MODELLING OF CONTROL SYSTEM ARCHITECTURE FOR NEXT-GENERATION ACCELERATORS

Shi-yao Liu* and Shin-ichi Kurokawa
National Laboratory of High Energy Physics, Tsukuba, Japan

Abstract

Functional, hardware and software system architectures define the fundamental structure of control systems. Modelling is a protocol of system architecture used in system design. This paper reviews various modellings adopted in past ten years and suggests a new modelling for next generation accelerators.

1. Introduction

System architecture has three basic features, i.e. functional, hardware and software architectures. Functional architecture denotes the hierarchical structure according to functional layers or levels. Hardware architecture is the hardware hierarchy used to implement a functional hierarchy. Software architecture is the software hierarchical layers used to realize the functional architecture. These three professional architectures define the fundamental structure of a system. Logical relationships between these three may be or may not be consistent: a functional hierarchy can be realized with hierarchical or flat hardware topology; one hardware architecture can be the basis of different software architectures. All three architectures aim at the same purpose, i.e. to achieve high effectivity, reliability and flexibility in control system.

Next generation accelerators requires new modellings for control system architectures. This paper overviews the historical evolution and modellings adopted in some representative control systems and suggests a few modellings of control system architectures for next generation accelerators.

2. Historical Evolution

Summarizing the history of evolution for accelerator computer control systems during 1964-1989, six important techniques can be identified as milestones:

1964 : Real time data acquisition : LAMPF linac
1969 : Computer communication link : FERMI NAL system
1975 : Computer network technique : CERN SPS system
1980 : Microprocessor technique : SLAC SLC system
1981 : Local Area Network technique: KEK TRISTAN sys.
1986 :Internetworking LAN technique: CERN LEP system

Microprocessor and LAN techniques are two largest impacts on control system architectures and provide the fundamental technical support for distributed systems, either geographically or functionally. They have radically changed the features of system architectures from centralization to distribution. Key technical items involved in evolution of accelerator control systems are:

NAL : Multi-computer system with master satellite mode
SPS : Explosive central to fragment, star link network
SLC : Using microprocessor SBC as MICRO and COW node
TRISTAN : Flat control system architecture on LAN
LEP : Multi-LAN and multi-processor interconnect

From the historical evolution veiw-point, we classify the accelerator control systems into seven types: 1. Single stand-alone system. 2. Multi-computer with communication link. 3. Multi-computer distributed system based on LAN. 4. Multi-computer based on LAN and local intelligence. 5. Multi-layer LAN hierarchy with local intelligence. 6. Multi-peer LAN flat structure with local intelligence. 7. Multi-layer and Multi-peer LAN complex. The last type would be the main developing direction for new control system.

3. Modelling of Functionality

The basic system functional concept and layers of a control system arise from a modelling of functionality. Modelling define the functional hierarchy and architecture. Hardware and software architecture is the derivation of functional architecture, among these three kinds of architectures, functional architecture is primary and basic.

Here we review some important modellings suggested by different people for different machines:

1. In 1975 M.C. Crowley-Milling divides SPS control system into three layers (1) : central, communication and control. The main contributions of SPS are: "controlled explosive" concept (2) in central layer; using network technique in communication layer; distributed process and NODAL multi-computer command and response interpreter in the control layer.
2. In 1983 M.C.Crowley-Milling further extends his SPS philosophy to LEP (3), introducing a three-level hierarchical architecture for LEP control system, i.e. central network, multiprocessor assembly and microprocessor level. The extended points of evolution are: using a network for central layer; using the assembly of microprocessors, each of which performs a single task and this assembly replaces the mini-computer in SPS; extensive utilization of the microprocessors in local control and device controllers.
3. In 1985, M.C.Crowley-Milling gives some suggestion for the future control system (4). Main opinions are: using multiplexing method to reduce cabling; using a fully and open distributed system which can prevent and overcome any bottle-neck; design of special modules with local self-diagnostic and fault finding to reduce the interface costs; increase the central computing capability for modelling operation.
4. In 1983 F.Peter introduces a highly modularity and standardrized control system architecture of DESY, which consists of an integral central control room and several "unit control cell". Each of which has the same hardware and software structure and is an independent control system for each sub-accelerator(5).
5. In 1987 TRUIMF Kaon Facility control group divides their control system into four functional levels (6): Top function level which provides library facilities, simulation, data management and software development services; operator level which provides operator interface; data concentrator level which provide data concentration and carries out alarm or other safety functions relating to subsystem; equipment process level which interacts directly with I/O modules.

4. Hardware Architecture

Control system is an assembly of some types of logical hardware units which are defined as an assembly of hardware components having a unit logical system function. Unit logical system functions are classified to: common resources in central control room, operator interface node, process computer node, local process node, intelligent device controller and the various communication links between them, i.e. various LAN. link and highway.

Hardware architecture is the hierarchical arrangement of the logical hardware units. Usually there are two basic hardware architectures: hierarchical and flat topology. In the former hardware hierarchy is the same as functional hierarchy and has a big traffic capacity with lower transfer effectivity. In flat topo-

* On leave from IHEP, Beijing

logy all the logical hardware units are linked to a single all-encompassing highway (fully flatness) or the layers of hardware hierarchy is less than functional hierarchy (partial flatness). Flat topology is efficient in data transfer between different functional levels and has higher reliability. Since all of the intercommunication is concentrated on a single network, it may result to triffic trouble. Designer always balance the merit and demerit inside this two typical topology and frequently adopt a realistic compromise between fully hierarchy and fully flatness.

Recently new progress in internetworking technique guide people to consider to use segmentation interconnected with bridges, gateways or relays, etc. to overcome the disadvantages involved in the flat topology. This leads us to a multi-peer control system complex with both huge signal capacity and effectivity of flatness.

5. Software Architecture

Application software architecture defines the software layer in the process control software and in the general purpose application tools which is the software interface between system software and process control software. Database is an important part for application software and should have a database architectures. The effectivity, reliability and flexibility of the application software are determined by its software architecture and database architecture. Here we review some important software modelling involved in the representative accelerator control systems.

1. In 1983, M.C.Crowley-Milling suggested to decompose the LEP database to several parts (3), let those common to all equipment stay in the process computer, those partcular to equipment are in its microprocessor memory. He proposes 5 levels of hierarchy for LEP database: i.e. database level, libaray level, console level, process level and equipment level.
2. Early in 1981 TRISTAN control group designed a compact, flat and easy-to-use application software architecture for KEK TRISTAN control system (7,8). Except few big application programs, application software only divide to two layers; interactive NODAL program in console computer, and remote (remote loading from console by NODAL multi-computer communication capability) and real time interpreter NODAL programs in process computers. Therefore operator can obtain more effectivity, reliability and flexibility in machine operation (for example, operator can check every CAMAC modules installed in any computer from any console station, any file on any computer can be accessed uniformly throughout the network within two levels of execution).
3. In 1985 G.P.Benincasa reported a new modelling for application software in new PS control system (9) . There are total 6 hierarchical layers: the first is operator modules layer (O.M) in console, the second is process modules layer (P.M), the third is composite variable modules (C.V.M), the fourth is equipment modules (E.M), the fifth is interface modules layer (I.M) in front-end minicomputers and the last layer is real-time Task (RT) in the microprocessor.
4. In 1987 J.P.Koutchoul, using the concept of separating data from process, suggested the LEP database architecture (10). According to the resource of data, LEP data are categorized to two types: interactive data and real time data. Interactive data is further classified to: equipment database, system database, accelerator database and archives database. Real-time database is further classified to: dataset, static data and transient data. Datasets are more further divided to three kinds: reference datasets, current datasets and target datasets.
5. In 1988 J.P.Koutchok reported LEP application software architecture using the "state" concept(11). Accelerator has three states: reference state, current state and target state, each state has a data structure: reference datasets, current datasets and target datasets; each data structure corresponds to one class of application software: model program, real-time program and study program. Transition between states is actuated by these application programs.Operation should be modular at the state level.

6. New Modelling for Next Generation accelerators

Large scale, extended location, extreme complexity, millions of signals and extra-high control requirements need a new control system architecture. Analysing various modellings used in existing machines, we have found that many valuable ideas applicable to next generation accelerators are scattered among various modellings: for example, "controlled explosive" concept in SPS; "unit control cell" concept in DESY; " Flatness architecture " concept in TRISTAN; "internetworking segments" concept in LEP. Currently they are scattered and can't sufficiently express out their big potential.

On the basis of the consideration of control features of next generation accelerator(12) and summary all of valuable concepts in individual modellings, we suggest a new modelling as follows:

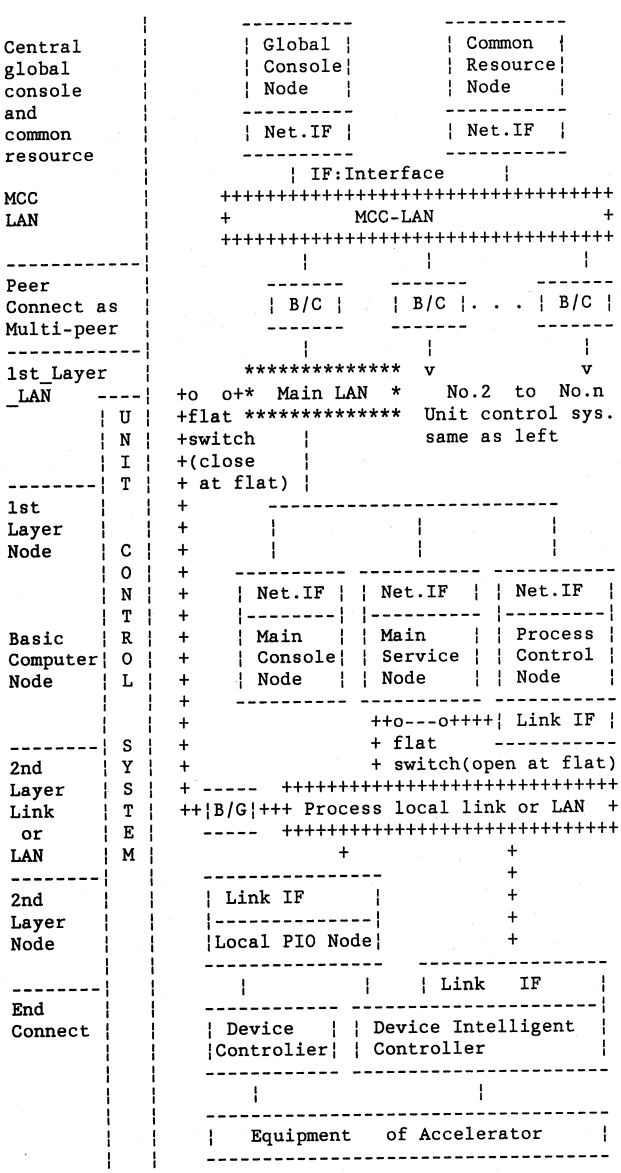
Multi-peer and multi-layer modelling

Basic considerations for this modelling are:

1. Whole control system consists of a central MCC part and several unit control systems, each for one sub-accelerator.
2. Central part consists of global consoles and common resources linked on a dedicated LAN segments (MCC-LAN). Central part is used for big accelerator complex only.
3. Each unit control system has a main LAN,(LAN 1-1), interconnection between main LAN and MCC-LAN are through internetworking connect devices, such as MAC-bridge, gateway, relay and convertor, etc. MCC-LAN and each main LAN in each unit control system (LAN 1-1, LAN 1-2,...., LAN 1-N) lie on the same hardware hierarchical layer. This LAN structure is called Multi-peer LAN complex.
4. For each unit system two kinds of hardware topology options exist: hierarchical and flat. In hierarchical topology there are two layers of communication called 1st_Layer_LAN and 2nd_Layer_Link; nodes on these two layers are called 1st_Layer_Node and 2nd_Layer_Node. In flat topology the 1st_layer_LAN and 2nd_Layer_Link are interconnected either directly or through bridges or gateways.
5. First_Layer_LAN is the main LAN for unit control system. We can use any network standards and protocol (ISO-8802, TCP/IP or any proprietary standard), we can use different standards for different unit control systems.
6. 1st_Layer_Node has three kinds of basic nodes: main console nodes for sub-accelerators; main service node for library, computing and software development computers, and process control nodes used as data concentrators and front-end computers.
7. 2nd_Layer_Link is the process I/O communication link and can be any kinds of link or LAN (such as CAMAC serial highway, SDLC link, MIL-STD 1553B, GPIB, Bitbus) used for distributing local control crates in broad areas.
8. 2nd_Layer_Node is the local process control crates with various microprocessor module boards inside and can be based on any standards(VME crate, G64, CAMAC, Intel iSBC, etc.).
9. The last layer is called End_Connect. There are two kinds of device controllers; intelligent or dumb. In the former case, device controllers can be directly connected either to 2nd_Layer_Link or to 2nd_Layer_ Node with only a message link interface. In some case there should be another local bus inside a device controller. Due to the various conditions inside this layer, End_Connect layer is defined by user.

10. In flat topology, 1st_Layer_LAN interconnects to 2nd_Layer_Link via bridges or gateways. The process control node computer must be connected either to 1st_Layer_LAN or 2nd_Layer_link; it can not be connected to both sides .

The block diagram of Modelling is shown in Fig 1.



Hierarchical CPU Modelling

For application software architecture, we should take into account the following view-points:

1. Application software is made to run by operator command and run until the process completes.
2. There are many CPUs within a control system. The relationship between application softwares and these CPUs has a pipe-line-layer charactristics.
3. In same layer there will be several CPUs running for executing an application program: as in console layer console CPU may call library CPU or Computing CPU. In local microprocessor layer, Multiprocessor involve several CPU.
4. From microscopic viewpoint the CPUs at the local microprocessor may be parallel. For simplification we can treat the multi-processor as one CPU.
5. The numbers of CPUs during execution of a dedicated application job is one of the qualified factor of effectivity for the application software architecture. The flatness topology concept used in the hardware architecture also can use for software.
6. According to the above multi-peer multi-layer modelling, the application software architecture are divided into 5 layers, each layer of which corresponds several modules and each module has a CPU inside.
   1st layer: Main console modules (MCM), Main Service Modules (MSM), Main Computing Modules (MPM).
   2nd layer: Unit console modules (UCM), Unit Service Modules (USM), Unit Computing Modules (UPM).
   3rd layer: Process Control Modules (PCM).
   4th layer: Local Microprocessor Modules (LMM).
   5th layer: Device Control Modules (DCM).
7. Each Module may have some mode. For example, there may be two operation modes in UCM; one is access LMM through PCM; other is directly access to LMM. According to the above consideration, the hierarchical CPU modelling is shown in Fig 2.

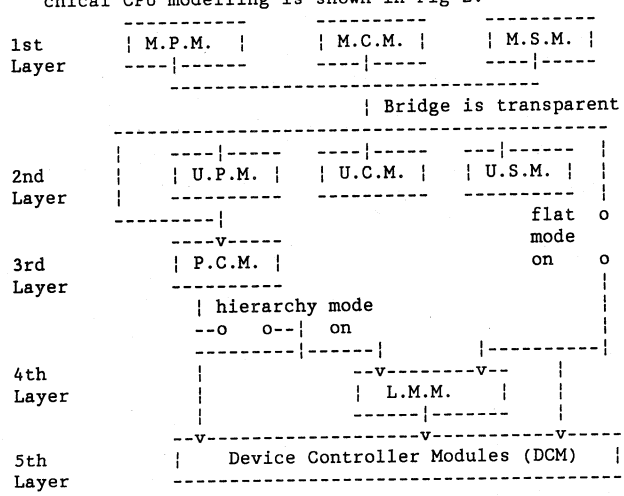


Fig.2: Hierarchical CPU Modelling

In flat topology the main console can access to the device controllers directly or through local microprocessor modules. Therefore to examine the numbers of CPU modules in a dedicated application job, people can measure the degree of flatness of the architecture.

7. Reference

1. M.C.Crowley-Milling, "The Design of the Control System for SPS", CERN 75-20, 1975
2. J.Altaber et al."Replacing Mini-Computers By Multi-Micrprocessor for the LEP Control System" IEEE, NS-30, No.4, 2287, Augest, 1983
3. M.C.Crowley Milling, "The Control System for LEP", IEEE NS-30, No.4, 2142, August, 1983
4. M.C.Crowley-Milling, "Control Problems in Very Large Accelerator", IEEE NS-32, No.5, 1874,Oct.1985
5. F.Peter, "Accelerator Controls at DESY" IEEE NS-30, No.4, 2299, August, 1983
6. W.K.Dawson et al."A Conceptual Design for the TRUMF Kaon Factory control system" TRI-97-1, 1987
7. H,Ikeda, S.I.Kurokawa et al. "Design of the Control System of TRISTAN " IEEE NS-28, No.3. 2359, 1981
8. Kurokawa S.I. et al. " The TRISTAN control system " Nuclear Inst, and Methods A247 (1986) 29-36
9. G.P.Benincasa et al. " Engineering a large Application Software Project: The Controls of the CERN PS Accelerator Complex"IEEE NS-32,No.5,2029,Oct.1985
10. J.P.Koutchouk et al." Data Management for LEP Control", CERN LEP-TH/PT-12, 1987
11. J.P.Koutchouk et al., " Architecture of the LEP Application software", CERN LEP-TH/88-28
12. S.I.Kurokawa,Liu Shi-yao," Consideration of Control System Architecture for Next-Generation Linear Colliders and Kaon Factories", in this conference