Web Utilization in Controls at Fermilab

K. Cahill B. Hendricks T. Zingelman
Fermi National Accelerator Laboratory
PO Box 500, Batavia, IL 60510, USA

Abstract

The World Wide Web is utilized at Fermilab in support of controls development, maintenance and accessibility. This paper will examine the web's utility in support of application program development, application specific help, data acquisition error and statistics reporting, GIF image production including application batch scripting, and a Java front end interface to X-Window console applications.

1 Introduction

The use of the World Wide Web, WWW, and its associated tools continuously increases at Fermi National Accelerator Laboratory. The development and operation of the control system relies and depends upon the information presented by Internet browsers to developers and operators of the accelerator complex.

This paper will focus upon the utilization of the HyperText Markup Language, HTML, and the language, Java, in their support of the development, maintenance, and operation of the control system.

2 HTML

HTML is the documentation standard within the Accelerator Controls Department at Fermilab.

2.1 Supporting development

Linked HTML pages are the first and only place of reference for the application program developer[1]. The ACNET console library, CLIB, consists of more than one thousand entry points. Each of these is described along with links to related functions and include files.

Users searching for hardware components to implement a control' function look to the hardware capability specifications, as well as database and driver specifications, which are found within linked HTML pages[2].

2.2 Supporting maintenance

Several central processes participate in collecting error, traffic and status information. The user task interfaces of the data acquisition and setting services forward reports containing setting, received error, and traffic usage to central processes on full queues, every few minutes and on task exit. These resulting reports encompass more than 400 process generated HTML documents[3]:

- all data acquisition errors
  - filtered, sorted, and annotated
  - organized for dispatch by system experts
- several up to the moment status reports
  - many < 10 minutes old
  - by node, crate, slot, and trouble type
- traffic reports of consoles and front ends
- save/restore status reports
- summary of all device settings

These files are often hyperlinked to each other, are the day 's first and last view of control system health for management, and generally maintain information for a one week period. One of these documents describes all bypassed alarms in the control system. One describes the range of settings seen by a device on a particular day. Several processes contribute to these reports. The process that maintains HTML pages of all settings within the control system also forwards those settings to a settings' datalogger.

2.3 Supporting operations

All applications support online help[4]. The keycap on most consoles F15 key has been replaced by a help key. The console environment is linked to web browsers to warp the browser to the appropriate help page. The help pages of applications are HTML files. A help editor edits these HTML files in place. An application help may include embedded HTML links to other documentation.

Special production of .gif image files is tied to the web. Users may make .gif copies of application windows and email themselves a location to paste into their browser link window. Users may also insert jobs into scheduled batch queues that will update web pages with images by:

- running a console application script on an unused console slot
- creating and copying a .gif image file
- linking the image to a web page

3 Java

This control system awaits the appearance of the tools, platforms, and paradigms that will spark the project leading to the next generation application programming environment.

3.1 The appeal

The Fermilab Accelerator Control System console environment is X-Window based with a large central library generally supporting the C language programmer. Commercial relational databases, other languages such as C++, and web based tools have some impact, but the next

generation of operator controls is imagined to truly exploit software development tools distinctly more sophisticated and productive rendering much of our graphical user interface library built upon X-Windows as obsolete.

Proprietary software architectures are disdained. Software tools with the reach and impact of X-Windows are anticipated. As Java emerges, the language and development environment are examined with the following questions in mind:

- can it become X-Windows++?
- is the language simplicity a plus?
- is the development environment futuristic?
- will Java beans enable commodity computing?
- will most software become web-centric?
- is it fun, exciting, and rewarding for the staff?

3.2 The console environment to emulate

As an introduction to the first Java applet, it is appropriate to describe Fermilab Sconsole environment. It is X-Window based, supporting:

- 6 concurrent user written applications
- 17 windows, more or less
- two X-Window managers, alphanumeric and graphical
- hundreds of users, each requiring:
  - X-Window software
  - an account
  - some knowledge to start a console
  - fair amount of RAM

3.3 Java console project goals

The JavaConsole project aims to support concurrent applications, offer real-time updates of alphanumeric windows and snapshot updates of graphical windows. Additional console security is provided through a WebUser console class implementation supporting read only performance, no access to shared sensitive data, and offering no hardcopy resources. The result provides Fermilab console accessibility to anyone with a web browser.

3.4 Java console implementation

The development of the Java applet was coupled with the development of a threaded server using a byte protocol connecting the alphanumeric X-Window manager and the applet. Simulated Xlib calls result in socket writes, faked X events are generated by socket reads, keystrokes are echoed, and the resulting applet is a browser capable front end to a Fermilab console. Some performance enhancements were necessary, such as giving the applet cursor control, as round trip times to handle mouse motion events introduced sluggishness.

This applet allows any user with access to a browser access to all of the hundreds of console applications. Graphical windows and applications are supported in a lesser manner. Since many graphical applications perform thousands of X operations per second, no attempt is made at this time to emulate the X behavior of the graphics manager in Java. Instead, screen snapshots, initially supported through .gif copies of application windows are posted to a single graphics window. The .gif copy performance was somewhat disappointing, taking about 8 seconds to capture and display an image. Consequently, a compressed, private protocol was extended and implemented in the Java applet. Graphical windows are captured, transmitted, and displayed in under two seconds on a LAN.

This applet has been downloaded and run on several platforms, operating systems, and browsers. The write once, run anywhere capability has been demonstrated, but not without testing and some implementation compromises.

The Java development environment supporting this project is Symantec SVisual Cafe. The software engineering tools in this and other development sets are easy to learn, well integrated, and a joy to use. The development cycle is very quick. The project began using the Java Development Kit, JDK version 1.02, and has since moved to JDK version 1.1.

4 Conclusions

The web and its tools are essential to the development, maintenance, and operation of the accelerator complex. The web is the common, acceptable repository for documentation. Automatically generated reports enhance the diagnostic toolset. Dynamic software development environments supporting web based application development promise to fulfill the software engineering goals of drag and drop, code generation and write once, run anywhere technology.

References

[1] www-bd.fnal.gov/consolesii.html
[2] www-bd.fnal.gov /controls/micro_p/micro_p.html
[3] www-bd.fnal.gov/errors/
[4] www-bd.fnal.gov/webhelp_edit/menu.html
[5] cns40.fnal.gov/javaconsole/autogen_javaconsole.html