ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV007

FAST CREATION OF CONTROL AND MONITOR GRAPHICAL USER INTERFACE FOR PEPC OF LASER FUSION FACILITY BASED ON ICSFF

Li Li†, Jun Luo, Zhigao Ni
Institute of Computer Application (CAEP) China Academy of Engineering Physics, MianYang, China

Abstract

Plasma electrode Pockels cell (PEPC) is the key unit of the multi-pass amplify system in laser fusion facility, whether the PEPC is effective determined the success rate of the facility experiment directly. The operator needs to conduct remote control and monitor during the facility is running, also can automatically judge whether the pulse discharge waveform is regular online. In traditional design and realization of control and monitor software, the fixed GUI cannot adapt frequent changes of the experiment requirements, and it will consume time and resources once more. We have designed a software framework (ICSFF) that loads all GUI widget elements related to control and monitor into board through plug-ins, and then by setting the respective properties, data source and built-in script of each widget achieve patterns like point control, flow control and other complex combined control, can also achieve data acquisition and varied display effects. It allows the operator drag and drop widget freely and configure the widget properties through the interface in a non-programming mode to quickly build the GUI they need. It not only apply to PEPC in facility, but also to other system in the same facility. ICSFF supports Tango control system right now, and more control systems will be supported in the future.

OVERVIEW

The control system of large-scale scientific experiment equipment is usually a heterogeneous system, including multiple subsystems with special functions, and PEPC of laser fusion facility is also a complex system, which contains a variety of hardware devices with different communication protocol interfaces. The typical large-scale control system like Figure 1. In the past, when developing remote integrated control system, we needed to design different interfaces for each different hardware protocol and design different operating interfaces for users. However, the solidified interface and interface design have great limitations. They are neither adapted to the frequently changing user operation requirements nor to the frequently changing hardware interfaces of experimental systems. Designers often need to spend a lot of time on the repetitive work of modifying and debugging code.

In order to change this situation, we have designed a software framework (ICSFF) that allows users to simply drag and drop and configure control properties through the interface in non-programming mode, and then quickly build the GUI what they need. This is not only applicable to the remote integrated control of the PEPC in facility, but also applicable to other systems in the facility.

† lili_top@163.com

FRAMEWORK INTRODUCTION

As mentioned above, we have developed a general control system framework for the integrated control system design of this large-scale facility, and provide editable functional modules for monitoring, control, data acquisition and storage functions with general requirements. The ICSFF software framework is a component-based distributed control system, mainly for non-real-time systems.

The software framework is suitable for the following network systems, and hardware devices are connected to the system through the network. The software of the control layer is deployed on the server or embedded controller to provide the control and data interaction of the hardware device. The monitoring layer software is deployed on the console computer to provide integrated operation, monitoring and other functions.

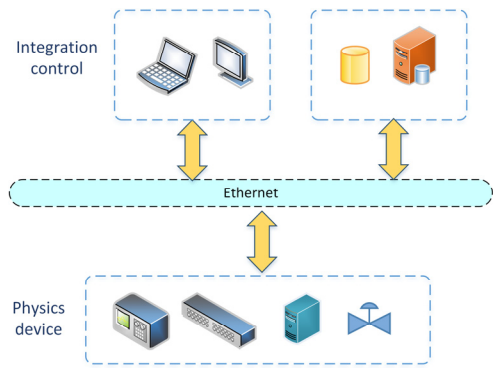

Figure 1: Typical large-scale scientific device control system structure.

The ICSFF software framework is divided into three layers: device service layer, system service layer and integrated monitoring layer [1]. See Figure 2.

The integrated monitoring layer provides a unified integrated operating environment for the control operation of the entire facility, enabling centralized control, monitoring and data management of the entire facility; the integrated monitoring layer can be divided into three different dimensions of integrated control functions according to actual needs: facilities, systems, groups.

The system service layer provides a combined control function for a single system under a bundle group, which is a large-scale control function aggregation of the device service layer, and this layer provides parameter delivery and experimental data archiving.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV007

The device service layer provides software mapping for independent devices to implement device control, status acquisition, device diagnostics, and self-test, using device drivers. Therefore, all heterogeneous devices implement the unification of software interfaces and provide consistent standard services on network protocols [2].

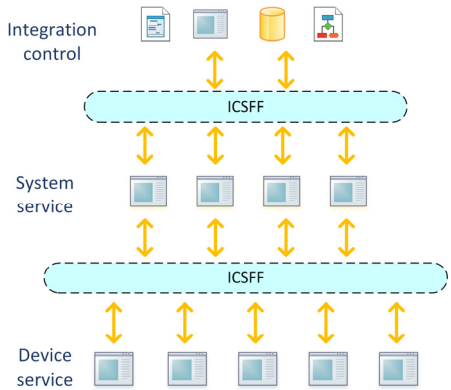

Figure 2: ICSFF software framework.

COMPONENT-BASED CONTROL AND MONITORING GUI DESIGN

Here we mainly discuss the GUI design of the software framework. To make the software framework have a flexible operation mode that can be configured at runtime, it is need to solve the problems of abstraction of the UI layer, separation of data and UI, and editability at runtime.

Direct interaction with users requires a variety of functional modules. Currently, our software framework provides login module, log module, panel module, service management module, scheme module, element module, and attribute editing module. A complete user integrated monitoring system can be built from these modules.

Element Module and Attribute Editing Module

Let's start with the smallest element module. At present, our element library contains 23 optional controls, including flow nodes, buttons, text edit boxes, multiple selection boxes, text displays, LCD digital displays, status lights, progress bars, Ring marquee, pump, valve, pipe, image, curve, time and other types. Currently supports Qt controls, Iocomp controls and Quc controls.

Each type of element contains its own different attributes, which include styles, data access interfaces, and data display styles. element properties can be edited. Users can edit the basic styles of the element such as the size, color, border, and font according to the needs of the display. Element style editing is shown in Figure 3, which shows the style editing interface of several element.

The purpose of separating the data I/O interfaces separately is to separate the data from the element, so that the software framework can adapt to more requirement. To configure the data in and out interface for each element, whether it is a button to send a command, or a status light, pump, etc. displayed by receiving data, you only need to select the interface provided in the corresponding service from the left side of Figure 4, and pass on the right side. Edit the script language file to realize the connection between the control and the data. The script language can also provide more complex calculations, so that the received raw data can be processed through calculations and then displayed on the element.

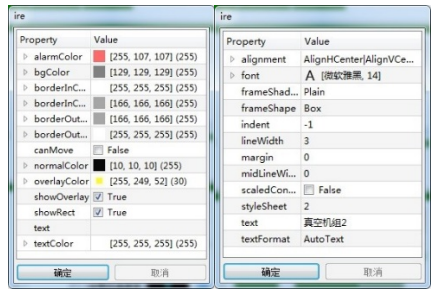

Figure 3: Element property editing interface.

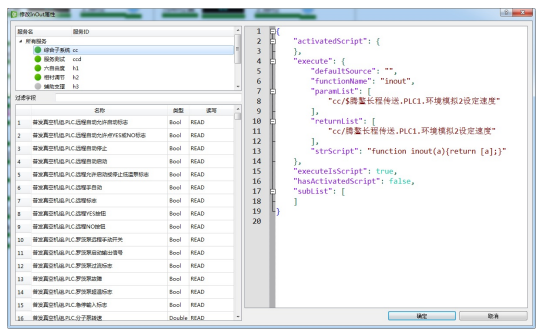

Figure 4: Data interface property configuration.

Data display style editing is shown in Figure 5, it realizes that certain controls, such as status light data, will undergo different color changes according to the received data. This is done by configuring keywords and values. Especially when the data types and data meanings defined by different manufacturers are different, users can uniformly set according to their own needs, which is conducive to the unification of the interface without changing the code.

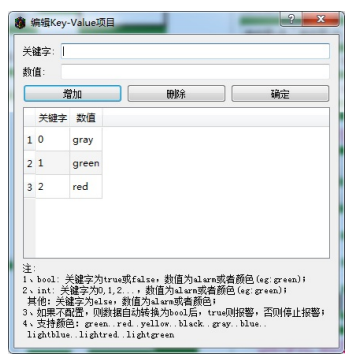

Figure 5: Data display style attribute configuration.

The data flow of the element is shown in Figure 6. The data input and the data received in the interface are all translated through the script language, and then sent and received by the FRTDB interface, which realizes the separation of the control and the data.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV007

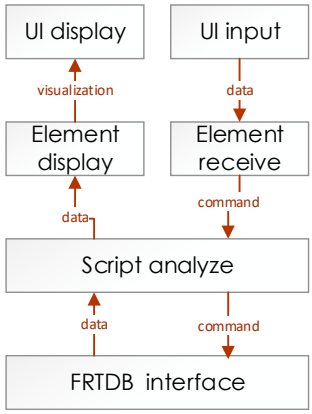

Figure 6: Data flow through element.

Panel Module and Scheme Module

A scheme is a container of controls, and a scheme can become a system interface with independent functions. The panel is a container for loading solutions. Programs provide users with the ability to build any system or subsystem, and programs can be nested. So, users can create multiple schemes to facilitate the use and management of the control system.

The panel provides program editing and running functions, it shown in Figure 7. User drags the elements into the schemes, and arranges the elements in the editing function provided by the panel. In order to realize the editability of the program, all the contents of the program need to be stored persistently, including the general attributes such as the position, size, and color of the control, as well as the configuration of the data in and out interface. The panel provides functions such as saving scheme to database, saving to file, loading scheme from file, and loading scheme from database. At the same time, it also provides convenient functions such as copying of controls, replacing control data in and out interfaces, so that users can create schemes more quickly.

Figure 7: Panel editing function.

Through the application model of this software framework, our development model has changed, the amount of code is reduced, the focus is more focused on the difficult points, the debugging efficiency is greatly improved, and the operation and maintenance cost are greatly reduced. Figure 8 shows our current working mode: after confirming the requirements with the users, confirming the interface requirements with various manufacturers, and then using the reuse of the ICSFF software framework to build the control system, and directly convert the fully reusable parts such as login, log, panel, etc. Utilize, some reusable parts such as program editing, attribute editing, etc., are changed according to user needs, and some functions and elements that are not available in the framework need to change the framework code or add new elements to meet user needs, and at the same time Reverse enriches the connotation of the software framework. After the iteration of unit testing and integration testing, the control system has completed the basic creation.

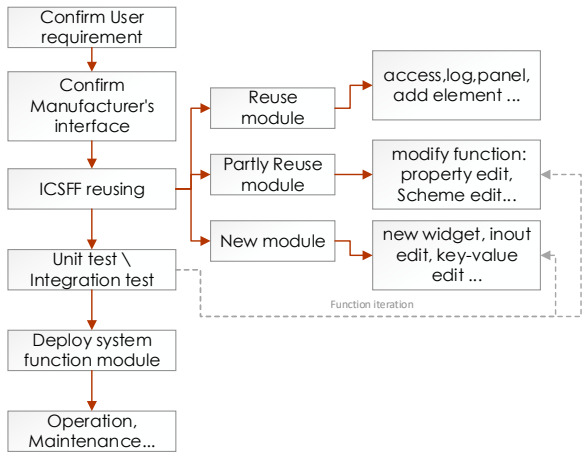

Figure 8: Software framework reuse process.

APPLICATION

The application of ICSFF in the software framework in the plasma switch control system is shown in Figure 9. As described in the previous chapter, the system framework is built through the software framework, and the required controls are added, and the properties and control data in and out of the interface content are edited. Complete the system. Compared with the actual time of our implementation and deployment, the previous system debugging and code modification took 1~2 man/month, while the current model only needs 0.5 man/month to debug and modify.

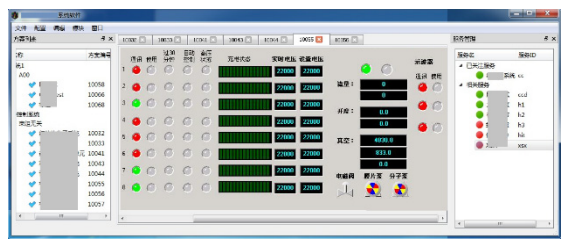

Figure 9: PEPC control system.

CONCLUSION

The ICSFF software framework contains many other components. This article only shows the application in the user interface. The goal of the software framework is to provide an open, scalable and reliable integrated control platform that can provide long-term operation and maintenance capabilities for various large and small systems and scientific experimental devices. In the future, we will conduct more exploration and research on the intelligent direction of the software framework, and add more intelligent modules to promote the current growing demand for data analysis.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-THPV007

REFERENCES

[1] Z. G. Ni *et al*., "The Design of Tango Based Centralized Management Platform for Software Devices", in Proc. *ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 1121-1124. doi:10.18429/JACoW-ICALEPCS2017-THBPL0

[2] Z. G. Ni *et al*., "The Design of Intelligent Integrated Control Software Framework of Facilities for Scientific Experiments", in Proc. *ICALEPCS 2019*, New York, NY, USA, Oct. 2019, pp. 132-136. doi:10.18429/JACoW-ICALEPCS2019-MOMPL007