Equipment Manager of the VME Control System for the SPring-8 Storage Ring

A. Taketani*, S. Fujiwara, T. Fukui, T. Masuda,
R. Tanaka, T. Wada, W. Xu, and A. Yamashita

SPring-8, Kamigori, Ako-gun, Hyogo 678-12, Japan

We havedeveloped an "Equipment Manager" server process on a VME system to control SPring-8 storage ring equipment. The design concept of the EM is a device abstraction that is to control accelerator components with a minimum knowledge of the low-level physical device configuration. It is close to the object-oriented concept, where the accelerator equipment can be handled as objects.

1. Introduction

In the control system of the SPring-8 storage ring, workstations are used for the operator consoles and VME systems control each accelerator equipment. They are connected by an optical Ethernet and a FDDI network. The general description of the control system is presented in this conference [1].

A client-server model between high level and low level control can produce a flexible software structure. It means multiple processes can run asynchronously and be connected through a small number of Application Program Interfaces (API). We haveadopted the client-server model for access to the accelerator equipment. The equipment management server, called the "Equipment Manager" (EM), is running on the VME-based CPU board under a UNIX-like real-time operating system (HP-RT). The EM drives I/O modules and lower-level controllers through the VME bus. Access to the accelerator components outside the EM is forbidden. The client process, called the "Access Server" (AS), is running on the operator console workstations [1]. The EM and the AS are connected by the ONC/RPC protocol with TCP/IP throughout the network. The AS sends a request to the EM as a function call. Then the result of the request is the return of the value of the function to the AS. A poller for periodical data taking runs on the same CPU board as the EM [1]. It accesses the EM in a similar manner.

2. Equipment access

The hardware address is needed to access every accelerator component through the VME system. It consists of the CPU name on the network, the device name in the local CPU, thechannel number on the VME I/O module and so on. The hardware addresses strongly depend on the configuration of the physical device, but not on that of the logical equipment. We adopted the device abstraction concept, which allows us to use the logical equipment access with a minimal knowledge of the low-level physical device configuration. The logical equipment can correspond to single or multiple physical device(s). The machine operator controls the logical equipment, but not each physical device. The EM resolves commands for a logical equipment and makes linkage(s) to the actual physical device(s). This scheme allows a flexible architecture for the design of application software separated from the complex device dependent part. For example, if a device is replaced the EM must be modified. The client of the EM and higher software can be changed minimally. Fig. 1 shows the logical software position of the EM between logical and physical layers.

We have five major equipment groups: the magnet, the RF, the vacuum, the beam monitoring, and the beamline. None of the HP-RT CPU boards are assigned to multiple equipment groups. The physical device structure of a single VME system is easy to implement in the logical structure.

3. Command and response

When the EM receives a logical command which consists of a character string from the AS/Poller, it translates it into commands for the physical devices. The physical device returns a result that is a binary string. The EM converts this to a logical response and replies to the AS/Poller. The logical command and response are described in an English-like syntax S/V/O/C, where the S is a sender's identification, the V is an action, the O is a target equipment, and the C is a value to be written or the status to be read. The elements are separated by the '/'. This message syntax is close to the object-oriented concept, which is quite easy to understand for accelerator operators. The S/V/O/C command syntax is widely used in the control system of the SPring-8 storage ring.

The S includes a host name, a process identification, an application name, and the owner of the process. It is planned to be used in the EM for security and access control.

The V is a simple word for the action. Currently "put" and "get" are supported in the EM. The former sends the requested value to the equipment. The later gets the value from the equipment.

The O is the most significant term for the device abstraction concept. It indicates a unique logical equipment without complicated hardware address information. Its naming structure is determined by the configuration of the logical equipment. For example, the power supply of a bending magnet in the storage ring is abstracted as "sr_mag_ps_B". The first steering magnet in the second cell is described as "sr_mag_ps_ST_2_1".

The C is the value to send to or the reply from an equipment. Only a logical value is used instead of the digital bit string of the physical layer. For instance, the current of a power supply should be expressed as "10.5A". The EM has the ability to convert from logical to physical values and vice versa.

4. Generic EM structure

The software structure of the EM is divided into two parts. One is the common part for all the equipment operation that is written by the control group. The other is the equipment specific.

The common part handles the interface to the client process of the EM. It resolves the logical commands by reference to a configuration table. The resolved result is passed to the equipment specific part. The low level response is translated to a logical one by a table. The configuration table includes the hardware address information and constants for the conversions. This table is created by a process on the workstation and downloaded to the EM at initialization.

The equipment group must supply three functions for each equipment and describe a configuration table as follows:
(1) a translation from logical C to physical value,
(2) a VME control function which calls the device driver,
(3) a translation from physical value to logical C.
The logical command is stored with the corresponding reference to the three functions with their arguments in the configuration table.

An example of a configuration table is shown in table 1.

When a physical device or its access scheme is replaced, the corresponded functions and the table must be modified.

Table 1: An example of a configuration table. The first line gives the V and O combination for the abstracted command to a magnet power supply. This is followed by the conversions to hardware addressesetc. for three actions the C may call - swirch on, swirch off and set value to one ampere.

```
put/sr_mag_ps_ST_1_1
  on          em_mag_st_on  /dev/rio_dev0 1
              none
              em_std_ret
```

```
 off        em_mag_st_off /dev/rio_dev0 1
             none
             em_std_ret
 %1.0A   em_mag_st_current_put /dev/rio_dev0 1 1 1
             em_mag_st_calib_put   1 6.5535e+3 3.27675e+4
             em_std_ret
```

5. Research and development

While designing the EM, we have been discussing the structured design concept. A CASE tool [4] was used for the structured analysis of the common part of the EM.

The common part of the EM has been developed and tested in a HP-UX environment for rapid development. Most of the EM functions do not require the real-time extensions of UNIX. First we implemented the EM for a steering magnet power supply. A software simulator for the power supply was used instead of the real device in the early stages. The EM with the simulated power supply was a great help in the development. After the test on HP-UX, the EM was ported to HP-RT with a few simple modifications. The EM without the physical device and RPC takes about 400 μs to handle an abstracted command, as measured while using a 742rt/50MHz CPU board.

6. Status

We implemented the lower-level software [2] in the EM instead of the simulator to control the real equipment. The lower-level software include device drivers and their dedicated libraries to drive the physical devices. The RIO system, which is a kind of field bus, was adopted for magnet control [2][3]. We used two steering magnets with their power supplies for the test. This was adequate for the test because it included most of the essential pieces for equipment control by the EM. In the test arrangement the EM is controlled by the AS [1] on a workstation through RPC. The EM on the 743rt/64MHz CPU board drives a RIO master module on the VME bus. The RIO master sends commands to the RIO slaves that are attached to the power supplies. The tests were successful in allowing the remote operation of the power supplies - switching on and off, setting and reading current, reading status, and resetting errors remotely.

7. Reference

[1] R. Tanaka, et al., these Proceedings.
[2] T. Masuda, et al., these Proceedings.
[3] H. Takebe, et al., Proc. of the 4th European Particle Accelerator Conf., London, June 1994, Page 1827-1829.
[4] Teamwork by Cadre Technologies Inc.

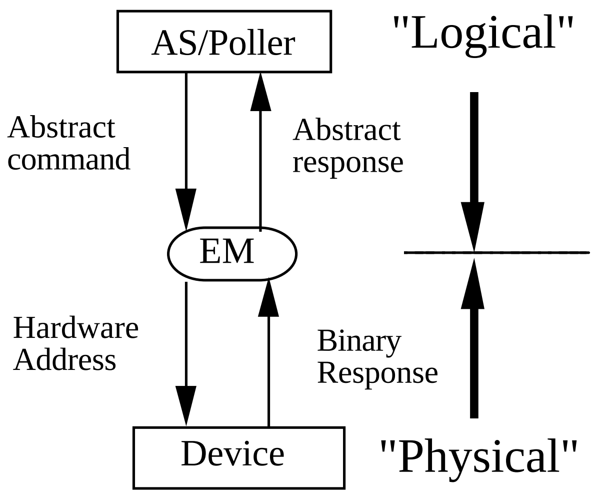

Fig. 1: The software boundary condition of the EM between the logical and physical layer. The EM exchanges the abstract message with the AS and the Poller. The physical device is controlled with the hardware address and binary value.