ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-M03BC006

WEB TECHNOLOGY ENABLING FAST AND EASY LARGE EXPERIMENTAL FACILITY CONTROL SYSTEM IMPLEMENTATION

W. Zheng, L.Y. Wang, M. Zhang, X.H. Xie, P.L. Zhang, W.J. Ye, H.B. Ma, International Joint Research Laboratory of Magnetic Confinement Fusion and Plasma Physics, State Key Laboratory of Advanced Electromagnetic Engineering and Technology, School of Electrical and Electronic Engineering, Huazhong University of Science and Technology, Wuhan, China

Abstract

Large experimental facilities are essential for pushing the frontier of fundamental research. Control system is the key for smooth operation for Large experimental facilities. Recently many new types of facilities have emerged, especially in fusion community, new machines with completely different designs are being built. They are not as mature as accelerators. They need flexible control systems to accommodate frequent changes in hardware and experiment workflow. The ability to quickly integrate new device and sub-systems into the control system as well as to easily adopt new operation modes important requirements for the control system. Here we present a control system framework that is built with standard web technology. The key is using HTTP RESTful web API as the fundamental protocol for maximum interoperability. This enables it to be integrated into the already well developed ecosystem of web technology. Many existing tools can be integrated with no or little development. Such as InfluxDB can be used as the archiver, NodeRed can be used as the scripter and docker can be used for quick deployment. It even made integration of in house developed embedded devices much easier. In this paper, we will present the capability of this control system framework, as well as a control system for field revers configuration fusion experiment facility implemented with it.

INTRODUCTIOIN

Large experimental facilities play a crucial role in advancing fundamental research, requiring sophisticated control systems to ensure smooth operations. One key characteristic of as large experimental facility control system is its ability to adopt and integrate new systems. This necessity becomes even more evident in emerging fields such as fusion research, where innovative machines with distinct designs are being constructed. There are various control system framework existing in big physics community, each with different characteristics. For fusion community, ITER chose the Experimental physics and industrial control system (EPICS) and Channel Access protocol (CA) as the common language. EPICS has been the common language to the accelerator control system for decades [1]. Now chosen by ITER, it is used by many tokamaks as well [2-5]. It is mature and well supported by the community.

But the technologies used in tokamaks and other experiment facilities are different from those in accelerators. It is not a straight forward job to create EPICS support for equipment used in fusion experiment. Later emerged control system frameworks such as Tango uses object-oriented technique to improve interoperability and flexibility [6-8]. But still, it is hard to have all the equipment in a control system supporting the control system framework that you chose.

Unlike traditional accelerators, many new facilities demand flexible control systems capable of adapting to frequent changes in hardware and experiment workflows. The ability to swiftly integrate new devices and subsystems, coupled with the ease of adopting novel operational modes, are critical requirements for such control systems.

This paper presents a control system framework developed using standard web technology, with HTTP RESTful web API serving as the foundational protocol to enhance interoperability. By leveraging web technology, the framework seamlessly integrates into the well-established web ecosystem, enabling easy integration with various tools and technologies. Existing tools like InfluxDB can be utilized as archivers, NodeRed as scripters, and Docker for rapid deployment. Additionally, the framework simplifies the integration of in-house developed embedded devices. This paper showcases the capabilities of this control system framework and its application in a field-reversed configuration fusion experiment facility.

The first section of this paper delves into the rationale behind the incorporation of web technology in control systems, highlighting its transformative potential [9]. The second part talks about a toolkit called CFET2 that is used to build a Web based control system. The 3rd part is about using CFET2 with existing web related technologies. Then in section 4 a few real world applications are presented.

WEB TECHNOLOGIES FOR CONTROL SYSTEMS

Web technologies is an important part of our internet life. We keep using it every day. Web technologies not only power the web site. Today from mobile apps, online games, to smart sensor and IoT application, web plays a big role in them. Many devices have embedded web servers, and many client apps is running in browsers. They communicate using HTTP. The foundation of web technology is HTTP, HTML, and browser.

At its core, the web embodies the essence of interoperability. It serves as a unifying platform where diverse technologies seamlessly converge, facilitating the exchange of information across disparate systems and devices. One of the fundamental building blocks enabling this interoperability is the Hypertext Transfer Protocol (HTTP). HTTP operates as a text-based request and response protocol, cre-

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-MO3BC006

ating a transparent channel for communication. This transparency ensures that data can be transmitted and received in a universally understandable format, transcending the barriers of varying hardware and software configurations. HTTP can be regarded as the common language for communication.

Moreover, the markup language that defines the structure of web content, known as Hypertext Markup Language (HTML), enjoys widespread support across all major web browsers. This ubiquity ensures that information presented in HTML format is accessible and comprehensible to users regardless of the browser they employ. The universality of HTML further solidifies the web's interoperability, allowing seamless dissemination and consumption of content across diverse browsing platforms. HTML can be regarded as the common language for visualization.

Modern browsers also run programs like JavaScript script to help user interacting with the content and the server. Browser standard is a widely accepted. It can be regarded as a common language for interact. With all those common language, one can easily think about using it in control system, since control system is all about different system being integrated using one common language [10]. These standards are widely used by all kinds of devices and platforms. So, bringing web technologies to control system may improve the interoperability.

The imperative to design control systems using web technology stems from the pivotal concept of interoperability. In the realm of experimental facilities, a multitude of disparate elements must seamlessly converge within the control system framework. These elements, ranging from diverse devices to complex subsystems, necessitate a cohesive integration strategy. Moreover, experimental facilities are dynamic entities, subject to constant evolution and change. The adaptability of the control system to these perpetual modifications is contingent upon its inherent interoperability.

So how do we use web technology I control system? Before delving into the application of web technology in control systems, it is imperative to establish a clear abstraction of the control system's fundamental components. Everyone in this system can be abstracted into this model and understand each other conceptually. This is the basis of interoperability. This The control system discussed here is mainly a SCADA system. At its core, a control system comprises various resources, each serving distinct purposes and functionalities. By defining these resources, we create a structured framework upon which the integration of web technology becomes not only feasible but also highly efficient.

The primary resource in this context is a "**Thing**," which functions as a container encapsulating other resources within the control system. Within the realm of these resources, two essential categories emerge: "**Status**" and "**Config**." The status of a Thing represents information that can be read but not altered, providing a snapshot of the current state of the resource. On the other hand, the configuration (**config**) encompasses parameters and settings that can be modified, allowing for dynamic adjustments within the system.

Additionally, the control system introduces two crucial elements: "**Method**" and "**Event**." A Method signifies an immediate action that can be triggered, inducing specific responses within the control system. Unlike Config, which defines the intended behavior of the, method is like a command, changing the Thing behavior as soon as this resource is accessed. An Event pertains to information related to a particular resource, but instead of being actively sought, this data is pushed to the recipient, providing real-time updates and insights. Those are shown in Fig. 1.

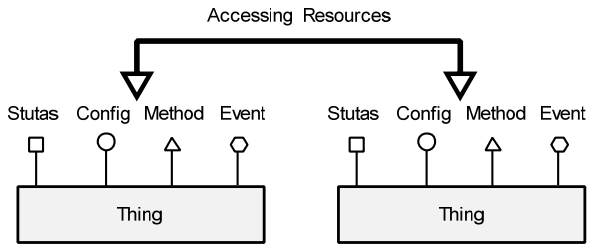

Figure 1: The control system model. Everything in a control system is a resource, and every activity is accessing resources.

To seamlessly integrate these resources into a web-based framework, the control system adopts a systematic approach. By utilizing Uniform Resource Locators (URLs) to pinpoint specific resources, employing HTTP verbs to indicate the access actions (such as GET for reading, PUT for modification, POST for creation, and DELETE for removal), and utilizing HTTP resources to return the results of these actions, the control system becomes inherently compatible with standard web technology.

Table 1: Control System Resource Access Action Map to HTTP Verbs Table

Resource Access Action	HTTP Verb
Get	GET
Set	Put
Invoke	Post
Subscribe	Post

This abstraction not only provides a foundational understanding of the control system's components but also lays the groundwork for implementing a control system using web technology. Through the judicious application of URLs, HTTP verbs, and resources, the control system gains the capability to interact seamlessly with diverse resources, enabling a robust and versatile framework that aligns seamlessly with the interoperable nature of the web.

Here in the Fig. 2 you can see an example of accessing a Status from a web based control system. Suppose you want to know the rpm of an pulse generator. You must first know the URL to that resources. Here we have a fictional URL : *http://pulsegenerator.powersystem.local/motor1/rpm* . The current speed of the motor is a status, so you use the Get verb. The response is shown in Fig. 2. The response is

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-MO3BC006

called a sample, which is an object generated by accessing the resources. The most important value in the sample is the sample value field. But, instead of just the value, it also contains many other useful information called hyper media. Just like hyperlinks on a web page, those hyper media data tells you how to interact with this sample, like its parent and children, how to visualized this sample.

```
{
    "CFET2CORE_SAMPLE_REOUSRCETYPE":1,
    "CFET2CORE_SAMPLE_VAL":590,
    "CFET2CORE_SAMPLE_PATH":"/moter1/rpm",
    "CFET2CORE_SAMPLE_ISREMOTE":false,
    "CFET2CORE_SAMPLE_ISVALID":true,
    "ResourceType":"Status",
    "DisplayType":"Gauge",
    "Unit":"rpm",
    "Action":{
        "get":{
            "Parameters":{

            },
            "OutputType":"Double"
        }
    },
    "ParentPath":"/motor1",
    "ChildrenPath":[

    ]
}
```

Figure 2: a response of a access to a status in web based control system.

CFET2: A WEB-BASED CONTROL SYSTEM FRAMEWORK

In the landscape of web-based control systems, the flexibility and adaptability of this approach are further highlighted by the ease with which it can be implemented using a myriad of existing web frameworks. Countless options abound, allowing developers to construct sophisticated control systems swiftly and efficiently. To exemplify this concept, a specialized control system framework, aptly named CFET2, has been developed for seamless integration within experimental facilities.

CFET2 Framework Overview

A toolkit like EPICS is needed to practically build control system applications. We developed the Control system Framework for Experimental Devices Toolkit (CFET). The CFET2 is implemented using .NET standard which supports major Linux distros, Mac OS and Windows. The design aims to let the control system engineer to focus on the control functions, and ignore the web completely.

At the heart of CFET2 lies a console application, CFET2app, designed akin to an Experimental Physics and Industrial Control System Input/Output Controller (EPICS IOC). This console application serves as the backbone of the control system, facilitating the management and interaction with diverse resources encapsulated within the experimental setup.

Complementing CFET2app is WidgetUI, a Single Page Application meticulously crafted for Human-Machine Interface (HMI) design. Built using the robust VUE.js framework, WidgetUI empowers users with an intuitive and interactive interface. This interface not only provides real-time insights into the status and configuration of resources but also facilitates dynamic adjustments through seamless user interactions.

How to make a CFET2 Thing

In the realm of CFET2, all resources are encapsulated within entities known as "Things." The process of defining a Thing is remarkably straightforward, underscoring the simplicity and elegance of the CFET2 framework. By leveraging the robust capabilities of C# programming language, developers can effortlessly create a Thing by implementing specific logic that facilitates communication with hardware components. Here's a step-by-step guide to creating a Thing in CFET2:

1. Begin by crafting a C# class that encapsulates the desired logic for interacting with hardware components. This is shown in Fig. 3.
2. To transform the methods and properties within the class into CFET2 resources accessible via HTTP, developers need to apply the CFET2 Resource Attribute. By strategically placing these attributes, developers indicate which methods and properties are exposed as CFET2 resources.
3. Once the methods and properties are adorned with CFET2 Resource Attributes, they become readily accessible as CFET2 resources. Through HTTP requests, other entities within the CFET2 ecosystem can effortlessly access these resources.

Inside the Thing, it can access other resource in the control system by using a hub object in the CFET2app. When accessing other resources, the Thing does not need to know anything about web, expect the URL to the resources.

```
public class NiDaqCard : Thing
{
    [CfetStatus(Name = "Data")] //a CFET Status resource with name "data"
    public double[] GetData(int channel)
    {
        return ReadDataFromChannel(channel);
    }
}

public class MdsUploader : Thing
{
    [CfetMethod] //this is a CFET Method resource
    public void Upload()
    {
        //assume the above DaqCard is mounted on a remote DaqHost
        var data = Hub.Get("http://DaqHost:8080/card1/data");
        uploadToMdsServer(data, shot, tag);
    }
}
```

Figure 3: The pseudocode illustrating how to make a thing. In the second method it gets data from a resource located possibly on a remote controller using the hub object.

How to make a CFET2 HMI

In the realm of CFET2, constructing dynamic and interactive Human-Machine Interfaces (HMIs) becomes a seamless process. The framework empowers users with a user-friendly interface, allowing effortless creation of HMIs through a guided demonstration. Here's a walkthrough of the process.

The process commences by adding widgets to the HMI canvas. Users can intuitively select from a variety of widgets, tailoring the interface to their specific requirements. These widgets serve as visual representations of the underlying CFET resources. By associating each widget with a corresponding CFET resource URL, users establish a direct link between the graphical interface and the control system's resources.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-MO3BC006

Through the specified URLs, the HMI establishes real-time connections with the underlying CFET2 framework. This connectivity enables users to monitor and control the associated resources directly from the graphical interface. The visual representation of data ensures an intuitive understanding of the system's status, facilitating informed decision-making.

An intriguing feature showcased in the demonstration is the State Machine widget. This widget requires a URL pointing to a specific state machine Thing within the CFET2 framework. Upon inputting the URL, the State Machine widget autonomously loads the state machine's definition. It dynamically generates and displays the corresponding state diagram, providing users with a visual representation of the system's states and transitions. The State Machine widget dynamically highlights the current state.

EXAMPLE OF IMPORTANT THINGS

State Machine Thing

Within the CFET2app ecosystem, the State Machine Thing stands as a cornerstone, playing a pivotal role in the orchestration of every subsystem within our facility. State machines serve as the linchpin, governing the behavior of diverse components in our experimental setup. One of the remarkable aspects of CFET2app is its intuitive approach to handling state machines, making it one of the most widely used features in our facility. Here's a closer look at how CFET2app simplifies complex state machine management.

Creating a state machine within CFET2app is streamlined into a straightforward process. Developers can define state machines using just three files. The first file establishes aliases for CFET resource URLs, enhancing clarity and simplifying resource identification. The second file delineates various states and transitions conditions within the state machine. Lastly, an optional file specifies corresponding actions to be taken when a transition is triggered. These configuration files are designed for human readability, ensuring accessibility and ease of understanding. Additionally, the state machine thing supports YAML format for state definitions, further enhancing readability and simplicity. There is no need to compile, and you can even change that state machine definition without restart the running CFET2app.

The state machine widget in the WidgetUI will automatically fetch the state machine definition when given the url to the thing. Using the defined state machine configurations, the State Machine widget generates dynamic state diagrams. These diagrams provide an instantaneous visual representation of the current state of our experiment, enabling operators and researchers to monitor the system's behavior in real-time.

EPICS Bridget Thing

EPICS is still the most popular SCADA software framework in experiment facility control system. Being able to be integrated into an existing EPICS based control system or integrates EPICS compatible device into an CFET2 based web control system is a must have ability. CFET2 ships with a EPICS Bridge Thing which does just that. Here's how the EPICS Bridge Thing in CFET2app is transforming cross-system compatibility.

The EPICS Bridge Thing acts as a bridge, enabling EPICS clients to seamlessly access CFET2 resources. This is done with the help of a soft EPICS IOC. To make this simple, we provide a preconfigured EPICS docker image. You just put the PV names which will be connected to CFET2 resources in the db file. You spin up the docker image, and the CFET2app containing the EPICS bridge thing, the EPICS bridge thing will automatically map the corresponding CFET2 resource into the PVs in the EPICS docker container.

For the other way around, the EPICS bridge thing has a EPICS client, which redirect the request to CFET2 resource to the corresponding PVs.

LEVERAGE ON EXISTING TECHNOLOGY ENABLING FAST AND EASY CONTROL SYSTEM IMPLEMENTATION

Effortless Deployment and Integration with Docker and CFET2apps

In the realm of modern control systems, agility and scalability are paramount. CFET2apps leverages the power of Docker, enabling rapid deployment and seamless integration of control system components. By adopting a microservices architecture, CFET2apps simplifies the deployment process, fostering a modular and scalable environment for control system management.

Docker, renowned for its containerization technology, serves as the backbone of CFET2apps' deployment strategy. There is no need to install, compile or make the environment for the CFET. Just get the docker image and prepare the configuration for things, then everything just runs.

Docker Compose acts as the orchestrator, simplifying the deployment and management of multiple CFET2apps. With a straightforward Docker Compose file, users can modify volume paths, allowing for easy organization and configuration of CFET2apps. The only modification needed is the thing configuration path. Mount a host folder containing the thing configuration to the docker volume you are done. The tree structure within the thing configure folder mirrors the organizational hierarchy of a control system. Each thing, represented as a folder, contains its respective configuration files. This intuitive structure streamlines the management of complex configurations, ensuring that settings are neatly organized and easily accessible.

As mentioned above, CFET2 also provide a docker image for EPICS soft IOC. This ensures an easy and smooth transition for control systems already utilizing EPICS, allowing for an incremental adoption approach without disrupting existing workflows.

Simplified Control System Development with NodeRed Integration in CFET2

In the realm of rapid control system development, NodeRED stands out as a versatile and powerful tool. CFET2

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-MO3BC006

harnesses the simplicity and efficiency of HTTP, enabling effortless integration with Node-RED. This synergy between CFET2 and Node-RED revolutionizes control system automation and visualization.

Utilizing the HTTP Request nodes within Node-RED, developers can seamlessly access CFET2 resources without the need for complex coding. Node-RED serves as a powerful platform for creating automation logics for the control system. By leveraging the intuitive drag-and-drop interface, developers can design intricate automation flows without writing extensive lines of code. The integration with CFET2 resources ensures that these flows can directly control and monitor the system, allowing for rapid prototyping and testing of automation logics.

Node-RED's visualization capabilities empower developers to construct dynamic Human-Machine Interface (HMI) panels effortlessly. The Node-RED Dashboard enables the creation of interactive and responsive interfaces, allowing users to visualize real-time data and control system parameters. By integrating CFET2 resources, developers can build comprehensive HMI panels, providing operators with a rich and intuitive environment for system monitoring and control.

Enhancing Event Distribution with MQTT

In the rapidly evolving landscape of IoT applications, MQTT (Message Queuing Telemetry Transport) has emerged as a pivotal communication protocol. Its widespread adoption and versatility have made it an industry standard. In recognizing the importance of MQTT in the IoT realm, CFET2 has seamlessly integrated MQTT event support, enhancing event distribution and enabling effortless integration with various applications.

Many existing applications and devices already support MQTT, making it a natural choice for CFET2's event distribution system. This compatibility ensures seamless integration with a wide array of IoT devices and platforms, allowing CFET2 to communicate effortlessly with other components within an IoT ecosystem.

One of the key benefits of MQTT integration lies in its simplicity. Subscribing to CFET2 events through MQTT involves subscribing to specific MQTT topics. This straightforward process allows developers familiar with MQTT to effortlessly receive CFET2 events without needing to delve into intricate CFET2-specific procedures.

CFET2's flexibility shines through as it allows both MQTT and WebSocket events to coexist within the same system. For developers creating CFET2 Things, this means having the freedom to choose the event distribution protocol that best suits their requirements. Whether it's the WebSocket-based communication tailored for WidgetUI in browsers or MQTT for broader existing software and hardware integration, CFET2 offers the versatility needed to accommodate diverse use cases.

The integration of MQTT event support in CFET2 harmonizes perfectly with Node-RED. Node-RED's native support for MQTT enables seamless communication between CFET2 resources and Node-RED flows. This interoperability further expands the possibilities.

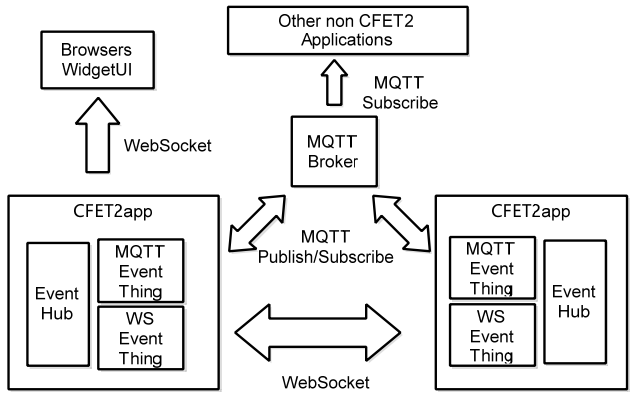

Figure 4: How event is distributed in CFET2 applications.

Efficient Data Archiving and Visualization with InfluxDB

In the realm of data archiving, InfluxDB stands out as a robust, open-source time series database, providing unparalleled performance and versatility. In the context of CFET2, InfluxDB serves as the backbone of the archiving system, ensuring seamless storage, retrieval, and visualization of time series data.

InfluxDB's design specifically for time series data makes it an ideal choice for CFET2's archiving needs. InfluxDB comes equipped with a wealth of tools designed to simplify data management. Its intuitive dashboard builder facilitates the creation of interactive and informative HMIs, allowing users to visualize archived data seamlessly. Additionally, InfluxDB offers robust search capabilities, enabling users to quickly locate specific data points within the archived datasets.

InfluxDB's flexibility allows it to seamlessly integrate with CFET2's data sources, supporting both HTTP and MQTT protocols out of the box. InfluxDB's ecosystem includes Telegraf, an application designed for automated data collection. Telegraf effortlessly fetches data from various sources, including CFET2's HTTP and MQTT endpoints, and deposits this data directly into InfluxDB.

Enhancing Control System Flexibility with Multicast DNS

In the dynamic landscape of web-based control systems, resolving resources efficiently and seamlessly is paramount. CFET2 adopts Multicast DNS (mDNS) as a fundamental solution, eliminates the dependency on static IP addresses, ensuring adaptability and effortless resource resolution.

EPICS PVs (Process Variables) are host agnostic, utilizing UDP to locate the host holding the PV. In a web-based control system powered by CFET2, mDNS allowing CFET2 resources to be resolve dynamically. This host-agnostic approach ensures that CFET2 resources can be located without manual configuration.

CFET2 leverages Avahi, a powerful zero-configuration networking tool, to manage domain names effectively. With Avahi, web-based control systems can assign multiple domain names to a single device. This flexibility proves invaluable, especially in scenarios where multiple

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-MO3BC006

CFET2app instances run on the same host initially but are later separated onto different hosts. By reassigning domain names, the control system adapts seamlessly without disrupting ongoing operations.

REAL WORLD APPLICATIONS

Below are some real-world applications of CFET. It has been used in China's 3rd largest tokamak. It is used to build the China's largest Field Reverse Configuration fusion experiment control system. Also one industrial application is introduced.

J-TEXT Tokamak

The J-TEXT Tokamak, as the third-largest Tokamak in China, relies on the established EPICS CA protocol for its control system. But many newly developed system are based on CFET2. One of the remarkable achievements lies in CFET2's ability to seamlessly bridge the gap between legacy EPICS CA-based systems and modern, CFET2-based technologies [10-11].

HUST FRC Fusion Experiment

In the heart of China's largest Field Reverse Configuration (FRC) fusion experiment at HUST (Huazhong University of Science and Technology), control systems is primarily driven by CFET2.

At HUST FRC (HFRC), CFET2's HTTP-based control system serves as the cornerstone, unifying diverse components under a common communication protocol. Whether it's the core control infrastructure, all the pulse power supplies, or even alarm controller with ESP32 micro-controllers, the integration via HTTP ensures seamless communication, enabling real-time data exchange and command execution.

CFET2's HTTP-based approach not only addresses current integration needs but also provides a scalable and future-ready architecture. As new technologies emerge or the facility expands, integrating them into the existing CFET2 framework is a streamlined process [12].

Smart Milling with CFET2

Utilizing CFET2, data acquisition from vibration sensors on ball mills enabled predictive maintenance strategies that redefine operational reliability. CFET2's ability to feed this data seamlessly to the AI backend ensures that the predictive analytics models receive a constant stream of real-time data, enhancing the accuracy of predictions. The integration of CFET2 with AI back ends enables proactive machine failure prevention, enhancing operational efficiency and minimizing disruptions.

CONCLUSION

In conclusion, the adoption of web technology, exemplified by CFET2, has ushered in a new era of interoperability and efficiency in control systems. By embracing web-based protocols like HTTP and MQTT, control systems can seamlessly integrate a myriad of components, fostering a harmonious ecosystem. Existing technologies like docker, InfluxDB, Node-RED, mDNS and many more can be seamlessly integrated. This interoperability not only enhances communication between diverse systems but also facilitates the incorporation of mature solutions from various fields into the realm of large experiments and scientific research.

ACKNOWLEDGMENTS

The authors wish to thank all the members in J-TEXT laboratory. This work is supported by the National Magnetic Confinement Fusion Science Program (No. 2017YFE0301803) and by National Natural Science Foundation of China (Nos. 11605068).

REFERENCES

[1] Experimental Physics and Industrial Control System Home Page, http://www.aps.anl.gov/epics/

[2] A. Wallander, L. Abadie, H. Dave, F. Di Maio, H. K. Gulati, C. Hansalia, *et al*., "ITER instrumentation and control—Status and plans," *Fusion Eng. and Des*., vol. 85, pp. 529-534, 2010. doi:10.1016/j.fusengdes.2010.01.011

[3] K. H. Kim, C. J. Ju, M. K. Kim, M. K. Park, J. W. Choi, M. C. Kyum, *et al*., "The KSTAR integrated control system based on EPICS," *Fusion Eng. and Des*., vol. 81, pp. 1829-1833, 2006. doi:10.1016/j.fusengdes.2006.04.026

[4] V. Vitale, C. Centioli, F. Di Maio, M. Napolitano, M. Panella, M. Rojo, *et al*., "FTU toroidal magnet power supply slow control using ITER CODAC Core System," *Fusion Eng. and Des*., vol. 87, pp. 2012-2015, 2012. doi:10.1016/j.fusengdes.2012.05.006

[5] W. Zheng, M. Zhang, J. Zhang, G. Zhuang, Y. He, and T. Ding, "The J-TEXT CODAC system design and implementation," *Fusion Eng. and Des*., vol. 89, pp. 600-603, 2014. doi:10.1016/j.fusengdes.2014.03.048

[6] A. Götz, *et al*., "The TANGO Controls Collaboration in 2015," in *Proc ICALEPCS'15,* Melbourne, Australia, Oct. 2015, pp. 585-588. doi:10.18429/JACoW-ICALEPCS2015-WEA3O01

[7] R. Bourtembourg, J. M. Chaize, T. Coutinho, A. Götz, V. Michel, J. L. Pons, *et al*., "TANGO Kernel Development Status," in *Proc. ICALEPCS' 17*, Barcelona, Spain, Oct. 2017, pp. 27-33. doi:10.18429/JACoW-ICALEPCS2017-MOBPL02

[8] T. Matsumoto, Y. Furukawa, and Y. Hamada, "MADOCA II DATA COLLECTION FRAMEWORK FOR SPring-8," in *Proc. ICALEPCS' 17,* Barcelona, Spain, Oct. 2017, pp. 39-44. doi:10.18429/JACoW-ICALEPCS2017-MOBPL04

[9] W. Zheng, N. Fu, S. Li, Y. Wang, F. Wu,M. Zhang, "Designing a Control System for Large Experimental Devices Using Web Technology, " in Proc. *ICALEPCS'19*, New York, NY, USA, Oct. 2019, pp. 28. doi:10.18429/JACoW-ICALEPCS2019-MOBPP02

[10] W. Zheng, Y. Wang, M. Zhang, Z. Yang,Y. Pan. "Designing CODAC system for tokamaks using web technology[J] ". *Fusion Eng. and Des*., 2019, 146: 2379-2383. doi:10.1088/1741-4326/ab1a72

[11] Y. Liang, N. C. Wang, Y. H. Ding, Z. Y. Chen, Z. P. Chen, Z. J. Yang, *et al*. "Overview of the recent experimental research on the J-TEXT tokamak[J]". *Nucl. Fusion*, 2019, 59(11): 112016

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-M03BC006

[12] W. Zheng, F. Wu, M. Zhang, B. Rao, Y. Yang, X. Xie, *et al*. "Design and implement of control system for HUST field-reversed configuration device[J] ". *Fusion Eng. and Des*., 2022, 180: 113138. doi:10.1016/j.fusengdes.2022.113138