

# Software Management

---

## DESCRIPTION

**Copyright**

© Ericsson AB 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

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Functions and Concepts</b>	<b>3</b>
2.1	Types of Operation	7
<b>3</b>	<b>Managed Object Model</b>	<b>9</b>
<b>4</b>	<b>Configuration Management</b>	<b>11</b>
<b>5</b>	<b>Fault Management</b>	<b>13</b>





# 1 Introduction

This document provides an overview of the management model and concepts associated with the Software Management managed area.

A managed area is represented by a group of Managed Object Classes (MOCs) within the Managed Object Model (MOM).





## 2 Functions and Concepts

Software Management provides a management interface to upgrade and delete software installed on the Managed Element (ME).

### Software Upgrade

A software upgrade is needed in the following situations:

- Adding new software domains or software items to the ME.
- Deleting or replacing software domains or software items on the ME.

For more information on software domains and software items, refer to *Software Inventory Management*.

A patch installation and a major software upgrade are both handled as software upgrades by Software Management.

A software upgrade is achieved through the installation of an Upgrade Package (UP). An UP is a collection of Load Modules and upgrade control information delivered by the Ericsson supply organization.

Depending on its design, a software UP can upgrade one or several software domains at a time.

Software management is able to perform rolling upgrades, that is, software upgrades that are not causing any downtime to the ME services. Rolling upgrades are applicable only for software domains and UPs designed for this purpose.

A software upgrade is divided into two main phases; a preparation phase and an execution phase. The preparation phase takes place during normal working hours. The execution phase takes place during “low traffic” hours.

### Preparation Phase

The preparation phase has the following responsibilities:

- Verifies that the ME has passed a health check routine.
- Makes the UP visible in the MOM using metadata from the UP file.
- Downloads the necessary parts of the UP content (for example, required Load Modules) from an external server.
- Performs the relevant software UP extraction activities or integrity checks. The integrity check performs checksum checks or equivalent (oblivious

hashing, check and guard system, active or passive tamper resistance) and is needed to secure that the correct upgrade package has been fetched.

- In case the UP is a CSP 1.2 or above, finalizes the system configuration that is put inside the package and backs up the old configuration to support the rollback of the UP later.
- If the UP is a CSP containing ESM configurations, compares the specified target to the current system as the base and generates the upgrade campaign and the target configuration.
- Verifies the ME ability to activate the UP. For example, it checks that the UP is consistent and matches the software version of the ME. This verification can also occur during the execution phase, depending on the design of the UP.
- Allows the user to configure the upgrade package upgrade execution method.

### Execution Phase

The execution phase has the following responsibilities:

- Verifies that the ME has passed a health check routine.
- Verifies the ME ability to activate the UP (if it is designed in this way).
- If the UP is a CSP containing ESM configurations and the system has changed since the completion of the preparation phase, regenerates the upgrade campaign and the target configuration.
- Activates (that is, applies) the UP on the ME.
- Provides an observation window that allows the user to monitor the success of the upgrade.
- Triggers automatic or manual fallback in failure situations.
- Commits the upgrade based on an explicit user request.
- Finalizes the target configuration.

The execution phase can be achieved using a step-by-step or one-step activation, depending on the design of the UP.

The first (or only) activation step triggers a system backup. It is recommended to keep this default behavior.

A step is a breakpoint. It represents a part of the upgrade after which the ME functionality can be observed manually and require user interaction. An UP with multiple steps, and therefore multiple breakpoints enable the user to verify that each step has been correctly executed.





After successfully completing a step-by-step or one-step activation, the ME waits for a final confirmation (that is, commit) from the user. With a commit, the user confirms that observations of the ME functions have been completed satisfactorily. After a commit, the user can no longer cancel an upgrade.

### **Fallback Operation**

The upgrade activation procedure stops and the ME starts a fallback operation in the following cases:

- The ME encounters a critical problem that prohibits a successful activation.
- The user cancels the activation operation after an intermediary step or after the last step.
- The fallback timer expires during the activation. The fallback timer is the maximum number of seconds between the activate operation finishing and the next operation (activate or commit) starting.

The ME raises the alarm *A Fallback Operation will soon be started* to indicate these conditions.

The fallback operation triggers a backup restore, which restores the ME to the state it had before the activation procedure started. It is recommended to keep this default behavior. The fallback operation is not supposed to fail under any circumstances.

### **Software Upgrade Package Life Cycle**

During the preparation and execution phases, a software UP goes through different life cycle states. An overview is provided in Figure 1.

### **Software Removal**

To gain disk storage space, the user can delete an inactive software version from the ME. An inactive software version is deleted with care. That is, software items shared with other software versions are not deleted. An upgrade can also automatically delete an obsolete software version from the ME.

The active software version cannot be deleted.

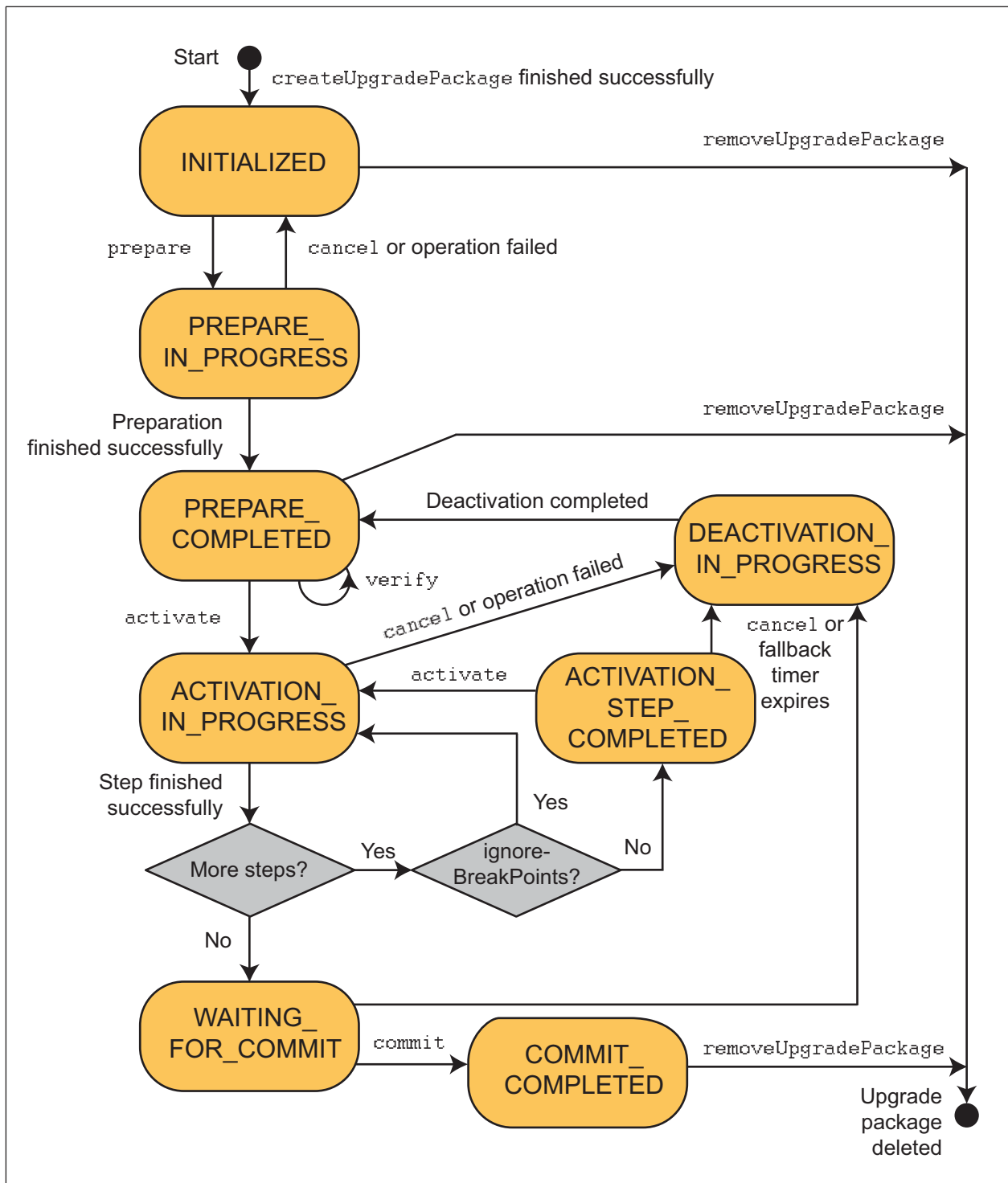


Figure 1 Life Cycle of an Upgrade Package



## 2.1 Types of Operation

Software Management supports the following operations:

### Software Upgrade

- Prepare UP

The user can initiate a UP preparation in the following two ways:

- Choose to specify over the Northbound Interface (NBI) from which Uniform Resource Identifier (URI) to retrieve the UP file.
- Download the UP file on the ME using the SSH File Transfer Protocol (SFTP).

The procedure in *Prepare Upgrade Package* provides further details on how to perform this operation.

- Configure upgrade execution method for the UP

The user can configure the upgrade to be executed as a Rolling, Single-Step, or Balanced In-Service upgrade.

For further details on how to perform these operations, refer to *Prepare Upgrade Package*.

- Upgrade using one-step activation

This operation is applicable to UPs designed for one-step or step-by-step activations. However, it is not recommended to apply it to a UP designed for multiple steps, since it implies reduced user interaction and increases the upgrade failure probability. The procedure in *Prepare Upgrade Package* provides further details on how to perform this operation.

- Upgrade using step-by-step activation

This operation is applicable only to UPs designed for a step-by-step activation.

When applying such UP, the user turns on step-wise execution. It makes it possible to interact with the ME after each step and to closely attend the correct progression of the upgrade. The procedure in *Prepare Upgrade Package* provides further details on how to perform this operation.

- Activate UP

The user can activate the execution phase of the UP to apply and commit the software upgrade. The procedure in *Activate Upgrade* provides further details on how to perform this operation.

- Cancel an upgrade operation



The user can perform different types of cancel operations:

- Cancel an ongoing *prepare* action.
- Cancel an ongoing activation step.
- Cancel a completed intermediary activation step during a step-by-step activation.
- Cancel the final and completed activation step (before action *commit*).

The procedure in *Cancel Upgrade Operation* provides further details on how to perform this operation.

An ongoing operation is usually canceled when it has been started by mistake, at the wrong time, or takes too long time.

A completed activation step is usually canceled when the user observation and assessment of the ME health check status is not according to expectations.

- Delete a UP

Once a UP has been applied and is no longer needed on the ME, the user can delete the corresponding Managed Object (MO). The procedure in *Delete Upgrade Package* provides further details on how to perform this operation.

If the upgrade file initially was retrieved from a URI, this operation deletes it together with the MO. If the UP file initially was downloaded with SFTP, this operation does not delete it from the ME.

## Software Removal

- Delete an inactive software version

The user can delete an inactive software version from the ME. This operation can be used to save disk space when an inactive software version, which is no longer needed, has not been deleted automatically by the latest software upgrade. The procedure in *Delete Inactive Software Version* provides further details on how to perform this operation.

Any attempt to delete an active software version fails.

### 3 Managed Object Model

The Software Management managed area is represented in the *Managed Object Model (MOM)* as follows:

```
ManagedElement
+-SystemFunctions
+-SwM
+-SwVersionMain
+-UpgradePackage
```

For general information about the MOM, MOCs, MOs, cardinality, and related concepts, refer to *Managed Object Model User Guide*.

The Software Management MOCs are described in Table 1.

*Table 1 Software Management Managed Object Class Descriptions*

Managed Object Class	Description
<i>SwM</i>	The root of the Software Management model. Supports the creation and deletion of <i>UpgradePackage</i> MOs, the deletion of inactive software versions, and cancellation of activation operations. Provides control of some software management capabilities such as automatic backup and automatic restore. Indicates the active software versions.
<i>SwVersionMain</i>	Indicates which software versions are present on the ME.
<i>UpgradePackage</i>	Handles the preparation phase and the execution phase for a UP except the <i>UpgradePackage</i> MO creation.





## 4 Configuration Management

Software Management is accessed using NETCONF or the Ericsson Command-Line Interface (ECLI) to manipulate the Management Information Base (MIB).

The following operations can be performed by the user and are described in Operating Instructions using the ECLI:

### **Software Upgrade**

- *Prepare Upgrade Package*
- *Activate Upgrade*
- *Cancel Upgrade Operation*
- *Delete Upgrade Package*

### **Software Removal**

- *Delete Inactive Software Version*







## 5 Fault Management

The Software Management alarm is described in Table 2.

*Table 2 Software Management Alarm*

Alarm	Description
<i>A Fallback Operation will soon be started</i>	Issued during a software upgrade process when waiting on action <i>commit</i> or <i>activate</i> to prevent the upgrade from being automatically canceled.