EPICS: CSS-Phoebus

Peter Bonneau
2022-11

EPICS Alarm System in Phoebus

I am developing an EPICS alarm system based on CS-Studio Phoebus. Phoebus will be used for new EPICS system development and will replace the existing Eclipse-based CS-Studio systems as detailed in my note DSG Note 2021-37 and talk DSG Talk 2021-17.

The CS-Studio Phoebus alarm system has a core set of programs which must be running at all times. All the other alarm system programs use the core set for communication between other alarm programs, the Phoebus user interface, and communication to external devices and programs.

The core set of programs (Figure 1) include Kafka Zookeeper, Kafka Server, and the Phoebus Alarm Server. The work I've done on the programming and configuration the Kafka programs are detailed in my DSG software memos 2022-02 and 2022-03. The work I've done on the programming of the alarm server is in my DSG software memo 2022-04.

The integration testing of the core set of programs is done with the Phoebus Alarm Test Station. The design, programming, implementation, and operation of the Phoebus Alarm Test System is discussed in my memos 2022-07, 2022-09 and DSG Note 2022-06.

From my research and programming on the Phoebus alarm system, I have determined that the core set of alarm programs must be to be initialized in sequence and operational before the alarm system can be used in any capacity whatsoever. The order of the alarm system core program startup sequence (Table 1) consists of Apache Kafka Zookeeper, Apache Kafka server, procServ, and Phoebus alarm server.

The automatic start and sequencing of the core set of alarm programs is done at the boot of the Linux alarm system computer. Following the running of the Linux boot loader and the initialization of the Linux kernel, the system and service manger for Linux is started (Figure 2). Referred to

- **Developing CS-Studio Phoebus based controls, monitoring, and alarm system - to be implemented on Hall C detectors**
- **System configuration for automatic start and sequencing of alarm support programs upon Linux system boot**
- **Plan to start the design of a high PV signal count IOC for the Phoebus alarm system test station**

EPICS: CSS-Phoebus

as systemd, it bootstraps user space manages user processes. In addition, systemd provides device management, login management, network connection management, and event logging.

For each of the core programs, I developed, tested, and implemented individual ". *service*" command files which encodes information about the alarm system core process controlled and supervised by systemd.

The *zookeeper.service* is the 1st run command file in the alarm system boot sequence and as a prerequisite, the computer networking be initialized and working correctly. The *Kafka.service* systemd command file runs 2nd in the sequence and starts and initializes the Kafka server which hosts the alarm system's three message communication streams.

The *alarm_server.service* command file starts the programs procServ and the alarm server. The Interactive command wrapper procServ allows remote access via telnet to the command console while the system is in operation. procServ launches the alarm server program (Figure 3).

Alarm System Program Name	Automatic Boot Sequence #	Phoebus Alarm System Core Program Function Summary
Kafka Cluster Management *(Zookeeper.service)*	1	Kafka management for alarm system inter-program communication. As a prerequisite, the computer network must be initialized and working.
Kafka server *(Kafka.service)*	2	Hosts the three alarm system message communication streams. Upon initialization, the alarm state, alarm command, and alarm configuration steams are active and ready to service any of the alarm sub-programs.
procServ *(alarm_server.service)*	3	Interactive command wrapper which allows remote access via telnet to the alarm system command console while the system is in operation. procServ launches the Phoebus Alarm Server program.
Alarm server	4	Monitors EPICS process variables (PVs) for alarm conditions via channel access. Stores alarm configuration settings for each PV.

Table 1. Alarm program automatic startup sequencing during Linux boot

I plan to start the design of a high PV signal count IOC for the test station as the next step in the development of the Phoebus alarm system.

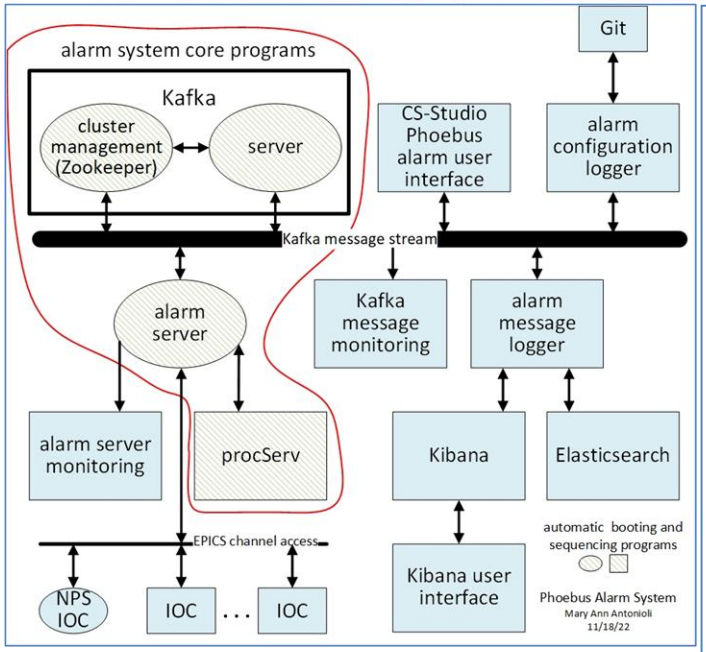

FIG.1. Alarm programs auto started and sequenced

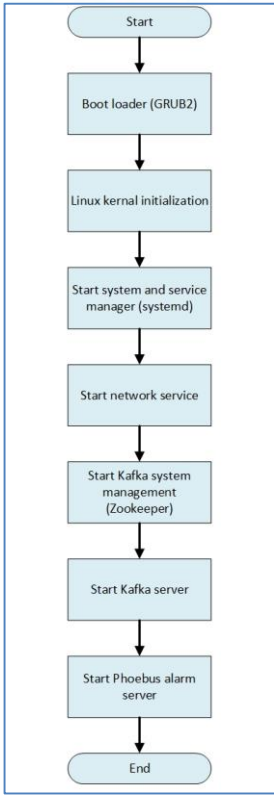

FIG.2. Alarm computer boot sequence

```
● alarm_server.service - Phoebus Alarm Server
     Loaded: loaded (/etc/systemd/system/alarm_server.service; enabled; vendor preset: disabled)
     Active: active (running) since Thu 2022-11-17 17:51:17 EST; 19s ago
   Main PID: 7737 (procServ)
      Tasks: 63 (limit: 37873)
     Memory: 121.6M
        CPU: 2.031s
     CGroup: /system.slice/alarm_server.service
             ├─ 7737 /usr/bin/procServ --foreground --noautorestart --name alarm-server --chdir /home/bonneau/Dow
             ├─ 7742 /bin/sh ./alarm-server.sh -config Hall-C-NPS
             ├─ 7744 java -jar ./target/service-alarm-server-4.6.10-SNAPSHOT.jar -config Hall-C-NPS
             └─ 7787 /usr/lib/jvm/java-11-openjdk-11.0.17.0.8-2.fc35.x86_64/bin/java -classpath ./target/service-

Nov 17 17:51:17 fedora systemd[1]: Started Phoebus Alarm Server.
```

FIG.3. Automatic launch of alarm server via procServ