

Core MW Management Guide

Core Middleware

USER GUIDE

Copyright

© Ericsson AB 2015, 2016. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

Trademark List

All trademarks mentioned herein are the property of their respective owners. These are shown in the document Trademark Information.



Contents

1	Introduction	1
1.1	Prerequisites	1
2	Core MW Command Overview	3
2.1	Deprecated Command	6
3	Administrative Operations	9
3.1	Reboot Cluster	9
3.2	Reboot Node	10
3.3	Lock Node	10
3.4	Unlock Node	11
3.5	Verify System Status	11
3.6	Save IMM Data	15
3.7	Get AMF Node of Given Hostname	16
3.8	Get Hostname of Given AMF Node	16
3.9	Configure IMM Access Control	16
3.10	Enable or Disable Core MW Features	16
3.11	Set Timeout for Core MW Scaling Feature	22
3.12	Clear Alarm Issued by AMF	22
4	Core MW Partial Backup and Partial Restore – Deprecated	25
4.1	Create Core MW Partial Backup – Deprecated	25
4.2	List Core MW Partial Backups – Deprecated	26
4.3	Delete Core MW Partial Backup – Deprecated	26
4.4	Restore Using Core MW Partial Backup – Deprecated	26
5	Software Management	29
5.1	Import Software Bundles and Campaigns	29
5.2	Delete Software Bundles and Campaigns	30
5.3	Read from Software Inventory	30
5.4	Use SMF Campaigns	31
5.5	Configure ECIM SWM	37
5.6	Use CLI for Upgrade Package/CSP	39
5.7	Configure ISP Events for Cluster Restarts	42



5.8	Configure Reboot Behavior of Single-Step Procedures	42
6	Performance Management	45
6.1	Create ECIM PM Jobs	45
6.2	Start ECIM PM Jobs	49
6.3	Stop ECIM PM Jobs	50
6.4	Delete ECIM PM Jobs	50
6.5	Report Status of ECIM PM Jobs	51
6.6	List ECIM PM Jobs	52
6.7	Modify ECIM PM Jobs	54
6.8	Display Active PM Values for an Instance	56
7	OpenSAF Tools	59
7.1	OpenSAF Tools Wrapper	59



1 Introduction

This document describes how to manage the Core Middleware (MW) component.

1.1 Prerequisites

This section states the prerequisites that must be fulfilled.

1.1.1 Conditions

The following is required:

- The system must be ready to accept logon attempts from users.
- In general commands are run as root user or prefixed with `sudo` and run by user belonging to `system-adm` group. The `sudo` configuration enforces the user to provide the password before to the execution of the command. Any exception to this are detailed in the relevant sections.





2 Core MW Command Overview

After successful logon the commands shown in Table 1 are available to root user or to user belonging to group `system-adm` and prefixed by `sudo`.

Note: The exact output from any command can differ depending on the release. Command completions can be used.

Command argument completion can be used if the OS supports it and bash shell is used.

In addition to what is listed in Table 1 all commands have a `--help` attribute.

Core MW has introduced interference checks and locking between a number of commands to minimize the risk that parallel execution disrupts the system. The following areas are included:

- Campaign execution
- Backup
- Resize of Core MW

The interference locking assures that it is only possible to execute one of these commands at the same time.

Table 1 Core MW Command Overview

Core MW Commands	Description
<code>cmw-alarm-clear [-v] <alarm-type> <Managed Object></code>	Clear alarm issued by AMF. See Section 3.12 Clear Alarm Issued by AMF on page 22.
<code>cmw-amfnode-get <hostname></code>	Get the AMF node of given <code>hostname</code> . See Section 3.7 Get AMF Node of Given Hostname on page 16.
<code>cmw-campaign-commit <campaign-name></code>	Commit campaign. See Section 5.4.1 Execute an SMF Campaign on page 33.
<code>cmw-campaign-rollback <campaign-name></code>	Rollback campaign. See Section 5.4.2 Roll Back a Campaign on page 34.



Table 1 Core MW Command Overview

Core MW Commands	Description
<code>cmw-campaign-start [--disable-backup] <campaign-name></code>	Start campaign. See Section 5.4.1 Execute an SMF Campaign on page 33. If more than one campaign is executed in sequence, <code>--disable-backup</code> can be used to inhibit backups during campaign execution, except for the first campaign where a backup is recommended.
<code>cmw-campaign-status <campaign-name></code>	Show campaign status. See Section 5.4.1 Execute an SMF Campaign on page 33.
<code>cmw-campaign-stop <campaign-name></code>	Suspend campaign. See Section 5.4.1 Execute an SMF Campaign on page 33.
<code>cmw-campaign-verify <campaign-name></code>	Verify campaign. See Section 5.4.6 Initiate SMF Verification on page 36.
<code>cmw-cluster-reboot [--no-rapid-reboot] [--yes]</code>	Reboot cluster. See Section 3.1 Reboot Cluster on page 9.
<code>cmw-hostname-get <amfnode></code>	Get hostname of given AMF node. See Section 3.8 Get Hostname of Given AMF Node on page 16.
<code>cmw-node-lock <hostname> [-t <timeout> --timeout <timeout>]</code>	Lock node. See Section 3.3 Lock Node on page 10.
<code>cmw-node-unlock <hostname> [-t <timeout> --timeout <timeout>]</code>	Unlock node. See Section 3.4 Unlock Node on page 11.
<code>cmw-node-reboot [<hostname>]</code>	Reboot single node. See Section 3.2 Reboot Node on page 10.
<code>cmw-pmjob-create <job_name> <option>... [<option>...]</code>	Create an ECIM PM Job. See Section 6.1 Create ECIM PM Jobs on page 45.
<code>cmw-pmjob-start <job_name></code>	Start an ECIM PM Job. See Section 6.2 Start ECIM PM Jobs on page 49.
<code>cmw-pmjob-stop <job_name></code>	Stop an ECIM PM Job. See Section 6.3 Stop ECIM PM Jobs on page 50.
<code>cmw-pmjob-delete <job_name></code>	Delete an ECIM PM Job. See Section 6.4 Delete ECIM PM Jobs on page 50.
<code>cmw-pmjob-modify</code>	Modify an ECIM PM Job. See Section 6.7 Modify ECIM PM Jobs on page 54.
<code>cmw-pm-show-counters</code>	Display active PM values for a specified instance. See Section 6.8 Display Active PM Values for an Instance on page 56.



Table 1 Core MW Command Overview

Core MW Commands	Description
<code>cmw-pmjob-status [-v] <job_name></code>	Show ECIM PM Job status. See Section 6.5 Report Status of ECIM PM Jobs on page 51.
<code>cmw-pmjob-list [<option>...]</code>	List all defined ECIM PM Jobs. See Section 6.6 List ECIM PM Jobs on page 52.
<code>cmw-repository-list [--campaign] [--no de] [<hostname>...]</code>	List imported software bundles and campaigns. See Section 5.3 Read from Software Inventory on page 30.
<code>cmw-sdp-import <file> [<file>...]</code>	Import software bundles and campaigns. See Section 5.1 Import Software Bundles and Campaigns on page 29.
<code>cmw-sdp-remove <name> [<name>...]</code>	Delete software bundles and campaigns. See Section 5.2 Delete Software Bundles and Campaigns on page 30.
<code>cmw-status [-v] <class-name> [<class-name>...]</code>	Show the system status. Only failing items are printed unless the <code>-v</code> flag is specified. See Section 3.5 Verify System Status on page 11.
<code>cmw-swm</code>	Consume an UP/CSP. See Section 5.6 Use CLI for Upgrade Package/CSP on page 39.
<code>cmw-swm-config-set <option> [<option>...]</code>	Set the installation-dependent attributes in the ECIM SWM object. See Section 5.5 Configure ECIM SWM on page 37.
<code>cmw-imm-policy-set <option></code>	Configure IMM access control. For more information, see Section 3.9 Configure IMM Access Control on page 16.
<code>cmw-node-alarm-timeout</code>	Set the Core MW node alarm time-out. This is the time before an alarm is raised when contact with a node is lost. It is also used to determine the time when housekeeping must have finished. The housekeeping is performed after a restore when scaling is enabled (=the time that the system can have a faulty state after a restore).
<code>cmw-configuration <option></code>	Enable or disable Core MW features. For more information, see Section 3.10 Enable or Disable Core MW Features on page 16.



Table 1 Core MW Command Overview

Core MW Commands	Description
<code>cmw-timeout-configuration</code>	Set time-out for scaling batch, node join, and parallel shutdown of Core MW scaling features. For more information, see Section 3.11 Set Timeout for Core MW Scaling Feature on page 22.
<code>cmw-utility [-h --help] <immfind/immli st/immcfg/immadm/amfadm> [Options]</code>	Core MW wrapper command for OpenSAF Tools. See Section 7.1 OpenSAF Tools Wrapper on page 59.

2.1 Deprecated Command

The deprecated commands are shown in Table 2.

Table 2 Deprecated Command

Core MW Command	Description
<code>cmw-immSave</code>	<p>This command is deprecated and cannot be used. Using this command only result in the logging of a warning message.</p> <p>The IMM data is, starting with Core MW R2A, persistently saved automatically.</p>
<code>cmw-partial-backup-create [--verbose] <label></code>	<p>This command is deprecated since Core MW R6A.</p> <p>Create Core MW partial backup. See Section 4.1 Create Core MW Partial Backup – Deprecated on page 25.</p>
<code>cmw-partial-backup-remove [--verbose] <label></code>	<p>This command is deprecated since Core MW R6A.</p> <p>Delete Core MW partial backup. See Section 4.3 Delete Core MW Partial Backup – Deprecated on page 26.</p>

*Table 2 Deprecated Command*

Core MW Command	Description
<code>cmw-partial-backup-restore</code> <code>[--verbose] <label></code>	<p>This command is deprecated since Core MW R6A.</p> <p>Restore Core MW from partial backup. See Section 4.4 Restore Using Core MW Partial Backup – Deprecated on page 26.</p>
<code>cmw-partial-backup-list</code>	<p>This command is deprecated since Core MW R6A.</p> <p>List Core MW partial backups. See Section 4.2 List Core MW Partial Backups – Deprecated on page 26.</p>





3 Administrative Operations

All commands in this section are to be run as root user.

All commands apart those for “Configuring IMM Access Control” can be prefixed with `sudo` and run by user belonging to `system-adm` group.

This section describes the following administrative operations:

- Rebooting cluster
- Rebooting node
- Locking node
- Unlocking node
- Verifying system status
- Saving IMM data
- Getting Application Management Framework (AMF) node given hostname
- Getting hostname given AMF node
- Configuring IMM Access Control

3.1 Reboot Cluster

To reboot the cluster in an ordered way, use the following command:

```
cmw-cluster-reboot [--no-rapid-reboot] [--yes]
```

If `--no-rapid-reboot` is specified, the cluster reboots using regular mechanism instead of utilizing `kexec` to facilitate fast rebooting as usual.

If `--yes` is specified, the command does not require confirmation.

A cluster reboot with confirmation is shown in Example 1.



```
SC-1:~ # cmw-cluster-reboot
Really want to reboot the entire cluster (yes/no)? yes
Rebooting all nodes
Stopping DHCP daemon on node 2 (SC-2)
Stopping DHCP daemon ..done
Stopping DHCP daemon on node 1 (SC-1)
Stopping DHCP daemon ..done
Rebooting node 4 (PL-4)
Rebooting node 3 (PL-3)
Waiting for payload nodes to shut down
Payload nodes have shut down
Rebooting node 2 (SC-2)
Rebooting node 1 (SC-1)

Broadcast message from root (pts/0) (Mon Jul  5 13:07:36 2010):

The system is going down for reboot NOW!
SC-1:~ #
```

Example 1 Rebooting Cluster with Interactive Confirmation

3.2 Reboot Node

To reboot a single node, use the following command:

```
cmw-node-reboot [<hostname>]
```

If no hostname is explicitly specified, the current node is rebooted.

Note: This command is normally not used, but can be necessary to recover from transient error situations on a node that cannot be resolved using procedures with less impact on the system.

A node reboot is shown in Example 2.

```
SC-1:~ # cmw-node-reboot PL-4
Rebooting node 4 (PL-4)
SC-1:~ #
```

Example 2 Rebooting Remote Node

3.3 Lock Node

To lock an AMF node, that is to set its administrative state to `LOCKED`, use the following command:

```
cmw-node-lock <hostname> [-t <timeout> | --timeout  
<timeout>]
```

All service units hosted on the node are prohibited from providing service.

To set the time-out in seconds, use the utility `timeout` command:

```
-t, --timeout <sec>
```

If time-out is not specified, the default time-out of 900 seconds is used.



Note: The affected workload is redistributed according to the AMF redundancy model of the application.

How to lock a node is shown in Example 3.

```
SC-1:~ # cmw-node-lock PL-4
SC-1:~ #
```

Example 3 Locking Node

3.4 Unlock Node

To unlock an AMF node, that is to set its administrative state to UNLOCKED, use the following command:

```
cmw-node-unlock <hostname> [-t <timeout> | --timeout
<timeout>]
```

All service units hosted on the node are administratively allowed to provide service.

To set the time-out in seconds, use the utility `timeout` command:

```
-t, --timeout <sec>
```

If time-out is not specified, the default time-out of 900 seconds is used.

How to unlock a node is shown in Example 4.

```
SC-1:~ # cmw-node-unlock PL-4
SC-1:~ #
```

Example 4 Unlocking Node

3.5 Verify System Status

To verify that the status of the system is normal, check the status of the following system items with following command:

```
cmw-status [-v] <class-name> [<class-name>...]
```

Here `v` is verbose and `class-name` is one of the following:

- | | |
|---------------|--|
| app | Application has an administrative state that must be verified. In a “normal” system state the application must be in an unlocked state. |
| csiass | Component Service Instance has a HA state that must be verified. No <code>csi</code> must be in quiescing, or quiesced state when the system status is “normal”. |



comp	An AMF <code>comp</code> has a HA state that must be verified. No <code>comp</code> must be in <code>quiescing</code> , or <code>quiesced</code> state when the system status is “normal”. The operational state of <code>comp</code> must be enabled.
node	An executing instance of the Host OS connected to the cluster. A node can be a virtual machine.
pm	Performance Management (PM) reports overall Ericsson Common Information Model (ECIM) PM Status. If all the ECIM PM jobs are in <code>Active</code> state then it returns <code>Status OK</code> . Otherwise it provides details about ECIM PM Jobs that are in <code>Stopped</code> or <code>Faulty State</code> . The command has a short and verbose format. The short format lists only the ECIM PM job name and state for <code>Stopped</code> or <code>Faulty</code> Jobs. The verbose format lists all ECIM PM jobs, including those in <code>Active</code> states.

**sg**

A service group is a logical entity that groups one or more SUs to provide service availability for a particular set of SIs. The redundancy model defines how the SUs in the service group are used to provide service availability. Supported redundancy models are as follows:

- 2N

In this redundancy model one SU is active for all SIs, and one SU is standby for all SIs.

- N+M

Flexible redundancy with possibility to control the number of active and standbys. A common use of this redundancy model is the $N+1$ redundancy in which a single SU is standby for N active SUs.

- N-Way

In this redundancy model, it is possible for an SU to have active HA state for some SIs and at the same time have standby HA state for some other SIs.

- N-WayActive

This is a load-sharing redundancy model with only active service units.

- No Redundancy

This redundancy model is typically used with non-critical components when the failure of a component does not cause any severe impact on the overall system and a restart is good enough.

si

As Components are aggregated into SUs, the AMF supports the aggregation of CSIs into a logical entity called an SI. An SI represents a single workload assigned to the entire SU. AMF assigns HA state to the SU on behalf of one or more of SIs.

siass

Defines the SUs assigned to an SI. All SUs of the same type can be assigned Service Instances (SIs) derived from the same set of service types.

**su**

A Service Unit (SU) is a logical entity that aggregates a set of components combining their individual functionalities to provide a higher-level service. It is the unit of redundancy in the sense that it is the smallest logical entity that can be instantiated in a redundant manner. Each SU always executes on only one node, that is, it cannot be distributed over several nodes.

The components that constitute an SU can be developed in isolation, and a component developer can be unaware of which components constitute an SU, as they are defined at deployment time.

When the system is in normal status, the output can look as shown in Example 5.

```
SC-1:~ # cmw-status node comp su
Status OK
SC-1:~ #
```

Example 5 *System in Normal Status*

When verifying system status the primary class names to specify are `su`, `node`, and `comp`. If the `si` class name is specified, `PARTIALLY_ASSIGNED` can be returned even when there is no fault in the system.

On a single node cluster (1 + 0 configuration) the output can look as shown in Example 6.

```
SC-1:~ # cmw-status si
Status OK
SC-1:~ #
```

Example 6 *Single Node System with Normal SI Status*

In verbose mode on a single node cluster (1 + 0 configuration) the output can look as shown in Example 7.

```
SC-1:~ # cmw-status -v si
safSi=SC-2N,safApp=OpenSAF
AdminState=UNLOCKED(1)
AssignmentState=PARTIALLY_ASSIGNED(3)
safSi=SC-2N,safApp=ERIC-CoreMW
AdminState=UNLOCKED(1)
AssignmentState=PARTIALLY_ASSIGNED(3)
SC-1:~ #
```

Example 7 *Single Node System with Normal Verbose SI Status*

When verifying that node status and one node is locked, the output can look as shown in Example 8.

```
SC-1:~ # cmw-status node
safAmfNode=PL-3,safAmfCluster=myAmfCluster
AdminState=LOCKED-INSTANTIATION(3)
OperState=ENABLED(1)
SC-1:~ #
```

Example 8 *System with One Node Locked*



When using verbose mode for node status, the output can look as shown in Example 9.

```
SC-1:~ # cmw-status -v node
safAmfNode=SC-2,safAmfCluster=myAmfCluster
  AdminState=UNLOCKED(1)
  OperState=ENABLED(1)
safAmfNode=SC-1,safAmfCluster=myAmfCluster
  AdminState=UNLOCKED(1)
  OperState=ENABLED(1)
safAmfNode=PL-4,safAmfCluster=myAmfCluster
  AdminState=UNLOCKED(1)
  OperState=ENABLED(1)
safAmfNode=PL-3,safAmfCluster=myAmfCluster
  AdminState=UNLOCKED(1)
  OperState=ENABLED(1)
SC-1:~ #
```

Example 9 Verifying System with Verbose Mode

When verifying PM Job status, with one job in state `Stopped`, the output can look as shown in Example 10.

```
SC-1:~ # cmw-status pm
pmJobId=TC-CMW-PM-CMD-STATUS-1, Stopped
SC-1:~ #
```

Example 10 Verifying PM Job Status with One Job in Stopped State

When verifying PM Job status, using verbose mode, the output can look as shown in Example 11.

```
SC-1:~ # cmw-status -v pm
Name                               Value(s)
=====
pmJobId                            TC-CMW-PM-CMD-STATUS-1
jobType                            Measurement(0x1)
granularityPeriod                   1 minute(0x3)
reportingPeriod                     1 minute(0x3)
jobPriority                         High(0x3)
requestedJobState                   Stopped(0x2)
currentJobState                     Stopped(0x2)
Overall Job State                   Stopped(0x2)
Name                               Value(s)
=====
pmJobId                            TC-CMW-PM-CMD-STATUS-2
jobType                            Threshold(0x2)
granularityPeriod                   5 minutes(0x4)
reportingPeriod                     15 minutes(0x5)
jobPriority                         Medium(0x2)
requestedJobState                   Active(0x1)
currentJobState                     Active(0x1)
Overall Job State                   Active(0x1)
SC-1:~ #
```

Example 11 Verifying PM Job Status Using Verbose Mode

3.6 Save IMM Data

The IMM data is automatically persisted incrementally for every Configuration Change Bundle (CCB) and persistent runtime object `create/update/delete` operation. During campaign execution, the persistency is disabled and then enabled again when the campaign reaches the completed state, that is, before commit is done.



3.7 Get AMF Node of Given Hostname

To get the AMF node of a given hostname, use the following command:

```
cmw-amfnode-get <hostname>
```

An example of the output is shown in Example 12.

```
SC-1:~ # cmw-amfnode-get SC-1  
SC-1  
SC-1:~ #
```

Example 12 Getting AMF Node Given Hostname

3.8 Get Hostname of Given AMF Node

To get the hostname of a given AMF node, use the following command:

```
cmw-hostname-get <amfnode>
```

An example of the output is shown in Example 13.

```
SC-1:~ # cmw-hostname-get SC-1  
SC-1  
SC-1:~ #
```

Example 13 Getting Hostname Given AMF Node

3.9 Configure IMM Access Control

IMM Access Control is configured by the following command:

```
cmw-imm-policy-set <enforced / disabled / permissive>
```

For compatibility reasons, the default setting of IMM Access Control is DISABLED.

By setting IMM Access Control to ENFORCED, the IMM users wanting to access IMM need to belong to the `cmw-imm-users` group. For more information, refer to Section *Information Model Management Service* in *Core MW System Architecture Description*.

Note: PERMISSIVE logs all violations to system logs without enforcing access control. Must be used with caution as it can introduce unwanted spam in the system logs.

3.10 Enable or Disable Core MW Features

To enable, disable, or check status of Core MW features, use the following command:



```
cmw-configuration <--enable / --disable / --status> <FEATURE>
[--reboot]
```

The Core MW features are the following:

- --enable

Enable <FEATURE> on all nodes

- --disable

Disable <FEATURE> on all nodes

- --status

Check status of <FEATURE> (enable/disable)

- --reboot

Force cluster reboot immediately to enable or disable the <FEATURE> on all nodes. A cluster reboot is needed to enable or disable some features. Information about this can be found in the description of each feature.

Description:

The following features are supported:

- TIPC_MULTICAST
- SC_ABSENCE_ALLOWED
- SCALING
- ISP_REPORT
- ONE_STEP_UPGRADE
- PM_REPORTALWAYS

For more details on each feature, see respective subsection.

3.10.1

TIPC Multicast Feature

The Message Distribution Service (MDS) broadcasts are implemented using `unicast` which adds load and does not scale on larger clusters. The Transparent Inter-Process Communication (TIPC) multicast feature solves this issue, deleting these performance impacts.

Before enabling TIPC Multicast in Core MW, the user must verify that the current TIPC driver fully supports the TIPC multicast feature.



Note: A cluster reboot is required for this action to take effect. An immediate cluster reboot can be triggered with option `--reboot` parameter.

If TIPC Multicast is already enabled, then after a Core MW upgrade it stays enabled and if it is disabled, it stays disabled.

How to enable TIPC Multicast shown in Example 14 with forced reboot, and Example 15 without forced reboot.

```
SC-1:~ # cmw-configuration --enable TIPC_MULTICAST --reboot
Rebooting cluster ...
```

Example 14 Enable TIPC Multicast with Forced Reboot

```
SC-1:~ # cmw-configuration --enable TIPC_MULTICAST
Cluster reboot is required for change to take effect
```

Example 15 Enable TIPC Multicast without Forced Reboot

If the TIPC Multicast feature is already enabled, the output is shown in Example 16.

```
SC-1:~ # cmw-configuration --enable TIPC_MULTICAST --reboot
Already enable
```

Example 16 Enable TIPC Multicast When Already Enabled

The status of TIPC Multicast can be checked, as shown in Example 17.

```
SC-1:~ # cmw-configuration --status TIPC_MULTICAST
Enable
```

Example 17 TIPC Multicast Status Check

3.10.2

SC Absence Feature

The System Controller (SC) Absence feature ensures that the cluster does not reboot while both SC nodes are down, allowing the payload to work with limited service. If the time taken for the SC nodes to recover exceeds a `timeout` value, all payload nodes shut down (current behavior).

Before enabling SC Absence in Core MW, the user must verify that the current system fully supports this feature.



Note: A cluster reboot is required for this action to take effect. An immediate cluster reboot can be triggered with option `--reboot` parameter.

The cluster requires at least two payload nodes alive in resilient state to keep IMM data safe. This limitation can be solved from OpenSAF 5.0 with SC roaming feature.

The `timeout` is an optional argument for the `cmw-configuration` command. By default, `timeout` has the same value as the maximum `timeout` (900 seconds).

If SC Absence is already enabled, then after a Core MW upgrade it stays enabled and if it is disabled, it stays disabled.

Example 18 and Example 19 are common examples to provide more information.

```
SC-1:~ # cmw-configuration --enable SC_ABSENCE_ALLOWED 300 --reboot
Rebooting cluster ...
```

Example 18 *Enable SC Absence with Time Out of 300 Seconds and Forced Reboot*

```
SC-1:~ # cmw-configuration --disable SC_ABSENCE_ALLOWED
Cluster reboot is required for change to take effect
```

Example 19 *Disable SC Absence without Forced Reboot*

When the status of SC Absence is checked, the `timeout` value is also shown. Example 20 shows the result of checking the feature.

```
SC-1:~ # cmw-configuration --status SC_ABSENCE_ALLOWED
WARNING: Configuration has been changed but cluster reboot isn't called
Enable
Current timeout is 300
```

Example 20 *Check Status of SC Absence (After Change without Reboot)*

3.10.3

SCALING

When scaling is enabled, the system can automatically be expanded with newly detected nodes, using a default scaling profile. The CRM NBI model allows the removal of scalable nodes which triggers a resize of the system. The scaling feature is enabled in run-time and does not require a reboot.

Example 21 shows how to enable scaling.

```
SC-1:~ # cmw-configuration --enable SCALING
```

Example 21 *Enable SCALING*

Example 22 shows how to check the status of SCALING if it is already enabled.

```
SC-1:~ # cmw-configuration --status SCALING
Enable
```

Example 22 *Check Status of SCALING*



Example 23 shows how to disable scaling.

```
SC-1:~ # cmw-configuration --disable SCALING
```

Example 23 Disable SCALING

3.10.4 In Service Performance Report Feature

The ISP functionality collects ISP information that had been generated by the Core MW and other components/applications (following the ISP API description, refer to *ISP API Interface Description 1.1.0*) and produces an XML file as output for processing by the ISP Tool.

The current support is as follows:

- Partial Outage reporting: partial outage events can be reported by applications or components following the specification in the ISP API. Reporting is done using the Log service.
- NE configuration change reporting: an upgrade/update event can be reported following the specification in the ISP API. Reporting is done using the Log service.
- ISP is delivering the needed security rules to make ISP Log files visible over NBI. These are accessed by ADC tool to fetch and deliver them to the ISP Tool.
- The ISP functionality keeps the XML report files for a period of 6 months. Core MW automatically cleans the old ISP reports. This activity happens once per month at the time of creation of output XML files.

Enabling/disabling the ISP report feature does not require reboot.

Example 24 shows how to enable `ISP_REPORT` feature.

```
SC-2-1:~ # cmw-configuration --enable ISP_REPORT
```

Example 24 Enable ISP_REPORT Feature

Example 25 shows how to check status of `ISP_REPORT` feature.

```
SC-2-1:~ # cmw-configuration --status ISP_REPORT
```

Example 25 Check Status of ISP_REPORT Feature

Example 26 show how to disable `ISP_REPORT` feature.

```
SC-2-1:~ # cmw-configuration --disable ISP_REPORT
```

Example 26 Disable ISP_REPORT Feature



3.10.5 One Step Upgrade Feature

The Software Management Framework (SMF) normally executes the procedures in a campaign in execution level order. When `ONE_STEP_UPGRADE` is enabled, SMF collects all actions calculated to be made by the originally written procedures into a new internal single-step procedure.

This new single-step procedure execute all upgrade actions in a single step. The originally written procedures in the campaign are not executed.

Example 27 shows how to enable scaling.

```
SC-1:~ # cmw-configuration --enable ONE_STEP_UPGRADE
```

Example 27 Enable ONE_STEP_UPGRADE

Example 28 shows how to check the status of `ONE_STEP_UPGRADE` if it is already enabled.

```
SC-1:~ # cmw-configuration --status ONE_STEP_UPGRADE
Enable
```

Example 28 Check Status of ONE_STEP_UPGRADE

Example 29 shows how to disable `ONE_STEP_UPGRADE`.

```
SC-1:~ # cmw-configuration --disable ONE_STEP_UPGRADE
```

Example 29 Disable ONE_STEP_UPGRADE

3.10.6 PM REPORT ALWAYS

The `PM_REPORTALWAYS` feature allows runtime configuration of the report file content generated for active PM Measurement jobs. It affects both existing and subsequently created PM jobs. If `PM_REPORTALWAYS` is enabled, report content is always generated in `MISSING_MOIDS` manner, independent of each PM job's `reportContentGeneration` attribute value. If it is disabled, report content is generated according to each PM job's `reportContentGeneration` attribute value.

`PM_REPORTALWAYS` is disabled by default.

Example 30 shows how to enable `PM_REPORTALWAYS` feature.

```
SC-1:~ # cmw-configuration --enable PM_REPORTALWAYS
```

Example 30 Enable PM_REPORTALWAYS

Example 31 shows how to disable `PM_REPORTALWAYS` feature.

```
SC-1:~ # cmw-configuration --disable PM_REPORTALWAYS
```

Example 31 Disable PM_REPORTALWAYS

Example 32 shows how to check the status of `PM_REPORTALWAYS` if it is already enabled.

```
SC-1:~ # cmw-configuration --status PM_REPORTALWAYS
Enable
```

Example 32 Check Status of PM_REPORTALWAYS

3.11 Set Timeout for Core MW Scaling Feature

To set time-out for scaling batch, node join and parallel shutdown of Core MW scaling features, use the following command:

```
cmw-timeout-configuration [--list | --set <timeout>
<value>]
```

The Core MW scaling features are the following:

- `--list`

Show the current value of these timeouts.

- `--set`

Set value to these timeouts through arguments `SCALING_BATCH`, `SCALING_JOIN` and `SCALING_SHUTDOWN`.

Example 33 shows how to set node join timeout.

```
SC-1:~ # cmw-timeout-configuration --set SCALING_JOIN 120
```

Example 33 Configure Timeout Value for Node Join Cluster

3.12 Clear Alarm Issued by AMF

To clear alarm issued by AMF, use the following command:

```
cmw-alarm-clear [-v] <alarm-type> <Managed Object>
```

The `v` is verbose, Managed Object is the DN of the alarming object, and alarm-type is one of the following:

- `proxy`

To clear an alarm of type “Proxy Status of a Component Changed to Unproxied” that was raised when a component, that was previously being proxied, has currently no proxy component mediating for it.

- `cleanup`



To clear an alarm of type “AMF Component Cleanup Failed” that was raised when AMF cannot successfully clean up a software component.

- `instantiation`

To clear an alarm of type “AMF Component Instantiation Failed” that was raised when AMF cannot successfully instantiate a software component.

- `unassign`

To clear an alarm of type “AMF SI Unassigned” that was raised when a Service Instance has no active assignments to any Service Unit.

- `model_error`

To clear an alarm of type “MDF Detected Model Error” that was raised when Core MW MDF detected error during model delivery.

Example 34 shows how to clear the alarm.

```
SC-1:~ # cmw-alarm-clear instantiation \
ManagedElement=1,SaAmfApplication.safApp=ERIC-CoreMW,SaAmfSG.safSg=2N,SaAmfSU.safSu=SC-1,
SaAmfComp.safComp=EcimSwm
```

Example 34 Clear Alarm





4 Core MW Partial Backup and Partial Restore – Deprecated

Note: This section and all its subsections contain information that is **deprecated** since Core MW R6A

This section describes different partial backup and partial restore operations.

A Core MW partial backup contains a snapshot of files for Core MW, which represents the system state of Core MW. The scope of a Core MW partial backup can be extended if application programs register application-specific backup commands. The partial backup mainly acts as a driver of backup where the OS and the applications keep their backup files on their own.

The Core MW partial backup can be used to restore a system in most cases but as it does not contain all files it cannot be used for restoring the system after a catastrophic event.

Core MW supports integration of custom backup manager frameworks. If custom backup manager is registered, the responsibility for backup and restore is transferred to the backup manager. The `cmw-partial-backup-create` and `cmw-partial-backup-register` commands are disabled and returns non zero error codes when executed.

4.1 Create Core MW Partial Backup – Deprecated

Note: This section contains information that is **deprecated** since Core MW R6A

To create a Core MW partial backup, use the following command:

```
cmw-partial-backup-create [--verbose] <label>
```

A partial backup with the specified label is created.

The label can be a maximum of 128 characters and can only contain characters that are valid in a Linux[®] filename. The partial backup is physically divided in multiple archives files. The partial backup covers the following areas:

- Host OS
- Core MW
- Software bundles and campaigns imported to Core MW repository
- Data provided by registered application backup commands



How to create a Core MW partial backup is shown in Example 35.

```
SC-1:~ # cmw-partial-backup-create mgmtug-example
Snapshot starting (/cluster/snapshot/.cmwea-snapshot-mgmtug-example.tar.gz)
Snapshot completed
SC-1:~ #
```

Example 35 Creating Core MW Partial Backup

4.2 List Core MW Partial Backups – Deprecated

Note: This section contains information that is **deprecated** since Core MW R6A

To list the labels of the created partial backups, use the following command:

```
cmw-partial-backup-list
```

Only the labels of the complete partial backups are listed. For incomplete labels of partial backups a warning message is printed.

How to list a Core MW partial backup label is shown in Example 36.

```
SC-1:~ # cmw-partial-backup-list
1st-CMW
mgmtug-example
SMF-BACKUP_2010-07-05T12:48:08
SMF-BACKUP_2010-07-05T12:58:56
SC-1:~ #
```

Example 36 Listing Core MW Partial Backup Labels

4.3 Delete Core MW Partial Backup – Deprecated

Note: This section contains information that is **deprecated** since Core MW R6A

To delete a previously created partial backup, use the following command:

```
cmw-partial-backup-remove [--verbose] <label>
```

How to delete a Core MW partial backup is shown in Example 37.

```
SC-1:~ # cmw-partial-backup-remove mgmtug-example
SC-1:~ #
```

Example 37 Deleting Core MW Partial Backup

4.4 Restore Using Core MW Partial Backup – Deprecated

Note: This section contains information that is **deprecated** since Core MW R6A



To restore the system from a previously created partial backup, use the following commands:

```
cmw-partial-backup-restore [--verbose] <label>
```

```
cmw-cluster-reboot --yes
```

The partial backup must be activated by rebooting the cluster, see Section 3.1 Reboot Cluster on page 9.

How to restore using Core MW partial backup is shown in Example 38.



```
SC-1 # cmw-partial-backup-restore mgmtug-example
Restore the Host OS ...
Snapshot restore starting (/cluster/snapshot/cmwea-snapshot-mgmtug-example.tar.gz)
Snapshot restore completed
Starting RPM synchronization of node 1
Synchronizing linux-control-R1A03-PRE1.x86_64.rpm
Synchronizing =>
COREMW_COMMON-R1A-72.x86_64.716a0b126ae0b34d2f841e5cd3c539e3.rpm
Synchronizing =>
coremw-opensaf-4.0-R1A01.x86_64.09ade38a707f803470fdcf58d92f5434.rpm
Synchronizing =>
opensaf-amf-libs-4.0.RC1-R1A01.1580.4.x86_64.a8774057970069565b178c62002b893f.rpm
Synchronizing =>
opensaf-amf-nodedirector-4.0.RC1-R1A01.1580.4.x86_64.a772b062ed3638ac48a59e8bfd30e2db.rpm
Synchronizing =>
opensaf-ckpt-libs-4.0.RC1-R1A01.1580.4.x86_64.b40335b7cfa2f3592f08246fad13844c.rpm
Synchronizing =>
opensaf-ckpt-nodedirector-4.0.RC1-R1A01.1580.4.x86_64.14866c4e3fbe964cda83e97d673b7447.rpm
Synchronizing =>
opensaf-clm-libs-4.0.RC1-R1A01.1580.4.x86_64.1574c1c5e29562731573eb045d45d520.rpm
Synchronizing =>
opensaf-clm-nodeagent-4.0.RC1-R1A01.1580.4.x86_64.b7c8544f5dd86d7d33e5b8575ae3fe00.rpm
Synchronizing =>
opensaf-imm-libs-4.0.RC1-R1A01.1580.4.x86_64.59f8198c1ccbcd5ef6f21e464e19b4a9.rpm
Synchronizing =>
opensaf-imm-nodedirector-4.0.RC1-R1A01.1580.4.x86_64.769e124e417a0711c4dd4771a48a3c26.rpm
Synchronizing =>
opensaf-libs-4.0.RC1-R1A01.1580.4.x86_64.e23173b8021f50927f1328a299f793f4.rpm
Synchronizing =>
opensaf-log-libs-4.0.RC1-R1A01.1580.4.x86_64.2fbc9be1867d7c307351aeb5ebba9425.rpm
Synchronizing =>
opensaf-ntf-libs-4.0.RC1-R1A01.1580.4.x86_64.ded91cefd458156125bed888782afe73.rpm
Synchronizing =>
opensaf-smf-libs-4.0.RC1-R1A01.1580.4.x86_64.11a6b0cad1300999bf9dfa8ce4bef3d6.rpm
Synchronizing =>
opensaf-smf-nodedirector-4.0.RC1-R1A01.1580.4.x86_64.cf8cce9975734cf5d0324a24c1565ff1.rpm
Synchronizing =>
opensaf-4.0.RC1-R1A01.1580.4.x86_64.77e4dc0c1c15c20cac63097935f54cfd.rpm
Synchronizing =>
opensaf-amf-director-4.0.RC1-R1A01.1580.4.x86_64.a79972ce3e15bf7c307526459e00349f.rpm
Synchronizing =>
opensaf-ckpt-director-4.0.RC1-R1A01.1580.4.x86_64.6f6c5eaf5f6b413b4c4b88fbb65ef9b4.rpm
Synchronizing =>
opensaf-clm-server-4.0.RC1-R1A01.1580.4.x86_64.fa5baf4a1fa773e50037bb7afc5c679f.rpm
Synchronizing =>
opensaf-controller-4.0.RC1-R1A01.1580.4.x86_64.ba49b887419f2c4753f4fda58024f758.rpm
Synchronizing =>
opensaf-imm-director-4.0.RC1-R1A01.1580.4.x86_64.41b0fb79df6fd3fa25076aa43b291fce.rpm
Synchronizing =>
opensaf-log-server-4.0.RC1-R1A01.1580.4.x86_64.308b7372726cb676e9e51da7d4dbca45.rpm
Synchronizing =>
opensaf-ntf-server-4.0.RC1-R1A01.1580.4.x86_64.932bfbcb668a973b6941054e99416988f.rpm
Synchronizing =>
opensaf-smf-director-4.0.RC1-R1A01.1580.4.x86_64.50ec99d3b2b0a6a3fafdbec9257d5b43.rpm
Synchronizing =>
COREMW_SC-R1A-53.x86_64.679a7116d9d5a9758c0c8f081c99ed5b.rpm
Completed RPM synchronization of node 1
.....
Completed RPM synchronization of node 9
Unpack the Core MW backup ...
Restore application data [backup-ERIC-Vip-CXP9013048_4-R1A03] ...
SC-1 # cmw-cluster-reboot --yes
```

Example 38 Restoring Using Core MW Partial Backup



5 Software Management

This section describes the following software management tasks:

- Importing software bundles and campaigns
- Deleting software bundles and campaigns
- Reading from Software inventory
- Using SMF campaigns
- Configuring ECIM Software Management (SWM)

Commands are to be run as root user or prefixed with sudo and run by user belonging to `system-adm` group.

5.1 Import Software Bundles and Campaigns

To import software bundles or campaigns to the repository, use the following command:

```
cmw-sdp-import <file> [<file>...]
```

The possible formats are the following:

- Bundle Software Delivery Package (SDP)
- Campaign SDP
- Bundle RPM

Note: Different file formats can be mixed in the command.

The names of successfully imported software bundles and campaigns are printed.

An SDP import is shown in Example 39.

```
# cmw-sdp-import /home/MyApp/incoming/TestApp.sdp  
/home/MyApp/incoming/TestAppInstall.sdp  
ERIC-TestApp-CXP12345-R1A01 imported (type=Bundle)  
ERIC-InstallTestApp imported (type=Campaign)  
#
```

Example 39 Bundle SDP Import

A third party product (3PP) software import in Bundle RPM format is shown in Example 40.



```
# cmw-sdp-import MySQL-server-community-5.1.61-1.sles11.x86_64.rpm
3PP-MySQL-server-community-5.1.61-1.sles11 imported (type=Bundle)
#
```

Example 40 Bundle RPM Import

5.2 Delete Software Bundles and Campaigns

To delete software bundles and campaigns from the repository, use the following command:

```
cmw-sdp-remove <name> [<name>...]
```

The possible formats are the following:

- Bundle SDP
- Campaign SDP
- Bundle RPM

Note: Different formats can be mixed in the command.

How to delete an SDP is shown in Example 41.

```
# cmw-sdp-remove ERIC-TestApp-CXP12345-R1A01 ERIC-InstallTestApp
Bundle SDP removed [ERIC-TestApp-CXP12345-R1A01]
Campaign SDP removed [ERIC-InstallTestApp]
#
```

Example 41 Bundle SDP Remove

Note: Removing already removed software bundles or campaigns SDPs is not considered an error.

How to delete a Bundle RPM is shown in Example 42.

```
sc-1:~ # cmw-sdp-remove 3PP-MySQL-server-community-5.1.61-1.sles11
Bundle SDP removed [3PP-MySQL-server-community-5.1.61-1.sles11]
sc-1:~ #
```

Example 42 Bundle RPM Remove

5.3 Read from Software Inventory

To list the imported campaigns, use the following command:

```
cmw-repository-list --campaign
```

A list of imported campaigns is shown in Example 43.



```
SC-1 #
cmw-repository-list --campaign
ERIC-InstallTestApp
SC-1 #
```

Example 43 List Imported Campaigns

To list the imported software bundles and if they are used or not in the system, use the following command:

cmw-repository-list

A list of imported software bundles and if they are used or not in the system is shown in Example 44.

```
sc-1:~ # cmw-repository-list
ERIC-COREMW_COMMON-CXP9017566_1-PlA34 Used
ERIC-COREMW_SC-CXP9017565_1-PlA31 Used
ERIC-COREMW_OPENSAP-CXP9017656_1-PlA26 Used
3PP-MySQL-server-community-5.1.61-1.sles11 Used
sc-1:~ #
```

Example 44 Reading from Software Inventory

The first column shows the name of the software bundle and the second column states whether the software bundle is used, without specifying the node.

To list all software bundles installed per node, use the following command:

cmw-repository-list --node [<hostname>...]

If the `hostname` is not used, the output looks like as in Example 45.

```
SC-1:~ # cmw-repository-list --node
PL-3 ERIC-COREMW_COMMON-CXP9017566_1-PlC72
PL-3 ERIC-COREMW_OPENSAP-CXP9017656_1-R1A01
PL-4 ERIC-COREMW_COMMON-CXP9017566_1-PlC72
PL-4 ERIC-COREMW_OPENSAP-CXP9017656_1-R1A01
SC-1 ERIC-COREMW_COMMON-CXP9017566_1-PlC72
SC-1 ERIC-COREMW_OPENSAP-CXP9017656_1-R1A01
SC-1 ERIC-COREMW_SC-CXP9017565_1-PlC53
SC-1 3PP-MySQL-server-community-5.1.61-1.sles11
SC-2 ERIC-COREMW_COMMON-CXP9017566_1-PlC72
SC-2 ERIC-COREMW_OPENSAP-CXP9017656_1-R1A01
SC-2 ERIC-COREMW_SC-CXP9017565_1-PlC53
SC-2 3PP-MySQL-server-community-5.1.61-1.sles11
SC-1:~ #
```

Example 45 Reading from Software Inventory Using Variable Node

5.4 Use SMF Campaigns

All kinds of software reconfiguration such as installation, upgrade, and deletion are done with a campaign.

A campaign is contained and delivered in an SDP. To make the campaign available to the Core MW, the campaign SDP is imported using the `cmw-sdp-import` command. For more information about the `cmw-sdp-import` command, see Section 5.1 Import Software Bundles and Campaigns on page 29.

Imported campaigns can be listed using the following command:

```
cmw-repository-list --campaign
```

For more information about the command, see Section 5.3 Read from Software Inventory on page 30.

The software reconfiguration specified in a `campaign.xml` file is executed by SMF. SMF implements a state machine that interprets the XML file and executes the software reconfiguration.

The transitions between SMF states depending on Core MW commands are shown in Figure 1.

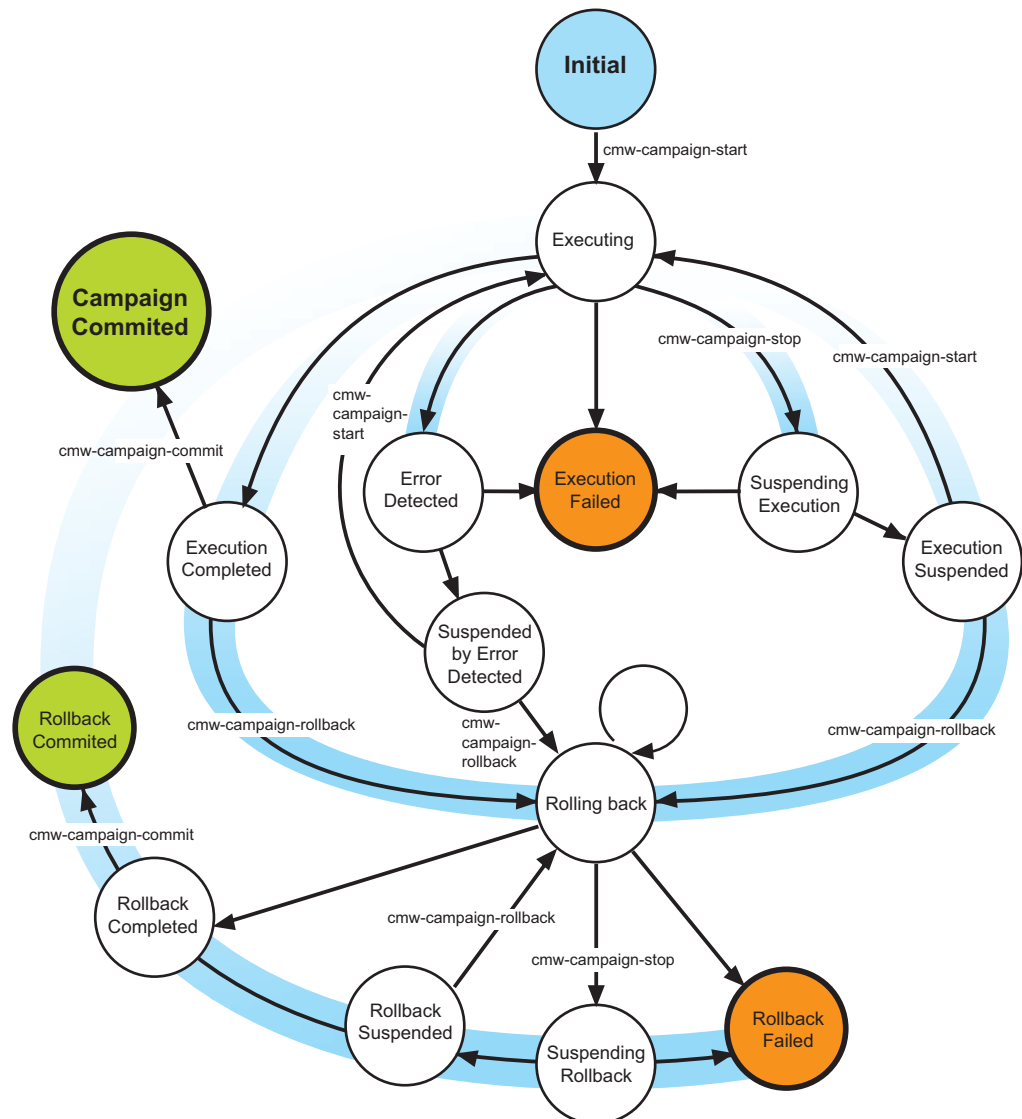


Figure 1 State Diagram



5.4.1 Execute an SMF Campaign

Make sure the units (Node, SU, component, and so on) are affected by the Campaign and that they are free of faults before executing.

To execute a campaign:

1. Get a list of the installed campaigns:

```
cmw-repository-list --campaign
```

2. Start an SMF campaign:

```
cmw-campaign-start <campaign-name>
```

If the SMF campaign is a template, the first step performed is that the campaign is updated with information fetched from the system. Any errors that occur during campaign update and validation results in an error message and the campaign execution is not started. If this occurs, a new campaign SDP must be provided, but no other measures are needed.

If more than one campaign is executed in sequence, the optional parameter `--disable-backup` can be used to inhibit backups during campaign execution. However, when the first campaign is started a backup is recommended and the optional parameter must not be used.

Note: Execution of a new campaign cannot be started until the previous campaign is committed.

3. Verify that the campaign status has state `COMPLETED`:

```
cmw-campaign-status <campaign-name>
```

If SMF detects that a campaign cannot be initiated the initial transition is `Cannot_initiate`, and the state remains `Initial`. The `Cannot_initiate` transition can be caused by an invalid campaign file, or a campaign that is already executing. Once executing, the campaign can either be successful or it can fail.

Note: As one of the first steps of campaign execution a backup is taken. A failed campaign requires that the system is restored from a backup, during which there is a service outage.

The backup is labeled according to the following syntax:

```
SMF-BACKUP_YYYY-MM-DDTHH:MM:SS
```

For example:

```
SMF-BACKUP_2010-07-05T12:48:08
SMF-BACKUP_2010-07-05T12:58:56
```

Depending on the state, different actions are to be taken, as follows:



State	Action
INITIAL without error	Retry status command.
INITIAL with error	Contact the campaign provider.
EXECUTING	Retry status command.
ERROR_DETECTED	Retry status command.
SUSPENDED_BY_ERROR_DETECTED	Either start or rollback.
COMPLETED	Proceed with commit step.
FAILED	Initiate a fallback by restoring and restarting the cluster, see Section 4.4 Restore Using Core MW Partial Backup – Deprecated on page 26.

4. Verify the status of the cluster, see Section 3.5 Verify System Status on page 11.

If the status is OK, then proceed with Step 5, otherwise repeat the status check.

5. Commit the campaign, if it executed successfully:

```
cmw-campaign-commit <campaign-name>
```

Note: Execute the `cmw-campaign-commit` command. Then execute the `cmw-campaign-status` command until the campaign is in status `COMMITTED` (it can take some time before the campaign is `COMMITTED`).

5.4.2 Roll Back a Campaign

To roll back a campaign:

1. Stop the executing campaign, if the campaign is executing:

```
cmw-campaign-stop <campaign-name>
```

2. Roll back the campaign:

```
cmw-campaign-rollback <campaign-name>
```

3. Verify that the campaign status is `ROLLBACK COMPLETED`:

```
cmw-campaign-status <campaign-name>
```

If the campaign status is `ROLLBACK COMPLETED`, continue to Step 4.

If the campaign status is `ROLLBACK FAILED`, initiate a fallback by restoring and restarting the cluster, see Section 4.4 Restore Using Core MW Partial Backup – Deprecated on page 26.



4. Commit the campaign, if it rolled back successfully:

```
cmw-campaign-commit <campaign-name>
```

5. Verify that the status is ROLLBACK_COMMITTED:

```
cmw-campaign-status <campaign-name>
```

If the status still is ROLLBACK_COMPLETED, retry the `cmw-campaign-commit` command.

Note: Execute the `cmw-campaign-rollback` command. Then execute the `cmw-campaign-status` command until the campaign is in status ROLLBACK_COMMITTED (it can take some time before the campaign is ROLLBACK_COMMITTED).

5.4.3

Add Software

Applications are installed using a campaign. The application is contained in one or more software bundles and the installation campaign in an SDP. The installation campaign is target system size specific.

To install an application using a campaign:

1. Copy the installation campaign SDP and the software bundles to a location at the target system, for example:

```
/home/MyApp1/incoming
```

2. Import all campaign SDPs and software bundles:

```
cmw-sdp-import /home/MyApp1/incoming/<installation-campaign-filename>
```

```
cmw-sdp-import /home/MyApp1/incoming/<application-filename>
```

3. Execute the installation campaign, see Section 5.4.1 Execute an SMF Campaign on page 33.
4. Delete the installation campaign SDP:

```
cmw-sdp-remove <installation-campaign>
```

5.4.4

Upgrade Software

Any software delivered as a software bundle is upgraded using a campaign. The new version of the software is contained in a software bundle and the upgrade campaign in a campaign SDP.

To upgrade software using a campaign:



1. Copy the campaign SDP and the software bundles to a location at the target system, for example:

```
/home/MyApp1/incoming
```

2. Import all campaign SDPs and software bundles:

```
cmw-sdp-import /home/MyApp/incoming/<upgrade-campaign-  
filename>
```

```
cmw-sdp-import /home/MyApp/incoming/<new-software-filena  
me>
```

3. Execute the upgrade software campaign, see Section 5.4.1 Execute an SMF Campaign on page 33.
4. Delete the campaign SDP and the old application software bundles:

```
cmw-sdp-remove <upgrade-campaign> <old-software-bundle-na  
me>...
```

5.4.5 Delete Software

Software is deleted using a campaign. The removal campaign is contained in an SDP. The removal campaign is target system specific.

To delete software using a campaign:

1. Import the removal campaign:

```
cmw-sdp-import /home/MyApp/incoming/<removal-campaign-  
sdp-filename>
```

2. Execute the removal campaign, see Section 5.4.1 Execute an SMF Campaign on page 33.

3. Delete the removal campaign SDP and the software bundle:

```
cmw-sdp-remove <removal-campaign> <software-bundle-name>...
```

5.4.6 Initiate SMF Verification

An application can provision itself with the ability to influence the progress of an SMF campaign by registering a callback function during application startup. Once the callback is registered, it is called whenever a campaign is initiated, see Section 5.4.1 Execute an SMF Campaign on page 33.

To initiate campaign verification manually and hence execute all currently registered callbacks, use the following command:

```
cmw-campaign-verify <campaign-name>
```



The command returns `exit code 0` on success and `exit code 1` on any failure.

Verification can also be initiated through `ECIM SWM verify()` which makes use of `cmw-campaign-verify`.

5.5 Configure ECIM SWM

If ECIM for SWM is to be used, then the installation-dependent attributes in the ECIM SWM object must be set.

For more information about these attributes, refer to *Core MW Software Management* and *Managed Object Model cmw_SwM*.

These attributes cannot be set on COM CLI. To set value to them, use the following command:

```
cmw-swm-config-set
```

The command takes the options shown in Table 3.



Table 3 Command Options

Options	Description
-l, --localFilePath	<p>localFilePath is used when creating Upgrade Package MOs using action createUpgradePackageLocal. Each local UP to be represented by a UP MO must be put in a unique directory directly under localFilePath.</p> <p>For more details, refer to createUpgradePackageLocal action in <i>Managed Object Model cmw_SwM</i>.</p> <p>Expectations on the file store path are as follows:</p> <ul style="list-style-type: none">• The directory structure must be created before setting cmw-swm-config-set.• The directory must be accessible by both controllers.• The directory must be writable to delete the packages as part of the removeUpgradePackage() operation in the ECIM SWM MOM. <p>Since no default setting exists, this attribute must be assigned for the ECIM SWM functionality to work.</p>
-s, --subType	<p>subType is the name of the node type. The subType describes the Managed Element, for example, "smallSystem", "mergedSystem". The attribute is used to select the correct campaign to apply from an Upgrade Package.</p> <p>For more detailed description about how it works, see Section <i>Element MESubtype</i> in <i>Core MW Software Management</i>.</p>
-b, --automaticBackup	<p>automaticBackup is set to 1 (true) if ECIM SWM is to take an automatic backup at activation.</p>



Options	Description
<code>-r, --automaticRestore</code>	<code>automaticRestore</code> is set to 1 (true) in order for ECIM SWM to perform automatic restore at failure.
<code>-f, --fallbackTimer</code>	<code>fallbackTimer</code> specifies the maximum number of seconds ECIM SWM waits for user action before fallback. The default setting is 1200.
<code>-a, --alarmBeforeTimeout</code>	<code>alarmBeforeTimeout</code> specifies the time in seconds that an alarm is sent out before a fallback occurs. The default setting is 180.

Configuration of ECIM SWM is shown in Example 46.

```
cmw-swm-config-set --automaticBackup 1 -s smallCluster -l /home/my/package/repository
```

Example 46 Configuring ECIM SWM

The command activates automatic backup, set `subType` to `smallCluster` and the local file store path to `/home/my/package/repository`.

5.6 Use CLI for Upgrade Package/CSP

Core MW is able to consume an UP/CSP by using the command `cmw-swm`. This command allows users not using COM to create an UP/CSP and upgrade a CBA system using the produced package. It also allows components/applications to use the same upgrade scenario with a limited CBA stack deployed. The command takes the options shown in Table 4.

Table 4 Command Options

Option	Description	Example
<code>--up-list</code>	Show a list ID of UPs or CSPs in system. The number following UP's ID is the action ID of the Create action that created the UP.	List UPs in the system: \$ <code>cmw-swm --up-list</code> Output: ERIC-Test-UP150922123955-P1A01 1.



Option	Description	Example
--up-create	<p>Create a new UP/CSP instance from URI - points to the directory where the UP/CSP content is located. The URI is local (for example file:///data/dir/subdir, /storage/system/.../UP.tgz) or a remote (for example sftp://hostname/dir/subdir).</p> <p>An UP is active when it has been created but not removed or confirmed. No other UPs can be created if another UP is active. If the user tries to do that error messages similar to the following are displayed:</p> <ul style="list-style-type: none">• CMW: error-string: @ComNbi@SwM::createUpgradePackage UpgradePackage=ERIC-Test-UP150922123955-P1A01 is still active• CMW: error - saImmOmAdminOperationInvoke_2 admin-operation RETURNED: SA_AIS_ERR_BUSY (10)• CMW: ERROR (cmw-swm): Failed to execute create UP/CSP command for uri: file:///SoftwareManagement/cmw-csp-KQJd-UP2.tgz <p>The result of a successful execution of this action is an action ID. User can use the action ID together with output of the cmw-swm --up-list command to identify the UP_ID, which is uniquely associated with each UP/CSP.</p>	<p>Examples of create option:</p> <ul style="list-style-type: none">• Create UP from local file path: \$ cmw-swm --up-create /home/UP.tgz• Create UP from remote location, protected with password: \$ cmw-swm --up-create sftp://user@hostname/dir/UP_dir pa55w0rd
--up-prepare	Prepare the Managed Element (ME) for the activation of the UP/CSP.	\$ cmw-swm --up-prepare ERIC-Test-UP150922123955-P1A01
--up-activate	Activate UP/CSP step by step or all at once.	\$ cmw-swm --up-activate ERIC-Test-UP150922123955-P1A01
--up-confirm	Confirm the upgrade procedure on the ME.	\$ cmw-swm --up-confirm ERIC-Test-UP150922123955-P1A01
--up-cancel	Cancel the upgrade procedure before confirmation or cancel Activate action at breakpoints between steps. This option can also be used to cancel long running actions such as Prepare or Activate.	\$ cmw-swm --up-cancel ERIC-Test-UP150922123955-P1A01



Option	Description	Example
<code>--up-remove</code>	Remove an UP/CSP from the ME	<code>\$ cmw-swm --up-remove ERIC-Test-UP150922123955-P1A01</code>
<code>--up-status</code>	Get state of specific UP/CSP ID, track the progress and result of actions. Output format: Action: <i><Action name></i> - <i><State of the action></i> Result: <i><Final result of the action></i> State: <i><State of UP></i> (this line is only displayed when a UP ID is specified in the command).	Examples of the option are: • Status of the last Create or Remove action. In this case, since UP is not created or removed, no UP ID can be specified. <code>\$ cmw-swm --up-status</code> Action: CreateUpgradePackage - FINISHED Result: SUCCESS • Status of a UP after other actions: <code>\$ cmw-swm --up-status ERIC-Test-UP150922123955-P1A01</code> Action: Prepare - RUNNING Result: NOT_AVAILABLE State: PREPARE_IN_PROGRESS

An example of upgrading software using `cmw-swm` is shown in the following steps:

1. `cmw-swm --up-create /storage/system/software/coremw/repository/UP/ERIC-ActivateCommand_UP/ERIC-ActivateCommand_UP.tgz`

The command creates an UP/CSP from the file argument. UP/CSP can be created from `file://`, `sftp://` or path of file. Use `cmw-cwm --up-list` to verify that a new UP/CSP was created and to get the `UP_ID` of UP/CSP. The user can also use `cmw-swm --up-status` (without `<UP_ID>`) to see the progress and result of the previous command. The status of the upgrade package is received by using `cmw-swm --up-status <UP_ID>`. Verify that the state of the upgrade package is INITIALIZED.

2. `cmw-swm --up-prepare <UP_ID>`

The command prepares the ME for the activation of the UP/CSP with `<UP_ID>`. The prerequisite of this action is that the state of the UP/CSP is INITIALIZED. This step is considered completed when the state of the UP/CSP is PREPARE_COMPLETED.

3. `cmw-swm --up-activate <UP_ID>`

The command activates the UP/CSP step by step or all at once. If the UP/CSP is designed with more than one step and `ignoreBreakpoints` is set to false, Activate action is called several times.

4. `cmw-swm --up-confirm <UP_ID>`

The command gives the final confirmation that the upgrade procedure is considered complete. The expected state is COMMIT_COMPLETED.

5. `cmw-swm --up-remove <UP_ID>`



The command removes the UP/CSP Managed Object (MO) given as action parameter. The command also removes all files temporarily stored in the ME and associated with the UP/CSP.

5.7 Configure ISP Events for Cluster Restarts

ISP events for cluster restarts are disabled by default.

This feature can be enabled or disabled during Automatic Installation Tool (AIT) installation of Core MW. For more information, see Section *Automatically Installing Core MW by Using AIT in Core MW SW Installation*.

ISP events for cluster restarts can be enabled within a campaign by setting the `ispClusterRebootLogEnabled` attribute of object with class type `CmwMonitorIsp` to a value of 1.

An example of a campaign Init section which enables ISP events for cluster restarts by setting the `ispClusterRebootLogEnabled` attribute to 1 is shown in Page 42.

```
<campInitAction>
  <immCCB ccbFlags="0">
    <modify objectDN="CMW_GETDN(^CmwMonitorIspId=1,CmwMonitorId=1,
      CmwSysConfigId=1$)" operation="SA_IMM_ATTR_VALUES_MODIFY">
      <attribute name="ispClusterRebootLogEnabled" type="SA_IMM_ATTR_SAUINT32T">
        <value>1</value>
      </attribute>
    </modify>
  </immCCB>
</campInitAction>
```

Example 47 ISP Events for Cluster Restarts

5.8 Configure Reboot Behavior of Single-Step Procedures

SMF single-step procedures are typically used for installation (or removal) of components/applications where a component/application is not in-service upgradeable. If a single-step upgrade procedure contains software bundles, which require a reboot to install or remove, SMF can either perform a cluster reboot or it can perform nodewise reboot of only the affected nodes.

The default SMF behavior is to perform nodewise reboot during single-step upgrade procedures.

To define the affected nodes for nodewise reboot, the `campaign.xml` must specify them in the `<singleStepUpgrade>` section, as shown in Example 48.



```
<singleStepUpgrade>
  <upgradeScope>
    <forAddRemove>
      <deactivationUnit>
        <actedOn>
          <byName objectDN="safAmfNode=SC-2,safAmfCluster=myAmfCluster"/>
        </actedOn>
      </deactivationUnit>
      <activationUnit>
        <actedOn>
          <byName objectDN="safAmfNode=SC-2,safAmfCluster=myAmfCluster"/>
        </actedOn>
      </activationUnit>
    </forAddRemove>
  </upgradeScope>
</singleStepUpgrade>
```

Example 48 *Campaign Defining Affected Nodes for Single-Step Upgrade Procedures*

The `smfSSAffectedNodesEnable` attribute of the SMF configuration controls the nodewise reboot feature and is set to a default value of 1. To change SMF behavior to reboot all nodes in the cluster during a single-step upgrade procedure, a campaign must disable nodewise reboot as shown in Example 49.

```
<campWrapupAction>
  <immCCB ccbFlags="0">
    <modify operation="SA_IMM_ATTR_VALUES_REPLACE" objectDN="smfConfig=1,
      safApp=safSmfService">
      <attribute name="smfSSAffectedNodesEnable" type="SA_IMM_ATTR_SAUINT32T">
        <value>0</value>
      </attribute>
    </modify>
  </immCCB>
</campWrapupAction>
```

Example 49 *Campaign Disabling Nodewise Reboot Feature*





6 Performance Management

PM jobs can be created, modified, deleted, started, and stopped programmatically using the IMM Object Management (OM) API.

This section describes the following shell command support for managing ECIM PM jobs:

- Creating an ECIM PM Job
- Starting an ECIM PM Job
- Stopping an ECIM PM Job
- Deleting an ECIM PM Job
- Reporting status of an ECIM PM Job
- Listing ECIM PM Jobs
- Modifying an ECIM PM Job

This section also describes the following PM operations:

- Display active PM values for an instance

Commands are to be run as root user or prefixed with sudo and run by user belonging to `system-adm` group.

6.1 Create ECIM PM Jobs

To create an ECIM PM job, use the `cmw-pmjob-create` command.

The command is used as follows:

```
cmw-pmjob-create <job_name>
                  (-u <pmgroup_id>
                   [-M <meas_reader>]
                   [-R <variation_rate>]
                   [-d <thresholdDirection>]
                   [(-i <mo_instance>)...])
                  [-m <meas_type>]
                  [-T <threshold_id>
                   -H <threshold_high>
                   -L <threshold_low> -S <severity>]
                  ) ...
                  [-t <job_type>]
                  [-r <reporting_period>]
```



```

[-g <granularity_period>]
[-p <job_priority>]
[-q <requested_state>]
[-c <job_control>]

```

The syntax is to be interpreted as follows:

- `job_name` is the `pmJobId` attribute in the `EcimMoClass PmJob` table.
- `pmgroup_id` is the Managed Object Class (`moClass`) name of the Measurement Type (MT)
- `meas_type` is the MT
- The part within () can occur multiple times, as indicated with three dots.

The command supports any number of Measurement Readers (MR) and Measurement Specifications (MS).

For threshold jobs, up to four thresholds object are supported.

The MS can use a direct reference to an MT or it can use indirect reference by referring to a `moClass`.

A direct reference to an MT occurs when the `moClass` (`-u` option) and the MT (`-m` option) is specified for the MR.

An indirect reference to a group occurs when only the `moClass` (`-u` option) is specified for the MR. However, indirect references are only valid for measurement jobs (`-t 1`) and are not valid for threshold jobs (`-t 2`).

The optional parameters for threshold jobs with default values in bold, are as follows:

```

-T <threshold_id>
  -H <threshold_high>
  -L <threshold_low>
  -S <severity>
      3 - CRITICAL
      4 - MAJOR
      5 - MINOR
      6 - WARNING
-R <variation_rate>
      0 - PER_SECOND
      1 - PER_GP
      Note: it only applies for CC collection method.
-d <thresholdDirection>
      1 - INCREASING
      2 - DECREASING

```

The optional Job parameters are as follows:

```

-i <mo_instance>

```



A specific MT instance to be referenced.
 If not specified, all MT instances will be referenced.
 It supports any number of instance parameters.
 The format is as follows:
 -i instance1 -i instance2 -i instance3

-t <job_type>
 1 - Measurement Job
 2 - Threshold Job

-r <reporting_period>
 3 - 1 minute
 4 - 5 minutes
 5 - 15 minutes
 6 - 30 minutes
 7 - 1 hour
 8 - 12 hours
 9 - 24 hours

-g <granularity_period>
 3 - 1 minute
 4 - 5 minutes
 5 - 15 minutes
 6 - 30 minutes
 7 - 1 hour
 8 - 12 hours
 9 - 24 hours

-p <job_priority>
 1 - low
 2 - medium
 3 - high

-q <requested_state>
 1 - Active
 2 - Stopped

-c <job_control>
 0 - Full
 1 - Start stop
 2 - View only

-G <job_group>

-x <compression_type>
 0 - GZIP

The creation of an ECIM PM job named Job3, which directly references MT rxOctets in moClass mocRx, is shown in Example 50.



```
SC-1:~ # cmw-pmjob-create job3 -u mocRx -m rxOctets
PM job job3 created
SC-1:~ #
```

Example 50 Create an ECIM PM Job

The command returns exit code 0 on success and exit code 1 on any failure to create the job.

Attempting to create an ECIM PM job without first defining the MT or moClass, results in failure, as shown in Example 51.

```
SC-1:~ # cmw-pmjob-create job4 -u mocTx -m txOctets
Creating job job4...
Failed to apply object creations 21
CMW: ERROR (cmw-pmjob-create): Job creation from command line failed.
SC-1:~ #
```

Example 51 Failed Attempt to Create an ECIM PM Job

How to create an ECIM PM Job that references an MT indirectly by the moClass mocRxRate (by not specifying the -m option), is shown in Example 52.

```
SC-1:~ # cmw-pmjob-create job4 -u mocRxRate
PM job job4 created.
SC-1:~ #
```

Example 52 Create an ECIM PM Job That Indirectly References an MT

Multiple MT references can be applied to the same job, as shown in Example 53.

```
SC-1:~ # cmw-pmjob-create job5 -u mocRx
-m rxDiscFr
-T rxDiscAlm
-H 300 -L 250 -S 3
-u mocRxRate
-m rxRate
-T rxRateOversub
-H 600 -L 550 -S 3
-T rxRateHigh
-H 400 -L 350 -S 4
-T rxRateBusy
-H 200 -L 150 -S 6
-t 2
PM job job5 created.
```

Example 53 Create an ECIM PM Job with Multiple MTs

Job5 is defined as a Threshold type job (-t flag set to 2), as shown in Example 53.

It contains two MR references which can be described as follows:

- The first MR directly references an MT named rxDiscFr in moClass mocRx. It defines a PmThresholdMonitoring threshold named rxDiscAlm of severity class 3 (CRITICAL). The high threshold is set to 300 and the low threshold is set to 250.



- The second MR directly references an MT named `rxRate` in the `moClass` `moCRxRate`. It defines three `PmThresholdMonitoring` thresholds (`rxRateOversub`, `rxRateHigh`, and `rxRateBusy`), each which has an assigned severity and threshold levels.

The `job_control` attribute is optional. The default value is `Full`, which means that the job can be started, stopped, or deleted. The `PMJob2` is defined as a measurement job, which has `job_control` as `Start stop`, as shown in Example 54.

```
SC-2-1:~ # cmw-pmjob-create PMJob2 -u PMGroup-1
          -t 1 -r 3 -g 3
          -p 3 -q 2 -c 1
```

Example 54 Create an ECIM PM Job with Start Stop Control

6.2 Start ECIM PM Jobs

To start an ECIM PM job, use the `cmw-pmjob-start` command.

Note: The precondition is that `job_control` is `Full` or `Start stop`.

The command is used as follows:

```
cmw-pmjob-start <job_name>
```

The syntax is to be interpreted as follows:

`job_name` is the `pmJobId` attribute in the `EcimMoClass PmJob` table.

A request to start an ECIM PM job is shown in Example 55.

```
SC-1:~ # cmw-pmjob-start TC-CMW-PM-JOB-CMD-START
PM Job TC-CMW-PM-JOB-CMD-START Started.
SC-1:~ #
```

Example 55 Start an ECIM PM Job

The command returns exit code 0 on success and exit code 1 on any failure to start the job.

A failure to start an ECIM PM Job, whose `job_name` does not exist, is shown in Example 56.

```
SC-1:~ # cmw-pmjob-start TC-CMW-PM-JOB-CMD-START_non-exist
CMW: ERROR (cmw-pmjob-start): PM Job =>
      TC-CMW-PM-JOB-CMD-START_non-exist does not exist
SC-1:~ #
```

Example 56 Starting a Non-existing ECIM PM Job

Attempting to start an already started ECIM PM job results in exit code 0 with the following message:

Warning: PM Job <job_name> is already Active.



6.3 Stop ECIM PM Jobs

To stop an ECIM PM job, use the `cmw-pmjob-stop` command.

Note: The precondition is that `job_control` is `Full` or `Start stop`.

The command is used as follows:

```
cmw-pmjob-stop <job_name>
```

The syntax is to be interpreted as follows:

`job_name` is the `pmJobId` attribute in the `EcimMoClass PmJob` table.

A request to stop an ECIM PM job is shown in Example 57.

```
SC-1:~ # cmw-pmjob-stop TC-CMW-PM-JOB-CMD-STOP
PM job TC-CMW-PM-JOB-CMD-STOP stopped.
SC-1:~ #
```

Example 57 Stop an ECIM PM Job

The command returns exit code 0 on success and exit code 1 on any failure to stop the job.

Attempting to stop an already stopped ECIM PM job results in an exit code 0 with the following message:

Warning: PM Job <job_name> is already Stopped.

6.4 Delete ECIM PM Jobs

To delete a stopped ECIM PM job, use the `cmw-pmjob-delete` command.

Note: The precondition is that `job_control` is `Full`.

The command is used as follows:

```
cmw-pmjob-delete <job_name>
```

The syntax is to be interpreted as follows: `job_name` is the `pmJobId` attribute in the `EcimMoClass PmJob` table.

A request to delete an ECIM PM job is shown in Example 58.

```
SC-1:~ # cmw-pmjob-delete TC-CMW-PM-JOB-CMD-CC-ECIM-001
PM job TC-CMW-PM-JOB-CMD-CC-ECIM-001 deleted.
SC-1:~ #
```

Example 58 Delete an ECIM PM Job

The command returns exit code 0 on success and exit code 1 on any failure to delete the job.

Attempting to delete an `Active` ECIM PM job results in exit code 1 with the following message:



```
CMW: ERROR (cmw-pmjob-delete): PM Job <job_name> is
Active - can not be deleted.
```

Attempting to delete a non-existent ECIM PM job results in an exit code 1 with the following message:

```
CMW: ERROR (cmw-pmjob-delete): PM Job <job_name> does not
exist.
```

6.5 Report Status of ECIM PM Jobs

To report the status of an ECIM PM job, use the **cmw-pmjob-status [-v] <job_name>** command.

The command is used as follows:

```
cmw-pmjob-status [-v] <job_name>
```

The syntax is to be interpreted as follows:

- **job_name** is the **pmJobId** attribute in the **EcimMoClass PmJob** table.
- The optional **-v** flag produces a verbose output. Omitting the **-v** flag only produces the job name and status.

The output using the short format, generated when omitting the **-v** flag, is shown in Example 59.

```
SC-1:~ # cmw-pmjob-status TC-CMW-PM-JOB-CMD-STATUS-2
PM job TC-CMW-PM-JOB-CMD-STATUS-2 is active
SC-1:~ #
```

Example 59 Reporting Status of an ECIM PM Job

Using the **-v** flag, the long format, for the same job is shown in Example 60.

```
SC-1:~ # cmw-pmjob-status -v TC-CMW-PM-JOB-CMD-STATUS-2
Name                               Value(s)
=====
pmJobId                            TC-CMW-PM-JOB-CMD-STATUS-2
jobType                            Measurement(0x1)
granularityPeriod                   1 minute(0x3)
reportingPeriod                     1 minute(0x3)
jobPriority                         High(0x3)
requestedJobState                   Active(0x1)
currentJobState                    Active(0x1)
Overall Job State                   Active(0x1)
SC-1:~ #
```

Example 60 Reporting Status of an ECIM PM Job Using Verbose Mode

If the job does not exist or if the status cannot be shown, the command returns exit code 1 and logs the following message:

```
CMW: ERROR (cmw-pmjob-status): Could not show the status
for PM Job <job_name>.
```



Otherwise the command returns exit code 0.

6.6 List ECIM PM Jobs

To list ECIM PM jobs, use the `cmw-pmjob-list [options]` command.

The command is used as follows:

cmw-pmjob-list [options]

The possible options are as follows:

- v Use long format for each job
- h --help - display this help and exit
- f List only jobs with problems (faulty)
- r <N> List only jobs with this reporting period <N>
- t <N> List only jobs of this job type <N>
- p <N> List only jobs with this job priority <N>
- g <N> List only jobs with this granularity period <N>
- q <N> List only jobs with this requested job state <N>
- s <N> List only jobs with this current job state <N>

where <N> is a numeral.

Using the command without any options results in a short listing, showing each ECIM PM Job name, requested job state, and current job state. A short listing is shown in Example 61.

```
SC-1:~ # cmw-pmjob-list
pmJobId=TCPMJCA_0, requestedJobState=Stopped(0x2), currentJobState=Stopped(0x2)
pmJobId=TCPMJCA_1, requestedJobState=Active(0x1), currentJobState=Active(0x1)
pmJobId=TCPMJCA_2, requestedJobState=Stopped(0x2), currentJobState=Stopped(0x2)
pmJobId=TCPMJCA_3, requestedJobState=Active(0x1), currentJobState=Active(0x1)
pmJobId=TCPMJCA_4, requestedJobState=Stopped(0x2), currentJobState=Stopped(0x2)
SC-1:~ #
```

Example 61 Listing ECIM PM Jobs

The command contains a `-v` flag for Verbose or long format. The output contains detailed information about each ECIM PM Job. A verbose listing is shown in Example 62.



```

SC-1:~ # cmw-pmjob-list -v
Name                               Value(s)
=====
pmJobId                           Job1
jobType                           Measurement(0x1)
granularityPeriod                  15 minutes(0x5)
reportingPeriod                    5 minutes(0x4)
jobPriority                         Medium(0x2)
requestedJobState                  Active(0x1)
currentJobState                   Active(0x1)
Overall Job State                 Active(0x1)
Name                               Value(s)
=====
pmJobId                           Job2
jobType                           Threshold(0x2)
granularityPeriod                  15 minutes(0x5)
reportingPeriod                    1 minute(0x3)
jobPriority                         Medium(0x2)
requestedJobState                  Active(0x1)
currentJobState                   Active(0x1)
Overall Job State                 Active(0x1)
Name                               Value(s)
=====
pmJobId                           Job3
jobType                           Threshold(0x2)
granularityPeriod                  15 minutes(0x5)
reportingPeriod                    1 hour(0x7)
jobPriority                         Medium(0x2)
requestedJobState                  Active(0x1)
currentJobState                   Active(0x1)
Overall Job State                 Active(0x1)
SC-1:~ #

```

Example 62 Listing ECIM PM Jobs Using Verbose Mode

Multiple options can be combined in one command to “OR” the result. For example, using multiple options on the same job list as in Example 62, jobs that have reportingPeriod equal to 1 minute and jobType equal to Measurement can be shown. A filtered list, using multiple options, is shown in Example 63.

```

SC-1:~ # cmw-pmjob-list -v -r 3 -t 1
Name                               Value(s)
=====
pmJobId                           Job1
jobType                           Measurement(0x1)
granularityPeriod                  15 minutes(0x5)
reportingPeriod                    5 minutes(0x4)
jobPriority                         Medium(0x2)
requestedJobState                  Active(0x1)
currentJobState                   Active(0x1)
Overall Job State                 Active(0x1)
Name                               Value(s)
=====
pmJobId                           Job2
jobType                           Threshold(0x2)
granularityPeriod                  15 minutes(0x5)
reportingPeriod                    1 minute(0x3)
jobPriority                         Medium(0x2)
requestedJobState                  Active(0x1)
currentJobState                   Active(0x1)
Overall Job State                 Active(0x1)
SC-1:~ #

```

Example 63 Listing ECIM PM Jobs Using Multiple Options

Multiple options can be used in short format (without the -v flag), as shown in Example 64.



```

SC-1:~ # cmw-pmjob-list -r 3 -t 1 -g 5
pmJobId=Job2, requestedJobState=Active(0x1), =>
    currentJobState=Active(0x1), reportingPeriod=3
pmJobId=Job1, requestedJobState=Active(0x1), =>
    currentJobState=Active(0x1), jobType=1
pmJobId=Job1, requestedJobState=Active(0x1), =>
    currentJobState=Active(0x1), granularityPeriod=5
pmJobId=Job2, requestedJobState=Active(0x1), =>
    currentJobState=Active(0x1), granularityPeriod=5
pmJobId=Job3, requestedJobState=Active(0x1), =>
    currentJobState=Active(0x1), granularityPeriod=5
SC-1:~ #

```

Example 64 Listing ECIM PM Jobs in Short Format with Multiple Flags

6.7 Modify ECIM PM Jobs

To modify a stopped ECIM PM job, use the `cmw-pmjob-modify` command.

The command is used as shown in Example 65.

```

cmw-pmjob-modify <job_name> add MOSs &| remove MOSs &| modify attributes
Add MeasurementReader MOSs:
    [( -C
        [ -M <meas_reader> ]
        [ -R <thresholdRateOfVariation> ]
        [ -d <thresholdDirection> ]
        [ ( -i <mo_instance> ) ... ]
        -u <pmgroup_id>
        [ -m <meas_type> ]
    )...]
Add PmThresholdMonitoring MOSs:
    [( -C
        -M <meas_reader>
        ( -T <threshold_id> -H <threshold_high> -L <threshold_low> -S <severity> )
    )...]
Add both MeasurementReader and PmThresholdMonitoring MOSs:
    [( -C
        [ -M <meas_reader> ]
        [ -R <thresholdRateOfVariation> ]
        [ -d <thresholdDirection> ]
        [ ( -i <mo_instance> ) ... ]
        -u <pmgroup_id>
        -m <meas_type>
        ( -T <threshold_id> -H <threshold_high> -L <threshold_low> -S <severity> )
    )...]

remove MOSs:
    [( -D
        -M <meas_reader>
        [ -T <threshold_id> ]
    )...]

modify attributes:
    [ -r <reporting_period> ]
    [ -g <granularity_period> ]
    [ -p <job_priority> ]
    [ -G <job_group> ]
    [ -x <compression_type> ]

```

Example 65 Modify ECIM PM Jobs

This command requires `-C` to indicate child MO creation, and `-D` to indicate child MO deletion. All the other options have the same meanings as with the `cmw-pmjob-create` command. The part within `()` can occur multiple times.



The addition of a MeasurementReader MO to an existing job named job3 is shown in Example 66.

```
SC-1:~ # cmw-pmjob-modify job3 -C -u mocRx -m rxOctets
Modifying job job3...
PM job job3 modified.
SC-1:~ #
```

Example 66 *Add a MeasurementReader MO to an Existing ECIM PM Job*

The command returns exit code 0 on success and exit code 1 on any failure to create the MO.

The addition of both a MeasurementReader MO and its child PmThresholdMonitoring MO to an existing threshold job named job5 is shown in Example 67.

```
SC-1:~ # cmw-pmjob-modify job5 -C -M mrRxRate -u mocRxRate -m rxRate -T
rxRateOversub -H 50 -L 45 -S 3
Modifying job job5...
PM job job5 modified.
SC-1:~ #
```

Example 67 *Add Both a MeasurementReader MO and its Child PmThresholdMonitoring MO to an Existing ECIM PM Job*

The addition of a PmThresholdMonitoring MO to an existing MeasurementReader MO of an existing job named job5 is shown in Example 68. Since PmThresholdMonitoring MO is a child of a MeasurementReader MO, the -M parameter must be specified.

```
SC-1:~ # cmw-pmjob-modify job5 -C -M mrRxRate -T rxRateOversub -H 50 -L 45 -S 3=>
Modifying job job5...
PM job job5 modified.
SC-1:~ #
```

Example 68 *Add a PmThresholdMonitoring MO to an Existing MeasurementReader MO*

The deletion of a MeasurementReader MO from an existing job named job3 is shown in Example 69.

```
SC-1:~ # cmw-pmjob-modify job3 -D -M mrRxRate
Modifying job job3...
PM job job3 modified.
SC-1:~ #
```

Example 69 *Delete a MeasurementReader MO from an Existing ECIM PM Job*

The deletion of a PmThresholdMonitoring MO from an existing threshold job named job5 is shown in Example 70. The -M parameter must be specified.

```
SC-1:~ # cmw-pmjob-modify job5 -D -M mrRxRate -T rxRateOversub
Modifying job job5...
PM job job5 modified.
SC-1:~ #
```

Example 70 *Delete a PmThresholdMonitoring MO from an Existing ECIM PM Job*

Modifying one or more ECIM PM job attributes for an existing job named `job3` is shown in Example 71.

```
SC-1:~ # cmw-pmjob-modify job3 -p 2 -x 0 -G group1
Modifying job job3...
PM job job3 modified.
SC-1:~ #
```

Example 71 Modify Attributes for an Existing ECIM PM Job

6.8 Display Active PM Values for an Instance

To display active PM values for a specified instance, use the command `cmw-pm-show-counters`.

Note: The precondition is that the instance is associated with measurement types being measured by an active ECIM PM job.

The command is used as follows:

```
cmw-pm-show-counters [-v] <instance> [-t <timeout>] [-j  
<job_name> <job_name> ...] ] [-m <meas_type> [<meas_type>  
...] ]
```

The syntax is interpreted as follows:

- `-v, --verbose` verbose
- `-t, --timeout <secs>`
- `job_name` is the `pmJobId` attribute of an active `PmJob` `EcimMO`
- `meas_type` is the `measurementTypeId` attribute of a `MeasurementType` `EcimMO`

The command displays the current value of all measurement types associated with the specified instance.

If job names are specified only instance values from the specified jobs are displayed.

If measurement types are specified, only instance values associated with the specified measurement types are displayed.

If time-out is not specified, the default time-out of 6 seconds is used.

The command returns exit code 0 on success and exit code 1 on any failure.

An example of the output is shown in Example 72.



```
cmw-pm-show-counters MeasObj=1  
  intCounterActive=123 SUSPECT  
  multivalueIntCounter=12,14,16
```

Example 72 Display Current Instance Values

An example of verbose output is shown in Example 73.

```
cmw-pm-show-counters -v MeasObj=1  
  intCounterActive=123 PmJob=1 Gp=5 min SUSPECT  
  multivalueIntCounter=12,14,16 PmJob=1 Gp=5 min
```

Example 73 Display Vebose Current Instance Values





7 OpenSAF Tools

This section provides information about OpenSAF tools.

7.1 OpenSAF Tools Wrapper

The `cmw-utility` command can be used to run the following OpenSAF tools:

- `immfind`
- `immlist`
- `immcfg`
- `immadm`
- `amfadm`

The syntax is:

```
cmw-utility [-h|--help]
<immfind|immlist|immcfg|immadm|amfadm> [Options]

    -h, --help
        Print help menu.
```

The options for each OpenSAF tool are specified in the following sections.

7.1.1 **immfind**

Search for IMM objects.

The syntax is:

```
cmw-utility immfind [path...] [options]
    -c, --class=NAME
        Only search for objects of the specified class.
    -t, --timeout=TIMEOUT
        Utility timeout in seconds.
```

Example: `cmw-utility immfind --class SaAmfCluster`

7.1.2 **immlist**

List IMM objects.

The syntax is:

```
cmw-utility immlist [options] <object name> [object name]
cmw-utility immlist [options] <class name>
-a, --attribute=NAME
    Specify attribute name to print.
-c, --class=NAME
    Print class definition.
-t, --timeout=TIMEOUT
    Utility timeout in seconds.
```

Examples:

- `cmw-utility immlist safAmfCluster=myAmfCluster`
- `cmw-utility immlist --attribute SaImmAttrClassName
CmwMonitorId=1,CmwSysConfigId=1`
- `cmw-utility immlist --class CmwMonitor`

7.1.3 immcfg

Create, delete or modify IMM objects.

The syntax is:

```
cmw-utility immcfg ([options] [object name])...
-a, --attribute name[+|-]=value
    Change attribute.
-c, --create-object <class name>
    Create an object of specified class.
-d, --delete-object [object name]...
    Delete objects.
-m, --modify-object [object name]...
    Modify objects.
-u, --unsafe
    The CCBs generated by immcfg will have
    SA_IMM_CCB_REGISTERED_OI
    set to false, allowing ccb commit when OIs are missing.
-t, --timeout=TIMEOUT
    Utility timeout in seconds.
```

Examples:

- `cmw-utility immcfg --create-object CmwPmPmJob
pmJobId=CmwUtilityTestJobObj,CmwPmpmId=1`
- `cmw-utility immcfg --attribute requestedJobState=2
CmwPmPmJob pmJobId=CmwUtilityTestJobObj,CmwPmpmId=1`



- `cmw-utility immcfg --delete-object pmJobId=CmwUtilityTestJobObj,CmwPmpmId=1`

7.1.4 immadm

Perform an IMM admin operation.

The syntax is:

```
cmw-utility immadm [options] [object name]...
--disable-tryagain
    Disable try again handling [default=no].
-o, --operation-id=<id>
    Numerical operation ID.
-p, --parameter=<p>
    Parameter(s) to admin op.
    Parameter syntax: <name>:<type>:<value>
    Value types according to imm.xsd.
    Valid types: SA_INT32_T, SA_UINT32_T, SA_INT64_T,
    SA_UINT64_T, SA_TIME_T, SA_NAME_T, SA_FLOAT_T,
    SA_DOUBLE_T, SA_STRING_T.
-t, --timeout=TIMEOUT
    Utility timeout in seconds.
```

Example: `cmw-utility immadm --operation-id 1 safJob=CmwUtilityTestJobObj,safPm=1`

7.1.5 amfadm

Perform an AMF admin operation.

The syntax is:

```
cmw-utility amfadm [options] <command> <object name>
-s, --su
    Specified object DN is checked to be a SU before
    performing AMF admin operation.
-t, --timeout=TIMEOUT
    Utility timeout in seconds.
Commands: lock, unlock, lock-in, unlock-in, shutdown,
restart, si-swap, sg-adjust, repaired, eam-start,
eam-stop.
```

Examples:

- `cmw-utility amfadm --su lock safSu=SC-1,safSg=2N,safApp=ERIC-CoreMW`
- `cmw-utility amfadm unlock safSu=SC-1,safSg=2N,safApp=ERIC-CoreMW`