Automatic data archiving and visualization at HLS-II

Yi-Fan Song[1] · Chuan Li[1] · Ke Xuan[1] · Gong-Fa Liu[1]

Received: 8 December 2017 / Revised: 25 February 2018 / Accepted: 27 March 2018 / Published online: 27 July 2018

Abstract The control system of Hefei Light Source II (HLS-II) is a distributed system based on the experimental physics and industrial control system (EPICS). It is necessary to maintain the central configuration files for the existing archiving system. When the process variables in the control system are added, removed, or updated, the configuration files must be manually modified to maintain consistency with the control system. This paper presents a new method for data archiving, which realizes the automatic configuration of the archiving parameters. The system uses microservice architecture to integrate the EPICS Archiver Appliance and RecSync. In this way, the system can collect all the archived meta-configuration from the distributed input/output controllers and enter them into the EPICS Archiver Appliance automatically. Furthermore, we also developed a web-based GUI to provide automatic visualization of real-time and historical data. At present, this system is under commissioning at HLS-II. The results indicate that the new archiving system is reliable and convenient to operate. The operation mode without maintenance is valuable for large-scale scientific facilities.

Keywords Automatic archiving · Data visualization · Microservice architecture · EPICS Archiver Appliance

1 Introduction

The Hefei Light Source (HLS) at the National Synchrotron Radiation Laboratory is the first dedicated synchrotron radiation facility in China, which provides radiation from infrared (IR) to vacuum ultraviolet (VUV) for various user programs. HLS was upgraded from 2010 to 2015 to improve its performance. The upgraded light source was renamed Hefei Light Source II (HLS-II) [1–3]. It comprises an 800-MeV linac, an 800-MeV storage ring, and a transport line connecting the linac and the storage ring. The HLS-II was fully open to users in January 2016. Experimental physics and industrial control system (EPICS) is a set of open-source software tools, libraries, and applications that were developed collaboratively and used to create distributed soft real-time control systems for scientific instruments, such as particle accelerators and telescopes [4–7]. The control system of HLS-II is a distributed system based on EPICS [8].

The operation of a particle accelerator complex is a long-term experiment. It is essential to record as many process variables (PVs) as possible with a reasonable rate. Therefore, all control system frameworks provide data-archiving tools to archive beam parameters, real-time equipment data, alarm records, etc. [9, 10]. With the continuous development of EPICS, a series of data-archiving systems has been released in the EPICS community, such as SDDSlogger, Channel Archiver, RDB Channel Archiver, and the EPICS Archiver Appliance. Around 2000, in the HLS control system, SDDSlogger was run on a Sun E250 server to archive the values of various channels [11]. In 2008, the Channel Archiver was used for obtaining and storing data from input/output controllers (IOCs). The Channel Archiver acts as a channel-access client and stores

This work was supported by the National Natural Science Foundation of China (No. 11375186).

✉ Gong-Fa Liu
gfliu@ustc.edu.cn

[1] National Synchrotron Radiation Laboratory, University of Science and Technology of China, Hefei 230029, China

recorded data, acquired via periodic scan or monitoring, into its indexed binary files [12]. However, it is not convenient for complex data queries, does not have database–transaction–processing capabilities, and also deficient in managing data and storing data for a long time. In addition, it is no longer maintained. In 2013, the archive engine of RDB Channel Archiver was used to collect data from the IOCs of the HLS-II control system and to store them in an Oracle database [13]. Thus, we can use the SQL language to write a complex data query program. However, data retrieval is slow. We moved data originally stored on iSCSI to a local disk with a faster read and write speed in 2016, but the overall system query is still not fast enough. In 2017, we began to evaluate the EPICS Archiver Appliance, which is developed by a collaboration of SLAC, BNL, and MSU, as a data-archiving tool to achieve high data retrieval performance [14]. It also provides a convenient HTTP interface to facilitate the data query and system management. Furthermore, it supports EPICS7 record types and will be the first choice for future data-archiving tools within the EPICS control system.

Maintaining the central configuration files to specify global parameters and channel-specific parameters is necessary for all the software mentioned above. When the PVs in the control system are added, removed, or updated, the configuration files must be manually modified to maintain consistency with the control system. This is inconvenient for the long-term operation of the accelerator complex, especially for the large-scale facilities of the future.

This paper presents a new method for data archiving in the EPICS control system that realizes automatic configuration and visualization. The system uses microservice architecture to integrate the EPICS Archiver Appliance and RecSync. With this method, all the archived meta-configurations from the distributed IOCs are gathered together and entered into Archiver Appliance automatically and dynamically. A web-based GUI was also developed to provide the automatic visualization of real-time data and historical data. This system is under commissioning at the HLS-II control system.

This paper focuses on the design and construction of the automatic archiving system. In detail, the paper is organized as follows: Sect. 2 provides an overview of the system architecture. Section 3 introduces the design of the automatic archiving process and the integration between the EPICS Archiver Appliance and RecSync. Section 4 describes the development of the web-based GUI in the data visualization component and explains in particular the implementation of automatic visualization based on the PVs' naming conventions. Finally, Sect. 5 summarizes the entire paper.

2 Architecture

The microservice architecture pattern is adopted in the design of the HLS-II archiving system. Microservice architecture is a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services [15], which are often used in complex systems composed of multiple components. Because the system is split up into smaller, separately deployed components, applications built using the microservice architecture pattern are more robust and extensible.

Figure 1 shows the overall structure of the HLS-II archiving system, which is divided into three components: the Auto Configurator, the EPICS Archiver Appliance, and Data Visualization. The Auto Configurator component is responsible for the acquisition, collection, and analysis of archived meta-configurations. The EPICS Archiver Appliance is the data-archiving tool. The Data Visualization component contains three types of applications: Web-Based GUI, Client Application and Script Program. Each component is designed, developed, deployed, and maintained independently. The dotted lines represent the flow of the configuration information, and the solid lines represent the flow of historical data. Communication between different distributed components or between different modules in a component is implemented by Message Middleware or Web API.

The EPICS Archiver Appliance is a well-designed archiving tool in the HLS-II archiving system and provides the functions of data acquisition, storage, migration, retrieval, and management. The EPICS Archiver Appliance uses an appliance model for deployment. Each appliance has multiple storage stages and multiple processes. It divides the data into three types: short-term store (STS), medium-term store (MTS), and long-term store (LTS).

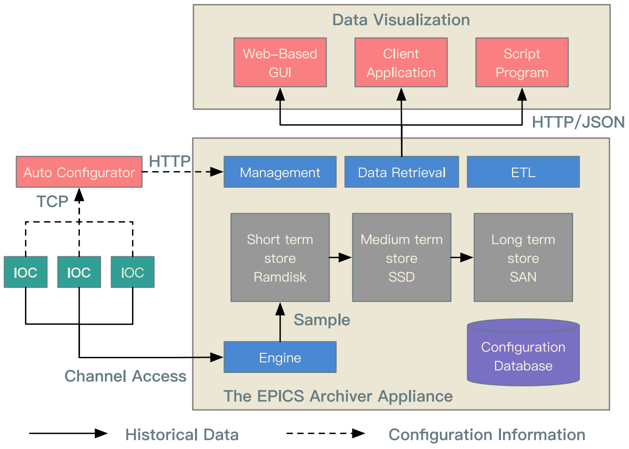


Fig. 1 (Color online) Architecture of the HLS-II archiving system

Each appliance has 4 processes. The engine establishes EPICS Channel Access monitors for each PV in the appliance. Data Retrieval gathers data from all the stores and stitches them together to satisfy data retrieval requests. ETL(Extract, Transform and Load) moves data from the STS to the MTS and from the MTS to the LTS. The management executes business logic, manages the other three components, and stores the runtime configuration state.

3 Automatic archiving

3.1 Meta-configuration

All the archiving tools mentioned in Sect. 1 use the central configuration files to specify global parameters, such as the write period and file-size threshold, as well as channel-specific parameters. In the Channel Archiver, every channel definition is composed of a set of XML tags specifying the parameters. In the EPICS Archiver Appliance, the archiving configuration is stored in a MySQL database table and each appliance has its own database table. The configuration database table has an entry for each PV. This entry is a JSON object that has details on how the PV is being archived, which can be called the *meta-configuration*.

In the EPICS IOC, an info() tag can be defined in a record as a piece of information. It provides a way to attach named string values to individual record instances that are loaded at the same time as the record definition. Hence, we define a string format to store *Meta Configuration* in the info() tag as follows:

$$info(arch, ``enable, period, method, appliance'')\ ,$$

where *arch* identifies the *meta-configuration* for the archiving system. “enable, period, method, appliance” is a string that is separated by commas into four segments, each of which specifies the different properties, as shown in Table 1. For example, the record SR_ID_EPU_Gap represents the gap in the EPU undulator in the HLS-II storage ring. We can add a row, $info(arch, ``1, 1, scan, appliance0'')$, to the record instance. With this method, all configuration information is distributed in the IOC runtime database and central configuration files are unnecessary.

Table 1 *Meta-configuration* properties for archiving system

Property	Description	Default
Enable	A boolean value to specify effectiveness	1
Period	The sampling period, the unit is second	1
Method	The sampling method, scan or monitor	Scan
Appliance	Specify an appliance in a cluster	appliance0

3.2 Improvement of RecSync

The RecSync is used to collect the *meta-configuration* distributed in each IOC runtime database automatically. It is a record synchronizer project that includes two parts: a client (RecCaster) that runs as part of an EPICS IOC, and a server (RecCeiver) that is a stand-alone daemon [16]. Both of them communicate through TCP/UDP and together work to ensure that the server has a complete list of all records currently provided by the client running in the IOCs, as shown in Fig. 2.

The RecCaster client sends the following information to the RecCeiver: the EPICS base version, a listed set of environment variables, the name and type of all records, and any info() tags associated with these records, as shown in Fig. 2. The RecCeiver server is based on the Twisted framework, which is an event-driven networking engine in Python. It uses the Twisted plugin interface to make client information available to one or more plugins [17]. For example, the db plugin stores the collected information in a SQLite database and the show plugin prints client information to screen/log.

The plugin system makes RecSync easy to expand. In order to facilitate integration with the Archiver Appliance, we developed a JMS plugin in Python. This plugin obtains all the collected information from the RecCeiver and filters out the *Meta Configuration*, which is packaged into a Java Message Server (JMS) message format. The JMS plugin publishes this information to JMS in the role of the message producer. The JMS publish/subscribe (Pub/Sub) message model is used to realize the asynchronous communication between the RecSync and Auto Configurator.

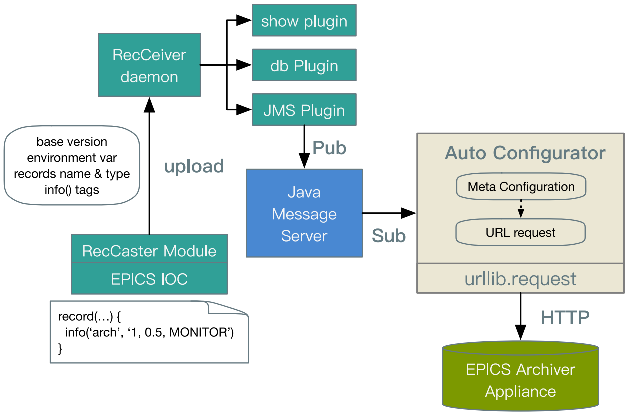


Fig. 2 (Color online) Principle of the automatic archiving

3.3 Auto Configurator

The EPICS Archiver Appliance comes with a web interface that supports various business processes. The web interface communicates with the server principally using HTTP/JSON. There is a rich catalog of business logic that allows the user to add, modify, or delete PVs from the archiver. This design provides convenience for its integration with other systems.

The Auto Configurator is the core part to connect the EPICS Archiver Appliance and RecSync. It is developed using Python and runs as a daemon. The Auto Configurator subscribes to the "archiver" topic in JMS. When a message containing the ?arch? tag is heard, it will parse the info string to obtain the *meta-configuration*. According to this message and the web interface format given by the EPICS Archiver Appliance, a URL request is created as:

http://192.168.125.102:17665/mgmt/bpl/archivePV?pv=SR_ID_EPU_Gap&samplingperiod=1&samplingmethod=MONITOR.

The Auto Configurator then uses the urllib.request module to send this request to the EPICS Archiver Appliance, as shown in Fig. 2. This archiving information is added to the configuration database, and the engine samples and stores the PVs according to the archiving parameters. In this way, automatic archiving is implemented.

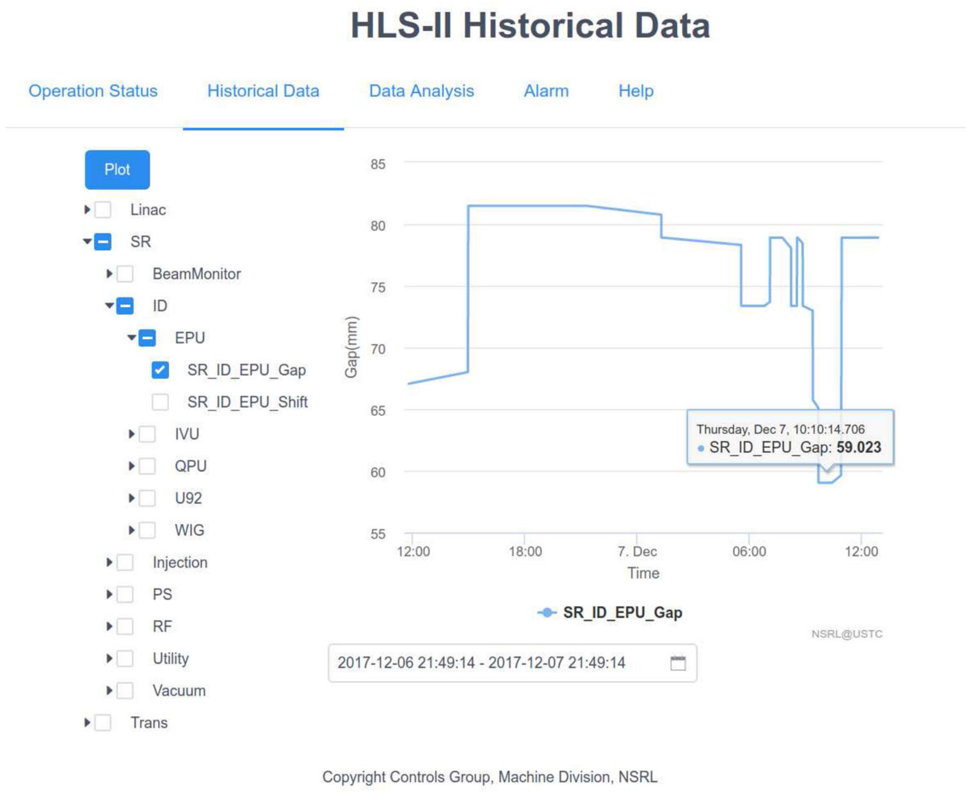
4 Automatic visualization

The Data Visualization component contains three types of applications: a web-based GUI, client application, and script program. The web-based GUI's operation interface is the most convenient way for the operator to interact with archiving systems [18]. The EPICS Archiver Appliance offers a web UI for typical configuration tasks and data presentation. However, this web UI does not meet our requirements in the PV classification and search. Therefore, we developed a new web application to provide a real-time data display and historical data query as a web-based GUI. It directly obtains the list of all archived PVs from the EPICS Archiver Appliance and constructs the hierarchical structure according to the name of the PVs, thereby realizing automatic visualization.

4.1 Single-page application

The web-based GUI is a single-page application (SPA) that is application-like and interacts with the user by dynamically rewriting the current page rather than loading entire new pages from a server. The JavaScript framework

Fig. 3 (Color online) Screenshot of the web-based GUI



2017-12-06 21:49:14 - 2017-12-07 21:49:14

Copyright Controls Group, Machine Division, NSRL

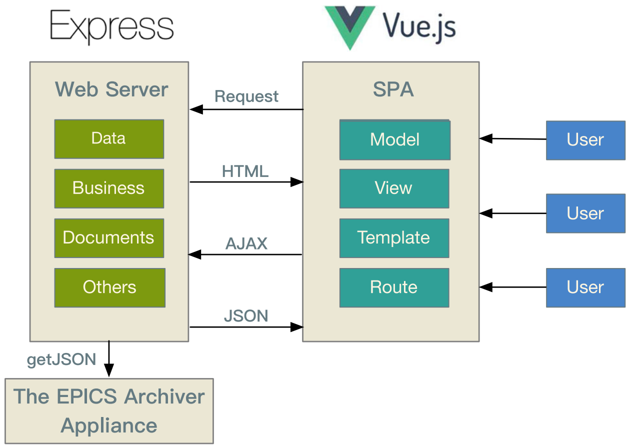


Fig. 4 (Color online) Structure diagram of the web-based GUI

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 SPAs when used in combination with single file components and supporting libraries [19]. Figure 3 shows a screenshot of the web-based GUI, which provides five tabs: Operation Status, Historical Data, Data Analysis, Alarm, and Help.

This project implements the complete separation of the front and back ends, as shown in Fig. 4. The users interact directly with the front end. This SPA is based on Vue.js, and contains the model, view, template, route, and so on. The back end runs in an Express web server. It is responsible for the implementation of the business logic and the storage of related data and documents. The data in the web page is obtained from the EPICS Archiver Appliance via HTTP/JSON.

4.2 Tree structure

In the development of the IOC application for large-scale scientific facilities, a well-designed naming convention is very important. HLS-II consists of a linac, a storage ring, and a transport line. In order to manage the corresponding records effectively and access them conveniently, the naming of all records follows the rule below:

Segment_System_Device_Channel

Instead of using an XML configuration file, the web-based GUI relies on parsing the name of the PVs to generate a tree structure, which represents the hierarchical structure of the PVs. At the highest level, all the PVs are divided into three main segments: Linac, Trans, and SR. The subtree separates the different systems, such as the insertion device (ID), power supply (PS), radio frequency (RF), and vacuum. At the bottom level, the leaf node represents the name of the PV. The operator can select the PV from a check box in front of the node. The chart component on the right is the data visualization part based on the Highcharts JavaScript library. Taking the record SR_ID_EPU_Gap as an example, we can find its node in the SR tree and ID subtree, then select the check box in front of it and click the plot button, and the right side will show the curve during the last 24 h, as shown in Fig. 3.

5 Conclusion

The paper presents a new method for data archiving in the EPICS control system, which realizes the automatic configuration of archiving parameters for HLS-II. The system integrates the Archiver Appliance and RecSync to collect all the archiving *meta-configurations* from the IOCs automatically under the microservice architecture. Furthermore, we also developed a web-based GUI to provide the automatic visualization of real-time and historical data. At present, this system is under commissioning at the HLS-II, 1776 PVs are stored in the EPICS Archive Appliance, and the data rate is about 1 GB/day. The automatic archiving and visualization reduce the maintenance of the HLS-II archiving system. The high data retrieval performance of the EPICS Archiver Appliance is helpful for machine status analysis and fault diagnosis. The results indicate that the new archiving system is reliable, convenient, and valuable for large-scale scientific facilities.

At present, this system only supports automatic archiving in the EPICS Version 3 environment. Further development is needed to support automatic archiving of structured data, which is introduced in EPICS 7. In addition, this automatic configuration method can be extended to other systems, such as the alarm system. This can be done by adding a new identifier “alarm” and defining a new corresponding configuration string format, like $info(alarm, ``enable, group, email, sms, duration'')$. The Auto Configurator can collect the *meta-configuration* for the alarm system at the same time as the other data.

Acknowledgements The authors would like to thank Dr. Guo-Bao Shen of the Argonne National Laboratory (ANL) for his beneficial inspiration and discussions about this paper.

References

1. Z. Bai, L. Wang, Q. Jia et al., Lattice optimization for the HLS-II storage ring. Chin. Phys. C **37**, 017001 (2013). https://doi.org/10.1088/1674-1137/37/1/017001
2. Z. Bai, L. Wang, Q. Jia et al., Lattice study for the HLS-II storage ring. Chin. Phys. C **37**, 047004 (2013). https://doi.org/10.1088/1674-1137/37/4/047004
3. H. Zhang, W. Li, G. Yao et al., The magnet design for the HLS storage ring upgrade project. Chin. Phys. C **36**, 91–95 (2012). https://doi.org/10.1088/1674-1137/36/1/016

4. W. Long, H. Sun, X. Li, et al., The CSNS RCS remote control system based on EPICS, in *The 2nd International Conference on Systems and Informatics, ICSAI 2014, Shanghai, China* (2014), pp. 89–93
5. T. Ma, Y. Yang, B. Sun et al., Development and application of the new BPM system data processing program at Hefei light source. Nucl. Sci. Tech. **23**, 261–266 (2012). https://doi.org/10.13538/j.1001-8042/nst.23.261-266
6. L. Han, Y. Chen, J. Cai et al., The application of EPICS in TMSR radiation protection and access control system. Nucl. Sci. Tech. **27**, 41 (2016). https://doi.org/10.1007/s41365-016-0040-7
7. EPICS home page. http://www.aps.anl.gov/epics/about.php
8. C. Li, J. Wang, K. Xuan et al., Event-driven timing system based on MRF cPCI hardware for HLS-II. Nucl. Sci. Tech. **26**, 060401 (2015). https://doi.org/10.13538/j.1001-8042/nst.26.060401
9. R. Muller, *Control Systems for Accelerators: Operational Tools. Synchrotron Light Sources and Free-Electron Lasers* (Springer, Cham, 2016), pp. 629–670
10. Z. Hu, Q. Mi, F. Zheng et al., EPICS data archiver at SSRF beamlines. Nucl. Sci. Tech. **25**, 020103 (2014). https://doi.org/10.13538/j.1001-8042/nst.25.020103
11. G. Liu, W. Li, J. Li, The status of HLS control system, in *The Particle AcceleratorConference, PAC 2005, Knoxville, USA* (2005), pp. 3862–3864. https://doi.org/10.1109/PAC.2005.1591649
12. X. Bao, C. Li, K. Xuan et al., Accelerator data collection system based on Channel Archiver. High Power Laser Part. Beams **20**, 589–592 (2008). **(in Chinese)**
13. J. Qiang, G. Liu, Jie Qiang, The data archive system and preliminary applications of Hefei light source based on Oracle. Nucl. Tech. **36**, 1–5 (2013). **(in Chinese)**
14. M. Shankar, L. Li, M. Park, et al., The EPICS Archiver Appliance, in *The 15th International Conference on Accelerator and Large Experimental Physics Control Systems , ICALEPCS 2015, Melbourne, Australia* (2015), pp. 761–764
15. M. Richards, *Software Architecture Patterns* (O'Reilly Media, Newton, 2015), pp. 27–36
16. RecSync home page. https://github.com/ChannelFinder/recsync
17. Twisted home page. https://twistedmatrix.com/trac/
18. G. Jud, A. Luedeke, W. Portmann, The PSI Web Interface to the EPICS Channel Archiver, in *The 13th International Conference on Accelerator and Large Experimental Physics Control Systems, ICALEPCS2011, Grenoble, France* (2013), pp. 141–143
19. Vue.js home page. https://vuejs.org/