EPICS BEAST ALARM SYSTEM HAPPILY PURRS AT ANKA SYNCHROTRON LIGHT SOURCE

I. Kriznar, Cosylab, Ljubljana
S. Marsching, Aquenos GmbH, Baden-Baden
E. Hertle, E. Huttel, W. Mexner, N. J. Smale, A.-S. Mueller, KIT, Eggenstein-Leopoldshafen

Abstract

The control system of the ANKA synchrotron radiation source at KIT (Karlsruhe Institute of Technology) is adopting new, and converting old, devices into an EPICS control system [1]. New GUI panels are developed in Control System Studio (CSS). EPICS alarming capabilities in connection with the BEAST alarm server tool-kit from the CSS bundle are used as an alarming solution. To accommodate ANKA future requirements as well as ANKA legacy solutions, we have decided to extend the basic functionality of BEAST with additional features in order to manage the alarming for different machine operation states. Since the database of alarm sources is been populated from scratch, we have been able to take a fresh approach in management and creation of alarm sources to build-up alarm trees. The new alarm system is being continuously used, tested and refined, and has been in the production environment since the end of 2013.

INTRODUCTION

ANKA is an electron synchrotron radiation light source located in Karlsruhe, Germany. The storage ring is generally operated at an energy of 2.5GeV with a typical beam current of 200 mA and a life time of 20 hours. The ANKA machine control system has been gradually moved towards EPICS based solutions. Since EPICS is relatively fresh at ANKA we tried to use the best that is available in the EPICS field.

Overview

An alarm system catches, transports, processes and visualizes error conditions (malfunctions) of hardware and software. An alarm system can report problems early, before serious consequences develop, and can provide data for later analysis. It propagates the alarm states from the producers (IOCs, drivers, services, applications) to the clients.

A complete alarm system includes the following components: an alarm source on IOC or services, an alarm aggregation service and a book-keeping service, an alarm archive, an alarm distribution service, a GUI viewer for new and archived alarms, a GUI configuration manager of the alarm service. All these building blocks are nicely covered by CSS alarm system [2], called “Best Ever Alarm System Toolkit” or BEAST [3].

Status Monitoring vs. Alarms

It is very important to distinguish between a status monitoring and an alarm event. Status monitoring system and associated displays gives an overview of information about certain control system components or areas. Changes of status are potential sources of alarms if certain conditions are met. The Alarm view of a system is therefore a dynamic view of status changes, which might be dangerous to the operation.

A short example for illustration: If a server panel displays control system servers with little LED lights attached; Then, if a light is green, the server is running, if red, the server is down. This is status display. On the contrary when an alarm panel is blank, everything should be OK. When one of the servers goes down, the change of status triggers a notification, this is an alarm message, which appears in a table of active alarms until it is acknowledged by the operator. When an operator sees an alarm he/she can open a related status display for more information and consequently take appropriate action.

ALARM SOURCES

We can distinguish several kinds of EPIC’S records in relevance to alarming, thus several kinds of alarm sources.

Control point alarm sources are low level alarm sources. These are records, which are used to control or monitor devices and connected hardware. They generate alarm notifications directly during EPICS record processing and are connected to record intrinsic conditions: like value outside alarm limit or bus errors.

Context dependant alarm sources. are the higher level alarms; they calculate alarm conditions from values obtained from several sources. For example, the machine operation status provides the context, which then defines if an alarm is raised or not for some condition.

Gateway alarms. These are software records which monitor alarm sources, which are not part of EPICS, and forwards their alarms as EPICS alarms.

Alarms which originate in an **interlock system** or any other safety systems, which are generally independent from the rest of control system.

The Context Dependent Alarms

Some machine or device status conditions present valid alarm source only under specific conditions. For example, a certain magnet (or its power supply) turned ON can present a reason for an alarm when the machine is in INJECTION mode, but the same status could be OK or irrelevant when the machine is in SR RAMPING mode.

ISBN 978-3-95450-146-5

Contextual alarms could be produced by implementing contextual logic directly as control point alarm source, such as device IOC, or as a separate server which intercepts and processes the alarm.

Device drivers at the IOC level need to be kept relatively simple as they might be very different in internal record set-up, so implementing contextual design on larger scales seems impractical.

However, this does not seems to be problem if contextual alarm processing is implemented on a separate server, which listens to PVs and lets their alarms through if context of machine state or some other condition allows it. This way alarm triggering logic can be quite complex without complicating device drivers.

In addition the control system must provide all contextual relevant information of the machine in the form of PV channels. This means that at minimum the machine operation status is an EPICS enumeration PV.

ALARM OPERATION POLICY

In business alarming **quality comes before quantity**! An alarm panel full of alarm messages, in which no-one has explanation or knows the cause, especially while the machine seems to be operating without problems, contribute little to the operation. Such an alarm system is not functional. This means that simply adding all (primary) alarm sources or PVs into the alarming system does not provide a functional alarm system.

There are two main rules for a functional alarm system:

- System should provide only alarms to which operator **must react**. If alarm information is nice-to-know, but does not require reaction from operator, then this piece of information is not an alarm. Even if technically it is defined as such.
- Alarm system must not provide to operator more alarms that operator can handle. There are several strategies to prevent this: do not include too many alarms in the first place, integrate/group alarms together or provide ways to easily disable alarmswhich carry no relevance.

Bottom line is that if an operator is bombarded with alarms, he will just stop paying attention to the alarm system and we don't want that.

ANKA ALARM SYSTEM IMPLEMENTATION

As we were introducing the alarm system in the ANKA control room we took care that operators would treat them as a serious information source. We made sure that all the alarms added to the alarm system conformed to the mentioned “Operation policy”. All alarms that were fired without obvious relevance to the operation were carefully examined and then fixed or removed.

Most low level alarms, coming from control point alarm sources and device drivers, do not comply with the operation policy. In addition to EPICS naming convention we have added additional functional requirement in the form of device type convention. This convention prescribes that all devices should implement one additional record of `bi` with PV name `<device>:Status:ErrorSum`. The Status:ErrorSum record should be 0 when a device functions properly and have value raised to 1 and alarm state raised when anything goes wrong with the device or its operation is limited. This requirement reduces the number of PVs to be included per device and makes device behaviour more standard and easier to integrate into alarms, and other tools.

The higher level contextual alarm sources are those sources that contribute the most relevant alarms to the alarm system. This requires a strong intermediate alarm processing level of alarm sources: a conditioning level, which provides functionality between the BEAST alarm server and low level PVs.

This additional functionality was provided with the ANKA Alarm Conditioning Server. The alarm conditioning server hosts alarm dedicated records, which are triggered by processing events from other alarm sources or by monitoring vital signals of the control system, which are not necessary integrated into EPICS. The BEAST alarm server connects to the alarm sources from the conditioning server as it would do to any other PV record. This way conditioned alarm PVs together with the PVs coming directly from low level alarm sources are building up the alarm hierarchy tree in a BEAST alarm configuration. By this no additional complexity is introduced into an already reliably working BEAST archiving, since it treats all alarms equally.

The ANKA alarm conditioning server is implemented on top of a Java CA server (CAS). It listens to other PVs and alarms. It filters alarms so particular alarms are generated or forwarded only when the machine is in an appropriate operation state. It also hosts alarm sources, which are generated by processing other control system parameters. The actual alarm processing is done by the configurable alarm processing units. Each of these alarm processing units provides one alarm dedicated PV into the BEAST alarm hierarchy tree and can be configured to listen to various parameters or PVs. The alarm processing units are loaded and configured from XML configuration files at the alarm conditioning server start-up. Additional scripts takes care that BEAST alarm configuration is updated.

Table 1: An Incomplete List of Alarm Processors, Which Provide Various Alarm Sources at ANKA

Alarm Source Type	Applications
Host ping	Makes periodic network ping to all important servers or IP enabled devices in the control system.

ISBN 978-3-95450-146-5

System Process Watchdog	Runs on Windows or Linux computer, raises alarm when certain process appears in OS task list. Designed to intercept Java Error s on Windows servers.
Status Check	Monitors PVs with bitset value (such as mbbiDirect record value) and raises alarm if bits matches configured bit-masks for on or off states. Status PVs of all power supplies are checked this way.
State Watchdog	A PV value of this processor must be reset in regular intervals by some remote process. If this fails, then alarm is raised. Some crucial application are monitored this way, such as slow orbit correction and some archiving processes.
Summary Alarm	Listens to one or more PVs, sums their alarms, and forwards them further if machine operation state allows. It can also provide configurable alarm sums of a sub-tree in the BEAST alarm hierarchy three.
Value Diff Check	Check value difference between set-point and read-back of a power supply. All power supplies are included. This alarm check has two implementations in use: one as part of conditioning server and one as independent dedicated EPICS server.

In addition to the mentioned processing capability, all these alarm processing units (Table 1) have in addition the capability to filter their raised alarms by machine operation status, which is provided by a control system wide PV. This is again completely configurable within the main XML configuration file.

Because of the general nature of an alarm server it is also used for additional tasks such as a general EPICS application server. It hosts processing tasks for feedback loops, power supply ramping and cycling, and wiggler operation.

The alarm system at ANKA is a project still in progress. Further alarm sources are planned to be added into the alarming. As new device groups are integrated into EPICS they get also their own alarm sources. In addition we want to further add higher level alarm checks for control system health and operation problems.

ALARM CLIENTS

Besides the mentioned addition of alarm conditioning server, the rest of the Alarm solution is basic CSS BEAST installation. ANKA alarming panel is a standard CSS application with preconfigured perspective areas, as shown in Figure 1. The ANKA Launcher opens this panel by a dedicated short-cut, which takes care that each time the alarm panel is opend it is organized in the same way.

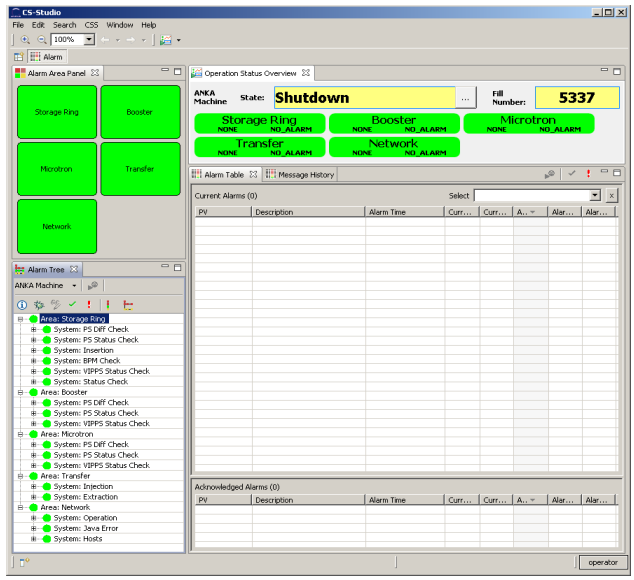

Figure 1: BEAST Alarm panel at ANKA as it is usually seen, without errors, even during a shut-down period, when operation related signals are in an undefined state and therefore their alarms are irrelevant and suppressed in context of shut-down machine status.

CONCLUSION

ANKA has introduced into operation a functioning and relevant alarm system. Operators have learned to trust information in the alarm panel and take alarms serious. This already proved very useful in day-to-day operation. The ANKA alarm system was carefully built with alarm sources that were giving information relevant for operation. The set of alarm sources might not be as wide as would be possible but the ones that are included are of high quality.

ACKNOWLEDGEMENT

Many thanks to wonderful community around CSS that provided solid solution, the BEAST, for delivering alarms to the operators.

REFERENCES

[1] N.J. Smale et al., “The ANKA Control System: On a Path to the Future”, MOPPC099, ICALEPCS’13.

[2] K, Kasemir el al., “Control System Studio Guide - Alarm System”, Http://Cs-Studio.Sourceforge.Net/Docbook/Ch13.Html

[3] K, Kasemir el al., “ Control System Studio Guide”, http://cs-studio.sourceforge.net/docbook/

ISBN 978-3-95450-146-5