Debugging the Phoebus Alarm System Annunciator

Peter Bonneau, Mary Ann Antonioli, Aaron Brown, Pablo Campero, Brian Eng, George Jacobs, Mindy Leffel,
Tyler Lemon, Marc McMullen, and Amrit Yegneswaran
Physics Division, Thomas Jefferson National Accelerator Facility, Newport News, VA 23606
May 4, 2023

After an upgrade of the Control System-Studio (CS-Studio) Phoebus alarm system development computer, the alarm system annunciator application failed. This note presents debugging of the failure, which required the development of a program that can read the alarm system inter-process communication messages.

The alarm annunciator failed after an upgrade [1] of the Linux computer being used for the development of the CS-Studio Phoebus alarm system [2]. To debug the failure, each section of code, controls, and monitoring that contributes to the operation of the alarm annunciator was checked.

The Phoebus alarm annunciator is started via the alarm applications menu within the Phoebus alarm user interface (UI), Fig. 1. Upon start of the application, the UI window for the annunciator indicated that it had started, however it would not announce the occurring alarms.

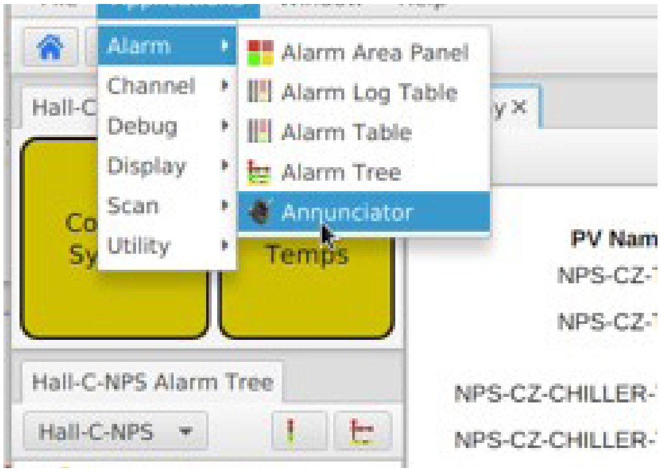
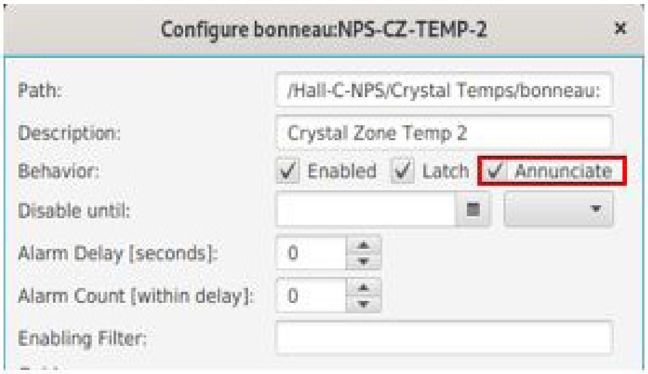


FIG. 1. Phoebus alarm applications menu.

To test the annunciator, the Phoebus alarm test system softIOC [3] was used to generate EPICS process variables (PVs) to test the annunciator. To verify the alarm system annunciator configuration settings were correct for the monitored PVs, the PV alarm configuration menu, Fig. 2, was checked. The readback indicated the annunciator and the PV alarm configuration settings were set correctly. Additionally, the ALARM_PREFERENCES.PROPERTIES file that determines how the annunciator responds to multiple and frequent alarms was checked. The settings in the file were correct.

Phoebus alarm system applications use Apache Kafka Zookeeper and Apache Kafka Server for inter-process communication. There are three Kafka message streams in the alarm system that communicate with the applications, Fig. 3. The first stream, Hall-C-NPS, is used by the alarm system applications to report the alarm status of PVs and to configure the alarm server. The second stream, Hall-C-NPSCommand, sends command messages originating from the alarm UI. The third stream, Hall-C-NPSTalk is exclusively used for the alarm annunciator.


Configure bonneau:NPS-CZ-TEMP-2 ×

Path: /Hall-C-NPS/Crystal Temps/bonneau:
Description: Crystal Zone Temp 2
Behavior: ✓ Enabled ✓ Latch ✓ Annunciate
Disable until:
Alarm Delay [seconds]: 0
Alarm Count [within delay]: 0
Enabling Filter:

FIG. 2. PV alarm configuration menu.

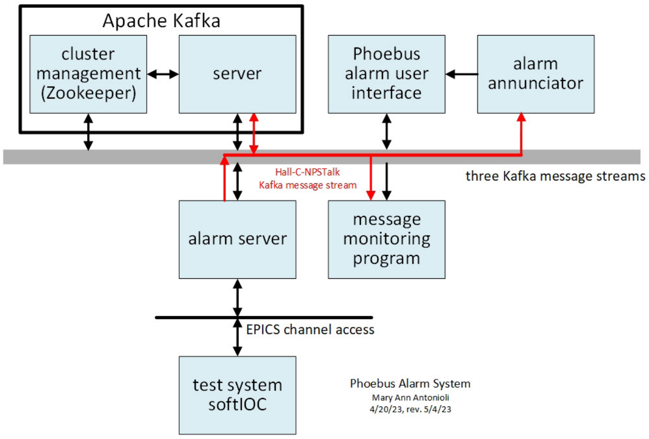

Apache Kafka
cluster management (Zookeeper)
server
Phoebus alarm user interface
alarm annunciator
Hall-C-NPSTalk Kafka message stream
three Kafka message streams
alarm server
message monitoring program
EPICS channel access
test system softIOC
Phoebus Alarm System
Mary Ann Antonioli
4/20/23, rev. 5/4/23

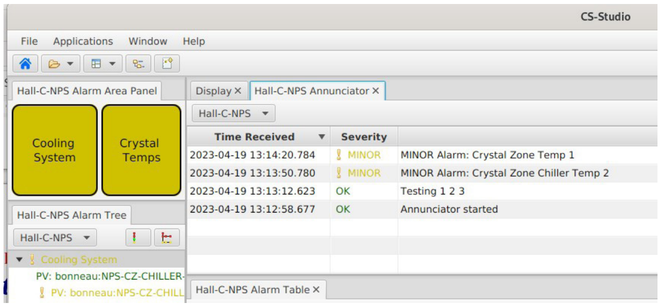
FIG. 3. Phoebus alarm annunciator applications.

The Phoebus alarm server monitors PVs for alarm conditions. Upon the detection of an alarm, the alarm server sends a message to the annunciator application via the unidirectional Kafka message stream Hall-C-NPSTalk. The alarm annunciator translates the received Kafka message, instructs the computer to play an audible warning, and displays information regarding the alarm on the annunciator UI.

Though the Kafka streaming messages are text-based, Phoebus does not have an application that displays the text of a message stream. To aid the debugging, a monitoring program, which can monitor any of the three Phoebus alarm system message streams, was written.

When starting the Kafka message monitoring program, written with Kafka version 2.13-3.3.1 script commands, via a Linux terminal window, the name of the message stream to be monitored is entered as a program switch. As an option, the message text can be written to a text file.

The Kafka message monitoring program revealed that the Hall-C-NPSTalk message stream was present, but no text was available to be read. Upon further investigation, it was found that the script used for the upgrade that created the Hall-C-NPSTalk message stream was incorrect and not compatible with the new Kafka version 2.13-3.3.1. The script was corrected and the Hall-C-NPSTalk message stream was successfully created and retested via the Kafka message monitoring program. The annunciator's audible alarm and the UI monitor screen, Fig. 4, were also correctly reporting the PVs that were in the alarm state.



Time Received	Severity	
2023-04-19 13:14:20.784	MINOR	MINOR Alarm: Crystal Zone Temp 1
2023-04-19 13:13:50.780	MINOR	MINOR Alarm: Crystal Zone Chiller Temp 2
2023-04-19 13:13:12.623	OK	Testing 1 2 3
2023-04-19 13:12:58.677	OK	Annunciator started

FIG. 4. Phoebus alarm annunciator UI monitor.

In conclusion, debugging the failure of the annunciator required the development of an alarm message monitoring program that reads the alarm system inter-process communication messages; this alarm message monitoring program can be used for debugging all Phoebus system alarm applications.

[1] P. Bonneau, et al., *Upgrade of CS-Studio Phoebus and Alarm System Core Support Programs*, DSG Note 2023-13, 2023.

[2] P. Bonneau, et al., *Proposal to Implement Alarm System in Control System Studio Phoebus for the Hall C Neutral Particle Spectrometer*, DSG Note 2021-37, 2021.

[3] P. Bonneau, et al., *Development of the EPICS Software Input/Output Controller for Testing the Phoebus Alarm System of the Hall C Neutral Particle Spectrometer*, DSG Note 2022-06, 2022.