EUROFUSION WPS1-CP(16) 15791

M Grahl et al.

W7-X ArchiveDB WEB API: a web service for the experiment data archive of Wendelstein 7-X

Preprint of Paper to be submitted for publication in Proceedings of 29th Symposium on Fusion Technology (SOFT 2016)

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

Archive WEB API: a web service for the experiment data archive of Wendelstein 7-X

M.Grahl, T.Bluhm, M.Grün, C.Hennig, A.Holtz, J.G.Krom, G.Kühner, H.Laqua, M.Lewerentz, H.Riemann, A.Spring, A.Werner, the W7-X Team

Max-Planck-Institut für Plasmaphysik, Wendelsteinstr. 1, D-17491 Greifswald

Wendelstein 7-X was built for the research on steady state operation of stellarators. This sets high requirements on the control and data acquisition (CoDaC) system, with the archive database as one of its main components. W7-X Archive database is the storage of all operational and scientific data. It stores raw data as well as processed data and provides a single database for a wide range of components and data types. The Archive WEB API provides a unified access to the archive database by using web service technologies. The API is based on the REST architecture style, using HTTP as the application protocol: the data is represented as a unified signal resource, which can be addressed via a URL. The users can read data in standard formats, like JSON or PNG. In the same way, the WEB API can be used to import new data to ArchiveDB. This paper describes the concepts of the WEB API and the experiences during the first operational phase of W7-X.

Keywords: Wendelstein 7-X, REST, web service, continuous data acquisition, data analysis, steady state

1. Introduction

The stellarator experiment Wendelstein 7-X (W7-X) started its operation in December 2015 [1]. W7-X with its superconducting coil system is designed for the research on steady-state operation. This sets high requirements on the W7-X control and data acquisition (CoDaC) system, with the archive database [2] as one of its main components. The archive is the central storage for all experiment data of W7-X. The archive system is an in-house developed Java data store, which is working on top of a distributed file system. The data is organized in continuous streams, with time stamps as the primary key. A central timing system [3] synchronizes all components' clocks, thus making the stored data comparable. Therefore time stamps in a nanosecond resolution are recorded together with every data sample. The main focus of the first operational campaign (OP 1.1) of W7-X was the commissioning of the machine, the ECRH heating system and the segment control system. Nevertheless, also over 20 diagnostics systems [4] were already available, producing first scientific results.

For a convenient and system-independent read and write access to the archive database a RESTful web service was implemented: the Archive WEB API.

2. Experiment Data

The W7-X archive stores operational as well as scientific data. This leads to a large variety of data: covering a wide spectrum of types and sizes, written with different frequencies to the archive.

The operational data is mostly machine status data, often from 24/7 running data acquisition (DAQ) systems. As W7-X operates superconducting coils, many of its operational systems have to run over a long time period, such as the vacuum or cryostat systems. These systems are continuously writing their status information and measured values to the archive. The machine monitoring contains over 8000 sensors mounted around the torus, which are measuring values like stress and displacements. The DAQ of operational systems mostly produce data in sustainable sizes (payloads of several Kbyte) and write frequencies in a range of a second or slower.

The archive also stores the raw and processed data from heating systems and plasma diagnostics. The DAQ of many of these systems are based on high-sampling ADCs: multiple channels with often mega sample data rates. This leads to large datasets per experiment run. In addition video diagnostics are becoming more and more an important part of the diagnostic portfolio. The cameras of such systems can produce high-resolution images, often in very fast sequence: e. g. 400 images per second. Thereby different filters are used to produce images for different spectrums e. g. visible light and near-infrared. The images are archived as raw pixel data to prevent information loss. Many diagnostic systems are making use of the stellarator symmetry and place identical sensors multiply times around the machine. This leads to even higher data rates per experiment run. For later campaigns, the annual growth of archive data is expected to be in the range of petabytes.

3. Web Services

In software development the term web service describes techniques for the integration of software over a network. Thereby, the web service API (Application Programming Interface) often includes standard protocols for the message encoding (e.g. XML or JSON) and transport (e.g. HTTP). This allows a system-independency design of server and client part

author's email:michael.grahl @ipp.mpg.de

of the software. Both can be written in different programming languages, running on all kinds of computer systems.

3.1 Web Service Techniques

Web services are often classified into two implementation principles (see e.g. [5]): Remote Procedures Call (RPC) or message oriented APIs (often based on the SOAP/WSDL standards) and web services, which are using the Representational State Transfer (REST) [6] architecture style. The first one uses function calls over a network, often with HTTP as the transport protocol. All functions and data types used in the service have to be described in an explicit interface definition. A REST API, on the other hand, often uses HTTP as a full application protocol and the data format is not limited to XML. The elements which typically constitute a RESTful web service are explained in Sec. 4, with reference to their usage in the Archive Web API.

3.2 Application in Fusion Research

Web services are commonly used in industry and science. At tokamaks and stellarators, web services are mainly used for data access, modeling, and integrated data analysis [7]. For the access on experiment data, there are examples using SOAP [8] and also REST: for example at TEXTOR [9] or at H1 [10]. At W7-X, several web services are in use, like a field line tracer [11] or a web service for MHD equilibrium data. These services build a Service Oriented Architecture (SOA).

Archive WEB API is the successor of a SOAP-based web service [8]. The development was started in December 2012 as a RESTful web service, which should provide a convenient access to the new W7-X archive system by supporting a listing of archive structures, reading in different formats and also providing write access.

4. Interface Design

There are some key elements, which characterize a RESTful web service: *resources* on a server can be addresses by a *URL*. The client can interact with a resource via *representations*, by using standard *HTTP operation*s. The navigation between resources is done via *hyperlinks*.

4.1 Resources on a Server

In REST APIs a client can interact with resources on a server. The resources within the Web API are the archived data, represented by universal signal objects. A signal can represent all kind of data in a unified way: it contains a one- or multidimensional data sample array, as well as an array of the corresponding time stamps (*values* and *dimensions*, see Fig. 1).

4.2 URL

Resources are addressable by a Uniform Resource Locator (URL). In the Archive WEB API, the basic address schema represents the data hierarchy of the archive database: Project, Stream Group and Data Stream (for bulk data) or Parameter Log (for meta data). In addition each single parameter of a log can be addressed by its own URL. For bulk data, it is possible to address a single channel (e.g. the channels of an ADC) of a data stream by its own URL.

In continuous systems, time is the key element. Therefore time stamps are also part of a signal URL: query strings are marking the start and end time of a requested data interval. Further query strings are used for different settings, e.g. for down sampling rates or special settings for image data. In addition to the basic schema, addresses can be assigned via an alias system. This can be used to define semantic URLs or to mark evaluated data. The alias definitions are also stored in the archive.

```
import requests

r = requests.get('http://archive-webapi/programs.json',params={'from':'20160310.010'})
program = r.json()['programs'][0]
print(program['name'])
start = program['from']
stop = program['upto']

url = 'http://archive-webapi/ArchiveDB/raw/W7X/CBG_ECRH/TotalPower_DATASTREAM'
r = requests.get(url + '/_signal.json', params={'from' : start, 'upto': stop}).json()

print(r['values'])
print(r['dimensions'])
```

Fig. 1 Sample API calls in Python using the requests library [12]. The first request returns the program log of discharge XP20160310.010. In the second call, the start and end timestamps of the discharge are used for reading the total power of the ECRH.

4.3 Usage of HTTP Operations

Unlike in RPC-style web services, the functions in a REST API are limited to a fixed number of standard operations (e.g. HTTP operations) which can be performed on a resource. For example by using the *GET* operation, the user can retrieve a representation (see Sec. 4.4) of a resource, which means in case of the Archive Web API: read experiment data from the archive. The *POST* operation is used to create a new resource on the server: write new data to the database.

4.4 Representations

For different use cases, resources of the archive can be received in a suitable format: for example a website in HTML for the interactive navigation and exploring manually via web browser, or a JSON representation for the processing data programmatically. The Archive Web API provides representations in following formats:

- The HTML representation provides a website for every signal. A signal website contains a preview of the data in table form or as plots (Fig. 2), which are generated on the client-side using JavaScript.

- JSON (JavaScript Object Notation) [13] is a simple data format, which is widely used for data encoding in REST web services. JSON data is compact and easy to parse in most programming language.
- Image data can also be read as JPEG or the lossless PNG format (Fig. 2). The image files are created on-the-fly from the archive raw data. For the upload of image data, the WEB API supports (a subset of) HDF5 [14] as an alternative format.

A separate website provides access to the archived experiment program logs [15] of W7-X discharges (Fig. 3). The user can select a day and a program on the interactive website and receive the corresponding log data. This includes informations like the program id, description, system settings, and session leader comments. The website contains links, which are pointing to archive data from the selected discharge. For the usage in software, users can read program logs in JSON format, too (Fig. 1).

4.5 Hypermedia

The Web API homepage contains comprehensive documentation with sample code in several programming languages. Starting from the homepage, the user can follow hyperlinks to navigate through the data hierarchy of the archive and find the data of all W7-X components and for all time periods. The representation (HTML or JSON) of an address part contains a link to the next address parts or to time interval which contain data. For hyperlink representations in JSON the WEB API follows the HAL specification [16].

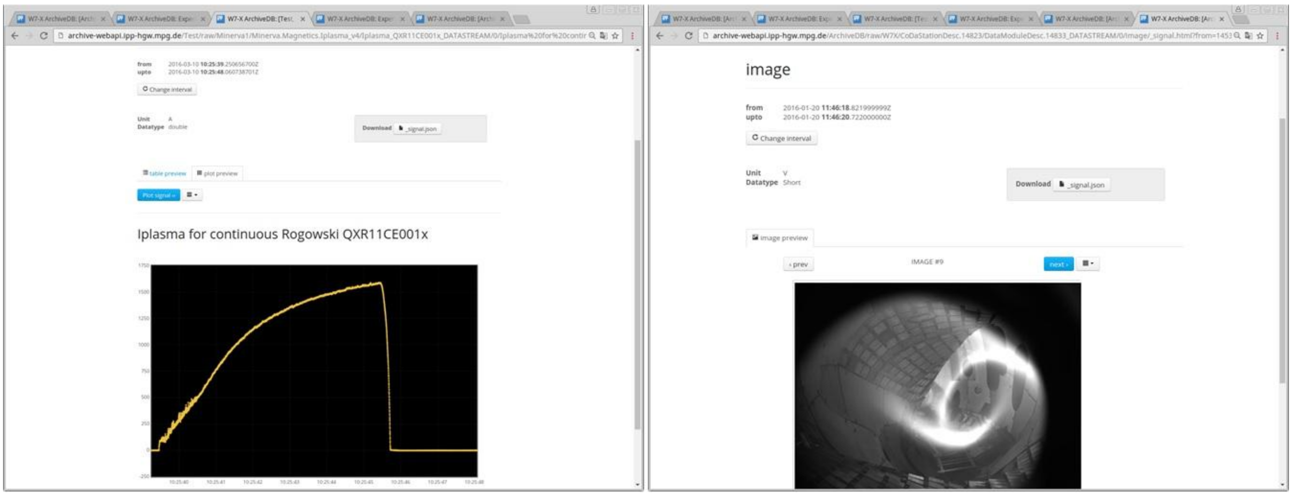

Fig. 2 The signal websites provides previews of the archived data, displayed as a plot (left) or as an image slide show (right).

5. Implementation, Performance & Scalability

The web service was implemented in Java using Jersey (JAX-RS framework [17]) on top of the Java libraries of W7-X Archive database. The Bootstrap framework [18] was used to create a pleasant and consistent design of the HTML pages.

The long discharge times and the evolving development of new diagnostic systems lead to a growing data amount. This makes performance and scaling of the system a key challenge. The stateless nature of the REST architecture style is making horizontal scaling a straightforward task. Several servers were added in the first operational phase to meet the growing user demand: starting with a single machine, this number was quickly extended to 8 servers. The distribution of requests between these servers was done by DNS round-robin. The reading performance depends on the data type and the requested time interval. It ranges from milliseconds for a single JPEG image and goes up to several seconds for receiving some seconds of 1 MHz ADC data as JSON. For a faster preview of large data sets, the user can apply a server-side down sampling.

6. User Experience

Starting with the machine commissioning phase in April 2015, the WEB API was used for monitoring of relevant status information (e.g. pressure, temperature, coil currents) from desktop applications and on the video wall in the control room. Since the first magnetic flux surface measurements, the members of the W7-X team are also using the Web API extensively for reading of data from diagnostic systems.

The Web API is used for inspecting experiment data via web browser or in self-written programs, used for visualization and analysis. Software is written in a wide range of programming languages, such as C, C++, Java, Python, MATLAB, IDL, and LabVIEW. Archive Web API is also used for writing new data to the archive.

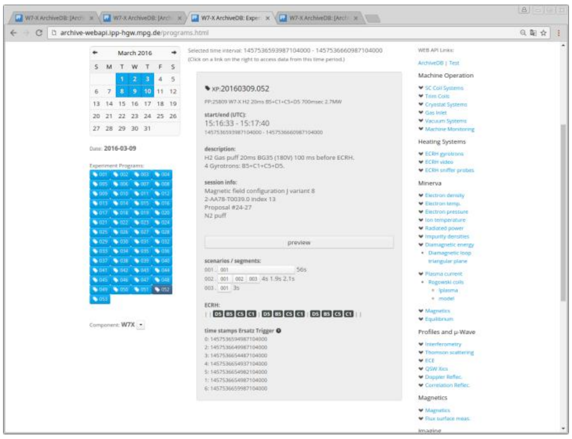

Fig. 3 Programs page: a website shows the experiment program log, containing information about W7-X discharges.

In the first operational phase, several diagnostic systems were still under commissioning. These systems started their local data acquisition by central triggers. The data was either uploaded during the discharge or was stored locally for a later upload via Archive Web API. The imported raw data from these diagnostics contribute ca. 7 TB, which is nearly half of the archived data volume of OP1.1. In addition the Archive Web API is also used for the writing of processed data: starting during the operational phase, the W7-X team is analyzing the data from the ca. 1000 discharges of OP1.1.

Acknowledgments

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

References

[1] T. Klinger: First operation of Wendelstein 7-X, this issue

[2] Ch. Hennig, et al., ArchiveDB—Scientific and technical data archive for Wendelstein 7-X, Fusion Eng. Des. (2016), http://dx.doi.org/10.1016/j.fusengdes.2016.05.026

[3] J. Schacht, et al., A trigger-time-event system for the W7-X experiment, Fusion Eng. Des., 60 (2002) 373–379

[4] R. König et al., 2015 The Set Of Diagnostics For The First Operation Campaign Of The Wendelstein 7-X Stellarator JINST 10 P10002

[5] R. Daigneau: Service Design Patterns, Addison Wesley, (2012)

[6] R.T. Fielding: Architectural Styles and the Design of Network-Based Software Architectures, Ph.D. thesis University of California (2000)

[7] J. Svensson, et al.: Integrating diagnostic data analysis for W7-AS using Bayesian graphical models, Rev. Sci. Instrum., 75 (2004) 4219- 4221

[8] T. Bluhm, et al.: A service based interface for scientific data retrieval, Fusion Eng. Des., 85 (2010), 579-582

[9] J.G. Krom, et al.: The TEC Web-Umbrella, Fusion Eng. Des., 60 (2002) 475–480

[10] D.G. Pretty, et al.: H1DS: A new web-based data access system, Fusion Eng. Des., 89 (2013) 731-735

[11] S.A. Bozhenkov, et al.: Service oriented architecture for scientific analysis at W7-X. An example of a field line tracer, Fusion Eng. Des., 88 (2013) 2997-3006

[12] Requests (Python Http library): http://docs.python-requests.org/en/master/

[13] JSON: http://www.json.org

[14] HDF5: http://www.hdfgroup.org

[15] A. Spring, et al.: Establishing the Wendelstein 7-X steady state control and data acquisition system during the first operation phase, this issue

[16] HAL JSON Specification: http://stateless.co/hal_specification.html

[17] Jersey JAX-RS framework: https://jersey.java.net/

[18] Bootstrap framework: http://getbootstrap.com/