PC based control system using ActiveX in the KEK e^-/e^+ Linac

Isamu Abe, Hitoshi Kobayashi, and *Masahiko Tanaka
KEK, Tsukuba, Japan
*Mitsubishi Electric Co., Ltd. Tsukuba, Japan

Abstract

The KEK 2.5GeV e-/p+ Linac (electron positron linear accelerator) was boosted to 8.0 GeV from 1995 to 1998. During this upgrade in energy, the functionality of the GUI is being increased, as is the capacity of the Database. An outline of the system and its results are presented along with its GUI and reinforced database.

1 INTRODUCTION

In accelerator control systems, smaller accelerators have begun to use the most advanced PC systems available. At the same time, in the domain of accelerator control, a software paradigm-shift has occurred. This change is not attributed just to the availability of faster, more powerful hardware, or a more robust OS, but also to the development of new software and network technologies. These advancements in both hardware and software were anticipated, so along with the increase in energy from 2.5 GeV to 8 GeV of the KEK electron/positron LINAC, we have upgraded to a new PC based accelerator control system. This paper proposes "COACK" (Component-ware Oriented Accelerator Control Kernel), and discusses COACK phase 2.

2 CONVENIONAL SYSTEM

Historically, accelerator control systems have been built using object oriented programming or structured programming, implementing a procedural language. This is the case of the old Linac control system. The new system is being created not through the use of a procedural language, but by using component-ware and tools. Before this idea appeared, the development of accelerator control software encountered problems, which are discussed below.

Construction was on a large scale, resulting in high production costs. Accelerator control software tools are readily available on the market, or if not available can be created through a joint project between institutes.

Commercially available tools can carry out most of the functions required by an accelerator control system, from controlling measuring tools (LabView, VEE etc) on a small to medium sized accelerator through to operating a large scale accelerator (EPICS, Vista etc). Although the tools that are currently available offer useful and powerful functions within their limits, combining tools to obtain the benefits of both is difficult. A solution to this problem is discussed in this report.

3 NEW GENERATION OF ACCELERATOR CONTROL

In the old system, each tool had its own structure and functions. In other words, there were few commonalties between them. (shown in Fig.1) The new control system was created using flexible tools or component-ware, which use common essential objects. All required functions were performed in the old system, but only through the utilisation of multiple tools. The new system was not created with fixed tools.

When accelerator control systems were analysed, it was found that there are many common requirements between accelerators. Recently, many kinds of component-ware have become available on the market to perform these requirements. For those that are not available, a joint project to develop them is practical. In these situations, the system described in this paper is particularly suitable. The basic kernel used to deliver component-ware to this system has been named " COACK "

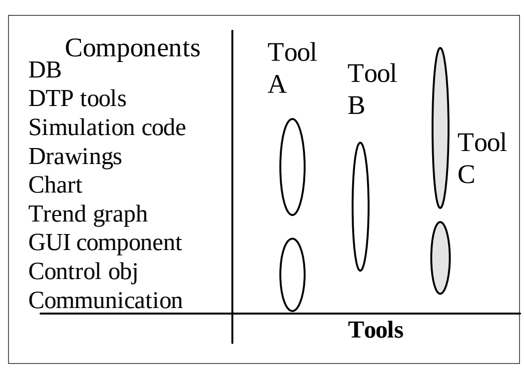

Fig.1 Components and tools

4 COMPONENT WARE

VBX and OCX are tools that were available on the marked which enabled component-ware development through the use of Visual Basic and C. In Europe, the idea of constructing and sharing of custom-made components appeared. This resulted in the ACOP project. These days, ACOP objects and OCX can be downloaded from the Internet freely, and ActiveX component-ware can be used to incorporate them into a Windows system. The accelerator control system underwent some major changes, through the introduction of these structures. The following are available in ActiveX forms:

1) Control ActiveX,
2) Code-component (exe, DLL)
3) ActiveX's document by using VB ver.5 or more.

Also, some additional benefits over Java were found. There are some drawbacks in this system: components that were available on the market were adopted, and those that were not available were created using ActiveX. These components can then become standard components. By using ActiveX, the components can be diversified, and can be stored on a server and downloaded to the users system via a network. This results in the ability to customise the accelerator control software. Recently, tools such as LabVIEW and VEE have begun to use component-ware through the introduction of ActiveX.

5 COMMON TASKS

Since task analysis makes it possible to define the ccelerator domain control system, we undertook a task analysis of the accelerator control domain. In short, it is clear that the current tools are more flexible than the former tools, if the requisite components are given freely and the essential requirements are met. The old type of accelerator control tool was not very productive, especially for the control of a large accelerator. An analysis of the tasks in the accelerator control domain was performed by classifying each layer. A three-layer structure applies as follows:

1) Human interface layer,
2) Middle process layer,
3) Device layer.

further analysis of these three layers shows the following:

a) operation system, trends, web, DTP, images:
b) database, static's process, logging, static tables,
c) device local control.

Components are available on the market for layers a) and b). The Layer c) is composed of accelerator-specific oriented components. The majority of component ware from an old system can be utilised on the new system

6 COACK

The COAK system does not necessarily require the tasks identified in the task analysis as a 'middle layer' to be located in the middle of the system. These tasks can be distributed to any other stations. Granted that the components are separated, the following two layers exist:

1) Top layer (human interface),
2) Lower layer (device layer),

Because there are distinct differences regarding to task purpose and function, the layers are clearly divided. Because the accelerator is an experimental device, constant modifications are made to meet the requirements of the user. This results in software upgrade or maintenance problems on the top layer (human interface). To counter this problem, we propose that the application server automatically upgrades each PC. The application server will also perform software maintenance on the PCs.
On the other hand, once built, changes in the lower device layer do not occur often.

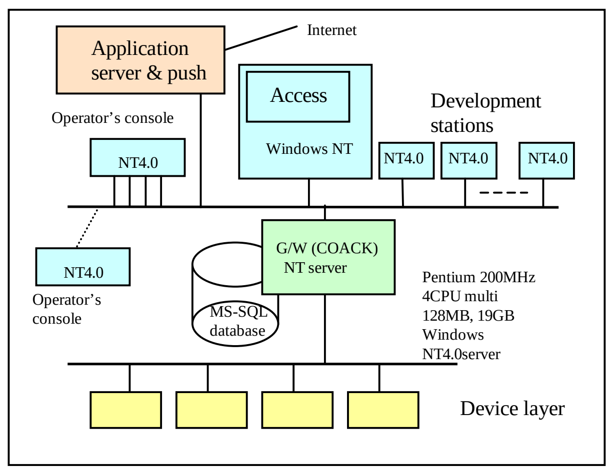

Fig.2 COACK and application server

a) Task distribution

Because tasks always ran on a specific CPU in the middle layer of the former system, it was difficult to relocate them. The COACK proposal, however, distributes the component ware from the application sever, meaning that the tasks can be carried out at any level of the system. The application server can also send and receive components from accelerators in other remote laboratories. Component-ware containers are used to accept distributed objects, and locate the tasks separately. Through division of the middle layer, CPU load is decreased, (see Fig 3) and ease of maintenance is increased..

b) Accelerator control system kernel software

The most important software-kernel is in the upper and lower layers. This is essential, regardless of any device changes or upgrades made to the system. The communication is hold N to N, which causes traffic explosion. Stations, which are managed by the gateway, are essential. Some changed information are sent to a specific station. The gateway system is located between both layers, and one of its functions includes separating traffic. Accelerator control kernel software, which can be used on any accelerator control system and does not belong to both layers, must be used. In addition, it is able to form the same component-ware as these layers. The essential components of the accelerator control kernel software are;

1) database engine,
2) application server, web server,
3) communication system.

"COACK" is necessary to manage accelerator control and is required in all accelerators. Being composed of component-ware, “COACK” has many advantages, and is very adaptable. "COACK" is currently based on machines running Windows NT; Phase 2 of the project will incorporate ActiveX.

During the first project, the component ware at the time was not useful, resulting in an successful project.

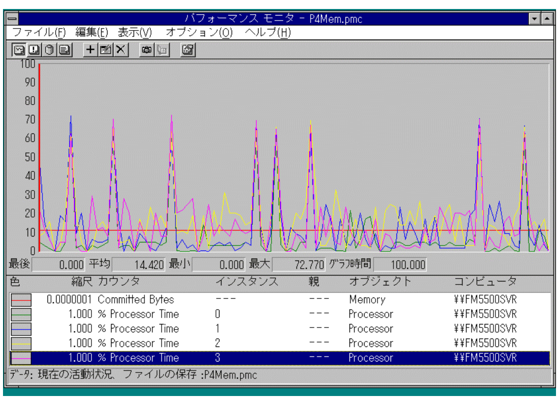

Fig.3 CPU load on COACK

7 APPLICATION SERVER

The application server in the current system plays a more important role than did the former server. The server manages parted flexible objects, and offers them to the user. The security implications of this server will require an in-depth analysis. Under these conditions, the object can be shared with many accelerators at the same time. Objects will be managed on the server, and will become a common object. Clients can download an object using a web browser or VB application. Clients can also upload these objects to the server.

8 APPLICATION PUSH (AP)

Although there are some AP technologies available on the market, we have developed a new system. The essential software works on the users station, and is delivered by an application server.
It is possible to boot a local machine from the application server using this AP system.
Through the use of this AP system, clients can perform upgrades on their local PC’s, or even perform a full installation. This saves much time and ensures that the current version of the software is installed, or at least available to every user.

SUMMARY

We have built part of the PC based accelerator control system on Windows NT. The situation has changed a lot since 1998. It has become possible to move into COM, which is able to be distributed using VB. In this situation, we demonstrated the possibilities of distributed software by reforming the three layered structure of the accelerator and by combining COACK with an ACOP like module. In a PC based control system, there was almost no accelerator control kernel or tool. It was thus expected that new control software would be created. We mentioned that it is possible to manage and distribute ActiveX objects through the use of an AP server. The new system will have applications beyond the realm of accelerator control.

REFERENCES

[1] M.Tanaka, I.Abe and Others,
“Database system for Linac operation support”
Linac conference ’97, Sendai, Japan

[2] I.Abe and M.Tanaka
"Feedback of operators' experiences to console programs in the KEK e^-/e^+ Linac"
ICALEPCS97, 1997 Beijing China

[3] M.Tanaka and I.Abe
" Database system in the Linac PC based control "
PCaPAC'99, this workshop