A JAVA-BASED EPICS ARCHIVE VIEWER WITH SOAP INTERFACE FOR DATA RETRIEVAL

K. Furukawa *, M. Satoh
High Energy Accelerator Research Organization (KEK), Tsukuba, Ibaraki, 305-0801, Japan
I. Mejuev
PFU Ltd., Kawasaki, Kanagawa, 212-8563, Japan
K. Nakao
Institute of Quantum Science, Nihon University, Funabashi, Chiba, 274-8501, Japan

Abstract

In advanced physics experiment systems like a large particle accelerators, it is important that physicists can easily analyze device behavior based on archived data to achieve an optimal result of the experiment. Standard network and software technologies such as Web, XML, Java, etc would be employed to enable a wider range of environment for analysis tools. To this end a Java tool for displaying EPICS ChannelArchiver has been developed for the KEKB injector linac. The EPICS Archive Viewer is implemented as a pure Java code, which utilizes a high-quality commercial software package for charting (JClass Chart). The viewer retrieves archive data by interfacing with an extensible set of "archive data providers". Current implementation includes providers for SOAP and CGI protocols and also allows support for custom, "in-house" archive data providers (e.g. a provider that relies on CORBA/IIOP). The server-side of SOAP archive data provider is represented by a backend service object deployed within the AXIS/Tomcat container. The implementation of all viewer modules is highly portable and offers a potential for data sharing among organizations via the emerging Web Services standards.

INTRODUCTION

The KEKB linac control system allows multi-user data access via home-grown remote procedure calls (RPC) over TCP or UDP connections [1]. There are several solutions deployed for linac data sampling such as custom archivers for subsystems that store data into compressed binary files. On-change archiving is managed by system surveillance processes that record the changes in the device states into text log-files available through NFS. The device history viewers based on commercial graphic package have been developed for existing archive formats [2]. Although those software tools and data formats are optimized for corresponding purposes and users, it became preferable to integrate them into more manageable forms.

The introduction of EPICS archives for data storage at KEKB linac would enables a standard and unified archive format, reuse of existing EPICS archive engine and viewers, and it offers a potential for data sharing with other organizations that also utilize the EPICS ChannelArchiver [3].

The EPICS ChannelArchiver includes a number of tools for data storage, manipulation and retrieval. The components essential for our development are the C++ archive I/O library (libIO), the archive engine and a number of end-user tools for data retrieval and presentation.

Introduction of the EPICS ChannelArchiver compatible with the KEKB linac controls requires a archive engine that retrieves information via linac control servers as well as the development of customizable viewers for retrieving and displaying archived data. This paper focuses on the latter issue and describes an EPICS archive viewer that has been implemented in Java.

As for the EPICS-compliant archive engine for the KEKB linac, it appeared to be preferable to implement an EPICS "device support" layer for I/O controllers (IOCs) wrapping linac servers and to reuse the existing ChannelArchiver engine for the EPICS channel-access protocol. The extended IOC features available in the recent EPICS release 3.14 simplified this implementation.

EPICS GATEWAY AND ARCHIVER

The EPICS archiving architecture is employed to enhance the linac archive system. The released ChannelArchiver was employed without any modifications[1]. While the linac control system is not based on EPICS, the implementation of archiver with EPICS may advance the sharing of resources between accelerator facilities, and the gateway software enabled such an integration.

The channel access gateway for the linac controls was reimplemented to provide more versatile information to the channel access archiver. Previously the gateway was implemented with the portable channel access server of EPICS 3.12[4]. The improved gateway was newly designed and developed with the IOC code of EPICS 3.14. It is expected to be easier to maintain the gateway with the flexible design of device-support C codes and configuration database.

* <kazuro.furukawa@kek.jp>

[1] The ChannelArchiver version 1.9.1 from LANL was used, which was the latest when the tool was developed.

EPICS ARCHIVE VIEWER

The archive viewer front-end is implemented as a pure Java application that can be also executed as an applet running in a web-browser equipped with a Java plug-in. An outline of the Archive Viewer modules is represented in Fig. 1.

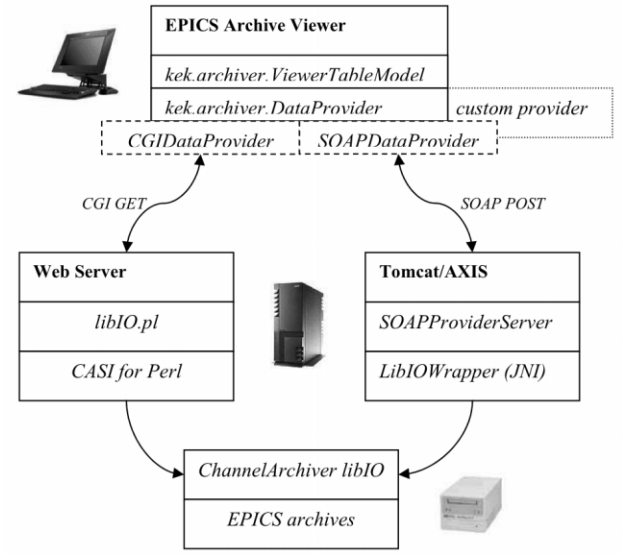

Figure 1: EPICS archive viewer outline.

The viewer is implemented with Java 1.3 SDK using the lightweight Swing user-interface (UI) library with an addition of a commercial charting component JClass JChart [5]. The core class in the viewer implementation is ViewerTableModel, which corresponds to a model component in terms of a standard model-delegate design pattern. A ViewerTableModel retrieves its data from an abstract data provider which implements a predefined interface (DataProvider). Essentially, a ViewerTableModel instance needs no knowledge of how the provider is implemented. The model-delegate architecture would also allow future integration of live updates into the viewer for JClass Chart.

Current implementation includes the following "archive data providers":

- kek.archiver.CGIDataProvider – it retrieves archive data by dispatching a request to a CGI/Perl script deployed on a web server. The server-side script uses the ChannelArchive scripting interface (CASI) for Perl to call libIO functions.
- kek.archiver.axis.SOAPDataProvider – based on Apache AXIS [6], this provider retrieves archive data by interfacing via SOAP [7] with a server-side backend object which is running within AXIS/Tomcat container. The backend object processes incoming requests by utilizing a Java Native Interface (JNI) wrapper for libIO (kek.archiver.axis.LibIOWrapper class).
- custom providers – archive viewer allows specifying a data provider class name and its location (an URL) via the command line options or via applet parameters. A custom provider is instantiated via the Java reflection API. The provider is only required to implement the DataProvider interface and may use alternative methods for data retrieval (e.g. CORBA/IIOP or a direct access to local files).

Both CGIDataProvider and SOAPDataProvider use the HTTP protocol to communicate with the corresponding servers. Amount of data in a SOAP transfer is larger comparing to CGI one since the former is represented in XML that is more portable. Simplified wire dumps for SOAP request and reply are presented below:

While not being optimal for network traffic, the use of SOAP and Web services infrastructure provides several conceptual benefits for data sharing within our community:

- Web services expose a metadata definition (WSDL) that describes custom data types used by provider.
- Since SOAP is HTTP-based, it can operate via firewalls without utilizing specialized gateways as in the case with CORBA/IIOP.
- SOAP allows interoperability with Microsoft .NET platform (C# and Visual Basic .NET clients).
- A growing number of tools and technologies is available for building Web services [7].

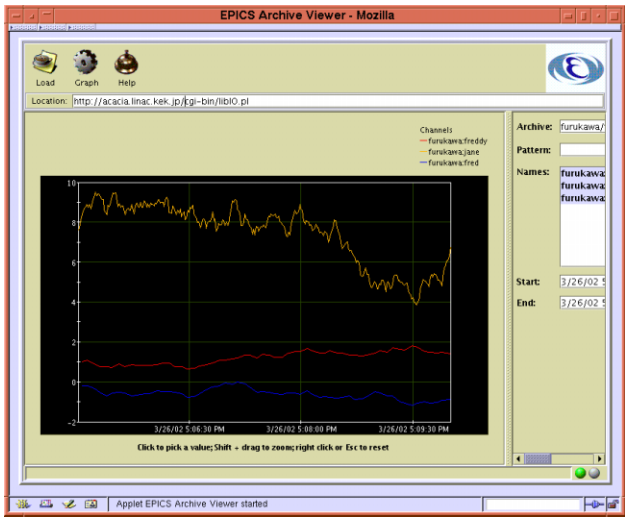

Figure 2: User interface panel of EPICS archive viewer.

The UI of the EPICS archive viewer provides the end users with the possibility to retrieve a list of channels available in a given archive, and filtered data by channel names and by time range. It is also possible to modify the various properties of the charting component (chart type, colors, line styles, labels, axes, etc.). In the graph chart the viewer allows zooming of a data range by dragging the mouse over the plot area and implements a "pick" interaction to display data for each point. Figure 2 and 3 show an EPICS archive

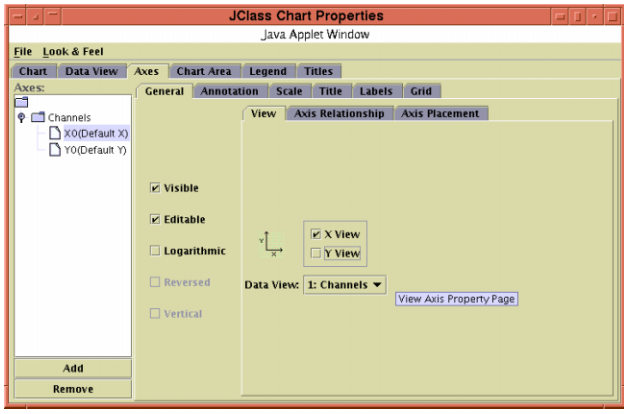

Figure 3: User interface panel for chart attribute customizations.

viewer which is executed on a web browser, and a chart customization panel.

```
POST /axis/services/EPICSArchiveService HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1beta
Host: acacia.linac.kek.jp
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 904

<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soapenv:Body>
  <ns1:getData soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns1="http://www-linac.kek.jp/SOAPProviderServer/">
   <arg0 xsi:type="xsd:string">furukawa/furukawa test</arg0>
   <arg1 xsi:type="xsd:string"></arg1>
   <arg2 xsi:type="xsd:dateTime">2002-03-26T08:05:00.000Z</arg2>
   <arg3 xsi:type="xsd:dateTime">2002-03-26T08:10:00.000Z</arg3>
   <arg4 href="#id0"/>
  </ns1:getData>
  <multiRef id="id0" soapenc:root="0"
      soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
      xsi:type="ns2:Vector"
      xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:ns2="http://xml.apache.org/xml-soap"/>
 </soapenv:Body>
</soapenv:Envelope>
```

Figure 4: An example of a SOAP request.

CONCLUSIONS

This paper described a development of an EPICS Archive viewer, implemented with Java SDK 1.3. In comparison with EPICS ChanelArchiver CGIExport tool, the following advantages of Java-based implementation can be pointed out:

- The Java viewer provides the possibilities for a direct user interaction (such as zoom and pick using a mouse) and for customizing plot preferences without reloading the data.
- Java implementation can employ wider selection of UI components comparing to HTML forms in CGIExport (such as split panes, custom dialogs, etc.)
- The EPICS Archive Viewer allows extensibility regarding support for custom “data providers”; the custom providers can be integrated without any modifications of underlying viewer code.

With these developments the archiving and viewer system would ease the future enhancement to the archivers, and we hope that it will improve the quality of the linac operation.

REFERENCES

[1] K. Furukawa *et al.*, “Accelerator Controls in KEKB Linac Commissioning”, *Proc. of ICALEPCS99*, Trieste, Italy, 1999, p.98.
N. Kamikubota *et al.*, “New Control System with VME and Workstations for the KEK e-/e+ Linac”, *Proc. of ICALEPCS93*, Nucl. Instrum. Meth. **A352**, 1994, p.131.

[2] N. Kamikubota and K. Furukawa, “Tool for Device Histories at the KEK Linac”, *Proc. of LINAC96*, Geneva, Switzerland, 1996, p.800.

[3] K. Kasemir *et al.*, “Overview of the Experimental Physics and Industrial Control System (EPICS) Channel Archiver”, *Proc. of ICALEPCS2001*, San Jose, USA., 2001, p.526.

[4] M. Kaji and K. Furukawa, “Operation of KEKB Linac and Ring with EPICS”, *Proc. of 21st Linear Accelerator Meeting*, Tokyo, Japan, 1996, p.207.

[5] Sitraka JClass Chart (Quest Software), http://www.sitraka.com/jclass/chart.shtml

[6] Apache AXIS, http://ws.apache.org/axis/

[7] SoapWare.Org, http://www.soapware.org/