CODAC Alarm Handling System

Software Test Plan (STP)
Based on QA Template Version <1.0>

This document describes the tests that should be performed for CODAC Alarm Handling System in order to be installed as part of Core System release. Different test cases are described, as well as and test pass-fail criteria.

Contents

1 INTRODUCTION

1.1 Purpose

This document describes the tests that should be performed for CSS BEAST - Best Ever Alarm System Toolkit - in order to be installed as part of CODAC Core System. These tests will ultimately compare the capabilities of BEAST against these described in CODAC System Requirement (SRD) Document [IDM 28C2HL].

Particular functions to be tested are the alarm configuration, alarm notification and alarm graphical user interface (GUI) – i.e. the main components of the alarm system as shown on Figure 1-I - BEAST Architecture, except the reporting and annunciation functions not yet part of CODAC Core System.

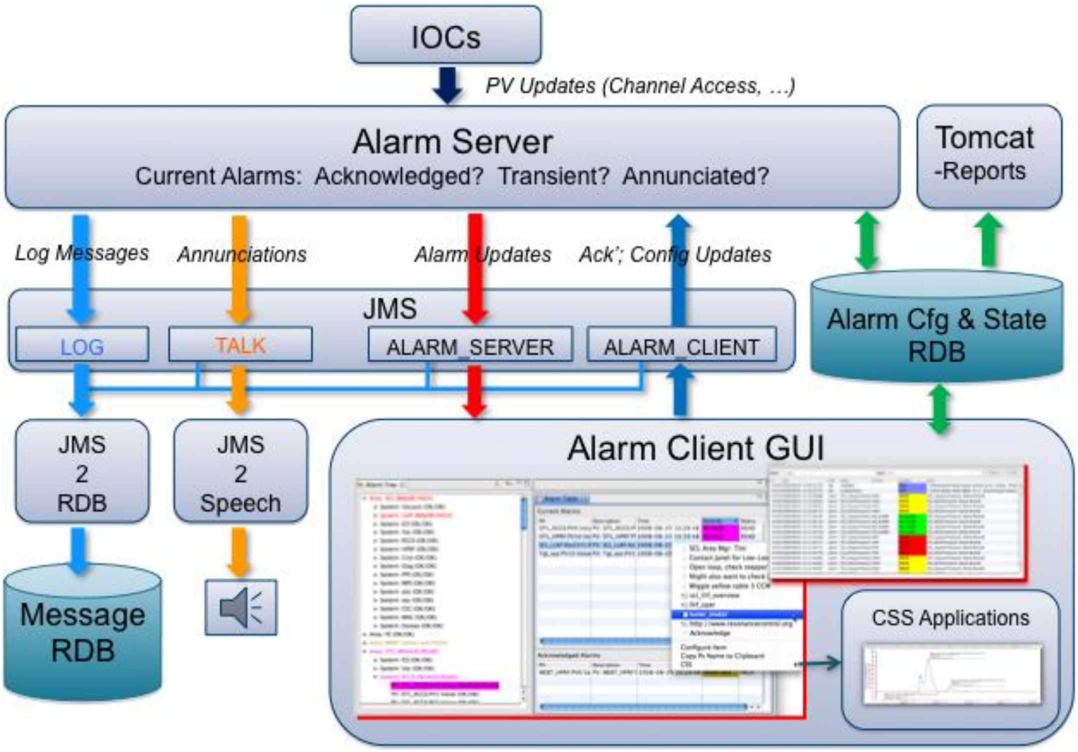


Figure 1-I - BEAST Architecture

1.2 Scope

The test items are:

- The operational version of BEAST,
- The data, including all the configuration data needed to run the alarm system,
- The documentation, including the online help and the release notes.

The installation and uninstallation of the components are not part of this test plan.

1.3 System/Software overview and key features

Best Ever Alarm System Toolkit (BEAST) - is a distributed alarm system consisting of:

- Alarm Server that monitors alarm triggers in the control system,
- Relational Database for configuration and logging,
- CSS user interface for viewing current alarms as a table or hierarchical tree.

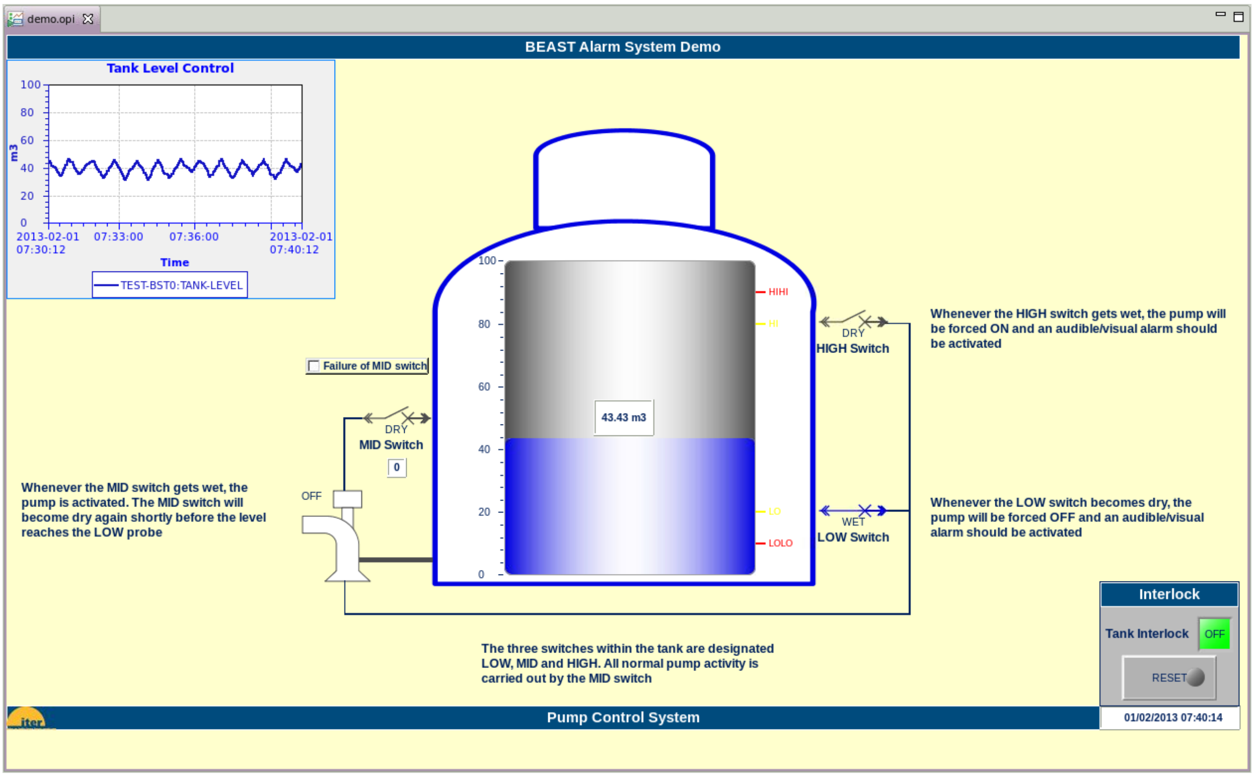
A classic pump control process which manages the level of liquid in a tank will be used to simulate abnormal situations that require the operator attention. When the level in the tank rises to an appropriate level, the pump must be activated. When a lower level is reached the pump must stop.

Three switches within the tank are designed: **LOW**, **MID** and **HIGH**. All normal pump activity is carried out by the **MID** switch. The **LOW** and **HIGH** switches provide the necessary redundancy and supervision to the installation.

Whenever the **MID** switch gets wet, the pump is activated. After some time, the **MID** switch will become dry again. At that point a timer of 25 seconds is started in order to stop the pump shortly before the level reaches the LOW probe.

Whenever the **HIGH** switch gets wet, the pump will be forced **ON** and an auditable / visual alarm will be activated for this abnormal situation.

Whenever the **LOW** switch becomes dry, the pump will be forced **OFF** and an auditable / visual alarm should be activated.



Whenever the MID switch gets wet, the pump is activated. The MID switch will become dry again shortly before the level reaches the LOW probe

Whenever the HIGH switch gets wet, the pump will be forced ON and an audible/visual alarm should be activated

Whenever the LOW switch becomes dry, the pump will be forced OFF and an audible/visual alarm should be activated

The three switches within the tank are designated LOW, MID and HIGH. All normal pump activity is carried out by the MID switch

1.4 References

CODAC Quality assurance plan - https://user.iter.org/?uid=6J7RW4

2 DETAILS OF THE TESTING PROCESS

2.1 Definition of test levels

The described component tests will focus on the desired features of CODAC Alarm Handling System.

Following test levels are defined in this test plan to organize the testing activity.

Alarm Configuration Component Test	**CFG**
Test of the different archiving modes import configuration	
Alarm Server Component Test	**SRV**
Test of archiving overrun and monitoring	
Alarm Display Component Test	**DSP**
Test of CSS Alarm Graphical Interface	
Alarm History Component Test	**HST**
Test of JMS Alarm Log	
Alarm Performance Test	**PRF**
Performance assessment of the User Interface and the server	

2.2 Test administration

2.2.1 Anomaly resolution and reporting

Anomaly Reports shall be submitted in Bugzilla.

2.2.2 Test reporting requirements

The test logs shall be generated to record the outcome of test procedures as described in section *.4 and *.5 of the level test plans.

2.2.3 Test deliverables

The test deliverables include:

- Component Test Logs / Reports
- Anomaly Reports with Bugzilla bug references.

Test input data are registered in SVN code repository.

No other test tool is needed.

The test reports may be submitted on ITER IDM.

3 Component Test Plan

3.1 Scope

3.1.1 Test items and their identifiers

CODAC Alarm Handling System is included the following unit:

- m-css organised in 2 main products:
  - org.csstudio.iter.alarm.beast.configtool.product/
  - org.csstudio.iter.alarm.beast.server.product/

  And 4 utilities

  - org.csstudio.iter.alarm.beast.annunciator.product/
  - org.csstudio.iter.alarm.beast.notifier.product/
  - org.csstudio.iter.jms2rdb.product/
  - org.csstudio.iter.utility.jmssendcmd.product/

CODAC Alarm Handling System mainly depends on different features:

- org.csstudio.iter.alarm.beast.configtool.app.feature/
- org.csstudio.iter.alarm.beast.server.app.feature/
- org.csstudio.iter.alarm.beast.ui.feature/
- org.csstudio.iter.alarm.beast.annunciator.app.feature/
- org.csstudio.iter.alarm.beast.notifier.app.feature/
- org.csstudio.iter.jms2rdb.app.feature/
- org.csstudio.iter.utility.jmssendcmd.app.feature/

3.1.2 Features to be tested

The main CODAC Alarm Handling System features to be tested are:

- Alarm configuration import/export
- Alarm Server startup
- Alarm Notification
- Limit Alarms
- JMS Monitor
- Message History

3.1.3 Features not to be tested

The annunciator and reporting functions are not part of CODAC Alarm Handling System for now.

3.2 Approach

3.2.1 Testing Methods

The overall approach for the level of testing is the Black box method to test the functionality of CODAC Alarm Handling System.

3.2.2 Item pass/fail criteria

Each major anomaly found determines whether each test item has passed or failed testing.

3.3 Environment / Infrastructure

Core System in its development role version should be installed on a CODAC standard machine. Access to SVN is required.

3.4 Component Test Procedures

CFG-01	**3.4.1 Alarm Configuration Import**	

Prerequisite	In a Linux console, create a working directory, download and start a demo IOC:
	1.$ mkdir test
	2.$ cd test
	3.$ svn co https://svnpub.iter.org/codac/iter/codac/dev/units/m-css/trunk/products/ITER/products/org.csstudio.iter.alarm.beast.server.product/demo/m-TEST-BEAST
	`A    ???`
	`Checked out revision xxx.`
	4.$ cd m-TEST-BEAST
	5.$ softIoc -s -d src/main/epics/TEST-BST0App/Db/PSH0-TEST-BST0.db
	`Starting iocInit`
	`############################################################################`
	`## EPICS R3.14.12.3-rc1 $Date: Mon 2012-12-03 16:39:27 -0600$`
	`## EPICS Base built Dec 10 2012`
	`############################################################################`
	`iocRun: All initialization complete`
	`epics> dbl`
	`TEST-BST0:HIGH-SWITCH`
	`TEST-BST0:LOW-SWITCH`
	`TEST-BST0:MID-SWITCH`
	`TEST-BST0:MID-SWITCH-FAIL`
	`TEST-BST0:PUMP`
	`TEST-BST0:TANK-MODE`
	`TEST-BST0:MID-TIMER`
	`TEST-BST0:TANK-LEVEL`
	`TEST-BST0:HIGH-CHECK`
	`TEST-BST0:LOW-CHECK`
	`TEST-BST0:MID-CHECK`
	`TEST-BST0:TANK-HIGH`
	`TEST-BST0:TANK-INTERLOCK`
	`TEST-BST0:TANK-LOW`
	`TEST-BST0:TANK-MID`
	`TEST-BST0:TIMER-CHECK`
	`TEST-BST0:TIMER-START`
	`TEST-BST0:IOC-TIME`
	`epics>`	

Test Cases	1. Positive confirmation of the alarm configuration loaded	
Procedure	In a new Linux console, import the alarm configuration for a “demo” alarm server:	

	1.$ cd test/m-TEST-BEAST 2.$ alarm-configtool -root demo -import -file src/main/beast/TEST-BST0-beast.xml Check that the Demo Archive Engine is configured 3. $ alarm-configtool -list	

Pass Criteria	2. The output of the command should be:
	`Alarm Config Tool <current Core System version>`
	`2013-01-23 13:33:36.332 INFO [Thread 19]`
	`org.apache.activemq.transport.failover.FailoverTransport (doReconnect) -`
	`Successfully connected to tcp://localhost:61616`
	`Reading RDB configuration of 'demo'`
	`Deleting existing RDB configuration for 'demo'`
	`Importing configuration 'demo' from src/main/beast/TEST-BST0-beast.xml`
	`Loading /demo/TEST-BST0`
	`Loading /demo/TEST-BST0/TEST-BST0`
	`Loading /demo/TEST-BST0/TEST-BST0/TEST-BST0:HIGH-SWITCH`
	`Loading /demo/TEST-BST0/TEST-BST0/TEST-BST0:LOW-SWITCH`
	3. The output of the command should contain the following declaration:
	`Alarm Config Tool <current Core System version>`
	`2013-01-23 13:35:09.030 INFO [Thread 19]`
	`org.apache.activemq.transport.failover.FailoverTransport (doReconnect) -`
	`Successfully connected to tcp://localhost:61616`
	`'demo'`
	`…`	

CFG-02	**3.4.2 Alarm Configuration Export**	
Prerequisite	1. Alarm Configuration Imported successfully	
Test Cases	1. Positive confirmation of the alarm configuration export	
Procedure	In the previous Linux console, export the Demo Alarm Server configuration: 1.$ alarm-configtool -root demo -export -file src/main/beast/export-beast.xml Check the exported configuration: 3. $ gedit src/main/beast/export-beast.xml& After the check, close gedit.	

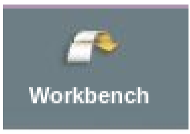
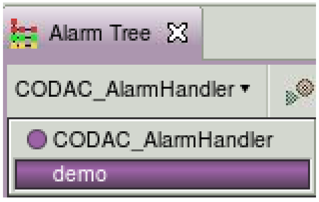
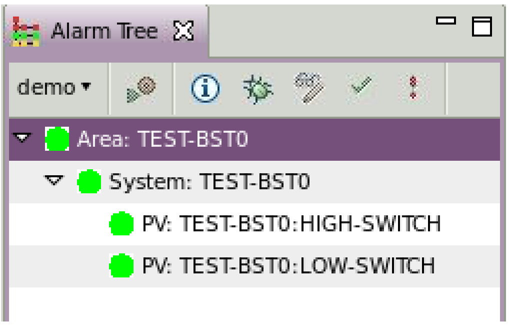
Pass Criteria	1. The output of the command should be:
	`archive-configtool -engine demo -export -config src/main/beauty/export-beauty.xml`
	`Exporting config for engine demo to src/main/beauty/export-beauty.xml`
	`2013-01-07 13:50:18.700 INFO [Thread 19]`
	`org.apache.activemq.transport.failover.FailoverTransport (doReconnect) -`
	`Successfully connected to tcp://localhost:61616`	

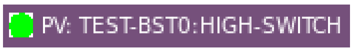
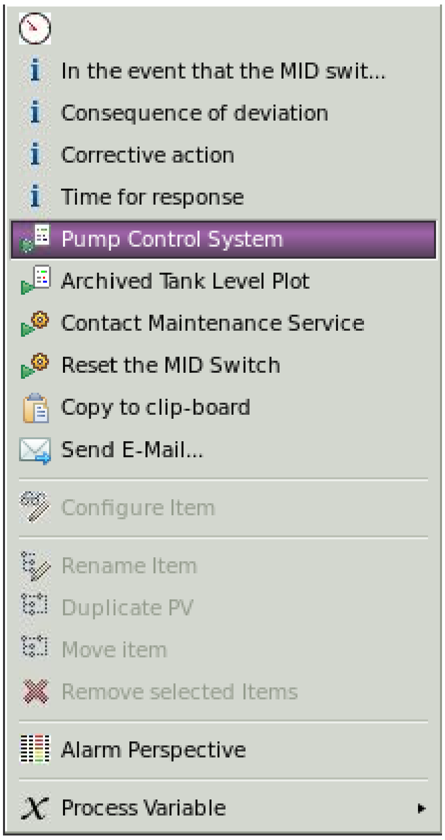
2. The xml configuration should be:

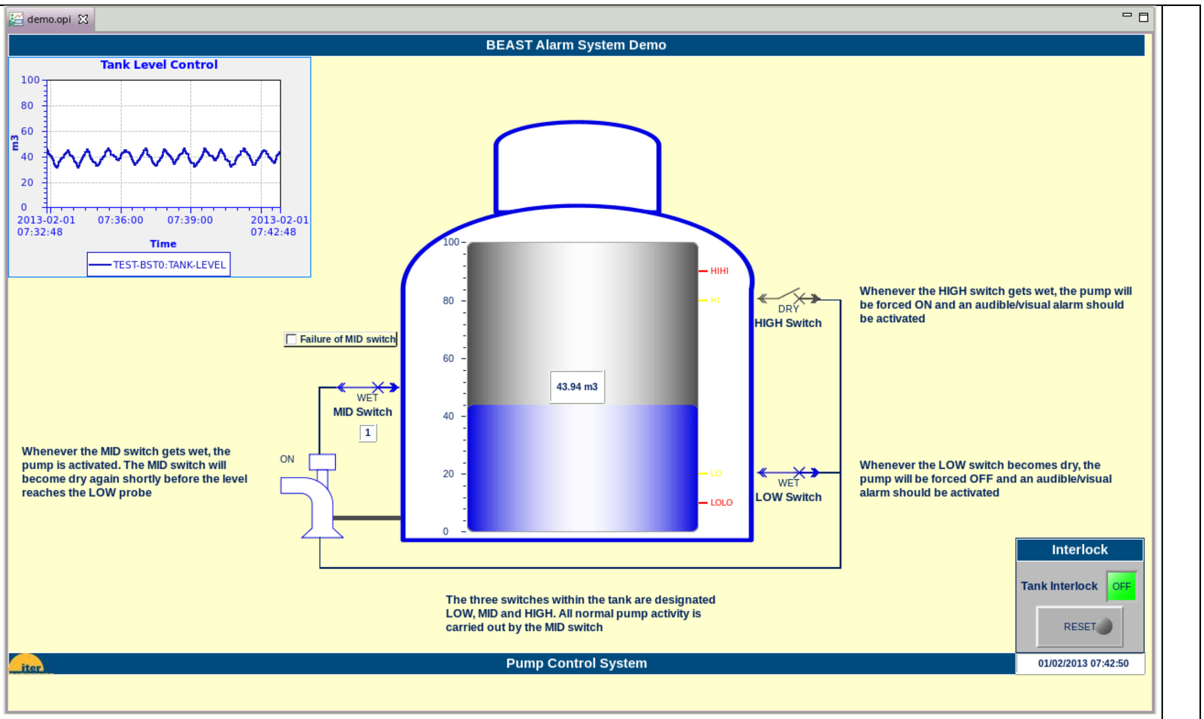
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!--
     Alarm configuration snapshot Wed Jan 30 08:58:28 UTC 2013
     URL : jdbc:postgresql://localhost/css_alarm_3_0_0
     Root: demo
  -->
<config name="demo">
    <component name="TEST-BST0">
        <component name="TEST-BST0">
            <pv name="TEST-BST0:HIGH-SWITCH">
                <description>Whenever the HIGH switch gets wet, the pump will be forced ON</description>
                <latching>true</latching>
                <annunciating>true</annunciating>
                <guidance>
                    <title>In the event that the MID switch has failed</title>
                    <details>The tank level has continued to increase and has reached the HIGH level generating a MAJOR
alarm</details>
                </guidance>
                <guidance>
                    <title>Consequence of deviation</title>
                    <details>The MID switch fails to operate: defective switch, blocked switch or damaged
wiring.&#10;As a consequence the pump remaining dormant, the tank will overflow, causing inconvenience and damage to
property.</details>
                </guidance>
                <guidance>
                    <title>Corrective action</title>
                    <details>Reset the MID Switch (if needed)</details>
                </guidance>
                <guidance>
                    <title>Time for response</title>
                    <details>The tank will overflow if the pump fails to be forced ON in 10 minutes</details>
                </guidance>
                <display>
                    <title>Pump Control System</title>
                    <details>/m-TEST-BEAST/src/main/boy/demo.opi</details>
                </display>
                <display>
                    <title>Archived Tank Level Plot</title>
                    <details>file:/m-TEST-BEAST/src/main/databrowser/tank-level.plt</details>
                </display>
                <command>
                    <title>Reset the MID Switch</title>
                    <details>caput TEST-BST0:MID-SWITCH-FAIL OFF</details>
                </command>
                <automated_action>
                    <title>Contact Maintenance Service</title>
                    <details>mailto:someone@iter.org,someone_else@iter.org?subject=*MID Switch fails to maintain the
tank level&amp; body=*{0} alarm - Tank Switch is {1}</details>
                    <delay>5</delay>
                </automated_action>
            </pv>
            <pv name="TEST-BST0:LOW-SWITCH">
                <description>Whenever the LOW switch becomes dry, the pump will be forced OFF</description>
                <latching>true</latching>
                <annunciating>true</annunciating>
                <guidance>
                    <title>In the event that the MID switch has failed</title>
                    <details>The tank level has continued to decrease and has reached the LOW level generating a MAJOR
alarm</details>
                </guidance>
                <guidance>
                    <title>Consequence of deviation</title>
                    <details>The MID switch fails. The outcome is simple, the pump will drain the tank until it is
empty, whereupon it will try to pump air indefinitely and burn out.&#10;Unfortunately, every time a MID switch fails
for too long, a new pump may have to be installed</details>
                </guidance>
                <guidance>
                    <title>Corrective action</title>
                    <details>Change the MID Switch and check if the pump has already burned out</details>
                </guidance>
                <guidance>
                    <title>Time for response</title>
                    <details>The pump will burn out in 30 minutes</details>
                </guidance>
                <display>
                    <title>Pump Control System</title>
                    <details>/m-TEST-BEAST/src/main/boy/demo.opi</details>
                </display>
                <display>
                    <title>Archived Tank Level Plot</title>
                    <details>file:/m-TEST-BEAST/src/main/databrowser/tank-level.plt</details>
                </display>
                <command>
                    <title>Reset the MID Switch</title>
                    <details>caput TEST-BST0:MID-SWITCH-FAIL OFF</details>
                </command>
                <automated_action>
                    <title>Contact Maintenance Service</title>
                    <details>mailto:someone@iter.org</details>
                    <delay>0</delay>
                </automated_action>
            </pv>
        </component>
    </component>
</config>
```

SRV-01	**3.4.3 Alarm Server Startup**	
Prerequisite	1. Demo IOC running 2. Alarm Configuration Imported successfully	
Test Cases	1. Positive confirmation of the demo alarm server and notifier started	
Procedure	In the previous Linux console, start the “demo” alarm server: 1.$ alarm-server -root demo& 2. $ alarm-notifier -root demo& 3. [optional - only if the machine has a sound card and speakers] alarm-annunciator&	

Pass Criteria	1. The output of the command should be:
	`$ 2012-05-31 18:45:37.847 INFO [Thread 10] org.csstudio.alarm.beast.server.Application (start) - Alarm Server <current Core System version> started for 'demo' configuration`
	`Alarm Server <current Core System version>`
	`Configuration Root: demo`
	`JMS Server Topic:   demo_SERVER`
	`JMS Client Topic:   demo_CLIENT`
	`JMS Talk Topic:     demo_TALK`
	`JMS Global Topic:   GLOBAL_SERVER`
	`<… many Info messages …>`
	`Read 2 PVs in xxx seconds: yyy PVs/sec`
	`<… many Info messages …>`
	2. The output of the command should be:
	`$ 2012-10-10 14:34:46.591 INFO [Thread 1] org.csstudio.alarm.beast.notifier.Application (start) - Alarm Notification <current Core System version> started for 'demo' configuration`
	`Alarm Notification <current Core System version>`
	`Configuration Root: demo`
	`JMS Server Topic:   demo_SERVER`
	`JMS Client Topic:   demo_CLIENT`
	`JMS Global Topic:   GLOBAL_SERVER`
	`Notifier timer threshold: 100`
	`Notifier thread threshold: 100`
	`… org.csstudio.alarm.beast.notifier.AlarmNotifier (start) - Alarm Notifier started`
	`<… many other messages …>`	

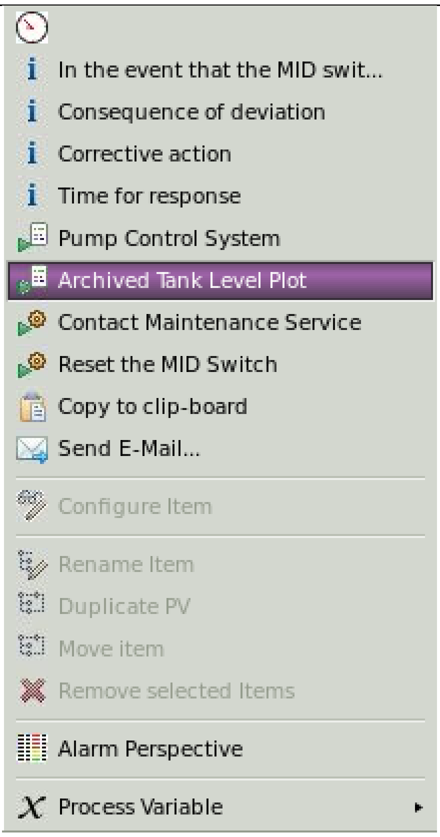
DSP-01	**3.4.4 Alarm PV Tree**	
Prerequisite	1. Demo IOC running 2. Alarm Configuration imported 3. Alarm server started	
Test Cases	1. Positive confirmation of Demo alarm configuration and state	
Procedure	In the previous Linux console, start the Operator Interface to monitor the alarms triggered: 0. $ cd .. 1.$ css& 2. Browse to select the working directory test 3. Close the Welcome screen by clicking on Workbench icon: . Import the demo project into the Workspace from the Navigator View: 4. Right-click and select the option Import… and then General -> “Existing Projects into Workspace”. Click on Next button. To select the root directory, click on Browse button, select m-TEST-BEAST and click OK. To import the selected project, click on Finish 5. Open the Alarm Perspective: menu Window -> Open Perspective -> Other… and select Alarm 6. Change the root element of the Alarm Tree View using the arrow near CODAC_AlarmHandler and select demo:  Alarm Tree CODAC_AlarmHandler ▾ CODAC_AlarmHandler demo	
Pass Criteria	6. The Alarm Tree should reflect the demo server structure and current alarms state:  Area: TEST-BST0 System: TEST-BST0 PV: TEST-BST0:HIGH-SWITCH PV: TEST-BST0:LOW-SWITCH	

DSP-02	**3.4.5 Alarm Related Display – Operator Interface**	
Prerequisite	1. Demo IOC running 2. Alarm Configuration imported 3. Alarm server started 4. Alarm Operator Interface started	
Test Cases	1. Positive confirmation of OPI related display	
Procedure	From CSS Alarm Tree View: 1. Right-click on the PV: TEST-BST0:HIGH-SWITCH   In the event that the MID swit... Consequence of deviation Corrective action Time for response Pump Control System Archived Tank Level Plot Contact Maintenance Service Reset the MID Switch Copy to clip-board Send E-Mail... Configure Item Rename Item Duplicate PV Move item Remove selected Items Alarm Perspective Process Variable 2. Select the option Pump Control System and check that the main monitoring operator interface is opened. In order to have a better view, you can switch to the dedicated perspective OPI Runtime: menu Window -> Open Perspective -> Other… and select OPI Runtime.	
Pass Criteria	2. The Pump Control System Operator Interface is displayed and you can monitor the tank level - All normal pump activity is carried out by the MID switch:	

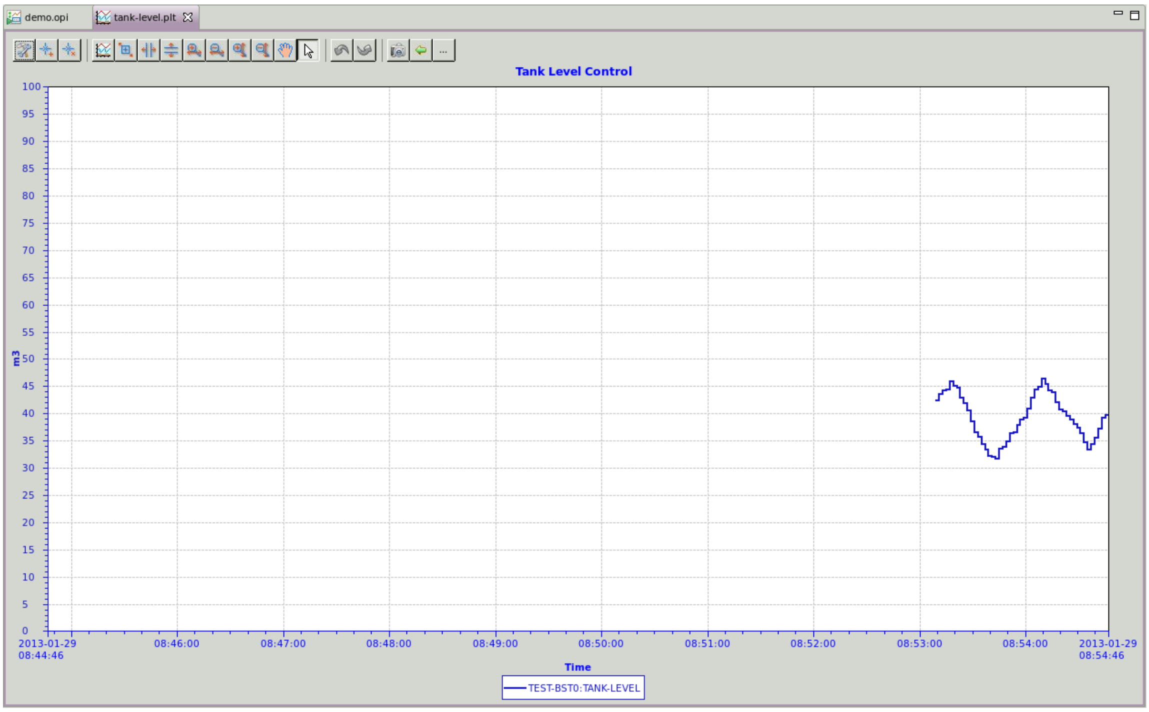
	 demo.opi BEAST Alarm System Demo Tank Level Control 100 80 60 m3 40 20 0 2013-02-01 07:32:48 07:36:00 07:39:00 2013-02-01 07:42:48 Time TEST-BST0:TANK-LEVEL □ Failure of MID switch WET MID Switch 1 ON 100 80 60 40 20 0 43.94 m3 HIHI HI LO LOLO DRY HIGH Switch WET LOW Switch Interlock Tank Interlock OFF RESET Pump Control System 01/02/2013 07:42:50 Whenever the HIGH switch gets wet, the pump will be forced ON and an audible/visual alarm should be activated Whenever the LOW switch becomes dry, the pump will be forced OFF and an audible/visual alarm should be activated Whenever the MID switch gets wet, the pump is activated. The MID switch will become dry again shortly before the level reaches the LOW probe The three switches within the tank are designated LOW, MID and HIGH. All normal pump activity is carried out by the MID switch	
DSP-03	### 3.4.6 Alarm Related Display – Data Plot	

Prerequisite	1. Demo IOC running
	2. Alarm Configuration imported
	3. Alarm server started
	4. Alarm Operator Interface started
	5. [optional only if you want some archived data]
	$ cd m-TEST-BEAST
	$ archive-configtool -engine demo -description 'Demo Test Engine' -port 5812 -import -config src/main/beauty/TEST-BST0-beauty.xml -replace_engine
	$ archive-engine -port 5812 -engine demo&	

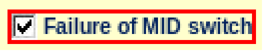
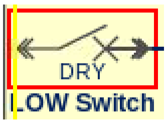
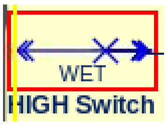
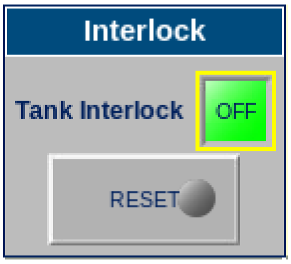
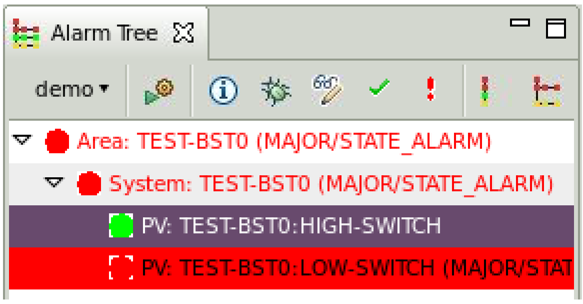
Test Cases	1. Positive confirmation of DataBrowser plot related display	
Procedure	From CSS Alarm Perspective - Alarm Tree View: 1. Right-click on the PV: TEST-BST0:HIGH-SWITCH PV: TEST-BST0:HIGH-SWITCH	

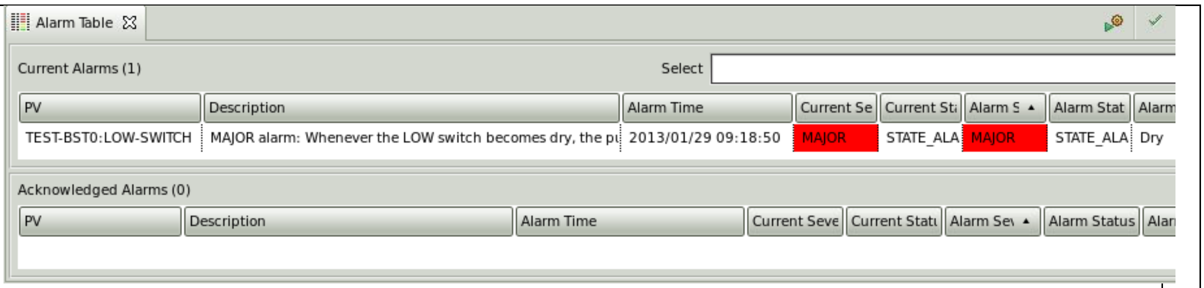
<table>
<tr>
<td></td>
<td>


2. Select the option Archived Tank Level Plot and check that the plot is opened
</td>
</tr>
<tr>
<td>Pass Criteria</td>
<td>
2. By default, the plot is configured to retrieve – 10 minutes from the Archive Database and then it subscribes directly to the EPICS PV to have live data. After some minutes, the tank level should oscillate under the mid-level of 50 m3:



</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</table>

SRV-02	**3.4.7 Alarm Notification**	
Prerequisite	1. Demo IOC running 2. Alarm Configuration imported 3. Alarm server started 4. Alarm Operator Interface started	
Test Cases	1. Positive confirmation of an alarm triggered	
Procedure	From the Operator Interface demo.opi, put the MID switch in error mode: 1. Check the box Failure of Mid switch  2. Wait for LOW switch to be dry (tank level less than LOW level of 20 m3) or the HIGH switch to be wet (tank level more than HIGH level of 80 m3):  or  Then switch to Alarm Perspective to check the current status 3. Click on Interlock Reset button on the bottom right of the operator interface  and confirm the action.	
Pass Criteria	2. If the pump was activated when the fault was triggered, the lower lever is reached after some time and a MAJOR alarm is generated on TEST-BST0:LOW-SWITCH:  Area: TEST-BST0 (MAJOR/STATE_ALARM) System: TEST-BST0 (MAJOR/STATE_ALARM) PV: TEST-BST0:HIGH-SWITCH PV: TEST-BST0:LOW-SWITCH (MAJOR/STAT The LOW alarm is also reflected in the Alarm Table View:	



Alarm Table

Current Alarms (1) Select

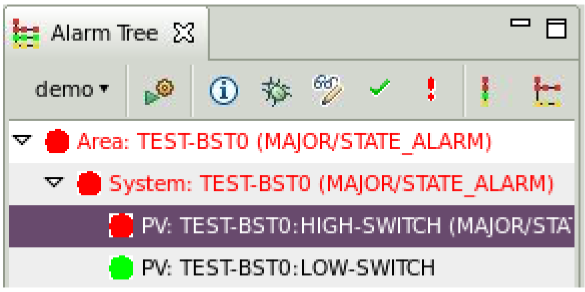
PV	Description	Alarm Time	Current Se	Current Sta	Alarm S	Alarm Stat	Alarm
TEST-BST0:LOW-SWITCH	MAJOR alarm: Whenever the LOW switch becomes dry, the pu	2013/01/29 09:18:50	MAJOR	STATE_ALA	MAJOR	STATE_ALA	Dry

Acknowledged Alarms (0)

PV	Description	Alarm Time	Current Seve	Current Statu	Alarm Sev	Alarm Status	Alar

When resetting the system, the pump is stopped and the tank level starts to increase.

On the other hand, if the pump was stopped when the fault was generated on MID switch, the tank will overflow and generate a MAJOR alarm on PV: TEST-BST0:HIGH-SWITCH:

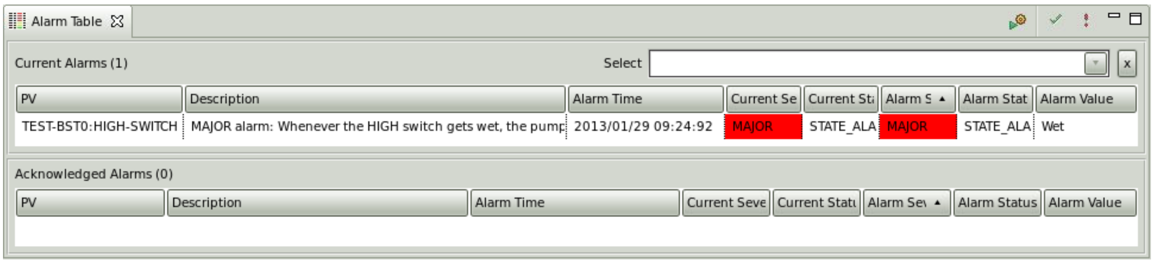


Alarm Tree

demo

- Area: TEST-BST0 (MAJOR/STATE_ALARM)
  - System: TEST-BST0 (MAJOR/STATE_ALARM)
    - PV: TEST-BST0:HIGH-SWITCH (MAJOR/STA
    - PV: TEST-BST0:LOW-SWITCH

The HIGH alarm appears also in the Alarm Table View:



Alarm Table

Current Alarms (1) Select

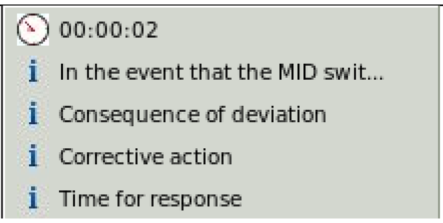
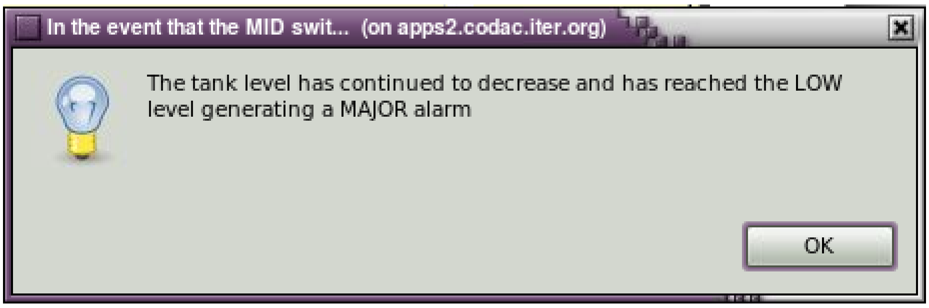
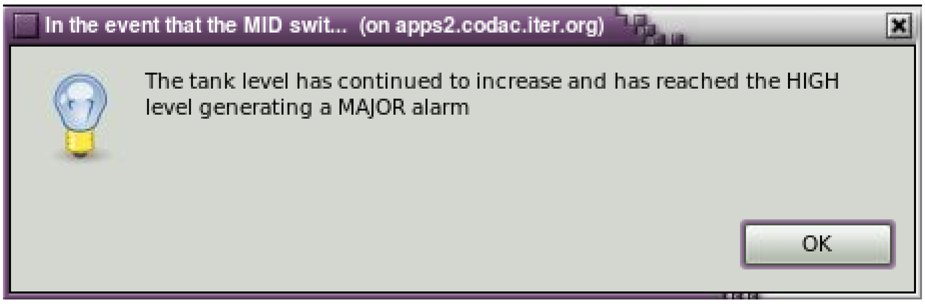
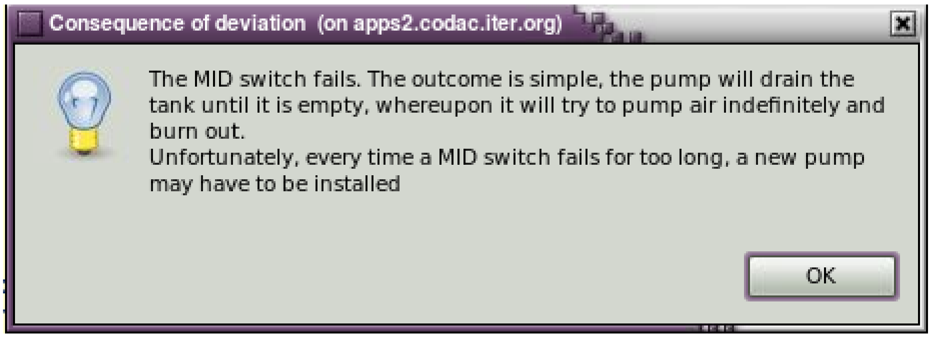
PV	Description	Alarm Time	Current Se	Current Sta	Alarm S	Alarm Stat	Alarm Value
TEST-BST0:HIGH-SWITCH	MAJOR alarm: Whenever the HIGH switch gets wet, the pump	2013/01/29 09:24:92	MAJOR	STATE_ALA	MAJOR	STATE_ALA	Wet

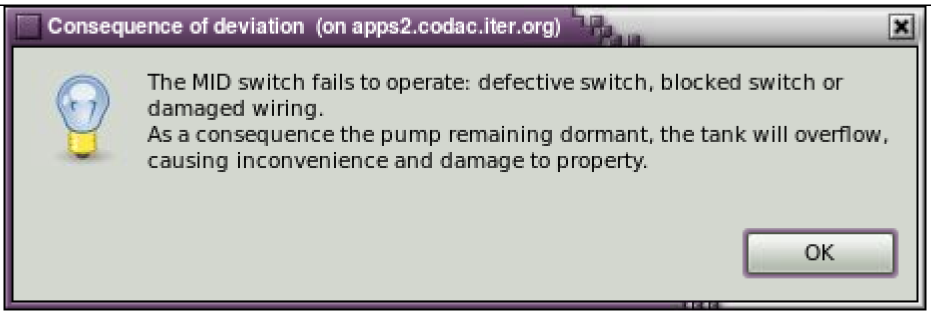
Acknowledged Alarms (0)

PV	Description	Alarm Time	Current Seve	Current Statu	Alarm Sev	Alarm Status	Alarm Value

When resetting the system, the pump is started and the tank level starts to decrease.

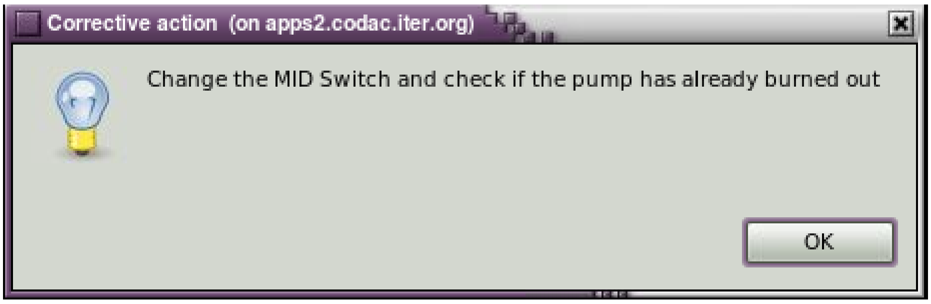
DSP-04	**3.4.8 Alarm Guidance**
Prerequisite	1. Demo IOC running 2. Alarm Configuration imported 3. Alarm server started 4. Alarm Operator Interface started
Test Cases	1. Positive confirmation of alarm guidance
Procedure	From the Alarm Perspective, right-click on the MAJOR alarm:

	 00:00:02 In the event that the MID swit... Consequence of deviation Corrective action Time for response 1. Check when (clock icon) the alarm has been triggered 2. Click on the first Guidance message “In the event that the MID switch…”. Click OK 3. Then click on the other Guidance messages: “Consequence of deviation”, “Corrective action”, “Time for response”	
Pass Criteria	2. Depending on the LOW or HIGH MAJOR alarm, the guidance is different. Guidance for TEST-BST0:LOW-SWITCH:  In the event that the MID swit... (on apps2.codac.iter.org) OK The tank level has continued to decrease and has reached the LOW level generating a MAJOR alarm Guidance for TEST-BST0:HIGH-SWITCH:  In the event that the MID swit... (on apps2.codac.iter.org) OK The tank level has continued to increase and has reached the HIGH level generating a MAJOR alarm 3. “Consequence of deviation” for TEST-BST0:LOW-SWITCH:  The MID switch fails. The outcome is simple, the pump will drain the tank until it is empty, whereupon it will try to pump air indefinitely and burn out. Unfortunately, every time a MID switch fails for too long, a new pump may have to be installed “Consequence of deviation” for TEST-BST0:HIGH-SWITCH:	



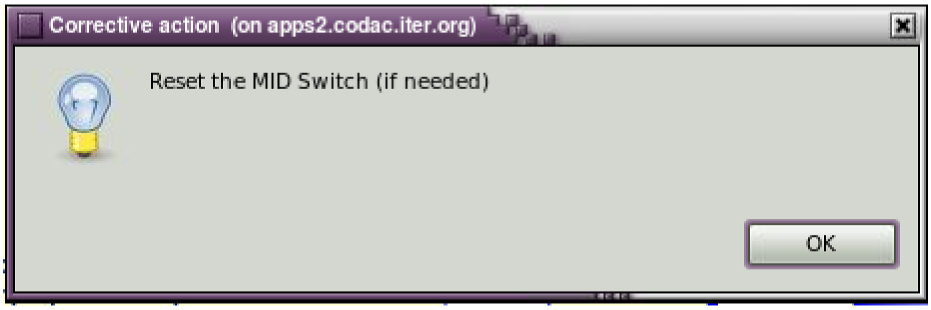
The MID switch fails to operate: defective switch, blocked switch or damaged wiring.
As a consequence the pump remaining dormant, the tank will overflow, causing inconvenience and damage to property.

“Corrective action” for TEST-BST0:LOW-SWITCH:



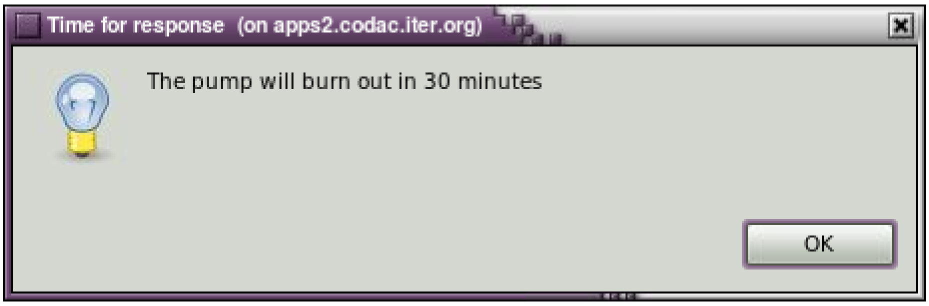
Change the MID Switch and check if the pump has already burned out

“Corrective action” for TEST-BST0:HIGH-SWITCH:



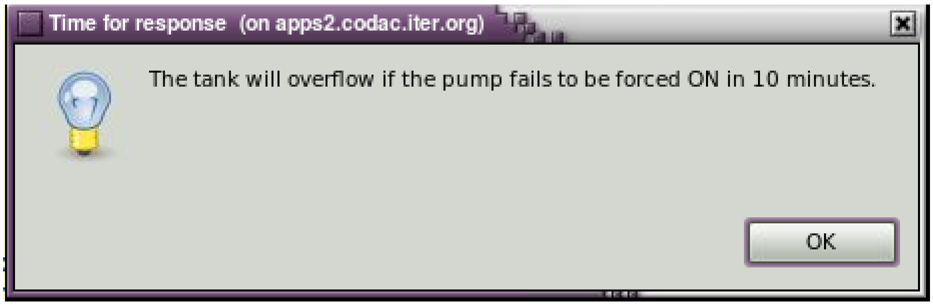
Reset the MID Switch (if needed)

“Time for response” for TEST-BST0:LOW-SWITCH:

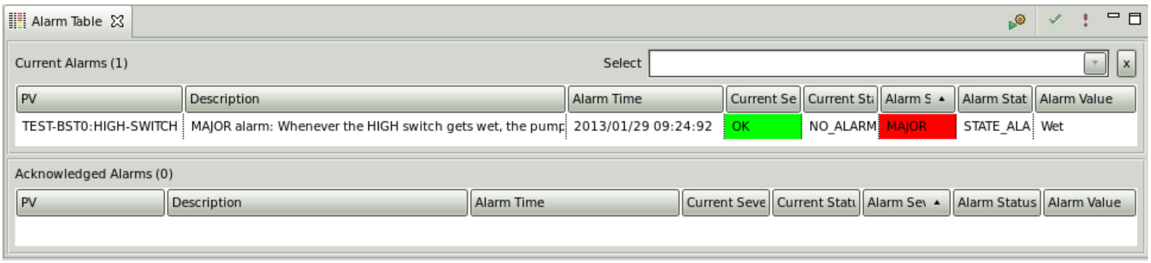


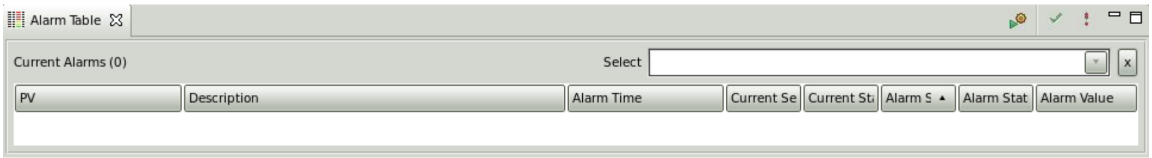
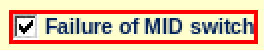
The pump will burn out in 30 minutes

“Time for response” for TEST-BST0:HIGH-SWITCH:

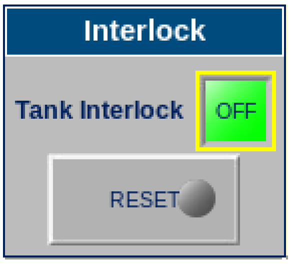
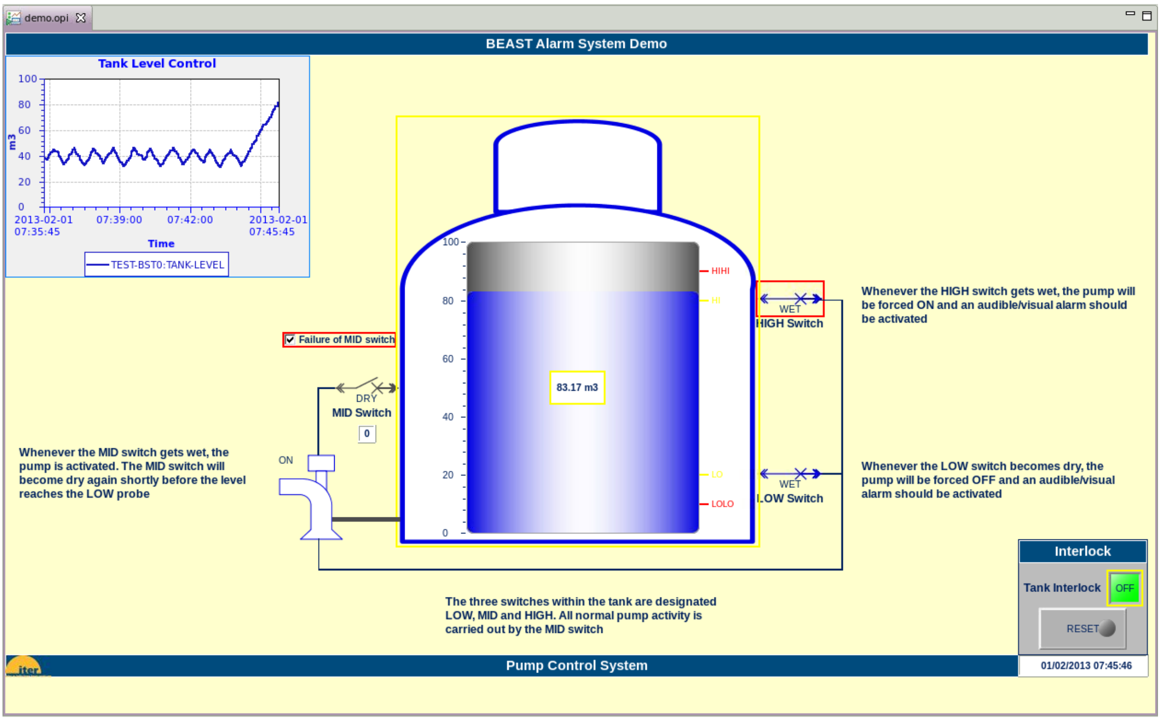


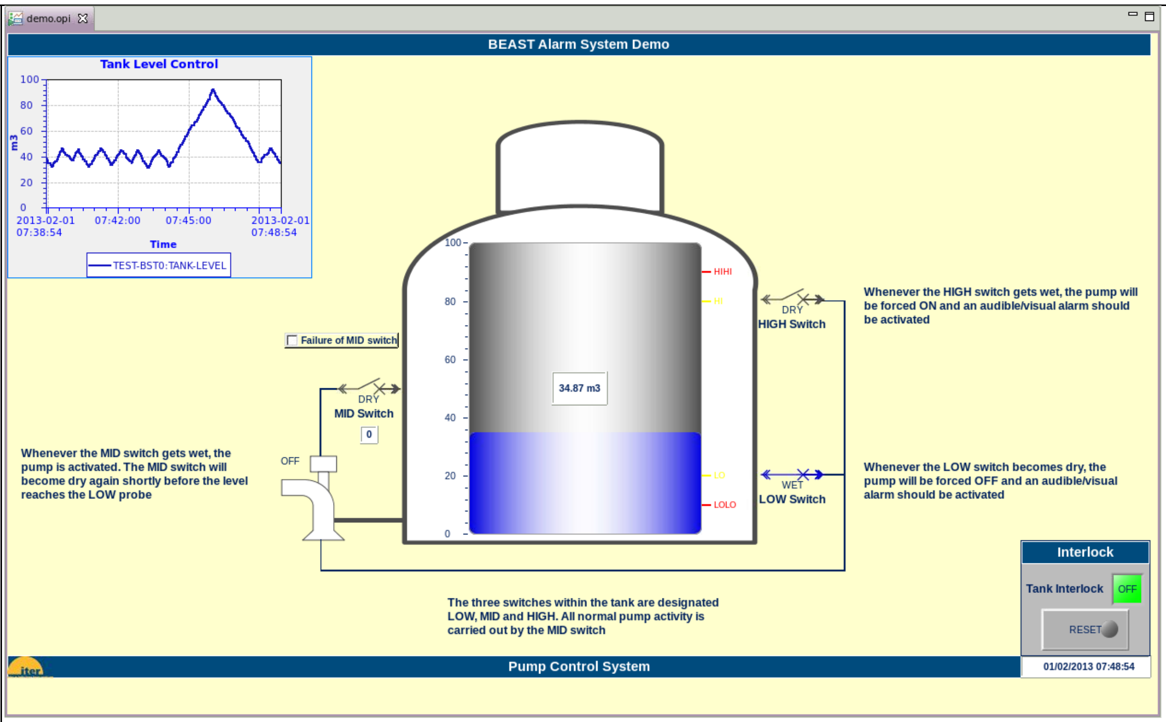
The tank will overflow if the pump fails to be forced ON in 10 minutes.

<table>
<tr><td>DSP-05</td><td>3.4.9 Latched Alarm</td><td></td></tr>
<tr><td>Prerequisite</td><td>1. Demo IOC running
2. Alarm Configuration imported
3. Alarm server started
4. Alarm Operator Interface started</td><td></td></tr>
<tr><td>Test Cases</td><td>1. Positive confirmation of latched alarm</td><td></td></tr>
<tr><td>Procedure</td><td>From the Alarm Perspective, even when the alarm has recovered (Current Severity is OK), the alarm should be still present in the Alarm Table and require to be acknowledged:

Alarm Table
Current Alarms (1) Select
<table>
<tr><th>PV</th><th>Description</th><th>Alarm Time</th><th>Current Se</th><th>Current Sta</th><th>Alarm S ▲</th><th>Alarm Stat</th><th>Alarm Value</th></tr>
<tr><td>TEST-BST0:HIGH-SWITCH</td><td>MAJOR alarm: Whenever the HIGH switch gets wet, the pump</td><td>2013/01/29 09:24:92</td><td>OK</td><td>NO_ALARM</td><td>MAJOR</td><td>STATE_ALA</td><td>Wet</td></tr>
</table>
Acknowledged Alarms (0)
<table>
<tr><th>PV</th><th>Description</th><th>Alarm Time</th><th>Current Seve</th><th>Current Statu</th><th>Alarm Sev ▲</th><th>Alarm Status</th><th>Alarm Value</th></tr>
</table>
1. Right-click on the recovered alarm and select the option Acknowledge:

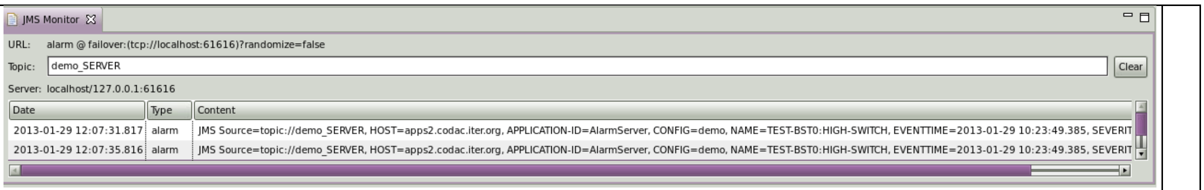
</td><td></td></tr>
<tr><td>Pass Criteria</td><td>1. The recovered and acknowledged alarm is suppressed from the Current Alarms Table:

Alarm Table
Current Alarms (0) Select
<table>
<tr><th>PV</th><th>Description</th><th>Alarm Time</th><th>Current Se</th><th>Current Sta</th><th>Alarm S ▲</th><th>Alarm Stat</th><th>Alarm Value</th></tr>
</table></td><td></td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>DSP-05</td><td>3.4.10 Display Limit Alarms</td><td></td></tr>
<tr><td>Prerequisite</td><td>1. Demo IOC running
2. Alarm Configuration imported
3. Alarm server started
4. Alarm Operator Interface started</td><td></td></tr>
<tr><td>Test Cases</td><td>1. Positive confirmation of display limit alarms</td><td></td></tr>
<tr><td>Procedure</td><td>In CSS, when the tank level reaches High / High-High limits or Low / Low-Low limits, the tank border reflects the display alarm status:
1. Check the box Failure of Mid switch


2. Wait for the tank level to reach an alarm limit</td><td></td></tr>
</table>

	3. Reset the tank interlock by clicking on the button:  And confirm the action.	
Pass Criteria	2. According to the alarm severity, the tank border should be displayed in MINOR or MAJOR color:  demo.opi BEAST Alarm System Demo Tank Level Control 100 80 60 40 20 0 m3 2013-02-01 07:35:45 07:39:00 07:42:00 2013-02-01 07:45:45 Time TEST-BST0:TANK-LEVEL ☑ Failure of MID switch DRY MID Switch 0 ON 100 80 60 40 20 0 HIHI HI 83.17 m3 LO LOLO WET HIGH Switch WET LOW Switch Interlock Tank Interlock OFF RESET Pump Control System 01/02/2013 07:45:46 Whenever the HIGH switch gets wet, the pump will be forced ON and an audible/visual alarm should be activated Whenever the MID switch gets wet, the pump is activated. The MID switch will become dry again shortly before the level reaches the LOW probe Whenever the LOW switch becomes dry, the pump will be forced OFF and an audible/visual alarm should be activated The three switches within the tank are designated LOW, MID and HIGH. All normal pump activity is carried out by the MID switch After the Interlock Reset, all normal pump activity is carried out again by the MID switch:	

	 demo.opi BEAST Alarm System Demo Tank Level Control Whenever the HIGH switch gets wet, the pump will be forced ON and an audible/visual alarm should be activated Whenever the MID switch gets wet, the pump is activated. The MID switch will become dry again shortly before the level reaches the LOW probe Whenever the LOW switch becomes dry, the pump will be forced OFF and an audible/visual alarm should be activated The three switches within the tank are designated LOW, MID and HIGH. All normal pump activity is carried out by the MID switch Pump Control System 01/02/2013 07:48:54 100 80 60 40 20 0 m3 2013-02-01 07:38:54 07:42:00 07:45:00 2013-02-01 07:48:54 Time TEST-BST0:TANK-LEVEL □ Failure of MID switch DRY MID Switch 0 OFF HIHI HI LO LOLO 34.87 m3 DRY HIGH Switch WET LOW Switch Interlock Tank Interlock OFF RESET	
SRV-04	**3.4.11 JMS Monitor**	
Prerequisite	1. Demo IOC running 2. Alarm Configuration Imported successfully 3. Alarm server started 4. Alarm Operator Interface started	
Test Cases	1. Positive confirmation of JMS notification messages sent	
Procedure	In CSS, monitor JMS messages sent by the “demo” alarm server: 1. Menu CSS -> Debugging -> JMS Monitor 2. In the Topic field of the JMS Monitor View, enter demo_SERVER and press <Enter> Generate an alarm by setting manually the tank level outside its limits. In the previous Linux console: 3. $ caput TEST-BST0:TANK-LEVEL 82 4. Check for new JMS messages in CSS JMS Monitor View 5. Close JMS Monitor View	

Pass Criteria	3. `caput TEST-BST0:TANK-LEVEL 82`
	`Old : TEST-BST0:TANK-LEVEL          xx.xxx`
	`New : TEST-BST0:TANK-LEVEL          82`
	4. IDLE messages and the new HIGH Switch alarm notification message:	



URL: alarm @ failover:(tcp://localhost:61616)?randomize=false

Topic: demo_SERVER

Server: localhost/127.0.0.1:61616

Date	Type	Content
2013-01-29 12:07:31.817	alarm	JMS Source=topic://demo_SERVER, HOST=apps2.codac.iter.org, APPLICATION-ID=AlarmServer, CONFIG=demo, NAME=TEST-BST0:HIGH-SWITCH, EVENTTIME=2013-01-29 10:23:49.385, SEVERIT
2013-01-29 12:07:35.816	alarm	JMS Source=topic://demo_SERVER, HOST=apps2.codac.iter.org, APPLICATION-ID=AlarmServer, CONFIG=demo, NAME=TEST-BST0:HIGH-SWITCH, EVENTTIME=2013-01-29 10:23:49.385, SEVERIT

HST-01

3.4.12 Message History

Prerequisite

1. Demo IOC running
2. Alarm Configuration Imported successfully
3. Alarm server started
4. Alarm Operator Interface started

Test Cases

1. Positive confirmation of alarm message history

Procedure

From the previous Linux console, edit a new jms plugin customisation ini file and enter the following org.csstudio.jms2rdb parameters:

1. $ gedit src/main/beast/jms.ini&

```
org.csstudio.jms2rdb/httpd_port=4914
org.csstudio.jms2rdb/jms_url=failover:(tcp://localhost:61616)
org.csstudio.jms2rdb/jms_topic=demo_SERVER, demo_CLIENT, demo_TALK, LOG, TALK,
WRITE
org.csstudio.jms2rdb/rdb_url=jdbc:postgresql://localhost/css_log_3_0_0?user=log
&password=$log
org.csstudio.jms2rdb/rdb_schema=
```

Save the file in the editor and from the previous Linux console, start jms2rdb to log JMS messages in the RDB:

2. $ jms2rdb -pluginCustomization src/main/beast/jms.ini&

Generate an alarm by setting manually the tank level outside its limits. In the previous Linux console:

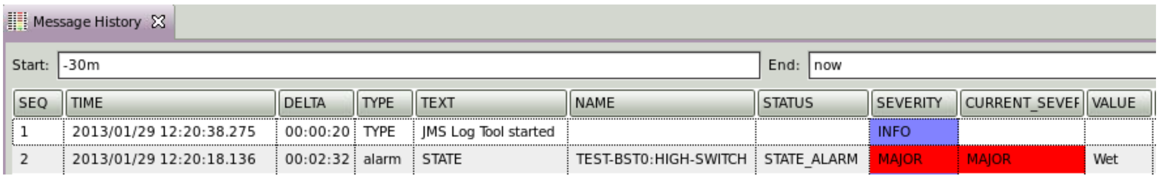
3. $ caput TEST-BST0:TANK-LEVEL 82

In CSS, view logged last 30 minutes JMS messages from RDB:

4. CSS -> Alarm -> Message History
5. In the Start field of the Message History View, enter -30m and press <Enter>
6. Check that the MAJOR alarm was stored in the history

Pass Criteria

6. The output of the command should shows all the message history since 30 minutes:

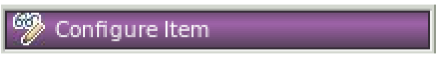


Start: -30m

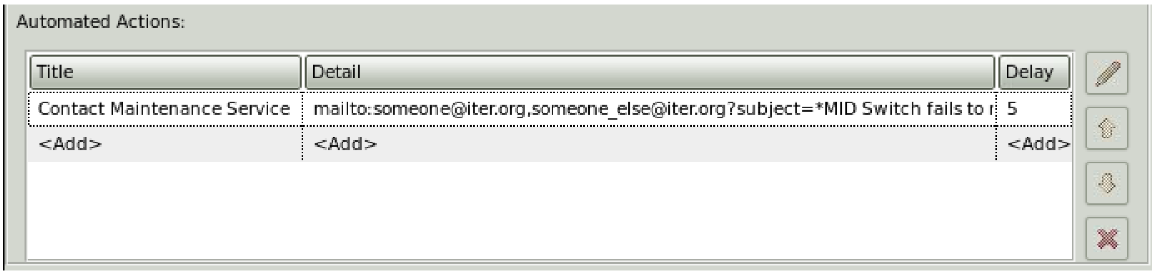
End: now

SEQ	TIME	DELTA	TYPE	TEXT	NAME	STATUS	SEVERITY	CURRENT_SEVEF	VALUE
1	2013/01/29 12:20:38.275	00:00:20	TYPE	JMS Log Tool started			INFO		
2	2013/01/29 12:20:18.136	00:02:32	alarm	STATE	TEST-BST0:HIGH-SWITCH	STATE_ALARM	MAJOR	MAJOR	Wet

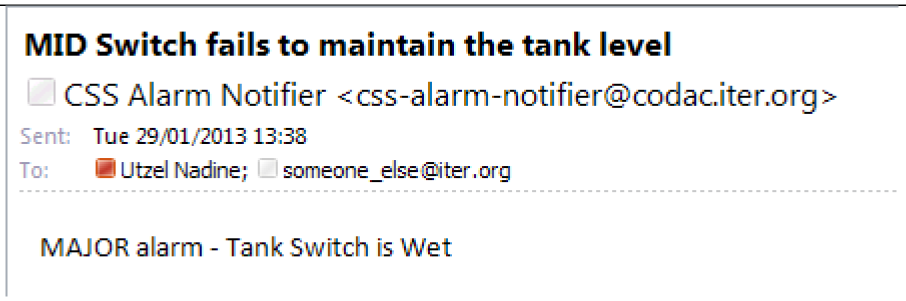
<table>
<tr><td colspan="3"></td></tr>
<tr><td>SRV-05</td><td><b>3.4.13 Automated Actions</b></td><td></td></tr>
<tr><td>Prerequisite</td><td>1. Demo IOC running
2. Alarm Configuration Imported successfully
3. Alarm server and notifier started
4. Alarm Operator Interface started</td><td></td></tr>
<tr><td>Test Cases</td><td>1. Configure an alarm to executed automated actions when triggered
2. When configured alarm is triggered, an email is automatically sent</td><td></td></tr>
<tr><td>Procedure</td><td>In CSS, log in as codac-dev to be able to modify dynamically the alarm configuration:
1. File -> Log in…
2. Enter codac-dev as User Name and type the Password. Click on OK to validate
3. In the Alarm Tree view from the Alarm Perspective, right-click on PV: TEST-BST0:HIGH-SWITCH and select the option Configure Item



4. In the Automated Actions section, modify the predefined Automated Action “Contact Maintenance Service”. In the Detail field, specify your email address in place of someone@iter.org:


Automated Actions:
<table>
<tr><th>Title</th><th>Detail</th><th>Delay</th></tr>
<tr><td>Contact Maintenance Service</td><td>mailto:someone@iter.org,someone_else@iter.org?subject=*MID Switch fails to r</td><td>5</td></tr>
<tr><td><Add></td><td><Add></td><td><Add></td></tr>
</table>

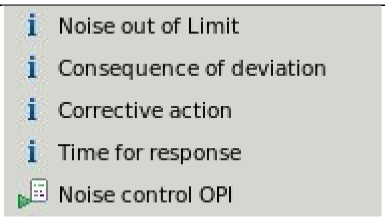
Click on OK to validate the new configuration in the Alarm RDB.
Generate an alarm by setting manually the tank level outside its limits. In the previous Linux console:
5. $ caput TEST-BST0:TANK-LEVEL 92
6. Wait for ~5 seconds and check your inbox</td><td></td></tr>
<tr><td>Pass Criteria</td><td>6. The email automatically sent when the configured alarm is triggered :</td><td></td></tr>
</table>

	
	MID Switch fails to maintain the tank level
	□ CSS Alarm Notifier <css-alarm-notifier@codac.iter.org>
	Sent: Tue 29/01/2013 13:38
	To: Utzel Nadine; someone_else@iter.org
	MAJOR alarm - Tank Switch is Wet
	The output in the Linux console looks like that:
	`xxx INFO [Thread 9663] org.csstudio.alarm.beast.notifier.WorkQueue (add) -`
	`TEST-BST0:HIGH-SWITCH: Contact Maintenance Service => SCHEDULED: 5s`
	`xxx INFO [Thread 9664]`
	`org.csstudio.alarm.beast.notifier.ExecuteActionTask$ExecuteActionThread (run) -`
	`TEST-BST0:HIGH-SWITCH: Contact Maintenance Service => EXECUTED`	

PRF-01	**3.4.14 Configuration load of 600 alarms**	

Prerequisite	1. Performance IOCs downloaded from SVN
	2. You can close the previous EPICS database
	`epics> exit`
	3. Start the EPICS IOC Performance Database:
	`$ softIoc -s -d src/main/epics/TEST-BST1App/Db/PSH0-TEST-BST1.db`	

Test Cases	1. Import the alarm configuration of 600 PVs 2. Check the performance of the resulting Alarm Tree	
Procedure	In the Linux console: 1. $ alarm-configtool -root demo -import -file src/main/beast/TEST-BST1-beast.xml In CSS, close the Alarm Perspective 2. Right-click on the button Alarm And select Close. 3. Reopen the Alarm Perspective to read the new alarm configuration from the database: menu Window -> Open Perspective -> Other… and select Alarm. Change the root element of the Alarm Tree View using the arrow near CODAC_AlarmHandler and select demo. Assess how fast the Alarm Tree View is able to load the 600 alarms. Browse and explore the entire tree. Right-click on any alarm to check and try the different guidance and actions:	

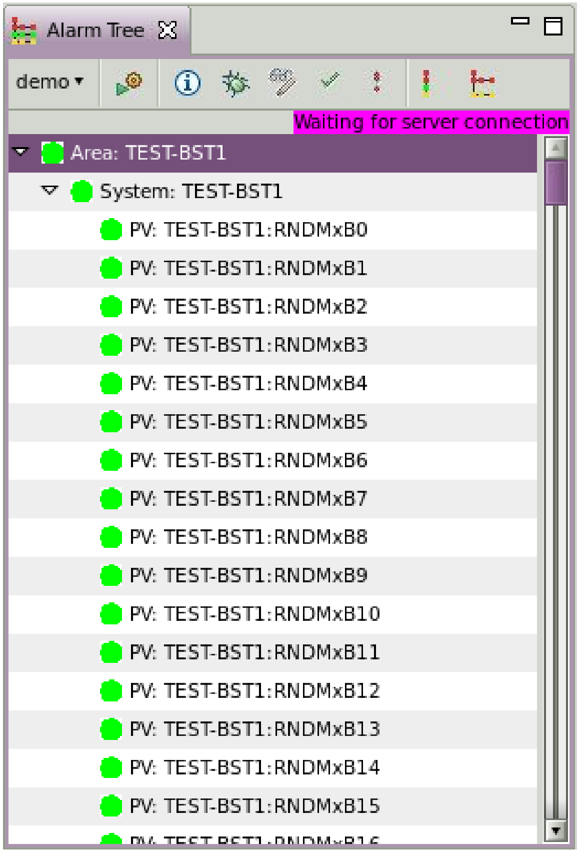


Pass Criteria

1. Only few seconds are required to load the full configuration. The output in the Linux console looks like that:

```
…
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB584
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB585
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB586
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB587
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB588
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB589
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB590
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB591
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB592
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB593
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB594
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB595
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB596
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB597
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB598
Loading /demo/TEST-BST0/TEST-BST1/TEST-BST1:RNDMxB599
```

2. The Alarm Server has not been started yet, but the Alarm configuration is read from the database. The exploration of the Alarm Tree should be reactive:



Waiting for server connection

- Area: TEST-BST1
  - System: TEST-BST1
    - PV: TEST-BST1:RNDMxB0
    - PV: TEST-BST1:RNDMxB1
    - PV: TEST-BST1:RNDMxB2
    - PV: TEST-BST1:RNDMxB3
    - PV: TEST-BST1:RNDMxB4
    - PV: TEST-BST1:RNDMxB5
    - PV: TEST-BST1:RNDMxB6
    - PV: TEST-BST1:RNDMxB7
    - PV: TEST-BST1:RNDMxB8
    - PV: TEST-BST1:RNDMxB9
    - PV: TEST-BST1:RNDMxB10
    - PV: TEST-BST1:RNDMxB11
    - PV: TEST-BST1:RNDMxB12
    - PV: TEST-BST1:RNDMxB13
    - PV: TEST-BST1:RNDMxB14
    - PV: TEST-BST1:RNDMxB15
    - PV: TEST-BST1:RNDMxB16

<table>
<tr><td colspan="3"></td></tr>
<tr><td>PRF-02</td><td>

3.4.15 Important number of Active Alarms

</td><td></td></tr>
<tr><td>Prerequisite</td><td>

1. Performance IOCs started

2. CSS running with Alarm Perspective opened

3. You need first to stop the previous instance of the alarm server:

$ ps -ef|grep demo|grep <user>

```
user    <PID1>  xxx  0 11:12 pts/14   00:00:00 /bin/bash /usr/bin/alarm-server
-root demo
user    <PID2>  xxx  0 11:12 pts/14   00:00:00 /opt/codac-3.1/css/alarm-
server/alarm-server -root demo -pluginCustomization
/tmp/plugin_customization.ini.5994
user    <PID3>  xxx  0 11:12 pts/14   00:00:09 /usr/bin/java -
Djava.awt.headless=true -Xms64m -Xmx256m -Declipse.exitdata= -jar /opt/codac-
3.1/css/alarm-server/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar …
user    <PID1>  xxx  0 11:12 pts/14   00:00:00 /bin/bash /usr/bin/alarm-
notifier -root demo
user    <PID2>  xxx  0 11:12 pts/14   00:00:00 /opt/codac-3.1/css/alarm-
notifier/alarm-notifier -root demo -pluginCustomization
/tmp/plugin_customization.ini.5994
user    <PID3>  xxx  0 11:12 pts/14   00:00:09 /usr/bin/java -
Djava.awt.headless=true -Xms64m -Xmx256m -Declipse.exitdata= -jar /opt/codac-
3.1/css/alarm-notifier/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar
…
```

$ kill -9 <PID1> <PID2> <PID3>

</td><td></td></tr>
<tr><td>Test Cases</td><td>

1. Alarm server running with 600 alarms preconfigured

2. Check the performance of the Alarm Table – Current Alarms

</td><td></td></tr>
<tr><td>Procedure</td><td>

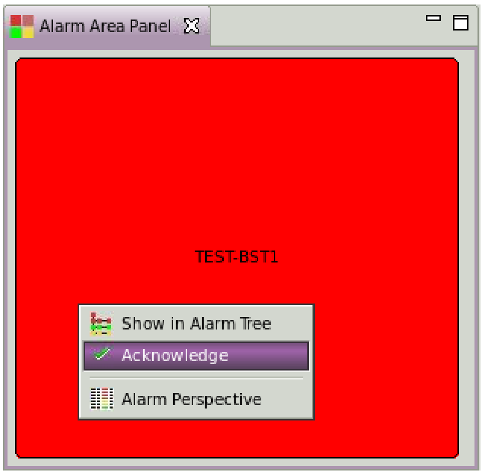
In the Linux console:

1. $ alarm-server -root demo&

In CSS:

2. Check the Alarm Table filled with triggered alarms

3. Acknowledge all the alarms at once with a right-click on the Alarm Area Panel -> Acknowledge option:



</td><td></td></tr>
</table>

Check how the Alarm Table is emptied and how the number of Current Alarms starts to increase again

Pass Criteria

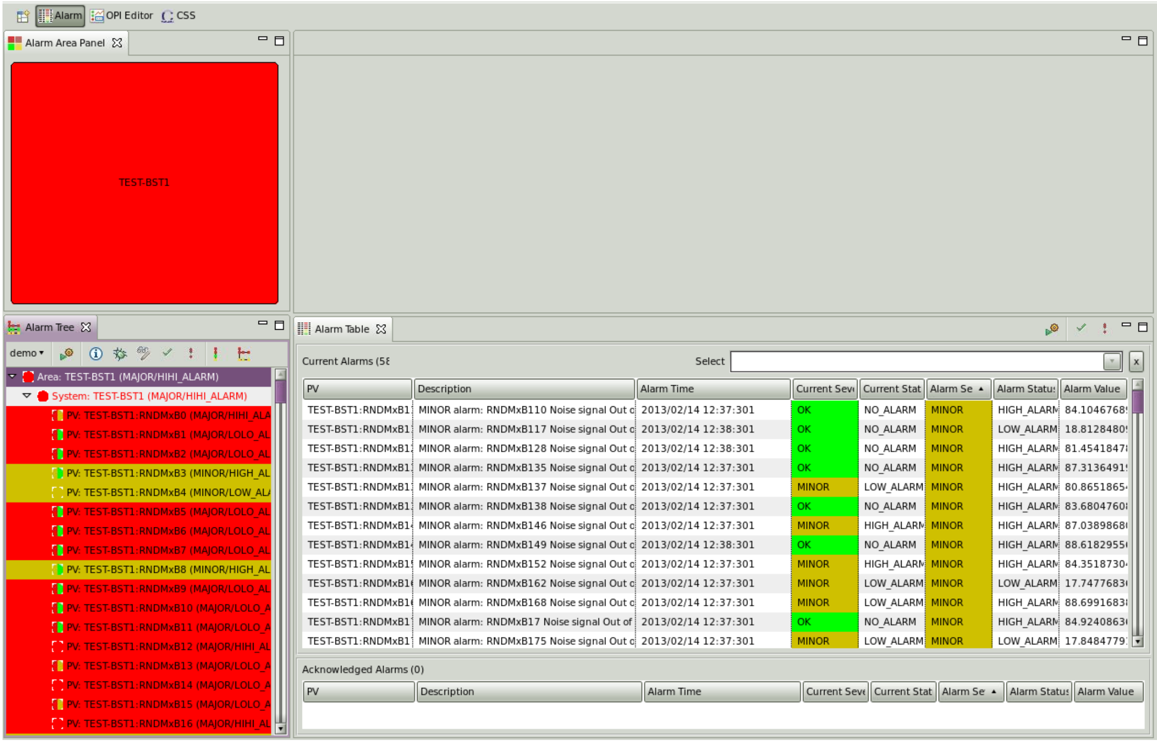
1. Check how fast the 600 PVs have been read. The output in the Linux console looks like that:

```
…
org.csstudio.alarm.beast.server.Application (start) - Alarm Server
1.0.0.codac_core_4_0b7 started for 'demo' configuration
Alarm Server 1.0.0.codac_core_4_0b7
Configuration Root: demo
Database URL:       jdbc:postgresql://localhost/css_alarm_3_0_0
JMS URL:            failover:(tcp://localhost:61616)?randomize=false
JMS Server Topic:   demo_SERVER
JMS Client Topic:   demo_CLIENT
JMS Talk Topic:     demo_TALK
JMS Global Topic:   GLOBAL_SERVER
2013-02-14 12:37:32.453 INFO [Thread 19]
org.apache.activemq.transport.failover.FailoverTransport (doReconnect) -
Successfully connected to tcp://localhost:61616
2013-02-14 12:37:32.921 CONFIG [Thread 1]
org.csstudio.utility.pvmanager.epics.Epics3DataSource (<clinit>) - Loading
epics data source parameters: com.cosylab.epics.caj.CAJContext - 4
2013-02-14 12:37:33.031 CONFIG [Thread 1]
…

Read 600 PVs in 0.xx seconds: yyyy.y PVs/sec

2013-02-14 12:37:33.164 INFO [Thread 23]
org.apache.activemq.transport.failover.FailoverTransport (doReconnect) -
Successfully connected to tcp://localhost:61616
```

2. After some minutes, there are roughly 600 Current Alarms, even if some of them have already recovered



Area: TEST-BST1 (MAJOR/HIHI_ALARM)
System: TEST-BST1 (MAJOR/HIHI_ALARM)
PV: TEST-BST1:RNDMxB0 (MAJOR/HIHI_ALA
PV: TEST-BST1:RNDMxB1 (MAJOR/LOLO_AL
PV: TEST-BST1:RNDMxB2 (MAJOR/LOLO_AL
PV: TEST-BST1:RNDMxB3 (MINOR/HIGH_AL
PV: TEST-BST1:RNDMxB4 (MINOR/LOW_AL
PV: TEST-BST1:RNDMxB5 (MAJOR/LOLO_AL
PV: TEST-BST1:RNDMxB6 (MAJOR/LOLO_AL
PV: TEST-BST1:RNDMxB7 (MAJOR/LOLO_AL
PV: TEST-BST1:RNDMxB8 (MINOR/HIGH_AL
PV: TEST-BST1:RNDMxB9 (MAJOR/LOLO_AL
PV: TEST-BST1:RNDMxB10 (MAJOR/LOLO_A
PV: TEST-BST1:RNDMxB11 (MAJOR/LOLO_A
PV: TEST-BST1:RNDMxB12 (MAJOR/HIHI_AL
PV: TEST-BST1:RNDMxB13 (MAJOR/LOLO_A
PV: TEST-BST1:RNDMxB14 (MAJOR/LOLO_A
PV: TEST-BST1:RNDMxB15 (MAJOR/LOLO_A
PV: TEST-BST1:RNDMxB16 (MAJOR/HIHI_AL

Current Alarms (58

Select

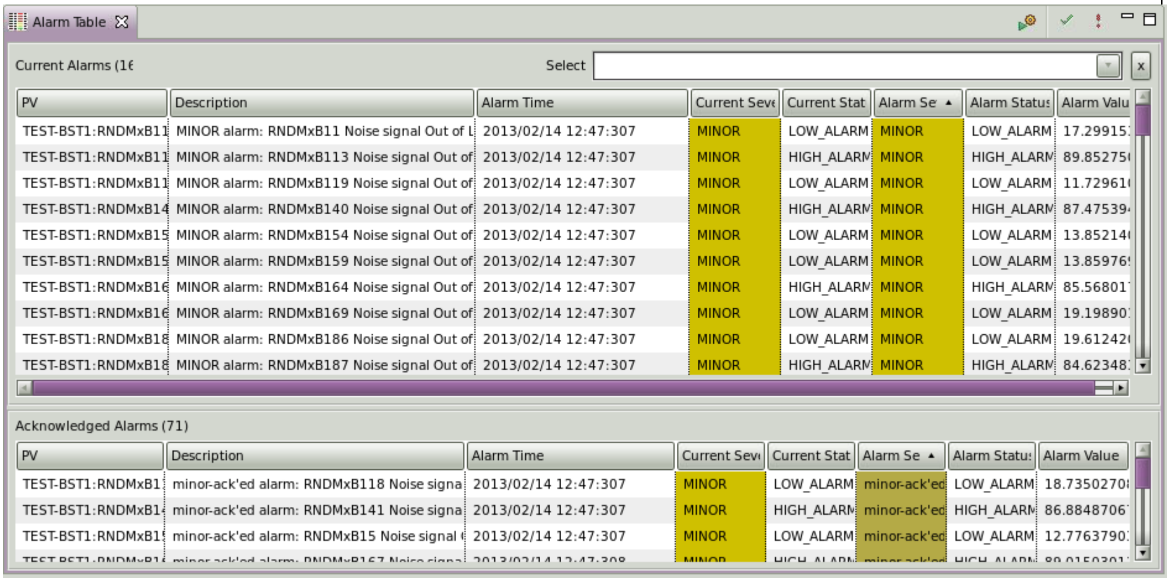
PV	Description	Alarm Time	Current Sev	Current Stat	Alarm Se ▲	Alarm Statu	Alarm Value
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB110 Noise signal Out c	2013/02/14 12:37:301	OK	NO_ALARM	MINOR	HIGH_ALARM	84.1046768
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB117 Noise signal Out c	2013/02/14 12:38:301	OK	NO_ALARM	MINOR	LOW_ALARM	18.8128480
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB128 Noise signal Out c	2013/02/14 12:38:301	OK	NO_ALARM	MINOR	HIGH_ALARM	81.4541847
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB135 Noise signal Out c	2013/02/14 12:37:301	OK	NO_ALARM	MINOR	HIGH_ALARM	87.3136491
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB137 Noise signal Out c	2013/02/14 12:37:301	MINOR	LOW_ALARM	MINOR	HIGH_ALARM	80.8651865
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB138 Noise signal Out c	2013/02/14 12:37:301	OK	NO_ALARM	MINOR	HIGH_ALARM	83.6804760
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB146 Noise signal Out c	2013/02/14 12:37:301	MINOR	HIGH_ALARM	MINOR	HIGH_ALARM	87.0389868
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB149 Noise signal Out c	2013/02/14 12:38:301	OK	NO_ALARM	MINOR	HIGH_ALARM	88.6182955
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB152 Noise signal Out c	2013/02/14 12:37:301	MINOR	HIGH_ALARM	MINOR	HIGH_ALARM	84.3518730
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB162 Noise signal Out c	2013/02/14 12:37:301	MINOR	LOW_ALARM	MINOR	LOW_ALARM	17.7477683
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB168 Noise signal Out c	2013/02/14 12:37:301	MINOR	LOW_ALARM	MINOR	HIGH_ALARM	88.6991683
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB17 Noise signal Out of	2013/02/14 12:37:301	OK	NO_ALARM	MINOR	HIGH_ALARM	84.9240863
TEST-BST1:RNDMxB1	MINOR alarm: RNDMxB175 Noise signal Out c	2013/02/14 12:37:301	MINOR	LOW_ALARM	MINOR	LOW_ALARM	17.8484779

Acknowledged Alarms (0)

PV	Description	Alarm Time	Current Seve	Current Stat	Alarm Se ▲	Alarm Statu	Alarm Value

3. Some seconds are required to empty the Current Alarm Table end fill the Acknowledged Alarms one. As the 600 noise records are processed every 10 seconds,

some other seconds are also required before the number of Current Alarms starts to increase again. Progressively the Acknowledged Alarms becomes empty:



Current Alarms (1€

PV	Description	Alarm Time	Current Seve	Current Stat	Alarm Se ▲	Alarm Status	Alarm Valu
TEST-BST1:RNDMxB11	MINOR alarm: RNDMxB11 Noise signal Out of L	2013/02/14 12:47:307	MINOR	LOW_ALARM	MINOR	LOW_ALARM	17.29915
TEST-BST1:RNDMxB11	MINOR alarm: RNDMxB113 Noise signal Out of	2013/02/14 12:47:307	MINOR	HIGH_ALARM	MINOR	HIGH_ALARM	89.85275
TEST-BST1:RNDMxB11	MINOR alarm: RNDMxB119 Noise signal Out of	2013/02/14 12:47:307	MINOR	LOW_ALARM	MINOR	LOW_ALARM	11.72961
TEST-BST1:RNDMxB14	MINOR alarm: RNDMxB140 Noise signal Out of	2013/02/14 12:47:307	MINOR	HIGH_ALARM	MINOR	HIGH_ALARM	87.47539
TEST-BST1:RNDMxB15	MINOR alarm: RNDMxB154 Noise signal Out of	2013/02/14 12:47:307	MINOR	LOW_ALARM	MINOR	LOW_ALARM	13.85214
TEST-BST1:RNDMxB15	MINOR alarm: RNDMxB159 Noise signal Out of	2013/02/14 12:47:307	MINOR	LOW_ALARM	MINOR	LOW_ALARM	13.85976
TEST-BST1:RNDMxB16	MINOR alarm: RNDMxB164 Noise signal Out of	2013/02/14 12:47:307	MINOR	HIGH_ALARM	MINOR	HIGH_ALARM	85.56801
TEST-BST1:RNDMxB16	MINOR alarm: RNDMxB169 Noise signal Out of	2013/02/14 12:47:307	MINOR	LOW_ALARM	MINOR	LOW_ALARM	19.19890
TEST-BST1:RNDMxB18	MINOR alarm: RNDMxB186 Noise signal Out of	2013/02/14 12:47:307	MINOR	LOW_ALARM	MINOR	LOW_ALARM	19.61242
TEST-BST1:RNDMxB18	MINOR alarm: RNDMxB187 Noise signal Out of	2013/02/14 12:47:307	MINOR	HIGH_ALARM	MINOR	HIGH_ALARM	84.62348

Acknowledged Alarms (71)

PV	Description	Alarm Time	Current Sev	Current Stat	Alarm Se ▲	Alarm Status	Alarm Value
TEST-BST1:RNDMxB1	minor-ack'ed alarm: RNDMxB118 Noise signa	2013/02/14 12:47:307	MINOR	LOW_ALARM	minor-ack'ed	LOW_ALARM	18.7350270
TEST-BST1:RNDMxB1	minor-ack'ed alarm: RNDMxB141 Noise signa	2013/02/14 12:47:307	MINOR	HIGH_ALARM	minor-ack'ed	HIGH_ALARM	86.8848706
TEST-BST1:RNDMxB1	minor-ack'ed alarm: RNDMxB15 Noise signal	2013/02/14 12:47:307	MINOR	LOW_ALARM	minor-ack'ed	LOW_ALARM	12.7763790
TEST-BST1:RNDMxB1	minor-ack'ed alarm: RNDMxB167 Noise signa	2013/02/14 12:47:308	MINOR	HIGH_ALARM	minor-ack'ed	HIGH_ALARM	89.0150301

To terminate the tests, stop the demo IOC, the demo alarm server and jms2rdb. Close css:

1. $ epics> exit

2. $ ps -ef|grep demo|grep <user>

```
user    <PID1>  xxx  0 11:12 pts/14   00:00:00 /bin/bash /usr/bin/alarm-server -root demo
user    <PID2>  xxx  0 11:12 pts/14   00:00:00 /opt/codac-3.1/css/alarm-server/alarm-server -
root demo -pluginCustomization /tmp/plugin_customization.ini.5994
user    <PID3>  xxx  0 11:12 pts/14   00:00:09 /usr/bin/java -Djava.awt.headless=true -Xms64m -
Xmx256m -Declipse.exitdata= -jar /opt/codac-3.1/css/alarm-
server/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar …
user    <PID1>  xxx  0 11:12 pts/14   00:00:00 /bin/bash /usr/bin/alarm-notifier -root demo
user    <PID2>  xxx  0 11:12 pts/14   00:00:00 /opt/codac-3.1/css/alarm-notifier/alarm-notifier
-root demo -pluginCustomization /tmp/plugin_customization.ini.5994
user    <PID3>  xxx  0 11:12 pts/14   00:00:09 /usr/bin/java -Djava.awt.headless=true -Xms64m -
Xmx256m -Declipse.exitdata= -jar /opt/codac-3.1/css/alarm-
notifier/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar …
```

3. $kill -9 <PID1> <PID2> <PID3>

4. $ ps -ef|grep jms2rdb|grep <user>

```
user    <PID1>  xxx  0 13:18 pts/14   00:00:00 /bin/bash /usr/bin/jms2rdb -pluginCustomization
jms.ini
user    <PID2>  xxx  0 13:18 pts/14   00:00:00 /opt/codac-3.1/css/jms2rdb/jms2rdb -
pluginCustomization /tmp/plugin_customization.ini.9273
user    <PID3>  xxx 16 13:18 pts/14   00:00:03 /usr/bin/java -Declipse.exitdata= -jar
/opt/codac-3.1/css/jms2rdb/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar -os linux -
ws gtk -arch x86_64 -showsplash -launcher /opt/codac-3.1/css/jms2rdb/jms2rdb -name Jms2rdb --
launcher.library /opt/codac-
3.1/css/jms2rdb/plugins/org.eclipse.equinox.launcher.gtk.linux.x86_64_1.1.100.v20110505/eclipse
_1407.so -startup /opt/codac-
3.1/css/jms2rdb/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar --
launcher.overrideVmargs -exitdata 888059 -pluginCustomization
/tmp/plugin_customization.ini.9273 -vm /usr/bin/java -vmargs -Declipse.exitdata= -jar
/opt/codac-3.1/css/jms2rdb/plugins/org.eclipse.equinox.launcher_1.2.0.v20110502.jar
user    xxx  xxx  0 13:18 pts/14   00:00:00 grep jms2rdb
```

5. $ kill -9 <PID1> <PID2> <PID3>

6. In CSS, use the menu

File -> Exit

7. Delete the alarm configuration in the database:

$ alarm-configtool -root demo -delete /demo

```
Alarm Config Tool 1.0.0.codac_core_xxx
2013-01-24 09:41:10.921 INFO [Thread 19]
org.apache.activemq.transport.failover.FailoverTransport (doReconnect) - Successfully connected
to tcp://localhost:61616
Reading RDB configuration of 'demo'
Deleting existing RDB configuration for '/demo'
```

3.5 Component Test Log

CFG-01	**3.5.1 Alarm Configuration Import**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
CFG-02	**3.5.2 Alarm Configuration Export**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
SRV-01	**3.5.3 Alarm Server Startup**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
DSP-01	**3.5.4 Alarm PV Tree**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
DSP-02	**3.5.5 Alarm Related Display – Operator Interface**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
DSP-03	**3.5.6 Alarm Related Display – Data Plot**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		

SRV-02	**3.5.7 Alarm Notification**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
DSP-04	**3.5.8 Alarm Guidance**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
SRV-03	**3.5.9 Latched Alarm**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
DSP-05	**3.5.10 Display Limit Alarms**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
SRV-04	**3.5.11 JMS Monitor**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
HST-01	**3.5.12 Message History**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		

SRC-05	**3.5.13 Automated Actions**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
PRF-01	**3.5.14 Configuration load of 600 alarms**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		
PRF-02	**3.5.15 Important number of Active Alarms**	[PASS / FAIL]
[Bug ID]	[Bug title to briefly describe the anomaly]	
Remarks		

Software Test Plan Checklist

For Assessment of:	
Agency Name	
Project Name	
Document Name	
Date	

Criteria	Yes / No / NA
DOCUMENT STANDARDS COMPLIANCE	
1 Have standards/guidelines been identified to define the work product?	
2 Does the work product format conform to the specified standard/guideline (Template)?	
3 Has the project submitted any request for deviations or waivers to the defined work product?	
4 Have the following areas been addressed completely:	
4a Approval authority?	
4b Revision approval?	
4c Revision control?	
TECHNICAL REFERENCE	
5 Is there evidence that the work product was reviewed by all stakeholders?	
6 Have acceptance criteria been established for the work product?	
7 Does the work product have a clearly defined purpose and scope?	
8 Are references to policies, directives, procedures, standards, and terminology provided?	
9 Does the work product identify any and all constraints/limitations?	
S/W TEST PLAN CONTENTS	
10 Does the S/W Test Plan address the following required information:	
10a Test levels?	
10b Test types (e.g., unit testing, software integration testing, systems integration testing, end-to-end testing, acceptance testing, regression testing)?	
10c Test classes?	
10d General test conditions?	
10e Test progression?	
10f Data recording, reduction, and analysis?	
10g Test coverage (breadth and depth) or other methods for ensuring sufficiency of testing?	
10h Planned tests, including items and their identifiers?	

Criteria	Yes / No / NA
10i Test schedules, Requirements traceability (or verification matrix)?	
10j Qualification testing environment, site, personnel, and participating organizations?	
11 Does the S/W Test Plan identify the environmental exposure as well as requirements for comprehensive, functional, aliveness, end-to-end, and mission simulation testing?	
12 Does the S/W Test Plan provide a System Overview that describes the unique complexities of the system?	
13 Does the S/W Test Plan address user guide, operations / maintenance validation?	
16 Does the S/W Test Plan identify any elements that will not be tested according to the test plan (e.g., externally developed software)?	
17 Does the S/W Test Plan address software architecture in terms of which software components will be based on heritage and which will be mostly or entirely new developments?	
18 Does the S/W Test Plan identify any software reuse? If so, is the extent of reuse or the anticipated modification described?	
S/W TEST ENVIRONMENT	
19 Does the S/W Test Plan include a figure of each system test environment? If so, does it reflect the system hardware approach, simulators, and special development?	
20 Does the S/W Test Plan identify specific test hardware and simulators for each external interface?	
TEST TOOLS	
21 Does the S/W Test Plan address test execution tools?	
TEST PROBLEM REPORTING & CORRECTIVE ACTION	
22 Does the S/W Test Plan provide a description of the problem reporting system to be used by the test team to report problems and/or recommended changes cited during the test activities?	
TEST PROGRESS PLANNING & TRACKING	
23 Does the S/W Test Plan describe the routine test progress reporting approach?	
24 Does the S/W Test Plan describe the Build Test verification methodology? If so, does the description address build verification test level objectives, environment, roles & responsibilities, entry/exit criteria, general guidelines, build test planning, build test scenario development, build test procedure preparation & dry run, build test execution, reporting, and archiving?	