Web-based Beam-status Display for the KEK Injector Linac

N. Kamikubota, S. Kusano* and K. Furukawa
High Energy Accelerator Research Organization (KEK)
Mitsubishi Electric System and Service Co. Ltd.*

Abstract

A beam-status display for the KEK injector linac has been developed as a Java applet. Actually this applet is a CORBA client, which communicates with the CORBA server of the KEK linac control system to update information every ten seconds. As a result, anyone can monitor the linac beam-status in real time by using a web browser of a remote PC. Recent updates of this system are reviewed and future prospects are given.

1 INTRODUCTION

The KEK linac has provided electron/positron beams to the rings: a) 3.5-GeV positions to KEKB LER (KEK B-factory Low-energy ring), b) 8-GeV electrons to KEKB HER (High-energy ring), c) 2.5-GeV electrons to the PF ring, and d) 2.5-GeV electrons to the PF-AR ring. However, simultaneous injections to multiple rings are not possible. When the linac is in an injection process for one ring, other rings must wait for the next turns. Thus, up-to-date information about the beam-injection status is highly important for the ring operators and corresponding physics users.

One of the modern solutions to provide such status information is to use the web. We have developed a Java applet, which runs on a web browser (Internet Explorer or Netscape Communicator) to provide the beam-status of the KEK linac. Once the applet starts to run, it acts as a CORBA client. The applet communicates with the CORBA server of the linac control system every predefined interval, and then updates the beam-status information.

Since 1998, we have studied web-based real time display systems, using both Java and CORBA [5, 6]. This early work involved feasibility studies, and we did not use them for real accelerator operation. However, recent improvement of Java development kit (JDK/SDK) has enabled us to use this scheme as a realistic solution to monitor the accelerator status.

2 WEB-BASED BEAM-STATUS DISPLAY

2.1 Overview of the KEK Linac Control

The present control system of the KEK linac has been used since 1993 [1, 2, 3]. The control system comprises 5 UNIX workstations (Compaq True64 Unix), 27 VME computers (OS-9 operating system), 140 PLC (Programmable logic controller), and 11 CAMAC crates with direct network ports. A home-made RPC (remote procedure call), based on the TCP and/or UDP protocols, is used for communication between them. A simplified view of the control system is shown in Fig. 1.

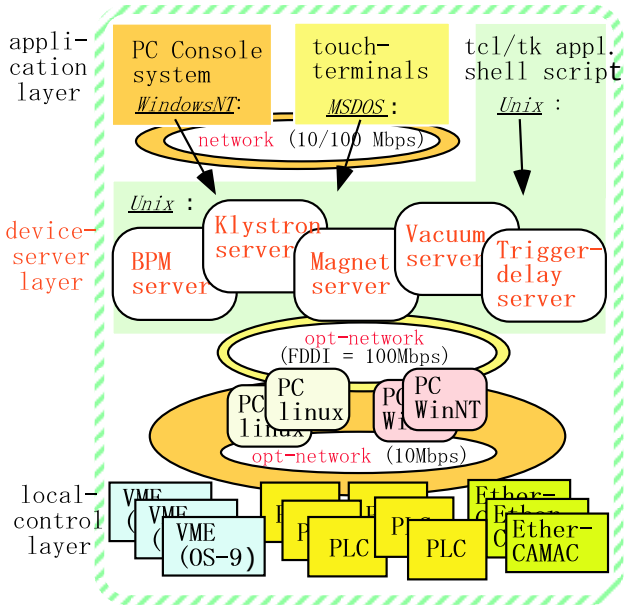

Figure 1: Simplified view of the KEK linac control system.

The number of signals in the control system is 6000 x 16bit. The recent operation time exceeds 7000 hours per year. During operation, the control servers handles 350 transactions per second in total (summer of 2001 [4]).

2.2 Web-based Real-time Display

In general, any control system has dedicated status-display tools. The MEDM[1] for the EPICS systems[2] is a typical example. It is often the case that such tools are available only in limited places (i.e. control rooms) and at limited computers. However, many potential users would want to know the overall status and histories of accelerator devices from outside of the control rooms.

Web service with CGI (Common Gateway Interface) is a popular technique to meet this request. Automatic update is possible by using the meta keyword ”refresh”. This method allows one to ”broadcast” accelerator information to anywhere where a web browser is available. The ”KEKB

[1] Motif Editor and Display Manager [7].
[2] Experimental Physics and Industrial Control System [8].

one-day Operation Summary" [9] is a typical example of this CGI technique. This example re-draws the graph every one minute. However, as the number of requests increases, CGI-based web pages produce heavy CPU loads of server machines as well as high network traffic between remote browse machines and the servers.

In 1998, a test program was developed as a Java applet (a CORBA client), which communicates with the CORBA server at a Unix workstation of the KEK linac control system. As a result, the status of the linac was shown at a web browser of a remote PC [5, 6]. This study showed that a) a 50 ms round-trip is possible between an applet and the CORBA server, and b) the CPU load of the server machine is much smaller than that of CGI-based cases. The problems were a) each time an applet takes a few minutes to start, mostly downloading and initializing CORBA class libraries, and b) the web browsers at that time[3] have early versions of Java AWT (JDK1.0.x), and accordingly have a poor ability to make graphic components.

2.3 Java and CORBA Environment

During the studies in 1998-2000, we used JDK 1.1.7 provided by Sun Microsystems for Java applet development. For the server side, we used VisiBroker 3.0 for C++ (Inprise Co.) at our Unix workstations.

During the summer of 2001, we decided to change our development environments. The JDK was upgraded to SDK after version 1.2 (so-called Java2). We chose SDK1.3 because it was already widely used. In addition, we changed the ORB (Object Request Broker - core processes of CORBA) to ORBacus 4.0.5 for C++ (OOC/Iona Co.). The main reason is that Inprise disconnected the support of VisiBroker for our platform (True64 Unix). Another important reason is that we prefer free software. [4] The changes are summarized in Table 1.

Table 1: Changes of the Java/CORBA environment

	after 2001	1998-2000
Java (GUI) (CORBA)	SDK1.3.x (Java2) Swing included in SDK	JDK1.1.7 AWT (JDK1.0.x) download necessary
CORBA for C++	ORBacus 4.0.5 OOC/Iona Co. Free for academic	VisiBroker v3.0 Inprise Co. True64 support stop
Aim	realistic demonstration	feasibility study

The change to SDK1.3 (Java2) is significantly important, because the SDK includes class libraries of basic CORBA communication. Thus, the startup of an applet becomes much faster than before. With the present environment, the amount of download files is 4 MB. [5]

[3] We used Netscape 3.x and 4.0.x .

[4] ORBacus is free for academic purposes.

[5] The ORBacus classes are of 3 MB, and the applet itself is 1 MB.

2.4 KEK-Linac Beam Status Display

We have developed a Java applet to show the beam-status of the KEK linac with the new environment. A CORBA server was also developed using the existing C-language libraries and beam-status databases of the KEK linac control system. The overall relationships are shown in Fig. 2.

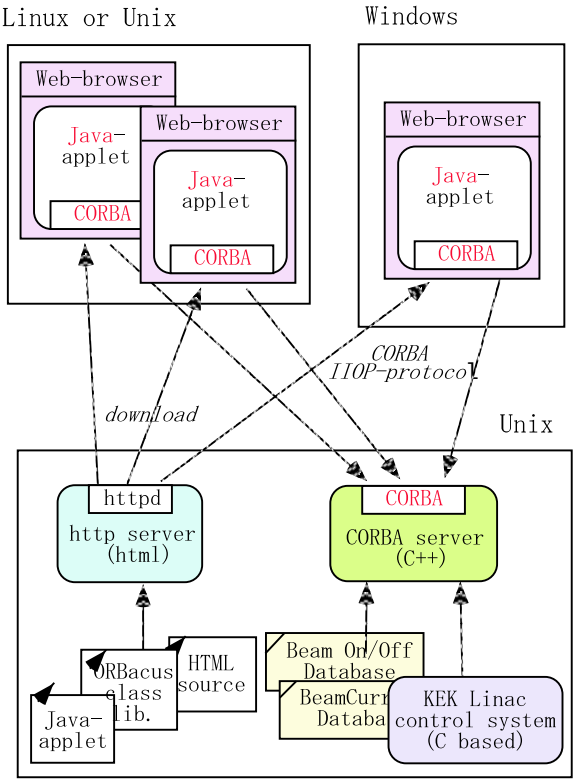

Figure 2: Relationship overview.

An example screen-shot of the beam-status display is shown in Fig. 3. The screen includes information about a) the present beam-status (beam-injection mode, beam switch [On/Off], and beam current), b) 2-hour history of the beam current, and c) 2-hour history of the beam-injection modes. The screen updates information at every 10-second interval by communicating with the CORBA server. At the upper-left location, there is a menu used to select the time-window (2-hour or 24-hour). When the time-window is changed to 24-hour, the applet calls the CORBA server to get recent 24-hour histories.

The present beam-status display is available at any platform where Java 1.3 (or later) is installed. We assume that although the major users use a browser at a Windows PC, other platforms (Linux, BSD, Mac OS-X, Solaris, HP-UX 11.0, and so on) should also be available. At the KEKB control room, a beam-status display has always running at a Windows PC since April, 2002. We have found that the display is sufficiently stable. [6] In addition, the service is already open for anybody at KEK.

[6] The only problem is that the browser always clashes when somebody clicks the back button.

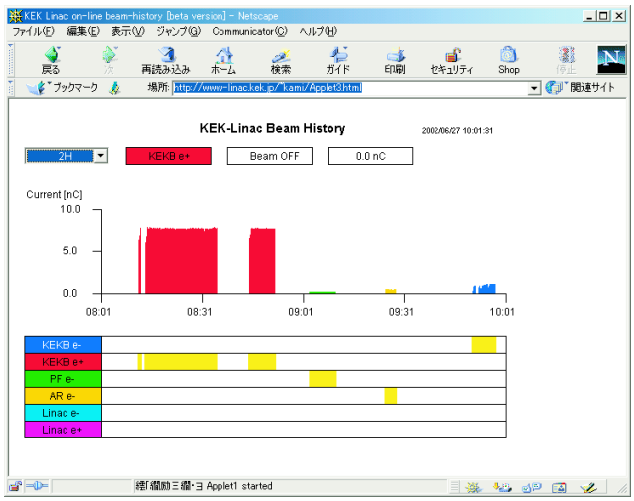

Figure 3: Screen-shot of beam-status display.

3 DISCUSSION

3.1 *Java plug-in*

When a Windows user connects to our beam-status web page for the first time, the user is requested to install ”Java plug-in 1.3” (Fig. 4). This is because the Java 1.3 environment is not included in native Windows. Although the installation is a simple procedure, it takes roughly 10 minutes to download necessary files from Sun Microsystems. In feasibility studies conducted during 1998-2000, we had to download the CORBA classes each time an applet started to run. However, with the new environment, the download of the basic CORBA classes is done only once.

For other platforms, Java 1.3 installation requires manual actions. For example, for Linux, one has to find appropriate rpm patches, and has to install them manually.

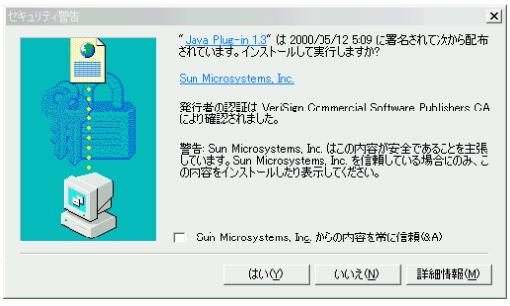

Figure 4: Wizard to assist plug-in download.

3.2 *Security*

CORBA communication requires a TCP/IP port, which is defined in html text. When a firewall exists between a remote PC and the server machine, we must make a hole that corresponds to the port number of the communication. However, a remote access can execute only pre-defined CORBA methods at the server-side. Unless we have no destructive methods, we have less risk of hacker’s attacks.

3.3 *Future Directions*

The present work can be applied to various fields which aim for web-based status display in real time. If the existing CGI-based web pages with ”refresh” were replaced by the present Java/CORBA pages, the total CPU load and network traffic would decrease considerably. For accelerator institutes, the present technique can be used as a CATV-like system. By preparing the servers of the accelerator status, any PC in the institute can be a browse terminal.

One of the problems in this scope is that preparing applets and CORBA servers is not easy. We are now considering a template system for the quick development of more applets and servers. The template applet changes its look and feel depending on the configuration file. The template CORBA server can provide the status of different accelerator devices by replacing pre-defined user functions. Studies are now in progress at the KEK linac.

It is interesting that the present work uses the resources (libraries and computers) of an existing control system. The introduction of CORBA enables us to use modern information technology (Java) with a legacy system.

4 ACKNOWLEDGMENT

We thank Drs. Gennadiy Obukhov and Matthias Clausen, DESY. Discussions with them at the beginning of this work helped us very much. The authors acknowledge Prof. Atsushi Enomoto for kindly supervising our work. We also thank the KEK linac operators for cooperative and successive work to improve our beam-status display.

5 REFERENCES

[1] N.Kamikubota, K.Furukawa, K.Nakahara and I.Abe, Nucl. Instr. Meth. A352(1994)131

[2] N.Kamikubota, K.Furukawa, K.Nakahara, I.Abe and A.Shirakawa, Proc. of the ICALEPCS’95, Chicago, October 1995, FERMILAB Conf-96/069 p.1052

[3] N.Kamikubota, K.Furukawa, S.Kusano and T.Obata, Proc. of the ICALEPCS 2001, San Jose, CA, Nov.2001, KEK-Preprint 2001-155, in press

[4] N.Kamikubota, K.Furukawa, T.Suwada and T.Urano, Proc. APAC’01, Beijing, Sep.2001; KEK-Preprint 2001-124

[5] S.Kusano, N.Kamikubota and K.Furukawa, Proc. of the PCaPAC’99, Tsukuba, Jan.1999, KEK-Proceedings 98-14

[6] S.Kusano, N.Kamikubota and K.Furukawa, Proc. of the ICALEPCS’99, Trieste, October 1999, p.535-537

[7] http://www.aps.anl.gov/epics/extensions/medm/index.php

[8] http://www.aps.anl.gov/epics/index.php

[9] http://www-linac.kek.jp/kekb/snapshop/