Mitel OpenScape Fault Management

Mitel OpenScape Fault Management V13 Enterprise MIB Plugin

User Guide

10/2025

Notices

Trademarks

Contents

1 Preface

This chapter discusses the following aspects:

- purpose of this guide
- audience of this guide
- terminology
- organization of this guide
- conventions used in this manual

1.1 Purpose

This manual gives an introduction into the work with the Enterprise MIB Plugin module for the OpenScape FM Desktop.

1.2 Audience

This guide is addressed to users who want to learn how to use the Enterprise MIB Plugin. Since this is not a standalone application but a Plugin to the OpenScape FM, you should know how to work with OpenScape FM before using the Plugin. For several functions of the Plugin, which are based on Desktop features, we will base our explanations on those given in the Desktop Manual, so please read the *OpenScape FM Desktop User Guide* for a comprehensive introduction into the work with OpenScape Desktop.

1.3 Terminology

- **OpenScape FM** is the shortcut for OpenScape Fault Management.
- **Server** means the OpenScape FM Server, i.e. the server where OpenScape FM Desktop with the Enterprise MIB Plugin has been installed.
- **Client** means the OpenScape FM Client, usually a web browser where OpenScape FM have been started.
- **Desktop** means the OpenScape FM Desktop.
- **EPM** is the shortcut for Enterprise MIB.

1.4 Organization of this Guide

This guide is organized as follows:

- *Chapter 1, “Preface”* explains the structure of the manual.
- *Chapter 2, “Introduction”* provides a brief explanation what the Enterprise MIB Plugin is.
- *Chapter 3, “Getting started”* explains how to initialize the Enterprise MIB Plugin.
- *Chapter 4, “Enterprise MIB Plugin”* explains how to add Enterprise MIB definitions to the system and describes the different Enterprise MIB Plugin functions.
- *Chapter 5, “Enterprise MIB: MIB II”* provides an example of an Enterprise MIB Definition. It explains how the MIB II definition can be added to the system. In addition the functionality of the MIB II definition, which is a part of the Enterprise MIB Definition Plugin package, will be illustrated.

1.5 Conventions Used in this Manual

The following font conventions are used in this document:

Bold Font: Indicates that a word is a new or important term.

Example: **Proxy Agent** or **OK**.

Bold Computer Font: Indicates data to be entered by the user.

Example: **`java`**.

Computer Font: Indicates computer output, including UNIX prompts, an explicit directory or a file name.

Example: `prompt%`.

Italics: Indicates a reference to another manual or to a different section within the current manual.

Example: *see Manager User Guide.*

Italic type is also used for emphasis.

Example: *All* users will be affected.

2 Introduction

The OpenScape FM contains a number of Plugins to integrate HiPath specific technologies. These technologies will be integrated seamless into the OpenScape DT and they provide a wide array of HiPath specific functionality. The Enterprise MIB Plugin can be used, to embed non HiPath specific technologies (like e.g. Cisco) into OpenScape DT. It enables the integration of Enterprise SNMP MIBs into OpenScape DT. An integrated Enterprise SNMP MIB will then be able to provide the following functions:

- the auto discovery of devices, that are using the supported technology
- the visualization of all devices that were discovered and are using the supported technology, by representing them as an object in the OpenScape FM GUI
- easy access to the respective MIB values using SNMP
- *the display of events in the event browser when Enterprise MIB specific traps are received*
- *the visualization of the object status*

Not all of these functionality may be provided for every Enterprise MIB definition.
This user guide provides an overview about the basic functionality of this Plugin for all possible integrated Enterprise MIB definitions.

Hint:
In the HiPath environment the *APP-4K-MIB* is a special case. It is automatically connected to identified Frames, for example to connect MAR reactions with the Frames. More about MAR connections can be found in the separate *Mobile Alarm Reaction User Guide*.

3 Getting started

The OpenScape FM has to be installed on the system in order to be able to use the Enterprise MIB Plugin for OpenScape FM V9. More about the installation can be found in the *OpenScape FM Desktop User Guide*

During the installation the Enterprise MIB Plugin for OpenScape FM V9 will be initialized automatically.

The licencing is handled by using the System Management Plugin licenses.

3.1 Licensing

When the Enterprise MIB plugin gets initialized, a 90 day demo license will be granted. After this time period a valid licence will be needed to use the Enterprise MIB plugin.

For each IP node for which at least one Enterprise MIB object has been created, that is managed and does not represent an excluded MIB, one license is needed (see *Section 4.3, “Visualization of Enterprise SNMP MIBs”*). Excluded (not counting) MIBs are the MIB-II, the IPV6-MIB and the IPV6-ICMP-MIB.

In other words: for each device or system for which the manufacturer provides enterprise MIBs, and one or more of these MIBs will be activated, one Enterprise MIB IP node license will be needed.

If e.g. a 250 Enterprise MIB IP node license is available, on up to 250 systems Enterprise MIB objects for not excluded MIBs can be added and managed at the same time. This is independent on the total number of IP nodes and independent on the total number of MIBs discovered for the IP nodes.

Enterprise MIB objects for excluded MIBs can be added and managed on any number of IP nodes.

4 Enterprise MIB Plugin

The Enterprise MIB Plugin supports two formats to load MIB definitions into the OpenScape FM.

- **EPM Definition Files** are jar files that use an EPM specific format to describe MIBs including e.g. discovery rules, state and trap definitions.
- **ASN.1** is a standard to define MIBs. These are text files.

The following sections within this chapter describe how definition files of both formats can be loaded into the OpenScape FM, and how the data contained within the MIBs can be evaluated within the OpenScape FM.

With the EPM Editor an additional tool is provided that can load both formats and that can be used to generate even complex configurations. More about the EMP Editor can be found in a separate user guide.

4.1 Loading an Enterprise MIB

The Enterprise MIB Plugin allows the loading of MIB Definitions and of ASN.1 MIB files.

Files of both formats can be selected in a file browser that can be opened by using the entry **SNMP->Enterprise MIB->Load MIB...** in the main menu bar.

The file browser displays the file system of the OpenScape FM Client.

If a MIB Definition or an ASN.1 MIB file gets selected in the browser and the button Open is pressed, the OpenScape FM tries to load the respective file and to provide it for further actions within the OpenScape FM.

The file browser supports the selection of more than on file at the same time. If the loading of all selected MIBs was successful, a dialogue will open that displays all imported MIBs. If at least one load failed, an error message will be displayed and the file browser will not be closed.

If the checkbox **Activate MIB Definition** is checked, successfully loaded MIBs will be activated automatically (see *Section 4.2, “Activation/Deactivation of Enterprise MIB definitions”*).

During the initialization of the Enterprise MIB Plugin a number of frequently used MIBs will be loaded automatically. These MIBs will not be activated automatically.

4.1.1 Loading of MIB Definitions

If an Enterprise MIB definition is selected in the file browser (see *Section 4.1*), it will be checked whether the file uses the correct format. If the file is not compatible to the Enterprise MIB definition format, an error message will be displayed. The file browser will not be closed and another file can be selected. In such cases it may be necessary to contact the creator of the respective definition file.

If the file has the correct format, it will be uploaded and a symbol appears on the submap of the Enterprise MIB symbol. The Enterprise MIB symbol is located on the submap of the navigation tree entry **System->Plugin->SNMP**. The new symbol represents the uploaded Enterprise MIB and has the status ‚managed‘ if the MIB was activated, else it will be in the status ‚unmanaged‘.

4.1.2 Loading of ASN.1 MIB files

If an ASN.1 MIB file is selected in the file browser (see *Section 4.1*), it will be checked whether the file uses the correct format. If this is the case, the OpenScape FM generates an Enterprise MIB definition file. This file will be treated as if it has been loaded as described in *Section 4.1.1*.

In some cases a MIB definition encloses other MIB definition files. If an enclosed definition file is not known by the OpenScape FM, the generation of the Enterprise MIB definition file will be aborted and an error message will be displayed. An enclosed MIB is known, if it has been loaded before or if its definition file is located in the same directory as the enclosing file.

4.2 Activation/Deactivation of Enterprise MIB definitions

To get an overview of the uploaded Enterprise MIB definitions or to activate, deactivate or delete any number of definitions, the ’**SNMP->Enterprise MIB->MIB Definitions...**’ main menu item can be used to open the ’**MIB Definitions...**’ window.

The window contains the following elements:

- The **MIB Definitions** table holds a list of all Enterprise MIB definitions already uploaded to the system. Each row of the table represents one of these MIB definitions.
  - The column **Name** displays the name of the definitions.
  - The column **Active Version** indicates whether this is the `Original` MIB, or whether the MIB reactions have been `Modified`.
  - The column **Description** displays a short note about the definitions.
  - The column **Author** displays the name of the definition’s creator.
  - The column **Activated** contains either a checked box, when the definition is activated, or an unchecked box if the definition is still not activated.
- The **Activate** button is active when only MIB definitions are selected in the table that are not yet activated. Pressing the button will activate all MIB definitions that are currently selected in the table.
- The **Deactivate** button is active when only MIB definitions are selected in the table that are already activated. Pressing the button will deactivate all MIB definitions that are currently selected in the table. All settings done for the MIB definition will be deleted. Additionally all MIB objects, based on the selected MIB definition(s) will be deleted. But the MIB definition will still be available for later activations.
  A single object can also be deactivated by its object menu item **Edit->Unmanage**.
- The **Delete** button deactivates all MIB definitions currently selected in the table. Additionally the MIB definitions will be deleted.
  A single object can also be deleted by its object menu item **Edit->Delete**.
- The **Event Configuration** button opens a Trap Editor window for each selected MIB definition. These windows can be used to e.g. define severities and messages for the different defined traps. More about the Trap Editor can be found in *Section 4.4*.

- The button **Open In Editor** opens the MIB Definitions Editor (see separate *Enterprise MIB Definition Editor User Guide*) for the selected MIB definition. The button is only active, if a single definition is selected in the table.
- The button **Reset to Factory Settings** resets the configuration of all selected MIB definitions to the delivery state at the installation of the OpenScape FM Server

  Important Note:
  This means that all intermediate modifications of the selected MIB definitions are also deleted.
- The **Reload** button will update the contents of the window.
- The **Close** button will close the window.

When an Enterprise MIB definition gets activated, the respective symbol will change its state from ’unmanaged’ to ’normal’.

4.3 Visualization of Enterprise SNMP MIBs

For each discovered IP node the OpenScape FM system checks whether Enterprise MIB Definitions exist that are uploaded and activated, and that are supported by the IP node. For each of these Enterprise MIB Definitions an entry is added to the IP node’s view (submap and tree). The entry consists of an **EPM MIB object** which has a connected popup menu. The entry itself may contain two additional objects (’Basis Status’ and ’Events’) on its submap or subtree. These two objects are used in connection with the status display (see *Section 4.3.6*). The exact appearance of the icons/menus depends on the respective Enterprise MIB Definition. As an example see *Figure 1*.

The EPM MIB objects can be unmanaged to disable their monitoring for a specific IP node.

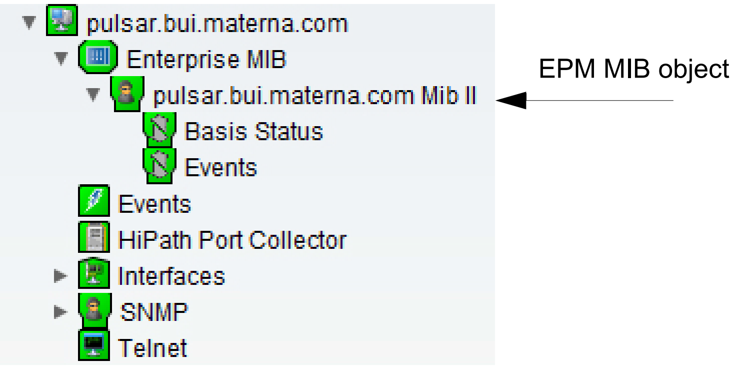


Figure 1 Example Enterprise MIB Definition entry

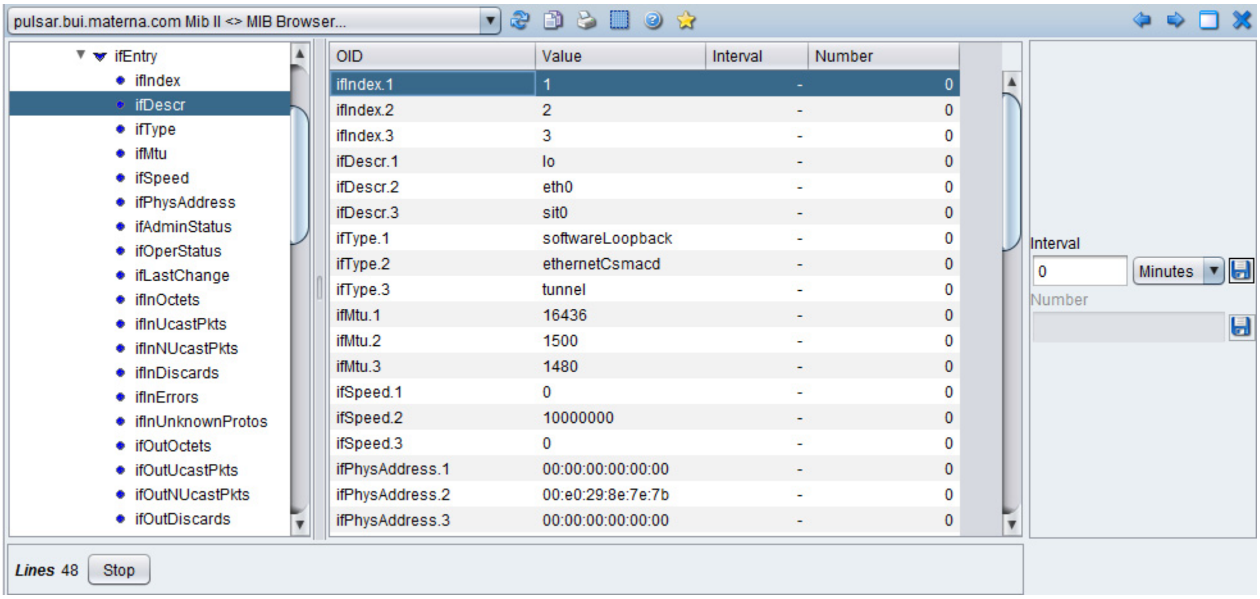
A number of basic types of information can be displayed, which will be explained in the following chapters:

- *Section 4.3.1, “MIB Browser”* shows how the whole MIB subtree covered by the Enterprise MIB Definition can be displayed. And how the MIB variable values can be automatically collected.
- *Section 4.3.2, “Values History”* describes how collected MIB variable values can be displayed.

- *Section 4.3.3, “Defining Threshold Values”* explains how thresholds for collected MIB variables can be defined and how these thresholds can be supervised.
- *Section 4.3.4, “Displaying all configured MIB Variable Entries”* describes how all supervised MIB variables can be displayed at a glance.
- *Section 4.3.5, “MIB Variables and Tables”* shows how variables covered by MIB tables can be displayed.
- *Section 4.3.6, “Status”* shows how the status provided by the MIB can be visualized.

4.3.1 MIB Browser

The **’MIB Browser...’ window** provides two major functions. It can be used to display and browse the MIB tree defined by an Enterprise MIB definition. Further it can be used to initiate the collection of MIB data. The ’MIB Browser...’ window (see *Figure 2*) is available for every Enterprise MIB Definition. It can be opened by activating the **’MIB Browser...’** context menu item of the EPM MIB object (see *Figure 1*), found under the IP node for which the MIB should be viewed.



Interval: 0

OID	Value	Interval	Number
ifIndex.1	1	-	0
ifIndex.2	2	-	0
ifIndex.3	3	-	0
ifDescr.1	lo	-	0
ifDescr.2	eth0	-	0
ifDescr.3	sit0	-	0
ifType.1	softwareLoopback	-	0
ifType.2	ethernetCsmacd	-	0
ifType.3	tunnel	-	0
ifMtu.1	16436	-	0
ifMtu.2	1500	-	0
ifMtu.3	1480	-	0
ifSpeed.1	0	-	0
ifSpeed.2	10000000	-	0
ifSpeed.3	0	-	0
ifPhysAddress.1	00:00:00:00:00:00	-	0
ifPhysAddress.2	00:e0:29:8e:7e:7b	-	0
ifPhysAddress.3	00:00:00:00:00:00	-	0

Figure 2 ’MIB Browser...’ window

The window consists of three major parts, which are explained below: A **tree**, a **list**, and a **configuration** area.

The left most part of the ‘MIB Browser...’ window consists of a **tree**. This tree represents the **structure of the MIB** for which the browser was opened. Every entry in the tree represents exactly one entry in the MIB and the entries in the tree are arranged (grouped into sub hierarchies) as they are arranged in the MIB. The sub hierarchies in the tree can be opened or closed by clicking on the pointer symbol which is on the left side next to the respective entry.

Each entry of the tree has a context menu, which consists of the following menu items:

- The menu item **Browse** will display the corresponding MIB entry in the **list** (see below). The value of the MIB entry itself and all entries found in its sub hierarchies (if available) will be displayed.
- The menu item **Configured** will execute the same action as the menu item **Browse**, but only entries which are configured for the MIB of the selected IP node (see below) will be displayed. Performing this action for the top level entry of a MIB will therefore display all entries that are configured for the selected MIB of the selected IP node. These entries are also the ones displayed when the window is initially opened for a MIB.

Since e.g. browsing a whole MIB can take longer than expected, a **Stop** button can be found below the tree. Pressing this button will stop the data discovering process immediately.

The **list** located in the middle section of the window is used to display the following information. It gets updated whenever the **Browse** or **Configured** menu item is activated for a MIB entry.

- The columns **OID** and **Value** display the OID of the variable displayed in the row and the value this MIB variable had when the window was last updated.
- The columns **Interval** and **Number** show the configured parameters for the corresponding MIB entry (see below) or '-' and '0' respectively when no parameters are configured for the entry.

When parameters are configured for a list entry, this entry has an context menu, which consists of the following menu items:

- The menu item **Values...** will open the 'Values...' window for the corresponding MIB entry (see *Section 4.3.2*). This window is used to display the variable values collected for the respective MIB entry.
- The menu item **Thresholds...** will open the 'Thresholds...' window for the corresponding MIB entry (see *Section 4.3.3*). This window can be used to define rules that affect the status of the respective EPM value collector object.

The right section of the window is used to **configure parameters** to sample the history of MIB entry variable values. These parameters define how often the value of a MIB entry shall be scanned, and how many of this values should be displayed.

- The textfield **Interval** and the associated time unit selector are used to define a time interval. This interval defines how often the value of a MIB entry will be scanned. When the 💾 button of the same row is pressed the time interval will be attached to all MIB entries currently selected in the list. When a defined time interval is configured for a MIB entry, the entry is treated as 'configured'. For each configured MIB entry one object will be added to the view of the respective EPM MIB object. This object will offer the same context menu as the list entry. Therefore values can be displayed and thresholds defined by selecting the icon's menu items.
  If a MIB entry should be removed from the list of configured entries, the Interval has to be set to '0'. This will remove the icon representing the MIB entry from the tree and stop the collection of values.
- The textfield **Number** is used to define how many scanned values should be remembered for a MIB entry. The 'Values...' window (see *Section 4.3.2*) can then be used to display these scanned values. When the 💾 button of the same row is pressed, the number defined in the textfield will be attached to all MIB entries currently selected in the list.

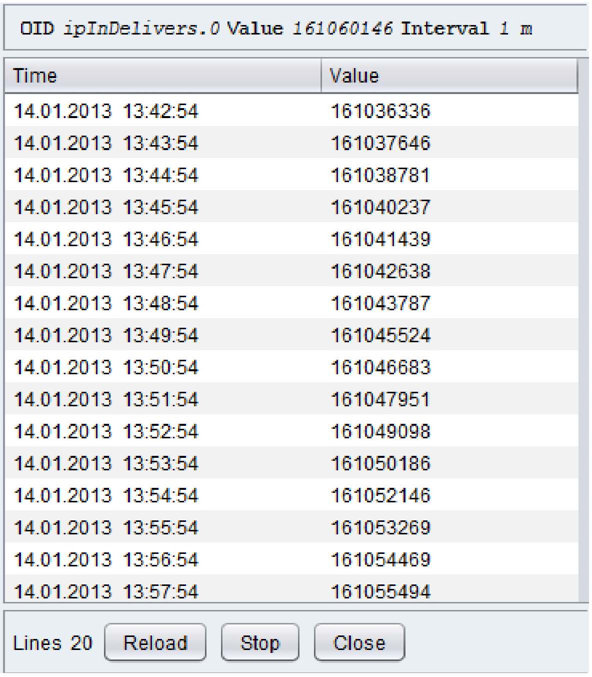
4.3.2 Values History

The values scanned for the configured MIB variable values can be displayed in the ‘Values...’ window. This window will be described in this chapter.

The window can be opened by activating the context menu item **Values...** of the value collector object or of a list entry which represents this object.

A list of all created value collections can be displayed be selecting the main menu entry **SNMP->Enterprise MIB->MIB Value Collectors**.

The ‘Values...’ window (see *Figure 3*) consists of the following elements providing the following functionality:



OID *ipInDelivers.0* Value *161060146* Interval *1 m*

Time	Value
14.01.2013 13:42:54	161036336
14.01.2013 13:43:54	161037646
14.01.2013 13:44:54	161038781
14.01.2013 13:45:54	161040237
14.01.2013 13:46:54	161041439
14.01.2013 13:47:54	161042638
14.01.2013 13:48:54	161043787
14.01.2013 13:49:54	161045524
14.01.2013 13:50:54	161046683
14.01.2013 13:51:54	161047951
14.01.2013 13:52:54	161049098
14.01.2013 13:53:54	161050186
14.01.2013 13:54:54	161052146
14.01.2013 13:55:54	161053269
14.01.2013 13:56:54	161054469
14.01.2013 13:57:54	161055494

Lines 20

Figure 3 ’Values...’ window

- The **table** displays the last values scanned for the related MIB entry, each row representing one scan. The table contains up to as many scans as were configured for the entry. Scans that are performed after the window is opened or refreshed will not be displayed automatically.
  The column **Time** displays the time when the respective scan displayed in the table row was performed. The column **Value** displays the result that was gathered by the respective scan.

- The **Reload** button will refresh the contents of the window when pressed. If new scans for the displayed MIB entry were performed since the window was opened, the results of these scans will then be displayed.

- The **Stop** button can be used to interrupt a currently active reload.

- The **Close** button can be used to simply close the window.

4.3.3 Defining Threshold Values

For each EPM value collector object (see *Figure 4*) the history of the MIB variable's values can be displayed (see *Section 4.3.2*). In addition it is possible to monitor the MIB variable. Whenever the variable reaches a predefined value or whenever it changes for a predefined amount in a given time interval, a status change to a configured state can be defined for the respective icon.

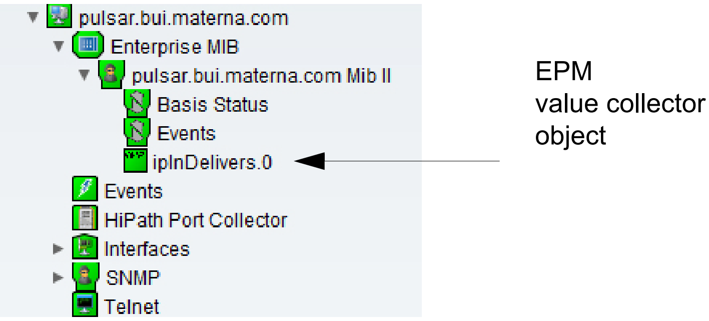


Figure 4 Tree with monitored EPM value collector

To define the conditions, for the status change of an EPM value collector object, thresholds can be defined. Each of these thresholds defines a relation that have to match to set the threshold active. When a threshold gets active the state connected to the threshold will be displayed by the EPM value collector object icon. In addition an event will be generated.

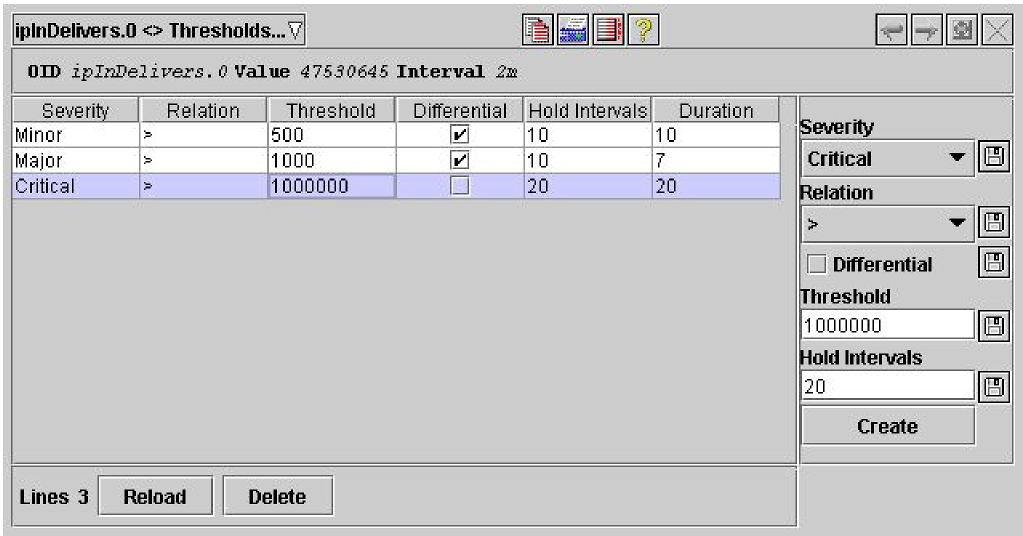
If more than one threshold matches for the same MIB entry and the same IP node at the same time the state of the respective EPM value collector object icon will be set to highest priority generated by the matching thresholds. Whenever the state of the EPM value collector object icon changes an event is created.

The 'Thresholds...' window (see *Figure 5* and *Figure 6*) is used to create, delete or reconfigure thresholds. The rest of this chapter will be used to describe the functionality of this window.

The 'Thresholds...' window can be opened by activating the context menu item **Thresholds...** of the value collector object or of a list entry which represents this object. If the value connected to the EPM value collector object is of type number the window explained in *Section 4.3.3.1* will be opened. If it is of type string the window described in *Section 4.3.3.2* will be opened.

4.3.3.1 Thresholds for Variables of the Type Number

For MIB entries with the variable type number the 'Thresholds...' window looks like *Figure 5*.



ipInDelivers.0 <> Thresholds...

OID *ipInDelivers.0* Value *47530645* Interval *2m*

Severity	Relation	Threshold	Differential	Hold Intervals	Duration
Minor	>	500	☑	10	10
Major	>	1000	☑	10	7
Critical	>	1000000	☐	20	20

Severity: Critical

Relation: >

☐ Differential

Threshold: 1000000

Hold Intervals: 20

Lines 3

Figure 5 The 'Thresholds...' window (Numbers)

The window is always opened for a single MIB entry. It basically consists of two parts:
A list representing the currently defined thresholds for the MIB entry.
And a number of buttons, textfields and selectors to create and reconfigure thresholds.

The list representing the already defined thresholds contains one row per threshold.
The first five columns (**'Severity'**, **'Relation'**, **'Differential'**, **'Threshold'** and **'Hold Intervals'**) contain the values configured for the threshold. More about these values can be found below.
The last value **'Duration'** has nothing to do with the configuration itself, and therefore cannot be set actively. When a threshold gets active, this value will be set to the value configured in 'Hold Values'. Whenever a scan is performed, getting a result that would not have activated the threshold, this value will be reduced by one. When the 'Duration' reaches '0' the threshold gets deactivated. Whenever a scan provides a result, that would activate the threshold, the value will be reset to the one defined in 'Hold Values'.
In other words: A threshold will get active, when the assigned relation matches. It will get inactive, when the relation does not match for a number of consecutive scans. This number is displayed in „Duration".

More than one entry of the list can be selected, to modify the threshold parameters (see below). If only a single list entry gets selected, its currently configured settings will be copied to the configuration textfields/selectors on the right hand side of the window.

When the **Reload** button is pressed, the contents of the list will be refreshed.

The following window elements can be used to create or reconfigure thresholds:

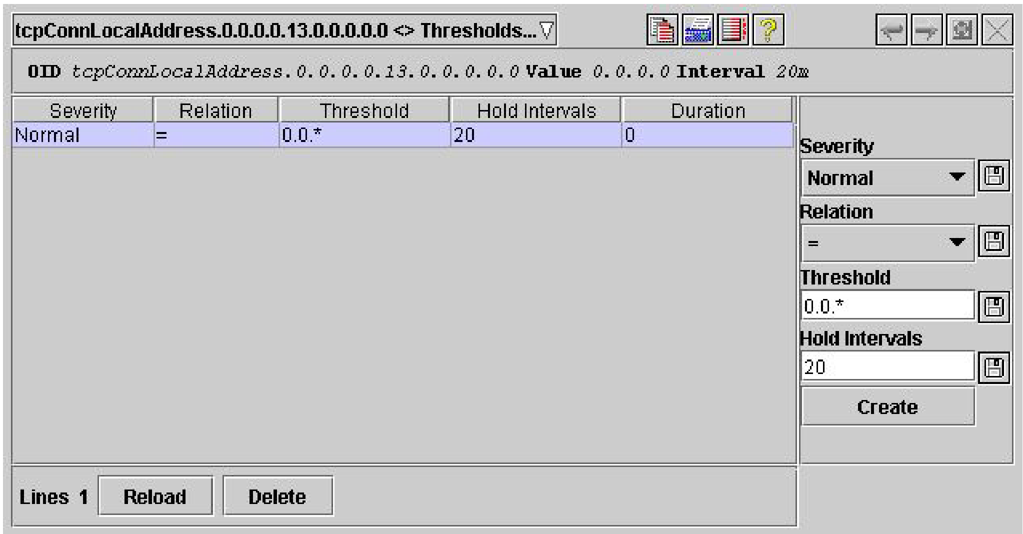
- The **Severity** selector can be used to select the severity which should be connected to a threshold. The severity is identical to the state in which the icon connected to the EPM value collector object will be displayed, when the threshold gets active.
- The **Relation** selector is used to configure the relational operator that has to match between the value defined in the **Threshold** textfield and the actual value of the MIB entry. A successful match activates the respective threshold.

- The **Differential** checkbox is used to define the type of check that should be performed. When the checkbox is not checked, the check defined by the **Relation** selector and the **Threshold** textfield will be performed against the current value of the connected MIB entry. If the checkbox is checked, the check will be made against changes of the connected MIB entry’s value. The system will perform the check against the result of the last scan minus the result of the current scan. Using the latter type makes it possible to change the state of the connected icon, when a number of events happened within a given time.
- The **Threshold** textfield is used to define the value against which the connected MIB variable should be checked. This field is used in connection with the **Relation** selector and **Differential** checkbox to create the expression that defines the check.
- The **Hold Intervals** textfield is used to define the number of consecutive scans for which a threshold stays active even when the relation no longer matches. For this time also no events will be generated for this threshold.
- When the **Create** button is pressed a new threshold definition for the actual MIB entry will be created. The new threshold will be initialized with the parameters currently selected by the selectors and textfields.
- When the **Delete** button is pressed all thresholds currently selected in the **list** will be removed.
- The 💾 buttons attached to the selectors and textfields can be used to reconfigure already defined thresholds. When one of these buttons is pressed the value/selection attached to the respective 💾 button will be copied into all thresholds currently selected in the **list**.

Example:
For the thresholds displayed in *Figure 5* the status will change to minor/major when the value increases for more than 500/1000 between two scans within in this case (as shown in the heading line), 2 minutes. The related thresholds will stay active for at least 10 scans. When the value gets larger than the absolute value of 1.000.000 the state will change to critical and will stay at that level for at least 20 scans. The column ’Duration’ shows for all thresholds for how many scans they will be at least kept active.

4.3.3.2 Thresholds for Variables of the Type String

For MIB entries with the variable type string the ‘Thresholds...’ window looks like *Figure 6*


tcpConnLocalAddress.0.0.0.0.13.0.0.0.0.0 <> Thresholds...

OID *tcpConnLocalAddress.0.0.0.0.13.0.0.0.0.0* **Value** *0.0.0.0* **Interval** *20m*

Severity	Relation	Threshold	Hold Intervals	Duration
Normal	=	0.0.*	20	0

Severity: Normal
Relation: =
Threshold: 0.0.*
Hold Intervals: 20

Lines 1

Figure 6 The 'Thresholds...' window (Strings)

The functionality of this window is virtually identical to the window for numbers described in *Section 4.3.3.1*. The differences are the missing column and checkbox **Differential** and the types of relations that can be selected by the **Relation** selector. Next to the relational operators **=** (equal) and **!=** (not equal), simple or regular expressions can be defined that are matched against the variables value string.

Simple expressions (**SimpleRegExp**) are expressions that may contain text and wild cards only. The '?' character is used to represent a single character, the '*' character is used to represent any number of characters (none included).
For Example: The expression `'ABC?E:1*3'` will match for `'ABCDE:123'` or `'ABCXE:13'` or `'ABC1E:1aaaa3'` but not for `'ABCE:123'`.

Since Regular Expressions (**RegExp**) are a very complex subject, we cannot provide an exhaustive explanation at this point. However, the following table contains examples which cover the most common usages.

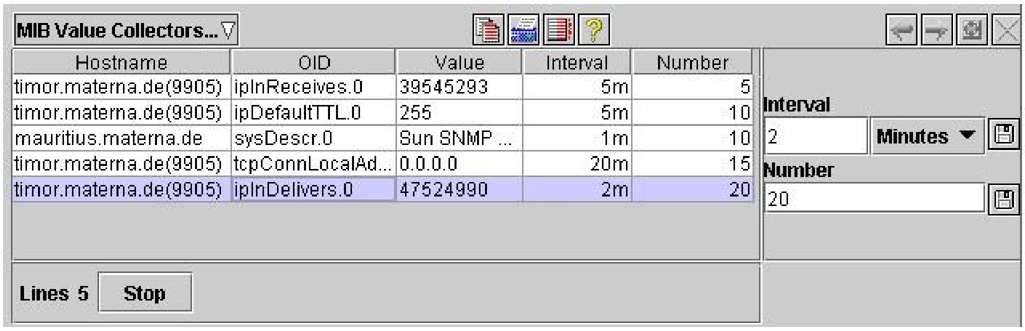
Regular Expressions: examples for search patterns:

<table>
<tr><th></th><th></th></tr>
<tr><td>.</td><td>Any character (equivalent to the ? of simple expressions)</td></tr>
<tr><td>.*</td><td>Any number of characters (none included) (equivalent to the * of simple expressions)</td></tr>
<tr><td>.*User.*</td><td>A pattern containing the substring “User”</td></tr>
<tr><td>[Ss].*</td><td>Any series of characters which starts with a small or capital s; i.e. “Smith”, “smith”, or “salmon”</td></tr>
<tr><td>[0-9]+</td><td>Any number of digits (but at least one)</td></tr>
<tr><td>[^0-9].*</td><td>Any pattern that does NOT start with a digit</td></tr>
<tr><td>(One|Two)</td><td>The pattern “One” or the pattern “Two”</td></tr>
<tr><td>.*\.sun\.com.*</td><td>A pattern that contains the substring “.sun.com”. Since the dot has a special meaning in Regular Expressions (one arbitrary character), it has to be masked by a backslash.</td></tr>
</table>

4.3.4 Displaying all configured MIB Variable Entries

Besides displaying all MIB entries of a single MIB which are configured for the collection of their values (as described in *Section 4.3.1*), it is possible to display all MIB entries configured for collection at a single glance.

The ‘MIB Value Collectors...’ window (see *Figure 7*) is used to display these entries. It can be opened by selecting the **SNMP->Enterprise MIB->MIB Value Collectors...** menu item from the main menu bar.



Hostname	OID	Value	Interval	Number
timor.materna.de(9905)	ipInReceives.0	39545293	5m	5
timor.materna.de(9905)	ipDefaultTTL.0	255	5m	10
mauritius.materna.de	sysDescr.0	Sun SNMP ...	1m	10
timor.materna.de(9905)	tcpConnLocalAd...	0.0.0.0	20m	15
timor.materna.de(9905)	ipInDelivers.0	47524990	2m	20

Interval
2 Minutes

Number
20

Lines 5

Figure 7 The ‘MIB Value Collectors...’ window

The look and the functionality of this window is very close to the one of the ‘MIB Browser...’ window (see *Figure 2*).

This window is missing the MIB tree section. Instead, the table is expanded by the column **Hostname**. This column holds the name of the IP node to which the respective EPM value collector object belongs to

4.3.5 MIB Variables and Tables

In MIBs, generally two types of data can be found.
The first type are **variables** which are identifier/value pairs that stand for themselves like the number of incoming IP datagrams.
The second type are groups of variables that are linked together to form a connected set of attributes/values, called **tables**. In contrast to variables, tables may have more than one entry per attribute, where each entry is part of a connected set. An example for a table is the IP address table. In this table, every entry consists of five attributes (IP address, Index, NetMask, Least significant bit, re-assembly max size). Here, every set of attributes describes the data for a single IP address.

Due to the different nature of these two types, entries are displayed by either the ’MIB Variables’ window (see *Section 4.3.5.1, “Displaying and changing MIB Variables”*) or the ’MIB Table’ window (see *Section 4.3.5.2, “Displaying and changing MIB Tables”*). Both windows are opened by selecting one of the defined menu items connected to the EPM MIB object. How MIB menu items, and the data that will be presented when they are selected, will be displayed, is defined in the EPM definition file. The menu items connected to an Enterprise MIB definition are part of the context menu of the EPM MIB object for an IP node.

As an example *Figure 8* shows a part of the menu items that are created by the MIB II definition.

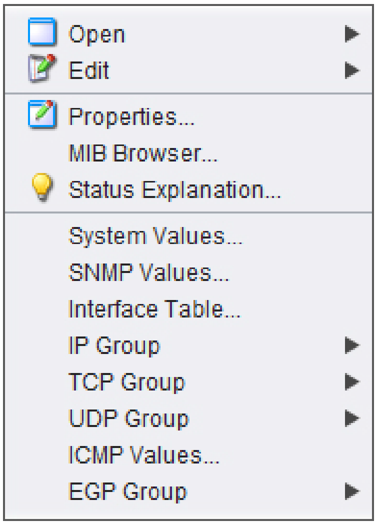
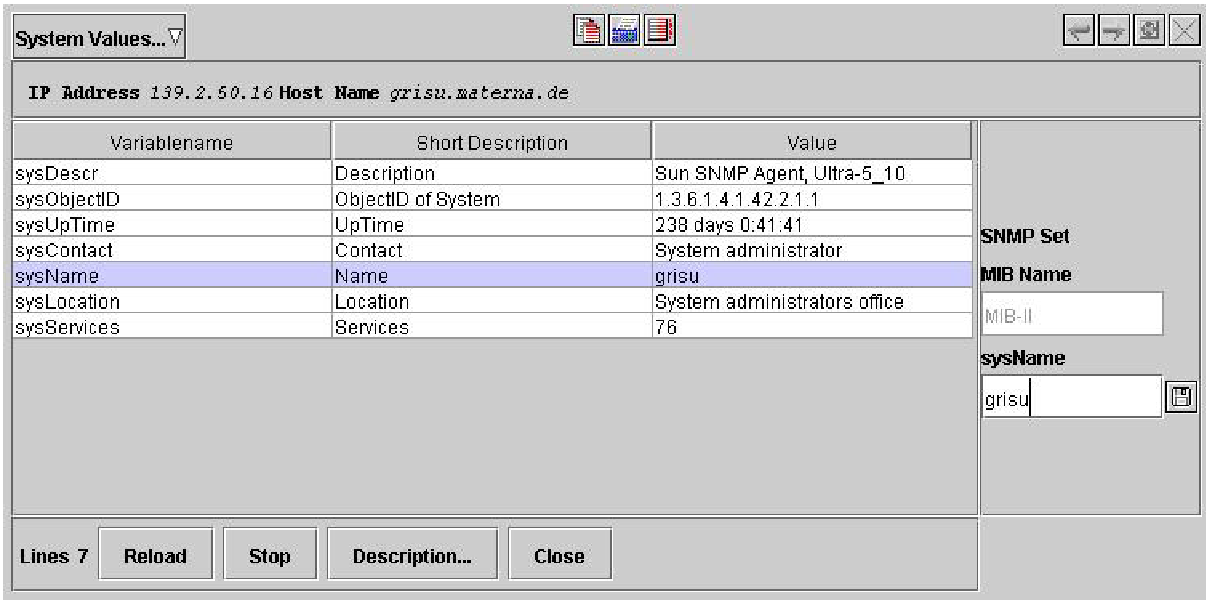


Figure 8 *MIB Enterprise Defined Menus (MIB II)*

4.3.5.1 Displaying and changing MIB Variables

When **MIB variables** should be displayed or possibly changed the 'MIB Variables' window is used (see *Figure 9*).

Basically, this window displays the information that was retrieved for a set of given MIB entry variables. In the case of a write access the selected MIB entry can be changed. This implies a support from the SNMP agent.



IP Address *139.2.50.16* Host Name *grisu.materna.de*

Variablename	Short Description	Value
sysDescr	Description	Sun SNMP Agent, Ultra-5_10
sysObjectID	ObjectID of System	1.3.6.1.4.1.42.2.1.1
sysUpTime	UpTime	238 days 0:41:41
sysContact	Contact	System administrator
sysName	Name	grisu
sysLocation	Location	System administrators office
sysServices	Services	76

SNMP Set
MIB Name
MIB-II
sysName
grisu

Lines 7

Figure 9 *The 'MIB Variables' window*

The window consists of a **table**, a panel with an **SNMP Set** functionality and four buttons which provide the following functionality:

- The **table** is used to display the values of variables as they are currently set in the MIB database. Each row of the table displays the information and value of a single variable. The table consists of the following columns: The column **Variablename** displays the name of the variable as it is used in the MIB database, the column **Short Description** displays a short description about the meaning of the variable and the column **Value** displays the value as it was set in the MIB database when the window was opened or when the **Reload** button was pressed the last time.

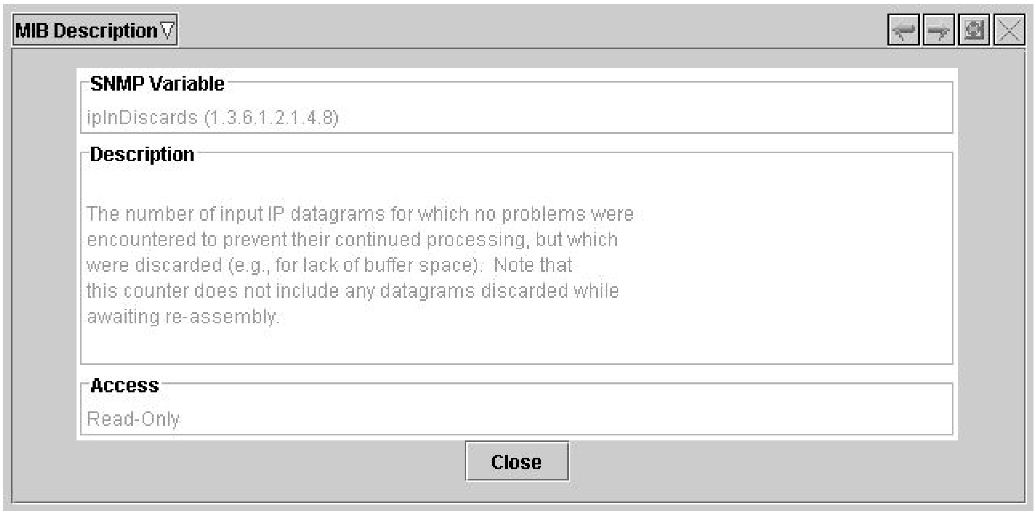
- The **Reload** button can be used to refresh the data displayed by the table. The column **Value** will be filled with the current data when the button is pressed.

- The **Stop** button can be pressed to stop the discovery of the actual data. This is useful when e.g. network problems occur after the window was initially opened or the **Reload** button was pressed.

- The **Close** button will simply close the window when pressed.

- The **Description...** button can be used to display descriptive information about the selected variable as it is defined in the MIB. This button is only active when a single table entry is selected. When the button is pressed, the ’MIB Variable Description’ window (see *Figure 10*) will be opened for the selected variable.

- The **SNMP Set** panel is only visible if at least one of the MIB variables has write access. The panel consists of a static text field and a text field or drop down list with a 💾 button. The static text field with the label **MIB Name** displays the name of the currently used MIB. This text field is not editable. The second text field or the drop down list is only visible when a single table entry is selected and the variable has write access (see **Access** in the MIB Variable Description window, *Figure 10*). The label of the text field or drop down box is the variable name of the selected table entry. The variable value can be changed and is written back to the SNMP agent by pressing the 💾 button.



SNMP Variable
ipInDiscards (1.3.6.1.2.1.4.8)

Description
The number of input IP datagrams for which no problems were
encountered to prevent their continued processing, but which
were discarded (e.g., for lack of buffer space). Note that
this counter does not include any datagrams discarded while
awaiting re-assembly.

Access
Read-Only

Figure 10 *The ’MIB Variable Description’ window*

The ’MIB Variable Description’ window consists of the following components:

- The **SNMP Variable** textfield displays the name of the variable as it is used in the MIB database. In addition, it displays the MIB Id used by the variable.

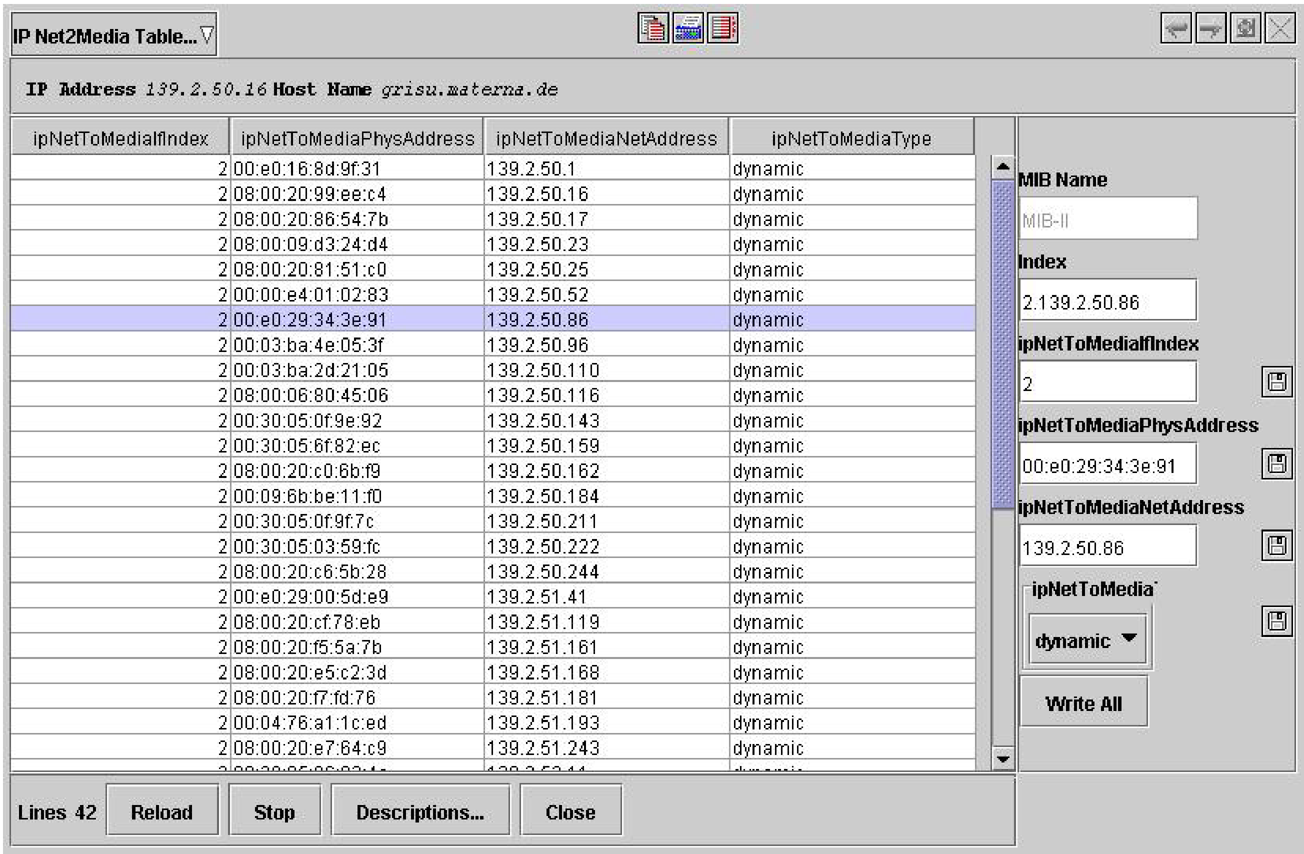
- The **Description** textfield displays the descriptive text connected to the variable as defined in the MIB database.

- The **Access** textfield displays the access type defined for the variable in the MIB database.
- The **Close** button will simply close the window.

4.3.5.2 Displaying and changing MIB Tables

In order to display **MIB tables,** the ’MIB Table’ window is used (see *Figure 11*). This window also provides the possibility to change variable values in a table entry if at least one variable of the MIB table has write access. This implies support by the SNMP agent.

Basically, this window displays the information that was retrieved for all defined MIB variables of an SNMP table. All these MIB variables must be contained within the same SNMP table.



IP Address *139.2.50.16* Host Name *grisu.materna.de*

ipNetToMediaIfIndex	ipNetToMediaPhysAddress	ipNetToMediaNetAddress	ipNetToMediaType
2	00:e0:16:8d:9f:31	139.2.50.1	dynamic
2	08:00:20:99:ee:c4	139.2.50.16	dynamic
2	08:00:20:86:54:7b	139.2.50.17	dynamic
2	08:00:09:d3:24:d4	139.2.50.23	dynamic
2	08:00:20:81:51:c0	139.2.50.25	dynamic
2	00:00:e4:01:02:83	139.2.50.52	dynamic
2	00:e0:29:34:3e:91	139.2.50.86	dynamic
2	00:03:ba:4e:05:3f	139.2.50.96	dynamic
2	00:03:ba:2d:21:05	139.2.50.110	dynamic
2	08:00:06:80:45:06	139.2.50.116	dynamic
2	00:30:05:0f:9e:92	139.2.50.143	dynamic
2	00:30:05:6f:82:ec	139.2.50.159	dynamic
2	08:00:20:c0:6b:f9	139.2.50.162	dynamic
2	00:09:6b:be:11:f0	139.2.50.184	dynamic
2	00:30:05:0f:9f:7c	139.2.50.211	dynamic
2	00:30:05:03:59:fc	139.2.50.222	dynamic
2	08:00:20:c6:5b:28	139.2.50.244	dynamic
2	00:e0:29:00:5d:e9	139.2.51.41	dynamic
2	08:00:20:cf:78:eb	139.2.51.119	dynamic
2	08:00:20:f5:5a:7b	139.2.51.161	dynamic
2	08:00:20:e5:c2:3d	139.2.51.168	dynamic
2	08:00:20:f7:fd:76	139.2.51.181	dynamic
2	00:04:76:a1:1c:ed	139.2.51.193	dynamic
2	08:00:20:e7:64:c9	139.2.51.243	dynamic

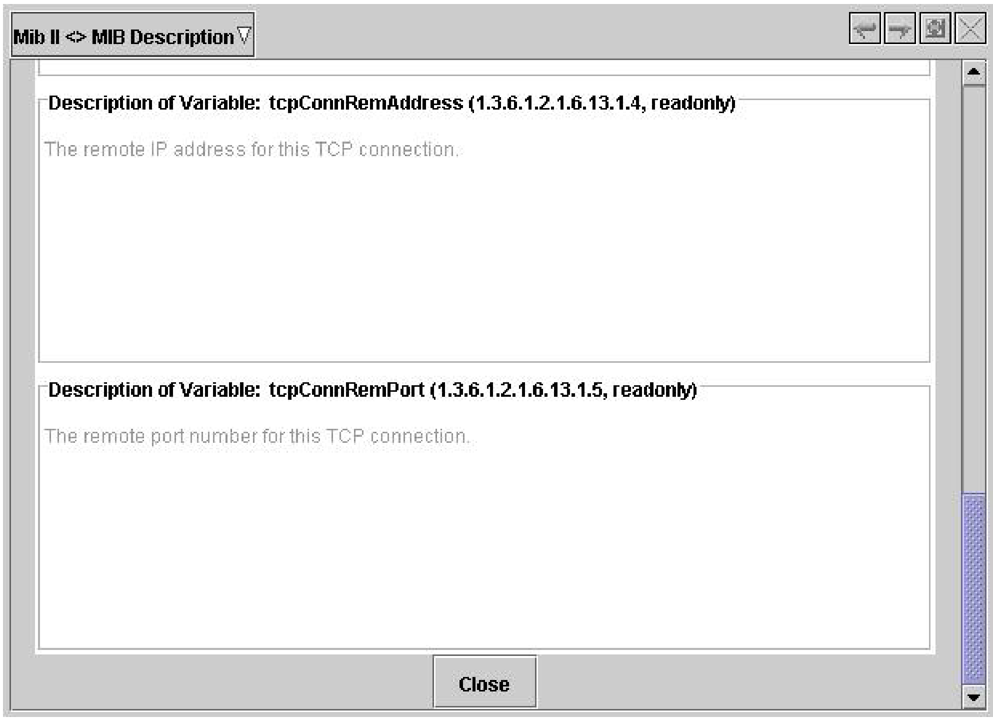
MIB Name: MIB-II
Index: 2.139.2.50.86
ipNetToMediaIfIndex: 2
ipNetToMediaPhysAddress: 00:e0:29:34:3e:91
ipNetToMediaNetAddress: 139.2.50.86
ipNetToMedia: dynamic

Figure 11 The ’MIB Table’ window

The window consists of a table, a panel with SNMP Set functionalities and four buttons which provide the following functionality:

- The **table** is used to display the values of the variable sets currently entered in the MIB database. Each **row** represents one set of variables or one table entry. Each **column** represents one variable in the table. When the mouse pointer is positioned onto one of the column headers, a tool tip text will be displayed which provides a short description of the variable displayed in the column. The column header itself displays the name of the variable as it is defined in the MIB database.

- The **Reload** button can be used to refresh the data displayed by the table. The rows will be filled with the current data when the button is pressed.
- The **Stop** button can be pressed to stop the discovery of the actual data. This is useful when e.g. network problems occur after the window was initially opened or the **Reload** button was pressed.
- The **Descriptions...** button can be used to display descriptive information of all variables defined in the MIB table. This button is only active when a single table entry is selected. When the button is pressed, the ’MIB Variable Descriptions’ window (see *Figure 12*) will be opened for the actual MIB table.
- The **Close** button will simply close the window when pressed.
- The panel with the SNMP set functionalities is only visible if at least one of the MIB table variables has write access. The panel consists of text fields or drop down lists with a 💾 button and finally a **Write All** button. The first text field with the label **MIB Name** always contains the currently used MIB name and is not editable. The second text field with the label **Index** contains the value of the currently selected index variable. The index variable text field is displayed without a 💾 button.The other text fields or drop down lists contain each a variable value with write access. The label is always the name of the variable. A variable value can be changed in the appropriate text field or drop down list and is written back to the SNMP agent by pressing the 💾 button on the right.
The **Write All** button can be used to set all values at once for the given index value in the **Index** text field.



Description of Variable: tcpConnRemAddress (1.3.6.1.2.1.6.13.1.4, readonly)

The remote IP address for this TCP connection.

Description of Variable: tcpConnRemPort (1.3.6.1.2.1.6.13.1.5, readonly)

The remote port number for this TCP connection.

Figure 12 The ’MIB Variable Descriptions’ window for the variables of a MIB table

The ’MIB Variable Descriptions’ window consists of a list containing the descriptions of all the MIB table variables. The header of each list entry contains the SNMP variable name and in brackets the MIB Id and the access modifier.
The **Close** button will simply close the window.

4.3.6 Status

If a license for OpenScape is available the Enterprise MIB Plugin provides different methods to connect status information to the different discovered EPM MIB objects.

The following chapters will introduce these methods.

4.3.6.1 Status using the Basis Status

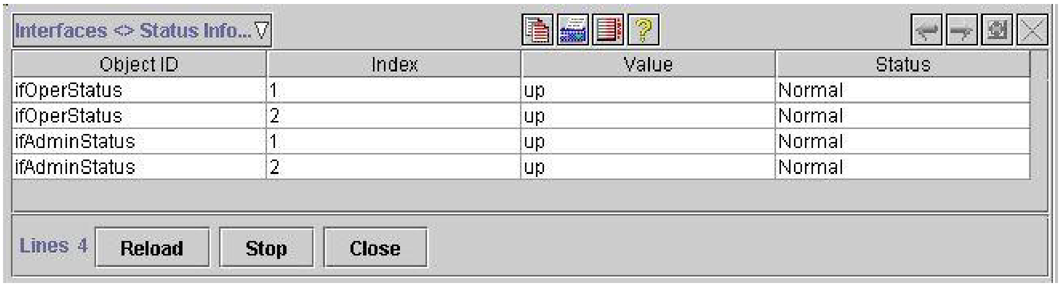
When a configured Enterprise MIB is discovered for an IP node an EPM MIB object will be added to the view of the IP node. The view of this object contains a status object called **Basis Status**. During the regular status polls, it will be established whether the SNMP agent is running on the IP node. If it is running, it will be checked whether the Enterprise MIB can be discovered. The state of the Basis Status object will change to critical when the MIB cannot be discovered. It turns to unknown, when the SNMP agent is down.

4.3.6.2 Status using EPM MIB variables

When this method is used, the status of the EPM MIB object will be determined by evaluating one or more EPM MIB variables that are defined in the Enterprise MIB definition file. The value of these defined EPM MIB variables will be checked whenever a regular status poll is conducted. The status of the EPM MIB object will then be determined by comparing their values to the status mapping also defined in the Enterprise MIB definition file.

For each group of variables that should be monitored, a status object is added under the MIB Definition symbol. The status object will have the label defined in the Enterprise MIB definition file and its status will represent the status determined by the value of the connected variables.

In addition, the status icon will provide a context menu that offers the menu item **Status Info...**. When this menu item is activated, the *Status Info...* window (see *Figure 13*) will be opened for the variables connected to the status icon.



Object ID	Index	Value	Status
ifOperStatus	1	up	Normal
ifOperStatus	2	up	Normal
ifAdminStatus	1	up	Normal
ifAdminStatus	2	up	Normal

Figure 13 The ’Status Info...’ window

The *Status Info...* window consists of a table and three buttons:

- The **table** displays the variables that are connected to the status icon from which the window was opened. The columns **Object ID** and **Index** identify the respective variable. The index is of interest if the variable is part of a table and therefore may exist in more than one instance.
  The value of the connected variables will be displayed in the column **Value**.
  The column **Status** displays the status that results from the value.
- The **Reload** button will refresh the content of the window, bringing the status of the different variables to the current value as it is found in the MIB database.
- The **Stop** button can be used to interrupt an active value collection.
- The **Close** button will simply close the window.

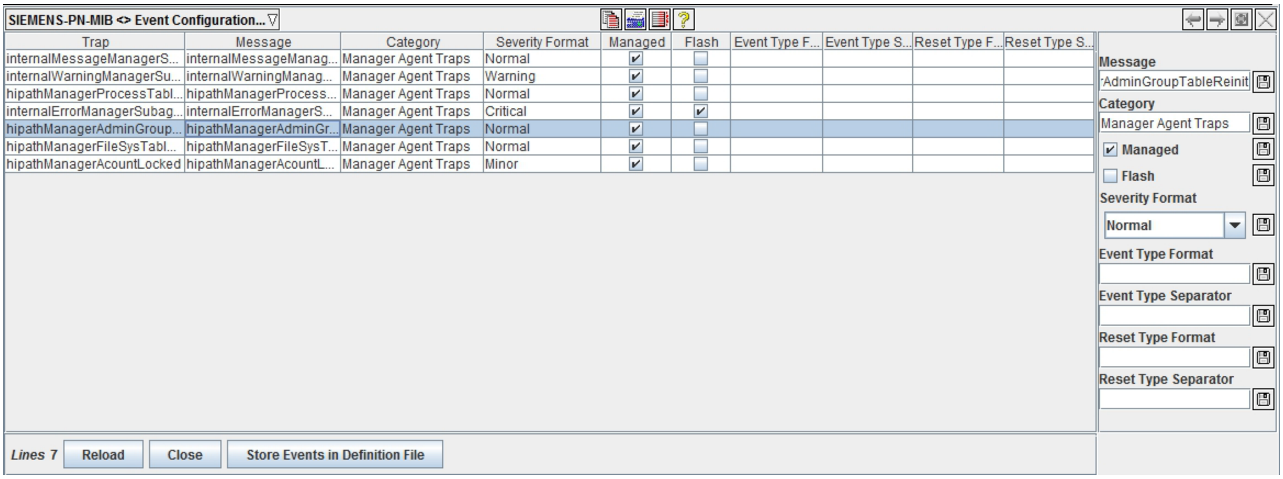
4.3.6.3 Status using Not Acknowledged Events

While creating an Enterprise MIB definition, it is possible to configure the display of traps if the MIB contains any. When these traps are received by the OpenScape FM server, events will be created for these traps. The events will be shown in the Event Browser.

The view of each EPM MIB object contains a status object called **Events**. The context menu item **Object->Events** of this object can be used to open the Event Browser displaying all events received from the corresponding MIB. In addition, the **Events** object will always display the status that is equivalent to the most critical status of all events within the Event Browser that are related to the corresponding MIB and that are unacknowledged. If no such event exist, the status of the **Events** object will be set to *Normal*.

4.4 Trap Editor

The Trap Editor (see *Figure 14*) can be used to define how traps should be handled and displayed in the Event Browser. Each Trap Editor is opened for a single MIB definition by pressing the button **Event Configuration** within the MIB Definitions window (see *Section 4.2*).



SIEMENS-PN-MIB <> Event Configuration...

Trap	Message	Category	Severity Format	Managed	Flash	Event Type F...	Event Type S...	Reset Type F...	Reset Type S...
internalMessageManagerS...	internalMessageManag...	Manager Agent Traps	Normal	☑	□				
internalWarningManagerSu...	internalWarningManag...	Manager Agent Traps	Warning	☑	□				
hipathManagerProcessTabl...	hipathManagerProcess...	Manager Agent Traps	Normal	☑	□				
internalErrorManagerSubag...	internalErrorManagerS...	Manager Agent Traps	Critical	☑	☑				
hipathManagerAdminGroup...	hipathManagerAdminGr...	Manager Agent Traps	Normal	☑	□				
hipathManagerFileSysTabl...	hipathManagerFileSysT...	Manager Agent Traps	Normal	☑	□				
hipathManagerAcountLocked	hipathManagerAcountL...	Manager Agent Traps	Minor	☑	□				

Message: rAdminGroupTableReinit
Category: Manager Agent Traps
☑ Managed
□ Flash
Severity Format: Normal
Event Type Format:
Event Type Separator:
Reset Type Format:
Reset Type Separator:

Figure 14 The 'Trap Editor' window

The Trap Editor window contains a table where each row represents one trap defined in the corresponding definition file. The column **Trap** contains the names of the different traps. All other entries can be reconfigured by using the respective fields, checkboxes and menus on the right hand side of the window.

All text fields may include macros which will be replaced when a trap is received. Macros start with a dollar symbol (’$’) followed by a string defining the content. The possible macros are shown in the following table.

`$1` - `$99`	value of the respective trap variable
`$#`	number of the variables
`$*`	all trap variables
`$@`	the time in seconds when the event was generated
`$T`	the system time of the SNMP agent system, when the trap was sent
`$x`	the time as a formatted character string, when the event was generated
`$R` or `$r`	the source IP
`$e`	the Enterprise OID of the event
`$h`, `$H`	the hostname / fully qualified hostname, or when it is not known, the IP adress of the event’s source

Table 1 Macros

The field **Message** configures the definition of the string that will be displayed in the column Description within the Event Browser. The column **Category** defines the content of the Event Browser column of the same name.

The field **Severity Format** configures the severity of the event which is displayed in the Event Browser’s column Severity. Here one of the constant values from the selection menu (e.g. ’Normal’, ’Critical’) can be chosen. If a ressource key has been defined within the MIB Editor it will be displayed here.

The checkbox **Managed** defines whether the respective trap should be displayed or not.

The checkbox **Flash** defines whether the affected object should flash in the OpenScape FM, if a respective unacknowledged event exists within the Event Browser.

The fields **Event Type Format**, **Event Type Separator**, **Reset Type Format** and **Reset Type Separator** are used to generate meta data for the automatic acknowledgement of matching traps (e.g. an Up-trap should acknowledge a matching Down-trap).

The field **Event Type Format** defines the string that is used to generate a key to identify the trap. If more than one key should be attached to the trap, the field **Event Type Separator** can be used to define a symbol that divides the string from the Event Type Format fields into more than one key string. The keys will be attached to the matching event within the Event Browser.

Example:

A trap is received and the trap variables $3, $4 and $5 contain the values ’100’, ’15’ and ’36’.

If the Event Type Format is defined as ’XTrap $3 $4’ and no separator was defined, the key for this trap will be: ’XTrap 100 15’

If the Event Type Format is defined as ’XTrap $3 $4&XTrap $3 $5’ and the separator ’&’ was defined, the keys for this trap will be: ’XTrap 100 15’ and ’XTrap 100 36’.

The fields **Reset Type Format** and **Reset Type Separator** are used in the same way to define one or more reset keys for an incoming trap. If reset keys are attached to an incoming trap, all events in the Event Browser that have at least one matching attached key will be automatically acknowledged.

Example:

> For the trap `Link-Down` the **Event Type Format** `$r:Down:$1` has been defined.
>
> For the trap `Link-Up` the **Reset Type Format** `$r:Down:$1` has been defined.
>
> If a `Link-Down` trap is received for which `$r` will be replaced by `192.168.1.1` and `$1` by `4`, the key `192.168.1.1:Down:4` will be attached to the event generated by the trap.
>
> If at a later time a `Link-Up` trap is received for which `$r` is also replaced by `192.168.1.1` and `$1` also by `4`, then the chosen Reset Type Format generates an identical reset key and the `Link-Down` event of the first trap will be acknowledged.

Pressing the buttons below the table will **Reload** the content of the table, **Close** the window or will save the configuration (**Store Events in Definition File**).

4.5 MIB Editor

The Enterprise MIB definition editor can be opened from within the OpenScape FM by using the main menu entry ‚**SNMP->Enterprise MIB->MIB Defintions**‘ to open the ‚MIB Definitions‘ window. After this, the button **Open** in Editor can be used to open the Enterprise MIB definitions editor (see separate User Guide) for a selected MIB definition. The button is only active, if a single definition is selected within the table.

5 Enterprise MIB: MIB II

5.1 Introduction

Since each upload able Enterprise MIB Definition is designed for a different technology, the functionality provided by different definitions also differs. To provide a better understanding, what functionality can be added to the OpenScape FM by activating an Enterprise MIB definition, this chapter will contain an example that will add the MIB II definition to the system. The MIB II definition is part of the Enterprise MIB Definition Plugin distribution.

5.2 Installation

Generally the main menu bar menu item **SNMP->Enterprise MIB->Load MIB Definition...** has to be selected to upload the definition file for a new Enterprise MIB type. When this menu item is selected a window containing a file selector browser will be opened. In this browser a MIB Definition file can be selected and loaded. Since the upload of the MIB II MIB Definition file is part of the installation of the Enterprise MIB Plugin, an upload is not necessary in the special case of this example.

When a MIB Definition file is successfully loaded into the system it is still not active and ip nodes are still not checked for the availability of the respective MIB. To activate a MIB Definition the **'MIB Definitions...'** window has to be opened. This can be done by selecting the **SNMP->Enterprise MIB->MIB Definitions...** menu item of the main menu bar.

5.3 Functionality

This chapter will provide a short overview of the functionality provided by the MIB II Enterprise MIB Definition.

The following popup menu items will be found for the MIB II EPM MIB objects, and they will provide the following functionality when activated:

- **'MIB Browser...'**: This will open the MIB browser (see *Section 4.3.1, "MIB Browser"*) with the MIB II entry as the root node. Using the browser, all variables and tables of the MIB II subtree can be displayed or monitored.
- **'System Values...'**: This will display the variables found in the '.iso.org.dod.internet.mgmt.mib-2.system' table. These variables are: `sysDescr`, `sysObjectID`, `sysUpTime`, `sysContact`, `sysName`, `sysLocation`, `sysServices`.
- **'SNMP Values...'**: This will display the variables found in the '.iso.org.dod.internet.mgmt.mib-2.snmp' table. These variables are: `snmpInPkts`, `snmpOutPkts`, `snmpInBadVersions`, `snmpInBadCommunityNames`, `snmpInBadCommunityUses`, `snmpInASNParseErrs`, `snmpInTooBigs`, `snmpInNoSuchNames`, `snmpInBadValues`, `snmpInReadOnlys`, `snmpInGenErrs`, `snmpInTotalReqVars`, `snmpInTotalSetVars`, `snmpInGetRequests`, `snmpInGetNexts`, `snmpInSetRequests`,

snmpInGetResponses, snmpInTraps, snmpOutTooBigs, snmpOutNoSuchNames, snmpOutBadValues, snmpOutGenErrs, snmpOutGetRequests, snmpOutGetNexts, snmpOutSetRequests, snmpOutGetResponses, snmpOutTraps, snmpEnableAuthenTraps.

- **’Interface Table’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry’ table. For each entry one table row will be displayed containing these variables: ifIndex, ifDescr, ifType, ifMtu, ifSpeed, ifPhysAddress, ifAdminStatus, ifOperStatus, ifLastChange, ifInOctets, ifInUcastPkts, ifInNUcastPkts, ifInDiscards, ifInErrors, ifInUnknownProtos, ifOutOctets, ifOutUcastPkts, ifOutNUcastPkts, ifOutDiscards, ifOutErrors, ifOutQLen, ifSpecific.
- **’IP Group’**
  - **’IP Group->IP Values...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.ip’ table. These variables are: ipForwarding, ipDefaultTTL, ipInReceives, ipInHdrErrors, ipInAddrErrors, ipForwDatagrams, ipInUnknownProtos, ipInDiscards, ipInDelivers, ipOutRequests, ipOutDiscards, ipOutNoRoutes, ipReasmTimeout, ipReasmReqds, ipReasmOKs, ipReasmFails, ipFragOKs, ipFragFails, ipFragCreates, ipRoutingDiscards.
  - **’IP Group->IP Address Table...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.ip.ipAddrTable.ipAddrEntry’ table. For each entry one table row will be displayed containing these variables: ipAdEntAddr, ipAdEntIfIndex, ipAdEntNetMask, ipAdEntBcastAddr, ipAdEntReasmMaxSize.
  - **’IP Group->IP Routing Table...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.ip.ipRouteTable.ipRouteEntry’ table. For each entry one table row will be displayed containing these variables: ipRouteDest, ipRouteIfIndex, ipRouteMetric1, ipRouteMetric2, ipRouteMetric3, ipRouteMetric4, ipRouteNextHop, ipRouteType, ipRouteProto, ipRouteAge, ipRouteMask, ipRouteMetric5, ipRouteInfo.
  - **’IP Group->IP Net2MediaTable...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.ip.ipNetToMediaTable.ipNetToMediaEntry’ table. For each entry one table row will be displayed containing these variables: ipNetToMediaIfIndex, ipNetToMediaPhysAddress, ipNetToMediaNetAddress, ipNetToMediaType.
- **’TCP Group’**
  - **’TCP Group->TCP Values...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.tcp’ table. These variables are: tcpRtoAlgorithm, tcpRtoMin, tcpRtoMax. tcpMaxConn, tcpActiveOpens, tcpPassiveOpens, tcpAttemptFails, tcpEstabResets, tcpCurrEstab, tcpInSegs, tcpOutSegs, tcpRetransSegs, tcpInErrs, tcpOutRsts.
  - **’TCP Group->Connection Table...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.tcp.tcpConnTable.tcpConnEntry’ table. For each entry one table row will be displayed containing these variables: tcpConnState, tcpConnLocalAddress, tcpConnLocalPort, tcpConnRemAddress, tcpConnRemPort.
- **’UDP Group’**
  - **’UDP Group->UDP Values’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.udp’ table. These variables are: udpInDatagrams, udpNoPorts, udpInErrors, udpOutDatagrams.

  - **’UDP Group->UDP Table...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.udp.udpTable.udpEntry’ table. For each entry one table row will be displayed containing these variables: `udpLocalAddress`, `udpLocalPort`.

- **’ICMP Values...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.icmp’ table. These variables are: `icmpInMsgs`, `icmpInErrors`, `icmpDestUnreachs`, `icmpInTimeExcds`, `icmpInParmProbs`, `icmpInSrcQuenchs`, `icmpInRedirects`, `icmpInEchos`, `icmpInEchoReps`, `icmpInTimestamps`, `icmpInTimestampReps`, `icmpInAddrMasks`, `icmpInAddrMaskReps`, `icmpOutMsgs`, `icmpOutErrors`, `icmpOutDestUnreachs`, `icmpOutTimeExcds`, `icmpOutParmProbs`, `icmpOutSrcQuenchs`, `icmpOutRedirects`, `icmpOutEchos`, `icmpOutEchoReps`, `icmpOutTimestamps`, `icmpOutTimeStampReps`, `icmpOutAddrMasks`, `icmpOutAddrMaskReps`.

- **’EGP Group’**

  - **’EGP Group->EGP Values...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.egp’ table. These variables are: `egpInMsgs`, `egpInErrors`, `egpOutMsgs`, `egpOutErrors`, `egpAs`.

  - **’EGP Group->Neighbour Table...’**: This will display the variables found in the ’.iso.org.dod.internet.mgmt.mib-2.egp.egpNeighTable.egpNeighEntry’ table. For each entry one table row will be displayed containing these variables: `egpNeighState`, `egpNeighAddr`, `egpNeighAs`, `egpNeighInMsgs`, `egpNeighInErrs`, `egpNeighOutMsgs`, `egpNeighOutErrs`, `egpNeighInErrMsgs`, `egpNeighOutErrMsgs`, `egpNeighStateUps`, `egpNeighStateDowns`, `egpNeighIntervalHello`, `egpNeighIntervalPoll`, `egpNeighMode`, `egpNeighEventTrigger`.

A Rights

The plugin’s access rights are integrated into the general access management (*see OpenScape FM Desktop User Guide*).

The description of the individual rights can be found within the tooltipps for the corresponding right symbols (tree or submap).

The names of the rights for this plugin begin with the plugin designation *Enterprise MIB*.

Index

mitel.com