

5

Retrieving Network Inventory

Network infrastructure objects (NEs, Modules, Ports, TPs and Port Connections) can be retrieved by accessing the tables below:

- *enmsNETable*
- *enmsModuleTable*
- *enmsPortTable*
- *enmsTPTable*
- *enmsPortConnTable*
- *enmsEquipHolderTable*

For all tables above except *enmsEquipHolderTable*, SNMP NBI sends object creation (OC) and object deletion (OD) notifications, as well as state change (SC) and attribute value change (AVC) notifications for selected object attributes.

5.1 Network Elements

5.1.1 List of NEs (*enmsNETable*)

Table *enmsNETable* contains all NEs in the network. It supports OC, OD, AVC and SC notifications.

Attribute name	Data type	Notif.	Description
<u>enmsNeNEId</u>	NEId		Global NE identifier (table index).
enmsNeType	DisplayString	AVC	NE type.
enmsNeName	DisplayString	AVC	NE name, as reported by the NE.
enmsNeLocation	DisplayString	AVC	NE location in the network map (topology container). If the NE is in a system container, the system container name is returned.
enmsNeAlarmSeverity	PerceivedSeverity		Highest severity of all alarms affecting the NE or its modules, ports and TPs.

Attribute name	Data type	Notif.	Description
enmsNeOperatingMode	OperatingMode	AVC	NE's ability to send notifications: - <i>Operation</i> : NE reports all events to TNMS; - <i>Maintenance</i> : NE is suppressing all notifications (typically for maintenance purposes, to prevent heavy DCN load).
enmsNeOpState	OperationalState	SC	NE operational state.
enmsNeCanBroadcast	Boolean		Indicates if at least one port has broadcast capabilities.
enmsNeCanPathProtection	Boolean		Indicates if at least one port allows SNCP creation via TNMS.
enmsNeClass	NEClass		(to be supported)
enmsNeExternalNEId	InfoString		Obsolete.
enmsNelsPseudoNE	Boolean		Obsolete.
enmsNeldName	DisplayString	AVC	NE name, as specified by the operator in TNMS.
enmsNeCommunicationState	OperationalState	AVC	Indicates whether the communication with the NE is fully operational.
enmsNeDCNLocation	DisplayString		Path to the NE in the DCN tree. (to be supported)
enmsNeSystemContainer	DisplayString	AVC	System container name.
enmsNeUserText	DisplayString	AVC	User Text, as specified by the operator in the DCN property page of the NE.

Table 11 enmNETable attributes

5.1.2 Notification of NE object creation (enmsNEObjectCreationTrap)

Notification sent when an NE is added to the NE table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsNeNEId	NEId	Global NE identifier.
enmsNeName	DisplayString	NE name, as reported by the NE.
enmsTrapEventDetails	DisplayString	Not used.
enmsTrapNeldName	DisplayString	NE name, as specified by the operator in TNMS.
enmsNeSystemContainer	DisplayString	System container of the NE.
enmsNeUserText	DisplayString	User Text as specified by the operator in the DCN property page of the NE.

Table 12 List of *enmsNEObjectCreationTrap* attributes

5.1.3 Notification of NE object deletion (*enmsNEObjectDeletionTrap*)

Notification sent when an NE is removed from the NE table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsNeNEId	NEId	Global NE identifier.
enmsNeName	DisplayString	NE name, as reported by the NE.
enmsTrapEventDetails	DisplayString	Not used.
enmsTrapNeldName	DisplayString	NE name, as specified by the operator in TNMS.
enmsNeSystemContainer	DisplayString	System container of the NE.
enmsNeUserText	DisplayString	User Text as specified by the operator in the DCN property page of the NE.

Table 13 List of *enmsNEObjectDeletionTrap* attributes

5.1.4 Notification of NE state change (*enmsNEStateChangeTrap*)

Notification sent when an NE state attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.

Variable name	Data type	Description
enmsNeNEId	NEId	Global NE identifier.
enmsNeName	DisplayString	NE name, as reported by the NE.
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapStateName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapStateOldValue	Integer	Old state value. Check the possible enumerated values of the changed attribute.
enmsTrapStateNewValue	Integer	New state value. Check the possible enumerated values of the changed attribute.
enmsTrapNeldName	DisplayString	NE name, as specified by the operator in TNMS.
enmsNeSystemContainer	DisplayString	System container of the NE.
enmsNeUserText	DisplayString	User Text as specified by the operator in the DCN property page of the NE.

Table 14 List of *enmsNEStateChangeTrap* attributes

5.1.5 Notification of NE attribute value change (enmsNEAttributeChangeTrap)

Notification sent when an NE attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsNeNEId	NEId	Global NE identifier.
enmsNeName	DisplayString	NE name, as reported by the NE. Note: If the AVC relates to the NE name, this field contains the new value.
enmsTrapEventDetails	DisplayString	Empty.

Variable name	Data type	Description
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapAttributeName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapAttributeOldValue	DisplayString	Old attribute value.
enmsTrapAttributeNewValue	DisplayString	New attribute value.
enmsTrapNeIdName	DisplayString	NE name, as specified by the operator in TNMS. Note: If the AVC relates to the NE Id name, this field contains the new value.
enmsNeSystemContainer	DisplayString	System container of the NE.
enmsNeUserText	DisplayString	User Text as specified by the operator in the DCN property page of the NE.

Table 15 List of *enmsNEAttributeChangeTrap* attributes

5.2 Modules

5.2.1 List of Modules (enmsModuleTable)

Table *enmsModuleTable* contains all modules of the NEs in the network. It supports OC, OD and SC notifications.

Attribute name	Data type	Notif.	Description
<u>enmsMoNEId</u>	NEId		NE identifier (table index).
<u>enmsMoModuleId</u>	ModuleId		Module Identifier within the NE (table index).
enmsMoType	DisplayString		Type of module. (Obsolete)
enmsMoName	DisplayString		Module name.
enmsMoOpState	OperationalState	SC	Operational state.

Attribute name	Data type	Notif.	Description
enmsMoShelf	DisplayString		Shelf and/or rack of the module, separated by a dash if both are present.
enmsMoSlot	DisplayString		Slot number of the module. If present, the sub-slot is appended, separated by a dash.
enmsMoObjectType	Integer32		Internal object type.
enmsMoNativeLocation	DisplayString	AVC	Optional native object location in the network element.

Table 16 List of *enmsModuleTable* attributes

5.2.2 Notification of Module object creation (enmsModuleObjectCreationTrap)

Notification sent when a new module is added to the module table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsMoNEId	NEId	Global NE identifier.
enmsMoModuleId	ModuleId	Module identifier.
enmsTrapEventDetails	DisplayString	Empty.

Table 17 List of *enmsModuleObjectCreationTrap* attributes

5.2.3 Notification of Module object deletion (enmsModuleObjectDeletionTrap)

Notification sent when a module is removed from the module table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsMoNEId	NEId	Global NE identifier.
enmsMoModuleId	ModuleId	Module identifier.
enmsTrapEventDetails	DisplayString	Empty.

Table 18 List of *enmsModuleObjectDeletionTrap* attributes

5.2.4 Notification of Module state change (enmsModuleStateChangeTrap)

Notification sent when a module state attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsMoNEId	NEId	Global NE identifier.
enmsMoModuleId	ModuleId	Module identifier.
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapStateName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapStateOldValue	Integer	Old state value. Check the possible enumerated values of the changed attribute.
enmsTrapStateNewValue	Integer	New state value. Check the possible enumerated values of the changed attribute.

Table 19 List of *enmsModuleStateChangeTrap* attributes

5.2.5 Notification of Module attribute value change (enmsModuleAttributeChangeTrap)

Notification sent when a module attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsMoNEId	NEId	Global NE identifier.
enmsMoModuleId	ModuleId	Module identifier.
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.

Variable name	Data type	Description
enmsTrapAttributeName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapAttributeOldValue	DisplayString	Old attribute value.
enmsTrapAttributeNewValue	DisplayString	New attribute value.

Table 20 List of *enmsModuleAttributeChangeTrap* attributes

5.3 Ports

5.3.1 List of Ports (enmsPortTable)

Table *enmsPortTable* contains all ports in the network. It supports OC, OD, AVC and SC notifications.

Attribute name	Data type	Notif.	Description
<u>enmsPtNEId</u>	NEId		NE identifier (table index).
<u>enmsPtPortId</u>	PortId		Port Identifier within the NE. (table index).
enmsPtName	DisplayString	AVC	Port name.
enmsPtModuleId	ModuleId		Obsolete.
enmsPtTechnology	PTTechnology		Transport technology. A value of zero means indeterminate technology, not applicable or unknown.
enmsPtServiceType	PTServiceType		Supported service types.
enmsPtInterfaceType	PTInterfaceType		Type of interface.
enmsPtBandwidth	Bandwidth		Port bandwidth.
enmsPtOpState	OperationalState		Obsolete.
enmsPtOperatingMode	OperatingMode		Obsolete.
enmsPtDirection	Directionality		Port direction.
enmsPtCanBroadcast	Boolean		Indicates if the port has broadcast capabilities.

Attribute name	Data type	Notif.	Description
enmsPtCanPathProtection	Boolean		Indicates if the port allows SNCP creation via TNMS.
enmsPtAlarmSeverity	PerceivedSeverity		Highest severity of all alarms affecting the port or its TPs.
enmsPtAdminState	AdministrativeState		Obsolete.
enmsPtOpStateTX	OperationalState	SC	Operational state in TX direction.
enmsPtOpStateRX	OperationalState	SC	Operational state in RX direction.
enmsPtModuleIdTX	ModuleId		Module ID for TX direction.
enmsPtModuleIdRX	ModuleId		Module ID for RX direction.
enmsPtLayerSet	LayerSet		Terminated layers.
enmsPtProtectionType	PTProtectionType	AVC	Protection type in the context of MS Protections.
enmsPtObjectType	Integer32		Internal object type.
enmsPtNativeLocation	DisplayString	AVC	Optional native object location in the network element.

Table 21 List of *enmsPortTable* attributes

5.3.2 Notification of Port object creation (enmsPortObjectCreationTrap)

Notification sent when a port is added to the port table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPtNEId	NEId	Global NE identifier.
enmsPtPortId	PortId	Port identifier.
enmsPtName	DisplayString	Port name.
enmsTrapEventDetails	DisplayString	Empty.

Table 22 List of *enmsPortObjectCreationTrap* attributes

5.3.3 Notification of Port object deletion (enmsPortObjectDeletionTrap)

Notification sent when a port is removed from the port table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPtNEId	NEId	Global NE identifier.
enmsPtPortId	PortId	Port identifier.
enmsPtName	DisplayString	Port name.
enmsTrapEventDetails	DisplayString	Empty.

Table 23 List of *enmsPortObjectDeletionTrap* attributes

5.3.4 Notification of Port state change (enmsPortStateChangeTrap)

Notification sent when a port state attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPtNEId	NEId	Global NE identifier.
enmsPtPortId	PortId	Port identifier.
enmsPtName	DisplayString	Port name.
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapStateName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapStateOldValue	Integer	Old state value. Check the possible enumerated values of the changed attribute.
enmsTrapStateNewValue	Integer	New state value. Check the possible enumerated values of the changed attribute.

Table 24 List of *enmsPortStateChangeTrap* attributes

5.3.5 Notification of Port attribute value change (enmsPortAttributeChangeTrap)

Notification sent when a port attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPtNEId	NEId	Global NE identifier.
enmsPtPortId	PortId	Port identifier.
enmsPtName	DisplayString	Port name. Note: If the AVC relates to the port name, this field contains the new value.
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapAttributeName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapAttributeOldValue	DisplayString	Old attribute value.
enmsTrapAttributeNewValue	DisplayString	New attribute value.

Table 25 List of *enmsPortAttributeChangeTrap* attributes

5.4 Termination Points

5.4.1 List of TPs (enmsTPTable)

Table *enmsTPTable* contains all termination points in the network. It supports OC, OD, AVC and SC notifications.

Attribute name	Data type	Notif.	Description
<u>enmsTpNEId</u>	NEId		NE Id (table index).
<u>enmsTpPortId</u>	PortId		Port Id (table index).
<u>enmsTpTPIdH</u>	TPId		Higher 32 bits of TP Id (table index).

Attribute name	Data type	Notif.	Description
<u>enmsTpTPIdL</u>	TPId		Lower 32 bits of TP Id (table index).
enmsTpTPIndex	TPIndex		TP index relative to the port (might be a timeslot index).
enmsTpNxCount	Integer32	AVC	The number of carried signals in an Inverse Multiplexing group, if applicable. Otherwise, value is 1.
enmsTpName	DisplayString	AVC	TP name.
enmsTpBandwidth	Bandwidth		TP bandwidth.
enmsTpTPTYPE	TPTYPE		Obsolete.
enmsTpTerminType	TPTerminationType		Termination type.
enmsTpDirection	Directionality		TP direction.
enmsTpOpStateTX	OperationalState	SC	Operational state in TX direction.
enmsTpOpStateRX	OperationalState	SC	Operational state in RX direction.
enmsTpAlarmSeverity	PerceivedSeverity		Highest severity of all alarms affecting the TPs.
enmsTpAdminState	OperationalState		Obsolete.
enmsTpUsageCountTX	Integer32		(to be supported) Number of cross connections using the TP for the TX direction.
enmsTpUsageCountRX	Integer32		(to be supported) Number of cross connections using the TP for the RX direction.
enmsTpUsageStateTX	UsageState		(to be supported) Usage state in the TX direction.
enmsTpUsageStateRX	UsageState		(to be supported) Usage state in the RX direction.
enmsTpReliability	TPReliability		Obsolete.

Attribute name	Data type	Notif.	Description
enmsTpLayerSet	LayerSet		TP transport layer set.
enmsTpBandwidthTX	Bandwidth		(to be supported) TP bandwidth in the TX direction.
enmsTpBandwidthRX	Bandwidth		(to be supported) TP bandwidth in the RX direction.
enmsTpParentPortId	PortId		Port Id of parent TP (if applicable).
enmsTpParentTPIdH	TPId		Higher 32 bits of TP Id of parent TP (if applicable).
enmsTpParentTPIdL	TPId		Lower 32 bits of TP Id of parent TP (if applicable).
enmsTpFragmentLayer	LayerSet		Fragment layer set, if the TP represents a VC group.
enmsTpObjectType	Integer32		Internal object type.
enmsTpMuxPartnerTPIdH	TPId		Higher 32 bits of TP Id of multiplex partner TP (if applicable).
enmsTpMuxPartnerTPIdL	TPId		Lower 32 bits of TP Id of multiplex partner TP (if applicable).
enmsTpNativeLocation	DisplayString	AVC	Optional native object location in the network element.

Table 26 List of *enmsTPTable* attributes

5.4.2 Notification of TP object creation (enmsTPObjectCreationTrap)

Notification sent when a TP is added to the TP table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsTpNEId	NEId	Global NE identifier.
enmsTpPortId	PortId	Port identifier.
enmsTpTPIdH	TPId	Higher 32 bits of TP Id.

Variable name	Data type	Description
enmsTpTPIdL	TPId	Lower 32 bits of TP Id.
enmsTpName	DisplayString	TP name.
enmsTpTPTYPE	TPTYPE	Obsolete. Always <i>unknown</i> .
enmsTrapEventDetails	DisplayString	Empty.

Table 27 List of *enmsTPOBJECTCreationTrap* attributes

5.4.3 Notification of TP object deletion (enmsTPOBJECTDeletionTrap)

Notification sent when a TP is removed from the TP table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsTpNEId	NEId	Global NE identifier.
enmsTpPortId	PortId	Port identifier.
enmsTpTPIdH	TPId	Higher 32 bits of TP Id.
enmsTpTPIdL	TPId	Lower 32 bits of TP Id.
enmsTpName	DisplayString	TP name.
enmsTpTPTYPE	TPTYPE	Obsolete. Always <i>unknown</i> .
enmsTrapEventDetails	DisplayString	Empty.

Table 28 List of *enmsTPOBJECTDeletionTrap* attributes

5.4.4 Notification of TP state change (enmsTPStateChangeTrap)

Notification sent when a TP state attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsTpNEId	NEId	Global NE identifier.
enmsTpPortId	PortId	Port identifier.
enmsTpTPIdH	TPId	Higher 32 bits of TP Id.
enmsTpTPIdL	TPId	Lower 32 bits of TP Id.

Variable name	Data type	Description
enmsTpName	DisplayString	TP name.
enmsTpTPType	TPType	Obsolete. Always <i>unknown</i> .
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapStateName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapStateOldValue	Integer	Old state value. Check the possible enumerated values of the changed attribute.
enmsTrapStateNewValue	Integer	New state value. Check the possible enumerated values of the changed attribute.

Table 29 List of *enmsTPStateChangeTrap* attributes

5.4.5 Notification of TP attribute value change (enmsTPAttributeChangeTrap)

Notification sent when a TP attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsTpNEId	NEId	Global NE identifier.
enmsTpPortId	PortId	Port identifier.
enmsTpTPIdH	TPId	Higher 32 bits of TP Id.
enmsTpTPIdL	TPId	Lower 32 bits of TP Id.
enmsTpName	DisplayString	TP name.
enmsTpTPType	TPType	Obsolete. Always <i>unknown</i> .
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.

Variable name	Data type	Description
enmsTrapAttributeName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapAttributeOldValue	DisplayString	Old attribute value.
enmsTrapAttributeNewValue	DisplayString	New attribute value.

Table 30 List of *enmsTPAttributeChangeTrap* attributes

5.5 Port Connections

5.5.1 List of Port Connections (enmsPortConnTable)

Table *enmsPortConnTable* contains all managed ports connections. It supports OC, OD and AVC notifications.

Attribute name	Data type	Notif.	Description
<u>enmsPcPortConnId</u>	PortConnId		Port Connection identifier (table index).
enmsPcSrcNEId	NEId		NE Id of source port.
enmsPcSrcPortId	PortId		Port Id of source port.
enmsPcDstNEId	NEId		NE Id of destination port.
enmsPcDstPortId	PortId		Port Id of destination port.
enmsPcName	DisplayString	AVC	Port connection name.
enmsPcSrcAlarmSeverity	PerceivedSeverity	AVC	Highest severity of all alarms affecting the source port or module.
enmsPcDstAlarmSeverity	PerceivedSeverity	AVC	Highest severity of all alarms affecting the destination port or module.
enmsPcBandwidth	Bandwidth		Port connection bandwidth.
enmsPcDirection	Directionality		Port connection direction.
enmsPcLayerSet	LayerSet		Port connection layer set.

Table 31 List of *enmsPortConnTable* attributes

5.5.2 Notification of Port Connection object creation (enmsPortConnObjectCreationTrap)

Notification sent when a Port Connection is added to the Port Connection table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPcPortConnId	PortConnId	Global port connection identifier.
enmsPcName	DisplayString	Port connection name.
enmsTrapEventDetails	DisplayString	Empty.

Table 32 List of *enmsPortConnObjectCreationTrap* attributes

5.5.3 Notification of Port Connection object deletion (enmsPortConnObjectDeletionTrap)

Notification sent when a Port Connection is removed from the Port Connection table.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPcPortConnId	PortConnId	Global port connection identifier.
enmsPcName	DisplayString	Port connection name.
enmsTrapEventDetails	DisplayString	Empty.

Table 33 List of *enmsPortConnObjectDeletionTrap* attributes

5.5.4 Notification of Port Connection attribute value change (enmsPortConnAttributeChangeTrap)

Notification sent when a Port Connection attribute has a new value.

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPcPortConnId	PortConnId	Global port connection identifier.
enmsPcName	DisplayString	Port connection name.

Variable name	Data type	Description
		Note: If the AVC relates to the port connection name, this field contains the new value.
enmsTrapEventDetails	DisplayString	Empty.
enmsTrapEventSeverity	PerceivedSeverity	Not relevant.
enmsTrapEventProbableCause	ProbableCause	Not relevant.
enmsTrapAttributeName	Integer	Indicates which table field has changed. Check possible values in MIB definition.
enmsTrapAttributeOldValue	DisplayString	Old attribute value.
enmsTrapAttributeNewValue	DisplayString	New attribute value.

Table 34 List of *enmsPortConnAttributeChangeTrap* attributes

5.6 Equipment Holders

5.6.1 List of Equipment Holders (enmsEquipHolderTable)

Table *enmsEquipHolderTable* contains all shelves. No notifications are sent for equipment holders.

Attribute name	Data type	Notif.	Description
<u>enmsEhNEId</u>	NEId		NE Identifier (table index).
enmsEhEquipHolderId	EquipmentHolderId		Equipment holder identifier within the NE (table index).
enmsEhType	EquipmentHolderType		Type of equipment holder. Note: Currently only shelves are supported.
enmsEhName	DisplayString		Equipment holder name.

Table 35 List of *enmsEquipHolderTable* attributes

10 Performance Monitoring

SNMP NBI allows managers to retrieve Performance Monitoring (PM) data associated to the network objects managed by TNMS:

- History data
- Current data
- PMP and threshold information



Retrieving history PM data via SNMP NBI requires PM data upload to be enabled in TNMS Server. Refer to TNMS documentation for information on how to enable PM data upload.

Given its nature and potentially large volume, PM data is not readily available in an MIB table. To retrieve PM data, the SNMP manager creates a request by inserting a row in the PM request table. Each request specifies the type of data to retrieve and for which network objects. The request is then executed, after which the resulting PM data may be retrieved.

The following diagrams exemplify the high level interaction between a manager wanting to retrieve PM data and SNMP NBI. In Figure 11, the manager creates a request and waits for a notification indicating that the PM data is ready for retrieval:

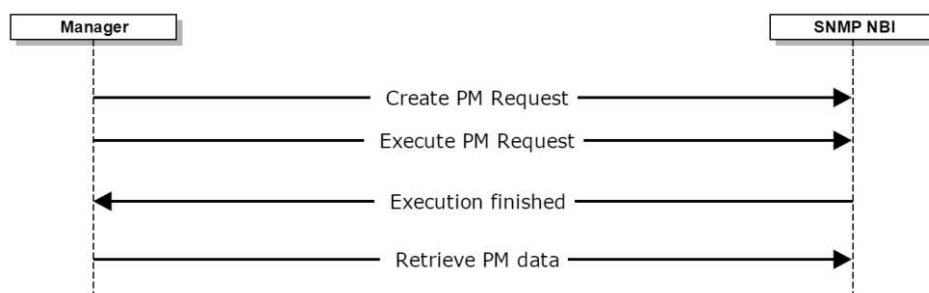


Figure 11 Example of PM data retrieval using notifications

Alternatively, the manager may poll the state of the request (instead of waiting for a notification):

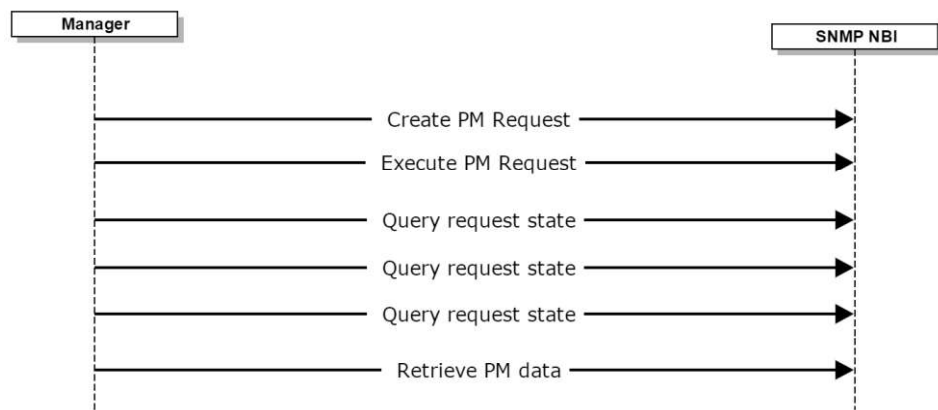


Figure 12 Example of PM data retrieval using polling

10.1 PM Requests

PM requests are entries of the MIB table *enmsPerfMonRequestTable*.

Each PM request specifies the type of data to retrieve and for which network objects.

10.1.1 PM request table (enmsPerfMonRequestTable)

This table contains all PM requests created by the SNMP manager.

Attribute name	Data type	Settable?	Description
<u>enmsPmRequestId</u>	PerfMonRequestId	No	Id of the request (table index).
enmsPmRequestName	DisplayString	Yes	Optional request name, for manager reference. Default is empty.
enmsPmRequestRowStatus	RowStatus	Yes (see descr.)	Standard SNMP RowStatus field for controlling row creation. Set to <i>createAndGo</i> (4) to create a new row, or <i>destroy</i> (6) to remove an existing row. Other values are not supported.
enmsPmRequestState	PerfMonRequestState	Yes (see descr.)	State of the PM request. Set this field to change the state of the request. Not settable at row creation.
enmsPmRequestLastUpdate	EnmsTimeStamp	No	Time (in UTC) of the last update of the request.
enmsPmRequestInfo	DisplayString	No	Information about request status.

Attribute name	Data type	Settable?	Description
enmsPmRequestType	PerfMonType	Yes	Type of PM data to retrieve: <ul style="list-style-type: none"> - <i>pmHistory</i> (1) – retrieve history data - <i>pmCurrent</i> (2) – retrieve current data - <i>pmPoints</i> (3) – retrieve PM point info and associated thresholds. Default is <i>pmCurrent</i> .
enmsPmRequestStartTime	EnmsTimeStamp	Yes	For history PM data, start time (in UTC) of the collection period. Default is empty.
enmsPmRequestEndTime	EnmsTimeStamp	Yes	For history PM data, end time (in UTC) of the collection period. Default is empty.
enmsPmRequestGranularity	PerfMonGranularity	Yes	Granularity of PM data: <ul style="list-style-type: none"> - <i>minutes15</i> (1) - <i>hours24</i> (2) Default is <i>minutes15</i> .
enmsPmRequestFilterType	FilterType	Yes	Type of object for which to retrieve PM data: <ul style="list-style-type: none"> - <i>tpObject</i> (1) - <i>portObject</i> (2) - <i>neObject</i> (3) - <i>sncObject</i> (4) - <i>ethernetPathObject</i> (5) - <i>moduleObject</i> (6) - <i>equipHolderObject</i> (7) (!) Filtering by NE object is unsupported for request type <i>pmCurrent</i> . Default is <i>sncObject</i> .

Attribute name	Data type	Settable?	Description
enmsPmRequestFilterValue	DisplayString	Yes	<p>Identifier of the object for which to retrieve PM data. The identifier of an object is the index of that object in the corresponding MIB table, with the individual index values separated by " " (pipe character). Examples:</p> <p>TP (enmsTPTable): 173 455 3453 99589454</p> <p>Port (enmsPortTable): 32 6734</p> <p>NE (enmsNETable) - history only 85</p> <p>SNC (enmsSNCTable): 8374</p> <p>Ethernet Path (enmsSNCTable): 2387</p> <p>Module (enmsModuleTable): 32 9334</p> <p>Port (enmsEquipHolderTable): 32 277</p> <p>Specify multiple objects by separating their identifiers with commas. Example for port objects: 32 6734,55 33928</p> <p>Default is 0.</p>

Table 73 List of *enmsPerfMonRequestTable* attributes

10.2 Creating a PM Request

Creating a PM request (adding a row to the MIB table *enmsPerfMonRequestTable*) follows the method defined in the SNMP RFC 2579, which uses a *RowStatus* attribute to control the row existence. The manager must perform the following steps:

- 1) Obtain an instance identifier for the new row, by reading the *enmsPmRequestNextId* leaf attribute using SNMP GET (this variable auto-increments with each GET access).
- 2) Using the new instance identifier, send an SNMP SET command to set *enmsPmRequestRowStatus* to *createAndGo* and optionally assign values to other attributes.

If the SNMP SET command succeeds, a new entry is added to *enmsPerfMonRequestTable*. If it fails, no entry will be added. Reasons for failure are listed below.

Error cause	SNMP error
- The provided instance identifier is already in use.	<i>inconsistentValue</i>

Error cause	SNMP error
<ul style="list-style-type: none"> - Attribute <i>enmsPmRequestRowStatus</i> set to <i>notReady</i>, <i>active</i>, <i>notInService</i> or <i>destroy</i>. - An attribute has been assigned a value inconsistent with other attributes. Examples: <ul style="list-style-type: none"> o The request type was set to <i>history</i>, but the start/end times are empty; o The filter type was set to <i>port</i>, but the filter value contains a TP identifier. 	<i>commitFailed</i>
<ul style="list-style-type: none"> - Attribute <i>enmsPmRequestRowStatus</i> not set. 	<i>inconsistentName</i> <i>commitFailed</i>
<ul style="list-style-type: none"> - Attribute <i>enmsPmRequestRowStatus</i> set to <i>createAndWait</i>. - Wrong type or invalid value assigned to an attribute. 	<i>wrongValue</i> <i>badValue</i> <i>commitFailed</i>
<ul style="list-style-type: none"> - Values assigned to non-settable attributes. 	<i>notWritable</i> <i>commitFailed</i>
<ul style="list-style-type: none"> - Maximum number of requests exceeded. 	<i>resourceUnavailable</i> <i>commitFailed</i>

Table 74 Possible errors during PM request creation.

For new requests, *enmsPmRequestRowStatus* is set to *active*, and *enmsPmRequestState* to *created*. The *enmsPmRequestId* index field is set to the supplied instance identifier. Other fields left unset are assigned default values.

Figure 13 exemplifies the creation of a new PM request:

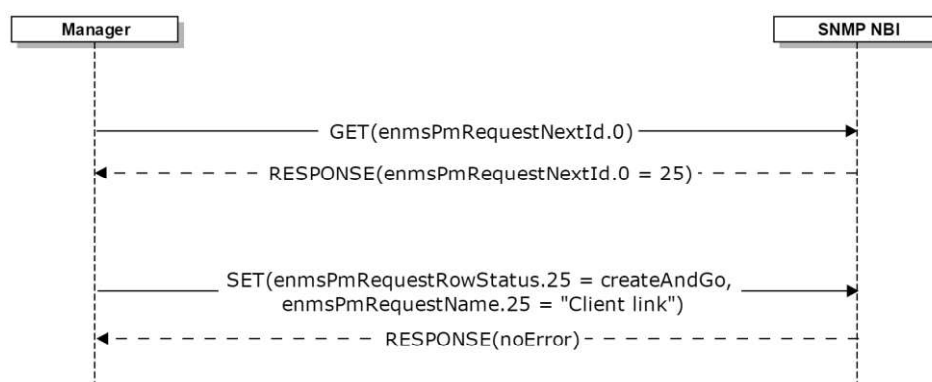


Figure 13 Example of interaction to create PM request

After the interaction above, *enmsPerfMonRequestTable* will contain a new entry for instance 25:

Id	Name	Row Status	State	Last Update	Info	Type	Start Time	End Time	Granul.	Filter Type	Filter Value
11	Access ports	active	finished	2016-03-20 14:01:12	Execution finished.	pmCurrent			minutes15	Port	123 98
19	Middle node	active	started	2016-03-20 14:01:12	Execution started.	pmHistory	2016-03-01 00:00:00	2016-03-15 00:00:00	hours24	SNC	320
25	Client link	active	created	2016-03-22 14:55:35	Request created.	pmCurrent			minutes15	SNC	0

Table 75 Example of *enmsPerfMonRequestTable* after adding a request

New requests are in the state *created*, meaning that the request exists but is not in execution.



Maximum number of PM requests is 5000. After this limit is reached, the manager must delete existing PM requests before adding new ones. Alternatively, the manager may reuse existing PM requests by updating its fields.

10.3 PM Request States

Table 76 lists the possible states of a PM request (reflected in the *enmsPmRequestState* attribute) and the actions the manager may perform for each state.

enmsPmRequestState	Description	Possible actions
created	Request is idle. This is the initial state of a newly created request.	<ul style="list-style-type: none"> - Execute the request - Update the request
pending	Request is awaiting execution.	<ul style="list-style-type: none"> - Cancel the request
started	Request is being executed.	<ul style="list-style-type: none"> - Cancel the request
finished	Request finished successfully. PM data is available for retrieving.	<ul style="list-style-type: none"> - Retrieve the PM data - Update the request - Re-execute the request - Discard the PM data
failed	Request failed and is idle.	<ul style="list-style-type: none"> - Update the request - Re-execute the request
cancelling	Request is being cancelled.	<ul style="list-style-type: none"> - None
cancelled	Request has been cancelled and is idle.	<ul style="list-style-type: none"> - Update the request - Re-execute the request

Table 76 States of a PM request and possible actions

The manager performs actions on a request by setting the *enmsPmRequestState* attribute using the SNMP SET operation (see section 10.5).

Figure 14 summarizes the main transitions between request states.

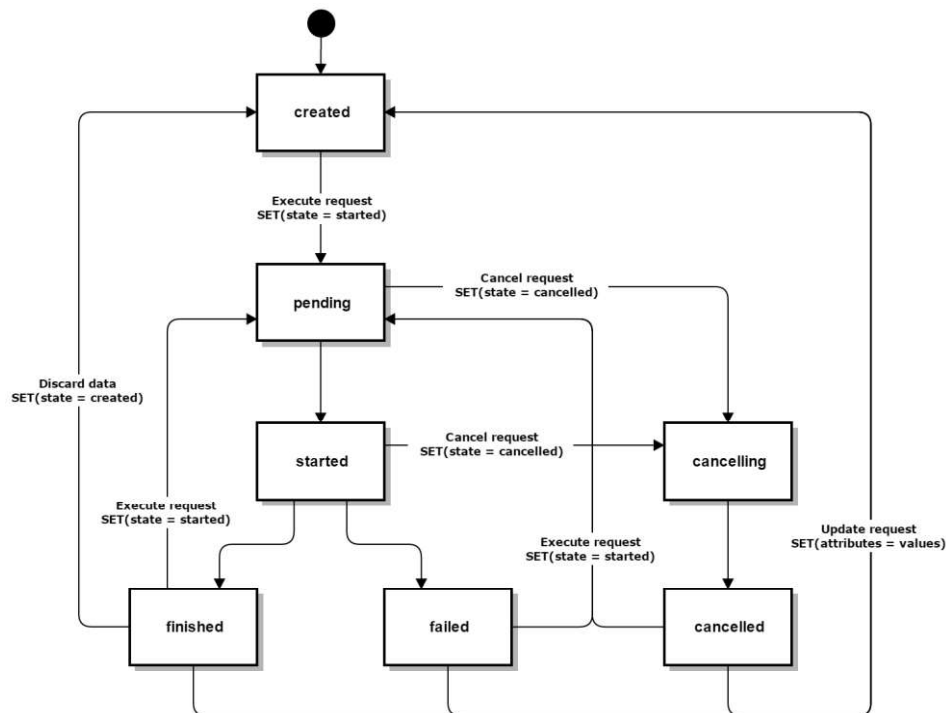


Figure 14 PM request states and main transitions

10.4 PM Request State Change Notifications

Whenever the state of a PM request changes, SNMP NBI sends an *enmsPerfMonRequestStateChangeTrap* notification to the managers. This notification carries the following attributes:

Variable name	Data type	Description
enmsTrapCounter	Counter32	Trap counter for synchronization.
enmsPmRequestId	PerfMonRequestId	Id of the request.
enmsPmRequestName	DisplayString	Name of the request.
enmsPmRequestState	PerfMonRequestState	New state of the request.
enmsPmRequestInfo	DisplayString	Information about request status.

Table 77 List of *enmsPerfMonRequestStateChangeTrap* attributes

10.5 Actions on PM Requests

Actions on PM requests are performed by issuing SNMP SET commands to change the PM request state and other attributes.

10.5.1 Executing a PM request

To execute a PM request, the manager sends an SNMP SET command to set *enmsPmRequestState* to *started*. The state will first change to *pending* (the request has been queued for execution). When execution starts, it will change to *started*.

i Executing a PM request is only possible if the request is in an idle state (either *created*, *finished*, *failed* or *cancelled*).



Figure 15 Example of interaction for executing a PM request

The manager can also update the PM request attributes (see 10.5.2) and execute it in a single SET command:

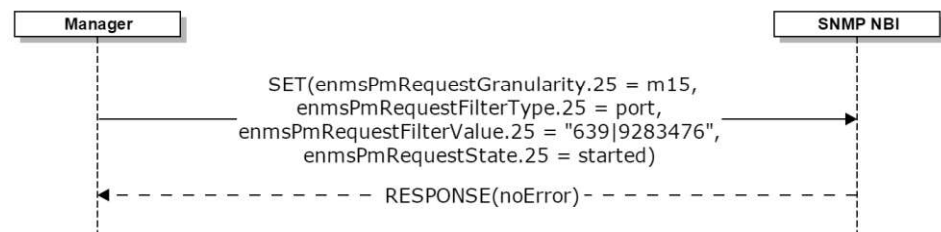


Figure 16 Updating and executing a PM request in a single SET command

If the request is executed successfully, the state will change to *finished*. The manager may retrieve the PM data (see section 10.6). If the request execution fails, the state will change to *failed*. The *enmsPmRequestInfo* field provides a hint on what caused the failure. Typical failure reasons include:

- Network objects for which to obtain PM data do not exist;
- A timeout occurred while collecting data from the NEs;
- An internal server error occurred.

In any case, the manager may update the PM request and re-execute it.



Executing a PM request in state *finished* causes associated PM data to be discarded.

10.5.2 Updating PM request attributes

When a PM request is created, unset attributes are assigned default values. Before executing the PM request, the manager must set those attributes with valid values, otherwise the PM request execution will fail. The manager may also want to reuse a previously executed PM request, or correct the attributes of a failed PM request.

To update the attributes of a PM request, the manager sends an SNMP SET. Multiple attributes can be set in a single SET command.

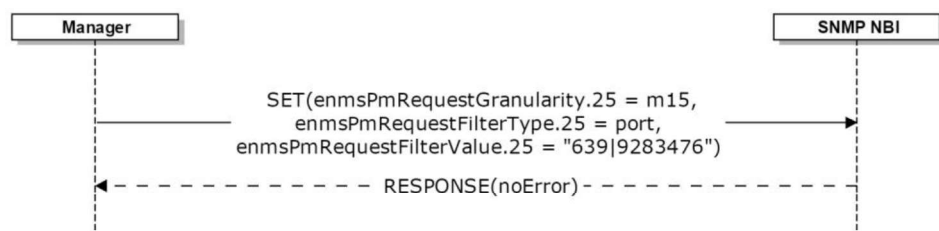


Figure 17 Example of interaction for updating a PM request

When assigning an invalid value to some attribute, the SET command fails with the error code *inconsistentValue*.

Updating a PM request moves it to state *created*, and any associated PM data is discarded.

It is also possible to update the PM request attributes and execute it at the same time, in a single SET command (see 10.5.1).



Updating a PM request is only possible if the request is in an idle state (either *created*, *finished*, *failed* or *cancelled*).



Updating a PM request in state *finished* causes associated PM data to be discarded.

10.5.3 Cancelling a PM request

To cancel the execution of a PM request, the manager sends an SNMP SET command to set *enmsPmRequestState* to *cancelled*. The state changes to *cancelling*, and then to *cancelled*.

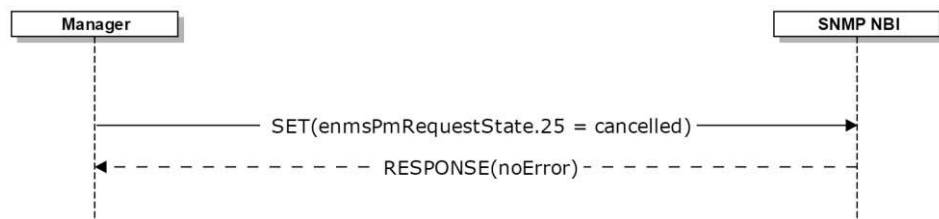


Figure 18 Example of interaction for cancelling a PM request



Cancelling a PM request is only possible if the request is in state *pending* or *started*.

10.5.4 Discarding PM data associated to a PM request

After a PM request is successfully executed, resulting PM data is preserved until the request is re-executed or deleted.

To keep the request for future re-execution, the manager may discard the data by sending an SNMP SET command to set *enmsPmRequestState* to *created*. This way, resources are freed up from the server, while the PM request is kept for reuse.





Figure 19 Example of interaction for discarding PM data of a PM request

10.5.5 Deleting a PM request

To delete a PM request, the manager sends an SNMP SET command to set *enmsPmRequestRowStatus* to *destroy*.



Figure 20 Example of interaction for deleting a PM request

-  Deleting a request is only possible if the request is in an idle state (either *created*, *finished*, *failed* or *cancelled*).
-  Deleting a PM request also deletes associated PM data.

10.5.6 Error exceptions

Table 78 describes common errors returned by SNMP NBI while performing SNMP SET operations on existing PM requests:

Error cause	SNMP error
<ul style="list-style-type: none"> - Attribute <i>enmsPmRequestState</i> set an invalid value given the current request state (see 10.3). - An attribute was assigned an inconsistent value. Examples: <ul style="list-style-type: none"> o The request type was set to <i>history</i>, but the start/end times are empty; o The filter type was set to <i>port</i>, but the filter value contains a TP identifier. 	<i>inconsistentValue</i> <i>commitFailed</i>
<ul style="list-style-type: none"> - Attribute <i>enmsPmRequestRowStatus</i> set to other value than <i>destroy</i>. - Wrong type or invalid value assigned to an attribute. 	<i>wrongValue</i> <i>badValue</i> <i>commitFailed</i>
<ul style="list-style-type: none"> - Values assigned to non-settable attributes. 	<i>notWritable</i> <i>commitFailed</i>

Table 78 Common errors while performing actions on PM requests

10.6 Retrieving PM Data

When the execution of a PM request finishes successfully, the PM request state changes to *finished*. The manager may now retrieve the associated PM data by accessing the following tables:

- *enmsPerfMonResultPmpTable*: Contains the PM Points of the request results (see 10.6.2).
- *enmsPerfMonResultValueTable*: For requests of type *pmHistory* and *pmCurrent*, contains the measured values for each PM Point (see 10.6.3).
- *enmsPerfMonResultThresholdTable*: For requests of type *pmPoints*, contains the threshold values for each PM Point (see 10.6.4).

These tables contain the PM data results for all finished requests. To retrieve data for a specific PM request, follow the method suggested in Section 4.5.3 (all tables above are indexed by PM request identifier).



Because of the potentially large volume of data, request only the strictly needed attributes to speed-up the retrieval operation.

10.6.1 PM data retention period

PM data is available for the following approximate times (after the PM request finishes):

- History data: 12 hours
- Current data: 1 hour
- PM points: 1 hour

Results older than the intervals above are periodically discarded and the associated requests are moved to the *created* state.



History data is only available as long as it exists in TNMS Server, which has its own retention period.

It is recommended that the manager forces the data to be discarded as soon as it finishes retrieving it (see section 10.5.4).

10.6.2 enmsPerfMonResultPmpTable

Table 79 contains the PM Points of all finished PM request results.

Attribute name	Data type	Description
<u>enmsPmResultPmpReqId</u>	PerfMonRequestId	PerfMon request identifier. (table index)
<u>enmsPmResultPmpPmpNumber</u>	Unsigned32	Number of the PMP in the result set. (table index)
enmsPmResultPmpNeld	NEId	NE Id of the PMP.
enmsPmResultPmpPortId	PortId	Port Id of the PMP.
enmsPmResultPmpTPIdH	TPId	Highest 32-bits of the TP Id of the PMP, if applicable.
enmsPmResultPmpTPIdL	TPId	Lowest 32-bits of the TP Id of the PMP, if applicable.
enmsPmResultPmpNeldName	DisplayString	NE Id name of the NE of the PMP.
enmsPmResultPmpObjLocation	DisplayString	Object location of the PMP.
enmsPmResultPmpName	DisplayString	PMP name.

Attribute name	Data type	Description
enmsPmResultPmpLocation	PerfMonLocation	PMP location (near end/far end).
enmsPmResultPmpDirection	PerfMonDirection	PMP direction.
enmsPmResultPmpRetrievalTime	EnmsTimeStamp	Retrieval time.
enmsPmResultPmpPeriodEndTime	EnmsTimeStamp	End time of the collection period.
enmsPmResultPmpMonitoredTime	Unsigned32	Total monitored time, or zero if not supported by the NE.
enmsPmResultPmpNumValues	Unsigned32	Number of values collected for this PMP.
enmsPmResultPmpRelatedPaths	DisplayString	Names of the paths related to this PMP (comma-separated list, truncated to field size limits). Only applicable to history data, empty for other types of request. Requires correlation of PM data to paths to be enabled in the server.
enmsPmResultPmpRelatedServices	DisplayString	Names of the services with paths related to this PMP (comma-separated list, , truncated to field size limits). Only applicable to history data, empty for other types of request. Requires correlation of PM data to paths to be enabled in the server.
enmsPmResultPmpRelatedSubscribers	DisplayString	Names of the subscribers related to this PMP (comma-separated list, truncated to field size limits). Only applicable to history data, empty for other types of request. Requires correlation of PM data to paths to be enabled in the server.
enmsPmResultPmpNativeLocation	DisplayString	(to be supported) Optional native object location of the PMP in the network element.

Table 79 List of *enmsPerfMonResultPmpTable* attributes

10.6.3 enmsPerfMonResultValueTable

Table 80 contains the measured values for the PM Points of all finished PM request results, for requests of type *pmHistory* or *pmCurrent*.

Attribute name	Data type	Description
<u>enmsPmResultValReqId</u>	PerfMonRequestId	PM request identifier (table index).
<u>enmsPmResultValPmpNumber</u>	Unsigned32	Number of the PMP in the result set. (table index)
<u>enmsPmResultValNumber</u>	Unsigned32	Number of value in the collected values for the PMP. (table index)
enmsPmResultValParam	DisplayString	Parameter name.
enmsPmResultValValue	DisplayString	Parameter value.
enmsPmResultValUnit	DisplayString	Parameter unit.
enmsPmResultValStatus	PerfMonStatus	Status of the collected value.

Table 80 List of *enmsPerfMonResultValueTable* attributes

10.6.4 enmsPerfMonResultThresholdTable

Table 81 contains the threshold values for the PM Points of all finished PM request results, for requests of type *pmPoints* only.

Attribute name	Data type	Description
<u>enmsPmResultThresholdReqId</u>	PerfMonRequestId	PM request identifier (table index).
<u>enmsPmResultThresholdPmpNumber</u>	Unsigned32	Number of the PMP in the result set (table index).
<u>enmsPmResultThresholdNumber</u>	Unsigned32	Number of value in the collected values for the PMP. (table index)
enmsPmResultThresholdParam	DisplayString	Parameter name.
enmsPmResultThresholdType	PerfMonThresholdType	Threshold type.
enmsPmResultThresholdTriggerFlag	Boolean	Indicates if the threshold is for the trigger (<i>true</i>) or the clear (<i>false</i>).

Attribute name	Data type	Description
		Only valid if the NE supports separate values for trigger and clear.
enmsPmResultThresholdValue	DisplayString	Threshold value.
enmsPmResultThresholdUnit	DisplayString	Threshold unit.

Table 81 List of *enmsPerfMonResultThresholdTable* attributes

12

Net-SNMP Examples for PM Data Retrieval

Net-SNMP (<http://www.net-snmp.org>) is a set of command-line tools widely used to interact with SNMP agents. This chapter provides examples on how to use Net-SNMP to request PM data via TNMS SNMP NBI.

The workflow for retrieving PM data is described in Chapter 10 – read it first.

For simplicity, the NET-SNMP examples in this document omit common arguments for specifying the MIB, credentials and output formatting options. A full complete command may look like the following:

```
snmptable -m TNMS-NBI-MIB -M C:/Home/MIB -v 2c -c public tnmsserver enmsPerfMonRequestTable
```

Common arguments

Please see Net-SNMP documentation for details.

12.1 Creating a New PM Request

Get a unique index for the new PM request by reading *enmsPmRequestNextId*, which auto-increments each time it is accessed.

You may use any other index generation method instead of *enmsPmRequestNextId* – just make sure the chosen index is not in use in *enmsPerfMonRequestTable*.

```
snmpget enmsPmRequestNextId.0
```

```
TNMS-NBI-MIB::enmsPmRequestNextId.0 = Gauge32: 4
```

Use the selected index (4 in this example) to create a new PM request. *RowStatus* must be set to *createAndGo(4)*:

```
snmpset enmsPmRequestRowStatus.4 i createAndGo
enmsPmRequestName.4 s "Demo request"
enmsPmRequestType.4 i pmHistory
enmsPmRequestStartTime.4 s "2017-03-07 12:00:00"
enmsPmRequestEndTime.4 s "2017-03-07 17:00:00"
enmsPmRequestGranularity.4 i minutes15
enmsPmRequestFilterType.4 i ethernetPathObject
enmsPmRequestFilterValue.4 s "32"
```

```
TNMS-NBI-MIB::enmsPmRequestRowStatus.4 = INTEGER: createAndGo(4)
TNMS-NBI-MIB::enmsPmRequestName.4 = STRING: Demo request
TNMS-NBI-MIB::enmsPmRequestType.4 = INTEGER: pmHistory(1)
TNMS-NBI-MIB::enmsPmRequestStartTime.4 = STRING: "2017-03-07 12:00:00"
TNMS-NBI-MIB::enmsPmRequestEndTime.4 = STRING: "2017-03-07 17:00:00"
TNMS-NBI-MIB::enmsPmRequestGranularity.4 = INTEGER: minutes15(1)
TNMS-NBI-MIB::enmsPmRequestFilterType.4 = INTEGER: ethernetPathObject(5)
TNMS-NBI-MIB::enmsPmRequestFilterValue.4 = STRING: 32
```

Listing all PM requests to confirm new row:

```
snmpstable enmsPerfMonRequestTable
```

```
SNMP table: TNMS-NBI-MIB::enmsPerfMonRequestTable
```

index	Name	RowStatus	LastUpdate	Info	State	Type	StartTime
1	ethernetCarrier	1	"2017-03-03 19:47:00"	Request reinitialized.	1	2	"
2	mpls_7090	1	"2017-03-07 08:36:43"	Request reinitialized.	1	1	"2017-03-06 20:00:00"
3		1	"2017-03-07 16:04:22"	Execution finished. Total values: 84.	4	1	"2017-03-07 13:43:31"
4	Demo request	1	"2017-03-07 16:35:18"	Request created.	1	1	"2017-03-07 12:00:00"

12.2 Executing the PM Request

Execute the request by setting *enmsPmRequestState* to *started*:

```
snmpset enmsPmRequestState.4 i started
```

```
TNMS-NBI-MIB::enmsPmRequestState.4 = INTEGER: started(3)
```

To check the execution state, read *enmsPmRequestState* and optionally *enmsPmRequestInfo*. In this example the execution is still pending:

```
snmpget enmsPmRequestState.4 enmsPmRequestInfo.4
```

```
TNMS-NBI-MIB::enmsPmRequestState.4 = INTEGER: pending(2)  
TNMS-NBI-MIB::enmsPmRequestInfo.4 = STRING: "Request queued for execution."
```

When the request execution finishes, the resulting data is ready for retrieval:

```
snmpget enmsPmRequestState.4 enmsPmRequestInfo.4
```

```
TNMS-NBI-MIB::enmsPmRequestState.4 = INTEGER: finished(4)  
TNMS-NBI-MIB::enmsPmRequestInfo.4 = STRING: "Execution finished. Total values: 156."
```

An existing PM request may be updated and re-executed any number of times, by setting *enmsPmRequestState* to *started* (see section 13212.4).

12.3 Retrieving the PM Request Results

To get the PM results, first retrieve resulting PMPs by reading the table *enmsPerfMonResultPmpTable*. Relevant rows will have *enmsPmResultPmpReqId* = 4, the PM request index.

Note: Net-SNMP does not offer an easy way to retrieve only the rows for a specific sub-index. The example below shows how to retrieve the entire table, which contains the results for all requests.

snmptable enmsPerfMonResultPmpTable

SNMP table: TNMS-NBI-MIB::enmsPerfMonResultPmpTable

index	ReqId	PmpNumber	NeId	PortId	TPidH	TPidL	NeIdName	ObjLocation	Name	Location	Direction	RetrievalTime
PeriodEndTime												
(...)												
3.80	3	80	105	112010005	0	112010015	7090_100CEM_14	ge.1.5	7090M-RMON-RX	1	2	"2017-03-07 15:16:51" "
3.81	3	81	105	112010005	0	112010015	7090_100CEM_14	ge.1.5	7090M-RMON-RX	1	2	"2017-03-07 15:16:51" "
3.82	3	82	105	112010005	0	112010015	7090_100CEM_14	ge.1.5	7090M-RMON-RX	1	2	"2017-03-07 15:16:51" "
4.1	4	1	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.2	4	2	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.3	4	3	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.4	4	4	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.5	4	5	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.6	4	6	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.7	4	7	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.8	4	8	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.9	4	9	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.10	4	10	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.11	4	11	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.12	4	12	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
4.13	4	13	97	112030004	0	112030014	7090_320M_7	ge.3.4	7090M-RMON-RX	1	2	"2017-03-07 15:16:49" "
(...)												

Next, retrieve counter values for the resulting PMPs, by reading the table *enmsPerfMonResultValueTable*. Again, relevant rows will have *enmsPmResultValReqId* = 4. Fields *enmsPmResultValReqId* + *enmsPmResultValPmpNumber* point to the PMP entries in *enmsPerfMonResultPmpTable*.

snmptable enmsPerfMonResultValueTable

SNMP table: TNMS-NBI-MIB::enmsPerfMonResultValueTable

index	ReqId	PmpNumber	Number	Param	Value	Unit	Status
(...)							
3.84.6	3	84	6	pOVERSIZE	0.0	1	
3.84.7	3	84	7	pBC	0.0	1	
3.84.8	3	84	8	pMC	876.0	1	
4.1.1	4	1	1	pOK	0.0	1	
4.1.2	4	1	2	pBAD	0.0	1	
4.1.3	4	1	3	oALL	0.0	1	
4.1.4	4	1	4	pUC	0.0	1	
4.1.5	4	1	5	fCSERR	0.0	1	
4.1.6	4	1	6	pOVERSIZE	0.0	1	
4.1.7	4	1	7	pBC	0.0	1	
4.1.8	4	1	8	pMC	0.0	1	
4.2.1	4	2	1	pOK	0.0	1	
4.2.2	4	2	2	pBAD	0.0	1	
4.2.3	4	2	3	oALL	0.0	1	
4.2.4	4	2	4	pUC	0.0	1	
(...)							

12.4 Re-executing the PM Request

To reuse a PM request, update its fields (typically the start/end times) and re-execute it by setting *enmsPmRequestState* again to *started*. The example in this chapter re-executes the previously created PM request, whose index is 4.

In the command below, the start/end times are set using relative times instead of absolute, using the ISO 8601 notation for durations. SNMP NBI automatically converts ISO 8601 durations to absolute timestamps by adding the duration to the current UTC time. Start time “-PT30m” will be converted to current UTC time minus 30 minutes, while end time “P0D” (which is a zero duration) will be converted to the current UTC time (now). This means we are requesting PM data for the last 30 minutes.

See https://en.wikipedia.org/wiki/ISO_8601#Durations for duration syntax information.

Note: only days, hours and minutes can be used when specifying durations. Other granularities are not accepted.

```
snmpset enmsPmRequestStartTime.4 s "-PT30m"
        enmsPmRequestEndTime.4 s "P0D"
        enmsPmRequestState.4 i started
```

```
TNMS-NBI-MIB::enmsPmRequestStartTime.4 = STRING: "-PT30m"
TNMS-NBI-MIB::enmsPmRequestEndTime.4 = STRING: "P0D"
TNMS-NBI-MIB::enmsPmRequestState.4 = INTEGER: started(3)
```

Listing the PM requests, the supplied durations were converted to absolute timestamps:

```
snmptable enmsPerfMonRequestTable
```

SNMP table: TNMS-NBI-MIB::enmsPerfMonRequestTable

index	Name	RowStatus	LastUpdate	Info	State	Type	StartTime	EndTime	
1	ethernetCarrier	1	"2017-03-03 19:47:00"	Request reinitialized.	1	2	"	"	1
2	mpls 7090	1	"2017-03-07 09:36:43"	Request reinitialized.	1	1	"2017-03-06 20:00:00"	"2017-03-06 20:30:00"	1
3		1	"2017-03-07 16:04:22"	Execution finished. Total values: 84.	4	1	"2017-03-07 13:43:31"	"2017-03-07 15:03:31"	1
4	Demo request	1	"2017-03-07 18:29:24"	Execution finished. Total values: 23.	4	1	"2017-03-07 17:59:24"	"2017-03-07 18:29:24"	

12.5 Deleting a PM Request

Remove an unwanted PM request by setting *enmsPmRequestRowStatus* to *destroy(6)*:

```
snmpset enmsPmRequestRowStatus.4 i destroy
```

```
TNMS-NBI-MIB::enmsPmRequestRowStatus.4 = INTEGER: destroy(6)
```

Confirming that the PM request has been deleted:

```
snmpwalk enmsPerfMonRequestTable
```

```
SNMP table: TNMS-NBI-MIB::enmsPerfMonRequestTable
```

index	Name	RowStatus	LastUpdate	Info	State	Type	StartTime	EndTime	
1	ethernetCarrier	1	"2017-03-03 19:47:00"	Request reinitialized.	1	2	"	"	1
2	mpls_7090	1	"2017-03-07 08:36:43"	Request reinitialized.	1	1	"2017-03-06 20:00:00"	"2017-03-06 20:30:00"	1
3		1	"2017-03-07 16:04:22"	Execution finished. Total values: 84.	4	1	"2017-03-07 13:43:31"	"2017-03-07 15:03:31"	1

13

Troubleshooting

The table below proposes solutions for the most common issues when operating with SNMP NBI. Also check TNMS System Event Log for messages related to SNMP NBI events.

Symptom	Possible cause	Solution
The SNMP NBI menu entries in TNMS Client are missing or greyed out.	SNMP NBI not installed in the server	Reinstall TNMS and select the SNMP northbound interface (see 2.5).
	SNMP NBI license not installed	Install SNMP NBI license (see 2.5).
The “Enable SNMP northbound interface” checkbox (SNMP NBI system settings) is greyed out.	An SNMP NBI license has been installed, but the server has not been restarted yet.	Restart TNMS server (see 2.5).
No response from SNMP NBI or timeout error.	SNMP NBI not installed or not licensed.	Install SNMP NBI or its license (see 2.5).
	Timeout configured on the SNMP manager is too low (see 0).	Increase the timeout value configured on the SNMP manager.
	Incorrect SNMP agent address configured on the SNMP manager.	Make sure that the SNMP agent address configured on the SNMP manager corresponds to the TNMS server machine.
	Incorrect SNMP agent port configured on the SNMP manager.	Make sure that the SNMP agent port configured on the SNMP manager matches the SNMP NBI listening port (see 3.1).
	The source address of the SNMP requests is not in the list of allowed manager addresses for the SNMP user.	Add the SNMP manager's IP address (or addresses, if it has multiple network interfaces) to the list of allowed manager addresses in the SNMP NBI user configuration (see 3.2.2).
	SNMP user doesn't have the appropriate permission.	Change permission of the SNMP user to 'Read' or 'Read/Write' as appropriate (see 3.2.2).
	Incorrect user data or SNMP protocol version configured on the SNMP manager.	Make sure the SNMP manager is using the correct user (SNMPv2 community or SNMPv3 user) and protocol version, as configured in the SNMP NBI user configuration (see 3.2.1).

Symptom	Possible cause	Solution
	SNMP NBI could not bind to the listening port.	Make sure that the configured listening port is not being used by any other service or application on the TNMS server machine (see 3.1). You may use a utility such as 'netstat' to list the ports on which the server computer is listening.
	Network connectivity problem.	Check network connectivity between the TNMS server machine and the SNMP manager machine. Confirm that the ports chosen for SNMP communication, in particular, the SNMP NBI listening port, are not blocked by any firewall.
SNMP error "No such name" received.	The requested object doesn't exist in the MIB. Usually occurs with GET requests.	Check if the requested OID is valid and belongs to the SNMP NBI MIB. Verify if the SNMP manager is trying to get a nonexistent table value (that is, the table is valid, but does not contain any value for the index specified in the OID).
	The SNMP manager is still using TNMS Core's SNMP Proxy MIB definition.	If the SNMP manager was previously configured to access TNMS Core's SNMP Proxy, then some adaptations are needed before redirecting it to the SNMP NBI. See 2.4.
	The SNMP manager is accessing the wrong SNMP agent (for example, a TNMS Core's SNMP Proxy installation).	Reconfigure the SNMP manager to access SNMP NBI instead.
SNMP error "Authentication error" received.	Incorrect user data or SNMP protocol version configured on the SNMP manager.	Make sure the SNMP manager is using the correct user authentication details, as configured in the SNMP NBI user configuration (see 3.2.1). This frequently occurs with SNMPv3, so confirm the user name, the authentication and privacy protocols, and the corresponding passwords.
SNMP error "Too big" received.	The response to the request does not fit in a single SNMP packet (see 4.5.4). Typically occurs when the SNMP manager requests too many OIDs in the same operation, or the max-repetitions value for a GETBULK operation is too high.	Split the failing GET / GETNEXT / GETNEXT operations into two or more requests. Use a lower max-repetitions value for GETBULK requests.

Symptom	Possible cause	Solution
No traps/informs received from SNMP NBI.	SNMP manager address not added to the trap destination list.	Add the SNMP manager address to the trap/inform destination list of the appropriate SNMP NBI user (see 3.2.3).
	Incorrect trap destination port.	Make sure the destination port of the traps/informs matches the port on which the SNMP manager is waiting for traps (see 3.2.3).
	Incorrect SNMP user data.	Make sure the SNMPv2 community or SNMPv3 user for which the traps/informs are sent is configured in the SNMP manager. For SNMPv3 users, check the user name, the authentication and privacy protocols, and the corresponding passwords.
	The SNMP manager is not listening to the trap receiving port.	Make sure the SNMP manager is really listening for traps/informs on the configured port.
	Network connectivity problem.	Check network connectivity between the TNMS server machine and the SNMP manager machine. Confirm that the ports chosen for SNMP communication are not blocked by any firewall.
SET operation returns an error. (See below for cases specific to PM/OPM request handing.)	SNMP user doesn't have write permission.	Change permission of the SNMP to 'Read/Write' (see 3.2.2).
	The target MIB object is not writable.	Check the object the SNMP manager is trying to set.
	The type of the value in the SET request is not compatible with the MIB object.	Use the correct data type.
SET operation returns error when creating a PM/OPM request. (See more possible causes below.)	The instance identifier is already in use.	Make sure the instance identifier is not used by another row in the PM/OPM request table. An auto-increment variable may be used to generate instance identifiers (see 10.2 / 11.2).
	RowStatus attribute unset or set to invalid value.	When adding a new request, set the RowStatus attribute to <i>createAndGo</i> .
	The maximum number of PM / OPM requests (5000) has been reached.	Delete unused requests, or reuse an existing request by updating its attributes.

Symptom	Possible cause	Solution
SET operation returns error when creating or updating a PM/OPM request.	Invalid value assigned to an attribute.	Check the values supplied to the attributes – non-existent enumeration value, malformed dates, malformed object identifiers, etc.
	Some attribute value was set to a value inconsistent with other values.	<p>Make sure the request attributes stay consistent with each other. Examples:</p> <ul style="list-style-type: none"> When setting the request filter type to <i>port</i>, the filter value must be a port identifier; the SNMP manager must change the filter type <i>and</i> the filter value at the same time, in a single SET command. When changing a PM request type to <i>history</i>, the start/end times must not be empty, otherwise the manager must set the request type and the start/end times simultaneously, in the same SET command.
SET operation returns error when updating the state of the PM/OPM request.	The state transition is not valid.	See section 10.3 / 11.3 for possible state transitions.
SET operation returns error when deleting a PM/OPM request.	The request is in state <i>pending</i> , <i>started</i> or <i>cancelling</i> .	It is only possible to delete requests that are in idle state (<i>created</i> , <i>finished</i> , <i>failed</i> and <i>cancelled</i>) – the manager must wait for the request to reach one of such states.

Table 89 SNMP NBI troubleshooting table