Real-time Display of Accelerator Status Using JAVA and CORBA

Shiro Kusano*, Norihiko Kamikubota and Kazuro Furukawa
High Energy Accelerator Research Organization (KEK)
1-1 Oho, Tsukuba, Ibaraki, 305-0801, Japan
Mitsubishi Electric System and Service Co. Ltd. *
Umezono 2-8-8, Tsukuba, Ibaraki, 305-0045, Japan

Abstract

Recently, the introduction of object-oriented technologies into accelerator controls has been understood as a promising future direction. Among object-oriented technologies, Java and CORBA have been widely accepted in all industrial fields. We, the KEK electron/positron injector-linac team members, have investigated the possibility to introduce such technologies for the real-time display of the accelerator status. A test program was developed with Java, 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 KEK linac was shown at a web-browser of a remote PC. Experiences and discussions are given in this article.

1 INTRODUCTION

Recently, the object-oriented technologies for distributed computer systems, such as Java[1] and CORBA[2], have been paid special attention in the field of accelerator controls as well as in all industrial fields. Java is a platform-independent language with the capability to create an interactive GUI (Graphical User Interface) at a web-browser. Thus, Java is a suitable language to develop an application program with high source portability. CORBA (Common Object Request Broker Architecture) is expected to become the standard communication protocol between distributed computers over networks. The specifications of CORBA are discussed by the consortium called OMG (Object Management Group)[2], which includes over 800 members. The use of CORBA enables smooth communication between different languages (C, C++, Java, etc.) and different operating systems.

The control system for the KEK e-/e+ injector-linac comprises the UNIX-based workstations, VME computers with the OS-9 operating system, and PLC stations [3, 4]. The TCP and/or UDP protocols (socket functions) are used for communication between them. Applications for status display have been developed with the man-machine interfaces at the KEK linac. Two man-machine interfaces are available: 1) the operator's console system with PCs and Windows NT[5, 6], and 2) the touch-terminal system with DOS-based PCs [7]. The operator's console system uses the Winsock functions to communicate with the servers at Unix workstations. Since the Winsock functions are different from the standard socket functions, the communication layers for the operator's console system should be maintained independently. In the touch-terminal system, the TCP/IP library for DOS-based systems has been used [8]. By using this library, the sources for the communication layers are the same as those for Unix workstations. However, software development with a DOS-based the PC is not easy because of the 640 kB memory-limit.

We have studied the feasibility of applications which use both Java and CORBA. If an application is written by Java and communicates with the CORBA protocol, it is expected to work at any platform, including both PCs and Unix workstations, with the same sources. Moreover, it can communicate with any control systems unless an appropriate CORBA server is ready. We developed an application with Java which communicates with the KEK linac with the CORBA protocol. The aim of the application is to provide a real-time display of the accelerator status at the KEK linac. Experiences and discussions are given in this article.

2 DEVELOPMENT OF A REAL-TIME DISPLAY OF ACCELERATOR STATUS

2.1 Overview

We have developed a real-time display of the high-voltage status of klystrons at the KEK linac. A commercial CORBA product [9] was installed and has been used in our tests. We selected product [9] among various possibilities, since the popular web-browser (the Netscape Communicator) supports the communication classes for Java of this product.

An overview of our environment is shown in Figure 1. An application, which acts as a CORBA client, was developed as a Java applet. The CORBA server was written in C++, which uses the C-based libraries of the KEK linac control system in order to obtain a high-voltage status of the klystrons. We inspected two points: (1) the source portability; the same Java applet is available at any of the different platforms, and (2) the communication availability with the CORBA protocol between an applet (Java) and the server (C++).

The communication interface must be described by an IDL file.[1] The use of IDL makes the development of the communication layer very simple, since the sources of the communication layer are automatically generated from the IDL file.

[1]IDL - Interface Definition Language. IDL is a platform-independent language to describe communication interfaces. The IDL is one of the key features of the CORBA specification.

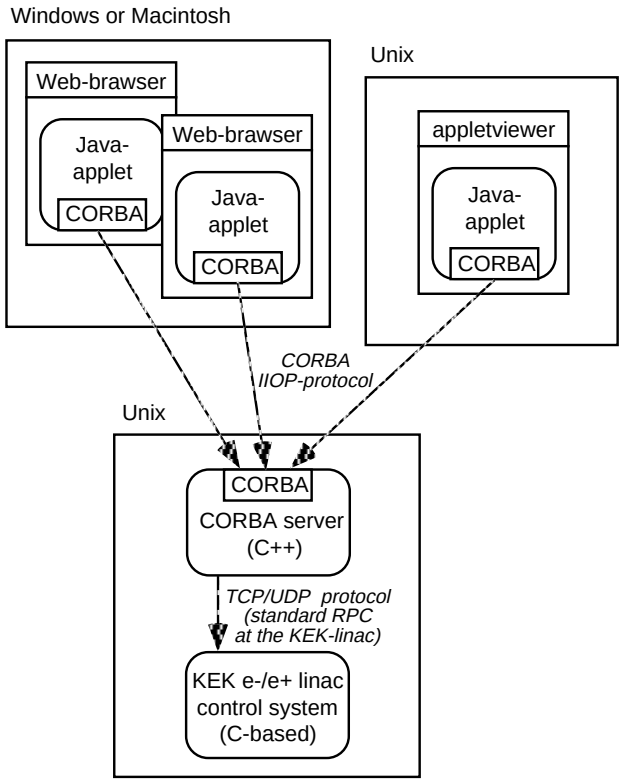

Figure 1: Overview of our test environment.

2.2 Implementation

IDL file The IDL file in the present test (kly.idl) is given. The sources for the communication layer (Java codes for the client, and C++ codes for the server) are generated automatically from this IDL file.

```
//kly.idl
typedef string Names[58];
typedef long Data[2784];
interface Kly {
    void klyhex(in Name names,
      out Data data);
};
```

Implementation of the klystron server The KEK linac control system includes the C-function, plc_localhex(), to provide information about the klystron status. This function is re-defined as a C++ function as:

```
kly_impl::klyhex(const Names names,
Data data){
  int rtn;
  char name[6];

  for(i=0;i<58;i++) {
    strcpy(name, names[i]);
    //klystron service in KEK-Linac control
    rtn = plc_localhex(name,
                (unsigned int *)data+(i*48));
    }

}
```

The server uses the above function as:

```
main() {
   Kly_var klyImpl = new kly_impl();
   CORBA_String_var ior =
        orb->object_to_string(klyImpl);
}
```

Implementation of a Java client The essential part of the Java applet is shown below:

```
org.omg.CORBA.Object orbObject =
        orb.string_to_object(url);
kly = KlyHelper.narrow(orbObject);
DataHolder data = new  DataHolder();
kly.klyhex(names, data);
```

Implementation of a html description A html description is nesessary to download the Java-applet from the web-server

```
//kly.html
<applet code=testApplet.class>
<param name=ior value=IOR:012020200.......>
</applet>
```

2.3 Results

When the Java applet starts with a web-browser (or by a tool “appletviewer”), it connects to a server by using the CORBA protocol. The current high-voltage status of the klystrons at the KEK linac is shown (Figure 2). Since the screen is refreshed every 1 second, this applet can serve as a simple klystron alarm, which can be used anywhere the control network is available.

The source portability of the Java applet has been inspected at various platforms. The summary in Table 1 demonstrates the high source portability of Java language.

Table 1: Availability at various platforms

Platform	Netscape4.0.x	Netscape4.5
Digital Unix	OK	OK
WindowsNT/95	OK	none
MacOS8	OK	none
Linux	OK	OK

3 DISCUSSION

3.1 Communication Throughput

A simple Java applet was developed in order to measure the communication throughput between a client (an applet)

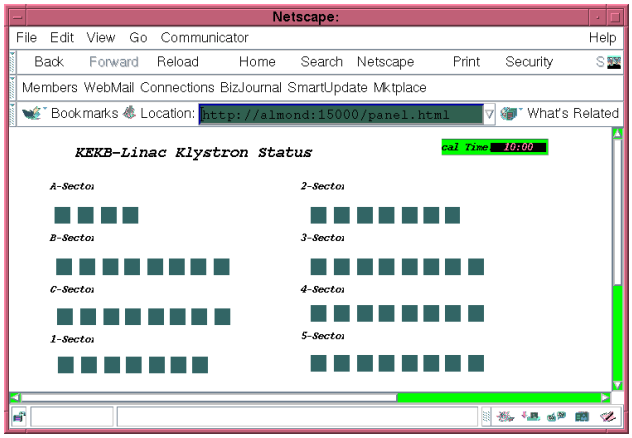

Figure 2: View of the Java applet with the real-time display of the klystron status.

and a CORBA server. The sizes of transfered data were 290 byte for sending to a server, and 2784 byte for return. The observed round-trip times were around 50 ms for two different cases: (a) the server and the client run at a same Unix workstation, and (b) the client runs at a PC (Linux, Pentium 133MHz) while the server at an Unix. This fact suggests the possibility that a Java applet, which runs at any remote PC, can communicate with a server at a refresh rate of 10 Hz (or more).

3.2 Experienced Problems

During the developments, we experienced inconsistencies between different versions of Java and web-browsers. The CORBA product in the current tests [9] assumes to use JDK 1.1.6, while the GUI functions in the web-browsers are based on JDK 1.0.x. We decided to use old classes in order to avoid this confusion.

The control system for the KEK linac provides C-based functions. However, the present CORBA product assumes to use C++. This incompatibility causes errors when we link the linac functions to a CORBA server. This problem was solved by adding the definitions of C-functions in header files.

3.3 Problems in Windows NT/95

At a PC with Windows NT/95/98, a client (a Java applet) can communicate with a CORBA server. However, Microsoft has proposed the DCOM/Active-X as default communication protocols instead of the CORBA protocol. At present, there is no appropriate way to communicate between the DCOM/Active-X and the CORBA protocols.

4 CONCLUSION

We have developed a Java applet, which communicates with a CORBA server at the control system of the KEK linac, in order to evaluate the availabilities of Java and CORBA. We confirmed that the Java applet is available at four different platforms, and that it can communicate with a CORBA server which resides on the KEK linac. The observed that throughput suggests that a real-time display at a rate of 10 Hz is possible.

We will proceed with investigations on there subjects; especially the availability of free CORBA products, and Java connectivity to a relational database (JDBC).

5 ACKNOWLEDGMENTS

We thank Dr. Gennadiy Obukhov (DESY) for various discussions. In fact, the sources of his work with VisiBroker (JOI) helped us very much, especially at the beginning of this work.

6 REFERENCES

[1] http://java.sun.com

[2] http://www.corba.org or http://www.omg.org

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

[4] N. Kamikubota, K. Furukawa, K. Nakahara, I. Abe and A. Shirakawa, Proc. Int'l Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'95), Chicago, October 1995, FERMILAB Conf-96/069 p.1052

[5] K. Nakahara, I. Abe, N. Kamikubota and K. Furukawa, Nucl. Instr. Meth. A293(1990)446

[6] I. Abe, “LINAC PC based control system using ActiveX”, this workshop

[7] N. Kamikubota, H. Akimoto and K. Furukawa, “PC as a touch-terminal controller”,this workshop

[8] PC/TCP Release 2.2 (Nov.1992), ftp Software, Inc., MA U.S.A.

[9] VisiBroker for C++ V3.0, and VisiBroker for Java V3.2, Inprise Corporation, CA U.S.A.