

COM Management Guide

Common Operation and Maintenance

USER GUIDE

Copyright

© Ericsson AB 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

1	Introduction	1
2	Fault Management	3
2.1	Basic Concepts	3
2.2	Act on Received Notification	6
2.3	Configure SNMP Master Agent	6
2.4	Configure SNMP Master Agent for DTLS	7
2.5	Create SNMPv1 Target	7
2.6	Create SNMPv2C Target	7
2.7	Create SNMPv3 Target	8
2.8	Create SNMPv3 Target for DTLS	8
2.9	Create SNMPv1 View	8
2.10	Create SNMPv2C View	9
2.11	Create SNMPv3 View	9
2.12	Ericsson Alarm MIB	9
2.13	Alarm Information Mapping	10
3	Configuration Management	25
3.1	Basic Concepts	25
3.2	Managed Object Models	26
3.3	Configuration Management Using NETCONF	27
3.4	Configuration Management Using CLI	27
3.5	Management of CM Interfaces	28
3.6	Model File Access	30
4	Security Management	31
4.1	Overview	31
4.2	Security Management Functions	36
4.3	Roles and Rules	37
4.4	Procedures	39
4.5	Cipher Configuration	40
4.6	Synchronization	41
5	File Management	43



6	Performance Management	45
6.1	Basic Concepts	45
6.2	Measurement Job Administration	46
6.3	Storage of Performance Measurement Results	47
6.4	ECIM PM Model Compliance	49



1 Introduction

This document describes how to use the Fault Management (FM), Configuration Management (CM), Security Management (SM), File Management (FileM), and Performance Management (PM) in the Common Operation and Maintenance (COM) product.





2 Fault Management

The COM FM allows the operator to detect faults and malfunctions on the system. Based on the information carried in the notifications, the operator can locate the source of the event and the relevant documentation. Based on this information action can be taken to resolve or prevent failures.

2.1 Basic Concepts

Within the COM FM, the following terms are used:

Active alarm	An alarm that has an alarm state that has been raised but not cleared.
Active Alarm List	The Active Alarm List (AAL) contains all the active alarms in a system using COM and shows the current operational state of the system. The AAL is stored by the persistent storage provided by the Middleware (MW). The operator can, at any time, retrieve all active alarms through the COM Northbound Interface (NBI) using the Command-Line Interface (CLI), Network Configuration (NETCONF), or SNMP.
Alarm	A persistent indication of the fault that has a life cycle or state. An alarm has at least the state raise (initial detection of the fault) or clear (the detection that the fault no longer exists). An alarm can also change state (update) regarding perceived severity. Alarms are also called stateful alarms to emphasize that they have a state.
Alert	A discrete alarm that has no state, an alarm without an associated clear. Alerts are also called stateless alarms to emphasize that they do not have a state.
Heartbeats	The COM FM sends on regular intervals traps to the Management System. A heartbeat trap contains the last sequence numbers and the last event time used for the alarms and alerts. These messages can be used to detect loss of notifications sent as SNMP traps from the COM FM to the Management System.
Notification	A message that can carry an alarm or alert instance.
Trap	An unacknowledged SNMP message that carries a notification or heartbeat.



2.1.1 Logs

The COM FM logs all alarm state changes and all alerts using the Log Service Provider Interface (SPI). Each alarm and alert log record is encoded in XML format.

It is the responsibility of the Support Agent (SA) to map the XML log records on physical files.

The XML log record consists of two elements. The first element indicates the time the record is logged. The second element contains the specific information about the alarm or alert and it is formatted as a string separated by semicolons. The format is explained in Table 1.

For a detailed description of each field, refer to class `FmAlarmin Managed Object Model com_fm`.

Table 1 Alarm and Alert Log Record Format

Tags and Information	Description
<FmLogRecord>	Log record start
<LogTimestamp>	
Time stamp tag	Formatted as YYYY-MM-DDThh:mm:ssZ indicates the time the record is logged
</LogTimestamp>	
<Alarm> or <Alert>	
Alarm or alert specific information	Formatted as a string separated by semicolons containing the following tokens: 1: stateful 2: eventTime 3: source 4: majorType 5: minorType 6: specificProblem 7: probableCause 8: severity 9: additionalText 10: sequenceNumber 11: eventType 12: originalEventTime 13: originalSeverity 14: originalAdditionalText 15: originalSequenceNumber 16: additionalInfoSize 17: additionalInfo (name;value)



Tags and Information	Description
</Alarm> or </Alert>	
</FmLogRecord>	Log record end

Examples of log records in an alarm log are shown in Example 1.

```
<FmLogRecord>
  <LogTimestamp>2014-03-27T10:53:55+01:00</LogTimestamp>
  <Alarm>
1;2014-03-27T10:53:55+01:00;ManagedElement=1,Equipment=1,PluginUnit=13;123;
429876;HW Fault;418;CRITICAL;Overheated;3;EQUIPMENTALARM;2014-03-
27T10:53:49+01:00;MAJOR;Overheated;1;2;core1 temp;87;core2
temp;93</Alarm>
</FmLogRecord>

<FmLogRecord>
  <LogTimestamp>2014-03-27T10:53:55+01:00</LogTimestamp>
  <Alarm>
1;2014-03-27T10:53:55+01:00;ManagedElement=1,Transport=1,Atm=1;421;792134;
Link Overload;613;MAJOR;;5;COMMUNICATIONSALARM;2014-03-
27T10:53:49+01:00;MINOR;Link Overload;2;1;overload;130
  </Alarm>
</FmLogRecord>

<FmLogRecord>
  <LogTimestamp>2014-03-27T10:53:55+01:00</LogTimestamp>
  <Alarm>
1;2014-03-27T10:53:55+01:00;ManagedElement=1,Transport=1,Atm=1;421;792134;
Link overload;613;CLEARED;;5;COMMUNICATIONSALARM;2014-03-
27T10:53:49+01:00;MINOR;Link Overload;2;0
  </Alarm>
</FmLogRecord>
```

Example 1 Log Records in Alarm Log

If COM has been configured to use a network Managed Element (ME) identity, this is used in the source Distinguished Names (DNs) in log records as shown in Example 2. For more information about network ME identity, see Section 3.1.2 Configuration on page 25.

```
<FmLogRecord>
  <LogTimestamp>2014-03-27T10:53:55+01:00</LogTimestamp>
  <Alarm>
1;2014-03-27T10:53:55+01:00;ManagedElement=Stockholm,Equipment=1,Plug=13;
123;429876;HW Fault;418;CRITICAL;Overheated;3;EQUIPMENTALARM;2014-03-
27T10:53:49+01:00;MAJOR;Overheated;1;2;core1 temp;87;core2 temp;93
</Alarm>
</FmLogRecord>
```

Example 2 Log Record in Alarm Log when Using Management Element Identity



2.1.2 Configuration

The COM FM can be configured through the COM NBI using the CLI or NETCONF. The following can be configured:

- The SNMP Master Agent address (host and port), see Section 2.3 Configure SNMP Master Agent on page 6.
- Different SNMP targets for v1, v2c, and v3, see Section 2.7 Create SNMPv3 Target on page 8.
- Different SNMP views for v1, v2c, and v3, see Section 2.11 Create SNMPv3 View on page 9.
- The heartbeat interval, specifying the frequency of the heartbeat traps.

For a detailed description of the parameters, refer to *Managed Object Model com_fm* and *Managed Object Model com_snmp*.

2.2 Act on Received Notification

Each notification received at the Management System has a corresponding alarm Operating Instruction (OPI). The alarm OPI contains information about the notification and the steps needed to eliminate or prevent failures.

The prerequisites for this procedure are as follows:

- All alarm OPIs must be available.
- The *ERICSSON ALARM MIB* document must be available.

To act on a received notification:

1. Find the alarm OPI, whose title is equal to the value of either the `eriAlarmActiveSpecificProblem` or `eriAlarmAlertSpecificProblem` attribute of the received notification.
2. Follow the steps in the alarm OPI.

2.3 Configure SNMP Master Agent

The SNMP Master Agent is configured by changing attributes `agentAddress` in the SNMP Managed Object Model (MOM). A CLI example to set the host and port attributes is shown in Example 3.

```
>configure
(config)>ManagedElement=1,SystemFunctions=1,SysM=1,Snmp=1,agentAddress
(config-Snmp=1-agentAddress)>host=127.0.0.1
(config-Snmp=1-agentAddress)>port=161
(config-Snmp=1-agentAddress)>up
(config-Snmp=1)>commit
```

Example 3 Configuring SNMP Master Agent



2.4 Configure SNMP Master Agent for DTLS

The SNMP Master Agent is configured for DTLS by changing attributes `agentAddressDtls`, `nodeCredential`, and `trustCategory` in the SNMP MOM. Attributes `nodeCredential` and `trustCategory` give information about certificates. If those attributes are not configured or invalid, SNMP uses the configuration file to get information about the certificates.

A CLI example to configure the Master Agent for DTLS is shown in Example 4.

```
>configure
(config)> ManagedElement=1, SystemFunctions=1, SysM=1, Snmp=1, agentAddressDtls
(config-Snmp=1-agentAddressDtls)>host=0.0.0.0
(config-Snmp=1-agentAddressDtls)>port=10161
(config-Snmp=1-agentAddressDtls)>up
(config-Snmp=1)>nodeCredential=
ManagedElement=1, SystemFunctions=1, SecM=1, CertM=1, NodeCredential=nc1
(config-Snmp=1)>trustCategory=
ManagedElement=1, SystemFunctions=1, SecM=1, CertM=1, TrustCategory=tc1
(config-Snmp=1)>commit
```

Example 4 Configuring Master Agent for DTLS

2.5 Create SNMPv1 Target

To create an SNMPv1 target, the address, port, community, and `administrativeState` attributes must be set. A CLI example to create an SNMPv1 target is shown in Example 5.

```
>configure
(config)>ManagedElement=1, SystemFunctions=1, SysM=1, Snmp=1, SnmpTargetV1=1
(config-SnmpTargetV1=1)>address="127.0.0.1"
(config-SnmpTargetV1=1)>port=1162
(config-SnmpTargetV1=1)>community="com"
(config-SnmpTargetV1=1)>administrativeState=UNLOCKED
(config-SnmpTargetV1=1)>commit
```

Example 5 Creating SNMPv1 Target

2.6 Create SNMPv2C Target

To create an SNMPv2C target, the address, port, community, and `administrativeState` attributes must be set. A CLI example to create an SNMPv2C target is shown in Example 6.

```
>configure
(config)>ManagedElement=1, SystemFunctions=1, SysM=1, Snmp=1, SnmpTargetV2C=1
(config-SnmpTargetV2C=1)>address="127.0.0.1"
(config-SnmpTargetV2C=1)>port=1162
(config-SnmpTargetV2C=1)>community="com"
(config-SnmpTargetV2C=1)>administrativeState=UNLOCKED
(config-SnmpTargetV2C=1)>commit
```

Example 6 Creating SNMPv2C Target



2.7 Create SNMPv3 Target

To create an SNMPv3 target, the `snmpSecurityLevel`, `privProtocol`, `authProtocol`, `user`, `address`, `port`, `privKey`, and `authKey` attributes must be set. A CLI example to create an SNMPv3 target is shown in Example 7.

```
>configure
(config)>ManagedElement=1, SystemFunctions=1, SysM=1, Snmp=1, SnmpTargetV3=1
(config-SnmpTargetV3=1)>snmpSecurityLevel=AUTH_PRIV
(config-SnmpTargetV3=1)>privProtocol=DES
(config-SnmpTargetV3=1)>authProtocol=MD5
(config-SnmpTargetV3=1)>user="user1"
(config-SnmpTargetV3=1)>address="127.0.0.1"
(config-SnmpTargetV3=1)>port=162
(config-SnmpTargetV3=1)>privKey="myprivatekey" cleartext
(config-SnmpTargetV3=1)>authKey="myauthorizationkey" cleartext
(config-SnmpTargetV3=1)>commit
```

Example 7 Creating SNMPv3 Target

2.8 Create SNMPv3 Target for DTLS

To create a SNMPv3 target for DTLS, the attributes `user`, `address`, and `port` must be set.

The `user` attribute must match the `rfc822-name`, DNS address, or IP address in the `subjectAltName` portion of the certificate presented by the connecting client. The `CommonName` of the certificate is not used when considering a match, as this is deprecated in the SNMP-TLS-TM MIB.

A CLI example to create an SNMPv3 target is shown in Example 8.

```
>configure
(config)>ManagedElement=1, SystemFunctions=1, SysM=1, Snmp=1, SnmpTargetV3Dtls=1
(config-SnmpTargetV3Dtls=1)>user="user@host.com"
(config-SnmpTargetV3Dtls=1)>address="127.0.0.1"
(config-SnmpTargetV3Dtls=1)>port=10162
(config-SnmpTargetV3Dtls=1)>up
(config-Snmp=1)>commit
```

Example 8 Creating SNMP DTLS Target

2.9 Create SNMPv1 View

To create an SNMPv1 view, the `community`, `readOids`, and `writeOids` attributes must be set. A CLI example to create an SNMPv1 view is shown in Example 9.

```
>configure
(config)>ManagedElement=1, SystemFunctions=1, SysM=1, Snmp=1, SnmpViewV1=1
(config-SnmpViewV1=1)>community="comuser"
(config-SnmpViewV1=1)>readOids="1.3.6.1.4.1.193.183.4.1.5.1"
(config-SnmpViewV1=1)>writeOids="1.3.6.1.4.1.193.183.4.1.5.1"
(config-SnmpViewV1=1)>commit
```

Example 9 Creating SNMPv1 View



2.10 Create SNMPv2C View

To create an SNMPv2C view, the `community`, `readOids`, and `writeOids` attributes must be set. A CLI example to create an SNMPv2C view is shown in Example 10.

```
>configure
(config)>ManagedElement=1,SystemFunctions=1,SysM=1,Snmp=1,SnmpViewV2C=1
(config-SnmpViewV2C=1)>community="comuser"
(config-SnmpViewV2C=1)>readOids="1.3.6.1.4.1.193.183.4.1.5.1"
(config-SnmpViewV2C=1)>writeOids="1.3.6.1.4.1.193.183.4.1.5.1"
(config-SnmpViewV2C=1)>commit
```

Example 10 Creating SNMPv2C View

2.11 Create SNMPv3 View

To create an SNMPv3 view, the `user`, `readOids`, and `writeOids` attributes must be set. A CLI example to create an SNMPv3 view is shown in Example 11.

```
>configure
(config)>ManagedElement=1,SystemFunctions=1,SysM=1,Snmp=1,SnmpViewV3=1
(config-SnmpViewV3=1)>user="user1"
(config-SnmpViewV3=1)>readOids="1.3.6.1.4.1.193.183.4.1.5.1"
(config-SnmpViewV3=1)>writeOids="1.3.6.1.4.1.193.183.4.1.5.1"
(config-SnmpViewV3=1)>commit
```

Example 11 Creating SNMPv3 View

2.12 Ericsson Alarm MIB

The COM FM is compliant with the Ericsson Alarm Management Information Base (MIB) with the following exemptions:

- The alert table, `eriAlarmAlertTable`, is always empty.
- `eriAlarmActiveTableURL` always returns an empty string.
- `eriAlarmAlertTableURL` always returns an empty string.

For more information about the Ericsson Alarm MIB, refer to *ERICSSON ALARM MIB*.

2.12.1 AlarmListRebuilt

To comply with the Ericsson Alarm MIB, the COM FM sends an `AlarmListRebuilt` notification after a COM restart and after deactivation of alarm suppression. It is an indication to the manager to perform an alarm resynchronization procedure.

For more information about the notification, refer to *COM, AlarmListRebuilt*.



2.12.2 Unknown Alarm

If an application raises an alarm or alert not described by any `FmAlarmType` Managed Object (MO) instance, an FM `Unknown Alarm` alert is sent out reusing the major and minor numbers received from the application.

2.12.3 Alarm Filters

Within the telecom domain, a common problem is that too many alarms are generated. To minimize the number of notifications sent, COM FM offers two different types of alarm filters: transient alarm filter and alarm toggling filter. These filters are by default turned off but can be configured and activated.

2.12.3.1 Alarm Toggling

An alarm is defined as toggling when the same (identical) alarm has been raised and cleared, multiple times, within a defined time.

When the toggling alarm filter is used and an alarm is toggling, alarm attribute `additionalText` is appended with the text “The alarm is toggling”. The appended text is kept until the alarm is cleared.

2.12.3.2 Transient Alarm

Short-term and low priority alarms are considered transient. A high number of transient alarms cause alarm pollution. If the transient alarm filter is used, alarms considered transient are not raised. For more information about alarm filters, refer to *COM Application Developer's Guide*.

2.12.4 Alarm Suppression

The main use cases for alarm suppression are planned maintenance tasks where the alarm client risks being flooded with non-important alarms. One example is when a technician is present on-site handling the equipment. All alarms are still available in the AAL but no alarm notifications are sent.

However, alarms with critical severity, for example, fire alarms, are sent regardless. Active critical alarms that are cleared or assigned a lower severity also generate a notification to indicate that the critical problem is resolved.

For more information on how to turn alarm suppression on/off, refer to *COM API and SPI Programmer's Guide*.

2.13 Alarm Information Mapping

This section describes the alarm information mapping.



2.13.1 Alarm Information in Multiple Interfaces

Alarm information is available in the SNMP, CLI, and NETCONF interfaces and in log files, as described in this section.

The heartbeat interval of the COM SNMP alarm interface is configurable as follows:

- SNMP object `eriAlarmHbInterval` of the Ericsson Alarm MIB.
- Attribute `heartbeatInterval` of MO `Fm` in the COM CLI and NETCONF.

A summary of the number of active alarms is available as follows:

- `eriAlarmActiveAlarms`, `eriAlarmSumCritical`, `eriAlarmSumMajor`, `eriAlarmSumMinor`, and `eriAlarmSumWarning` SNMP objects of the Ericsson Alarm MIB.

Note: The number of alarms of indeterminate severity is available in the SNMP as `eriAlarmSumIndeterminate` only.

- `totalActive`, `sumCritical`, `sumMajor`, `sumMinor`, and attributes `sumWarning` of MO `Fm` in the COM CLI and NETCONF.

Note: While reading the FM subtree in a particular transaction, the statistics attributes in FM (that is `totalActive`, `sumMajor`, `sumMinor` and so on) cannot always be in sync with the actual `FmAlarm` instances.

Information on the last issued alarm is available as follows:

- `eriAlarmActiveLastSequenceNo` and `eriAlarmActiveLastChanged` SNMP objects of the Ericsson Alarm MIB
- `lastChanged` and attributes `lastSequenceNo` of MO `Fm` in the COM CLI and NETCONF.

Alarms instances are available in multiple interfaces, as summarized in Table 2.

Table 2 Representations of Alarms

Alarm and Alert Log	FmAlarm MO in COM_FM	SNMP Object	SNMP Notification
<code>eventTime</code>	<code>lastEventTime</code>	<code>eriAlarmActiveEventTime</code> , <code>eriAlarmActiveOriginalEventTime</code>	<code>eriAlarmActiveEventTime</code>
<code>source</code>	<code>source</code>	<code>eriAlarmActiveManagedObject</code>	<code>eriAlarmActiveManagedObject</code>



Table 2 Representations of Alarms

Alarm and Alert Log	FmAlarm MO in COM_FM	SNMP Object	SNMP Notification
majorType	majorType	eriAlarmActiveMajorType	eriAlarmActiveMajorType
minorType	minorType	eriAlarmActiveMinorType	eriAlarmActiveMinorType
specificProblem	specificProblem	eriAlarmActiveSpecificProblem	eriAlarmActiveSpecificProblem
probableCause	probableCause	eriAlarmActiveProbableCause	eriAlarmActiveProbableCause
severity	activeSeverity	eriAlarmActiveSeverity, eriAlarmActiveOriginalSeverity	The notification type indicates severity information. ⁽¹⁾
additionalText	additionalText	eriAlarmActiveAdditionalText, eriAlarmActiveOrigAdditionalText	eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText
sequenceNumber	sequenceNumber	eriAlarmActiveLastSequenceNo	eriAlarmActiveLastSequenceNo
eventType	eventType	eriAlarmActiveEventType	eriAlarmActiveEventType
Not applicable	additionalInfo	Not applicable	Not applicable
Not applicable	fmAlarmId	Not applicable	Not applicable
Not applicable	Not applicable	eriAlarmActiveResourceId	eriAlarmNObjResourceId

(1) The notification types are *eriAlarmCritical*, *eriAlarmMajor*, *eriAlarmMinor*, *eriAlarmWarning*, *eriAlarmIndeterminate*, and *eriAlarmCleared*.

2.13.2 SNMP Objects

Properties of the supported SNMP objects are summarized in Table 3. The deviations are indicated.

Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
ericssonAlarmMIB OID: .1.3.6.1.4.1.193.183.4	



Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmObjects OID: .1.3.6.1.4.1.193.183.4.1	
eriAlarmSummary Type: Gauge32 OID: .1.3.6.1.4.1.193.183.4.1.1 Max Access: read-only	
eriAlarmSumIndeterminate Type: Gauge32 OID: .1.3.6.1.4.1.193.183.4.1.1.1 Max Access: read-only	This object shows the number of currently active alarms with perceived severity <i>indeterminate</i> . Only stateful alarms are included.
eriAlarmSumCritical Type: Gauge32 OID: .1.3.6.1.4.1.193.183.4.1.1.2 Max Access: read-only	This object shows the number of currently active alarms with perceived severity <i>critical</i> . Only stateful alarms are included.
eriAlarmSumMajor Type: Gauge32 OID: .1.3.6.1.4.1.193.183.4.1.1.3 Max Access: read-only	This object shows the number of currently active alarms with perceived severity <i>major</i> . Only stateful alarms are included.
eriAlarmSumMinor Type: Gauge32 OID: .1.3.6.1.4.1.193.183.4.1.1.4 Max Access: read-only	This object shows the number of currently active alarms with perceived severity <i>minor</i> . Only stateful alarms are included.
eriAlarmSumWarning Type: Gauge32 OID: .1.3.6.1.4.1.193.183.4.1.1.5 Max Access: read-only	This object shows the number of currently active alarms with perceived severity <i>warning</i> . Only stateful alarms are included.
eriAlarmActiveAlarms OID: .1.3.6.1.4.1.193.183.4.1.3	
eriAlarmActiveNumber Type: Unsigned32 OID: .1.3.6.1.4.1.193.183.4.1.3.1 Max Access: read-only	This object shows the total number of currently active alarms, that is, the total number of entries in the alarm table.
eriAlarmActiveLastChanged Type: DateAndTime OID: .1.3.6.1.4.1.193.183.4.1.3.2 Max Access: read-only	A time stamp when the active alarm table was last changed. The value can be used by a manager to initiate an alarm resynchronization procedure. All fields of <i>DateAndTime</i> must be filled out, including the hours and minutes from UTC. As such, the value must be 11 octets long. Reference: <i>DateAndTime</i> is defined in RFC 2579, page 18.

Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmActiveLastSequenceNumber Type: EriAlarmSequenceNumber OID: .1.3.6.1.4.1.193.183.4.1.3.3 Max Access: read-only	The last used sequence number for an alarm state change notification. A management system can poll this variable to detect lost alarm change notifications.
eriAlarmActiveTableURL Type: SnmpAdminString OID: .1.3.6.1.4.1.193.183.4.1.3.4	A URL, in accordance with RFC 4248, pointing to a location where the contents of the alarm table can be retrieved as a file. The actual format of the file is out of the scope of this MIB and is found in the system documentation. A string of length zero means that the agent does not have a URL to provide. Deviation: This object always contains an empty string.
eriAlarmActiveMajorType Type: EriAlarmType OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.2 Max Access: read-only	In combination with <code>eriAlarmActiveMinorType</code> , this provides a unique identification of the fault type. Different MO types and instances can share alarm types, but if the same MO reports the same alarm type, it is to be considered as the same alarm state. The alarm type is a simplification of the different X.733 and 3GPP® alarm IRP alarm correlation mechanisms based on <code>EventType</code> , <code>ProbableCause</code> , <code>SpecificProblem</code> , and <code>NotificationId</code> . In systems where <code>eriAlarmActiveMajorType</code> is not needed for identification purposes, it is not used and must be zero.
eriAlarmActiveMinorType Type: EriAlarmType OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.3 Max Access: read-only	In combination with <code>eriAlarmActiveMajorType</code> , this provides a unique identification of the fault type, not including the MO. Different MO types and instances can share alarm types, but if the same MO reports the same alarm type it is to be considered as the same alarm state. The alarm type is a simplification of the different X.733 and 3GPP alarm IRP alarm correlation mechanisms based on <code>EventType</code> , <code>ProbableCause</code> , <code>SpecificProblem</code> , and <code>NotificationId</code> .
eriAlarmActiveSpecificProblem Type: EriAlarmSpecificProblem OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.4 Max Access: read-only	This is a clear-text unique identification of the alarm type. There is a one-to-one mapping between <code>eriAlarmActiveSpecificProblem</code> and <code>[eriAlarmActiveMajorType, eriAlarmActiveMinorType]</code> .



Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmActiveManagedObject Type: EriMO OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.5 Max Access: read-only	<p>The 3GPP naming convention must be used as format for the MO parameter. The granularity must be good enough to guarantee unique alarm states and relevant resource identification to the operator.</p> <p>The DN must be “relative” to the nodes “own” root.</p> <p>Reference: 3GPP TS 32.106-8 V3.2, Name convention for Managed Objects.</p>
eriAlarmActiveEventType Type: IANAItuEventType OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.6 Max Access: read-only	<p>The event type as defined in X.733/X.736.</p> <p>Reference: ITU Recommendation X.733, “Information Technology – Open Systems Interconnection – System Management: Alarm Reporting Function”, 1992.</p>
eriAlarmActiveEventTime Type: DateAndTime OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.7 Max Access: read-only	<p>A time stamp of the alarm state change event. This variable represents the last change of the alarm state, such as changed severity or additional text. If the alarm has not changed state, this variable represents the alarm raise time and is the same as <code>originalEventTime</code>.</p> <p>All fields of <code>DateAndTime</code> must be filled out, including the hours and minutes from UTC. As such, the value must be 11 octets long.</p> <p>Reference: <code>DateAndTime</code> is defined in RFC 2579, page 18. ITU Recommendation X.733, “Information Technology – Open Systems Interconnection – System Management: Alarm Reporting Function”, 1992.</p>
eriAlarmActiveOriginalEventTime Type: DateAndTime OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.8 Max Access: read-only	<p>The time stamp of the original alarm raises notification.</p> <p>All fields of <code>DateAndTime</code> must be filled out, including the hours and minutes from UTC. As such, the value must be 11 octets long.</p> <p>Reference: <code>DateAndTime</code> is defined in RFC 2579, page 18.</p>
eriAlarmActiveProbableCause Type: EriProbableCause OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.9 Max Access: read-only	<p>The probable cause for the alarm originally defined by X.733 and subsequent standards. Refer also to the ERICSSON ALARM PC MIB. Because of the history of problems in maintaining a standardized probable cause, the probable cause is not unique. A best effort mapping of the alarm to existing probable causes is used.</p> <p>Reference: ITU Recommendation X.733, “Information Technology – Open Systems Interconnection – System Management: Alarm Reporting Function”, 1992.</p>



Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmActiveSeverity Type: ItuPerceivedSeverity OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.10 Max Access: read-only	The severity of the alarm as defined by X.733. This cannot be the original severity since the alarm has changed severity. Reference: ITU Recommendation X.733, "Information Technology – Open Systems Interconnection – System Management: Alarm Reporting Function", 1992.
eriAlarmActiveOriginalSeverity Type: ItuPerceivedSeverity OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.11 Max Access: read-only	The original severity as reported by the alarm raise notification.
eriAlarmActiveAdditionalText Type: EriLargeAdditionalText OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.12 Max Access: read-only	A user-friendly text describing the alarm. The text is both static, depending on the alarm type (probable cause), and dynamic depending on the MO instance and other conditions. This string is longer than the corresponding varbind in the notification to manage large strings. Reference: ITU Recommendation X.733, "Information Technology – Open Systems Interconnection – System Management: Alarm Reporting Function", 1992.
eriAlarmActiveOrigAdditionalText Type: EriLargeAdditionalText OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.13 Max Access: read-only	A user-friendly text describing the alarm. The text is both static, depending on the alarm type (probable cause), and dynamic depending on the MO and other conditions. This is the original text as reported by the first alarm notification, after which it is not altered. Reference: ITU Recommendation X.733, "Information Technology – Open Systems Interconnection – System Management: Alarm Reporting Function", 1992.
eriAlarmActiveResourceId OID: .1.3.6.1.4.1.193.183.4.1.3.5.1.14 Max Access: read-only	If the alarm refers to an object which is instrumented by the SNMP this variable indicates the corresponding OID for the MO. Deviation: Not supported. The value is null OID (0.0).
eriAlarmAlerts OID: .1.3.6.1.4.1.193.183.4.1.4	
eriAlarmAlertNumber Type: Unsigned32 OID: .1.3.6.1.4.1.193.183.4.1.4.1 Max Access: read-only	The number of rows in the alert table.



Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmAlertLastChanged Type: DateAndTime OID: .1.3.6.1.4.1.193.183.4.1.4.2 Max Access: read-only	<p>A time stamp when the alert table was last changed. Can be used by a manager to initiate an update of alerts procedure.</p> <p>All fields of DateAndTime must be filled out, including the hours and minutes from UTC. As such, the value must be 11 octets long.</p> <p>Reference: DateAndTime is defined in RFC 2579, page 18.</p>
eriAlarmAlertLastSequenceNo Type: EriAlarmSequenceNumber OID: .1.3.6.1.4.1.193.183.4.1.4.3 Max Access: read-only	<p>The last sequence number used in alert notifications. This can be used to detect a lost notification.</p>
eriAlarmAlertTableURL OID: .1.3.6.1.4.1.193.183.4.1.4.4 Max Access: read-only	<p>A URL, in accordance with RFC 4248, pointing to a location where the contents of alert table can be retrieved as a file. The format of the file is out of the scope of this MIB and system-dependant.</p> <p>Deviation: This object always contains an empty string.</p>
eriAlarmAlertTable OID: .1.3.6.1.4.1.193.183.4.1.4.5	<p>Deviation: The alert table is always empty.</p>
eriAlarmHeartBeat OID: .1.3.6.1.4.1.193.183.4.1.5 Max Access: read-only	
eriAlarmHbInterval Type: Unsigned32 OID: .1.3.6.1.4.1.193.183.4.1.5.1 Max Access: read-write	<p>Notification eriAlarmHeartBeatNotif is sent every eriAlarmHbInterval. Managers can subscribe to the notification using the SNMP framework MIBs by using the snmpNotifyName "heartbeat" (SNMP NOTIFICATION MIB, snmpNotifyTable).</p>
eriAlarmNObjAdditionalText Type: EriAdditionalText OID: .1.3.6.1.4.1.193.183.4.1.2.1 Max Access: accessible-for-notify	<p>This is a scalar variable, only used in notifications. It is size-ranged with a smaller size than the corresponding type used in the active alarm table. This is to ensure that all alarm information fits into one UDP packet. The additional text can be further appended by using the dedicated append notification.</p>

Table 3 Alarm Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmNObjMoreAdditionalText Type: TruthValue OID: .1.3.6.1.4.1.193.183.4.1.2.2 Max Access: accessible-for-notify	<p>This is a scalar variable, only used in notifications. It informs the management system that the additional text is appended by subsequent append additional text notifications.</p> <p>This object exists simply because of the limitations of the PDUS size inherent in the SNMP over UDP, which limits the size of the PDU+headers to under 1500 octets. Otherwise, the notification becomes fragmented.</p>
eriAlarmNObjResourceId Type: TruthValue OID: .1.3.6.1.4.1.193.183.4.1.2.3 Max Access: accessible-for-notify	<p>This is a scalar variable only, used in notifications. It tells the management system that there is an SNMP-based resource ID (OID) that identifies the alarming resource.</p> <p>This object exists simply because of the limitations of the PDU size inherent in the SNMP over UDP, which limits the size of the PDU+headers to under 1500 octets. Otherwise, the notification becomes fragmented.</p> <p>Deviation: Not supported. Always equals 0.</p>

2.13.3 SNMP Notifications

Properties of the supported SNMP notifications of the Ericsson Alarm MIB are summarized in Table 4. The deviations are indicated.



Table 4 Notification Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmIndeterminate Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmActiveSpecificProblem, eriAlarmActiveLastSequenceNo, eriAlarmActiveEventType, eriAlarmActiveEventTime, eriAlarmActiveProbableCause, eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText, eriAlarmNObjResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.1	<p>This notification is sent when a resource detects a new alarm state with severity <code>indeterminate</code>. The notification is also used to change the severity or additional text of an alarm, or both. The combination of <code>ManagedObject</code> and <code>MajorType/MinorType</code> is always unique and can be used by management systems to correlate alarm, alarm change, and alarm clear. A corresponding row is created in the alarm table. The sequence number increases for every notification and can be used to detect lost notifications.</p> <p>A management system is to be prepared for appending text to additional text, indicated by the <code>eriAlarmNObjMoreAdditionalText</code> varbind, and sent with <code>eriAlarmAppendInfo</code>.</p> <p>Do not confuse this with a change of additional text.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to "true" and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p>
eriAlarmWarning Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmActiveSpecificProblem, eriAlarmActiveLastSequenceNo, eriAlarmActiveEventType, eriAlarmActiveEventTime, eriAlarmActiveProbableCause, eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText, eriAlarmNObjResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.2	<p>This notification is sent when a resource detects a new alarm state with severity <code>warning</code>. The notification is also used to change the severity or additional text of an alarm, or both. The combination of <code>ManagedObject</code> and <code>MajorType/MinorType</code> is always unique and can be used by management systems to correlate alarm, alarm change, and alarm clear. A corresponding row is created in the alarm table (<code>eriAlarmActiveAlarmTable</code>). The sequence number increases for every notification and can be used to detect lost notifications.</p> <p>A management system is to be prepared for appending text to additional text, indicated by the <code>eriAlarmNObjMoreAdditionalText</code> varbind, and sent with <code>eriAlarmAppendInfo</code>.</p> <p>Do not confuse this with a change of additional text.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to "true" and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p>



Table 4 Notification Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmMinor Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmActiveSpecificProblem, eriAlarmActiveLastSequenceNo, eriAlarmActiveEventType, eriAlarmActiveEventTime, eriAlarmActiveProbableCause, eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText, eriAlarmNObjResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.3	<p>This notification is sent when a resource detects a new alarm state with severity <code>minor</code>. The notification is also used to change the severity or additional text of an alarm, or both. The combination of <code>ManagedObject</code> and <code>MajorType/MinorType</code> is always unique and can be used by management systems to correlate alarm, alarm change, and alarm clear. A corresponding row is created in the alarm table (<code>eriAlarmActiveAlarmTable</code>). The sequence number increases for every notification and can be used to detect lost notifications.</p> <p>A management system is to be prepared for appending text to additional text, indicated by the <code>eriAlarmNObjMoreAdditionalText</code> varbind, and sent with <code>eriAlarmAppendInfo</code>.</p> <p>Do not confuse this with a change of additional text.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to “true” and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p>
eriAlarmMajor Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmActiveSpecificProblem, eriAlarmActiveLastSequenceNo, eriAlarmActiveEventType, eriAlarmActiveEventTime, eriAlarmActiveProbableCause, eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText, eriAlarmNObjResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.4	<p>This notification is sent when a resource detects a new alarm state with severity <code>major</code>. The notification is also used to change the severity or additional text of an alarm, or both. The combination of <code>ManagedObject</code> and <code>MajorType/MinorType</code> is always unique and can be used by management systems to correlate alarm, alarm change, and alarm clear. A corresponding row is created in the alarm table (<code>eriAlarmActiveAlarmTable</code>). The sequence number increases for every notification and can be used to detect lost notifications.</p> <p>A management system is to be prepared for appending text to additional text, indicated by the <code>eriAlarmNObjMoreAdditionalText</code> varbind, and sent with <code>eriAlarmAppendInfo</code>.</p> <p>Do not confuse this with a change of additional text.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to “true” and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p>



Table 4 Notification Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmCritical Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmActiveSpecificProblem, eriAlarmActiveLastSequenceNo, eriAlarmActiveEventType, eriAlarmActiveEventTime, eriAlarmActiveProbableCause, eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText, eriAlarmNObjResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.5	<p>This notification is sent when a resource detects a new alarm state with severity <code>critical</code>. The notification is also used to change the severity or additional text of an alarm, or both. The combination of <code>ManagedObject</code> and <code>MajorType/MinorType</code> is always unique and can be used by management systems to correlate alarm, alarm change, and alarm clear. A corresponding row is created in the alarm table (<code>eriAlarmActiveAlarmTable</code>). The sequence number increases for every notification and can be used to detect lost notifications.</p> <p>A management system is to be prepared for appending text to additional text, indicated by the <code>eriAlarmNObjMoreAdditionalText</code> varbind, and sent with <code>eriAlarmAppendInfo</code>.</p> <p>Do not confuse this with a change of additional text.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to “true” and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p>



Table 4 Notification Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmCleared Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmActiveSpecificProblem, eriAlarmActiveLastSequenceNo, eriAlarmActiveEventType, eriAlarmActiveEventTime, eriAlarmActiveProbableCause, eriAlarmNObjAdditionalText, eriAlarmNObjMoreAdditionalText, eriAlarmNObjResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.7	<p>This notification is sent when a resource detects a cleared alarm state. The combination of <code>ManagedObject</code> and <code>MajorType/MinorType</code> is always unique and must be used by management systems to correlate alarm and alarm clear. The corresponding row in the alarm table is deleted (<code>eriAlarmActiveAlarmTable</code>). The sequence number increases for every notification and can be used to detect lost notifications.</p> <p>A management system is to be prepared for appending text to additional text, indicated by the <code>eriAlarmNObjMoreAdditionalText</code> varbind, and sent with <code>eriAlarmAppendInfo</code>.</p> <p>Do not confuse this with a change of additional text.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to “true” and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p> <p>A management system is also to be prepared to receive a resource ID (OID) identifying the alarming resource if the system sending the notification can provide that information. In that case, <code>eriAlarmNObjResourceId</code> is set to “true” and the resource ID is sent in an <code>eriAlarmAppendInfo</code> notification.</p>
eriAlarmAppendInfo Type: { eriAlarmActiveManagedObject, eriAlarmActiveMajorType, eriAlarmActiveMinorType, eriAlarmNObjAdditionalText, eriAlarmActiveResourceId } OID: .1.3.6.1.4.1.193.183.4.2.0.8	<p>This notification is sent to append further information to an existing alarm. It can be additional text or a resource ID (OID), identifying the alarming resource using an OID.</p> <p>If additional text is sent, do not confuse this with an actual change of additional text, which is reported using the <code>eriAlarm</code> severity notification.</p> <p>A zero-length string value for <code>eriAlarmNObjAdditionalText</code> means that no additional text is sent in this notification.</p> <p>A null OID (0.0) value for <code>eriAlarmActiveResourceId</code> means that no resource ID is sent in this notification.</p>



Table 4 Notification Objects Defined in Ericsson Alarm MIB

Object Name and Properties	Description
eriAlarmHeartBeatNotif Type: { eriAlarmActiveLastSequenceNo, eriAlarmAlertLastSequenceNo, eriAlarmActiveLastChanged, eriAlarmAlertLastChanged } OID: .1.3.6.1.4.1.193.183.4.2.0.20	This is a heartbeat notification with interval according to eriAlarmHbInterval. It contains the last sequence numbers used for alarms and alarm events. These varbinds can be used to detect lost notifications. The notification eriAlarmHeartBeatNotif is sent every eriAlarmHbInterval. Managers can subscribe to the notification using the SNMP framework MIBs by using the snmpNotifyName "heartbeat" (SNMP NOTIFICATION MIB, snmpNotifyTable).
eriAlarmAlarmListRebuilt Type: { eriAlarmActiveTableURL } OID: .1.3.6.1.4.1.193.183.4.2.0.30	This notification is sent when the AAL has reached a stable situation after a restart or after a system internal audit process. It is an indication to the manager to perform an alarm resynchronization procedure. Deviation: Not supported. eriAlarmActiveTableURL always returns an empty string.

In the real notifications, OIDs for all varbinds that belong to eriAlarmActiveAlarmEntry or eriAlarmActiveAlertEntry are appended with corresponding unique index of alarms or alerts in eriAlarmActiveAlarmTable or eriAlarmActiveAlertTable respectively. For example, OID of varbind eriAlarmActiveManagedObject in an alarm notification corresponding to index "4" in eriAlarmActiveAlarmTable ends with a ".4", that is .1.3.6.1.4.1.193.183.4.1.3.5.1.5.4.

The same is applicable for other varbinds belonging to eriAlarmActiveAlarmTable or eriAlarmActiveAlertTable.





3 Configuration Management

The COM CM allows the operator to view and change the configuration data in the system.

3.1 Basic Concepts

Within the COM CM, the following term is used:

MIM File	The structure of the configuration data is specified in terms of Management Information Model (MIM) files. The MIM XML files correspond to the MOMs that are expressed using the UML™. The MIM XML files comply with the Ericsson internal DTD named <code>mp.dtd</code> . COM uses the MIM files to validate CM requests.
-----------------	--

The COM Runtime provides a tool to add, delete, and upgrade model files. For a detailed description, refer to *COM Application Developer's Guide*.

3.1.1 Logs

COM separates logging of important system events from debugging information. Audit information is also generated. To separate the log information from the debugging information, the NBI Agents use the Log SPI and the Trace SPI offered by COM.

For more information and exact location of the log files, refer to the documentation of the MW SA for each specific MW.

3.1.2 Configuration

The identity of the root MO, `ManagedElement`, is set at factory to some default value, normally 1. In a live network, the identity must be set to a unique value provided by the operator. The most important reasons for this are to be able to identify the Network Element (NE) in alarms, PM data, logs, and so on, and to be able to connect the NE MIB into the overall network MIB.

The naming attribute (identity) of an MO cannot be changed after creation, therefore COM provides an additional attribute on `ManagedElement` called `networkManagedElementId`. This attribute, if set, is used as the `ManagedElement` identity, replacing attribute `managedElementId`. If set to an empty string, the `managedElementId` is used.

The `networkManagedElementId` is intended to be used during commissioning of a NE but the attribute can be changed at any time. If it is



changed in runtime, all alarms in the AAL have the source MO updated so, but other data or files that have been generated in the system are not updated.

An example of factory setting is shown in Example 12.

```
* managedElementId=1
* networkManagedElementId=""
* Valid MO DN: ManagedElement=1, SystemFunctions=1, Fm=1
```

Example 12 Factory Setting

An example of live network settings is shown in Example 13.

```
* managedElementId=1
* networkManagedElementId=Stockholm
* Valid MO DN: ManagedElement=Stockholm, =>
SystemFunctions=1, Fm=1
```

Example 13 Live Network Settings

Change ManagedElementId

To change ManagedElementId:

1. Log on to the COM CLI, refer to *COM Command-Line Interface*.
2. Enter configure mode:

```
>configure
```

3. Change networkManagedElementId to the desired value:

```
(config)>ManagedElement=1,networkManagedElementId=<new value>
```

4. Commit changes and return to execution mode:

```
(config)>commit
```

3.2 Managed Object Models

The COM CM function delivers the MOMs specified in Table 5. These MOMs constitute a true subset of the Ericsson Common Information Model (ECIM). Other application-specific MOMs are also accessible from the COM NBI, but these are out of scope of this document.

Table 5 Managed Object Models

Managed Object Model	Description
Refer to <i>Managed Object Model com_top</i> .	Defines the top-level structure of the entire MO class hierarchy.



Managed Object Model	Description
Refer to <i>Managed Object Model com_fm</i> .	Defines the model that includes MO classes for the alarm model.
Refer to <i>Managed Object Model com_snmp</i> .	Defines the model that includes MO classes for the SNMP interface.
Refer to <i>Managed Object Model com_secm</i> .	Defines the model that includes MO classes for SM.
Refer to <i>Managed Object Model com_secm_ldap_authentication</i> .	Defines the model that includes MO classes for the LDAP authentication model.
Refer to <i>Managed Object Model com_secm_local_authorization</i> .	Defines the model that includes MO classes for the local authorization model.
Refer to <i>Managed Object Model com_filem</i> .	Defines the model that includes MO classes for FileM.

3.3 Configuration Management Using NETCONF

NETCONF is an XML-based protocol that defines operations for accessing and updating a configuration data store.

For detailed information on how to use the NETCONF interface, refer to *COM NETCONF Interface Base*.

3.4 Configuration Management Using CLI

The COM CLI is an interface to the OAM model. The model has a tree-like structure with a `ManagedElement` as the root, and the subparts and their attributes as child nodes.

When logging in to the CLI from the SSH, a session is started with the current location placed at a top node, named the ghost node, which is just above the root `ManagedElement` node.

Nodes are identified by their type name and instance ID. The instance ID is usually (by convention) a number, but can be any string. For example, a field or node storing the serial number of a board can be called `serialNumber=1`.

A field is equivalent to an attribute. A node is equivalent to an MO.

For detailed information on how to use the CLI, refer to *COM Command-Line Interface*.



3.5 Management of CM Interfaces

This section describes how to configure CLI and NETCONF services.

3.5.1 Configure CLI over SSH

Note: **INSTRUCTION FOR DOCUMENT REUSE:** Use this section only if the SSH management feature is supported by the product.

CLI over SSH can be enabled or disabled by changing the `administrativeState` attribute (LOCKED/UNLOCKED) of the `cliSsh` MOC in COM SysM model.

The behavior of CLI over SSH based on the state of the attribute `administrativeState` is defined as follows:

- If the administrative state attribute is LOCKED:
 - All the existing SSH-based CLI connections are closed.
 - No new SSH-based CLI connections are accepted.
- If the administrative state attribute is UNLOCKED:
 - Users are allowed to make new connections.
- The default value of administrative state attribute is UNLOCKED.

3.5.2 Configure NETCONF over SSH

Note: **INSTRUCTION FOR DOCUMENT REUSE:** Use this section only if the SSH management feature is supported by the product.

NETCONF over SSH can be enabled or disabled by changing the `administrativeState` attribute (LOCKED/UNLOCKED) of the `netconfSsh` MOC in COM SysM model.

The behavior of NETCONF over SSH based on the state of the attribute `administrativeState` is defined as follows:

- If the administrative state attribute is LOCKED:
 - All the existing SSH-based NETCONF connections are closed.
 - No new SSH-based NETCONF connections are accepted.
- If the administrative state attribute is UNLOCKED:
 - Users are allowed to make new connections.
- The default value of administrative state attribute is UNLOCKED.



3.5.3 Configure NETCONF over TLS

COM NETCONF over TLS is configured for certificates by changing attributes `nodeCredential` and `trustCategory` in the `NetconfTls` MO. If these attributes are not configured, then NETCONF TLS component configuration file is used for certificate information.

The configuration of NETCONF TLS is invalid in the following cases:

- When `trustCategory` or `nodeCredential` attribute refers to non-existing MO.
- When `trustCategory` or `nodeCredential` attribute is empty and the other refers to `CertM` MO.
- When one or both attributes refer to `CertM` MO, which has corrupted certificates data. Then connections are not accepted.

Note: Credential User API is required to use Certificates from `CertM` for NETCONF over TLS. If not found, NETCONF TLS component configuration file is used for Certificates.

The Example 14 show how to configure `NetconfTls` MO using CLI.

```
>configure
(config)>ManagedElement=1,SystemFunctions=1,SysM=1,NetconfTls=1
(config-NetconfTls=1)>nodeCredential="ManagedElement=1,SystemFunctions=1,SecM=1,
  CertM=1,NodeCredential=1"
(config-NetconfTls=1)>trustCategory="ManagedElement=1,SystemFunctions=1,SecM=1,
  CertM=1,TrustCategory=1"
(config-NetconfTls=1)>commit
```

Example 14 Configuring NetconfTls MO Using CLI

The Example 15 show how to configure `NetconfTls` MO using NETCONF.



```

<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="1" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <ManagedElement xmlns="urn:com:ericsson:ecim:ComTop">
        <managedElementId>1</managedElementId>
        <SystemFunctions xmlns="urn:com:ericsson:ecim:ComTop">
          <systemFunctionsId>1</systemFunctionsId>
          <SysM xmlns="urn:com:ericsson:ecim:ComSysM">
            <sysMId>1</sysMId>
            <NetconfTls xmlns="urn:com:ericsson:ecim:ComSysM" xc:operation="merge">
              <netconfTlsId>1</netconfTlsId>
              <nodeCredential>"ManagedElement=1, SystemFunctions=1, SecM=1, CertM=1,
                NodeCredential=1"</nodeCredential>
              <trustCategory>"ManagedElement=1, SystemFunctions=1, SecM=1, CertM=1,
                TrustCategory=1"
              </trustCategory>
            </NetconfTls>
          </SysM>
        </SystemFunctions>
      </ManagedElement>
    </config>
  </edit-config>
</rpc>
]]>]]>

```

Example 15 Configuring NetconfTls MO Using NETCONF

3.5.3.1 Administrative State

Attribute `administrative state` under the `NetconfTls` Managed Object Class (MOC) controls the NETCONF connections over TLS. The behavior of NETCONF over TLS based on the state of the attribute `administrative State` is defined as follows:

- If the `administrative state` attribute is `LOCKED`:
 - All the existing TLS-based NETCONF connections are closed.
 - No new TLS-based NETCONF connections are accepted.
- If the `administrative state` attribute is `UNLOCKED`:
 - Users are allowed to make new connections.
- The default value of `administrative state` attribute is `LOCKED`.

3.6 Model File Access

COM provides access to the MIM files loaded by COM. Data related to the MIM files is available through the NBI, according to the Schema class of `SysM` MIM. In runtime, there is one Schema instance for each configured MIM file.

The MIM files themselves can be downloaded with the export action of the Schema instances.



4 Security Management

This section describes the SM capabilities of COM and how to use them.

4.1 Overview

The COM SM consists of authorization support of the MOM for users accessing through the NBI. The MOM authorization is based on a combination of roles and rules. User authentication is performed by the SSH.

For an overview of the COM SM, see Figure 1. The illustrated SM sequence is as follows:

1. The SSH authenticates the Operation and Maintenance (O&M) by the Authentication Information Store to check the user credentials.
2. The SSH passes the user identity to the COM NBI agent.
3. The COM NBI agent indicates the start of a new user session to the COM SM.
4. COM retrieves the roles of the O&M user from the central Authorization Information Store.
5. COM retrieves the rules for the roles of the O&M user from the local Authorization Information Store.
6. On each O&M operation, the COM NBI agent asks the COM SM if access can be granted for the O&M user on the provided operation.

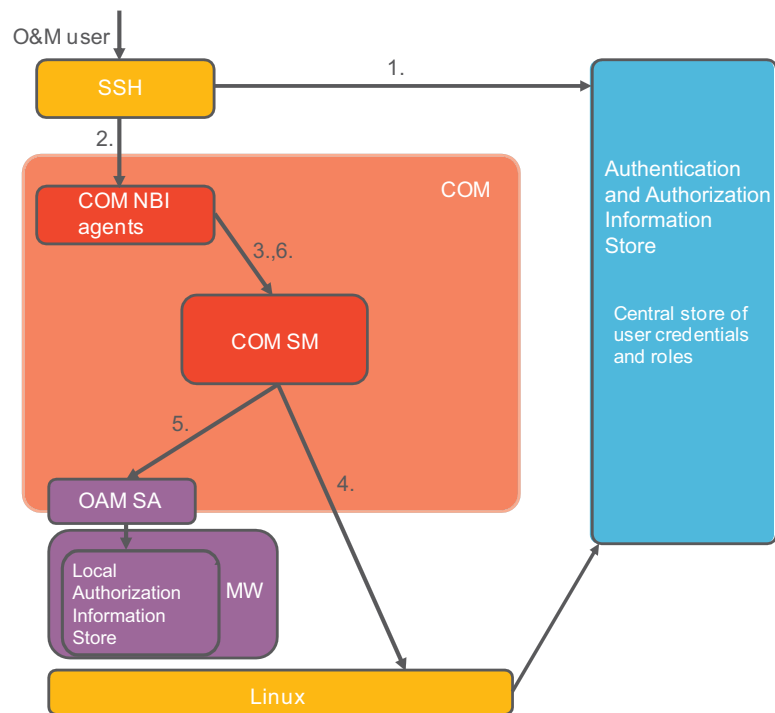


Figure 1 SM within COM

4.1.1 Authentication of NBI Users

COM configures the LDAP authentication method based on the attributes of the LDAP Authentication MOM.

For authentication, COM supports filtering on standard POSIX® accounts or Ericsson extended POSIX accounts.

Selective Authentication Based on Target Type

In some networks, it is required to not let all O&M users serving the network the right to log on to all MEs. To be able to categorize the MEs this way, the nodes must be configured with one or more target type identities. For example, the network can span several countries and it is needed to let an O&M user to be allowed to log on only to MEs in one of the countries.

4.1.2 Authorization Support of MOM for Users Accessing NBI

The roles for a user are stored in authentication data storage.

For gathering roles COM supports filtering on standard POSIX groups, Ericsson extended POSIX accounts, or by using custom filters. For more information, refer to *COM LDAP Interface*.



4.1.2.1 Target Based Access Control

COM supports Target Based Access Control (TBAC). In some networks, it is required to let an O&M user have different management roles on different MEs. To be able to categorize the MEs this way, the nodes must be configured with one or more target type identities by the LDAP Authentication fragment. For example, the network can span several countries and it is needed to let an O&M user act as “admin” in one country, but only as “operator” in another.

TBAC configuration of an ME needs a set of target classifiers. The target types of the ME can contain any classifier string for the ME, for example, geographical, such as `stockholm`, or network, such as `ims`, or functional identifiers, such as `cscf`, and any combination of those.

4.1.2.2 Roles Grouping, Role Aliases

COM supports grouping of roles. Different types of MEs have its own set of roles, because definition of roles is often done locally on the individual ME. For example, the role “administrator” has been specified as “admin” on one node and “adm” on another. To make administration of equal or similar roles easier, COM supports the concept of role alias. For example, the alias “admin” can be assigned to “admin”, “adm”, and “administrator”. When configuring authorization only the alias is used.

Role groups enable grouping real roles, where only real roles have meaning to the local system. The role group can be stored in the Ericsson extended POSIX account of the users. As a result, the LDAP administrator does not need to configure each real role into the POSIX account of the users, but configure them only into the alias role.

4.1.3 OS Level Authentication and Authorization

The classical OS level authentication and authorization is handled by the OS layer. Examples are file-level permission, binary execute permissions, and storage media permissions.

4.1.4 SM Behavior

This section describes the SM behavior, from a user perspective.

4.1.4.1 Administrative States

The administrative states are as follows:

- When the SM service is in locked state, no authorization is performed. Hence all MO operations, MO actions, and CLI commands are allowed.



Note: The terms “SM service in locked state” and “SM service in unlocked state” means the local state in the SM component based on administrative state for the ECIM authorization MO.

- When the SM service is in unlocked state, authorization is performed. The change of the `administrativeState` affects existing and new sessions.
- When the LDAP authentication method is in locked state, the LDAP Authentication Information Store becomes disabled. COM configures to use the local Authentication Information Store.

Note: When an O&M user has locked the LDAP authentication method or made a faulty configuration of the attributes of the LDAP class, the consequence is that COM does not authenticate any O&M user.

To reconfigure an LDAP object, refer to the “Authentication Failure” emergency operation in *COM Advanced Troubleshooting Guideline*.

- When the LDAP authentication method is in unlocked state, the LDAP Authentication and Authorization Information Store is enabled.

4.1.4.2 LDAP Searches

The LDAP Authentication MOM allows the configuration of an LDAP search base DN and a search base for LDAP role aliases (see Section 4.1.2 Authorization Support of MOM for Users Accessing NBI on page 32) to restrict LDAP searches to a specified portion of the LDAP directory. It also supports extending the LDAP search domain by enabling client referral chasing.

4.1.4.3 LDAP Filters

Based on the LDAP Authentication MOM, COM can use predefined filter types or a custom filter for searching user roles in the LDAP store.

Unlocking TBAC in the LDAP Authentication MOM also results in LDAP filtering (see Section 4.1.1 Authentication of NBI Users on page 32 and Section 4.1.2 Authorization Support of MOM for Users Accessing NBI on page 32), when the `ERICSSON_FILTER` filter is configured.

4.1.4.4 LDAP Servers

The LDAP Authentication MOM allows the configuration of a primary and a secondary LDAP server. First, an attempt is done to establish a connection to the primary server. An LDAP server can be addresses by a fully qualified domain name or an IP address.



4.1.4.5 LDAP Bind Method

If a bind DN and a bind password are provided, then COM uses a password-based simple bind method to the LDAP server, as described in the *RFC 4513* standard.

If a bind DN is not provided, COM binds anonymously.

4.1.4.6 LDAP Transport Layer Security

The LDAP sessions initiated by COM can be protected by TLS if configured in the LDAP Authentication MOM. If TLS is configured by `useTls`, TLS is used for both the primary and the secondary server; COM ignores `useTlsFallback`.

4.1.4.7 LDAP Certificates

COM supports the use of certificates for the LDAP connection. The LDAP Authentication MOM stores references to the SEC CertM MOM for the following:

- A node credential as client key and certificate represented by a `NodeCredential` MO in SEC CertM.
- A set of trusted certificates represented by a `TrustCategory` MO in SEC CertM.

For more information about the SEC CertM MOM, refer to *Managed Object Model SEC Certificate Management*.

To configure a `NodeCredential` and a `TrustCategory` MO in SEC CertM fragment, refer to *SEC Management Guide*.

The referred MOs in the LDAP MOM are set according to the generic rules of creating object references:

```
trustCategory="ManagedElement=1, SystemFunctions=1, SecM=1, CertM=1, TrustCategory=1"
nodeCredential="ManagedElement=1, SystemFunctions=1, SecM=1, CertM=1, NodeCredential=1"
```

Attributes `tlsClientCertificate`, `tlsClientKey`, `tlsCaCertificate`, and the corresponding OPI *Install Certificate for LDAP User Management in COM* are deprecated.

4.1.4.8 General SM Behavior

The general SM behavior is characterized by the following:

- At the start of every user NBI agent session, SM data for that session (user roles, rules) is read from the authorization data storage.

The SM data is updated during the session if an O&M user with the appropriate privileges changes, any or a combination of the following in the Local Authorization MOM:

a administrativeState

b CustomRoles

c CustomRules

d Roles

e Rules

- When the SM MIB is not initialized, the SM service allows any user to execute any NBI agent operation.
- Authorization exceptions: If any exceptions occur during authorization rule evaluation, such as rule syntax error, the operation is denied. The exception is logged in the system error log by the SM component.
- Access management failure in retrieving roles: If the SM service is unlocked, all MO operations, MO actions, and CLI commands are denied since role data is not available.

4.2 Security Management Functions

The conceptual view of the user authentication and authorization data is shown in Figure 2.

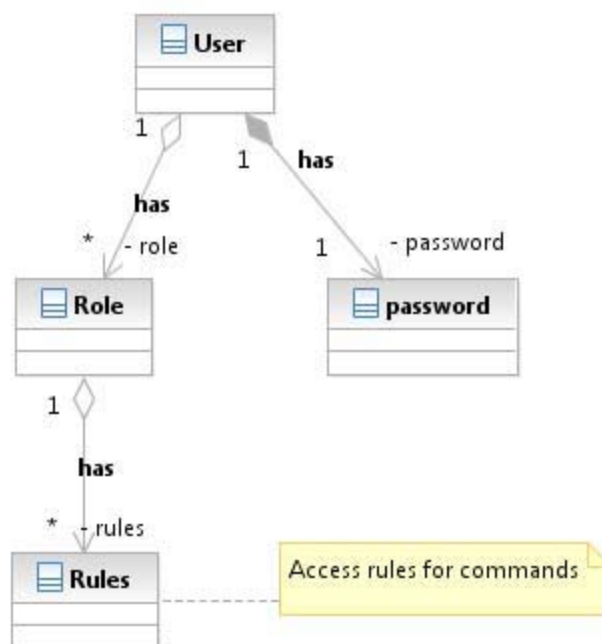


Figure 2 User Authentication and Authorization Data



4.2.1 Fault Management

The SM sends no notifications.

4.2.2 Audit Logging

Audit logging (that is, who logs on and off from the system) is not performed by the SM. The MW or OS layer authenticates the NBI user.

4.2.3 Audit Trail Logging

Audit trail logging (that is, who has done what and when) is not performed by the SM. Instead, it is performed by the NBI agents, using the Log SPI.

4.2.4 Capacity and Resource Consumption

The resource consumption depends on the size of the role and rule data in the authorization model.

4.2.5 Response Time

The response time depends on the authenticate data storage failover time.

4.3 Roles and Rules

COM delivers the two role instances `SystemAdministrator` and `SystemSecurityAdministrator`. COM also delivers the rule instances connected to these roles. These rule instances only cover the MOM fragments delivered by COM, thus other components and applications must contribute with their own rules.

COM supports implicit navigation through parent fragments. There is no need to specify read access to parent fragments explicitly. Thus the syntax `'MOC1=abc,MOC2=pqr,...,MOCN=xyz$'` for `ruleData` is deprecated. This is shown in Example 16.

```
ReadWriteExecute access to SecM:
    Rule name: SecurityManagement_1
    Permission: ReadWriteExecute
    Rule data: ManagedElement, SystemFunctions, SecM, *
```

Example 16 Example

Access is granted as follows:

- Read access to the `ManagedElement` MO, but not its attributes. Enables navigation in the CLI.



- Read access to the `SystemFunctions` MO, but not its attributes. Enables navigation in the CLI.
- RWX access to the `SecM` MO and all MOs below, including all attributes and actions.

4.3.1 Default Roles and Rules

This section describes the default roles and rules.

4.3.1.1 System Administrator

A system administrator is responsible for the administration of all non-security-related attributes and capabilities of an ME, for example, ME features, capabilities, configuration parameters, and monitoring of the ME.

The following rules are delivered by COM to provide the system administrator with proper access to model fragments supported by COM:

- Full access to the `ManagedElement` MO and its attributes.
- Full access to the `SystemFunctions` MO and its attributes.
- Full access to the `Transport` MO and its attributes.
- Full access to the `Fm` MO and all MOs below, including all attributes and actions.
- Full access to the `SysM` MO and all MOs below, including all attributes and actions.

4.3.1.2 System Security Administrator

A system security administrator is responsible for the administration of the attributes and capabilities of an ME related to security of the ME, regardless of which applications that execute on the ME, for example, ME administrative, user accounts, and authorizations.

The following rules are delivered by COM to provide the system security administrator with proper access to model fragments supported by COM:

- Read access to the `ManagedElement` MO, but not its attributes. Enables navigation in the CLI.
- Read access to the `SystemFunctions` MO, but not its attributes. Enables navigation in the CLI.
- Read access to the `Fm` MO and all MOs below, including all attributes.
- Full access to the `SecM` MO and all MOs below, including all attributes and actions.



4.3.2 Define Management System-Created Roles and Rules

The COM Local Authorization MOM provides for creation of roles and rules using the Management System. Roles and rules are created by instantiating `CustomRule` and `CustomRole` MO.

To define a custom rule:

1. Create an instance of the `CustomRule` MOC in `LocalAuthorizationMethod` MO.
2. Set appropriate values to attribute `ruleData` and permissions for the created `CustomRule` MOC.

To define a custom role:

1. Create an instance of the `CustomRole` MOC in `LocalAuthorizationMethod` MO.
2. Ensure that the rules attribute of `CustomRole` MO is referred to at least one `CustomRule` MO.
3. Set the `roleName` attribute.

It is mandatory to refer to at least one `customRule` MO in a `customRole` MO. It is not allowed to define a new `CustomRole` MO with an already existing `roleName`.

4.4 Procedures

This section provides examples of SM procedures.

The prerequisites for these procedures are as follows:

- The user must be assigned a system authentication role having sufficient privileges to access the SM branch.
- `ManagedElement`, `SystemFunctions`, `SecM`, `UserManagement`, and `LocalAuthorizationMethod` must exist with instance ID = 1. Attribute `networkManagedElementId` of instance `ManagedElement` must be unset or set to one.

4.4.1 Change Administrative State Using NETCONF

To change the administrative state using NETCONF, in this case setting the SM authorization to state `UNLOCKED`:

1. Use a NETCONF client to send the following to the NE where COM executes:



```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="1" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <ManagedElement xmlns="urn:com:ericsson:ecim:ComTop">
        <managedElementId>1</managedElementId>
        <SystemFunctions>
          <systemFunctionsId>1</systemFunctionsId>
          <SecM xmlns="urn:com:ericsson:ecim:ComSecM">
            <secMid>1</secMid>
            <UserManagement>
              <userManagementId>1</userManagementId>
              <LocalAuthorizationMethod =>
                xmlns="urn:com:ericsson:ecim:ComLocalAuthorization" xc:operation="merge">
                  <localId>1</localId>
                  <administrativeState>UNLOCKED</administrativeState>
                </LocalAuthorizationMethod>
              </UserManagement>
            </SecM>
          </SystemFunctions>
        </ManagedElement>
      </config>
    </edit-config>
  </rpc>
```

4.4.2 Change Administrative State Using CLI

To change the administrative state using the CLI, in this case setting the SM authorization to state UNLOCKED:

1. Use a CLI client to send the following to the NE where COM executes:

```
>configure
(config)>ManagedElement=1,SystemFunctions=1,SecM=1,UserManagement=1,LocalAuthorizationMethod=1
(config-LocalAuthorizationMethod=1)>administrativeState=UNLOCKED
(config-LocalAuthorizationMethod=1)>commit
```

4.5 Cipher Configuration

The COM SM TLS component handles the TLS MOC in ECIM SecM. The TLS MOC displays the ciphers supported by OpenSSL. The TLS MOC enables the O&M user to set a cipher filter based on the supported TLS ciphers listed by COM. This is a system-wide TLS configuration.

The COM SM TLS component uses the Object Implementer SPI to receive callbacks about configuration changes of the `cipherFilter` attribute, and validates if the configuration results in a valid `cipherSuite`. After a successful configuration change, the `enabledCiphers` attribute is updated and the COM components implementing NBI interfaces supporting TLS apply the cipher configuration. All protocols in the ME supporting TLS must apply the same configuration by reading the `cipherFilter` attribute after a configuration change. The cipher must be ordered according to strength; strongest first.

The attributes `supportedCipher` and `enabledCiphers` of the TLS MOC are not stored as configuration data, instead they are runtime data. The



`enabledCiphers` must be resolved from the OpenSSL stack and cached when the `cipherFilter` is updated.

4.5.1 Change cipherFilter Using CLI

To change the cipher Filter using the CLI:

1. Using a CLI client, enter the following to the NE where COM executes:

```
(config)>ManagedElement=1,SystemFunctions=1,SecM=1,Tls=1
(config-Tls=1)>cipherFilter=DEFAULT
(config-Tls=1)>commit
```

4.5.2 Change cipherFilter Using NETCONF

To change the cipher Filter using NETCONF:

1. Using a NETCONF client, enter the following to the NE where COM executes:

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc message-id="1" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <edit-config>
    <target>
      <running/>
    </target>
    <config xmlns:xc="urn:ietf:params:xml:ns:netconf:base:1.0">
      <ManagedElement xmlns="urn:com:ericsson:ecim:ComTop">
        <managedElementId>1</managedElementId>
        <SystemFunctions>
          <systemFunctionsId>1</systemFunctionsId>
          <SecM>
            <secMid>1</secMid>
            <Tls>
              <tlsId>1</tlsId>
              <cipherFilter>DEFAULT</cipherFilter>
            </Tls>
          </SecM>
        </SystemFunctions>
      </ManagedElement>
    </config>
  </edit-config>
</rpc>
```

4.6 Synchronization

With the introduction of ECIM LDAP 2.0 model some attributes in LDAP fragment are mapped to attributes that carry the same information, as follows:

- Attribute `profileFilter` in LDAP MO is mapped to `filterType` in the LDAP MO.
- Attribute `roleAliasesBaseDN` in `ericssonFilter` MO is mapped to `roleAliasesBaseDN` in LDAP MO.



- Attribute `targetType` in `UserManagement` MO is mapped to `nodeType` in `LDAP` MO.

To ensure that this COM implements a synchronization feature, so that whenever a user sets one of these attributes, the corresponding attribute is set so in the same `set` operation. The mapping is bidirectional, that is, synchronization is performed regardless of which mapped attribute is set.

If two synchronized attributes in the same MO are set to different values in the same `set` operation, the synchronization module returns an error to the NBI and the operation is ended.



5 File Management

FileM is a service implementing parts of the ECIM fragment `FileM` and it is characterized by the following:

- Provides a way for file producers and file consumers on the ME to organize and maintain (housekeep) files and folders in the file system.
- Provides and facilitates services to expose files and folders to NBI services, for example, the Secure SFTP, CLI, and NETCONF.

Note: Special characters, such as “+” in filenames, is displayed as “??” in the CLI. See 3GPP standard for MO Name convention, 3GPP TS 32.300 V9.0.0, for a full list of illegal characters.

All access to files and folders through the NBI is authorized by rules defined in the ECIM `SecM` fragment, see Section 4 on page 31.





6 Performance Management

The 3GPP Technical Specification (TS) 32.401 (3GPP TS 32.401 V10.0.0) defines PM as follows:

“Any evaluation of a system’s behavior requires performance data collected and recorded by its NEs according to a schedule established by the NE manager. This aspect of the management environment is termed Performance Management. The purpose of any Performance Management activity is to collect data, which can be used to verify the physical and logical configuration of the network and to locate potential problems as early as possible.”

TS 32.401 also defines the different phases in the process of performance measurement administration. COM is responsible of two of them; measurement job administration (section 4.2.1 in the TS) and local storage of the results at the NE (section 4.2.3 in the TS). Generation of the performance measurement results is the responsibility of the MW system.

6.1 Basic Concepts

The 3GPP TS 32.401 describes all the PM concepts. The most relevant concepts from a COM user point of view are as follows:

Job granularity period

Time between the initiation of two successive gatherings of measurement data triggered by this job.

Job reporting period

Time between the generations of two report files including measurement data generated by this job.

Measurement job

Process executed in the NE to accumulate measurement result data and assemble it for collection or inspection, or both.

ROP report file

XML file compliant with the 3GPP TS 32.432 (3GPP TS 32.432 V10.0.0) containing the results of one or more measurement collections triggered by one or more measurement jobs along one or more Granularity Periods (GPs).



6.2 Measurement Job Administration

The ECIM PM fragment provides support to perform all the tasks related with the administration of measurement jobs. The Pm MO is present under MO `SystemFunctions` and this is present under MO `ManagedElement`.

6.2.1 PmJob

Measurement jobs are represented as instances of class `PmJob` defined in the ECIM PM fragment. Administration of jobs (creation, deletion, and modification of job attributes) is made by accessing the ECIM through the NBI (through the CLI or NETCONF) and instantiating and deleting `PmJob` MOs, or changing the values of the attributes present in these MOs.

COM requires ECIM PM Version 1.2 or higher. For information about the meaning of the attributes and possible values, refer to *ECIM PM Documentation*.

Deviations in COM from the ECIM PM Version 1.2 are as follows:

- COM PM assumes that the Job Reporting Period is equal to the Job GP.
- COM PM does not support job priority.
- COM PM does not limit the number of measurements in a report.

6.2.2 Configure PmGroup

Several instances of class `PmGroup` can be found under the `Pm` MO instance. They describe the measurements that are available on the NE.

According to ECIM PM fragment documentation “*the PmGroup is a grouping of the measurements into logical grouping*”. Each `PmGroup` has one or more instances of class `MeasurementType`.

To add a measurement to a certain job:

1. Add an instance of class `MeasurementReader` under the `PmJob`.
2. Set the members of struct `measurementSpecification` so they contain a reference to the `PmGroup` and `MeasurementType` to be added to the job.

6.2.3 PmMeasurementCapabilities

Class `PmMeasurementCapabilities` of the ECIM PM fragment contains attributes describing the measurement capabilities of the current NE. For details of the meaning of each attribute, refer to *ECIM PM Documentation*.



6.3 Storage of Performance Measurement Results

Once the measurement jobs are set up, the MW system is responsible of collecting the measurement results for every job. Once the results are ready, at the end of the GP of every measurement job, the COM PM functionality writes results into Result Output Period (ROP) report files.

6.3.1 Location of ROP Files

The location where the ROP files are stored can be retrieved from the `fileLocation` attribute in `PmMeasurementCapabilities`.

If `FileM` is used:

- `fileLocation` = the relative path `/PerformanceManagementReportFiles`.
- `fileGroup` = DN of a `FileGroup` MO.

If `FileM` is not used:

- `fileLocation` = absolute path, for example: `/var/filem/internal_root/PerformanceManagementReportFiles`.
- `fileGroup` = empty.

The COM PM functionality performs housekeeping of PM report files after each granularity period ends. The number of reports published does not exceed the value of the attribute, `maxNoOfPmFiles` in `PmMeasurementCapabilities`.

Note: It is not allowed to set up `FileM` housekeeping policies of PM report files or the PM report files directories. Such a configuration leads to undefined behavior in the system. The `fileGroup` attribute is “deprecated” from ECIM PM 2.0 onwards.

6.3.2 Content of ROP Files

The PM report file format is according to *3GPP® TS 32.432* type A and is built on the example in *3GPP® TS 32.435* Annex A.2. In addition to this COM supports multivalue counters, which are not part of the 3GPP specification.

COM PM supports job grouping. This is indicated by the `jobGroupingSupport` attribute in the `PmMeasurementCapabilities` MO. The attribute is set to `TRUE` during COM PM startup. ROP files are written as follows:

- If `PmJob::jobGroup` attribute is present, a separate report file is generated containing all measurements for each `PmJob` containing this `jobGroup`. The `<UniqueId>` part of the filename is appended with “`_<jobGroup>`”.



- If `PmJob::jobGroup` attribute is absent, the default behavior is to include the measurement results for all such jobs in a single PM report file per granularity period, containing the results of those jobs for which no `jobGroup` tag was assigned.

If a `MeasurementType` instance (counter) is being measured in more than one PM Job, the measurement result is reported separately for each job in a measurement report file.

If COM does not support ECIM PM 2.1, the `jobGroupingSupport` attribute is not set and all measurement results are written to a single PM report file per granularity period.

6.3.3 Job Reporting Period

COM does not support measurements from several GPs in one ROP file. The `reportingPeriod` attribute and `granularityPeriod` attribute of the `PmJob` MO is used to write the report. COM PM expects these attributes to have the same value. The validation of these attributes is expected to be done at job creation time by the PM service in the SA/MW. COM PM sets the `PmMeasurementCapabilities::fileRPSupported` attribute to false during startup. This signals to the SA/MW that COM PM does not support `reportingPeriod!=granularityPeriod` and the SA/MW can validate so.

Example

The collection of measurement results is started when the time is 0:00. Six jobs are set up, two of them (job 1 and 2) have a GP of one minute, two of them (job 3 and 4) have a GP of five minutes, and the other two (job 5 and 6) have a GP of 15 minutes. The creation of the files is as shown in Table 6.

Table 6 Example of Created Files

Time	Number of Created Files	Content
Between 0:00 and 0:01	0	No files created. Measurement time collection has started at 0:00 and the GP has not ended.
Between 0:01 and 0:02	1	Measurement results for jobs 1 and 2 during period (0–1).
Between 0:02 and 0:03	1	Measurement results for jobs 1 and 2 during period (1–2).
Between 0:03 and 0:04	1	Measurement results for jobs 1 and 2 during period (2–3).
Between 0:04 and 0:05	1	Measurement results for jobs 1 and 2 during period (3-4).



Time	Number of Created Files	Content
Between 0:05 and 0:06	2	Measurement results for jobs 1 and 2 during period (4–5).
		Measurement results for jobs 3 and 4 during period (0–5).
Between 0:06 and 0:07	1	Measurement results for jobs 1 and 2 during period (5–6).
Between 0:10 and 0:11	2	Measurement results for jobs 1 and 2 during period (9–10).
		Measurement results for jobs 3 and 4 during period (5–10).
Between 0:14 and 0:15	1	Measurement results for jobs 1 and 2 during period (13–14).
Between 0:15 and 0:16	3	Measurement results for jobs 1 and 2 during period (14–15).
		Measurement results for jobs 3 and 4 during period (10–15).
		Measurement results for jobs 5 and 6 during period (0–15).

One minute is the upper time bound that it takes to get the results for a job written into a file after the end of the GP. After this time, the absence of an expected report file or the absence of measurement data for a configured active job is to be considered as “suspect”. This situation can be because of failures in the PM service in the SA/MW or because of file operation errors in which case the COM PM functionality logs an error.

6.4 ECIM PM Model Compliance

COM does not deliver the PM model. However, the ownership of the fragment is split between COM and the SA/MW. The ownership of COM lies in the ROP file-related attributes of the `PmMeasurementCapabilities` MOC. For more information, refer to *COM ECIM Statement of Compliance*.