

Start Measurement Collection Job

OPERATING INSTRUCTIONS

Copyright

© Ericsson AB 2016, 2017. 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	Description	1
2	Procedure	1
2.1	Start Measurement Collection Job	1



Start Measurement Collection Job



1 Description

This instruction describes how to start a measurement collection job.

A measurement collection job is started to enable collection of performance data for the corresponding measurement types. This is usually done when the managed element is operational and manageable as part of the normal operations.

2 Procedure

2.1 Start Measurement Collection Job

Prerequisites

- No documents are required.
- No tools are required.
- The following conditions must apply:
 - The job name is known.
In this instruction, the job name example is `POT_15min_Job`.
 - `currentJobState` for the job is `STOPPED`.
 - An Ericsson Command-Line Interface (ECLI) session in Exec mode is in progress.

Steps

1. Navigate to the *PmJob* managed object, for example:

```
>dn ManagedElement=NODE06ST, SystemFunctions=1, Pm=1, PmJob=POT_15min_Job
```

2. Enter Config mode:

```
(PmJob=POT_15min_Job) >configure
```

3. Set the `PmJob` state to `ACTIVE`:



```
(config-PmJob=POT_15min_Job) > requestedJobState=ACTIVE
```

4. Commit the PmJob configuration:

```
(config-PmJob=POT_15min_Job) > commit
```

5. Verify that currentJobState is changed to ACTIVE:

```
(PmJob=POT_15min_Job) > show -v
```

The following is an example output:

```
PmJob=POT_15min_Job
compressionType=[] <empty>
currentJobState=ACTIVE <read-only>
granularityPeriod=FIFTEEN_MIN <default>
jobControl=FULL <default> <read-only>
jobGroup=[] <empty>
jobPriority=MEDIUM <default>
jobType=MEASUREMENTJOB <default>
pmJobId="POT_15min_Job"
[...]
requestedJobState=ACTIVE <default>
MeasurementReader=POT_mr
```

6. Wait up to 5 minutes after the end of the granularity period.
7. Navigate to the *FileGroup* managed object where all performance measurement files are exposed, for example:

```
(PmJob=POT_15min_Job) > dn ManagedElement=NODE06ST, System
Functions=1, FileM=1, LogicalFs=1, FileGroup=PerformanceMa
nagementReportFiles
```

8. Verify that the *FileGroup* managed object corresponding to the created performance measurement report file exists:

```
(FileGroup=PerformanceManagementReportFiles) > show -v
```

The following is an example output:

```
FileGroup=PerformanceManagementReportFiles
internalHousekeeping=false
files="A20141124.1045+0100-1100+0100_NODE06ST.xml"
  fileName="A20141124.1045+0100-1100+0200_NODE06ST.xml"
  fileType="xml"
  modificationTime="2014-11-24T01:26:00"
  size=1388
[...]
```



Note: This output means that file `A20141124.1045+0100-1100+0100_NODE06ST.xml` exists in the file system. The + sign is rendered as ?? by the ECLI.

The file can contain additional performance measurement data corresponding to other jobs with the same granularity period.