ISBN: 978-3-95450-200-4 doi:10.18429/JACoW-PCaPAC2018-WEP06

DATA ARCHIVING AND VISUALIZATION OF IRFEL*

Y. Song, X. Chen, C. Li, K. Xuan, J. Wang, G. Liu†, NSRL, USTC, Hefei, Anhui 230029, China

Abstract

An Infrared Free Electron Laser Light (IRFEL) is being constructed at National Synchrotron Radiation Laboratory. The EPICS Archiver Appliance provides the functions of historical data acquisition, archiving, migration, retrieval and management in the IRFEL facility. A Single-Page Web Application is developed for the data visualization based on Vue.js framework and Highcharts JavaScirpt library. A unified interface is developed for the visualization to integrate multiple data sources and provide the same retrieval entry of the historical data from EPICS Archiver Appliance, the real-time data from EPICS IOC, the statistical data from database and the alarm information from the Phoebus. This paper will describe the implementation details of data archiving and visualization of IRFEL.

INTRODUCTION

Tunable Infrared Laser for Fundamental of Energy Chemistry (FELiChEM) is the significant scientific instrument, which is supported by the National Natural Science Foundation of China in 2013. Infrared Free Electron Lasers (IRFEL) is the core part of FELiChEM, which can accelerate beam to 60MeV and generate middle-infrared and far-infrared laser [1]. The control system of IRFEL is developed based on Experimental Physics and Industrial Control System (EPICS) [2].

The operation of a particle accelerator complex is a long-term experiment. It is essential to record Process Variables (PVs) for further data analysis. With the continuous development of EPICS, a series of data archiving systems has been released in the EPICS community, such as, Channel Archiver, RDB Channel Archiver and the EPICS Archiver Appliance [3]. The EPICS Archiver Appliance (AA) is used as the data archiving tool for the IRFEL. A Web-based GUI is developed for the data visualization. It not only provides the function of historical data query, but also integrates real-time data, statistical data and alarm information into a web application. This paper will describe the implementation details of data archiving and visualization of IRFEL.

SYSTEM ARCHITECTURE

The EPICS Archiver Appliance (AA) is a data archiving tool released in the EPICS community and provides the functions of data acquisition, storage, migration, retrieval and management in the IRFEL facility [4]. AA divides the data into three types: short term store (in RAM disk), medium term store (in local SSD disk) and long term store (in NFS).This mechanism makes AA have high data retrieval performance, which is helpful to machine status analysis and fault diagnosis.

Figure 1 shows the overall structure of the whole system. It is a typical web application and implements the complete separation of the front and back end. The data displayed on the front end can be divided into four types: real-time status, historical data, statistical data and alarm information. As a reverse proxy server, Nginx provides a unified interface for querying these four types of data. The initial sources of the data are all the IOCs in the control system, but they are processed through different ways. The implementation will be described in detail about how to archive, store, process and visualize the data in the following sections.

IMPLEMENTATION OF SERVER SIDE

Real-time Data

It is necessary to display the real-time state of accelerator (such as beam current, running state, etc.) on the web page. However, data archiving tools generally have some time delay to write data to the database and real-time data can not be obtained from archiving tools. In order to solve this problem, the real-time state push is implemented by WebSocket communication from IOC to web page directly.

The WebSocket protocol provides a way of creating web applications that support real-time bidirectional communication between clients and servers. Figure 2 shows the principle of the status push program. Its core is the *Websocketd*, which is a small command-line tool that will wrap an existing command-line interface program, and allow it to be accessed via a WebSocket. As long as you can write an executable program that reads STDIN and writes to STDOUT, you can build a WebSocket server in any programming language [5].

A python script is developed to monitor the PV change in IOC Record via Channel Access and writes them into STDOUT. Any text printed by the process to STDOUT shall be sent as a WebSocket message whenever a newline is encountered. In this way, the JavaScript script running in the browser can get the latest change values of the PVs through the WebSocket and display them on the web page.

Historical Data

The EPICS Archiver Appliance comes with a web interface that has support for various business processes. The web interface communicates with the server principally using HTTP/JSON. There is a rich catalog of business logic that lets the user add/modify/delete PVs from the archiver. This design provides convenience for its integration with other systems. Therefore, client application can get historical data directly from this interface.

* Work supported by National Natural Science Foundation of China (No.11375186, No.21327901)
† Corresponding author, gfliu@ustc.edu.cn

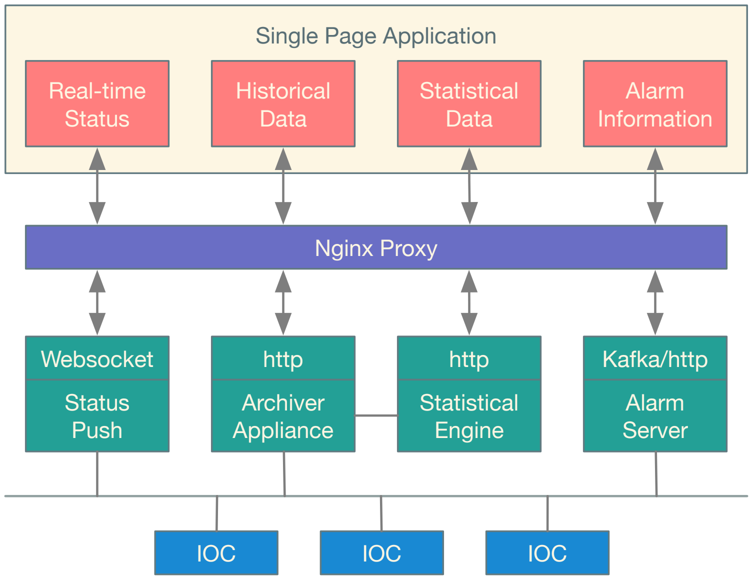

Figure 1: Architecture of the whole system.

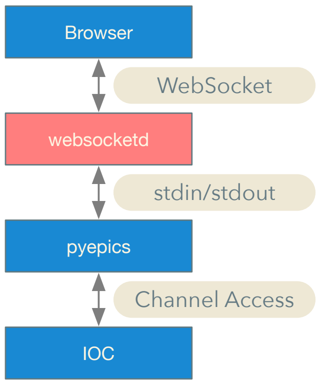

Figure 2: Principle of the status push program.

Statistical Data

The statistical data are obtained after processing historical data in AA. It mainly includes operation mode statistics and integral current query. A daemon is developed using Python based on Flask to get data from AA, perform calculations within itself, and provide HTTP interfaces for client side.

Alarm Information

The Alarm system of IRFEL is build based on Phoebus. Compared with BEAST using Java Message Server(JMS), Phoebus uses Kafka as the communication method between alarm server and alarm client [6]. Similar to the way of processing Statistical Data, a daemon is also developed based on Flask to get alarm message from Kafka using kafka-python library and provide HTTP interfaces for client side query.

IMPLEMENTATION OF CLIENT SIDE

The EPICS Archiver Appliance offers a Web UI for typical configuration tasks and data presentation. However, this Web UI doesn't meet our requirements in the PV classification and search. So we developed a new web application to provide real-time data display and historical data query as a Web-Based GUI.

The Web-based GUI is a Single-Page Application (SPA) which is more application-like and interacts with the user by dynamically rewriting the current page rather than loading entire new pages from a server, which is used as the client side to interact with users. The JavaScript framework Vue.js is used to develop the SPA. Vue.js is a progressive framework for building user interfaces and is also perfectly capable of powering sophisticated SPA when used in combination with Single File Components and supporting libraries [7].

Figure 3 shows the screenshot of the Web-Based GUI, which provides five tabs: Operation Status, Historical Data, Data Analysis, Alarm and Help. The front end runs in an Express web server and the data in the web page is obtained directly from the the Nginx via HTTP/JSON. The operator can select the PV by the check box in front of the node. The chart component on the right is the data visualization part based on Highcharts JavaScirpt library [4]. We can find the pv nodes in the tree, then select the check box in front of it, and the right side will show the curve during the last 24 hours. The buttons below provide many features, such as viewing and downloading raw data, setting coordinate axes, switching between linear and logarithmic coordinates, and so on.

SUMMARY

In order to facilitate the construction and commissioning of IRFEL facility, we developed a new system to archive, store, process and visualize the data. The back end service contains data from multiple sources. The front end web application provides a operation interface for users. Nginx acts as a reverse proxy server to connect the front and back-

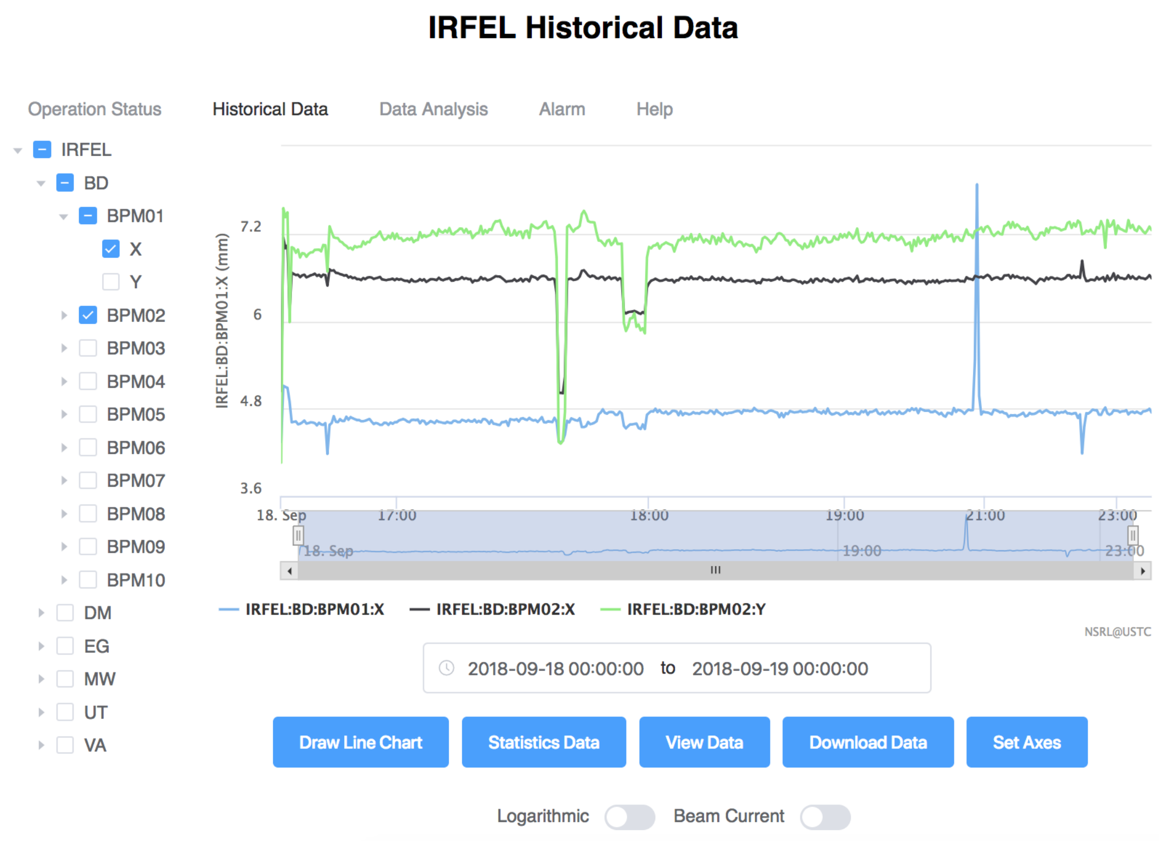

Figure 3: Screenshot of the Web-Based GUI.

end. The design of the complete separation of the front and back end is also convenient for future expansion, like adding new data types and adding new methods for interaction with users.

The system was put into operation in July 2018. The test results indicate that the new archiving and visualization system is reliable, flexible and convenient to operate.

REFERENCES

[1] H.T. Li, Z.G. He, Q.K. Jia, Q. Luo, L. Wang, and S.C. Zhang, "Status of FELiCHEM, a New IR-FEL in China", in *Proc. IPAC'16*, Busan, Korea, May 2016. doi:10.18429/JACoW-IPAC2016-MOPOW026

[2] S. Xu, G. Liu, Y. Song, and X.K. Sun, "Control System Design for Front End Devices of IRFEL", in *Proc. IPAC'18*, Vancouver, BC, Canada, Apr/May 2018, pp. 4920–4922. doi:10.18429/JACoW-IPAC2018-THPML109

[3] M. Shankar, L. Li, M. Park, *et al.*, "The EPICS Archiver Appliance", in *Proc. ICALEPCS'15*, Melbourne, Australia, Oct. 2015. doi:10.18429/JACoW-ICALEPCS2015-WEPGF030

[4] Highcharts home page, https://www.hcharts.cn/

[5] Websocketd home page, http://websocketd.com/

[6] Phoebus Documentation, https://phoebus-doc.readthedocs.io/en/latest/

[7] Vue.js home page, https://vuejs.org/