ISBN: 978-3-95450-208-0 doi:10.18429/JACoW-IPAC2019-THPRB038

# ALARM SYSTEM OF IRFEL AT NSRL*

Xin Chen, Yifan Song, Zhuoxia Shao, Chuan Li, Ke Xuan, Jigang Wang, Gongfa Liu†,
NSRL, USTC, Hefei, Anhui 230029, China

*Abstract*

An InfraRed Free Electron Laser Light (IRFEL) is under commissioning at National Synchrotron Radiation Laboratory (NSRL). The control system of IRFEL is a distributed system based on Experimental Physics and Industrial Control System (EPICS). The alarm system is an essential part of the control system. It is developed based on the software Phoebus. The module named "Alarms" in Phoebus can store states and configuration information of the Process Variable (PV) in the Kafka topics. To meet our requirements, 3 kinds of alarm message distribution applications are developed, i.e. Web-Based GUI, WeChat and SMS. This paper will introduce the alarm system architecture and the implementations of the applications for alarm message distribution.

## INTRODUCTION

An InfraRed Free Electron Lasers (IRFEL) is under commissioning at National Synchrotron Radiation Laboratory (NSRL). It can accelerate electron beam to 60MeV and generate middle-infrared and far-infrared laser [1]. The control system of IRFEL is a distributed system based on Experimental Physics and Industrial Control System (EPICS). EPICS is a set of open source software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as a particle accelerators, telescopes and other large scientific experiments [2].

The alarm system is an essential part of the IRFEL control system. With the continuous development of EPICS, a series of alarm systems have been released, such as the EPICS alarm handler (ALH), the Best Ever Alarm System Toolkit (BEAST) and Phoebus [3–5]. For example, the alarm system of the Hefei Light Source (HLS-II) is developed based on BEAST [6]. However, before the using of Phoebus, it was very hard to manage and integrate a collection of tools to monitor and operate large scale control systems. Phoebus is a framework which has a lot of modules such as "Alarms", "PV table", etc. The module named "Alarms" in Phoebus consists of a user interface and an alarm server which stores states and configuration information of the Process Variable (PV) in the Kafka topics. Kafka is a distributed streaming platform and allows not only publishing and subscribing to streams of records but also storing streams of records in a fault-tolerant durable way [7].

The primary goal of the IRFEL alarm system is to effectively help the relevant staffs take the correct action at the correct time. In China, WeChat is the most widely used mobile messaging app which offers a free instant messaging application service for mobile phone, and SMS is a traditional way of distributing messages. Therefore, we develop the applications to distribute WeChat and SMS messages. Furthermore, in order to remotely query the alarm information, we also develop a Web-Based GUI application.

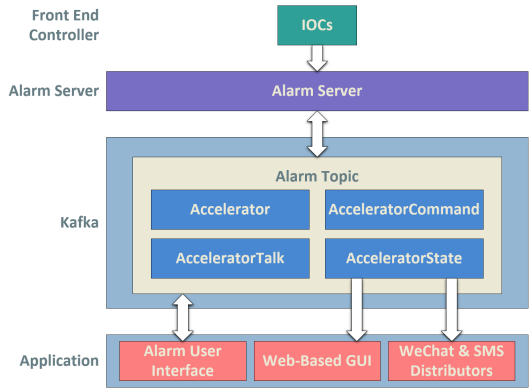



Figure 1: Architecture of the IRFEL alarm system.

## ARCHITECTURE

As shown in Fig. 1, the IRFEL alarm system consists of 4 layers: the Front End Controller layer, the Alarm Server layer, the Kafka layer and the Application layer.

In the Front End Controller layer, there are many Input Output Controller (IOC) applications. When a state of a PV in these IOCs is changed, the IOC will update this information to the Alarm Server layer.

In the IRFEL alarm system, Kafka acts as an integration of a Message-Oriented Middleware (MOM) and a database. As a middleware, Kafka allows publishing and subscribing to streams of records in topics, which can improve the alarm system scalability and flexibility. As a database, Kafka allows storing streams of records in topics in a fault-tolerant durable way, which can prevent the loss of historical data in the alarm system. There are 4 Kafka topics named "Accelerator", "AcceleratorState", "AcceleratorCommand", "AcceleratorTalk". These topics are created with distinct purposes. The configuration information of the alarm system is stored in the Kafka compacted topic named "Accelerator" while the state information is stored in the Kafka compacted topic named "AcceleratorState". "AcceleratorCommand" and "AcceleratorTalk" are employed for communication such as acknowledgement and annunciation between Alarm Server and Alarm User Interface in the module "Alarms". In general, most messages in these topics use the path to the alarm tree item as the key. For example, a message in "AcceleratorState" is shown as follows:

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

ISBN: 978-3-95450-208-0 doi:10.18429/JACoW-IPAC2019-THPRB038

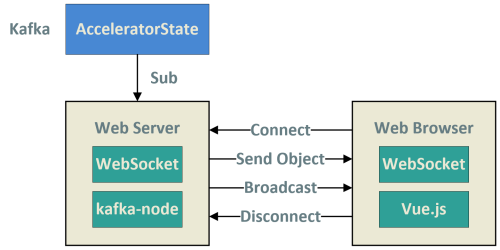



Figure 2: Architecture of the Web-Based GUI.

*/Accelerator/IRFEL/VA/IRFEL:VA:Waveguide1:pressure:ai:*
*{"severity":"MINOR","message":"HIGH_ALARM", "value":"5.4e-06", "time":{"seconds":1537442697, "nano":618896169},"current_severity":"MINOR", "current_message":"HIGH_ALARM"}*

The key of this message is /Accelerator/IRFEL/VA/IRFEL:VA:Waveguide1:pressure:ai and the left is value.

In the Alarm Server layer, Alarm Server connects to all the requested PVs, monitors their state changes and generates alarms, handling acknowledgement, annunciation, latching, and some amount of filtering according to the configuration information in "Accelerator" [4]. For example, when an alarm state of a PV in the IOCs is changed, the alarm information will be posted to Alarm Server through Channel Access (CA) protocol. Alarm Server processes this information via analyzing configuration and previous state information. Then, Alarm Server generates a key-value pair and overwrites it into "AcceleratorState". If the configuration of this PV enable annunciation, Alarm Server will also write a key-value pair into "AcceleratorTalk".

In the application layer, Alarm User Interface is a component of the module "Alarms". Besides displaying alarm information, this interface can also achieve other functions such as acknowledgement and annunciation by subscribing to "AcceleratorCommand" and "AcceleratorTalk". To meet our requirements, we develop 3 applications for alarm message distribution: Web-Based GUI, WeChat and SMS messages distributors. Web-Based GUI is a user interface providing a way to remotely query the alarm information. WeChat and SMS messages distributors are integrated into a Python program. The basic functionality of this program is to distribute alarm or recovery messages via WeChat and SMS.

## SOFTWARE DEVELOPMENT

### *Web-Based GUI*

In order to remotely query the alarm information, we develop Web-Based GUI based on browser/server (B/S). The messages in Kafka is not available to a web browser directly. In order to get the states information, we develop a web server for the browser in NodeJS. As shown in Fig. 2, the web server subscribes to "AcceleratorState". Because the messages in "AcceleratorState" fits the tree structure, the server can consume and store these messages as an object in

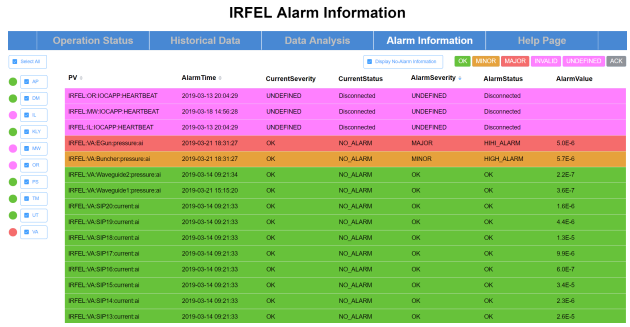



Figure 3: The interface of Web-Based GUI.

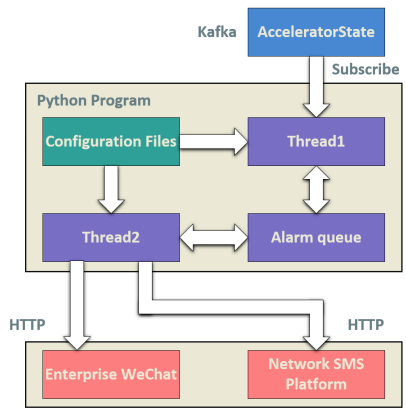



Figure 4: Architecture of WeChat and SMS messages distributors.

NodeJS. When a browser connects to the server, which will trigger the event "connect" in WebSocket, the server will send this object and register this browser (e.g., IP address, port). After that, whenever the server receives a message, it will broadcast this message to the browser which has been registered. When a browser disconnects to the server, which will trigger the event "disconnect" in WebSocket, the server will unregister this browser.

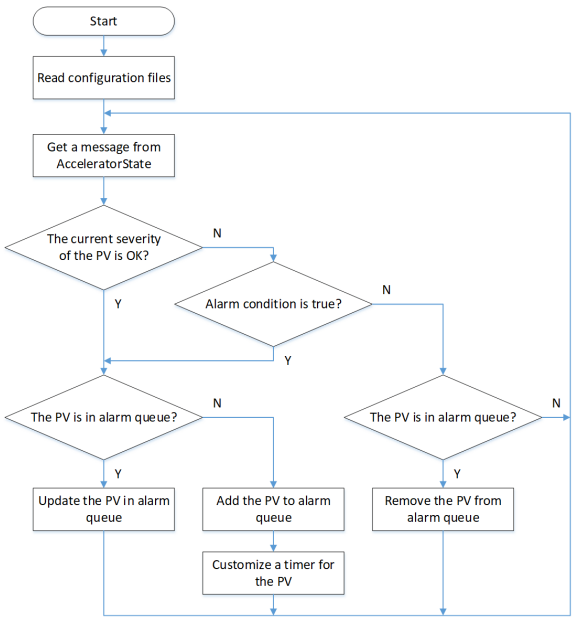
Inspired by Alarm User Interface, the interface of Web-based GUI is designed in a table which can filter and sort PVs. As shown in Fig. 3, the table has 7 columns and can be sorted by "PV", "AlarmTime" and "AlarmSeverity" by clicking the arrow next to the table column name. The PVs can be filtered by group and "AlarmSeverity" by using the checkboxes on the left side of the interface and the checkbox named "Display No-Alarm Information". In the initial configuration, this table will display the abnormal information in all groups in order of "AlarmSeverity". Each row in the table will display different colors depending on "AlarmSeverity". The signal of each group will change color according to the most severe alarm in the group.

### *WeChat and SMS Messages Distributors*

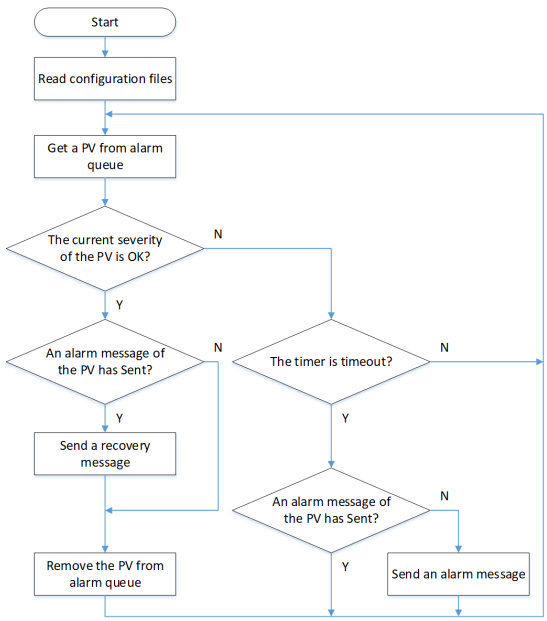
To meet our requirements, WeChat and SMS Messages Distributors have different alarm strategies from the module "Alarms" and can send recovery messages besides alarm messages. As shown in Fig. 4, WeChat and SMS messages

ISBN: 978-3-95450-208-0 doi:10.18429/JACoW-IPAC2019-THPRB038






(a) Message collecting



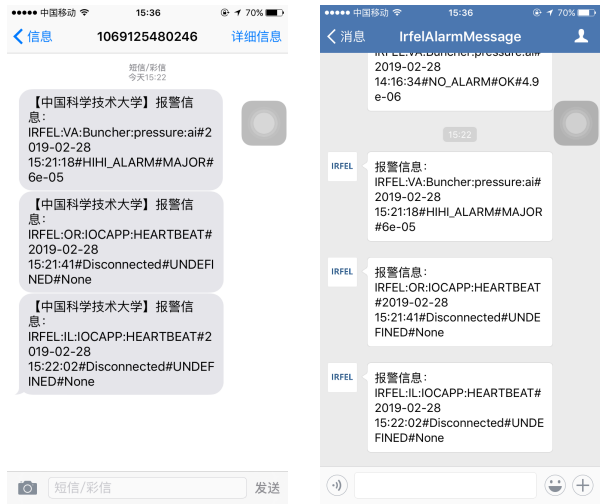

(b) Message processing

Figure 5: Program flow chart.

distributors are integrated into a Python program which has two threads. Communication between these two threads is achieved by an alarm queue. In order to improve this program scalability and flexibility, this Python program has Configuration files include the name of PVs, alarm strategies and staffs information.

*<pv name="IRFEL:VA:IOCAPP:HEARTBEAT" delay="90" ioc="vacuum" group="IRFEL:VA" UNDEFINED="true"/>*

Fig. 5(a) shows the program flow chart of the first thread whose task is to collect messages in "AcceleratorState". Fig. 5(b) shows the program flow chart of the second thread




(a) WeChat (b) SMS

Figure 6: Screenshot of the WeChat and SMS messages.

whose task is to build an alarm or recovery message and send this message. When the Alarm Server produces a message into "AcceleratorState", the first thread will consume this message immediately. If the current severity of this message is not OK, the first thread will determine whether it needs to add the PV to alarm queue according to alarm condition. For example, when an IOC is powered off, the second thread don't have to send a message for each PV in this IOC. We can just allow the "HEARTBEAT" PV whose severity is "UNDEFINED" to be send. The configuration is shown as follows.

Figure 6(a) shows the screenshot of the WeChat messages and Fig. 6(b) shows the screenshot of the SMS messages. The development of SMS alarm application is based on the Network SMS platform while the development of WeChat alarm application is based on Enterprise WeChat. By following the Application Programming Interface (API) [8, 9], the alarm or recovery message which contains the name, timestamp, current status, current severity, value of the PV can be sent to relevant staffs.

## CONCLUSION

To meet our requirements, 3 kinds of alarm message distribution applications are developed. Web-Based GUI provides a way to remotely query the alarm information. WeChat and SMS Messages Distributors have different alarm strategies from the module "Alarms" and can send recovery messages besides alarm messages. At present, the alarm system is steadily running, which can help the relevant staffs to handle alarm information quickly and effectively.

## ACKNOWLEDGEMENT

The authors would like to thank Kay Kasemir of Oak Ridge National Laboratory for his beneficial inspiration and the discussion with him about Phoebus.

ISBN: 978-3-95450-208-0 doi:10.18429/JACoW-IPAC2019-THPRB038

## REFERENCES

[1] S. Xu *et al.*, "Control System Design for Front End Devices of IRFEL", in *Proc. of IPAC2018*, Vancouver, Canada, 2018.

[2] EPICS, https://epics.anl.gov/

[3] ALH, http://www.aps.anl.gov/epics/extensions/alh

[4] K. Kasemir *et al.*, "The Best Ever Alarm System Toolkit", in *Proc. of ICALEPCS2009*, Japan, 2009.

[5] Phoebus, https://github.com/shroffk/phoebus

[6] L.-M. Xu *et al.*, "The Alarm System of Hefei Light Source Based on Beast", in *Nuclear Electronics & Detection Technology*, 2015.

[7] Kafka, https://kafka.apache.org/

[8] The API of Enterprise WeChat, https://work.weixin.qq.com/api/doc

[9] The API of the SMS platform, Shanghai Xi Ao, http://www.sioo.com.cn/APIdoc/smssend.html