
hp Unified Correlation Analyzer



Unified Correlation Analyzer for Event Based Correlation

Version 3.2

Value Pack Examples

Edition: 1.0

April 2015

Legal Notices

Warranty

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

License Requirement and U.S. Government Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

© Copyright 2015 Hewlett-Packard Development Company, L.P.

Trademark Notices

Adobe®, Acrobat® and PostScript®s are trademarks of Adobe Systems Incorporated.

HP-UX Release 10.20 and later and HP-UX Release 11.00 and later (in both 32 and 64-bit configurations) on all HP 9000 computers are Open Group UNIX 95 branded products.

Java™ is a trademark of Oracle and/or its affiliates.

Microsoft®, Windows® and Windows NT® are U.S. registered trademarks of Microsoft Corporation.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

UNIX® is a registered trademark of The Open Group.

X/Open® is a registered trademark, and the X device is a trademark of X/Open Company Ltd. in the UK and other countries.

Contents

Preface	7
Chapter 1.....	9
Introduction	9
Chapter 2.....	10
Low Level Event Filter Value Pack	10
2.1 Software Prerequisites	11
2.2 Deploying the Low Level Event Filtering Value Pack.....	11
2.2.1 Installing the Value Pack	11
2.2.2 Deploying the Value Pack.....	12
2.2.3 Starting the Low Level Event Filtering Value Pack	12
2.2.4 Stopping the Low Level Event Filtering Value Pack.....	13
2.2.5 Un-deploying the Low Level Event Filtering Value Pack	13
2.3 Low Level Event Filtering Value Pack Scenarios	14
2.3.1 The Time Wait scenario	14
2.3.2 The Statistical scenario	15
2.3.3 The Grouping scenario.....	17
2.3.4 The Inactivity scenario	19
2.3.5 The Up/Down scenario	21
2.4 Testing the Low Level Event Filtering Value Pack	22
Chapter 3.....	24
An “Orchestration of Scenarios Cascading” Value Pack	24
3.1 The “Orchestration of Scenarios Cascading” Value Pack Description	24
3.1.1 The Scenarios taking part in the Orchestration	24
3.1.2 The Orchestration Routes	25
3.1.3 The Orchestration Event Flow	26
3.1.4 The Software Prerequisites.....	26
3.2 Deploy and start the “Orchestration of Scenarios Cascading” Value Pack	27
3.2.1 Install the Value Pack	27
3.2.2 Deploy the Value Pack.....	27
3.2.3 Set the Orchestration Routes.....	28
3.2.4 Start the Value Pack	29
3.3 Stop and undeploy the “Orchestration of Scenarios Cascading” Value Pack.....	29
3.3.1 Stop the Value Pack.....	30
3.3.2 Undeploy the Value Pack.....	30
3.4 Test the “Orchestration of Scenarios Cascading” Value Pack	31
3.4.1 Event sample files	31
3.4.2 Injecting events with the uca-ebc-injector	31
3.4.3 Results.....	32
3.4.4 Checking the results	32
Chapter 4.....	36
An “Orchestration of Scenarios Cascading in JOIN routes” Value Pack	36

4.1	The “Orchestration of Scenarios Cascading in JOIN Routes” Value Pack Description	36
4.1.1	The Scenarios taking part in the Orchestration	37
4.1.2	The Orchestration Routes	38
4.1.3	The Orchestration Event Flow	39
4.1.4	The Software Prerequisites	39
4.2	Deploy and start the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack	39
4.2.1	Install the Value Pack	40
4.2.2	Deploy the Value Pack	40
4.2.3	Set the Orchestration Routes	41
4.2.4	Start the Value Pack	42
4.3	Stop and undeploy the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack	42
4.3.1	Stop the Value Pack	42
4.3.2	Undeploy the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack	43
4.4	Test the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack	43
4.4.1	Event sample files	43
4.4.2	Injecting events with the uca-ebc-injector	44
4.4.3	Results	44
4.4.4	Checking the results	44
Chapter 5		60
The “Persistence Example” explained		60
5.1	How does it work?	60
5.2	Installing the example	60
5.3	Looking at the configuration	61
5.4	Testing the value pack	61

Figures

Figure 1 - Alarm flow in Low Level Event Filtering Value Pack.....	10
Figure 2 - Time Wait – Both Fault and Clearance are discarded	14
Figure 3 - Time Wait – Fault or Clearance is kept	15
Figure 4 - Statistical – Number of faults is above the threshold.....	16
Figure 5 - Statistical – Number of faults is below the threshold.....	16
Figure 6 - Statistical – Number of faults is several times the threshold.....	17
Figure 7 - Grouping – No alarm clearance received during the time window	18
Figure 8 - Grouping – No symptom alarm received during the time window	18
Figure 9 - Grouping – Clearance on symptom alarms received during the time window	19
Figure 10 - Grouping – Clearance on root cause and symptom alarms received during the time window	19
Figure 11 - Inactivity– Inactivity detected	20
Figure 12 - Inactivity – Inactivity not detected	21
Figure 13 - Inactivity – Inactivity detection in mix technology context	21
Figure 14 - Up/Down – Independent clearance	22
Figure 15 - Up/Down – Independent clearance cascaded with Time Wait scenario	22
Figure 16 - OrchestraConfigurationCascadingExample.xml Routes	26
Figure 17 - Event flow in “Orchestration of Scenarios Cascading” Value Pack	26
Figure 18 - Event cascading in “Orchestration of Scenarios Cascading” Value Pack test example	32
Figure 19 - OrchestraConfigurationCascadingJoinExample.xml Routes	38
Figure 20 - Event flow in “Orchestration of Scenarios Cascading in JOIN routes” Value Pack	39

Tables

Table 1 - Software versions8

Table 2 - File structure of the LLEF value pack..... 12

Table 3 - File structure of the “Orchestration of Scenarios Cascading” Value Pack 28

Table 4 - File Structure of the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack41

Preface

This guide provides some examples of Unified Correlated Analyzer for Event Based Correlation (EBC) Value Packs.

Such examples should be taken as good practice examples for developing new Value Packs.

Product Name: Unified Correlation Analyzer for Event Based Correlation (also referred in this document as UCA for EBC)

Product Version: 3.2

Kit Version: V3.2

Intended Audience

Here are some recommendations based on possible reader profiles:

- Solution Developers and integrators
- Software Development Engineers

Software Versions

The term UNIX is used as a generic reference to the operating system, unless otherwise specified.

The software versions referred to in this document are as follows:

Product Version	Supported Operating systems
UCA for Event Based Correlation Server Version V3.2	<ul style="list-style-type: none">• HP-UX 11.31 for Itanium• Red Hat Enterprise Linux Server release 5.9 & 6.5
UCA for Event Based Channel Adapter V3.2	<ul style="list-style-type: none">• HP-UX 11.31 for Itanium• Red Hat Enterprise Linux Server release 5.9 & 6.5
UCA for Event Based Correlation Software Development Kit Version V3.2	<ul style="list-style-type: none">• Windows XP / Vista• Windows Server 2007• Windows 7

Table 1 - Software versions

Typographical Conventions

Courier Font:

- Source code and examples of file contents.
- Commands that you enter on the screen.
- Pathnames
- Keyboard key names

Italic Text:

- Filenames, programs and parameters.
- The names of other documents referenced in this manual.

Bold Text:

- To introduce new terms and to emphasize important words.

Associated Documents

The following documents contain useful reference information:

References

[R1] *UCA for EBC Reference Guide*

Support

Please visit our HP Software Support Online Web site at <https://softwaresupport.hp.com/> for contact information, and details about HP Software products, services, and support.

The Software support area of the Software Web site includes the following:

1. Downloadable documentation,
2. Troubleshooting information,
3. Patches and updates,
4. Problem reporting,
5. Training information,
6. Support program information.

Chapter 1

Introduction

This guide gives some examples of standard correlation value packs developed for the UCA for Event Based Correlation product.

Throughout this document, we use the `${UCA_EBC_HOME}` environment variable to reference the root directory (“static” part) of UCA for EBC. The default value for the `${UCA_EBC_HOME}` environment variable is `/opt/UCA-EBC`. The `${UCA_EBC_HOME}` environment variable thus references the `/opt/UCA-EBC` directory unless UCA for EBC “static” part has been installed in an alternate directory.

We also use `${UCA_EBC_DATA}` environment variable to reference the data directory (“variable” part) of UCA for EBC. The default value for the `${UCA_EBC_DATA}` environment variable is `/var/opt/UCA-EBC`. The `${UCA_EBC_DATA}` environment variable thus references the `/var/opt/UCA-EBC` directory unless UCA for EBC “variable” part has been installed in an alternate directory.

Since UCA-EBC V2.0, the `${UCA_EBC_DATA}` directory may contain multiple instances of UCA-EBC. In this document, we will use the value `${UCA_EBC_INSTANCE}` for referring to `${UCA_EBC_DATA}/instances/<instance-name>` directory. At installation, a single `<instance-name>` is configured: *default*.

Low Level Event Filter Value Pack

The Low Level Event Filter Value pack delivers a predefined set of Scenarios that demonstrate event stream processing and provide standard low-level alarm filtering capability.

The following correlation scenarios are provided:

1. **Time Wait:** this low-level filtering scenario discards fault alarms and their associated alarm clearances if they are received during a configurable time window.
2. **Statistical:** this low-level filtering scenario counts all fault alarms received during the configurable time window. If the total number of faults reaches a configurable threshold, then a statistic alarm is generated.
3. **Grouping:** this low-level filtering scenario is similar to the “Time Wait” scenario, but adds root cause correlation functionality by grouping alarms.
4. **Inactivity:** this low-level filtering scenario identifies when a specific technology stops forwarding raw messages.
5. **Up/Down:** this low-level filtering scenario is a complementary scenario, created for the “Time Wait” and “Grouping” scenarios. It handles alarms that have a common clearance alarm.

In the Low Level Event Filter Value Pack, the input Alarms flow is dispatched to the different scenarios according to the following figure:

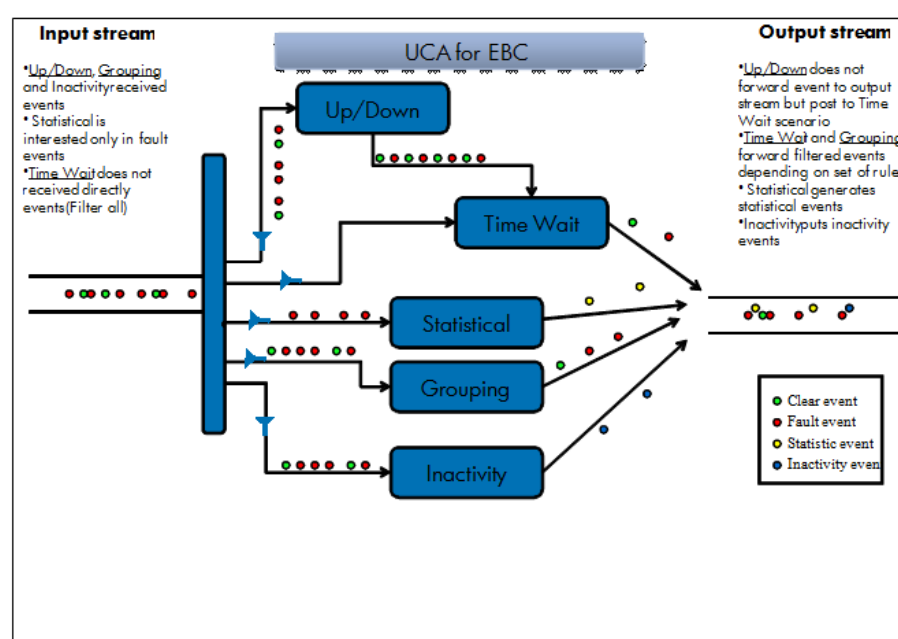


Figure 1 - Alarm flow in Low Level Event Filtering Value Pack

The following assumptions have been used:

- A cleared alarm is an alarm that has the “Perceived Severity” attribute equal to “CLEAR”
- Parent (Root Cause) alarms are identified by the “AddText” attribute equal to “Root Cause ...”
- Child alarms are identified by the “AddText” attribute equal to “Symptom ...”
- Common clear alarms (that need to ‘clears’ several faults) are identified by the “AddText” attribute equal to “Common clearance ...”

2.1 Software Prerequisites

The Low Level Event Filtering Value Pack is delivered with the UCA for EBC Development product under the `vp-examples` directory:

```
${UCA_EBC_DEV_HOME}/vp-examples/llf-example
```

2.2 Deploying the Low Level Event Filtering Value Pack

Several steps are needed to deploy an EVP (UCA for EBC Value Pack):

1. Install the EVP package in the `${UCA_EBC_INSTANCE}/valuepacks` directory (`${UCA_EBC_INSTANCE}` translates to `/var/opt/UCA-EBC/instances/<instance name>` by default unless UCA for EBC was installed at an alternate location)
2. Deploy the Value Pack
3. Start the Value Pack.

2.2.1 Installing the Value Pack

The Low Level Event Filtering EVP package example is installed with the UCA for EBC Server kit. The Low Level Event Filtering EVP is located in the `${UCA_EBC_HOME}/defaults/valuepacks` directory. You will need to copy the Low Level Event Filtering Value Pack zip file (named `llf-example-vp-3.2.zip`) to the `${UCA_EBC_DATA}/valuepacks` directory, so that it can be seen by UCA for EBC.

Alternatively, if you have installed the UCA for EBC Development Toolkit, you can (modify and) re-build the Low Level Event Filtering EVP from the source code by executing the following commands:

On Windows:

```
$ cd %UCA_EBC_DEV_HOME%\vp-examples\llf-example
$ ant all
```

On Linux:

```
$ cd ${UCA_EBC_DEV_HOME}/vp-examples/llf-example
$ ant all
```

When rebuilt, the package is ready to be deployed on UCA for EBC. You just need to copy the Value Pack package you have just generated to the `${UCA_EBC_INSTANCE}/valuepacks` directory.

2.2.2 Deploying the Value Pack

To deploy the Low Level Event Filtering Value Pack, please use the “--deploy” option of the **uca-ebc-admin** command-line administration tool (executed as **‘uca’** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --deploy -vpn llef-example -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Value Pack name: Llef-example version: 3.2 has been
successfully deployed
INFO - Exiting...
```

Or simply deploy the Value Pack from the UCA for EBC User Interface.

2.2.2.1 File organization

At the end of the deployment step, the files delivered by the Value Pack are deployed in `${UCA_EBC_INSTANCE}/deploy/llef-example-3.2` directory, according to the following file structure:

Directories	Description
<i>lib/</i>	Some additional jar files are installed for this package
<i>conf/</i>	Configuration files that defines the Value Pack, and the scenarios
<i>grouping/</i>	Directory containing all files defining Grouping scenario
<i>inactivity/</i>	Directory containing all files defining Inactivity scenario
<i>statistical/</i>	Directory containing all files defining Statistical scenario
<i>timewait/</i>	Directory containing all files defining Time Wait scenario
<i>updown/</i>	Directory containing all files defining Up/Down scenario

Table 2 - File structure of the LLEF value pack

2.2.3 Starting the Low Level Event Filtering Value Pack

Value Packs can be started in two different manners depending on whether UCA for EBC is already started or not.

If UCA for EBC is stopped, restarting the application will automatically start all Value Packs deployed in the `${UCA_EBC_INSTANCE}/deploy` directory.

If UCA for EBC is already running, use the “--start” option of the **uca-ebc-admin** command-line administration tool (executed as **‘uca’** user) to start the Value Pack:

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --start -vpn llef-example -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Exiting...
```

Or simply start it from the UCA for EBC Web User Interface.

2.2.4 Stopping the Low Level Event Filtering Value Pack

You can stop the Value Pack when UCA for EBC is running using the “--stop” option of the **uca-ebc-admin** command-line administration tool (executed as **uca** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --stop -vpn llef-example -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Exiting...
```

Or simply stop it from the UCA for EBC Web User Interface.

2.2.5 Un-deploying the Low Level Event Filtering Value Pack

To undeploy the Low Level Event Filtering Value Pack, use the “--undeploy” option of the **uca-ebc-admin** command-line administration tool (executed as **uca** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --undeploy -vpn llef-example -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Value Pack name: llef example version: 3.2
has been successfully undeployed
INFO - Exiting...
```

Or simply un-deploy it from the UCA for EBC Web User Interface.

2.3 Low Level Event Filtering Value Pack Scenarios

2.3.1 The Time Wait scenario

2.3.1.1 Functional description

The Time Wait scenario receives fault alarms and waits during a configurable time period for a clear alarm to arrive. If a clear alarm is received during the time period both alarm and alarm clearance are discarded and none of them are forwarded to the Trouble Ticket (TT) generator external application. The purpose of this scenario is to avoid unreasonable trouble ticket to be opened due to unusual situations, such as a temporary system overload or an equipment restart.

The main rules in the Time Wait scenario are:

- Discard both fault alarm and fault alarm clearance when both are received during the time wait period
- Keep fault alarms (or fault alarm clearance) when no corresponding fault alarm clearance (or fault alarm) is received during the time wait period

The following sections describe the possible use cases in detail.

Note

The rules are parameterized. You can set the value for the following parameters in the *timewait-params.xml* file:

- **timewait:** duration of the time wait period (the default value is 2 seconds)

2.3.1.2 Both Fault and Clearance are discarded

This use case applies when both the alarm and the associated alarm clearance are received during the time wait period:

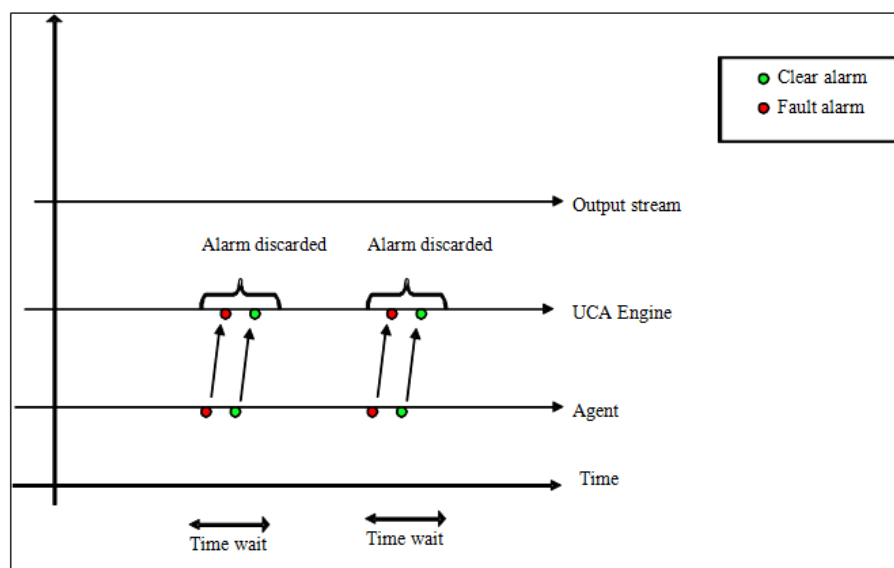


Figure 2 - Time Wait – Both Fault and Clearance are discarded

2.3.1.3 Fault or Clearance is kept

This use case applies when an alarm is not cleared during the time wait period or when a clearance is received without the corresponding alarm during the time wait period:

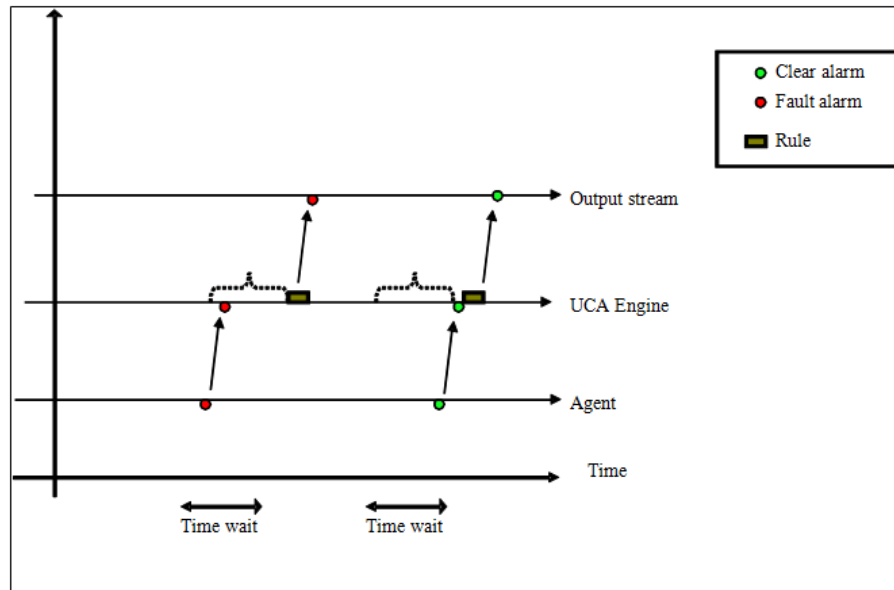


Figure 3 - Time Wait – Fault or Clearance is kept

2.3.2 The Statistical scenario

2.3.2.1 Functional description

The Statistical scenario counts all fault alarms received during a configurable period of time. If the total number of alarms reaches a configurable threshold, then a statistical alarm is generated. If not, then no statistical alarm is generated. In any case the original alarms are discarded.

The main rules in the Statistical scenario are:

- Create a Statistical alarm whenever a configurable number of alarms (alarm clearances excepted) is received during a configurable time window
- Don't create any Statistical alarm if the number of alarms (alarm clearances excepted) received during a configurable time window does not reach a configurable number of alarms (threshold)
- Discard all original alarms

The following sections describe the possible use cases in detail.

Note

The rules are parameterized. You can set the value for the following parameters in the *statistical-params.xml* file:

- **threshold:** number of alarms required before sending a Statistical output alarm (the default value is 3 alarms)
- **timewindow:** duration of the time window period during which alarms are counted (the default value is 5 seconds)

2.3.2.2 Number of faults is above the threshold

This use case applies when the number of alarms received during the configurable *timewindow* is higher than the *threshold* parameter:

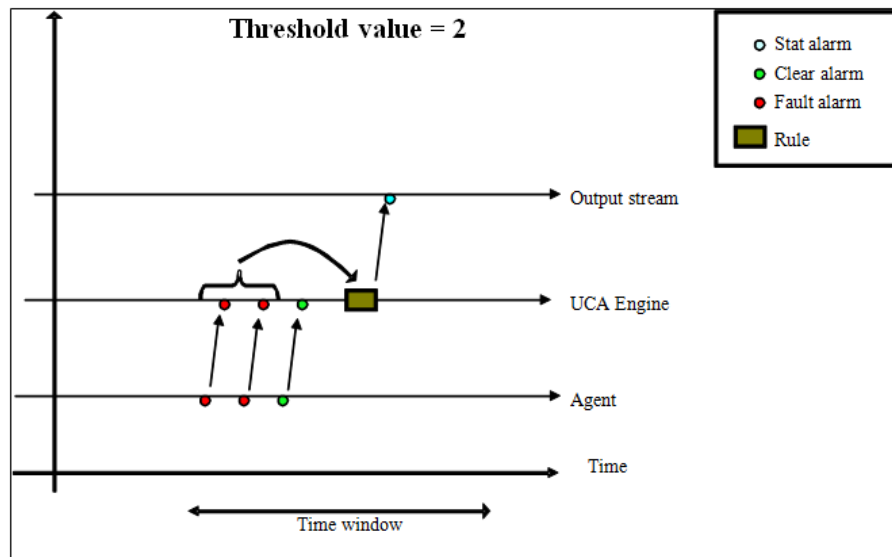


Figure 4 - Statistical – Number of faults is above the threshold

2.3.2.3 Number of faults is below the threshold

This use case applies when the number of alarms received during the configurable *timewindow* is lower than the *threshold* parameter:

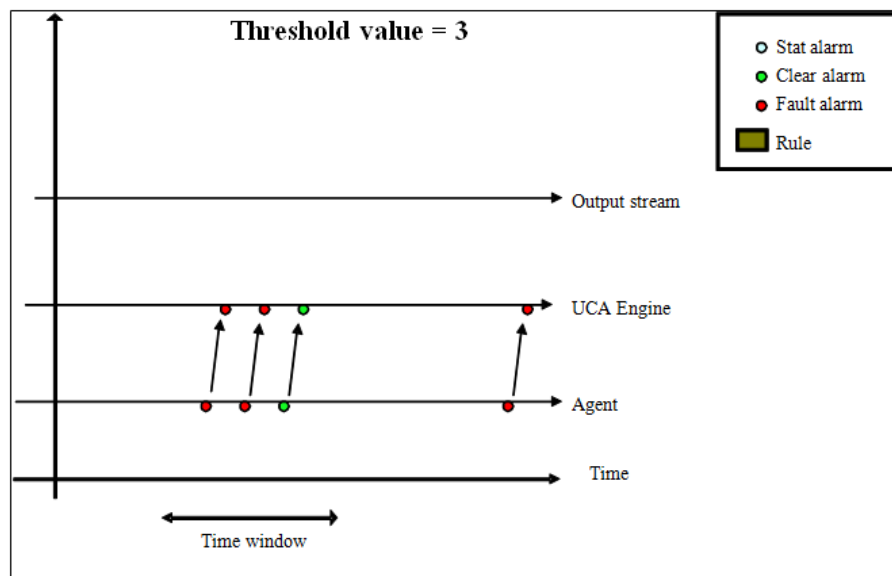


Figure 5 - Statistical – Number of faults is below the threshold

2.3.2.4 Number of faults is several times the threshold

This use case applies when the number of alarms received during the configurable *timewindow* is equal to several times the value of the *threshold* parameter:

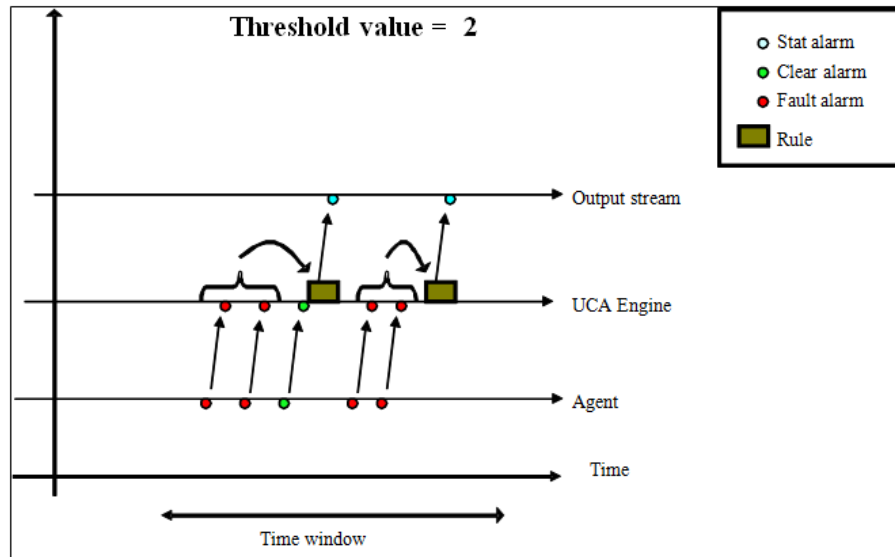


Figure 6 - Statistical – Number of faults is several times the threshold

2.3.3 The Grouping scenario

2.3.3.1 Functional description

Besides discarding fault alarms that are cleared during a configurable period of time, the Grouping scenario groups fault alarms that share another alarm as a common root cause. The root cause alarm is called “parent” and symptom alarms are called “children”.

The scenario works during a configurable period of time, starting at the receipt of a root cause alarm. During this period, if a “child” alarms is received, the ‘parent’ alarm is enriched with information from the “child” alarm. The “child” alarm is discarded and the “parent” alarm is forwarded.

If no “child” alarm is received, the “parent” alarm is forwarded unchanged. If an alarm clearance is received during the configurable time period, only the cleared alarm is discarded. If the “parent” alarm is cleared, it is discarded and all not-cleared “child” alarms are forwarded.

The following sections describe the possible use cases in detail.

Note

The rules are parameterized. You can set the value for the following parameters in the *grouping-params.xml* file:

- **timeslot:** duration of the time window period (the default value is 10 seconds)

2.3.3.2 No alarm clearance received during the time window

This use case applies when the root cause alarm is followed by child alarms:

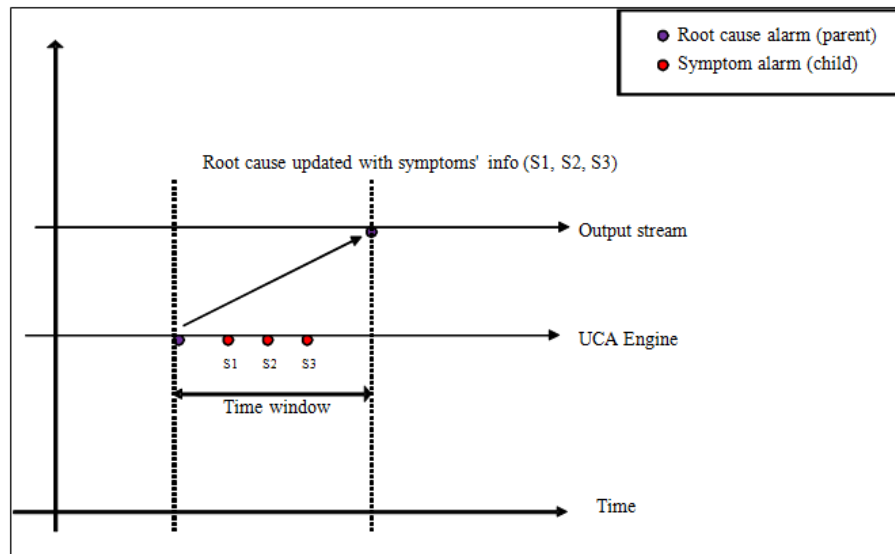


Figure 7 - Grouping – No alarm clearance received during the time window

2.3.3.3 No symptom alarm received during the time window

This use case applies when the root cause alarm is not followed by child alarms:

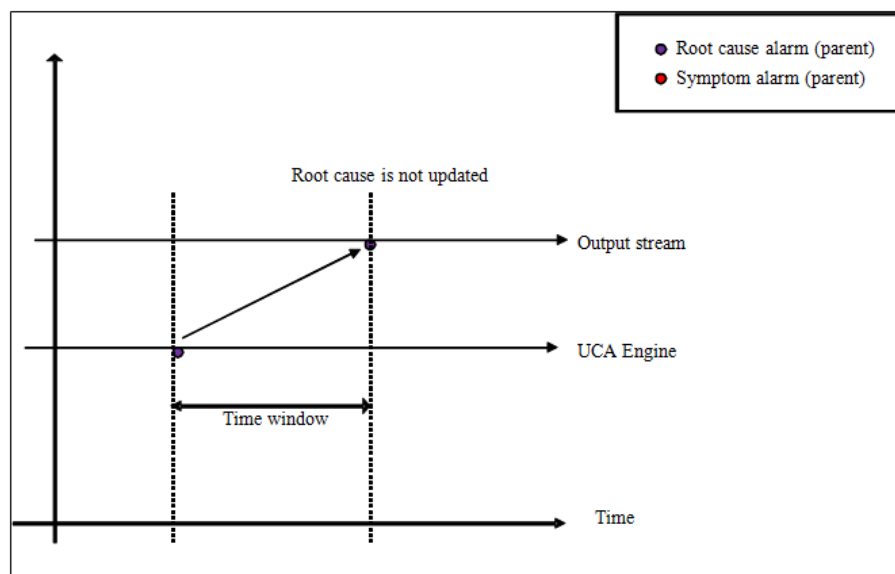


Figure 8 - Grouping – No symptom alarm received during the time window

2.3.3.4 Clearance on symptom alarms received during the time window

This context applies when the root cause alarm is followed by child alarms and child clearance.

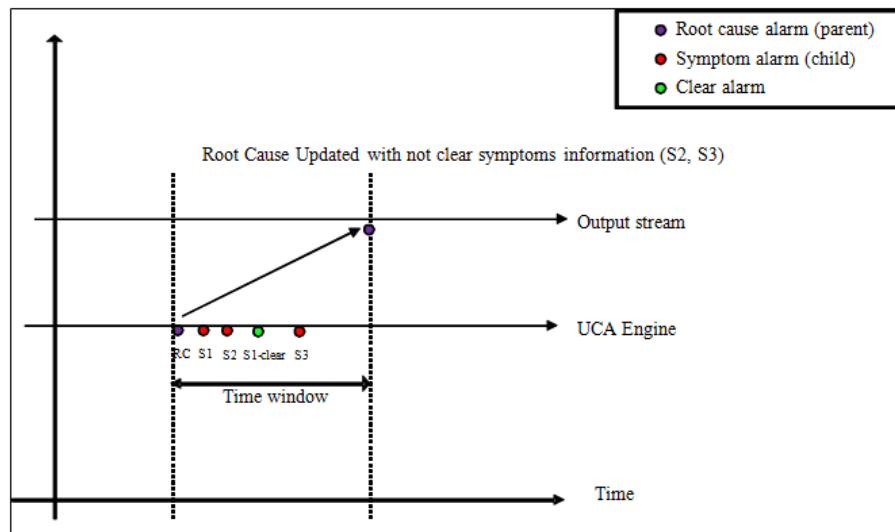


Figure 9 - Grouping – Clearance on symptom alarms received during the time window

2.3.3.5 Clearance on root cause and symptom alarms received during the time window

This use case applies when the root cause alarm is followed by the root alarm clearance as well as child alarms and clearances:

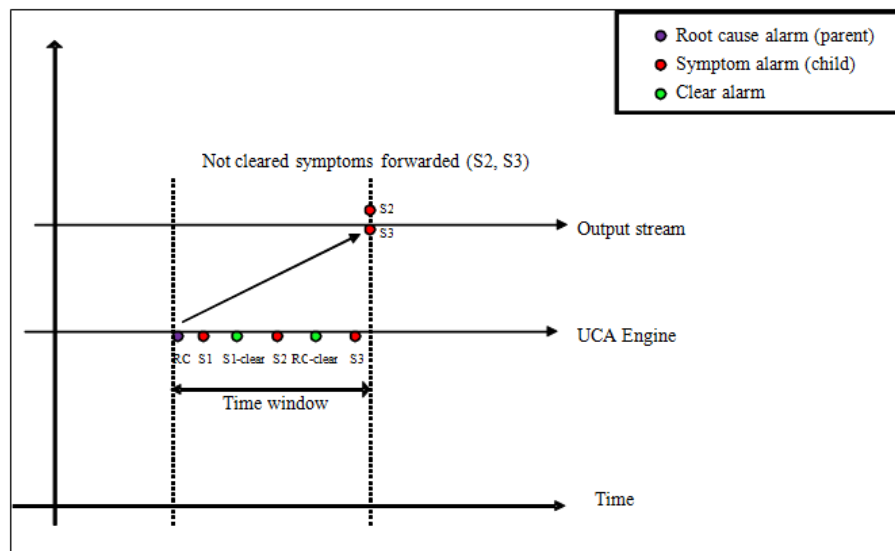


Figure 10 - Grouping – Clearance on root cause and symptom alarms received during the time window

2.3.4 The Inactivity scenario

2.3.4.1 Functional description

The Inactivity scenario identifies when a specific technology stops forwarding raw messages. The Inactivity scenario receives all alarms for the specific technology.

For each valid alarm received, the scenario resets a counter associated with the monitored item. If no valid alarm is received during a configurable period of time, an “Inactivity” alarm is created and injected into the Alarm Management system, so it can be enriched and moved forward.

The main rules in the Inactivity scenario are:

- Generate an “Inactivity” alarm if no alarm is received during a configurable time period.
- Normal alarms are discarded

The following sections describe the possible use cases in detail.

Note

The rules are parameterized. You can set the value for the following parameters in the *inactivity-params.xml* file:

- **inactivityTime**: duration of the time window period (the default value is 5 seconds)
-

2.3.4.2 Inactivity detected

This uses case applies when no recurrent alarm is received during the configurable time window:

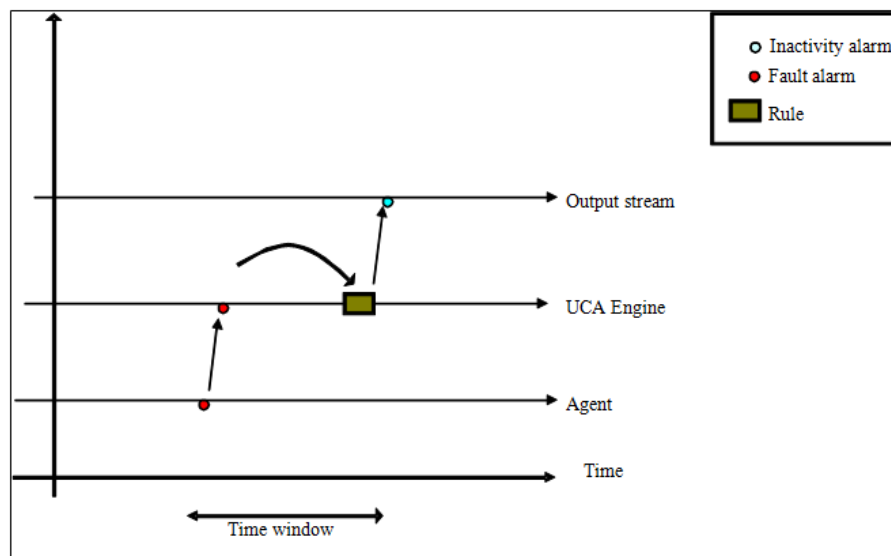


Figure 11 - Inactivity– Inactivity detected

2.3.4.3 Inactivity not detected

This use case applies when recurrent alarms are received during the configurable time window:

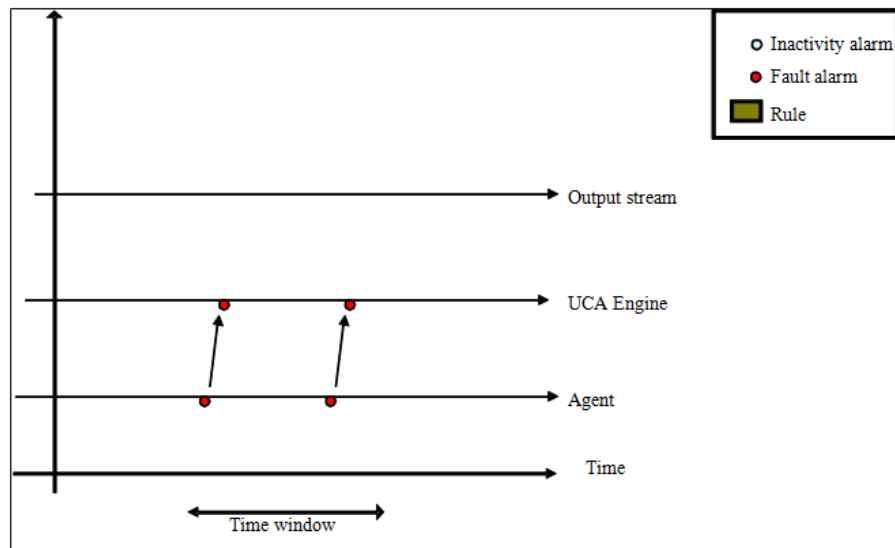


Figure 12 - Inactivity – Inactivity not detected

2.3.4.4 Inactivity detection in mix technology context

This use case applies when one technology is active and another one is not:

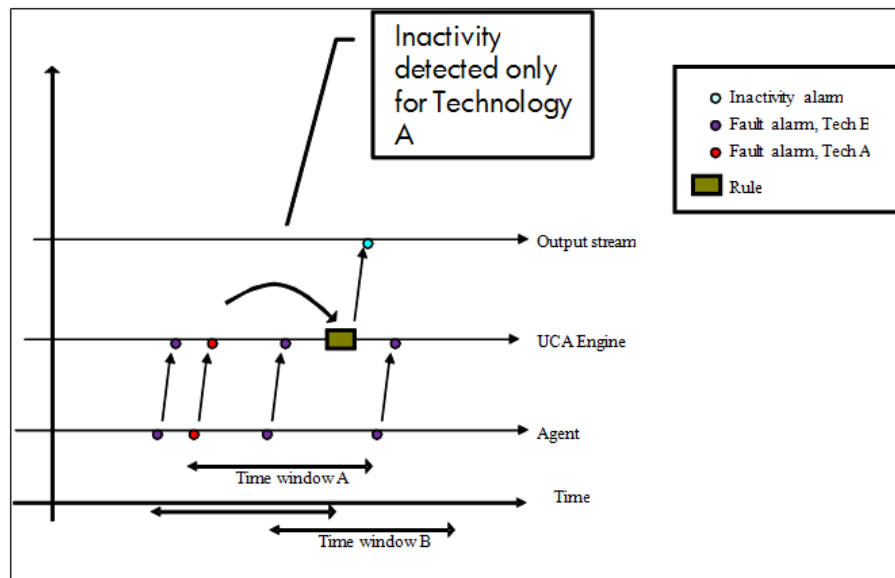


Figure 13 - Inactivity – Inactivity detection in mix technology context

2.3.5 The Up/Down scenario

2.3.5.1 Functional description

The Up/Down scenario is a base scenario, used by both the Time Wait and Grouping scenarios. The purpose is to increase performance. It handles alarms that have a common clearance alarm.

All alarms that share the same clearance enter this scenario. When one of such alarms is received, the scenario keeps a copy of it. When the common clearance

alarm is received, an independent clearance alarm is generated for each alarm stored by the scenario.

The following sections describe the possible use cases in detail.

2.3.5.2 Independent clearance

In this use case, the Up/Down scenario stores incoming alarms until it receives a common clearance alarm, in which case an independent clearance alarm is created for each alarm stored by the scenario:

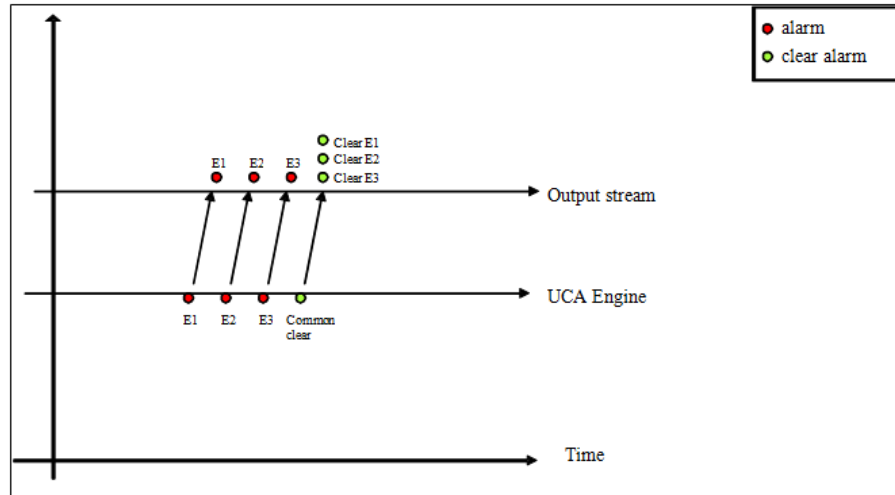


Figure 14 - Up/Down – Independent clearance

2.3.5.3 Independent clearance cascaded with Time Wait scenario

In this use case the Up/Down scenario feeds (cascades) into the Time Wait scenario:

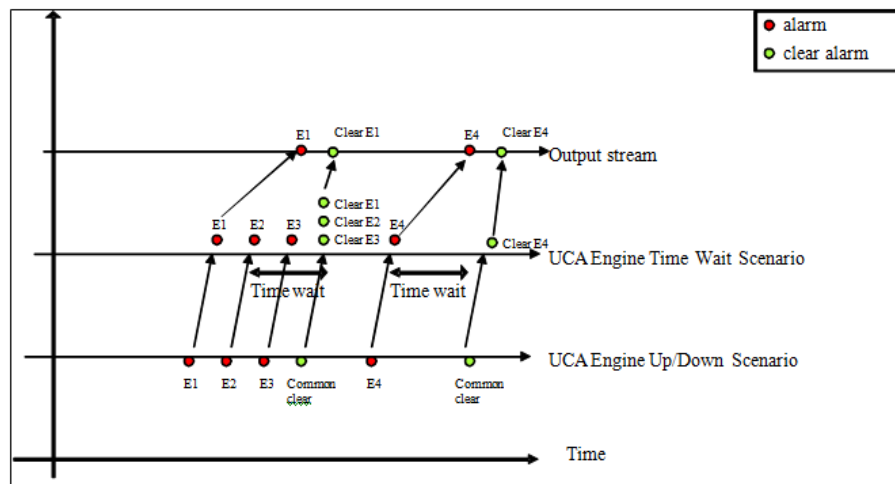


Figure 15 - Up/Down – Independent clearance cascaded with Time Wait scenario

2.4 Testing the Low Level Event Filtering Value Pack

For all the LLEF example scenarios described earlier, some alarm sets are delivered in order to test the scenario behavior.

These alarm samples are present in the

`${UCA_EBC_INSTANCE}/deploy/llef-example-3.2/<scenario name>/Alarms.xml` files after the LLEF Value Pack has been deployed.

The Alarm.xml files can be injected into UCA for EBC by using the **uca-ebc-injector** command-line tool as follows:

On both HP-UX and Linux:

```
$ ${UCA_EBC_HOME}/bin/uca-ebc-injector -file Alarms.xml
```

Checking the scenario result:

Rules actions of the LLEF example are designed to simulate real alarm actions by logging the actions into a log file:

`${UCA_EBC_INSTANCE}/logs/llef_example.log`.

Playing the Up/Down scenario with the Up/Down alarm sample file

(`${UCA_EBC_INSTANCE}/deploy/llef-example-3.2/updown/Alarms.xml`) generated the following output in the `${UCA_EBC_INSTANCE}/logs/llef_example.log` log file:

```
2012-01-04 17:22:44.982:
com.hp.uca.expert.vp.llef.timewait.TimeWait: Dummy Action
processed on alarm: 8036aa86-046e-4cbc-9dd4-ba67ff528452
2012-01-04 17:22:46.995:
com.hp.uca.expert.vp.llef.timewait.TimeWait: Dummy Action
processed on alarm: 12303
2012-01-04 17:22:46.995:
com.hp.uca.expert.vp.llef.timewait.TimeWait: Dummy Action
processed on alarm: 12307
```

You will notice that according to the Up/Down scenario design, alarms have indeed been forwarded to the Time Wait scenario which performed the actions.

An “Orchestration of Scenarios Cascading” Value Pack

This chapter provides user documentation about the “**Orchestration of Scenarios Cascading**” Value Pack example provided with the UCA-EBC 3.2 Server and with the UCA-EBC 3.2 Development kit. It uses the Orchestration of events feature introduced in UCA-EBC 3.1, which is an extension of the alarms cascading between scenarios provided in UCA-EBC 3.0.

3.1 The “Orchestration of Scenarios Cascading” Value Pack Description

The “Orchestration of Scenarios Cascading” Value Pack delivers a predefined set of Scenarios that demonstrate the Orchestration feature of UCA-EBC. The scope of this Value Pack is to offer an example of event propagation through **COPY** routes for scenarios in different modes (CLOUD and STREAM) using basic event enrichment, grouping and correlation capabilities.


In order to take part in an Orchestration route, each scenario has to call the *applyOrchestration(Event e)* method from its rules. This is done after the event has been processed by the scenario, to send the event to the Orchestration component.

In the case of the “Orchestration of Scenarios Cascading” Value Pack, each of the scenarios (except the last scenarios in the workflow: the Correlation and DBLogger scenarios) has rules defined for sending events to the Orchestration component.

Also, for the events to be routes between Value Packs, Orchestration Routes have to be defined in the *OrchestraConfiguration.xml* file of the UCA-EBC server instance, under in *\${UCA_EBC_INSTANCE}/conf*. This file is only loaded at UCA-EBC server instance start (**static loading**), so **if this file is modified, the server has to be restarted so that the new Orchestration configuration can be taken into consideration.**

For this purpose, an example of routing configuration is provided in the *OrchestraConfigurationCascadingExample.xml* file (found in the */conf* folder of the “Orchestration of Scenarios Cascading” Value Pack).

Note

 For more information on how to use the Orchestration methods in a value pack and on Orchestra Routes configuration, please refer to [R1] *UCA for EBC Reference Guide*.

3.1.1 The Scenarios taking part in the Orchestration

The following scenarios take part in the orchestration:

1. **Communication scenario:** this scenario in Stream mode receives “Communication” events from the Dispatcher. These events are correlated by grouping them during a configurable time window.
2. **Environmental scenario:** this scenario in Stream mode receives “Environmental” events from the Dispatcher. As in the case of the Communication Scenario, events are correlated by grouping them during a configurable time window.
3. **Enrichment scenario:** this scenario in Cloud mode receives all the events correlated by either the Communication, or the Environmental scenarios. On the reception of events of type Alarm (*AlarmCreation*) having *justInserted* boolean to true, this scenario enriches them with information in the Additional Text.
4. **Correlation scenario:** this scenario in Cloud mode receives a copy of all the events enriched by the Enrichment scenario (*i.e.* enriched “Communication” and “Environmental” grouped events) and will apply correlation by grouping events based on the extra information added by the Enrichment scenario in the Additional Text of the event.
5. **DBLogger scenario:** this scenario in Stream mode simulates logging to a database but actually logs to the Console. This scenario receives a copy of all the events enriched by the Enrichment scenario (*i.e.* enriched “Communication” and “Environmental” grouped events) and logs them to the Console (using the Java class `com.hp.uca.expert.vp.cascading.dblogger.AcmeDBLogger`).

Some lines of code (corresponding to actions that will group alarms together on TeMIP) have been commented out in the Communication, Environmental and Correlation scenarios of the “Orchestration of Scenarios Cascading” Value Pack because they require access to OSS OpenMediation and TeMIP. If wanting to use OSS OpenMediation and TeMIP, they have to be uncommented. For example, in the Communication scenario:

```
#Actions.associateAlarms(theScenario,firstAlarm,newAlarms);
```

3.1.2 The Orchestration Routes

In the “Orchestration of Scenarios Cascading” Value Pack only the Communication and Environmental scenarios are “eligible for broadcast” (*i.e.* they receive events from the Dispatcher).

The Enrichment, Correlation, and DBLogger scenarios are not “eligible for broadcast” and thus they can only receive events from Orchestra, when they are the Target scenario in at least one route defined in the *OrchestraConfiguration.xml* configuration file. The workflow of events in the “Orchestration of Scenarios Cascading” Value Pack is the following:

- Events are received by the Communication and Environmental scenarios and after, they are grouped. The resulting grouped events are sent to the Orchestration component of UCA EBC
- The Orchestration component of UCA EBC then routes events to the other scenarios according to its configuration file: the *OrchestraConfigurationCascadingExample.xml* file (found in the */conf* folder of the “Orchestration of Scenarios Cascading” Value Pack), as show in the figure:

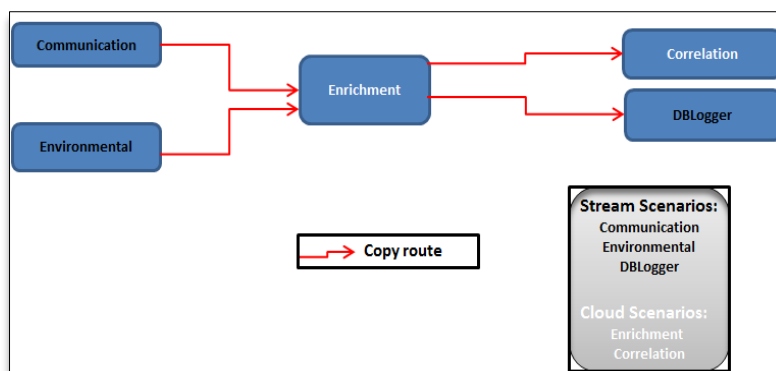


Figure 16 - OrchestraConfigurationCascadingExample.xml Routes

3.1.3 The Orchestration Event Flow

In the “Orchestration of Scenarios Cascading” Value Pack, the Event flow is dispatched to the different scenarios according to the following figure:

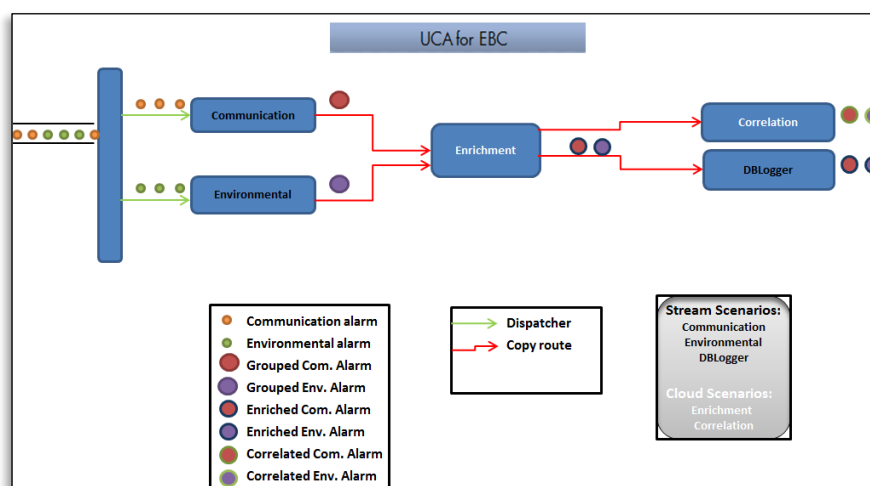


Figure 17 - Event flow in “Orchestration of Scenarios Cascading” Value Pack

3.1.4 The Software Prerequisites

The “Orchestration of Scenarios Cascading” Value Pack is delivered with the UCA for EBC Development Toolkit product under the *vp-examples/* directory:

```
${UCA_EBC_DEV_HOME}/vp-examples/cascading-example
```

The Orchestration routes have to be set in the main *OrchestraConfiguration.xml* file of UCA-EBC server (in the `${UCA_EBC_INSTANCE}/conf` folder), as described in section 3 *Set the Orchestration Routes on the UCA-EBC server instance.*

3.2 Deploy and start the “Orchestration of Scenarios Cascading” Value Pack

Several steps are needed to deploy and start the “Orchestration of Scenarios Cascading” value pack on the `${UCA_EBC_INSTANCE}` server:

1. Install the Value Pack package (ZIP file) in the `${UCA_EBC_INSTANCE}/valuepacks` directory (see 3.2.1)
2. Deploy the Value Pack (see 3.2.2)
3. Set the Orchestration Routes on the UCA-EBC server instance (see 3.2.3)
4. Start the Value Pack (see 3.2.4)

These steps are detailed in the following sections.

Note

`${UCA_EBC_INSTANCE}` translates to `/var/opt/UCA-EBC/instances/<instance name>` by default unless UCA for EBC was installed at an alternate location.

3.2.1 Install the Value Pack

The “Orchestration of Scenarios Cascading” value pack package (ZIP file) is installed by default with the UCA for EBC Server kit. This value pack is located in the `${UCA_EBC_HOME}/defaults/valuepacks` directory. You will need to copy the zip file of the value pack (named `cascading-vp-3.2.zip`) to the `${UCA_EBC_INSTANCE}/valuepacks` directory, so that it can be seen by UCA for EBC.

Alternatively, if you have installed the UCA for EBC Development Toolkit, you can (modify and) re-build the “Orchestration of Scenarios Cascading” value pack from the source code by executing the following commands:

On Windows:

```
$ cd %UCA_EBC_DEV_HOME%\vp-examples\cascading-example
$ ant all
```

On Linux:

```
$ cd ${UCA_EBC_DEV_HOME}/vp-examples/cascading-example
$ ant all
```

Once built, the value pack package (ZIP file) is ready to be deployed on UCA for EBC. You need to copy the Value Pack package you have just generated to the `${UCA_EBC_INSTANCE}/valuepacks` directory.

3.2.2 Deploy the Value Pack

To deploy the “Orchestration of Scenarios Cascading” value pack, please use the “--deploy” option of the **uca-ebc-admin** command-line administration tool (executed as **‘uca’** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --deploy -vpn cascading -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Value Pack name: cascading version: 3.2 has been
successfully deployed
INFO - Exiting...
```

Or simply deploy the Value Pack from the UCA for EBC User Interface.

File organization

At the end of the deployment step, the files delivered by the Value Pack are deployed in `${UCA_EBC_INSTANCE}/deploy/cascading-3.2` directory, according to the following file structure:

Directories	Description
lib/	Some additional jar files are installed for this package.
conf/	A configuration file that defines the Value Pack, and the scenarios (ValuePackConfiguration.xml) and an example of the Orchestration Routes (<i>OrchestraConfigurationCascadingExample.xml</i>)
common/	Stream mode rules used by the Communication and Environmental scenarios.
communication/	Specific rule file for the Communication scenario, filter file and an <i>Alarms.xml</i> sample file.
correlation/	Specific rule file for the Correlation scenario and filter file.
dblogger/	Specific rule file for the DBLogger scenario and filter file.
enrichment/	Specific rule file for the Enrichment scenario, filter file, and <i>AlarmsCascading1.xml</i> and <i>AlarmsCascading2.xml</i> sample files.
environmental/	Specific rule file for the Environmental scenario, filter file and an <i>Alarms.xml</i> sample file.

Table 3 - File structure of the “Orchestration of Scenarios Cascading” Value Pack

3.2.3 Set the Orchestration Routes

After deploying the Value Pack, under `conf/` directory there is an example of the Orchestration Routes in the *OrchestraConfigurationCascadingExample.xml*.


The *OrchestraConfiguration.xml* file of any UCA EBC Server instance is located in the `${UCA_EBC_INSTANCE}/conf` folder.

In order to test the Orchestration of the different scenarios of this Value Pack, there are two possibilities:

1. The *OrchestraConfigurationCascadingExample.xml* file has to be copied in the `conf/` folder of the UCA-EBC server instance where the Value Pack is (to be) deployed and renamed to *OrchestraConfiguration.xml*. Attention, this will replace the *OrchestraConfiguration.xml* file of the server so the only Orchestration routes will be the ones defined inside. If you had other Orchestra routes that you would like to keep, please use option 2.
2. Copy all of the routes defined inside the *OrchestraConfigurationCascadingExample.xml* (represented by each of the `<Route>` Tags) inside the `<Routes>` tag of the existing *OrchestraConfiguration.xml* file of the UCA EBC Server instance where Value Pack is (to be) deployed.

After any change in the *OrchestraConfiguration.xml* file of the UCA EBC server instance, the server has to be restarted, in order for the Routes to be taken into consideration.

Note

 For more information on how to configure the Orchestration feature of UCA-EBC, please refer to [R1] *UCA for EBC Reference Guide*.

3.2.4 Start the Value Pack

Value Packs can be started in two different manners depending on whether UCA for EBC is already started or not.

If UCA for EBC is stopped, restarting the application will automatically start all Value Packs deployed in the `${UCA_EBC_INSTANCE}/deploy` directory and load the Orchestration routes.

If UCA for EBC is already running, use the “--start” option of the **uca-ebc-admin** command-line administration tool (executed as ‘**uca**’ user) to start the Value Pack:

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --start -vpn cascading -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Starting [ cascading, 3.2, all scenarios ]
INFO - Status: [ cascading, 3.2, all scenarios ]Value pack
has been successfully started. Status of the value pack:
Running
```

Or simply start it from the UCA for EBC Web User Interface.

3.3 Stop and undeploy the “Orchestration of Scenarios Cascading” Value Pack

Several steps are needed to stop (if running) and undeploy the “Orchestration of Scenarios Cascading” value pack from the `${UCA_EBC_INSTANCE}` server:

1. Stop the Value Pack (see 3.3.1)

2. Undeploy the Value Pack (see 3.3.2)

These steps are detailed in the following sections.

Note

`${UCA_EBC_INSTANCE}` translates to `/var/opt/UCA-EBC/instances/<instance name>` by default unless UCA for EBC was installed at an alternate location.

3.3.1 Stop the Value Pack

You can stop the Value Pack when UCA for EBC is running using the “--stop” option of the **uca-ebc-admin** command-line administration tool (executed as **‘uca’** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --stop -vpn cascading -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Stopping [ cascading, 3.2, all scenarios ]
INFO - Status: Value pack has been successfully stopped.
Status of the value pack: Stopped
```

Or simply stop it from the UCA for EBC Web User Interface.

Even if the Value Pack is stopped, the Orchestration Routes are kept. If you want to remove the Orchestration Routes, they have to be removed from the `OrchestraConfiguration.xml` file of the UCA-EBC server instance and the server has to be restarted.

3.3.2 Undeploy the Value Pack

To undeploy the Orchestration Value Pack, use the “--undeploy” option of the **uca-ebc-admin** command-line administration tool (executed as **‘uca’** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --undeploy -vpn cascading -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Undeploying [ cascading, 3.2, all scenarios ]
INFO - Status: Value pack has been successfully undeployed.
Status of the value pack: NotDeployed
```

Or simply undeploy it from the UCA for EBC Web User Interface.

Even if the Value Pack is unloaded, the Orchestration Routes are kept. If you want to also remove the Orchestration Routes, they have to be removed from the

OrchestraConfiguration.xml file of the UCA-EBC server instance and the server has to be restarted.

3.4 Test the “Orchestration of Scenarios Cascading” Value Pack

This section described the steps to follow in order to test the “**Orchestration of Scenarios Cascading**” Value Pack.

3.4.1 Event sample files

For the “Orchestration of Scenarios Cascading” Value Pack described, some event files are delivered in order to **test the scenario orchestration behavior**.

For the two scenarios that are “**eligible for broadcast**” (the Communication and Environment scenarios), event samples are present in the `${UCA_EBC_INSTANCE}/deploy/cascading-3.2/<scenario name>/Alarms.xml` file (where `<scenario name>` can be *Communication* and *Environmental*) after the Value Pack has been deployed.

In order to test with different event types for both scenarios, the *AlarmsCascading1.xml* and *AlarmsCascading2.xml* files are present under `${UCA_EBC_INSTANCE}/deploy/cascading-3.2-SNAPSHOT/enrichment/` folder.

The *AlarmsCascading1.xml* file contains *AlarmCreation* events for the Communication and Environmental scenarios. When we inject this file into UCA for EBC, the alarms are grouped by the Communication and Environment scenarios then sent to the Orchestration component.

The *AlarmsCascading2.xml* file contains *AlarmAttributeValueChange* and *AlarmStateChange* events. When we inject this file into UCA for EBC, the events are sent directly to the **Orchestration component** (without any grouping).

Events can be injected into UCA for EBC using the **uca-ebc-injector** command-line tool as follows:

3.4.2 Injecting events with the uca-ebc-injector

On both HP-UX and Linux:

```
$ ${UCA_EBC_HOME}/bin/uca-ebc-injector -file  
${UCA_EBC_INSTANCE}/deploy/cascading-3.2/enrichment/  
AlarmsCascading1.xml
```

After injecting the events from the *AlarmsCascading1.xml* file, please wait at least 3 seconds (this is the maximum time it takes for the events to be processed by the Communication (2 seconds) and Environment (3 seconds) scenarios) and then insert AVC (Attribute Value Change) and SC (State Change) events from the *AlarmsCascading2.xml* file.

```
$ ${UCA_EBC_HOME}/bin/uca-ebc-injector -file  
${UCA_EBC_INSTANCE}/deploy/cascading-3.2/enrichment/  
AlarmsCascading2.xml
```

3.4.3 Results

Rules actions of the Orchestration example are designed to simulate real event actions. Several JUnit tests showing different propagation of events (Alarm Creation, Attribute Value Change (AVC), Alarm State Change (SC) and Alarm Deletion) can be found under *src/test/java*, in the *com.hp.uca.expert.vp.cascading* Java package of the Value Pack source code.

More precisely, the JUnit test describing the Value Pack's default configuration is the *OrchestraCascadingWithFlagGroupTimeWindowTest.java* test. This test's log gives us the same log as when we test the deployed Value Pack with the *AlarmsCascading1.xml* and *AlarmsCascading2.xml* event sample files.

As described in [R1] *UCA for EBC Reference Guide*, when delegating from a CLOUD scenario to a STREAM scenario, for each AVC or SC an Alarm Creation is done before.

We observe the following **event propagation** between the scenarios:

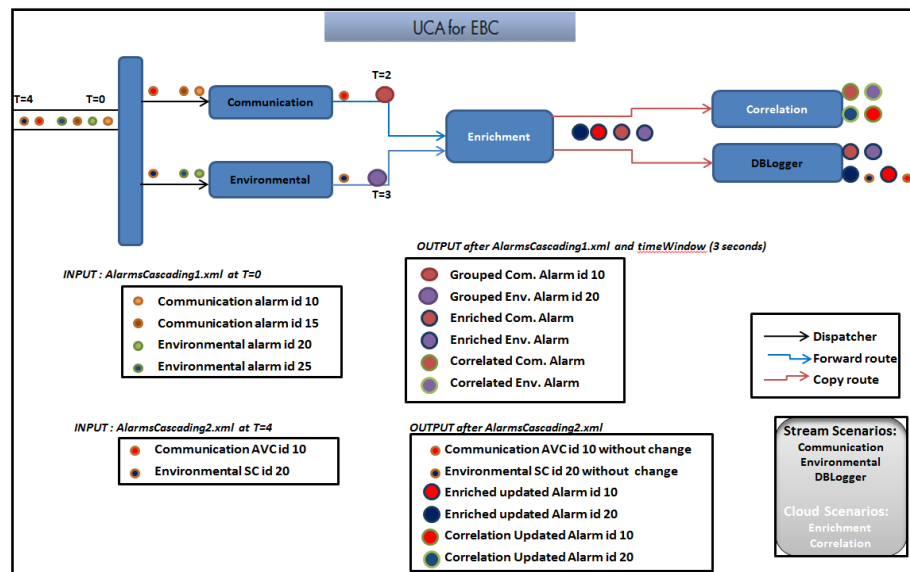


Figure 18 - Event cascading in “Orchestration of Scenarios Cascading” Value Pack test example

3.4.4 Checking the results

The event propagation can be tracked in the `${UCA_EBC_INSTANCE}/logs/uca-ebc.log` log file, when the log4j log level of each scenario is set to INFO (or DEBUG). Key information is highlighted below:

At start-up:

```
[2014-04-17 12:39:42,506][INFO ][T-Main
][com.hp.uca.expert.orchestra.WorkflowConfiguration][ 47]Loading Orchestra
Workflow from OrchestraConfiguration.xml
```

Deploy and start Orchestra Cascading Value Pack:

```
[2014-04-17 12:41:36,341][INFO ][206299513@qtp-86386279-
3][com.hp.uca.expert.vp.internal.ValuePackLoader][ 188]Value pack 'cascading-
3.2' deployed
```


[2014-04-17 12:41:39,920][INFO][cascading-3.2][206299513@qtp-86386279-3][com.hp.uca.expert.gui.ValuePackServices][165]Starting Value pack 'cascading-3.2'

[2014-04-17 12:41:45,691][INFO][][206299513@qtp-86386279-3][com.hp.uca.expert.gui.ValuePackServices][172]Value pack 'cascading-3.2' started

After insertion of the *AlarmsCascading1.xml* file and after waiting for at least 3 seconds:

[2014-04-17 12:43:29,440][INFO][cascading-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication][com.hp.uca.expert.vp.cascading.Communication][9]Inserting Flag for Context: BOX B1

[2014-04-17 12:43:29,440][INFO][cascading-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Environmental][com.hp.uca.expert.vp.cascading.Environmental][9]Inserting Flag for Context: BOX B1

[2014-04-17 12:43:34,769][INFO][cascading-3.2][T-Watchdog-com.hp.uca.expert.vp.cascading.Communication][com.hp.uca.expert.vp.cascading.Communication][11]Grouping Alarm: operation_context.uca_cri_oc alarm_object 10

[2014-04-17 12:43:34,770][INFO][cascading-3.2][T-Watchdog-com.hp.uca.expert.vp.cascading.Communication][com.hp.uca.expert.vp.cascading.Communication][24]Send to Orchestra operation_context.uca_cri_oc alarm_object 10

[2014-04-17 12:43:34,783][INFO][cascading-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Enrichment][com.hp.uca.expert.vp.cascading.Enrichment][11]Send to Orchestra Just inserted operation_context.uca_cri_oc alarm_object 10

[2014-04-17 12:43:34,787][INFO][cascading-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblogger.AcmeDbLogger][34]==> Alarm: id=operation_context.uca_cri_oc alarm_object 10, t=2013-12-16T12:00:00.000+02:00, e=BOX B1, type=COMMUNICATIONS_ALARM, s=MINOR, ns=NOT_CLEARED, os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false, ret=false

[2014-04-17 12:43:34,793][INFO][cascading-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Correlation][8]Correlation - Create Group operation_context.uca_cri_oc alarm_object 10

[2014-04-17 12:43:34,949][INFO][cascading-3.2][T-Watchdog-com.hp.uca.expert.vp.cascading.Environmental][com.hp.uca.expert.vp.cascading.Environmental][11]Grouping Alarm: operation_context.uca_cri_oc alarm_object 20

[2014-04-17 12:43:34,950][INFO][cascading-3.2][T-Watchdog-com.hp.uca.expert.vp.cascading.Environmental][com.hp.uca.expert.vp.cascading.Environmental][25]Send to Orchestra operation_context.uca_cri_oc alarm_object 20

```
[2014-04-17 12:43:34,951][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Enrichment][com.hp.uca.expert.vp.cascading.Enric
hment][ 11]Send to Orchestra Just inserted
operation_context .uca_cri_oc alarm_object 20

[2014-04-17 12:43:34,953][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc
alarm_object 20, t=2013-12-16T12:00:03.000+02:10, e=BOX B1,
type=ENVIRONMENTAL_ALARM, s=MAJOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false,
ret=false

[2014-04-17 12:43:34,953][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 8]Correlation - Create Group
operation_context .uca_cri_oc alarm_object 20
```

After insertion of the *AlarmsCascading2.xml* file:

```
[2014-04-17 12:44:52,642][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Communication][com.hp.uca.expert.vp.cascading.
Communication][ 8]Send to Orchestra AVC
operation_context .uca_cri_oc alarm_object 10

[2014-04-17 12:44:52,642][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Environmental][com.hp.uca.expert.vp.cascading.E
nvironmental][ 8]Send to Orchestra SC
operation_context .uca_cri_oc alarm_object 20

[2014-04-17 12:44:52,646][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Enrichment][com.hp.uca.expert.vp.cascading.Enric
hment][ 8]Send to Orchestra AVC
operation_context .uca_cri_oc alarm_object 10

[2014-04-17 12:44:52,649][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc
alarm_object 10, t=2013-12-16T12:00:00.000+02:00, e=BOX B1,
type=COMMUNICATIONS_ALARM, s=MINOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=true, sc=false,
ret=false

[2014-04-17 12:44:52,649][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Enrichment][com.hp.uca.expert.vp.cascading.Enric
hment][ 8]Send to Orchestra SC
operation_context .uca_cri_oc alarm_object 20

[2014-04-17 12:44:52,650][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 8]Correlation (AVC Updated)
operation_context .uca_cri_oc alarm_object 10

[2014-04-17 12:44:52,653][INFO ][cascading-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 8]Correlation (SC Updated)
operation_context .uca_cri_oc alarm_object 20
```

```
[2014-04-17 12:44:52,658][INFO ][cascading-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog  
ger.AcmeDbLogger][ 56]==> AlarmAttributeValueChange: id=operation_context  
.uca_cri_oc alarm_object 10, t=2013-12-16T13:00:05.000+01:00, e=BOX B1,  
type=COMMUNICATIONS_ALARM, s=MINOR, os=NOT_ACKNOWLEDGED  
  
[2014-04-17 12:44:52,659][INFO ][cascading-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog  
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc  
alarm_object 20, t=2013-12-16T12:00:03.000+02:10, e=BOX B1,  
type=ENVIRONMENTAL_ALARM, s=MAJOR, ns=CLEARED, os=NOT_ACKNOWLEDGED,  
ps=NOT_HANDLED, ins=true, avc=false, sc=true, ret=false  
  
[2014-04-17 12:44:52,661][INFO ][cascading-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog  
ger.AcmeDbLogger][ 45]==> AlarmStateChange: id=operation_context .uca_cri_oc  
alarm_object 20, t=2013-12-16T10:50:03.000+01:00, e=BOX B1, type=
```

An “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

This chapter provides user documentation about the “**Orchestration of Scenarios Cascading in JOIN Routes**” Value Pack example provided with the UCA-EBC 3.2 Development kit. It uses the Orchestration of events feature introduced in UCA-EBC 3.1, which is an extension of the alarms cascading between scenarios provided in UCA-EBC 3.0.

4.1 The “Orchestration of Scenarios Cascading in JOIN Routes” Value Pack Description

The “Orchestration of Scenarios Cascading in JOIN routes” Value Pack delivers a predefined set of Scenarios that demonstrate some of the features of the Orchestration component of UCA-EBC: how to cascade events between scenarios in *STREAM* and *CLOUD* modes for **JOIN** operations.

As described in [R1] *UCA for EBC Reference Guide*, a JOIN operation consists in aggregating the information (**orchestraData**) added by several scenarios on copies of the same event, and sending the resulting aggregate event to another scenario.

The scope of this Value Pack is to offer an example of event propagation through scenarios in different modes (CLOUD and STREAM) using basic event enrichment capabilities: adding specific information to events and joining these events so that we end up with merged events containing the combined enrichment information.

As described in the “Orchestration of Scenarios Cascading” Value Pack, in order to take part into an Orchestration route, each scenario has to trigger the ***applyOrchestration(Event e)*** method in its rule, after applying a specific treatment (like enrichment or other). Therefore, each of the scenarios has rules defined for applying orchestration on the following event types:


AlarmCreation, *AlarmAttributeValueChange*,
AlarmStateChange and *AlarmDeletion*.

In the case of the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack, each of the scenarios (except the last scenarios in the workflow: the Correlation and DBLogger scenarios) has rules defined for sending events to the **Orchestration component**.

Also, for the events to be routed between Value Packs, Orchestration Routes have to be defined in the *OrchestraConfiguration.xml* file of the UCA-EBC server instance, in the `${UCA_EBC_INSTANCE}/conf` folder. This file is only loaded at UCA-EBC server instance start (**static loading**), so **if this file is modified, the server has to be restarted so that the new Orchestration configuration can be taken into consideration**.

For this purpose, an example of routing configuration is provided in the *OrchestraConfigurationCascadingJoinExample.xml* file (found in the `/conf` folder of the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack).

Note

 For more information on how to use the Orchestration methods in a value pack and on Orchestra Routes configuration, please refer to [R1] *UCA for EBC Reference Guide*.

4.1.1 The Scenarios taking part in the Orchestration

The following scenarios will be orchestrated:

1. **Communication1 and Communication2**: these scenarios in Stream mode receive “Communication” events and add specific information to these events in the *orchestraData* object attached to them. These events are to be joined by the Orchestration component into one event and sent to the **EnrichmentC** scenario.
2. **Environmental1 and Environmental2**: these scenarios in Stream mode receive “Environmental” events and add specific information to these events in the *orchestraData* object attached to them. These events are to be joined by the Orchestration component into one event and sent to the **EnrichmentS** scenario.
3. **EnrichmentC**: this scenario in Cloud mode receives the joined “Communication” events when:
 - both the Communication1 and Communication2 scenarios send their enriched event to the Orchestration component before the JOIN timeout expires (the timeout for this JOIN route is set by the **expireTime** element in the Orchestration configuration file: it represents the maximum time to wait for a JOIN route to complete) : the joined “Communication” events are complete (they have their **convergenceComplete** flag set to **true**)
 - either one (or both) of the Communication1 and Communication2 scenarios doesn’t send their enriched event to the Orchestration component before the timeout expires (the timeout is set to 5 seconds in the Orchestration configuration file): the joined “Communication” events are incomplete (they have their **convergenceComplete** flag set to **false**)

Once the joined “Communication” events are received, they will be enriched with additional information in the additional Text.

4. **EnrichmentS**: this scenario in Cloud mode receives the joined “Environmental” events when:
 - both the Environmental1 and Environmental2 scenarios send their enriched event to the Orchestration component before the JOIN timeout expires
 - either one (or both) of the Environmental1 and Environmental2 scenarios doesn’t send their enriched event to the Orchestration component before the timeout for the JOIN route expires (5 seconds)

Once the joined “Environmental” events are received, they will be enriched with additional information in the additional Text.

5. **Correlation**: this scenario in Cloud mode receives a copy of all the events enriched by the EnrichmentC and by the EnrichmentS scenarios (*i.e.* enriched Communication and Environmental events) and will apply correlation by grouping events based on specific information (additional Text) added by the EnrichmentC or by the EnrichmentS scenario.

6. **DBLogger**: this scenario in Stream mode receives a copy of all the events enriched by the EnrichmentC and by the EnrichmentS scenarios (i.e. enriched Communication and Environmental events) and logs them to the console (using the Java class `com.hp.uca.expert.vp.cascading.dblogger.AcmeDBLogger`).

In order to be able to test the Value Pack with the **uca-ebc-injector** command-line tool provided with UCA-EBC, event grouping using TeMIP Actions has been commented out in the rules files of the Correlation scenario (in the `correlation.drl` file). Please uncomment these lines if you want to use TeMIP Actions.

4.1.2 The Orchestration Routes

In the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack only the Communication1, Communication2, Environmental1 and Environmental2 scenarios are “eligible for broadcast” (i.e. they receive events from the Dispatcher).

The EnrichmentC, EnrichmentS, Correlation, and DBLogger scenarios are not “eligible for broadcast” and thus they can only receive events from Orchestra, when they are the Target scenario of at least one route defined in the *OrchestraConfiguration.xml* configuration file.

The Orchestration component of UCA EBC routes events between the scenarios according to its configuration file: the *OrchestraConfigurationCascadingJoinExample.xml* file (found in the `/conf` folder of the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack), as show in the following figure:

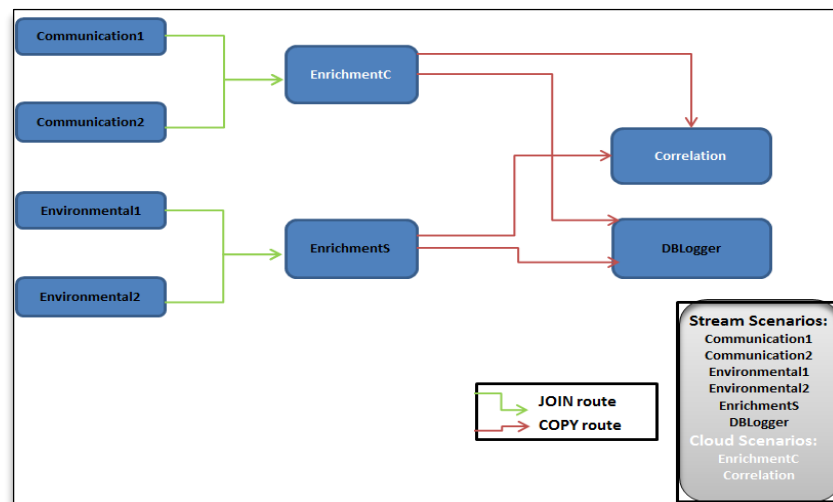


Figure 19 - OrchestraConfigurationCascadingJoinExample.xml Routes

4.1.3 The Orchestration Event Flow

In the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack, the event flow is dispatched to the different scenarios according to the following figure:

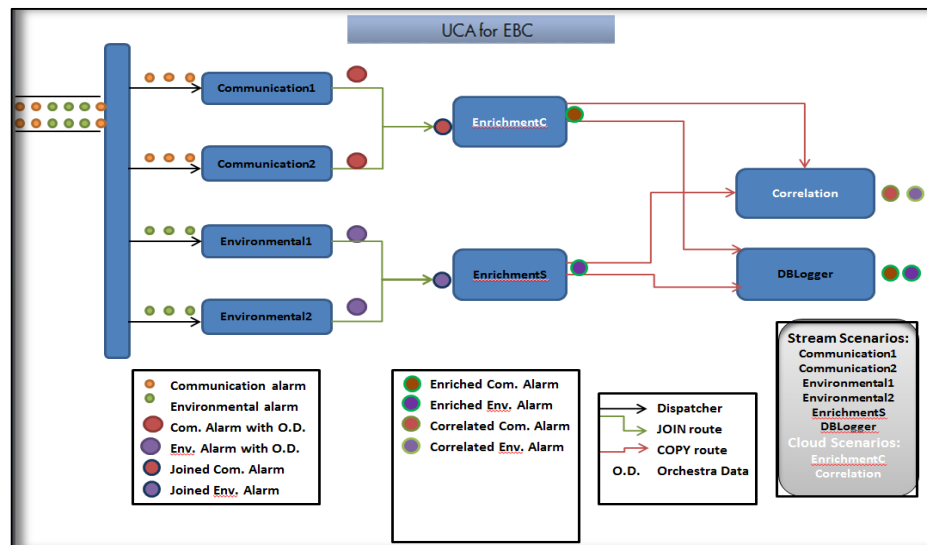


Figure 20 - Event flow in “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

The event flow is the following:

1. Events are received by the Communication1/2 (respectively by Environmental 1/2) scenarios, enriched, merged by the Orchestra component, then sent to the EnrichmentC (respectively EnrichmentS) scenario.
2. From EnrichmentC (respectively EnrichmentS) scenario events are copied to the Correlation and DBLogger scenarios

4.1.4 The Software Prerequisites

The “Orchestration of Scenarios Cascading in JOIN routes” Value Pack is delivered with the UCA for EBC Development Toolkit product under the `vp-examples/` directory:

```
${UCA_EBC_DEV_HOME}/vp-examples/cascading-example-join
```

The Orchestration routes have to be set in the main `OrchestraConfiguration.xml` file of UCA-EBC server, as described in 4.2.3 “Set the Orchestration Routes”.

4.2 Deploy and start the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

Several steps are needed to deploy the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack:

1. Install the Value Pack package (ZIP file) in the `${UCA_EBC_INSTANCE}/valuepacks` directory (see 3.2.1)
2. Deploy the Value Pack (see 3.2.2)
3. Set the Orchestration Routes on the UCA-EBC server instance (see 3.2.3)
4. Start the Value Pack (see 3.2.4)

These steps are detailed in the following sections.

Note

`${UCA_EBC_INSTANCE}` translates to `/var/opt/UCA-EBC/instances/<instance name>` by default unless UCA for EBC was installed at an alternate location.

4.2.1 Install the Value Pack

The “Orchestration of Scenarios Cascading in JOIN routes” value pack package (ZIP file) is packaged with the UCA for EBC Development Toolkit. If wanting, it can be (modified and) re-built from the source code by executing the following commands:

On Windows:

```
$ cd %UCA_EBC_DEV_HOME%\vp-examples\cascading-example-join
$ ant all
```

On Linux:

```
$ cd ${UCA_EBC_DEV_HOME}/vp-examples/cascading-example-join
$ ant all
```

Once built, the value pack package (ZIP file) is ready to be deployed on UCA for EBC. You need to copy the Value Pack package you have just generated to the `${UCA_EBC_INSTANCE}/valuepacks` directory.

4.2.2 Deploy the Value Pack

To deploy the “Orchestration of Scenarios Cascading in JOIN routes” value pack, please use the “--deploy” option of the **uca-ebc-admin** command-line administration tool (executed as **uca** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --deploy -vpn cascading-join -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Value Pack name: cascading-join version: 3.2 has been
successfully deployed
INFO - Exiting...
```

Or simply deploy the Value Pack from the UCA for EBC User Interface.

4.2.2.1 File organization

At the end of the deployment step, the files delivered by the Value Pack are deployed in `${UCA_EBC_INSTANCE}/deploy/cascading-join-3.2` directory, according to the following file structure:

Directories	Description
lib/	Some additional jar files are installed for this package.

conf/	A configuration file that define the Value Pack and the scenarios (ValuePackConfiguration.xml) and an example of the Orchestration Routes (<i>OrchestraConfigurationCascadingJoinExample.xml</i>)
communication/	Specific rule files for the Communication1 and Communication2 scenarios, and the filter file.
correlation/	Specific rule file for the Correlation scenario and filter file.
dblogger/	Specific rule file for the DBLogger scenario and filter file.
enrichment/	Specific rule files for the EnrichmentS and EnrichmentC scenarios filter files, and <i>AlarmsJoinStreams[i].xml</i> , where <i>i</i> goes from 1 to 4 sample files.
environmental/	Specific rule files for the Environmental1 and Environmental2 scenarios, and the filter file.

Table 4 - File Structure of the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

4.2.3 Set the Orchestration Routes

After deploying the Value Pack, under *conf/* directory there is an example of the Orchestration Routes in the *OrchestraConfigurationCascadingJoinExample.xml*.


The *OrchestraConfiguration.xml* file of any UCA EBC Server instance is located in the *\${UCA_EBC_INSTANCE}/conf* folder.

In order to test the Orchestration of the different scenarios of this Value Pack, there are two possibilities:

- **Option 1:** The *OrchestraConfigurationCascadingJoinExample.xml* file has to be copied in the *conf/* folder of the UCA-EBC server instance where the Value Pack is (to be) deployed and renamed to *OrchestraConfiguration.xml*. Please be aware that this will replace all previously defined Orchestration routes for the UCA-EBC server instance. If you have Orchestra routes that you would like to keep, please use option 2 instead.
- **Option 2:** Copy all of the routes defined in the *OrchestraConfigurationCascadingJoinExample.xml* (each route is represented by a `<Route>` XML tag) to the existing *OrchestraConfiguration.xml* file of the UCA EBC Server instance where the Value Pack is to be deployed.

After any change of the *OrchestraConfiguration.xml* file, the UCA EBC server instance has to be restarted, in order for the new/updated routes to be taken into consideration.

Note

 For more information on how to configure the Orchestration feature of UCA-EBC, please refer to [R1] *UCA for EBC Reference Guide*.

4.2.4 Start the Value Pack

Value Packs can be started in two different manners depending on whether UCA for EBC is already started or not.

If UCA for EBC is stopped, restarting the application will automatically start all Value Packs deployed in the `${UCA_EBC_INSTANCE}/deploy` directory and load the Orchestration routes.

If UCA for EBC is already running, use the “--start” option of the **uca-ebc-admin** command-line administration tool (executed as ‘**uca**’ user) to start the Value Pack:

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --start -vpn cascading-join -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Starting [ cascading-join, 3.2, all scenarios ]
INFO - Status: [ cascading-join, 3.2, all scenarios ]Value
pack has been successfully started. Status of the value pack:
Running
```

Or simply start it from the UCA for EBC Web User Interface.

4.3 Stop and undeploy the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

Several steps are needed to stop (if running) and undeploy the “Orchestration of Scenarios Cascading in JOIN Routes” value pack from the `${UCA_EBC_INSTANCE}` server:

3. Stop the Value Pack (see 3.3.1)
4. Undeploy the Value Pack (see 3.3.2)

These steps are detailed in the following sections.

Note

`${UCA_EBC_INSTANCE}` translates to `/var/opt/UCA-EBC/instances/<instance name>` by default unless UCA for EBC was installed at an alternate location.

4.3.1 Stop the Value Pack

You can stop the Value Pack when UCA for EBC is running using the “--stop” option of the **uca-ebc-admin** command-line administration tool (executed as ‘**uca**’ user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --stop -vpn cascading-join -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Stopping [ cascading-join, 3.2, all scenarios ]
INFO - Status: Value pack has been successfully stopped.
Status of the value pack: Stopped
```

Or simply stop it from the UCA for EBC Web User Interface.

Even if the Value Pack is stopped, the Orchestration Routes are kept. If you want to remove the Orchestration Routes, they have to be removed from the *OrchestraConfiguration.xml* file of the UCA-EBC server instance and the instance has to be restarted.

4.3.2 Undeploy the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

To undeploy the Orchestration of Scenarios Cascading in JOIN Routes Value Pack, please use the “--undeploy” option of the **uca-ebc-admin** command-line administration tool (executed as **‘uca’** user):

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --undeploy -vpn cascading-join -vpv 3.2
```

An output similar to the following will be displayed:

```
UCA for EBC Home directory set to: /opt/UCA-EBC
UCA for EBC Data directory set to: /var/opt/UCA-EBC
INFO - Undeploying [ cascading-join, 3.2, all scenarios ]
INFO - Status: Value pack has been successfully undeployed.
Status of the value pack: NotDeployed
```

Or simply undeploy it from the UCA for EBC Web User Interface.

Even if the Value Pack is unloaded, the Orchestration Routes are kept. If you want to also remove the Orchestration Routes, they have to be removed from the *OrchestraConfiguration.xml* file of the UCA-EBC server instance and the instance has to be restarted.

4.4 Test the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack

This section described the steps to follow in order to test the “**Orchestration of Scenarios Cascading in JOIN Routes**” Value Pack.

4.4.1 Event sample files

For the “Orchestration of Scenarios Cascading in JOIN routes” Value Pack described, some event files are delivered in order to **test the scenario orchestration behavior**.

In order to test with different event types, four sample event files are present in the `${UCA_EBC_INSTANCE}/deploy/cascading-join-3.2/enrichment/` folder:

- The *AlarmsJoinStreams1.xml* file contains “Communication” *AlarmCreation* events
- The *AlarmsJoinStreams2.xml* file contains an *AlarmAttributeValueChange* event for one of the “Communication” alarms in *AlarmsJoinStreams1.xml*.
- The *AlarmsJoinStreams3.xml* file contains “Environmental” *AlarmCreation* events
- The *AlarmsJoinStreams4.xml* file contains an *AlarmAttributeValueChange* event and an *AlarmStateChange* event for one of the “Environmental” alarms in *AlarmsJoinStreams3.xml*.

4.4.2 Injecting events with the uca-ebc-injector

Events can be injected into UCA for EBC using the **uca-ebc-injector** command-line tool as follows:

On both HP-UX and Linux (for i from 1 to 4):

```
$ cd ${UCA_EBC_INSTANCE}/deploy/cascading-join-3.2/enrichment/
$ ${UCA_EBC_HOME}/bin/uca-ebc-injector -file
AlarmsJoinStreamsi.xml
```

4.4.3 Results

Rules actions of the Orchestration example are designed to simulate real event actions. Several JUnit tests showing different propagation of events (Alarm Creation, Attribute Value Change (AVC), Alarm State Change (SC) and Alarm Deletion) can be found under *src/test/java*, in the *com.hp.uca.expert.vp.cascading* *Java* package of the Value Pack source code. There are two JUnit tests describing the Value Pack’s default configuration, with the different JOIN and COPY operations from/to CLOUD/STREAM scenarios.

More precisely, when inserting the *AlarmJoinStreams1.xml* and *AlarmJoinStreams2.xml* files we obtain the same results as running the *OrchestraCascadingJoinStreamsToCloudTest.java* JUnit test and when inserting the other two sample files we obtain the same results as running the *OrchestraCascadingJoinStreamsToStreamTest.java* JUnit test. As described in the [R1] *UCA for EBC Reference Guide*, when delegating from a CLOUD scenario to a STREAM scenario, for each AVC or SC an Alarm Creation is done before.

4.4.4 Checking the results

The event propagation can be tracked in the *\${UCA_EBC_INSTANCE}/logs/uca-ebc.log* file, when the log4j log level of each scenario is set to INFO. When the log4j log level of each scenario is set to DEBUG, the orchestraData propagation can be tracked with even more detail. Key information is highlighted below:

At start-up:

```
[2014-04-17 14:51:49,260][INFO ][T-Main
][com.hp.uca.expert.orchestra.WorkflowConfiguration][ 47]Loading Orchestra
Workflow from OrchestraConfiguration.xml
```

Deploy and start Orchestra Cascading in JOIN routes Value Pack:

[2014-04-17 14:52:34,388][INFO][457048813@qtp-733907953-2][com.hp.uca.expert.vp.internal.ValuePackLoader][188]Value pack 'cascading-join-3.2' deployed

[2014-04-17 14:52:41,973][INFO][cascading-join-3.2][618609084@qtp-733907953-0][com.hp.uca.expert.gui.ValuePackServices][165]Starting Value pack 'cascading-join-3.2'

[2014-04-17 14:52:47,923][INFO][618609084@qtp-733907953-0][com.hp.uca.expert.gui.ValuePackServices][172]Value pack 'cascading-join-3.2' started

After insertion of the *AlarmsJoinStreams1.xml* file:

INFO

[2014-04-17 14:53:42,347][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication1][com.hp.uca.expert.vp.cascading.Communication1][8]Send to Orchestra alarm
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:53:42,347][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication2][com.hp.uca.expert.vp.cascading.Communication2][8]Send to Orchestra alarm
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:53:42,352][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication2][com.hp.uca.expert.vp.cascading.Communication2][8]Send to Orchestra alarm
operation_context.uca_cri_oc alarm_object 333

[2014-04-17 14:53:42,352][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication1][com.hp.uca.expert.vp.cascading.Communication1][8]Send to Orchestra alarm
operation_context.uca_cri_oc alarm_object 333

[2014-04-17 14:53:42,364][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.EnrichmentC][com.hp.uca.expert.vp.cascading.EnrichmentC][11]Send to Orchestra Just inserted
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:53:42,366][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.EnrichmentC][com.hp.uca.expert.vp.cascading.EnrichmentC][11]Send to Orchestra Just inserted
operation_context.uca_cri_oc alarm_object 333

[2014-04-17 14:53:42,368][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblogger.AcmeDbLogger][34]==> Alarm: id=operation_context.uca_cri_oc alarm_object 222, t=2014-01-16T12:00:00.000+02:00, e=BOX B1, type=COMMUNICATIONS_ALARM, s=MINOR, ns=NOT_CLEARED, os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false, ret=false

[2014-04-17 14:53:42,369][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblogger.AcmeDbLogger][34]==> Alarm: id=operation_context.uca_cri_oc alarm_object 333, t=2014-01-16T12:00:00.000+02:00, e=BOX B1, type=COMMUNICATIONS_ALARM, s=MAJOR, ns=NOT_CLEARED,

os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false, ret=false

[2014-04-17 14:53:42,376][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.**Correlation**][8]Correlation - Create Group

operation_context .uca_cri_oc alarm_object 222

[2014-04-17 14:53:42,382][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.**Correlation**][8]Correlation - Create Group

operation_context .uca_cri_oc alarm_object 333

DEBUG

[2014-04-17 15:13:40,286][DEBUG][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication1][com.hp.uca.expert.vp.cascading.**Communication1**][10] - identifier = operation_context .uca_cri_oc alarm_object 222

....

- **orchestraData**

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check server

- var = none

[2014-04-17 15:13:40,286][DEBUG][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication2][com.hp.uca.expert.vp.cascading.**Communication2**][10] - identifier = operation_context .uca_cri_oc alarm_object 222

...

- **orchestraData**

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 = [extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave]

[2014-04-17 15:13:40,291][DEBUG][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.**Communication1**][com.hp.uca.expert.vp.cascading.**Communication1**][10] - identifier = operation_context .uca_cri_oc alarm_object 333

...

- **orchestraData**

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check server

[2014-04-17 15:13:40,291][DEBUG][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Communication2][com.hp.uca.expert.vp.cascading.**Communication2**][10] - identifier = operation_context .uca_cri_oc alarm_object 333

...

- **orchestraData**

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 = [extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave]

```
[2014-04-17 15:13:40,303][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.EnrichmentC][com.hp.uca.expert.vp.cascading.Enr  
ichmentC][ 12] - identifier = operation_context .uca_cri_oc  
alarm_object 222
```

```
.....  
- sourceScenarios = [com.hp.uca.expert.vp.cascading.Communication1,  
com.hp.uca.expert.vp.cascading.EnrichmentC]  
- sourceScenariosDescription = [cascading-join-  
3.2:com.hp.uca.expert.vp.cascading.Communication1, cascading-join-  
3.2:com.hp.uca.expert.vp.cascading.EnrichmentC]  
.....  
- orchestraData  
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =  
[extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave]  
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check  
server
```

```
[2014-04-17 15:13:40,307][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.EnrichmentC][com.hp.uca.expert.vp.cascading.Enr  
ichmentC][ 12] - identifier = operation_context .uca_cri_oc  
alarm_object 333
```

```
....  
- orchestraData  
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =  
[extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave]  
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check  
server
```

```
- var =  
-> Site [java.lang.String]  
= Sophia (France)
```

```
[2014-04-17 15:13:40,309][INFO ][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog  
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc  
alarm_object 333, t=2014-01-16T12:00:00.000+02:00, e=BOX B1,  
type=COMMUNICATIONS_ALARM, s=MAJOR, ns=NOT_CLEARED,  
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false,  
ret=false
```

```
[2014-04-17 15:13:40,318][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr  
elation][ 9] - identifier = operation_context .uca_cri_oc alarm_object  
222
```

```
-....  
- additionalInformation = Site effected by this problem is Sophia (France)  
- additionalText = Command to do to fix the problem: ps auxw  
....
```

```

- sourceScenarios = [com.hp.uca.expert.vp.cascading.Communication1,
com.hp.uca.expert.vp.cascading.EnrichmentC,
com.hp.uca.expert.vp.cascading.Correlation]

- sourceScenariosDescription = [cascading-join-
3.2:com.hp.uca.expert.vp.cascading.Communication1, cascading-join-
3.2:com.hp.uca.expert.vp.cascading.EnrichmentC, cascading-join-
3.2:com.hp.uca.expert.vp.cascading.Correlation]

- ....

- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =
{extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave}

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check
server

- var =
-> Site [java.lang.String]
= Sophia (France)

[2014-04-17 15:13:40,327][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 9] - identifier = operation_context.uca_cri_oc alarm_object
333

- ...

- additionalInformation = Site effected by this problem is Sophia (France)
- additionalText = Command to do to fix the problem: ps auxw
- ...

- sourceScenarios = [com.hp.uca.expert.vp.cascading.Communication1,
com.hp.uca.expert.vp.cascading.EnrichmentC,
com.hp.uca.expert.vp.cascading.Correlation]

- sourceScenariosDescription = [cascading-join-
3.2:com.hp.uca.expert.vp.cascading.Communication1, cascading-join-
3.2:com.hp.uca.expert.vp.cascading.EnrichmentC, cascading-join-
3.2:com.hp.uca.expert.vp.cascading.Correlation]

- ....

- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =
{extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave}

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check
server

- var =
-> Site [java.lang.String]
= Sophia (France)

```

After insertion of the *AlarmsJoinStreams2.xml* file:

INFO

```
[2014-04-17 14:54:50,777][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Communication1][com.hp.uca.expert.vp.cascading
.Communication1][ 8]Send to Orchestra AVC
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:54:50,777][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Communication2][com.hp.uca.expert.vp.cascading
.Communication2][ 8]Send to Orchestra AVC
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:54:50,783][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentC][com.hp.uca.expert.vp.cascading.Enr
ichmentC][ 8]Send to Orchestra AVC
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:54:50,786][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context.uca_cri_oc
alarm_object 222, t=2014-01-16T12:00:00.000+02:00, e=BOX B1,
type=COMMUNICATIONS_ALARM, s=MINOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=true, sc=false,
ret=false

[2014-04-17 14:54:50,787][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 8]Correlation (AVC Updated)
operation_context.uca_cri_oc alarm_object 222

[2014-04-17 14:54:50,793][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 56]==> AlarmAttributeValueChange: id=operation_context
.uca_cri_oc alarm_object 222, t=2014-01-16T13:00:05.000+01:00, e=BOX B1,
type=COMMUNICATIONS_ALARM, s=MINOR, os=NOT_ACKNOWLEDGED
```

DEBUG

```
[2014-04-17 15:14:26,080][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Communication1][com.hp.uca.expert.vp.cascading
.Communication1][ 10] - identifier = operation_context.uca_cri_oc
alarm_object 222

- ...
- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check
site Grenoble

[2014-04-17 15:14:26,080][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Communication2][com.hp.uca.expert.vp.cascading
.Communication2][ 10] - identifier = operation_context.uca_cri_oc
alarm_object 222

- ...
- orchestraData
```

```

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =
{extraInfoCommBOX2AVC=execute a ping on the IP address}

[2014-04-17 15:14:26,087][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentC][com.hp.uca.expert.vp.cascading.Enr
ichmentC][ 9] - identifier = operation_context .uca_cri_oc alarm_object
222

- ...
- sourceScenarios = [com.hp.uca.expert.vp.cascading.Communication1,
com.hp.uca.expert.vp.cascading.EnrichmentC]
- ....
- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =
{extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave}
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check
server

- var =
-> Site [java.lang.String]
= Sophia (France)

[2014-04-17 15:14:26,090][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc
alarm_object 222, t=2014-01-16T12:00:00.000+02:00, e=BOX B1,
type=COMMUNICATIONS_ALARM, s=MINOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=true, sc=false,
ret=false

[2014-04-17 15:14:26,093][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 9] - identifier = operation_context .uca_cri_oc alarm_object
222

- ....
- sourceScenarios = [com.hp.uca.expert.vp.cascading.Communication1,
com.hp.uca.expert.vp.cascading.EnrichmentC,
com.hp.uca.expert.vp.cascading.Correlation]
- ....
- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication2 =
{extraInfoComm0=check site Sophia, extraInfoComm1=check server HPslave}
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Communication1 = check
server

- var =
-> Site [java.lang.String]
= Sophia (France)

[2014-04-17 15:14:26,099][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog

```

```
ger.AcmeDbLogger][ 56]==> AlarmAttributeValueChange: id=operation_context
.uca_cri_oc alarm_object 222, t=2014-01-16T13:00:05.000+01:00, e=BOX B1,
type=COMMUNICATIONS_ALARM, s=MINOR, os=NOT_ACKNOWLEDGED
```

After insertion of the *AlarmsJoinStreams3.xml* file:

INFO

```
[2014-04-17 14:55:52,969][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.
Environmental1][ 8]Send to Orchestra alarm
```

```
operation_context.uca_cri_oc alarm_object 44
```

```
[2014-04-17 14:55:52,971][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.
Environmental2][ 8]Send to Orchestra alarm
```

```
operation_context.uca_cri_oc alarm_object 44
```

```
[2014-04-17 14:55:52,971][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.
Environmental1][ 8]Send to Orchestra alarm
```

```
operation_context.uca_cri_oc alarm_object 55
```

```
[2014-04-17 14:55:52,973][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.
Environmental2][ 8]Send to Orchestra alarm
```

```
operation_context.uca_cri_oc alarm_object 55
```

```
[2014-04-17 14:55:52,976][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.Enr
ichmentS][ 8]Send to Orchestra alarm
```

```
operation_context.uca_cri_oc alarm_object 44
```

```
[2014-04-17 14:55:52,978][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.Enr
ichmentS][ 8]Send to Orchestra alarm
```

```
operation_context.uca_cri_oc alarm_object 55
```

```
[2014-04-17 14:55:52,979][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.dbl
ogger.AcmeDbLogger][ 34]==> Alarm: id=operation_context.uca_cri_oc
alarm_object 44, t=2014-01-21T12:00:00.000+02:00, e=BOX B1,
type=ENVIRONMENTAL_ALARM, s=MAJOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false,
ret=false
```

```
[2014-04-17 14:55:52,979][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 8]Correlation - Create Group
```

```
operation_context.uca_cri_oc alarm_object 44
```

```
[2014-04-17 14:55:52,981][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.dbl
ogger.AcmeDbLogger][ 34]==> Alarm: id=operation_context.uca_cri_oc
alarm_object 55, t=2014-01-21T12:00:00.000+02:10, e=BOX B1,
type=ENVIRONMENTAL_ALARM, s=MAJOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false,
ret=false
```

```
[2014-04-17 14:55:52,982][INFO ][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr  
elation][ 8]Correlation - Create Group  
operation_context .uca_cri_oc alarm_object 55
```

DEBUG

```
[2014-04-17 15:15:05,110][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.  
Environmental2][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

...

- **orchestraData**

```
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =  
{extralInfoEnv1=check server HPmaster, extralInfoEnv0=callFireman2}
```

- var = none

```
[2014-04-17 15:15:05,113][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.  
Environmental2][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 55
```

-

- **orchestraData**

```
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =  
{extralInfoEnv1=check server HPmaster, extralInfoEnv0=callFireman2}
```

```
[2014-04-17 15:15:05,115][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.  
Environmental1][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

-

- **orchestraData**

```
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =  
callFireman1
```

```
[2014-04-17 15:15:05,116][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.  
Environmental1][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 55
```

- ...

- **orchestraData**

```
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =  
callFireman1
```

```
[2014-04-17 15:15:05,119][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.Enr  
ichmentS][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

- ...

```

- additionalInformation      = Site effected by this problem is Sophia (France)
- additionalText            = Command to do to fix the problem: ps auxw
- ...
- sourceScenarios          = [com.hp.uca.expert.vp.cascading.Environmental2,
com.hp.uca.expert.vp.cascading.EnrichmentS]
- ...
- orchestraData
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
[extralInfoEnv1=check server HPmaster, extralInfoEnv0=callFireman2]
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
callFireman1
- var                      =
-> Site [java.lang.String]
  = Sophia (France)

[2014-04-17 15:15:05,121][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc
alarm_object 44, t=2014-01-21T12:00:00.000+02:00, e=BOX B1,
type=ENVIRONMENTAL_ALARM, s=MAJOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false,
ret=false

[2014-04-17 15:15:05,121][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.Enr
ichmentS][ 10] - identifier      = operation_context .uca_cri_oc
alarm_object 55
- ...
- additionalInformation      = Site effected by this problem is Sophia (France)
- additionalText            = Command to do to fix the problem: ps auxw
- ...
- sourceScenarios          = [com.hp.uca.expert.vp.cascading.Environmental2,
com.hp.uca.expert.vp.cascading.EnrichmentS]
- ...
- orchestraData
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
[extralInfoEnv1=check server HPmaster, extralInfoEnv0=callFireman2]
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
callFireman1
- var                      =
-> Site [java.lang.String]
  = Sophia (France)

[2014-04-17 15:15:05,122][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr

```

```

elation][ 9] - identifier          = operation_context .uca_cri_oc alarm_object
44

- ....

- additionalInformation          = Site effected by this problem is Sophia (France)

- additionalText                = Command to do to fix the problem: ps auxw

- ....

- sourceScenarios              = [com.hp.uca.expert.vp.cascading.Environmental2,
com.hp.uca.expert.vp.cascading.EnrichmentS,
com.hp.uca.expert.vp.cascading.Correlation]

...

- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
[extraInfoEnv1=check server HPmaster, extraInfoEnv0=callFireman2]

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
callFireman1

- var                          =

-> Site [java.lang.String]

= Sophia (France)

[2014-04-17 15:15:05,123][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 34]==> Alarm: id=operation_context .uca_cri_oc
alarm_object 55, t=2014-01-21T12:00:00.000+02:10, e=BOX B1,
type=ENVIRONMENTAL_ALARM, s=MAJOR, ns=NOT_CLEARED,
os=NOT_ACKNOWLEDGED, ps=NOT_HANDLED, ins=true, avc=false, sc=false,
ret=false

[2014-04-17 15:15:05,125][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 9] - identifier          = operation_context .uca_cri_oc alarm_object
55

- ...

- additionalInformation          = Site effected by this problem is Sophia (France)

- additionalText                = Command to do to fix the problem: ps auxw

- ....

- sourceScenarios              = [com.hp.uca.expert.vp.cascading.Environmental2,
com.hp.uca.expert.vp.cascading.EnrichmentS,
com.hp.uca.expert.vp.cascading.Correlation]

....

- orchestraData

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
[extraInfoEnv1=check server HPmaster, extraInfoEnv0=callFireman2]

-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
callFireman1

- var                          =

-> Site [java.lang.String]

```

= Sophia (France)

After insertion of the *AlarmsJoinStreams4.xml* file:

INFO

[2014-04-17 14:56:43,625][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.Environmental2][8]Send to Orchestra AVC

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,625][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.Environmental1][8]Send to Orchestra AVC

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,629][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.Environmental1][8]Send to Orchestra SC

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,629][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.Environmental2][8]Send to Orchestra SC

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,629][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.EnrichmentS][8]Send to Orchestra AVC

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,634][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.dblogger.AcmeDbLogger][56]==> AlarmAttributeValueChange: id=operation_context .uca_cri_oc alarm_object 44, t=2014-01-21T12:00:00.000+02:05, e=BOX B1, type=ENVIRONMENTAL_ALARM, s=MAJOR, os=null

[2014-04-17 14:56:43,636][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.EnrichmentS][8]Send to Orchestra SC

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,638][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Correlation][8]Correlation (AVC Updated)

operation_context .uca_cri_oc alarm_object 44

[2014-04-17 14:56:43,640][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.dblogger.AcmeDbLogger][45]==> AlarmStateChange: id=operation_context .uca_cri_oc alarm_object 44, t=2014-01-21T12:00:00.000+02:15, e=BOX B1, type=ENVIRONMENTAL_ALARM, s=MAJOR, os=null

[2014-04-17 14:56:43,642][INFO][cascading-join-3.2][T-Scenario-com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Correlation][8]Correlation (SC Updated)

operation_context .uca_cri_oc alarm_object 44

DEBUG

```
[2014-04-17 15:15:47,199][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.  
Environmental1][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

```
- ....
```

```
- orchestraData
```

```
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 = check  
site Sophia
```

```
[2014-04-17 15:15:47,199][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.  
Environmental2][ 10] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

```
- ...
```

```
- orchestraData
```

```
-> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =  
[extraInfoEnvBOX1AVC=check site Grenoble]
```

```
[2014-04-17 15:15:47,202][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental2][com.hp.uca.expert.vp.cascading.  
Environmental2][ 9] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

```
- alarmRaisedTime = 2014-01-21T12:00:00.000+02:15
```

```
- ...
```

```
- attributeChanges
```

```
-> Attribute: networkState
```

```
New value: CLEARED
```

```
Old value: NOT_CLEARED
```

```
- orchestraData = none
```

```
[2014-04-17 15:15:47,202][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.Environmental1][com.hp.uca.expert.vp.cascading.  
Environmental1][ 9] - identifier = operation_context .uca_cri_oc  
alarm_object 44
```

```
- ...
```

```
- attributeChanges
```

```
-> Attribute: networkState
```

```
New value: CLEARED
```

```
Old value: NOT_CLEARED
```

```
- orchestraData = none
```

```
[2014-04-17 15:15:47,203][DEBUG][cascading-join-3.2][T-Scenario-  
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.Enr  
ichmentS][ 9] - identifier = operation_context .uca_cri_oc alarm_object  
44
```



```

- ...
- sourceScenarios      = [com.hp.uca.expert.vp.cascading.Environmental1,
com.hp.uca.expert.vp.cascading.EnrichmentS]
...
- attributeChanges
  -> Attribute: problemInformation
    New value: Another Problem information
    Old value:
- orchestraData
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
[extraInfoEnvBOX1AVC=check site Grenoble]
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 = check
site Sophia
[2014-04-17 15:15:47,206][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.EnrichmentS][com.hp.uca.expert.vp.cascading.Enr
ichmentS][ 9] - identifier      = operation_context .uca_cri_oc alarm_object
44
- ...
- sourceScenarios      = [com.hp.uca.expert.vp.cascading.Environmental1,
com.hp.uca.expert.vp.cascading.EnrichmentS]
- ...
- attributeChanges
  -> Attribute: networkState
    New value: CLEARED
    Old value: NOT_CLEARED
- orchestraData
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
[extraInfoEnvBOX1SC=execute a ping on the IP adress]
  -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
execute a ping on the IP adress
[2014-04-17 15:15:47,208][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 9] - identifier      = operation_context .uca_cri_oc alarm_object
44
- ...
- additionalInformation  = Site effected by this problem is Sophia (France)
- ...
- sourceScenarios      = [com.hp.uca.expert.vp.cascading.Environmental2,
com.hp.uca.expert.vp.cascading.EnrichmentS,
com.hp.uca.expert.vp.cascading.Correlation]
- attributeValueChanges  =
  -> Time: 2014/01/21 13:00:00.000 +0100
    Attribute: problemInformation

```

```

    New value: Another Problem information
    Old value:
    - customFields          = none
    - orchestraData

    -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
    [extraInfoEnv1=check server HPmaster, extraInfoEnv0=callFireman2]

    -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
    callFireman1

    - var                    =

    -> Site [java.lang.String]
    = Sophia (France)

[2014-04-17 15:15:47,208][INFO ][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.DBLogger][com.hp.uca.expert.vp.cascading.dblog
ger.AcmeDbLogger][ 45]==> AlarmStateChange: id=operation_context.uca_cri_oc
alarm_object 44, t=2014-01-21T12:00:00.000+02:15, e=BOX B1,
type=ENVIRONMENTAL_ALARM, s=MAJOR, os=null

[2014-04-17 15:15:47,212][DEBUG][cascading-join-3.2][T-Scenario-
com.hp.uca.expert.vp.cascading.Correlation][com.hp.uca.expert.vp.cascading.Corr
elation][ 9] - identifier          = operation_context.uca_cri_oc alarm_object
44

- ...

- additionalInformation      = Site effected by this problem is Sophia (France)
- additionalText            = Command to do to fix the problem: ps auxw
- ...

- sourceScenarios           = [com.hp.uca.expert.vp.cascading.Environmental2,
com.hp.uca.expert.vp.cascading.EnrichmentS,
com.hp.uca.expert.vp.cascading.Correlation]

...

- stateChanges              =

->   Time: 2014/01/21 10:45:00.000 +0100
    Attribute: networkState
    New value: CLEARED
    Old value: NOT_CLEARED

- hasAVCChanged             = false
- attributeValueChanges      = none
- customFields              = none
- orchestraData

    -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental2 =
    [extraInfoEnv1=check server HPmaster, extraInfoEnv0=callFireman2]

    -> cascading-join-3.2:com.hp.uca.expert.vp.cascading.Environmental1 =
    callFireman1

    - var                    =

```

-> Site [java.lang.String]
= Sophia (France)

The “Persistence Example” explained

This is a new example delivered with the UCA-EBC Development Kit.

This value pack contains a very simple scenario that showcases the use of the DB persistence and DB alarm forwarder features introduced with UCA-EBC 3.1.

5.1 How does it work?

The “Persistence example” value pack is configured with an H2 database so that alarms can be stored in the database using a DB forwarder. It is also configured with a DB flow so that alarms stored in the H2 database are fed into the value pack at value pack start-up, and every time an alarm is added to the DB.

Each alarm received from the network is going to be put in Working Memory and stored in a H2 database. Identifiers of alarms stored in the DB will be prefixed with the “CORRELATED-” string so that they can be distinguished from alarms coming from the network whose identifiers don’t have this prefix.

Upon new alarm reception:

- If the alarm comes from the H2 database (the identifier of the alarm has the “CORRELATED-” prefix), then this information is logged.
- If the alarm does not come from the H2 database (the identifier of the alarm doesn’t have the “CORRELATED-” prefix), then the identifier of the alarm is prefix with the “CORRELATED-” string and the alarm is put in Working Memory and also stored in the H2 database (this information is logged)

On alarm Attribute Value Change, alarm State Change or alarm Deletion, the same thing happens: if it comes from network (the identifier of the alarm doesn’t have the “CORRELATED-” prefix), it is forwarded to the DB.

5.2 Installing the example

The “Persistence example” value pack is delivered with the UCA-EBC Development Toolkit in the following folder:

```
${UCA_EBC_DEV_HOME}/vp-examples/persistence-example
```

You’ll need to build the value pack using **ant** then deploy it to a UCA-EBC Server instance using the **uca-ebc-admin** command-line tool (or the UCA-EBC Admin GUI).

Please use the following commands to build the value pack using **ant**:

On Windows:

```
$ cd %UCA_EBC_DEV_HOME%\vp-examples\persistence-example
$ ant all
```

On Linux:

```
$ cd ${UCA_EBC_DEV_HOME}/vp-examples/persistence-example
$ ant all
```

The you need to copy the value pack .zip file to the `${UCA_EBC_INSTANCE}/valuepacks` folder and deploy it using the following command:

On both HP-UX and Linux:

```
$ cd ${UCA_EBC_HOME}/bin
$ uca-ebc-admin --deploy -vpn persistence-example -vpv 3.2
```

5.3 Looking at the configuration

The H2 DB used to showcase the DB flow feature in the “Persistence example” value pack is configured in the value pack’s Spring application context file at the following location:

```
${UCA_EBC_INSTANCE}/deploy/persistence-example-
3.2/conf/context.xml
```

This database itself is located by default in the following folder:

```
${UCA_EBC_INSTANCE}/db
```

The DB flow is named `scenarioDBFlow` and is configured in the value pack’s configuration file at the following location:

```
${UCA_EBC_INSTANCE}/deploy/persistence-example-
3.2/conf/ValuePackConfiguration.xml
```

5.4 Testing the value pack

It is recommended that you first configure a `com.hp.uca.ebc.vp.examples.persistence.SimpleScenario` logger to INFO in the `${UCA_EBC_INSTANCE}/conf/uca-ebc-log4j.xml` file in order to be able to see the log messages from the “Persistence example” value pack.

Then you need to start your UCA-EBC Server instance with the “Persistence example” value pack already deployed.

When the “Persistence example” value pack starts, the H2 DB is created automatically (when the `JDBCAlarmForwarder` thread is started) if it does not exist.

In order to test the “Persistence example” value pack, please inject the value pack’s sample alarm file using the **uca-ebc-injector** command-line tool as shown below:

On Windows:

```
$ cd %UCA_EBC_INSTANCE%\deploy\persistence-example-3.2\scenario
$ %UCA_EBC_HOME%\bin\uca-ebc-injector -file Alarms.xml
```

On Linux:

```
$ cd ${UCA_EBC_INSTANCE}/deploy/persistence-example-3.2/scenario
$ ${UCA_EBC_HOME}/bin/uca-ebc-injector -file Alarms.xml
```

The following messages should be logged:

```
[2014-04-18 17:01:50,206][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.internal.ValuePackLoader][ 400]Starting Value Pack
: C:\UCA-EBC\deploy\persistence-example-3.2...

[2014-04-18 17:01:52,332][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.flow.internal.ValuePackMediationFlowImpl][ 183]Flow
Status: [persistence-example-3.2##tempFlow][Inactive]

[2014-04-18 17:01:53,985][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.expert.
rulesession.internal.RuleSession][ 326]
```

----- SCENARIO CONFIGURATION

```
Session Name      : com.hp.uca.ebc.vp.examples.persistence.SimpleScenario
Clock Mode       : NORMAL
Event Processing Mode : CLOUD
FireAllRules policy : EACH_ACCESS
Alarm eligibility policy: NetworkState!="CLEARED"
Eligible for Broadcast : true
```

----- RULE LIST

```
KnowledgePackage Name : com.hp.uca.ebc.vp.examples.persistence
-- Rule Name      : Rule [New Alarm Creation]
-- Rule Name      : Rule [Alarm Attribute Value Change]
-- Rule Name      : Rule [Alarm State Change]
-- Rule Name      : Rule [Alarm no more eligible]
```

```
[2014-04-18 17:01:53,985][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.expert.
rulesession.internal.RuleSession][ 367]
```

----- WORKING MEMORY DUMP

```
[2014-04-18 17:01:54,001][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.flow.db.AlarmFlow][ 32]DB FlowStatus: [persistence
-example-3.2##scenarioDBFlow][Starting]
```

```

[2014-04-18 17:01:54,001][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.expert.
scenario.internal.ScenarioImpl][ 267] Scenario Thread : START

[2014-04-18 17:01:54,017][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.flow.db.AlarmFlow][ 52]DB FlowSynchronization:[per
sistence-example-3.2##scenarioDBFlow][Synchronizing]

[2014-04-18 17:01:54,017][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.alarm.store.AlarmNotifier][ 133]Subscribing AlarmListe
ner com.hp.uca.expert.vp.flow.internal.DBFlow@1b0b85b8

[2014-04-18 17:01:54,032][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.alarm.store.AlarmNotifier][ 138]com.hp.uca.expert.vp.f
low.internal.DBFlow@1b0b85b8 subscribed at 1397833314017

[2014-04-18 17:01:54,032][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.flow.db.AlarmFlow][ 52]DB FlowSynchronization:[per
sistence-example-3.2##scenarioDBFlow][Synchronized]

[2014-04-18 17:01:54,032][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.flow.db.AlarmFlow][ 32]DB FlowStatus: [persistence
-example-3.2##scenarioDBFlow][Active]

[2014-04-18 17:01:54,032][INFO ][persistence-example-3.2][T-Main
][com.hp.uca.expert.vp.internal.ValuePackLoader][ 377]DB Flow scenarioDBFl
ow started

[2014-04-18 17:02:39,346][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.ebc.vp.
examples.persistence.SimpleScenario][ 71]Alarm received:
  - identifier           = 1000
  - alarmRaisedTime      = 2013-05-07T15:00:00.000+02:00
  - sourceIdentifier     = TeMIP EMS
  - originatingManagedEntity = BOX B1
  - originatingManagedEntityStructure = null
  - alarmType            = COMMUNICATIONS_ALARM
  - probableCause        = Fire
  - perceivedSeverity     = MINOR
  - networkState         = NOT_CLEARED
  - operatorState        = NOT_ACKNOWLEDGED
  - problemState         = NOT_HANDLED
  - problemInformation    = null
  - specificProblem       = null
  - additionalInformation = null
  - additionalText        = null
  - proposedRepairActions = null
  - notificationIdentifier = null

```

```

- correlationNotificationIdentifiers = null
- timeInMilliseconds      = 1367931600000 [2013/05/07 15:00:00.000 +0200]
- targetValuePack        = null
- sourceScenarios        =
[com.hp.uca.ebc.vp.examples.persistence.SimpleScenario]
- sourceScenariosDescription = [persistence-example-
3.2:com.hp.uca.ebc.vp.examples.persistence.SimpleScenario]
- passingFilters          = [test1, test2]
- passingFiltersTags      = {test1=[Tag1, Tag3, Tag2], test2=[Tag3, Tag2,
DummyNoParam]}
- passingFiltersParams    = {test1={TagX=12}, test2={DummyWithParam=123,
DummyWithEnum=F1}}
- hasParents              = false
- parentsNumber           = 0
- parents                 = null
- hasChildren             = false
- childrenNumber          = 0
- children                = null
- justInserted            = true
- aboutToBeRetracted      = false
- hasStateChanged         = false
- stateChanges            = none
- hasAVCChanged           = false
- attributeValueChanges   = none
- customFields            = none
- orchestraData           = none
- var                    = none

[2014-04-18 17:02:39,361][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.ebc.vp.
examples.persistence.SimpleScenario][ 73]Rule has fired correctly, and new alarm
has been inserted in working memory

[2014-04-18 17:02:39,361][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.ebc.vp.
examples.persistence.SimpleScenario][ 105]Alarm forwarded to DB:
- identifier              = CORRELATED-1000
- alarmRaisedTime         = 2013-05-07T15:00:00.000+02:00
- sourceIdentifier        = TeMIP EMS
- originatingManagedEntity = BOX B1
- originatingManagedEntityStructure = null
- alarmType               = COMMUNICATIONS_ALARM

```



```

- probableCause      = Fire
- perceivedSeverity   = MINOR
- networkState        = NOT_CLEARED
- operatorState       = NOT_ACKNOWLEDGED
- problemState        = NOT_HANDLED
- problemInformation   = null
- specificProblem     = null
- additionalInformation = null
- additionalText       = null
- proposedRepairActions = null
- notificationIdentifier = null
- correlationNotificationIdentifiers = null
- timeInMilliseconds   = 1367931600000 [2013/05/07 15:00:00.000 +0200]
- targetValuePack      = null
- sourceScenarios      =
[com.hp.uca.ebc.vp.examples.persistence.SimpleScenario]
- sourceScenariosDescription = [persistence-example-
3.2:com.hp.uca.ebc.vp.examples.persistence.SimpleScenario]
- passingFilters        = [test1, test2]
- passingFiltersTags     = {test1=[Tag1, Tag3, Tag2], test2=[DummyNoParam,
Tag3, Tag2]}
- passingFiltersParams   = {test1={TagX=12}, test2={DummyWithParam=123,
DummyWithEnum=F1}}
- hasParents            = false
- parentsNumber         = 0
- parents               = null
- hasChildren           = false
- childrenNumber        = 0
- children              = null
- justInserted          = false
- aboutToBeRetracted    = false
- hasStateChanged       = false
- stateChanges          = none
- hasAVCChanged         = false
- attributeValueChanges = none
- customFields
-> AlarmId = 1000
- orchestraData         = none
- var                   = none

```

[2014-04-18 17:02:40,219][INFO][persistence-example-3.2][T-Scenario-com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][71]Alarm received:

- identifier = CORRELATED-1000
- alarmRaisedTime = 2013-05-07T15:00:00.000+02:00
- sourceIdentifier = DB
- originatingManagedEntity = BOX B1
- originatingManagedEntityStructure = null
- alarmType = COMMUNICATIONS_ALARM
- probableCause = Fire
- perceivedSeverity = MINOR
- networkState = NOT_CLEARED
- operatorState = NOT_ACKNOWLEDGED
- problemState = NOT_HANDLED
- problemInformation = null
- specificProblem = null
- additionalInformation = null
- additionalText = null
- proposedRepairActions = null
- notificationIdentifier = null
- correlationNotificationIdentifiers = null
- timeInMilliseconds = 1367931600000 [2013/05/07 15:00:00.000 +0200]
- targetValuePack = persistence-example-3.2##scenarioDBFlow
- sourceScenarios =
[com.hp.uca.ebc.vp.examples.persistence.SimpleScenario]
- sourceScenariosDescription = [persistence-example-3.2:com.hp.uca.ebc.vp.examples.persistence.SimpleScenario]
- passingFilters = [test1, test2]
- passingFiltersTags = {test1=[Tag1, Tag3, Tag2], test2=[Tag3, Tag2, DummyNoParam]}
- passingFiltersParams = {test1={TagX=12}, test2={DummyWithParam=123, DummyWithEnum=F1}}
- hasParents = false
- parentsNumber = 0
- parents = null
- hasChildren = false
- childrenNumber = 0
- children = null
- justInserted = true
- aboutToBeRetracted = false

```
- hasStateChanged      = false
- stateChanges         = none
- hasAVCChanged       = false
- attributeValueChanges = none
- customFields
  -> AlarmId = 1000
- orchestraData        = none
- var                  = none
```

```
[2014-04-18 17:02:40,219][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.ebc.vp.
examples.persistence.SimpleScenario][ 73]Rule has fired correctly, and new alarm
has been inserted in working memory
```

```
[2014-04-18 17:02:40,235][INFO ][persistence-example-3.2][T-Scenario-
com.hp.uca.ebc.vp.examples.persistence.SimpleScenario][com.hp.uca.ebc.vp.
examples.persistence.SimpleScenario][ 88]This alarm was recovered from DB.
```

Glossary

DB: Database

EVP: UCA for EBC Value Pack

GUI: Graphical User Interface

JMS: Java Messaging Service

JMX: Java Management Extension, used to access or process action on the UCA for EBC product.

JNDI: Java Naming and Directory Interface

Inference engine: Process that uses a Rete algorithm

DRL: Drools Rule file

NMS: Network Management System

SDK: Software Development Kit

TT: Trouble Ticket

XML: Extensible Markup Language

XSD: Schema of an XML file, describing its structure. XSD stands for XML Schema Definition

X733: Standard describing the structure of an Alarm used in the telecommunications environment.