because the technical network can be completely isolated from the external world, independent dedicated redundant name and time servers have been implemented. These servers are installed at different locations and connected to different regional routers to ensure full redundancy of the system. It must be noted that because the technical network infrastructure is interconnected with the general purpose network, security break-ins can be attempted on the devices connected to this infrastructure. End nodes security survey and updates must be made regularly.

9.3 Equipment access

9.3.1 The VME and PC Front End Computers

Most of the non-industrial accelerator hardware in the PS, SPS, and LHC sites is already connected to the control system via VME or PC based Front End Computers (FEC). These systems run a real-time operating system called LynxOS from LynuxWorks [46], or Red Hat Linux [47]. There will be several hundred FECs distributed in the surface buildings of the LHC and in some cases in the underground areas. Wherever possible, they will be diskless to increase reliability and will boot over the network.

A set of commercial or CERN-made hardware interface boards is being standardised for the LHC, together with the necessary device drivers (e.g. timing generators, timing receivers, beam interlock controllers, digital acquisition boards, WFIP cards, digitisers, etc.). The type of hardware (VME or PC), as well as the Operating System (Linux or LynxOS), will be selected according to the performance needed and the availability of the specific drivers.

These FECs will execute the equipment access software which is part of the *resource tier* (figure 9.1). This software accesses data from the hardware interface boards and provides it, either via subscription or command/response, to any software agent in the control system.

2008 JINST 3 S08001

A dedicated FEC software framework (section 9.7.1) has been developed in order to simplify and standardise the software running in the FECs.

9.3.2 The PLCs

PLCs are increasingly used for controlling industrial equipment for LHC. This type of controller can be chosen when the process is not synchronised to accelerator timing and when the sampling period required is not smaller than 100 msec. PLCs offer ease of programming process logic via standard high-level languages (IEC-61131-3), cost efficiency, high reliability and adaptation to industrial environment. They are used in typical industrial controls fields such as electricity, water, gas, cooling, ventilation, but also in accelerator-specific fields of machine cryogenics, interlocks, vacuum, radio-frequency and beam extraction. PLCs are generally part of a complete subsystem including supervision (section 9.7.6). In a few cases, they are accessed through VME FECs. As with any other FEC in the Controls Infrastructure, PLCs benefit from generic facilities such as remote reset, monitoring and optional clock synchronisation.

9.3.3 The supported fieldbuses

The two fieldbuses used in the LHC accelerator, namely WorldFIP and Profibus, are among the three fieldbuses recommended at CERN. They are both supported by an internal CERN service. These fieldbuses typically allow longer distance and more robust protocols than Ethernet. In addition, they are the only means of connecting equipment located in the LHC tunnel or other radioactive areas.

9.3.4 The WorldFIP fieldbus

Table 9.1 summaries the LHC WorldFIP installation needs. WorldFIP is selected when the following features are required:

- Determinism (up to10 μs); is required for:
  - The real-time control and synchronisation of LHC equipment.
  - High precision time distribution.
  - The management of periodic data.
- Robustness in severe environments and in particular:
  - Its resistance to electromagnetic noise (level 3).
  - Its good resistance to high radiation levels (based on robust semiconductors and a magnetic coupling ensuring galvanic isolation).
- Data Rates:
  - WorldFIP will be used to control largely distributed LHC systems at high data rates (1 MBits/s and 2.5 MBits/s).
  - High load factor possible (70 to 80% of the network bandwidth).

2008 JINST 3 S08001

Table 9.1: WorldFIP Needs for the LHC machine.

LHC System	Required Time precision (ms)	Cable length (km)	Data rates
Magnet Protection	1	61	1 MBit/s
Power Converters	0.01	45	2.5 MBit/s
Beam Instrumentation	500	50	31.25 KBit/s
Radio Frequency	1	5	1 MBit/s
Cryogenics	500	90	1 MBit/s
Survey	1000	44	31.25 KBit/s

9.3.5 The Profibus fieldbus

Profibus has been selected for several applications in the LHC, such as fast kicker magnets, cooling and ventilation, vacuum, cryogenics, magnet and power interlock controllers. The main reasons to select Profibus for these applications are:

- The robustness of the protocol and simplicity of configuration.
- The large variety of remote I/O systems available with Profibus.
- The ease of integration with Siemens PLCs.
- The availability of radiation-tolerant remote I/O on Profibus.
- Its capacity to be used as an instrumentation bus (Profibus-PA) with a wide range of instrumentation products offering Profibus as standard means of connection.

9.4 Servers and operator consoles

The upper layers of the LHC control system (figure 9.1) will be deployed on operation consoles and fixed displays, files and applications servers to meet the requirements of the LHC applications software. The servers will be used to run the LHC middle-tier software, to host operational programs and data files, and also to offer specific services (web services for operation, fixed displays, SCADA servers, Database servers, etc.). The servers will run the Linux operating system. Emphasis will be put on the hardware reliability and availability issues by selecting multi-CPU architectures with redundant and hot swappable power supplies, discs and fans. Mirroring and RAID techniques will be used to ensure data integrity and error recovery.

9.5 Machine timing and UTC

9.5.1 Central beam and cycle management

The LHC beam production involves a long chain of injectors which need to be tightly synchronised. Moreover, the different types of beam to be produced for LHC are only a subset of the beams that the injectors have to produce for the rest of the experimental programme. All these beams are

produced in sequences of typically a few tens of seconds. The composition of these sequences changes many times a day, and the transition between different sequences has to be seamless. To manage the settings of the machines and to synchronise precisely the beam transfer from one injector to the other up to the LHC, a Central Beam and Cycle Manager (CBCM) is required. The CBCM will:

- Deliver General Machine Timing (GMT).
- Provide Beam Synchronous Timings (BST).
- Elaborate the "telegrams" specific to each machine and broadcast over the GMT networks.

A "telegram" describes the type of beam that has to be produced and provides detailed information for sequencing real-time tasks running in the FECs and for setting up equipment.

The CBCM drives seven separate GMT networks dedicated to the different CERN accelerators, including the LHC, and four BST networks, of which three are for the LHC and one for the SPS. To achieve extreme reliability, a CBCM runs in parallel on two different machines, with a selector module capable of seamlessly switching between them.

9.5.2 Timing generation, transmission and reception

Several hardware timing modules are required either to produce reference clocks and timing events, or to extract the timing pulses or the interrupts needed to drive the equipment from the distributed timing data. The following is a list of these modules.

- The CTSYNC module provides the "master" clocks from a 10 MHz oscillator which has a stability which is better than 10^{-10}.
- The CTGU module encodes events to be transmitted over the GMT network: the millisecond events, machine timing events, telegrams, and the Universal Coordinated Time (UTC) events.
- The CTGB is the driver module for the BST network; it uses the TTC technology [48]. The CTGB will also send the 1 pulse per second (1PPS) UTC time and some telegram events over these networks, hence its inclusion in the CBCM.
- The CTRP is a GMT multi-channel reception module. It comes in three formats: PMC, PCI, and VME and can generate timing pulses with very low jitter. The CTRP recognises the various types of timing events: UTC time, the telegrams, the millisecond and other machine events and can use them to trigger pre-programmed actions.

9.5.3 UTC for LHC time stamping

A key requirement for the LHC control system is the need for a timing reference to timestamp the accelerator data. UTC has been adopted as the standard of date and time for all CERN accelerators [49, 50]. UTC is the basis for the worldwide system of civil time. The motivation behind this decision was to eliminate the problems associated with the changes between the winter and summer standards, particularly for the change in October (CET changes from UTC+2H to UTC+1H) when the time is repeated for one hour.

2008 JINST 3 S08001

Table 9.2: Required LHC time stamping accuracy.

System	Timestamp Accuracy
Beam Dump	$<$ 0.05 ms
Beam Instrumentation	
Radio Frequency	
Injection (Kickers...)	
General Machine Timing	
Machine Interlocks	1 ms or better
Quench Protection	
Power Converter	
Feedbacks (orbit, tune ...)	1 – 10 ms
Cryogenics	
Vacuum	
All other systems	

9.5.4 UTC generation, transmission and reception

The source of date and time for all CERN accelerators will be a Global Positioning System (GPS) time receiver connected to the CTGU. The GPS provides UTC referenced date and time plus a very accurate 1 PPS tick, which is used by the CTGUs to synchronise the UTC time.

At initialisation the CTGUs receive the date and time information from the GPS receiver then run independently, using the clock delivered by the CTSYNCH until a manual resynchronisation is carried out. CTGUs provide the time every second in UNIX standard format. UNIX time can then easily be converted by software into year, month, day, hour, minute and second.

UTC time can be provided by the CTRP modules to timestamp the data to be logged with a 25 ns resolution. Using an optional CERN-developed chip the resolution can be brought down to 1 ns. The required time-stamping granularity for the different LHC systems [51] is shown in table 9.2. The 1 ns performance is required principally in the injector chain.

9.5.5 NTP time protocol

PLCs will use the Network Time Protocol (NTP) [52] to synchronise the time (section 9.2.2). NTP provides accuracies typically within a few milliseconds on LANs and up to a few tens of milliseconds on WANs. This accuracy is sufficient to properly tag most of the PLC data. Some PLCs that require higher precision for time-stamping their data will use the high precision 1 PPS tick from the CTRP; in this case, NTP will be used only to relate this tick to UTC time.

9.6 Data management

The LHC Machine will be of unparalleled complexity in terms of number of components and in the manipulation of operational parameters. A large number of signals will be made available to the control room, including over 100'000 from the cryogenics, machine protection, and beam

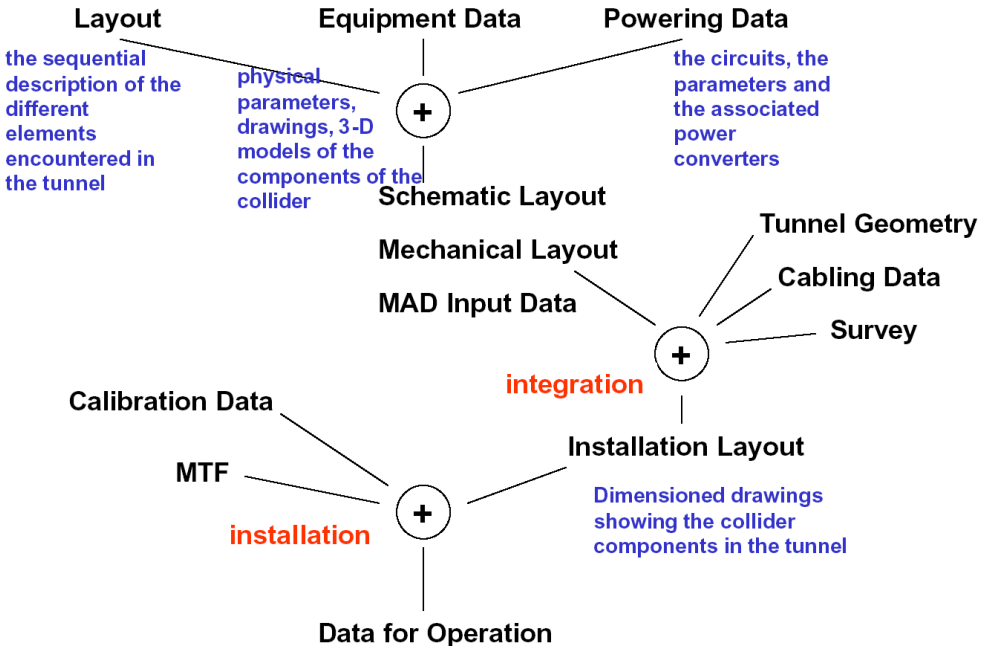

Figure 9.3: LHC Reference Database.

monitoring systems. Over 1'600 electrical circuits will power a wide variety of magnet systems. The quantity of data to be recorded for diagnostics after a "beam abort" has been estimated as a few gigabytes; during Hardware Commissioning, it is anticipated that 1 terabyte of information will be accumulated. Operation of the machine will rely on stringent control of the hardware settings. A complete beam operational history will be created in order to build a long term understanding of the accelerator performance. Such a variety and volume of data cannot be exploited without data management tools. Building on the experience of LEP construction and operation [53], Oracle has been selected as the strategic choice for data management at the LHC.

9.6.1 Offline and online data repositories

As shown in figure 9.3, the LHC Reference Database is being populated during the machine construction in order to manage the information essential for integration studies, installation, and hardware commissioning. This offline Database will be one of the sources of data that is required to configure the online LHC control room application software, providing essential information such as the layout, accelerator optics, and calibration information. The electrical circuit data, described in the next section, are an example of the use of the database both during construction and operation.

While the Reference Database will be an offline source of information during machine operation, several online databases — alarm archives, periodic data logging, Post Mortem event archive, and accelerators settings management — will be required to capture the operational history. These are described in the later sections of this chapter.

9.6.2 Electrical circuits

The powering layout of the LHC is extremely complex, comprising 1'232 main dipole magnets, about 450 quadrupole magnets, and several thousand corrector magnets, powered in 1'612 electrical circuits. The currents feeding the magnets range from 60 A (for small correctors) to 12 kA for main dipole and quadrupole magnets, while the energy stored in the superconducting magnets is about 10 GJ. About 80'000 high current connections have to be made and tested in the tunnel during the installation and commissioning phases.

To ensure the correct connection of the superconducting bus bars around the 27 km long machine, a detailed description of the electrical circuits and their elements has been added to the layout description of the machine in the LHC Reference Database [54]. This description has been linked to the existing data describing the physical and magnetic layouts. Changes of this powering information are very rare, and updates have to be performed only if there are major changes in the physical or powering layout of the machine. Information about the powering will be provided to users to:

- Perform the interconnections between cryo-assemblies in the long arc cryostats.
- Configure the powering interlock PLCs.
- Create MAD input files to calculate the beam optics.
- Display detailed circuit and equipment information via the web and the LHC equipment catalogue.
- Configure hardware state and beam parameter control room application software.

Integrity will be ensured by using the same information to generate the machine optics, make the circuit connections during assembly, configure the behaviour of the interlock system, and operate hardware and beams.

9.6.3 Control system configuration

The control system is composed of a large number of application programs, a middleware layer, VME- and PC-based FECs with control software, industrial PLCs, fieldbuses, and hardware interface modules. These components have shared characteristics, associated with their type or class, and specific addresses and parameters associated with their instances. All these data will be stored and described in an Oracle relational database, the *Controls Configuration Database*. This centrally managed configuration information allows the use of generic software on all levels. A layer of Java and C++ interfaces will enable all control room software to be data-driven, by fetching at runtime the layout, interconnection, access methods, and parameters of remotely controlled LHC equipment. All FECs can be automatically configured and bootstrapped with files generated from the database. Thanks to a generic design, the database also holds the controls configuration of the injection accelerator chain of the PS and SPS complexes. This common source of data will make it possible to have uniform controls data management throughout the accelerator chain, as well as effective sharing of tools and methods that use this data [56]. On top of the Configuration Database,

a Web browser service will allow exploration of all the information about the underlying controls hardware and software.

9.7 Communication and software frameworks

9.7.1 FEC software framework

The software running in the LHC FECs will be developed using a specific framework, known as FESA. This framework is a complete environment for the equipment specialists to design, develop, test, and deploy real-time control software for the FECs. This framework is also the new standard for the LHC injector chain. The recurrent components of the equipment control software have been formalised into a generic model. It defines the structure of the real-time part of the software, which handles hardware and interrupts, together with the structure of the server task that handles external requests. In order to enhance the productivity and the maintainability of the code, a clear separation between the generic and the equipment specific parts is enforced, and automatic code generation is extensively used. As a result, the ratio between the generic/generated code and the developer-specific code is usually around 10 to 1. An object-oriented design and a C++ implementation have been used for supporting this approach.

For building the control software, the programmer is provided with a set of configuration tools for the creation and evolution of any FESA component. These tools are directly linked to the control system configuration DB (section 9.6.3). They are used for the definition of internal data structures, syntax of external commands, and binding between hardware events or external requests and treatment code. External requests are handled by dedicated server tasks that fully implement the device access model of the Controls Middleware, which is described in the next section. Compound data types are supported for data coherency and for performance purposes.

System-level services are provided for the operation of the FECs: handling of errors from the equipment software, monitoring of processes, etc. These dedicated services preserve the real-time behaviour of the FECs, and they support remote control for adjusting parameters, such as the trace level of a task, or for executing corrective action like restarting a process. The developers and the users (equipment-specialists, operators, and maintenance team) are also provided with a set of interactive tools, test programs and Java libraries which allow immediate testing of all the functionality of the software during its development and operation. These facilities are all generated in an automatic manner, without any specific code being required. The FESA framework is built for LynxOS and Linux platforms. The software tools are written in Java and are thus platform independent. No commercial software requiring a run-time license has been used.

9.7.2 Controls Middleware

The Controls Middleware (CMW) is the ensemble of protocols, Application Programming Interfaces (API), and software frameworks, which allow seamless communication between the software entities of the control system. Two conceptual models are supported: the device access model and the messaging model. The device access model is mainly used in the communication between the

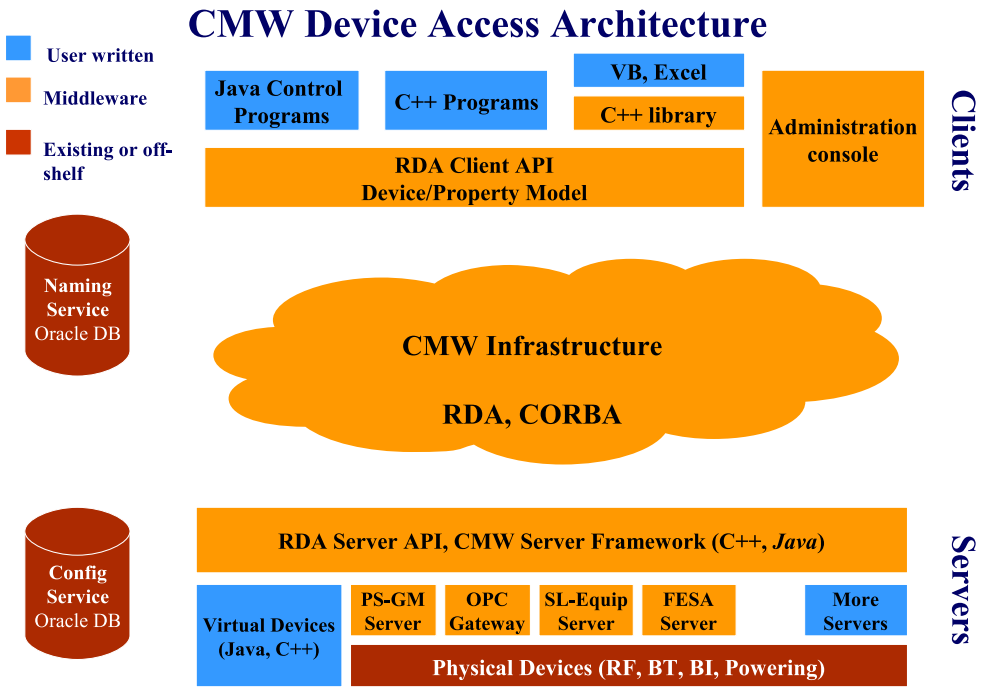

Figure 9.4: The Device Access Model.

resource and middle tiers, while the messaging model is mainly used within the business tier, or between the business tier and applications running in the presentation tier.

9.7.3 Device access model

Figure 9.4 depicts the full architecture of the CMW device access model. The typical use of this communication model is between Java applications running in the middle tier and CMW equipment servers running in FECs. These processes communicate together through the Remote Device Access (RDA) API, which is available for both Java and C++ languages.

Within the device access model, all accelerator equipment can be accessed as “devices”. A device is a named entity within the control system, which normally corresponds to a physical device (e.g., beam position monitor, power converter) or to a virtual controls entity (e.g., transfer line). Conceptually, each device has a number of properties that characterise its state. By getting the value of a property, the state of the device can be read. Accordingly, a device can be controlled by setting one of its properties with the required value. ‘Get’ and ‘Set’ operations can be either synchronous or asynchronous. In addition to the Get and Set operations on device properties, it is possible to monitor changes to the property via listeners (callbacks). It is often necessary to specify under which conditions the operation has to take place, for instance, to which cycle of the machine the operation applies. These conditions are commonly referred to as context or filter. A list of properties, together with their type description and the semantic of the assessors, is referred to as a contract. Contracts typically serve a specific purpose, such as measurements or settings of a device.

9.7.4 Messaging model

Complementary to the device access model, the messaging model allows any software process to send and receive asynchronous messages in a loosely coupled way. Various application programs for accelerator controls naturally exchange data via asynchronous messages: timing and logging events, notifications, and alarms are typical examples of data which is asynchronously generated from a various set of producers, and which is potentially of interest for multiple consumers. The LHC control system software relies on the Java Message Service (JMS) as the messaging solution for Java-based controls application. The JMS specification adds a standard and vendor-independent API that enables the development of portable, message-based applications in the Java programming language. Moreover, JMS is a strategic technology for the Java 2 Platform Enterprise Edition (J2EE). The LHC Alarm Service is a typical example of a messaging system relying on the publish-and-subscribe paradigm: it subscribes to alarm notifications published by several processes and, after processing, redistributes the result to dedicated consoles and any other controls software component that subscribed to the appropriate alarm content hierarchy.

9.7.5 The J2EE framework for machine control

Applications controlling the LHC beams must handle a great variety of tasks, such as visualisation of data, and significant computation, together with database and equipment access. These applications rely on several services such as security, transactions (to make sure that complex operations are performed automically), and remote access or resource management. The requirements dictate a move towards a modular and distributed architecture. Such an architecture offers a clear separation between the GUI (the user interface), the control (the core business of the application), the model (the abstract model of the accelerator control), and the devices (the physics equipment) that are controlled.

For LHC, a 3-tier architecture is being implemented, in which the middle-tier is responsible for providing all services and for coordinating the client applications running on the operator consoles. In this architecture, shown in figure 9.4, applications are running both in the operational consoles of the presentation tier and in dedicated servers of the middle-tier. The consoles are responsible for the GUI applications and translate the operator's actions into command invocations in the middle-tier. The middle-tier, through its centralised and shared processing power, is in charge of accessing databases and accelerator devices. The middle-tier also ensures the coherency of operator actions and shelters resources. It enforces separation between presentation and application logic and provides shared services to many clients.

To achieve this new programming model, a platform is needed to support the middle-tier. The platform provides the infrastructure with all the necessary common basic services, i.e., all parts of a control system that are not specific to accelerator controls. By using such a platform, the developers can concentrate on writing code for the accelerator control components, such as parameter management, setting generation, cycle handling, and trim management; they do not have to write system level services. The platform of choice today is the J2EE, which is an industrial standard defined by a set of specifications and APIs, implemented by many vendors. J2EE is based on the Java programming language and on a component model known as Enterprise JavaBeans (EJB). Components are the key technology used to write a modular object-oriented distributed

2008 JINST 3 S08001

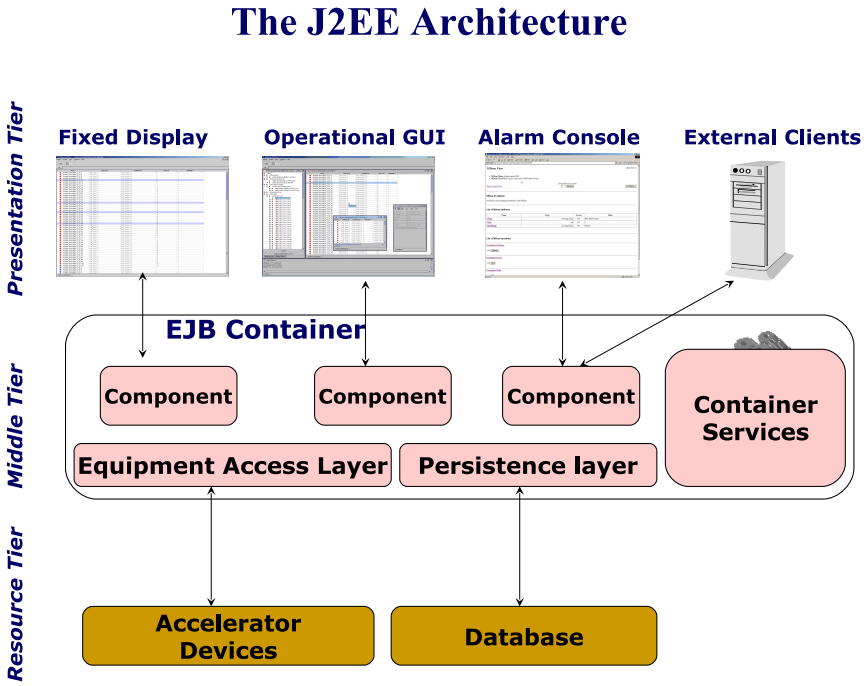

Figure 9.5: The J2EE Architecture.

application. EJB components are "Lego® pieces" that implement a set of interfaces, allowing them to run inside a so-called "EJB container".

As shown in figure 9.5, developers write accelerator-specific code in the form of EJB components, and the EJB container provides the necessary infrastructure to run them, such as remote accessibility, components management, concurrency support, fault tolerance, high availability, scalability, resource pooling, transaction management, and security.

To summarise, the 3-tier approach and the J2EE framework are the basis on which the controls for the LHC are built. This allows effort to be focused on the LHC controls challenges and not on the infrastructure. It exploits existing technology and provides the modular and distributed architecture needed.

9.7.6 The UNICOS framework for industrial controls

Reflecting a trend of the last decade, a single industrial supplier has been chosen for the procurement of the hardware and software for the control of the cryogenic equipment and experimental magnets installed in the LHC accelerator complex. This control system is referred to using the acronym UNICOS, standing for UNified Industrial COntrol System. UNICOS builds on a classic industrial controls architecture, using the PVSS (object-oriented process visualization and control system) SCADA in the Supervision layer, Schneider Quantum PLCs for process control at the Control layer and Schneider Premium PLCs to connect process channels in the Field layer (an alternative is Quantum remote I/O). Communication is based on Ethernet and FIPIO (Factory Instrumentation Protocol Input Output). The software design is an evolution of an object-oriented philosophy used with former control systems [55]. In this approach, each process device (sen-

sor, actuator, or set of devices constituting a process entity) is modelled as an object. This object integrates the process behaviour and the GUI.

9.7.7 The UNICOS object model

In the UNICOS concept, the object implementation is split into two parts:

- The GUI functionality - programmed in the SCADA.
- The process behaviour - programmed in the PLC.

The GUI part includes the interaction with the operator by means of graphical components and dedicated panels called “faceplates”; these GUIs inform the operator of the object status, and allow him to send orders. The PLC part contains the process behaviour of the object. The object is linked to the plant through I/O channels that may be linked to the PLC via either a fieldbus or a direct I/O connection. The SCADA and PLC object parts are connected together by a communication layer based on the TCP-Modbus protocol. As shown in figure 9.6, each object has several interfaces and receives the following information:

- Requests from the operator via the SCADA object. These requests are transmitted to the PLC by manual requests through the communication middleware.
- Configuration parameters (GUI or PLC) set during the programming phase and accessible for modification by a program specialist.
- Information from the process (process inputs), consisting of analogue or binary values from sensors and status of other objects.
- Orders from the control logic programmed into an object of a higher hierarchical level via the Auto Requests.

Three main types of object are defined in the UNICOS architecture:

- Input-output objects which provide the interface to the plant. They link the devices and actuators to the control system. Some basic treatments may be embedded in these objects. Input/output channels are exclusively accessed through such objects.
- Field objects that represent hardware elements, such as valves, heaters, motors, and other devices. Each type of field object has its own associated process logic. This logic integrates specific functions (e.g., alarm generation, transition between set points, interlocking).
- Process control objects that control equipment units grouping several field objects and/or other process control objects. The process logic of these objects is split between a standard part (insuring a homogeneous interface to the environment) and a specific part to handle the process to control.

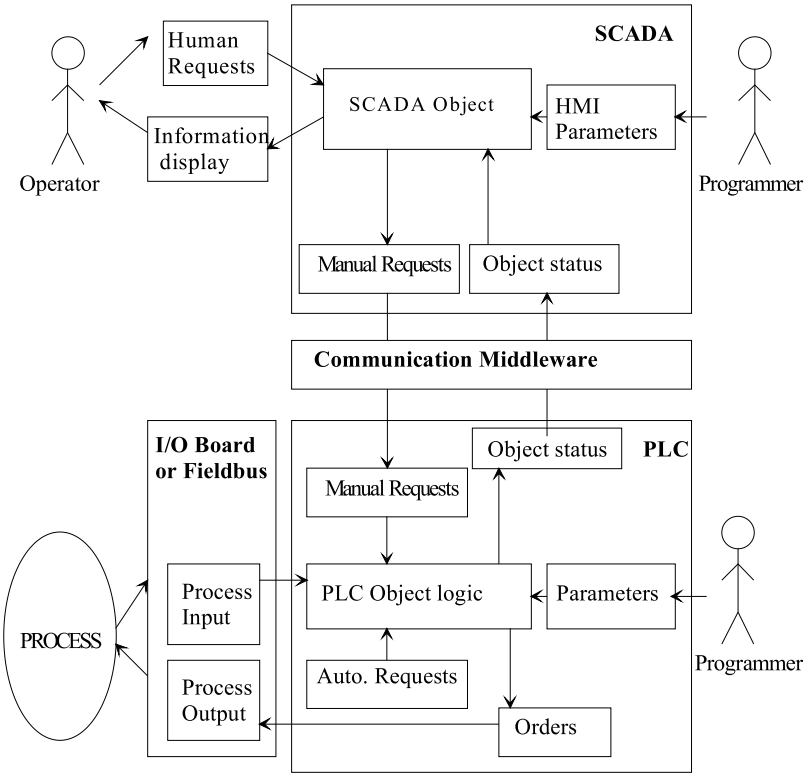

Figure 9.6: The UNICOS Object Model.

9.8 Control room software

9.8.1 Software for LHC beam operation

The LHC control room applications will be based on a set of common control system services and components, in order to avoid duplication of effort and solutions. The operational software development process relies on common development tools, guidelines, and procedures for construction, testing, integration, deployment, and change management.

9.8.2 Software requirements

The LHC aims at injecting, accelerating, and colliding beams with well controlled beam parameters in an efficient, reliable, and reproducible manner. This is a non-trivial task, since the small aperture, the high stored beam energy, and the sensitivity of the machine to beam losses impose very tight accelerator physics constraints. The superconducting magnets will generate field errors that have large static and dynamic components. The destructive power of the LHC beams and the low tolerances to beam losses will place very stringent demands on the LHC control system. A non-exhaustive list of the main software application categories for LHC is given below:

- **On-line monitoring** of the state and tolerances of every LHC component and interlock system.

- **Measurements** to allow for synchronised acquisition as part of scans, feedback, or at a given point in the ramp.
- **Real time control** to implement feedback on key beam parameters (orbit, tune, chromaticity), plus the feed-forward of corrections from the reference magnets.
- **On-line modeling** including calculation of parameter variations, and feed-forward from the machine into the model.
- **Automatic sequencer** to allow complicated sequences for injection and ramp to be performed systematically.
- **Fixed displays** including transfer lines, beam size, bunch currents, mountain range turn-by-turn, tune FFT, beam loss monitors, global orbit plus crossing angles, beam separation, and transverse position along a batch.
- **Settings management system** to deal with the generation and management of complex settings.
- **Fast, accurate control of the main beam parameters** in both the physics coast and during injection and ramping.
- **Trim facility** incorporating persistent versus geometric correction, scaling, accumulation, smoothing out of errors and corrections, and time and history dependency of errors.
- **Communication with experiments** and other major systems (cryogenics, vacuum, technical services).
- **Scripting environment** for ad-hoc rapid application development for machine development.
- **Miscellaneous support applications** such as an Electronic Logbook.
- **Console Manager** to launch and manage controls applications in a unique manner.

9.8.3 The software development process

The software development follows the Unified Software Development Process, a practical software process model followed by many industrial OO projects. Java is the programming language chosen for the implementation of the high-level services and control room applications, as it enables platform-independent development. XML is playing an increasingly important role in the exchange of data.

The software developers are provided with a suite of selected software tools for code optimisation, quality assurance, and testing, to guarantee the quality of the control room software. Software configuration management facilities are provided on top of the archive engine (RCS) to provide version management services, making it possible to trace and identify any component of an operational application, and to deliver consistent operational software releases. In addition, tools for code building and distribution are available to release operational software components in a multi-versioned repository residing on dedicated fileservers (section 9.4).

Operational software, once released, is deployed locally to the operators' consoles using standard Java-distributed deployment technology, which guarantees automatic delivery of software updates without operator intervention. Local deployment ensures the speed and performance required by operations.

9.8.4 Software for LHC Industrial Systems

Several LHC accelerator industrial subsystems will be accessible from the control consoles via a SCADA supervision interface. SCADA systems are used in a wide variety of industrial domains and therefore typically provide a flexible, distributed, and open architecture to allow customisation to a particular application area. In addition to basic SCADA functionality, these systems also provide strong support for GUI tools, a set of standard interfaces to hardware, as well as an API to enable integration with other applications or software systems. PVSS, the industrial SCADA product recommended at CERN, has the following specific features:

- It can run in a distributed manner.
- It has multi-platform support (Linux and Windows).
- It is device oriented with a flexible structural concept.
- It has advanced scripting capabilities.
- It has a flexible API, allowing access to all features of PVSS from an external application.

The SCADA development process, in particular when based on frameworks such as UNICOS (section 9.7.6), can be based mostly on configuration and graphic editors rather than programming. A unique device configuration database contains all characteristics of the devices, including device addressing, alarm, and data logging parameterisation.

9.9 Services for operations

9.9.1 Analogue signals transmission

While most signals from the LHC will be digitised for internal usage by the respective accelerator systems, some are required for visualisation during tuning and measurement. These include some 200 signals, mainly in the 0 to 10 kHz range from the RF system, and over 300 signals with a typical bandwidth of 50 MHz from the kicker systems for injection and beam extraction. In the injector chain, the nAos system [56] is successfully used to fulfil the need for coherent acquisition and display of analogue signals. The system's main feature is the digitisation of signals in acquisition crates as close as possible to their sources, thereby ensuring optimal fidelity. Whenever possible, in order to save costly oscilloscope modules, signals are multiplexed, allowing some 100 signals to be serviced by one crate containing four two-channel oscilloscope modules. The triggers for all the oscilloscope modules are elaborated from the GMT distribution (section 9.5.1), ensuring a precise time-correlation between signals. The digitised signals are sent through Ethernet to control room

2008 JINST 3 S08001

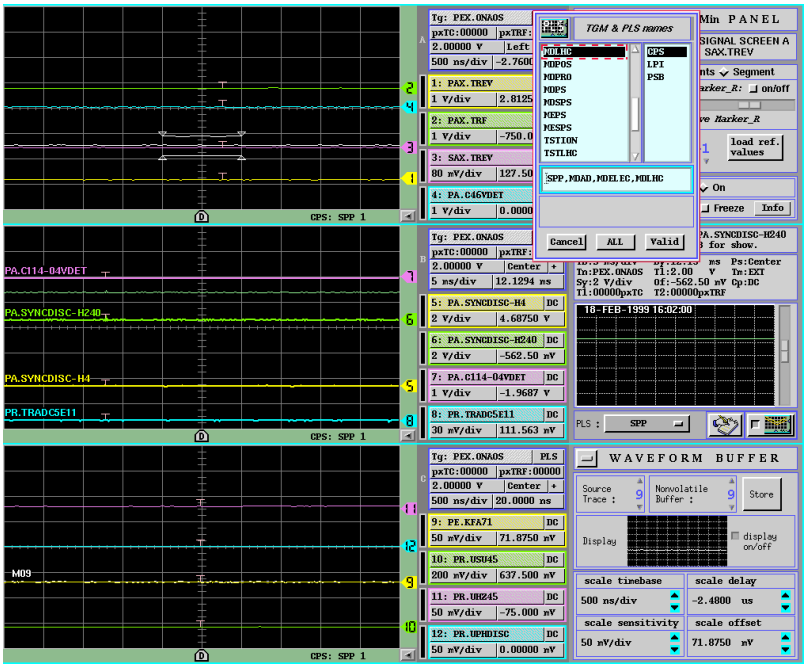

Figure 9.7: Snapshot of nAos GUI.

consoles, where a GUI application (figure 9.7) allows operators to monitor them on four-channel "virtual oscilloscope" screens.

The nAos system is currently being upgraded to more modern, commercial hardware and software platforms. The new version of the system is called Open Analogue Signals Information System (OASIS). It will implement all the above features, including arbitration of resource (i.e., oscilloscopes), allocation to clients, and additional functionality for LHC requirements. This approach will ensure that signals from the injector chain may also be integrated into LHC operation; the older, lower energy machines are more richly instrumented by this system. As with all LHC systems, OASIS must provide information to the Logging (section 9.9.3) and Post Mortem (section 9.9.4) systems. Analogue signals will be monitored by compact PCI-based acquisition systems; the total number of crates will depend on the amount of multiplexing achievable with regard to the number of clients.

9.9.2 Alarms

The detection, collection, management, and distribution of information concerning abnormal situations ranging from severe alarm states to warning states, hereafter referred to as Fault States (FS), uses one global system. The system will accept information concerning any problem associated with the CERN accelerator complex and offer this information, in a structured way, to any interested party. The core part of this system will be the LHC Alarm SERvice (LASER), [57]. As a result of a detailed technology survey, a 3-tier architecture was chosen and is now being implemented according to the J2EE specification (see 9.7.5).

FS detection is performed (figure 9.5) in the resource tier by surveillance software written

by equipment specialists, application programmers, and operators. The LASER system offers a standard interface for these sources to push a FS, by means of either a C or Java API based on the messaging system. FS time stamping at the point of detection, using UTC time, down to the microsecond level if needed, will be crucial in correlating data. The middle-tier will collect the FS by subscribing to all FS sources. A number of services and facilities will be offered at this level including:

- FS persistence.
- Logging of FS transitions using the LHC Logging facility.
- FS dependency and reduction management.
- FS browsing and interaction.
- Administration of the overall system.
- User and configuration management.
- Scaling, using clustering facilities.
- FS distribution according to a FS hierarchy, representing the interest of users.

Finally, the presentation tier will allow clients interested in those services to access them via a Java client interface. The major user of this interface will be the alarm console, which will be used by equipment groups and control centres to select, receive, and display FS, and in order to access all LASER services. Important configuration facilities will allow the alarm console to be personalised.

9.9.3 Logging

The information to be logged is heterogeneous, but usually concerns values of variables that evolve over time, for example cryogenic temperatures, vacuum pressures, magnetic field measurements, bunch intensities, or beam profiles. The total number of logging variables will be on the order of 10^5-10^6, with logging frequencies up to 0.1 Hz. The main purpose of LHC logging is to improve the machine performance and that of the individual subsystems. Therefore the recorded data will be stored and kept accessible over consecutive years, in order to permit comparison of historical data, off-line analysis to search for data correlations, and compilation of operational statistics.

For each of the logged records of the data variables, the date-time value is a key attribute, and therefore correct *"time stamping"* is vital for the exactness of the data. For time stamping, the usage of UTC is endorsed throughout the LHC Logging system (section 9.5.3), with microsecond precision where applicable. In order to enable effective exploitation of the logged data by users such as equipment engineers, control room operators, machine physicists, and CERN managers, the LHC Logging system will employ web-enabled interfaces to graphically visualise the information and to extract it to common desktop tools.

2008 JINST 3 S08001

9.9.4 Post mortem

The LHC will be protected by over 10'000 interlock channels, thousands of quench detectors, and around 3'000 beam loss monitors. Many tens of thousands of signals will be available for the operators and engineers to interpret the circumstances of Beam and Power Aborts. The Quench Protection System alone will provide around 80'000 signals for Alarms, Logging, and Post Mortem purposes. Recovery from a perturbation to the nominal magnet cycling will require a minimum of 2 hours (7 TeV back to 7 TeV), which precludes learning by trial and error, as practiced with normal conducting machines.

The LHC Post Mortem System (figure 9.8) [51, 58] is required as a suitable diagnostic tool. The purpose of this system is:

- To ensure comprehensive monitoring of machine protection systems.
- To improve the operational efficiency of the LHC by:
  - Providing diagnostics of power and beam aborts - the aim is to quickly identify the origin of failures in order to initiate appropriate action and restore operation,
- Building long-term understanding of accelerator operation,
  - Providing diagnostics for beam losses resulting from equipment malfunction.
- To explain the mechanism if damage occurs.

To achieve these aims, the post mortem data must be *complete* and *coherent* across systems. While it is intended to make full use of diagnostics from Alarms and Logging, these systems will not provide all the facilities required for the understanding of quenches and beam losses in the LHC. In particular, the underlying hardware must capture transient data pertaining to conditions prevailing before, during, and after such events. These transients may be relatively slow when they are provoked by quenches; indeed, in this case, the Logging system will gather some pertinent information, such as cryogenic temperatures. However, in the case of beam losses, the timescales are extended downwards to turns of the machine.

In order to satisfy the aims of being complete and coherent across all LHC systems, the following essential ingredients are required:

- Every piece of LHC equipment and diagnostic system must implement a circular Post Mortem buffer of appropriate depth holding the latest data (for example, 1'000 turns for beam instrumentation).
- This data must be time stamped using a common clock, with a precision related to the corresponding process (see table 9.2).
- The Post Mortem buffers must freeze at the Post Mortem timing event, or by self-triggering in the case of the protection systems.
- Data must be self describing, so that it can be managed by the event builder and analysed by generic tools.

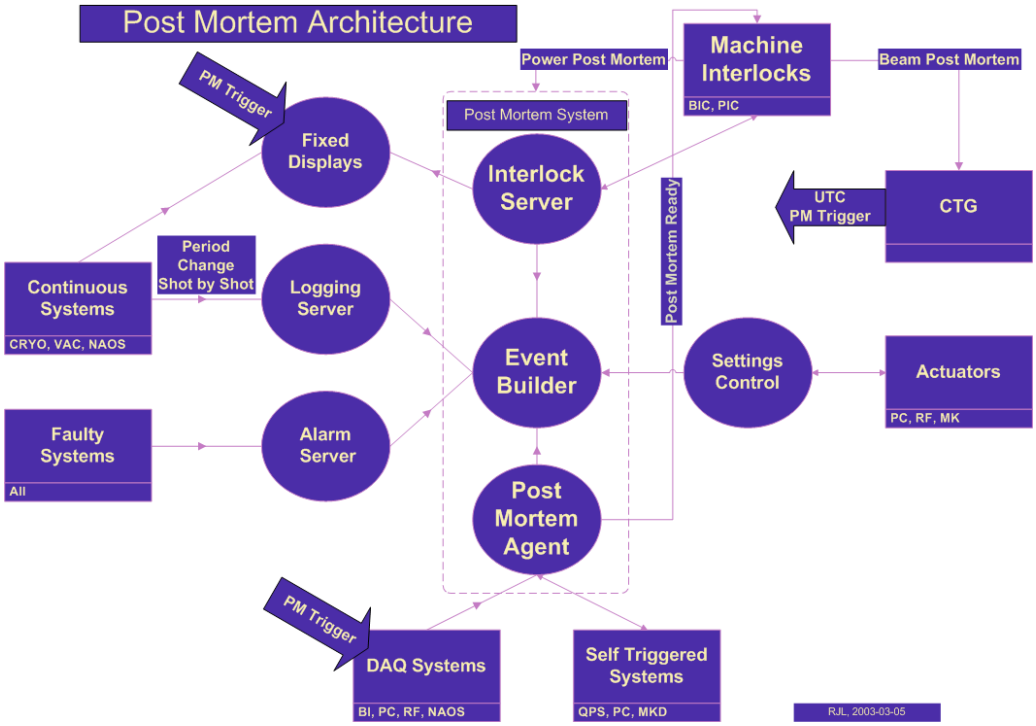

Figure 9.8: The LHC Post Mortem Architecture.

The Post Mortem events will be large - several gigabytes - and therefore analysis must be automatic in order to generate digested information for operators. After analysis, the information must be stored in order to build up the longer term history and understanding of the machine. The most relevant data will be stored for the lifetime of the LHC.

2008 JINST 3 S08001