

# Upgrade Using Step-by-Step Activation

---

## OPERATING INSTRUCTIONS

**Copyright**

© Ericsson AB 2014, 2015. 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>
1.1	Prerequisites	1
<b>2</b>	<b>Procedure</b>	<b>3</b>





# 1 Introduction

This document describes how to perform the execution phase of a software upgrade.

A software upgrade has a limited time window during which some planned service impact is allowed. The execution phase of the software upgrade can have service impact and must be executed only within the upgrade time window and not during normal “traffic hours”.

The procedure in this document covers the following:

- How to activate (that is, apply) the upgrade package on the Managed Element (ME)
- How to commit the upgrade

This document describes how to upgrade using a step-by-step-activation. For upgrade packages designed for step-by-step activations, the procedure in this document supports a step-wise execution with several break points. This allows the user to check and interact with the ME after each step.

The procedure is illustrated by an example where a software version with product name `ERIC-COREMW_RUNTIME`, product number `CXP9020355_1`, and product revision `R7E01` is running in the system. A software upgrade package `ERIC_UP-CXP9020355_1-R7F01`, which is designed to upgrade this software version step-by-step to product revision `R7F01`, has already been prepared and is going through the activation phase. This upgrade package has an activation fallback-timer, attribute `activationFallbackTimer` in the Managed Object (MO) *UpgradePackage* equal to 1200 seconds.

## 1.1 Prerequisites

This section describes the prerequisites, which must be fulfilled before using the procedure.

### 1.1.1 Conditions

The following conditions must apply:

- The ME has passed a health check routine.
- The upgrade package is prepared.
- The fallback capability is supported by the ME, that is, attribute `timeoutFallbackCapability=SUPPORTED` in the *SwM* MO.



- Attribute `ignoreBreakPoints` is set to `true`.
- An Ericsson Command-Line Interface (ECLI) session in Exec mode is in progress.



## 2 Procedure

To upgrade using step-by-step-activation:

1. Navigate to the upgrade package, for example:

```
>dn ManagedElement=NODE06ST,SystemFunctions=1,SwM=1,UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
```

2. Enter Config mode:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >configure
```

3. Turn on step-wise execution:

```
(config-UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >ignoreBreakPoints=false
```

4. Commit the setting:

```
(config-UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >commit
```

5. Activate the first step:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >activate
```

The system returns output `true` for a successfully triggered activation or `false` otherwise.

6. Verify the result for a successfully triggered activation:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```

In the following example output, `state=RUNNING` and `progressPercentage=15`. It shows that 15% of the activation is completed.

```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
[...]
state=ACTIVATION_IN_PROGRESS <read-only>
[...]
reportProgress <read-only>
[...]
actionName="Activate" <read-only>
[...]
progressInfo="Activate UpgradePackage" <read-only>
progressPercentage=15 <read-only>
[...]
state=RUNNING <read-only>
[...]
```

7. Continue to check the progress of the first step until it is completed:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```



The following example output shows the final result with `state=FINISHED` and `progressPercentage=100`.

```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
[...]  
state=ACTIVATION_STEP_COMPLETED <read-only>  
[...]  
reportProgress <read-only>  
[...]  
progressInfo="Activate UpgradePackage" <read-only>  
progressPercentage=100 <read-only>  
result=SUCCESS <read-only>  
resultInfo="Campaign execution suspended by breakpoint" <read-only>  
state=FINISHED <read-only>  
step=1 <read-only>  
[...]
```

8. Verify that the ME behaves properly after the software upgrade activation. This includes, but is not necessarily limited to, passing a health check routine.

9. Is the ME state or behavior according to expectations?

Yes: Proceed with the next step.

No: Cancel the activation, refer to *Cancel Upgrade Operation*.

10. Activate the next step:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >activate
```

The system returns output `true` for a successfully triggered activation or `false` otherwise.

11. Verify the result for a successfully triggered activation:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```

In the following example output, `state=RUNNING` and `progressPercentage=30`. It shows that 30% of the activation is completed.

```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
[...]  
state=ACTIVATION_IN_PROGRESS <read-only>  
[...]  
reportProgress <read-only>  
[...]  
progressInfo="Activate UpgradePackage" <read-only>  
progressPercentage=30 <read-only>  
[...]  
state=RUNNING <read-only>  
[...]
```

12. Continue to check the progress of this activation step until it is completed:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```

The following example output shows the resulting state of an activation step which is followed by at least one more activation step:





```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01[...]
state=ACTIVATION_STEP_COMPLETED <read-only>
[...]
reportProgress <read-only>
[...]
progressInfo="Activate UpgradePackage" <read-only>
progressPercentage=50 <read-only>
result=SUCCESS <read-only>
resultInfo="Campaign execution suspended by breakpoint" <read-only>
state=FINISHED <read-only>
step=2 <read-only>
[...]
```

**Note:** State `ACTIVATION_STEP_COMPLETED` implicitly indicates that there is at least one more activation step to execute in this example.

The following example output shows the resulting state of a final activation step:

```
UpgradePackage=ERIC_UP-CXP9020355_1-R7F01[...]
state=WAITING_FOR_COMMIT <read-only>
[...]
reportProgress <read-only>
[...]
progressInfo="Activate UpgradePackage" <read-only>
progressPercentage=100 <read-only>
result=SUCCESS <read-only>
resultInfo="Campaign execution successfully completed" <read-only>
state=FINISHED <read-only>
step=2 <read-only>
[...]
```

**Note:** State `WAITING_FOR_COMMIT` indicates that the final activation step has been executed successfully and that the upgrade can be committed.

13. Verify that the ME state or behavior is according to expectations at this breakpoint. This is specific for the software upgrade and outside the scope of this document.
14. Is the ME state or behavior according to expectations?  
 Yes: Proceed with the next step.  
 No: Cancel the activation, refer to *Cancel Upgrade Operation*.
15. Select operation according to the value of parameter `state` in the output:
  - If `state=ACTIVATION_STEP_COMPLETED`, proceed with Step 10 to activate the next step.
  - If `state=WAITING_FOR_COMMIT`, proceed with the next step.




---

---

## Do!

Carefully read the following information before proceeding.

---

---



Once the activation is effective, indicated by `state=WAITING_FOR_COMMIT`, the fallback countdown starts. Attribute `timeRemainingBeforeFallback` is automatically set to the value (in this example 1200 seconds) contained in attribute `fallbackTimer` in the *SwM* MO.

If the user does not commit the operation before `timeRemainingBeforeFallback` reaches zero, a fallback is automatically triggered.

16. Navigate to the *SwM* MO:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >up
```

17. Verify how much time remains before fallback:

```
(SwM=1) >show timeRemainingBeforeFallback
```

The following example output shows that 1031 seconds remain before fallback:

```
1031
```

18. Navigate to the upgrade package:

```
(SwM=1) >UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
```

19. Commit the upgrade by using action `commit` in the *Upgradepackage* MO:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >commit
```

The system returns output `true` or `false`.

**Note:** A Software Management upgrade is confirmed/committed by executing the Software Management action `commit` in Exec mode. This is different from ECLI command `commit` used to apply configuration changes in Config mode.

20. Check the progress of action `commit`:

```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >show -v
```

The following is an example output of the final result for a successful upgrade in three steps:

```
[...]
state=COMMIT_COMPLETED <read-only>
[...]
reportProgress <read-only>
[...]
  progressPercentage=100 <read-only>
  result=SUCCESS <read-only>
  resultInfo="Campaign committed successfully. Updated model successfully." <read-only>
  state=FINISHED <read-only>
  step=3 <read-only>
[...]
```

21. Navigate to the *SwM* MO:



```
(UpgradePackage=ERIC_UP-CXP9020355_1-R7F01) >up
```

22. Verify that the active software state is the expected one:

```
(SwM=1) >show -v
```

The following is an example output. The active software version corresponds to the activated software upgrade scope. A software version with product name ERIC-COREMW\_RUNTIME, product number CXP9020355\_1, and product revision R7F01 is running in the system.

```
SwM=1
  activeSwVersion
    "ManagedElement=NODE06ST,SystemFunctions=1,SwM=1,SwVersionMain=>
ERIC-COREMW_RUNTIME-CXP9020355_1-R7F01" <read-only>
[...]
```

```
  SwVersionMain=ERIC-COREMW_RUNTIME-CXP9020355_1-R7F01
  UpgradePackage=ERIC_UP-CXP9020355_1-R7F01
```

A software version with product name ERIC-COREMW\_RUNTIME, product number CXP9020355\_1, and product revision R7F01 is now committed and running in the system.