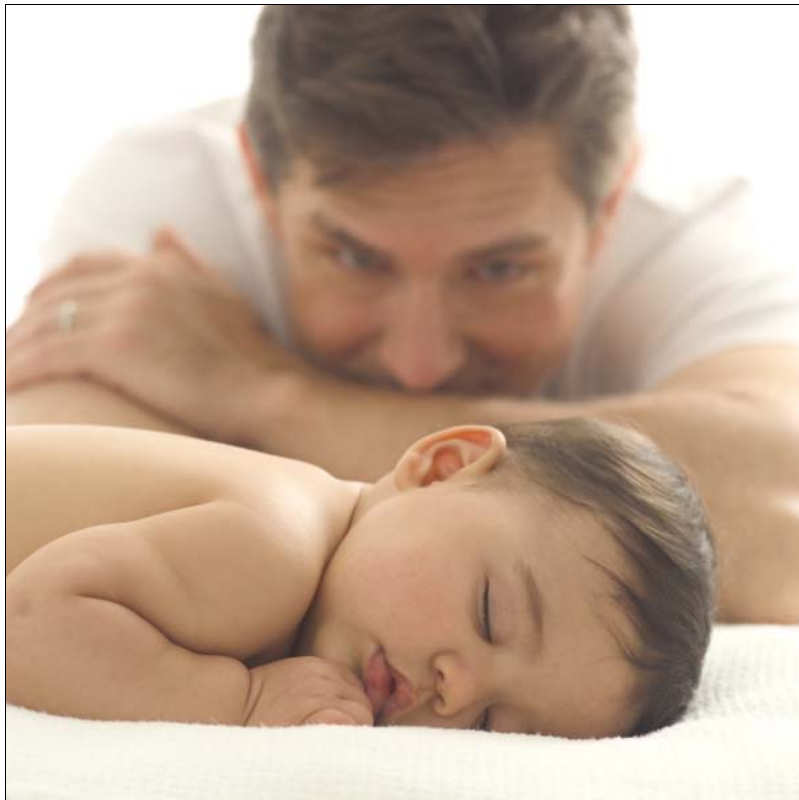


# ESA Technical Product Description

## Ericsson SNMP Agent 16.0

---

### TECHNICAL PRODUCT DESCRIPTION



**Copyright**

© Ericsson AB 2004-2016. All Rights Reserved.

**Disclaimer**

No part of this document may be reproduced in any form without the written permission of the copyright owner.

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**

Ericsson is the trademark or registered trademark of Ericsson AB. All other products or service names mentioned in this document are trademarks of their respective companies.



# Contents

<b>1</b>	<b>About This Document</b>	<b>1</b>
1.1	Purpose	1
1.2	Target Group	1
1.3	Prerequisites	1
<b>2</b>	<b>Product Overview</b>	<b>3</b>
2.1	Introduction	3
2.2	ESA Usage Scenarios	3
2.3	The OSS View	5
2.4	The ESA Software	6
<b>3</b>	<b>Network View</b>	<b>11</b>
<b>4</b>	<b>Function Overview</b>	<b>13</b>
<b>5</b>	<b>Architecture</b>	<b>17</b>
5.1	Layers	17
5.2	Internal Modules and Interfaces	19
<b>6</b>	<b>Deployment</b>	<b>21</b>
<b>7</b>	<b>System Requirements</b>	<b>23</b>
7.1	Hardware Specifications	23
7.2	Operating Environments	23
7.3	Java	24
<b>8</b>	<b>Application Programming Interfaces</b>	<b>25</b>
8.1	Fault Management API	25
8.2	Performance Management API	27
<b>9</b>	<b>Appendix A: SNMP Overview</b>	<b>29</b>
9.1	SNMP Architecture	29
9.2	The Network Entities	29
<b>10</b>	<b>Appendix B: RFC List</b>	<b>31</b>
	<b>Glossary</b>	<b>35</b>
	<b>Reference List</b>	<b>37</b>





# 1 About This Document

## 1.1 Purpose

The purpose of this document is to give a high level overview and introduction to the architecture and modules within the Ericsson SNMP Agent (ESA) and the features that come with it. Also, the document gives proposals and ideas about how to get the most out of the ESA when monitoring a system.

For more information about SNMP in general see Section 9 on page 29.

## 1.2 Target Group

This document is intended for anyone who needs an introduction to the ESA.

## 1.3 Prerequisites

-





## 2 Product Overview

### 2.1 Introduction

The Ericsson SNMP Agent (ESA) is a full featured O&M Framework that supports sending of SNMP alarms when faults occur in the system and receiving SNMP requests for reading/writing data in published MIBs on the system where the component is installed. The ESA also provides the capability to collect data from various data sources in the system in order to provide data for reports on system health and behavior as well for threshold monitoring when critical limits are reached indicated by the collected data.

The ESA comes with a flexible SNMP architecture, which means handling a Master agent and several sub agents. It allows configurable interfaces with potential already existing SNMP agents. Also, the ESA provides many features for the system designer for integration of SNMP functionality in completely new systems, without existing SNMP agents, as well as in mature systems, with SNMP agents already being installed and in operation.

The ESA supports a number of RFCs released by the IETF. The supported RFCs are listed in Section 10 on page 31. With the ESA you get a component that is based on international and widely used standards and not specifically for Ericsson proprietary solutions.

### 2.2 ESA Usage Scenarios

This section is meant to be a trigger to help and to give ideas to the system architect about how to successfully use the ESA component and the ESA features. Different systems have different needs and the ESA may be used for most of them. A new system may benefit from this SNMP agent as it is a full featured SNMP agent with a complete package of features, while a mature system may easily integrate the ESA component and use features that are not present in the existing SNMP agent.

The following scenarios, and others, are supported by the ESA component.

#### ☐ **You are developing a new system**

A new system is to be built and SNMP support for Fault Management and Performance Management is required. That is easily achieved by using the ESA, which provides a full featured SNMP agent that sends SNMP traps in an extended format based on international standard X.733. As the Fault Management solution provided by the ESA is vendor independent, it can be used by any OSS. By using the Performance Management in the ESA, any PM data in the system can be collected and provided as output on SNMP and/or 3GPP XML file format. Standardized formats makes the ESA output useful for any OSS.



☐ **You want your system to send SNMP alarms in a common format**

A system has several SNMP agents installed, but the alarms generated by the agents are in different formats. Using the ESA you could use the alarm translator, which translates the traps in any format to the ESA format. This solution makes it possible to get a single format out of your system without affecting your existing SNMP agents. One single trap format simplifies the OSS integration.

☐ **Your system report errors in log files only**

A system reports all or some of the system errors in one or several log files. With the ESA you may setup log file scanning and trigger SNMP notifications, based on the contents of the log file.

☐ **Your system needs SNMP reporting from Java components**

A system runs Java code and would like to have the Java components to send SNMP alarms. The ESA provides a Java API that may be used to trigger SNMP alarms directly from the Java component.

☐ **Your system need SNMP reporting from shell scripts**

A system runs scripts and it is desired that the script failures are reported as SNMP traps. The ESA provides an script API, which may be triggered directly from the scripts.

☐ **Your system need SNMP reporting when threshold limits are reached**

A system needs alarm notification when a threshold limit is reached. An example of a threshold limit is when the disk usage reaches 90%, and so an SNMP alarm is sent. The ESA provides setting of threshold limits, and sending SNMP alarms when a limit is reached. This may be done by only configuring the ESA. No redesign of existing system components is needed.

☐ **Your system is distributed on a multi-platform environment**

A system that has functionality distributed on multiple servers, where the servers are built on different operating environments, needs a common framework for handling the Operation and Maintenance. The use of ESA make the O&M handling simplified as the same configuration can be reused and distributed to several ESA installations on different operating environments.

☐ **Your system needs a secure SNMP v3 interface to the NMS**

A system that does not have SNMP support or have only SNMP v1/v2c support may use the ESA to get SNMP v3. This will allow system integrators to setup access rights using passwords and hide content by setting up encryption.

☐ **Your system needs to provide PM data**





A system that is required to provide PM data must consider issues like capturing of PM data, implement the scheduling for capturing the PM data and finally reporting of PM data on both the SNMP interface and to XML files in a 3GPP standard based format. By using the ESA all this is taken care of.

☐ **Your system needs to synchronize active alarms**

A system that is required to provide an active alarm list to an NMS can benefit from using the Subagent AAL Synchronization (SAS) function in the ESA. The SAS function gives the possibility to synchronize the ESA AAL with any other AAL in the system, resulting in having one single AAL to which the OSS can synchronize its alarms.

☐ **Your multi-node system must be seen as a single box**

A system with multiple nodes leads to alarms from many nodes. In case the OSS would like to have a single point for handling the alarm flow, the ESA can work in *Cluster Mode*. The cluster mode makes the ESAs on all the nodes in a system cluster work together as one single agent from an OSS point of view.

☐ **Your system needs to provide service level data**

A system is today not only monitored from a network monitoring point of view. There are also KPIs and SLAs to consider for the services being executed in the system. The ESA provides a flexible way of managing service level counters by instantiating counters on the fly depending on system feature entities, such as service level per client, service level per link, and more.

## 2.3 The OSS View

The ESA does not expect an Ericsson OSS to be receiving or collecting data from the ESA. Any OSS may be used as long as it supports using standardized IETF SNMP and 3GPP XML.

However, if Ericsson OSS systems are used the system using the ESA benefits from some degree of plug-and-play between the ESA and the Ericsson OSS system. The evolution of the ESA strives for enhanced plug-and-play in order to lower the amount of system integration needed at site between the Ericsson OSS systems and the ESA.

For an update of the ESA support in Ericsson OSS systems, please contact the support of the OSS system.



## 2.4 The ESA Software

### 2.4.1 Default Modules

The ESA contains a number of modules/processes to provide the O&M functionality to the users. The modules Master Agent, FM Agent and PM Agent are active by default while the Netcool System Service Monitor (SSM) is optional.

#### 2.4.1.1 Master Agent

The main features of the Master Agent (MA) are:

- **SNMP**

The MA handles the SNMP settings for the ESA.

- **SNMP Proxy**

All SNMP traffic that reaches the ESA goes through the MA. If the proxy is setup to forward the traffic to an SNMP subagent, the MA will redirect the operation to the correct subagent.

- **AgentX**

The AgentX has the same capability as the SNMP Proxy, which means redirecting operations to subagents, but using AgentX means the subagent registers itself to the MA.

- **ESA and System information**

The ESA comes with a few parameters that identifies the ESA itself and the system where the ESA resides. The MA maintains that data on the SNMP interface.

#### 2.4.1.2 FM Agent

The Fault Management Agent (FMA) provides the capability to send SNMP alarms to an OSS by simple operations using either a Java API or Script API (Command Line operations). The FMA is setup by using XML configuration files only. In other words, no programming is needed.

The main features of the Fault Management Agent (FMA) are:

- **Send alarms and events**

The FMA is responsible for creating SNMP messages when alarms/events are being triggered by an ESA user.



An alarm is stateful, while an event is stateless. An alarm is active until it is cleared. An event is not and can not be cleared.

- **Active Alarm List**

The FMA holds an Active Alarm List (AAL) which is a list of active alarms. The AAL is useful for the OSS in case there is a need to refresh its alarm list with the active alarms at the node. The ESA holds all active alarms and can be used for alarm synchronization at any time.

- **Filtering alarms**

The FMA controls the alarm flow and stops the flooding of the OSS when alarms of the same kind are repeated.

- **Alarm and Event logs**

The FMA provides history logs of alarms and events. This is important for fault tracing and root cause analysis.

- **API**

The FMA provides simple to use Java and Script APIs for triggering alarms.

### 2.4.1.3

#### **PM Agent**

The Performance Management Agent (PMA) provides the capability to make PM data from any data source in the system accessible from any OSS using either SNMP or FTP. The PMA is setup by using XML configuration files only. In other words, no programming is needed. When the PMA has been setup and the PM data collection is started, the OSS should be configured to read/capture the PM data from the ESA.

The main features of the Performance Management Agent (PMA) are:

- **Capture and Manage PM data**

The PMA can capture PM data from the following data sources:

- CSV
- JMX
- Script
- SNMP
- XML

By using the script option basically any data in the system can be captured by the PMA.

- **Receive and manage PM data**

The PMA provides APIs that allows users to push PM data to the ESA.

- **Publish PM data on SNMP**

The PM data that is held by the ESA is published on the SNMP interface. This means that it is very easy for any OSS to use SNMP to read PM data from a system using the ESA.

- **Publish PM data on 3GPP XML files**

The PM data that is held by the ESA can be published to files in 3GPP XML format. The files are created on intervals, which means a number of files can present a historical view of the PM data. The XML files are read from the ESA directories using common technologies, such as FTP.

The PMA in the ESA supports the following 3GPP standards:

- 3GPP TS 32.432 v11.0.0 (201209)
- 3GPP TS 32.435 v11.0.0 (2012-09)

- **Publish PM data in a Customized format**

The PMA supports customizing the output. It is possible to create a *output reporter*, which publishes the data in a format of own choice to any data storage. The output may be configured to fit the needs of the OSS collecting the files.

- **Threshold settings on PM data**

Thresholds can be defined on the PM data. This means that an SNMP alarm can be triggered when a PM data counter breaches a defined level. Also, the SNMP alarm can be cleared when the PM data falls back below the defined level.

- **API**

The PMA provides simple to use Java and Script APIs for counter handling.

## 2.4.2 Optional Modules

### 2.4.2.1 System Service Monitors

The System Service Monitors (SSM) is an optional module that is deployed with a separate license and has a separate deployment procedure.

The SSM is a system monitoring component, which means it can actively monitor many different sources and aspects of a system. The following features are the most common ones being used.

- **System Resource Monitoring**



The System Resource Monitoring is about monitoring system resources, such as CPU, Disk, Memory, Interfaces, and more.

- **Log File Monitoring**

The Log File Monitoring is about scanning log files on the file system and trigger SNMP alarms when searched strings are found.

- **Process Monitoring**

The Process Monitoring is about monitoring processes on the system and trigger SNMP alarms when processes are killed and/or restarted.

- **Heartbeat Operations**

This is a feature for sending heartbeat messages from an ESA to the OSS. The purpose of the heartbeat messages is to indicate that the system is alive.

- ***and more...***

The SSM contains more than 20 different monitoring capabilities. This is extensive enough to be able to setup any kind of system monitoring being required by customers.

Also, the SSM is setup with small and simple configuration files. No design is needed. The monitoring being setup is system and platform independent, which means the monitoring can easily be reused with other users and/or ported to other platforms.

## 2.4.3

### Tools

Together with the optional component SSM, which is part of the ESA product package, in the Windows version the tool **MIB Explorer** is included. The MIB Explorer provides basic tools for monitoring and debugging SNMP agents on your networks. It allows you to:

- Display a visual representation of MIB and OID structures
- View and graph data from MIB tables
- View output from subagents
- Send SNMP queries to agents
- Monitor SNMP traps



### 3 Network View

The ESA is a software component that provides an O&M Framework for Fault Management and Performance Management. It is installed on the system that is to be managed from an NMS and configured to provide the needed and required monitoring of the system.

The network view from the ESA point of view is rather simple. The ESA communicates using SNMP and provides data that can be captured using FTP.

The figure below shows the ESA located on the managed system sending alarm information using SNMP to an NMS and also responding to SNMP operations, either push or pull operations using *SNMP set* or *SNMP get*, from the NMS.

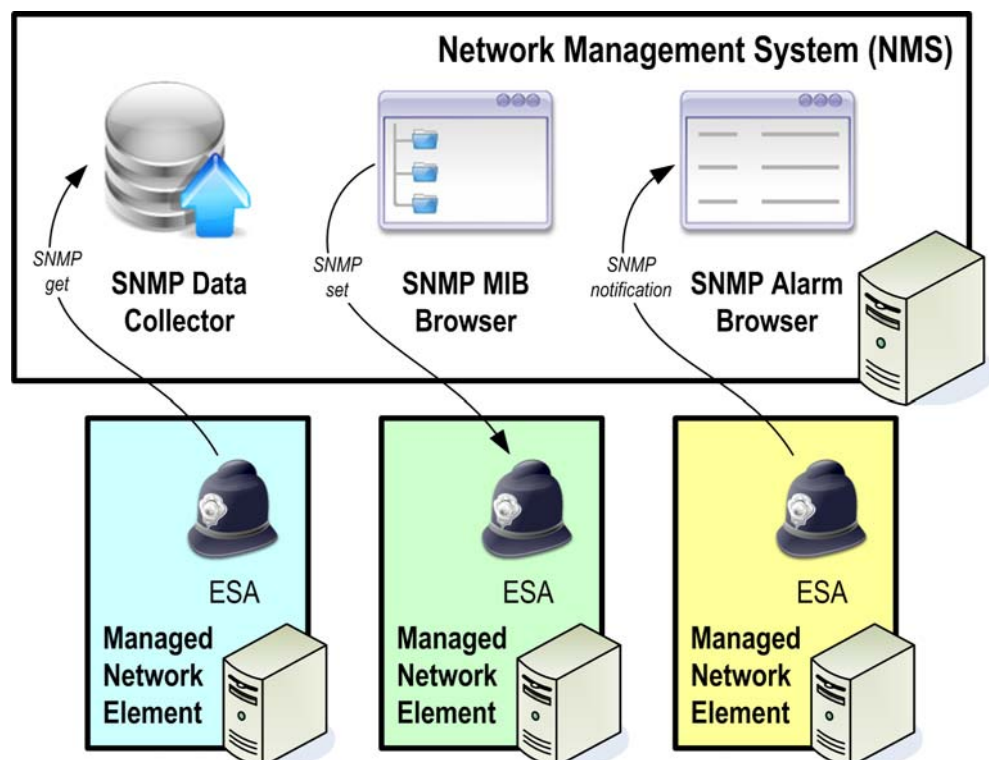


Figure 1 The ESA in the network interfaced using SNMP.

The figure below shows the ESA located on the managed system providing PM data stored in 3GPP XML files. The ESA stores all the collected data into XML files in a directory on the file system. Any OSS can be configured to fetch the PM data files from the ESA storage for further data processing in reporting tools on the OSS.

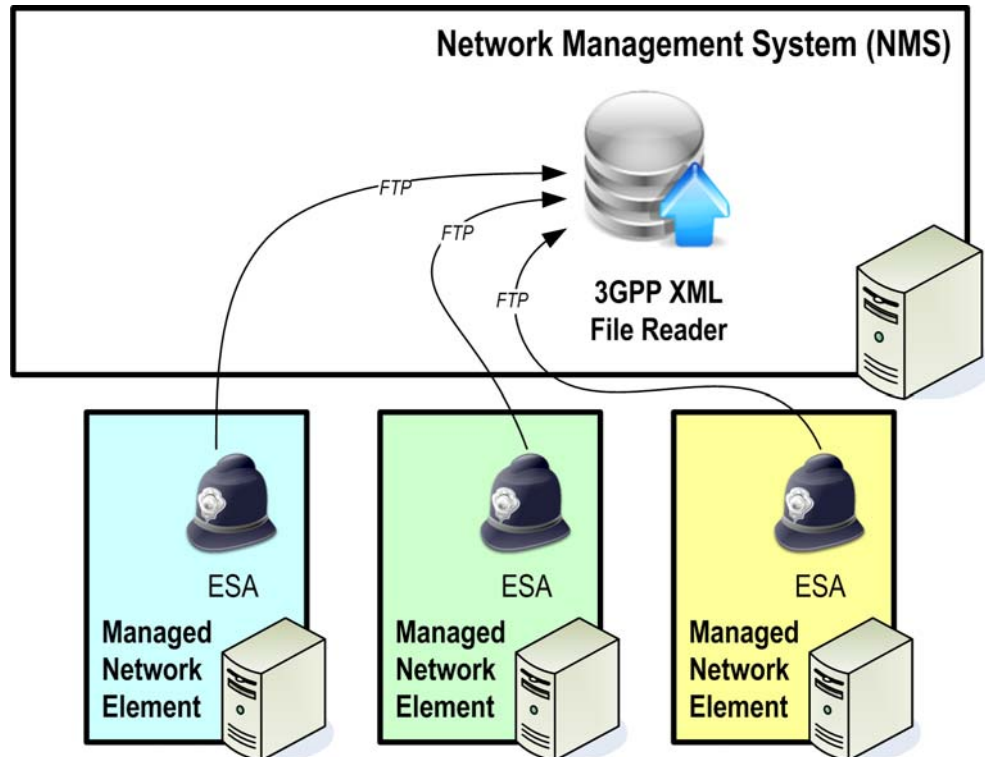


Figure 2 The ESA in the network interfaced using FTP.





## 4 Function Overview

This section gives an overview of the functions and concepts that come with the ESA. They are listed in alphabetical order.

### ☐ **Active Alarm List (AAL)**

This function keeps a list of active alarms and it gives an NMS the possibility at any time to synchronize the alarm status on the node. In case the connection between the NMS and the managed system was lost or if the NMS was out of order, the AAL always contains the current alarm status and the NMS may reload all the alarms even though it was unavailable at the time the alarm was sent.

### ☐ **Alarm/Event Handling Manager including APIs**

This function provides the system with an SNMP notification sending function. The function is configured by defining all possible alarms/events to send from a system in a configuration file in the XML format. The triggering of sending a notification is by using the Java API or the Script API.

Also, the function is responsible for managing the Active Alarm List (AAL). If the SNMP notification is an *alarm raise* the notification is put into the AAL in order to be cleared only by a succeeding *alarm clear*. If the SNMP notification is an *event*, the SNMP notification is sent to the NMS without being put into the AAL.

### ☐ **Alarm/Event Translation**

This function gives the possibility to change the format of any trap to a common trap format. The trap format used by the ESA is a format based on the IETF Alarm MIB (X.733). One common format from the managed nodes makes the integration towards the NMS simple. The function is setup in configuration files and does not require design of program code.

### ☐ **Alarm Filtering**

This function makes it possible to filter alarms on the system before being sent to the NMS. This is useful for cyclic and toggling alarms that are of low or no interest. Filtering the alarms on the system side makes NMS less loaded and the NMS users can focus on more important alarms. The function is setup in configuration files and does not require design of program code.

### ☐ **Alarm API**

This function provides the system with the capability to trigger alarms from any application using java and/or scripts. The alarms may be sent to any OSS capable of handling SNMP messages.



☐ **Automatic Clear Control**

This function is automatically hindering “alarm cease” to be sent to the NMS if no “alarm raise” have been sent before. This is useful in situations where one or several system components are badly designed and are sending “alarm cease” even though no “alarm raise” have been sent.

☐ **Automatic Flooding Control**

This function is automatically hindering multiple alarms of the same kind being sent to the NMS. Once an alarm is sent, the following alarms of the same kind are stopped until the active alarm is cleared. This is useful in situations where one or several system components are badly designed and repeatedly are sending the same alarm, which may flood and put a heavy load on the NMS.

☐ **Cluster Mode**

This capability allows multiple ESAs to operate as a single unit from an OSS point of view. The OSS will get the alarms from one single point only instead from each ESA in the cluster separately. Also, see *High Availability*.

☐ **High Availability**

This capability allows the ESAs in the cluster to be setup to manage the alarms for other members in the cluster. If the Master in the cluster is lost, another Master is automatically assigned to continue. This is also known as the *Cluster Mode*. This setup comes with additional capabilities, such as Alarm Redundancy and Alarm Persistency.

☐ **Information Agent**

This function gives the OSS the possibility read ESA and System information using SNMP get operations for use in decision making in relation to possible operations on the ESA as well as using the information in OSS output, such as in reports. The information is related to identification, version numbering and dates.

☐ **Log File Scanning**

This function provides the system with the capability to scan log files. When a certain string is found in the log file scanning process an alarm/event may be triggered. This function is specifically useful for 3PP components that only uses log files for faults and that need an SNMP interface. The function is setup in configuration files and does not require design of program code.

☐ **Microsoft Windows Performance Management**

This function is specifically targeting the Microsoft Windows operating system. It accesses the performance metrics provided by the Windows performance monitoring utility `perfmon` and provide functions for monitoring and controlling Windows services, which allow you to start or stop a service or driver in response to changes in its activity.



## ☐ **Monitoring of System Resources**

This function provides the system with the capability to monitor system resources. Devices such as processor, memory, disks and network interfaces are monitored just by configuration of ESA. To the monitoring thresholds may be configured in order to send alarms on wanted levels. The function is setup in configuration files and does not require design of program code.

## ☐ **PM API**

This function provides the system with the capability to push PM data from any application using java and make it available on SNMP and in XML files. An OSS can then be connected to the system by reading either SNMP data, the XML files or both in order to capture the PM data of the system. The function is setup in configuration files and does not require design of program code.

## ☐ **PM Data Collector**

This function provides the system with the capability to capture PM data from basically any source and make it available on SNMP and in XML files. An OSS can then be connected to the system by reading either SNMP data, the XML files or both in order to capture the PM data of the system. The function is setup in configuration files and does not require design of program code.

## ☐ **Process Monitoring**

This function provides the system with the capability to monitor system processes. The process may be restarted when found being not running. Also, alarms may be triggered when the process is killed as well as when it is restarted. The function is setup in configuration files and does not require design of program code.

## ☐ **Proxy**

This function allows the reuse of existing SNMP agents in a system by structuring the agents in a Master/sub relationship. The SNMP proxy is setup using a simple configuration file in the XML format. The Master/sub agent protocol is SNMP v1/v2c/v3.

Also, AgentX is supported. This does not require any XML configuration, but the subagent to connect to the ESA must also support AgentX to get this to work.

## ☐ **Subagent AAL Synchronization**

This functions allows the ESA AAL to be synchronized with other AALs in the system, resulting in having one single AAL to which the OSS can synchronize its alarms. When the ESA or the subagent starts/restarts the AALs can be synchronized with the one already running.

☐ **Threshold Monitoring including triggering of alarms**

This function makes it possible to set thresholds on any MIB variable. This means that alarms may be triggered when a value on a MIB leaf is exceeding the defined threshold. The function is setup in configuration files and does not require design of program code.

☐ **Timer Events**

This function makes it possible to schedule events, such as “clear the log files once a day”. The function is setup in configuration files and does not require design of program code.

☐ **Trap Forwarding**

This function is useful when there is a need to have the ESA as the one and only trap handler in a large system with multiple servers. The surrounding servers can send all their alarms to the ESA and the ESA can either translate the alarms into the ESA format before sending them to the NMS or transparently send the alarms to the NMS, which means the alarms are sent in its original format.

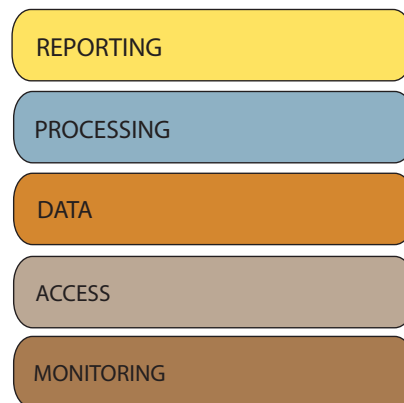
☐ ***...and more...***

*Not all the functions are listed in this list. Only the most important, most useful and most used ones are listed. For a complete set of functions see the documents in Reference [1] and Reference [2].*

## 5 Architecture

### 5.1 Layers

The ESA is built with the following layers.



*Figure 3 The ESA is built with five layers in mind.*

The layers from bottom-up have the following responsibilities and features:

- **Monitoring Layer**

This layer handles the data capturing to the ESA. The data captured can be anything that results in an alarm notification or is to be used as PM data in the ESA layers above. This layer can be enhanced with own monitoring components.

- **Access Layer**

This layer handles the transport of information from any source to the ESA by providing APIs and data capturing functions. The incoming data is stored in the Data layer.

- **Data Layer**

This layer manages the O&M Data that flows through the ESA.

When cluster mode is active the data layer is synchronized over multiple ESA instances. See below.

- **Processing Layer**

This layer handles the processing of O&M data. The incoming data, which is alarm and PM data stored in the Data layer, is processed according to the configuration setup in the ESA and made ready to be reported to external users.

When cluster mode is active the processing layer is active over multiple ESA instances. See below.

- **Reporting Layer**

This layer handles the reporting of data to external users, such as OSS technicians receiving SNMP alarms, according to the configuration setup in the ESA.

When cluster mode is active the reporting layer is active over multiple ESA instances. See below.

The layers provide additional capabilities when working in *Cluster Mode*, which means multiple ESAs work together as a single unit.

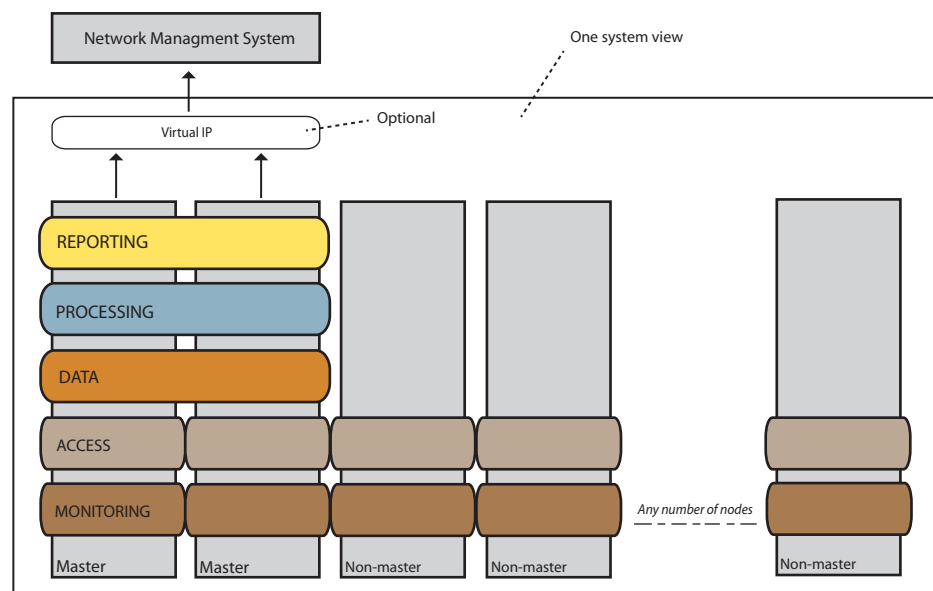


Figure 4

In the cluster the monitoring of a node is done locally and the reporting of alarms and events are done to the local ESA as well. The alarm being triggered at the local Access layer is stored in the alarm queue in the Data layer. The Processing layer is only active in the Master nodes. These will handle the O&M for the entire cluster in order to not put load on the Non-Master nodes. The Processing layer operates on the alarm queue in the Data layer. After processing the alarm status is updated and again it is stored in the Data layer. This means that the alarm status for the entire cluster is accessible and viewable from any Master in the cluster. The Reporting layer is responsible for handling the alarm transfer, such as SNMP messages, from the ESA to the receivers defined.



## 5.2 Internal Modules and Interfaces

The ESA internal architecture is shown in the picture below. The shape and color of the boxes shows which layer they belong to.

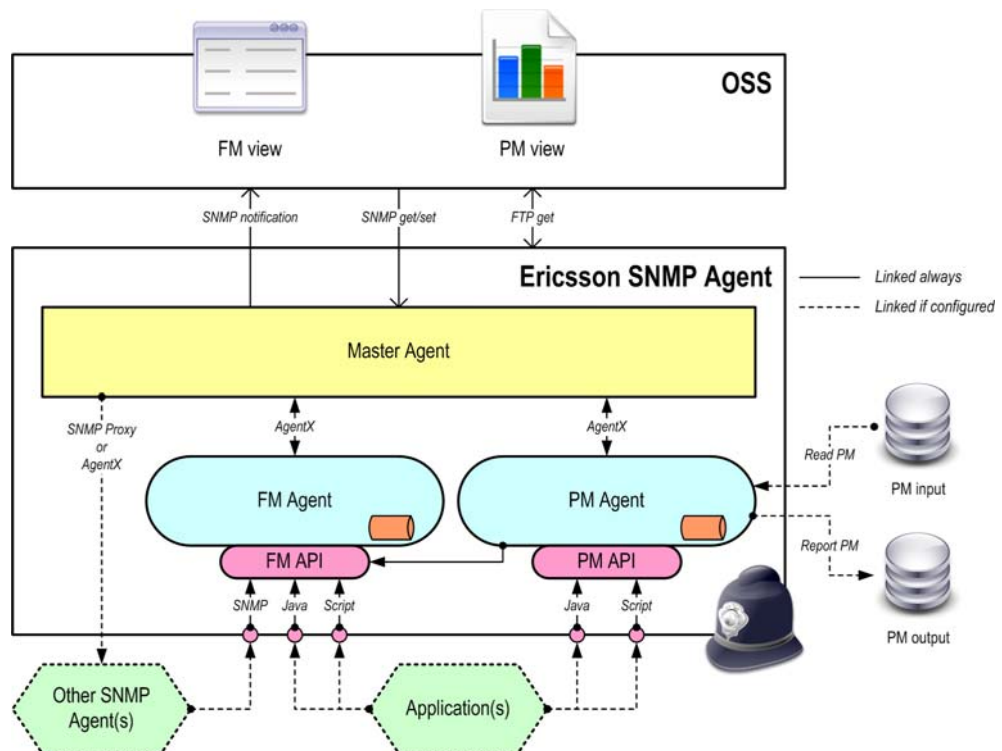


Figure 5 The ESA internal architecture and external interfaces.

The following list is a short description of the ESA modules.

- **Master Agent**

This module handles the proxy functionality as well as the SNMP interface. Also, it contains the ESA Information Agent.

- **FM Agent**

This module receives alarm triggers and manages the sending of alarms/events to the OSS. Also, the Active Alarm List (AAL) is managed as well as the alarm filtering and translations from integrated SNMP subagents. The module holds FM data.

- **PM Agent**

This module handles the collection of data from system data sources and the presentation of PM data on SNMP and to File. The module holds PM data.

- **System Service Monitor**



The SSM module is an optional component in the ESA software package and is related to the box "Other SNMP agent" in the figure above.

It handles the capturing of system data and the triggering of alarms on defined events and thresholds.

- **FM API**

Information about the FM API is found in Section 8.1 on page 25.

- **PM API**

Information about the PM API is found in Section 8.2 on page 26.





## 6 Deployment

The ESA is a single package software component. It is installed using one of the following procedures:

- **GUI Mode**

This installation process allows a user to monitor the ESA installation in a GUI. Also, the GUI provides forms that allows the user to give input data when it comes to the ESA configurable parameters. Default values are presented, but the user may change them.

- **Console Mode**

This installation process allows a user to monitor the ESA installation in a console. Also, the console provides forms that allows the user to give input data when it comes to the ESA configurable parameters. Default values are presented, but the user may change them.

- **Silent Mode**

This installation process allows a user to before the installation create an ESA installation response file. This response file contains the defined values for all the configurable parameters for the ESA. The response file is used as input to the installation process, which makes the installation process to not ask for parameter data, that is making the installation simple and silent with only one single command.

This installation is specifically useful when installing the ESA with the same parameters on multiple servers.

- **Linux RPM**

The Linux RPM installation is an installation specifically for the Linux platforms by using the RPM package format. The RPM installs the ESA with default values. Customized parameters are not possible during the installation procedure. Instead, if needed, the parameters are customized after the installation.

The deployment procedures are independent of which platform the ESA is installed into.

The ESA software comes in three different formats.

- **esa-<version>.exe**

The `esa-<version>.exe` is used on the Windows platforms.

- **esa-<version>.sh**



The `esa-<version>.sh` is used on any Unix or Linux platform.

- **esa-<version>.rpm**

The `esa-<version>.rpm` RPM package format is used on the Linux platforms.



## 7 System Requirements

### 7.1 Hardware Specifications

The hardware specifications as such are no issue for the ESA. The modern hardware of today is more than enough for running the ESA.

However, the following architectures are required to run the ESA:

- For Solaris; Sparc and x86/x64 architectures are supported.
- For Linux; x86/x64 architecture is supported.
- For Windows; x86/x64 architecture is supported.

### 7.2 Operating Environments

Figure 6 shows the operating environments supported.

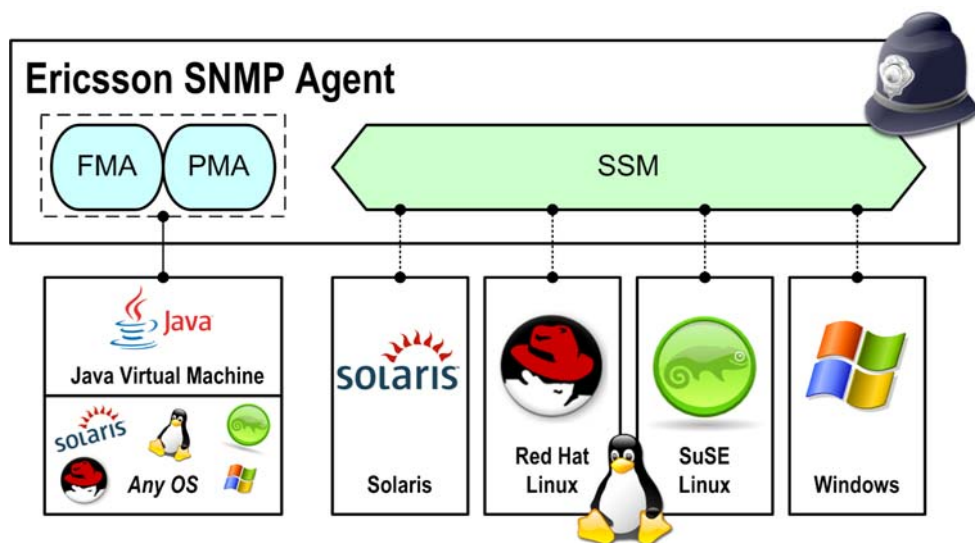


Figure 6 The operating platforms supported by the ESA.

The figure above shows that the ESA basic package is developed in java, runs in a Java Virtual Machine and can thus be run in any operating environment that supports java. The optional component SSM is however only supported on a selected number of operating environments. The following variants are valid for the SSM:

- **Red Hat Linux**
  - Red Hat Enterprise Linux 6

- Red Hat Enterprise Linux 7
- **SuSE Linux**
  - SuSE Linux Enterprise Server 10
  - SuSE Linux Enterprise Server 11
  - SuSE Linux Enterprise Server 12
- **Solaris**
  - Solaris 10
  - Solaris 11
- **Windows**
  - Windows Vista
  - Windows Server 2008
  - Windows Server 2008 R2
  - Windows 7
  - Windows 8
  - Windows Server 2012

Also, other variants *may* work properly, but only the ones listed above are tested, verified and officially supported.

## 7.3 Java

The ESA is supported on the following Java environments.

- Oracle Java 8 or later
- OpenJDK 8 or later



## 8 Application Programming Interfaces

### 8.1 Fault Management API

#### 8.1.1 Send Alarm

The ESA comes with an API for handling the SNMP alarms being sent from the applications in your system. System Integrators and System Developers can benefit from using SNMP, Java and Script interfaces in their design of SNMP handling.

Also, even though there are several interfaces for triggering SNMP alarms, the SNMP alarms sent from the ESA are always in the same format. Having only one SNMP format from the system is a benefit for the integration activities of your system to the NMS.

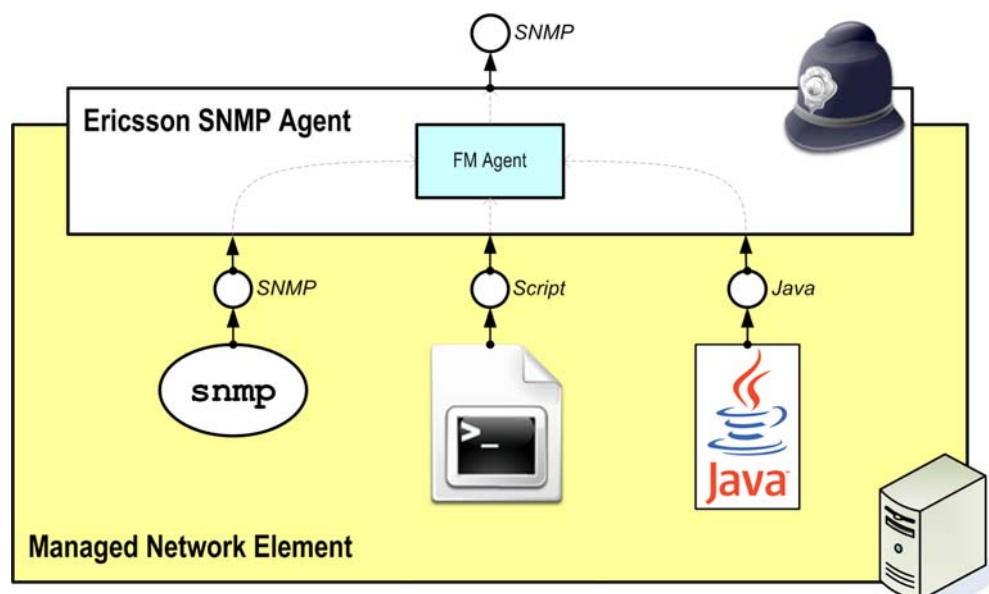


Figure 7 The ESA Fault Management Application Programming Interface.

As seen in Figure 7 the ESA API provides the following interfaces:

- **SNMP**

Already existing SNMP agents in the system can be connected to the ESA.

**Note:** The SNMP interface is actually not considered being an “API”, but it is mentioned here as it is a useful interface for SNMP design.

- **Script**

This interface allows sending SNMP alarms using Shell scripts.

- **Java**

This interface allows sending SNMP alarms using Java code.

### 8.1.2 Active Alarm List

The ESA comes with an API for handling the Active Alarm List (AAL). Using the API it is possible to read the content of the AAL, but also modify the content.

This is useful for applications that require a status update of the active alarms in the system, such as a GUI application. Also, applications could read the AAL content to see if an alarm is active or not before taking action on an operation.

### 8.1.3 Synchronize with Subagent Alarms

The ESA comes with an API used for synchronizing Active Alarm Lists in other SNMP agents or other resources, such as files, holding active alarms.

The figure below gives an overview of the function; The ESA starts up, which triggers the Alarm Service module to initiate the SAS API to call the remote SAS server. The SAS server must be developed to suit the data sources holding active alarms in the system where the ESA resides. The SAS server collects the active alarms from the Active Alarm Lists and returns them to the ESA. The Alarm Service builds up the Active Alarm List in the ESA and starts up.

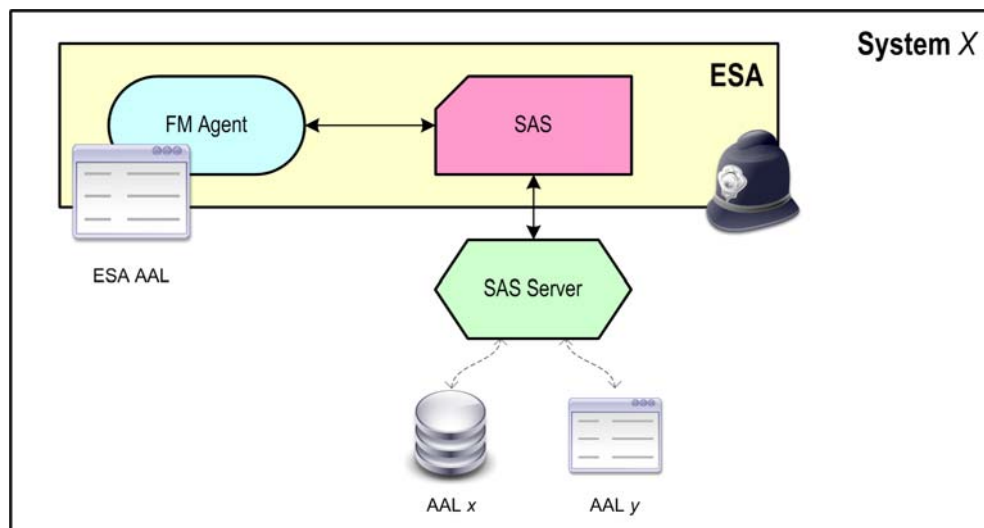


Figure 8 An overview of the Subagent AAL Synchronization API and function.



## 8.2 Performance Management API

### 8.2.1 Counter Handling

The ESA comes with an API for handling the PM data within the system. System Integrators and System Developers can benefit from using the java and script interfaces in their design of collecting PM data.

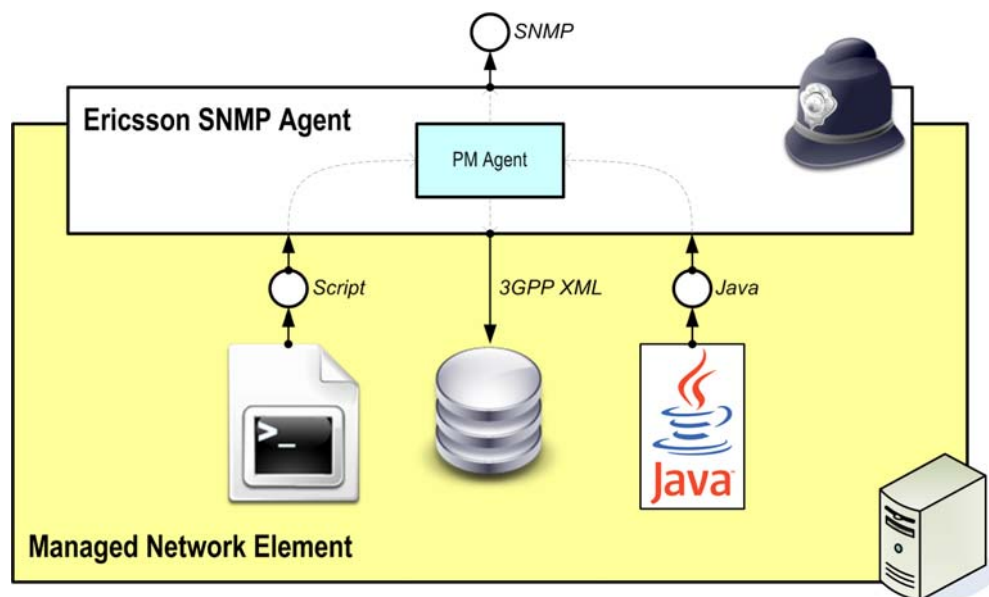


Figure 9 The ESA Performance Management Application Programming Interface.

As seen in Figure 9 the ESA API provides the following interfaces:

- **Java**

This interface allows publishing PM data using java code.

- **Script**

This interface allows publishing PM data using shell scripts.







## 9 Appendix A: SNMP Overview

### 9.1 SNMP Architecture

The definition of the SNMP architecture is as follows:

*“Implicit in the SNMP architectural model is a collection of network management stations and network elements. Network management stations execute management applications which monitor and control network elements. Network elements are devices such as hosts, gateways, terminal servers, and the like, which have management agents responsible for performing the network management functions requested by the network management stations. The Simple Network Management Protocol (SNMP) is used to communicate management information between the network management stations and the agents in the network elements.”*

The definition simply says that in the network to manage there are (at least) one SNMP manager which manages the SNMP clients and that the communication between the manager and the clients are provided by using the protocol SNMP. Also, each client contains an agent, which is the component managing the SNMP traffic to/from the client.

The protocol SNMP is described formally in the Internet Engineering Task Force (IETF) Request for Comment (RFC) 1157 and in a number of other related RFCs.

### 9.2 The Network Entities

*The SNMP Manager* may be used for the following.

- Receive event information from an SNMP Agent.
- Request data from an SNMP Agent.
- Sending orders to an SNMP Agent.

*The SNMP Agent* is running on a managed system in order to send and receive SNMP messages to/from the SNMP Manager.





## 10 Appendix B: RFC List

The following RFCs are supported by the ESA software component.

- **RFC 1155**  
SNMP v1; Structure and Identification of Management Information for TCP/IP-based Internet
- **RFC 1157**  
SNMP v1; Simple Network Management Protocol (SNMP)
- **RFC 1213**  
Management Information Base for Network Management of TCP/IP-based internets: MIB-II
- **RFC 1303**  
A Convention for Describing SNMP-based Agents
- **RFC 1513**  
Token Ring Extensions
- **RFC 1757**  
Remote Network Monitoring Management Information Base
- **RFC 1901**  
SNMP v2; Introduction to Community-based SNMPv2
- **RFC 1905**  
Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)
- **RFC 2012**  
TCP
- **RFC 2013**  
UDP
- **RFC 2021**  
Remote Network Monitoring Management Information Base Version 2 (RMON-II) using SMIv2
- **RFC 2233**  
Interfaces
- **RFC 2578**  
Structure of Management Information Version 2 (SMIv2)
- **RFC 2741**

- Agent Extensibility (AgentX) Protocol Version 1
  - **RFC 2742**
- Definitions of Managed Objects for Extensible SNMP Agents
  - **RFC 2790**
- Host Resources MIB
  - **RFC 3014**
- Notification Log MIB
  - **RFC 3410**
- Introduction and Applicability Statements for Internet Standard Management Framework
  - **RFC 3411**
- An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks
  - **RFC 3412**
- Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)
  - **RFC 3413**
- Simple Network Management Protocol (SNMP) Applications
  - **RFC 3414**
- User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)
  - **RFC 3415**
- View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)
  - **RFC 3416**
- Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)
  - **RFC 3417**
- Transport Mappings for the Simple Network Management Protocol (SNMP)
  - **RFC 3418**
- Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)
  - **RFC 3419**
- Textual Conventions for Transport Addresses
  - **RFC 3584**



## Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework





# Glossary

## **Glossary**

*ESA Glossary of Terms and Acronyms,*  
0033-CSH 109 532







## Reference List

- [1] *IBM Netcool/System Service Monitors Version 4.0.1, Administration Guide*
- [2] *IBM Netcool/System Service Monitors Version 4.0.1, Reference Guide*