



Fault Manager

North Bound Interface (NBI)

Developer's Guide

March 2018

Copyrights

The Motorola Solutions products described in this document may include copyrighted Motorola Solutions computer programs. Laws in the United States and other countries preserve for Motorola Solutions certain exclusive rights for copyrighted computer programs. Accordingly, any copyrighted Motorola Solutions computer programs contained in the Motorola Solutions products described in this document may not be copied or reproduced in any manner without the express written permission of Motorola Solutions.

Furthermore, the purchase of Motorola Solutions products shall not be deemed to grant either directly or by implication, estoppel or otherwise, any license under the copyrights, patents or patent applications of Motorola Solutions, except for the normal nonexclusive, royalty-free license to use that arises by operation of law in the sale of a product.

Disclaimer

Please note that certain features, facilities and capabilities described in this document may not be applicable to or licensed for use on a particular system, or may be dependent upon the characteristics of a particular mobile subscriber unit or configuration of certain parameters. Please refer to your Motorola Solutions contact for further information.

Trademarks

Motorola Solutions, the Motorola Solutions logo, and all other trademarks identified as such herein are trademarks of Motorola Solutions, Inc. All other product or service names are the property of their respective owners.

Copyrights

© 2018 Motorola Solutions, Inc. All rights reserved.

No part of this document may be reproduced, transmitted, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, without the prior written permission of Motorola Solutions, Inc.

European Union (EU) Waste of Electrical and Electronic Equipment (WEEE) directive



The European Union's WEEE directive requires that products sold into EU countries must have the crossed out trash bin label on the product (or the package in some cases).

As defined by the WEEE directive, this cross-out trash bin label means that customers and end-users in EU countries should not dispose of electronic and electrical equipment or accessories in household waste.

Customers or end-users in EU countries should contact their local equipment supplier representative or service centre for information about the waste collection system in their country.

Revision History

Date	Version	Author	Description
Nov-2007	0.1	Motorola Global Content Development	Initial publication
Jan-2011	0.2		Rebranding. Added more details on the provided services
May-2011	0.3		Changed File-Based Synchronization process mechanism section.
Apr-2013	0.4		Added Send test trap functionality
Sept-2015	0.6		Rebranding. Some minor updates and mib file rename. Versions alignment.
March-2016	0.7		NBI Enhancements
April-2016	0.8		Added Entities to the REST Interface
June-2016	1.0		Minor updates
July-2016	1.1		Added info about certificates for REST interface
March-2018	1.2		Mib files cleanup

This page is intentionally blank

Table of Contents

1. Introduction	5
1.1. Overview	5
1.2. Scope	5
1.3. Terms and Acronyms	5
1.3.1. Terms	5
1.3.2. Acronyms	5
2. FM Northbound Interface Overview	7
2.1. Software Architecture	7
2.1.1. SNMP Interface	7
2.2. NBI Services	8
3. Configuration	9
3.1. FM Configuration for NBI Services	9
3.2. NMS Configuration for NBI Service	9
3.3. NMS Registration	10
3.3.1. Multiple NMS Support	10
3.3.2. Send test trap functionality	10
4. Operations	11
4.1. General FM Operation Concepts	11
4.1.1. Sequence Tag and Correlation Tag	11
4.1.2. File Download Procedure	11
4.1.3. Operation Status	11
4.1.4. Operation Concurrency	12
4.1.5. Operation Status Return and Notification	13
4.1.6. Date Time Stamp	13
4.2. Notification Service	14
4.2.1. NBI Notifications	14
4.2.1.1. Network Events Notification	15
4.2.1.2. Management Operation Event Notification	16
4.2.2. Notification Persistence	17
4.3. Fault Management Service	17
4.3.1. Quick Event Synchronization	17
4.3.2. File Based Event Synchronization	19
4.3.3. Alarm Synchronization	23
4.3.4. FM-NMS Communication Link Management	24
4.4. SNMP Definition - NBI Notifications	27
4.4.1. NBI Network Event Trap Definition	27
4.4.2. NBI Management Event Trap Definition	31
4.5. SNMP Definition - Management Operations	33
4.5.1. Quick Synchronization SNMP Interface Definition	33
4.5.2. Alarm Synchronization SNMP Interface Definition	44
4.5.3. Event Synchronization SNMP Interface Definition	49
4.5.3.1. Filtering events – time constraints	51
5. REST API (not supported in MOTOTRBO)	59
5.1. Authentication	59
5.2. Inventory service	59
5.2.1. Inventory query	60
5.2.2. Detailed Managed Resource query	60
5.2.3. Search query	60
5.2.3.1. Search based on class name <code>nbiClass</code>	60
5.2.3.2. Search based on parent identifier <code>nbiSourceDevice</code>	60
5.2.3.3. Search based on resource type <code>nbiType</code>	61

5.2.3.4.	Search for resources which lost communication or had successful synchronization some time ago or in specified time range	61
5.2.4.	MOSCAD RTU entities inventory and management.....	61
5.2.4.1.	Get specific entities from MOSCAD RTU LMR	61
5.2.4.2.	Get specific entity from MOSCAD RTU LMR	62
5.2.4.3.	Update value for Digital Output Fault Object in RTU.....	62
Appendix A	NBI Management Operation Return Status Code	69
A.1	SNMP SET and GET Request Response Status Code	69
A.2	SNMP Request Validation Return Codes.....	70
Appendix B	NBI Notifications	71
B.1	NBI Supported Events.....	71
B.2	NBI Supported Management Events	72
B.3	Reason Code of NBI Management Notifications	72
B.4	Additional Information of NBI Management Notifications.....	73
B.5	HTTP status codes using during synchronization file download.....	74
Appendix C	File Format.....	76
C.1	File Name Convention	76
C.2	XML Schema	78
C.3	Sample Alarm Summary XML File	81
C.4	Sample Event Summary XML File.....	82
Appendix D	SNMP v3 USM Configuration	84
Appendix E	ASCII codes (7-bit) table.....	85
Appendix F	NBI Fields mapping	86
Appendix G	REST Error Codes.....	87

1. Introduction

1.1. Overview

Fault Manager is an element of Management System which provides reliable fault management services for the radio systems. One of the salient features of the Fault Manager is to receive events from network elements, process them, correlate events to alarms and present them to the operator. FM supports an interface to send these events to North Bound Managers (like Network Management Systems). In addition to event notification interface, FM also supports interface to access information in its data-store. This interface is termed as North Bound Interface (NBI).

1.2. Scope

This document describes various services supported on the NBI and their interface specifications. Users of the NBI, like NMS or MoM, can use this document to implement an interface to manage FM(s).

1.3. Terms and Acronyms

1.3.1. Terms

File Id File identifier (UUID form) used in file-based synchronization to find specific file on FM server during stream opening

Receiver Registered NMS that receives notification from FM.

NMS (Network Management System), Manager of Managers and Receiver are used interchangeably in this document

1.3.2. Acronyms

EMS	Element Management System
FM	Fault Manager which can be: Unified Event Manager, Dimetra Unified Event Manager or Prioritization Service Fault Manager
JMX	Java Management Extension
MIB	Management Information Base
MoM	Manager of Managers
NBI	North Bound Interface
NE	Network Element
NMS	Network Management System
OID	Object Identifier
SNMP	Simple Network Management Protocol
SNMPv3	Simple Network Management Protocol version 3
SSC	System Support Center
UML	Unified Model Language

USM	User-based Security Model
VACM	View-based Access Control Model
XML	Extensible Markup Language

2. FM Northbound Interface Overview

FM provides an interface to receive, request and access management data for a Network Management System or Manager of Managers to use. This section provides an overview of the FM North Bound Interface Architecture.

2.1. Software Architecture

The FM provides the north bound interface functionality using JMX (Java Management Extensions) technology. The management features are provided by a software component, known as Agent, on the FM. The NBI agent on the FM follows the JMX specification. It supports SNMP protocol interface for NMS to access, request and receive information from the FM. Additionally inventory of Network Elements managed by FM are provided by REST interface, more at section 5.

High level software architecture of FM NBI is shown in **Figure 1 FM NBI Software Architecture**.

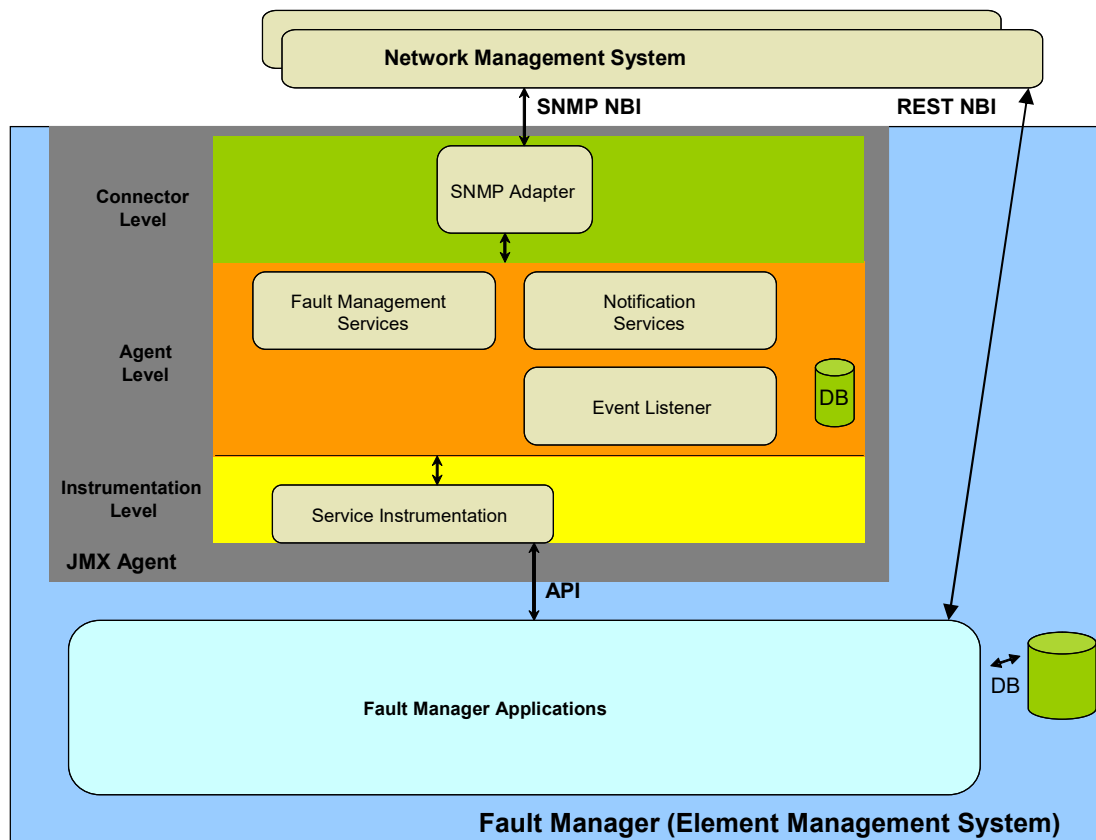


Figure 1 FM NBI Software Architecture

2.1.1. SNMP Interface

SNMP NBI supports notifications to registered NMSs or Northbound managers and also supports access to management data. The NBI agent supports SNMPv3 and uses SNMPv3 User-based Security Model (USM) for secure communication with the NMSs. The supported SNMPv3 USM follows RFC 3414. Even though the USM configuration (credentials) on the FM can be user defined, the FM will restrict the NBI to support only the pre-defined user credentials. All managers shall use these pre-defined USM credentials for SNMPv3 communication. The supported credentials are defined in Appendix D- SNMP v3 USM Configuration

2.2. NBI Services

FM NBI agent supports the following management services:

1. **Notification Service:** This service includes notification of network events and management events to the registered managers. The network events are the events originated at the NEs or the FM regarding network behavior. The management events are events originated at the FM to provide feedback on the progress of management services like Synchronization etc. The NBI supports SNMPv3 TRAP notifications with USM support for secure communication.
2. **Fault Management Service:** FM provides NBI fault management services for NMS to supervise the FM-NMS communication link and to synchronize events and alarms with FM. The fault management services are
 - a. **Quick Synchronization:** FM NBI provides a way to replay the recently notified events as traps to NMS. NMS has to request the events that need to be replayed. This service can be used to quickly synchronize missed events and hence termed as Quick Synchronization service.
 - b. **File-Based Synchronization:** FM provides a way to synchronize events and alarms using a bulk transfer mechanism. Based on the input request from NMS, FM generates an event data file or active alarm file locally on the FM. FM sends a notification to NMS with information on the ID of the generated file. NMS is responsible for accessing the generated file by connecting to jsp page.
 - c. **FM-NMS Communication Link Supervision:** Fault Management service also includes providing access to health of FM and access to information on NBI forwarded notifications. This health or notification information can be used to monitor or supervise FM-NMS management link by the NMS.

3. Configuration

3.1. FM Configuration for NBI Services

All management services on FM NBI are supported using secure SNMPv3 protocol. FM is pre-configured with default SNMPv3 configuration for Customer NMS or Motorola SSC NMS to use. Operators can configure authentication and privacy SNMP credentials for FM to use when providing these NBI services. Refer to North Bound Interface section of the Fault Manager manual for details on FM's NBI SNMPv3 configuration. Refer to Appendix-D for the default SNMPv3 configuration supported on FM.

Engine Id for the FM NBI SNMP engine (for communication with Motorola NMS and non-Motorola NMS) is configured internally by the FM software based by default on MAC Address of the Ethernet Interface of the FM and follows the derivation suggested in RFC3411.

As per RFC,

- The first four octets are set to the binary equivalent of the Motorola's SNMP management private enterprise number as assigned by the Internet Assigned Numbers Authority (IANA). That is, for Motorola the management private enterprise number is 161. Hence the first four octets will be '000000A1' (Hex).
- The very first bit is set to 1. The above value changes to be '800000A1' (Hex)
- The fifth octet is set to 03 which specify that the remaining octets are based on MAC Address of the device.
- The remaining octets (sixth octet and onwards) are the value of MAC Address of the Ethernet Interface on the FM.

For Example, If MAC Address of the Ethernet interface on FM is 0A 1B 2C 3D 4E 5F and Motorola's SNMP management private enterprise number is 161 (A1 Hex), the engine Id is
Engine Id (Hex) = 80 00 00 A1 03 0A 1B 2C 3D 4E 5F

FM is also configured with the NBI port to receive and service requests. By default, NBI port on FM is configured as 8001.

3.2. NMS Configuration for NBI Service

For NMS to use the FM NBI, NMS operator needs to configure the following information on the NMS:

- **SNMP credentials of the FM:** This includes SNMP User Name, FM Engine Id, Authentication protocol setting and pass phrase and Privacy protocol setting and pass phrase. These SNMP credentials of the FM are obtained by the NMS operator using out-of-band mechanism. That is, FM operator has to share FM's NBI credentials with the NMS operator.
The FM Engine Id is derived based on the MAC address of the FM's Ethernet interface as specified in section 3.1. Contact Motorola System Support to obtain the MAC address of the FM's Ethernet interface,
- **IP Address of the FM:** this address is based on network configuration
- **FM Port to send SNMP requests:** FM NBI Port to receive SNMP requests from NMS is configured as 8001.

3.3. NMS Registration

To use the FM NBI services, NMS or Manager of Managers have to register with the FM. The NMS Registration on FM includes information on:

- **NMS IP Address** – The NMS IP Address to which the 'Events' shall be forwarded.
- **Port Number** – The Port number at which the NMS listens for SNMP traps (The default SNMP trap listening port is considered as 162, unless specified during registration)
- **UserName** – The SNMPv3 USM user name that is used for communication with this NMS.
- **Operational State** – This state defines whether the NBI for the registered NMS is currently enabled or disabled. If it is disabled, the traps are not forwarded and the event/alarm synchronization services are not available to the NMS.

FM persistently stores the manager registration information.

Refer to North Bound Interface section of the Fault Manager manual for details on NMS Registration.

3.3.1. Multiple NMS Support

FM NBI supports multiple NMS registration for usage of NBI services. FM NBI shall support a maximum of four NMSes. Of the four supported NBIs, two are reserved for Motorola System Support Center.

3.3.2. Send test trap functionality

FM NBI allows to verify if NMS [is available](#). It's possible to send test trap using **Send test trap** button. The test trap should be visible on the NMS application registered as NBI client. If user wants to send test trap directly from **NBI Configuration** window (fig. 3-10) just selects registered NMS server and click **Send test trap** button.

With details, the test trap has such defined varbinds:

varbind name	value
nbiEventId	„testTrapIdentifier”
nbiEventRecordId	0
nbiSourceAgentIP	“testTrapReportingAgent”
nbiEventType	256 (Informational Event)
nbiEntityId	„testTrapEntity”
nbiEntityName	„testTrapEntityName”
nbiManagedObjectUserName	„testTrapManagedResource”
nbiSeverity	7 (Info)
nbiSourceDevice	„testTrapSource”
nbiDetectionEventTime	TIMESTAMP
nbiSourceEventTime	TIMESTAMP
nbiText	„NBI connection test – timestamp: TIMESTAMP”

NOTE: This option is not available on Dimetra Fault Manager.

4. Operations

This section provides details on FM Management operations that are used by NMS to interface with FM. Some general FM supported operation concepts are introduced in the following sections.

4.1. General FM Operation Concepts

4.1.1. Sequence Tag and Correlation Tag

All NBI management notifications are required to include a sequence tag that uniquely identifies the message and a correlation tag that can be used to correlate two or more relevant messages. The correlation tag is also used as a Job Identifier. That is, throughout a management operation all requests and responses shall use the same correlation tag. This means that the originator of the Job (management operation) should assign a correlation tag and send it in the operation request. Other use of the sequence tag and correlation tag is to identify duplicate requests or duplicate notifications. FM NBI has the following rule:

Rule 1. Sequence Tag and Correlation Tag for FM NBI Notification

It is recommended for NMS to send the correlation and sequence tag as part of management operation requests (event/alarm file requests). If the tags are not included in the request, NMS is responsible to identify the notifications related to the operation request.

4.1.2. File Download Procedure

For management operation involving file transfers, NMS is responsible for transferring files from the data server (FM). In response to the operation request that involves file generation, FM NBI generates the file and sends a notification that includes file ID. To download the file, NMS needs to open a stream to the specific file by using the URL returned by the FM as a part of the response. The connection must be established by means of the secure https protocol (non secure http in case of Dimetra FM). If download mechanism on NMS site honors "Content-Disposition" http header, file will be automatically saved and named according to the name convention specified in Appendix C-1. Dimetra FM supports also File Transport Protocol for file transfer.

4.1.3. Operation Status

In addition to management notification to registered NMS during various stages of the management operation, FM NBI also supports operation status query. NMS can query the operation status to monitor the progress. The generic operation status model supports following values:

- **inProgress**: the operation is still in progress;
- **inQueue**: the operation request is in the waiting queue as defined in operation concurrency matrix (Table 1);
- **rejected**: the operation request is rejected as defined in operation concurrency matrix (Table 1);
- **done**: the operation is successfully executed;
- **failed**: the operation fails to complete.

Rule 2. Operation Status Support

FM NBI maintains the progress status of every management operation.

Refer to NBI-FM-MIB for SNMP support for operation status.

4.1.4. Operation Concurrency

Operation concurrency or synchronization is required on FM as an overload control mechanism and to efficiently use system resources to achieve better performance.

Management operation synchronization is used to ensure synchronous accesses of concurrent operations to NMS in a multiple-NMS environment. Following are rules for handling operation synchronization:

Rule 3. Write Operation Serialization

FM NBI shall ensure only one write operation is performed on a given management data. If more than one write operation is issued on the same management data at the same time, FM will serialize the write operations.

Rule 4. Operation request Queuing

FM NBI shall support the capability to queue the operation requests that are blocked due to operation synchronization restriction. FIFO rule shall be used to resume the queued operations immediately after the blocking operation is complete.

Rule 5. Duplicate Operation Request Rejection

FM NBI shall support the capability to reject the duplicate operation requests. The duplicate requests are identified using the combination of Job Id (correlation tag) and Sequence tag.

Rule 6. Dependent Operation Request Rejection

NBI agent shall support the capability to immediately reject operation requests if their dependent operations are still in progress. The operation dependencies shall be predefined.

The following concurrency matrix summarizes all rules NBI agent shall verify before handling concurrent operations. The rules are applied to operations for each receiver (NMS). The operations in columns are in progress while the ones in rows are incoming requests. Following are the three options to handle incoming operation request while there is one operation in progress:

- **(Y)es:** Incoming operations can be handled in parallel to in-progress operations. Note that NBI agent will ensure access synchronization when both operations try to access the same resource.
- **(Q)ueue:** Incoming operations cannot be handled in parallel to in-progress operations. But they can be queued and processed automatically after in-progress operations are complete.
- **(R)eject:** Incoming operations should be rejected immediately.

Table 1 Management Operation Concurrency Matrix

Note: The operations in the columns are in- progress; The operations in the row are incoming request.	Get	Set	Event Sync Events	Alarm Sync
Get	Y	Y	Y	Y
Set (on the same attribute)	Y	R	Y	Y
Event Sync (duplicate requests only)	Y	Y	R	Y
Alarm Sync	Y	Y	Y	R

4.1.5. Operation Status Return and Notification

FM NBI supported management operations are asynchronous. FM NBI agent will return the status when it completes an asynchronous operation.

Rule 7. Asynchronous Operation Return

FM NBI agent will report asynchronous operation status using management notification.

It is recommended to set a wait timer at NMS to receive the operation status notifications. In some cases, NMS can miss notifications. Some of the possible causes for NMS to miss notifications are:

- Slowness of processing an operation request due to unexpected amount of information to process or due to resource contention.
- Management communication link failure where the operation is complete but the notification fails to reach NMS.
- Software defect.

FM NBI will support operation status query as specified in Section 4.1.3 Operation Status. NMSs shall follow the recommendation stated below to handle missing notification.

Rule 8. Operation Status Query

It is recommended that NMSs query the operation status and/or verify other possible causes before re-issuing the operation request when it does not receive operation notification.

4.1.6. Date Time Stamp

Rule 9. UTC Time Support

All management data specifying date/time in FM NBI notifications, data files and requests shall follow Universal Time Clock (UTC) date/time format. For example, Feb 10, 2006 at 11.45.40 AM EDT will be displayed as 2006-02-10T11:45:40-4:0.

The last 4 digits are optional and represent time zone as difference from UTC. EDT is 4 hours behind Coordinated Universal Time (UTC).

4.2. Notification Service

4.2.1. NBI Notifications

Notifications are primarily used by FM NBI to report network events and to report management operation status. The notifications are sent as SNMP traps to the registered receivers. Figure 2 shows that all notifications are sent as SNMPv3 traps.

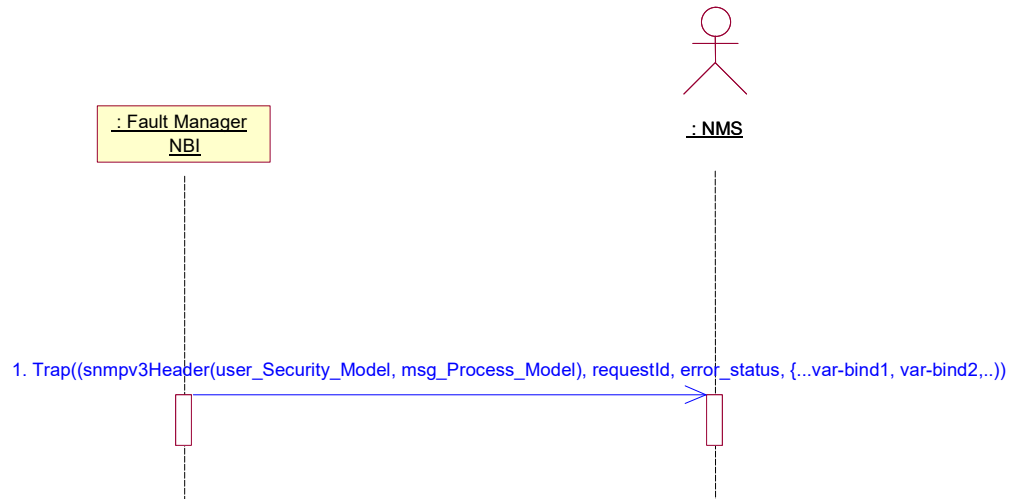


Figure 2 Send Event

FM NBI supports two types of event notifications: Network Event Notification and Management Operation Event Notification.

4.2.1.1. Network Events Notification

These are notifications of events originated at the NEs or the FM regarding the network behavior. For details on SNMP definition of this event, refer to section 4.4.1.

Figure 3 shows the structure of the event and denotes the event sent as an SNMPv3 Trap (denoted by Triggers Send Event in the figure)

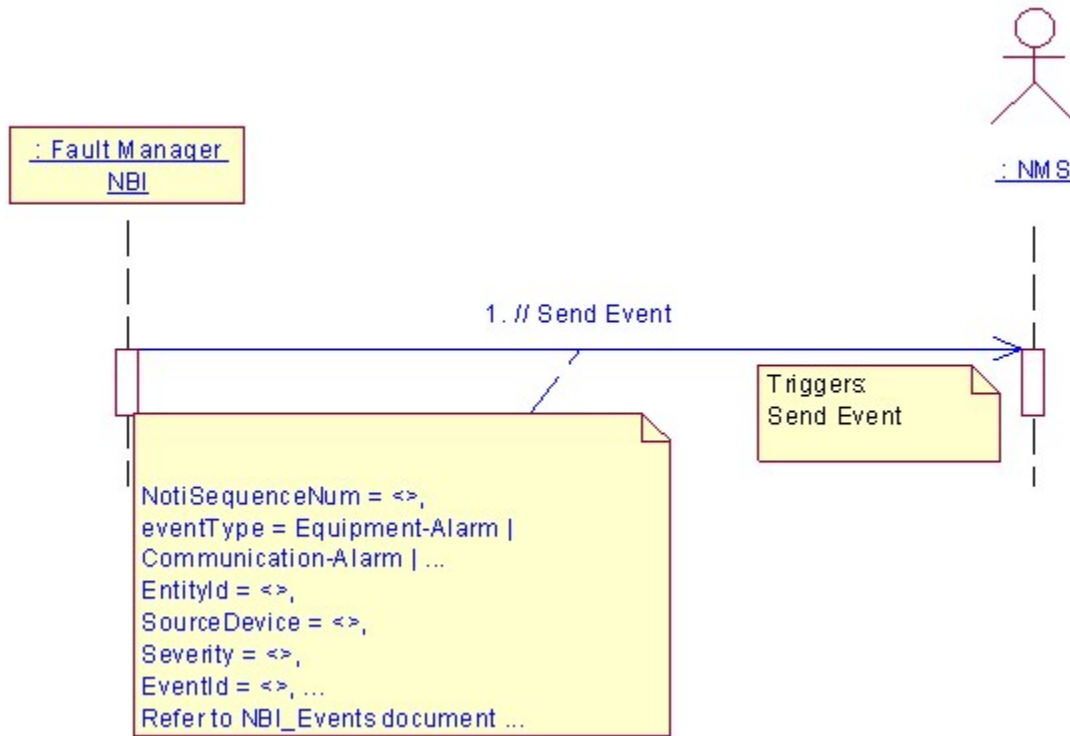


Figure 3 Notify Network Event

4.2.1.2. Management Operation Event Notification

These are notifications of events originated at the FM to support management services like event file generation, alarm file generation etc. that denote the result of the management operation. For details on SNMP definition if this event, refer to section 4.4.2.

Figure 4 shows the structure of the event and denotes the event sent as a SNMPv3 Trap (denoted by Triggers Send Event in the figure).

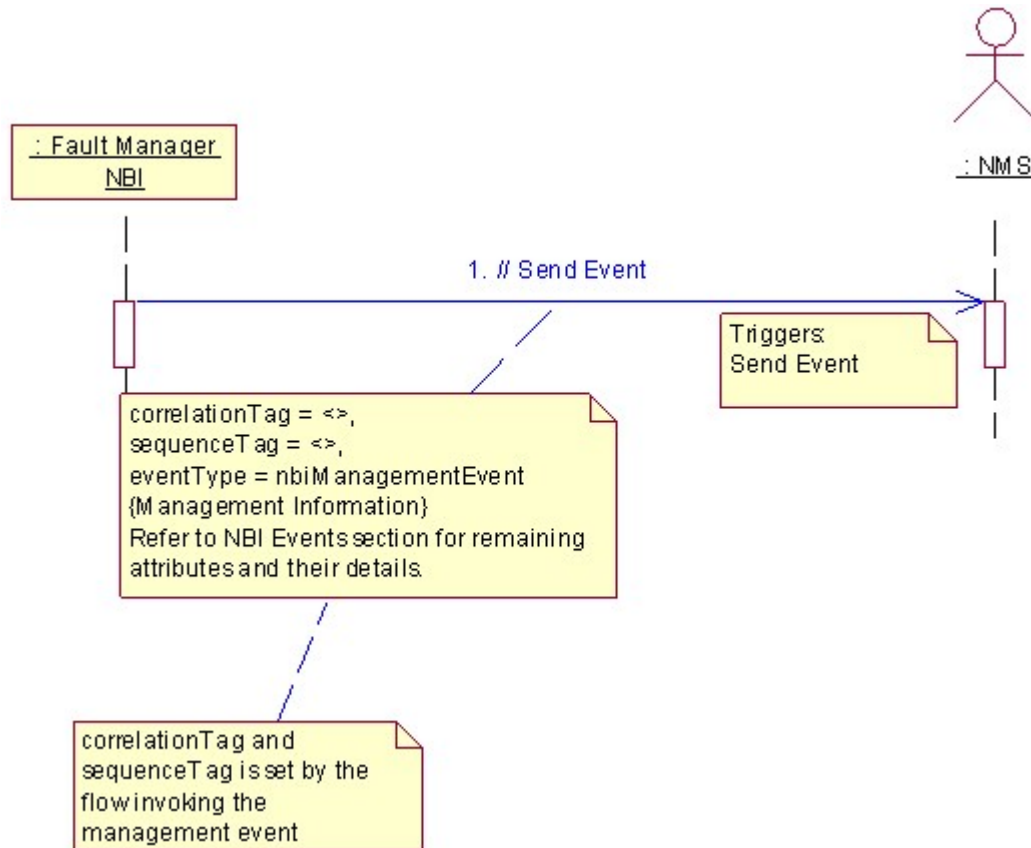


Figure 4 Notify Management Event

All notifications sent to the NMS, except management operation events, are assigned a sequence number. The sequence number is maintained sequential per registered receiver. This sequence number can be used to detect missed notification on the NMS which could lead to a synchronization request.

4.2.2. Notification Persistence

FM NBI supports persistent storage of notifications that are sent to each registered receiver. The default configuration is to store last forwarded notifications up to 300 seconds old. Anyway the persistent store will store maximum last 20,000 number of notifications. The persisted notifications can be retrieved for a desired receiver by qualifying the request with the receiver Identifier (IP Address). FM NBI supports SNMP interface for the notification retrieval. The request can also be qualified with the receiver Id and the notification sequence number. Management Operation notifications are excluded from persistence.

4.3. Fault Management Service

4.3.1. Quick Event Synchronization

With features of Notification Persistence and Notification Sequence Number assignment to all NBI forwarded notifications, NMS can synchronize events by fetching the missed event (may be due to loss of FM-NMS communication) from the Notification Persistence data-store on the FM. NMS can request specific notification by providing the required notification sequence number or a range of notification sequence numbers in the request. FM provides NBI Event Record Identifier in notifications, NMS may also use it for Quick Event Synchronization. The same rules apply as for notification sequence number – single identifiers or range of Event Record Identifiers may be specified in the request. The FM on receiving the request will send asynchronous notifications of the missed events. This mode of synchronization is more efficient than file based synchronization, if the relevant data is available in the persistent data-store. Hence this is termed as “Quick Event Synchronization”. Refer to section 4.5.1 for details on SNMP interface definition for this operation.

Figure 5 below shows the flow of events between FM and NMS on a quick synchronization operation.

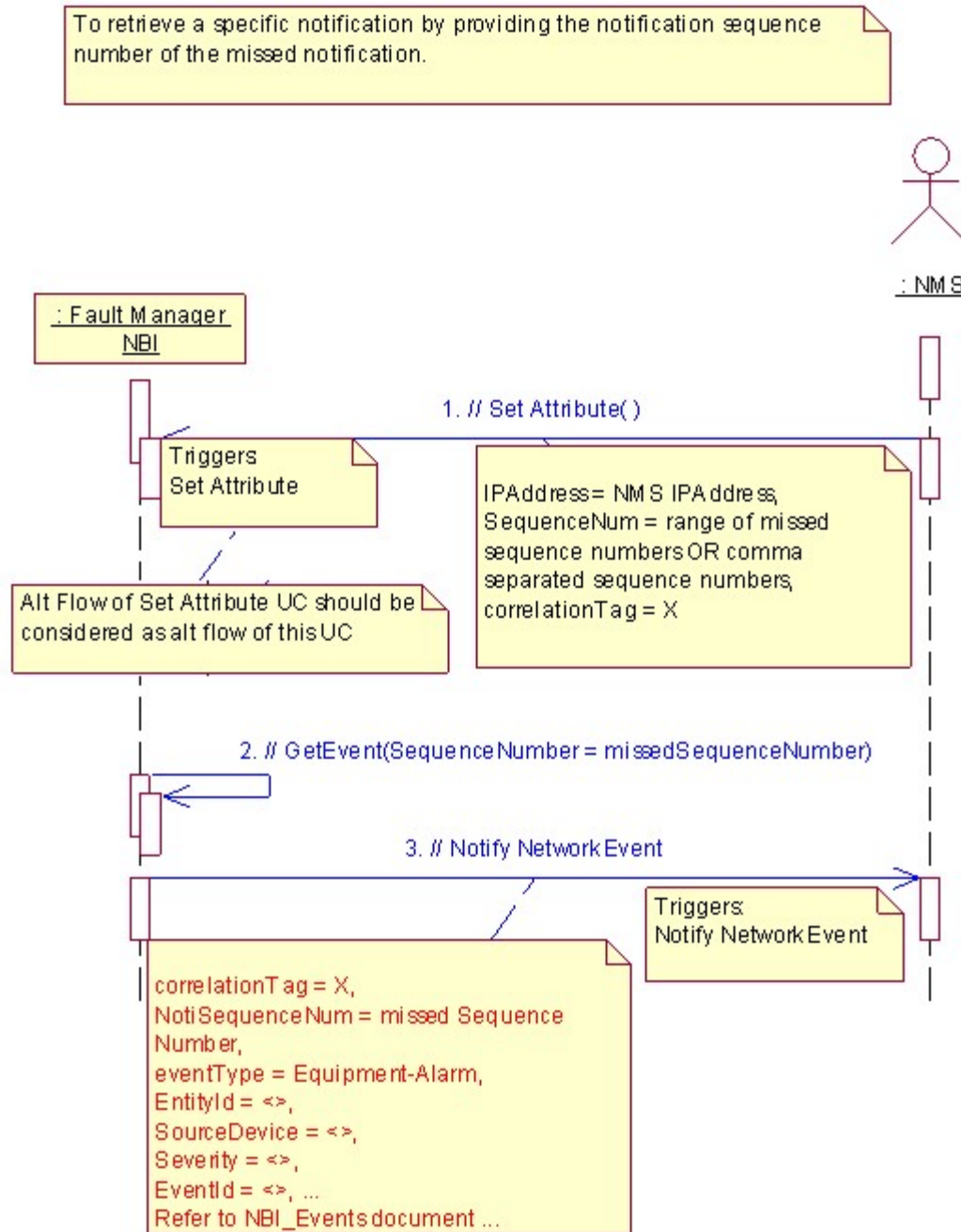


Figure 5 Synchronize Events - Quick Synchronization

4.3.2. File Based Event Synchronization

FM NBI supports generation of event summary file containing the events that reside on the FM NBI persistent storage. NBI supports quantification of events based on the event time stamp. That is, FM can generate event summary file where the events in the file has event time within a requested time period. FM notifies the requestor (NMS), using management events, of the completion of the file creation providing information about file Id.

FM provides also NBI Event Record Identifier thus NMS may request event synchronization based on NBI Event Record Identifiers range. NBI Event Record Identifier might be used to correlate events received on the stream with the events received from the File.

NOTE:

The file ID of the file containing generated events is returned by the Management Operation Event Notification as the value of the **nbiSyncFileToken** attribute. The requestor (NMS) can fetch the file invoking the URL in format described below:

`http(s)://<FM address and port>/jsp/NbiFileDownload.jsp?fileId=<file ID>`

Deprecated: File is accessible directly on FM server, full path to the location is available in returned by the Management Operation Event Notification as the value of the **nbiAdditionalInformation** attribute.

Structure of the file is described in appendix C. If requestor (NMS) download mechanism honors "Content-Disposition" http header file will be saved according to file naming convention described in Appendix C-1.

NOTE:

The default path where generated event synchronization files are stored on FM is **/opt/Motorola/nms/data/NBI/eventsdir** and can be changed by invoking script: **/opt/Motorola/nms/bin/configureNbi.sh** on FM server.

This does not apply to Dimetra FM.

This event summary file generation capability is used by the NMS for event synchronization. It is the responsibility of the NMS to transfer the generated file from FM to the desired destination. By default, the FM guarantees the generated file to be available for 30 minutes after which the file may be automatically purged. The file availability time can be configured on the FM by modifying the NbiFilePurgePolicy using the FM Policy Administration Tool. Refer to the Fault Manager manual for details on FM Policies.

For details on SNMP interface definition, refer to section 4.5.3.

Figure 6 shows the flow of events between FM and NMS for a File Based Event Synchronization.

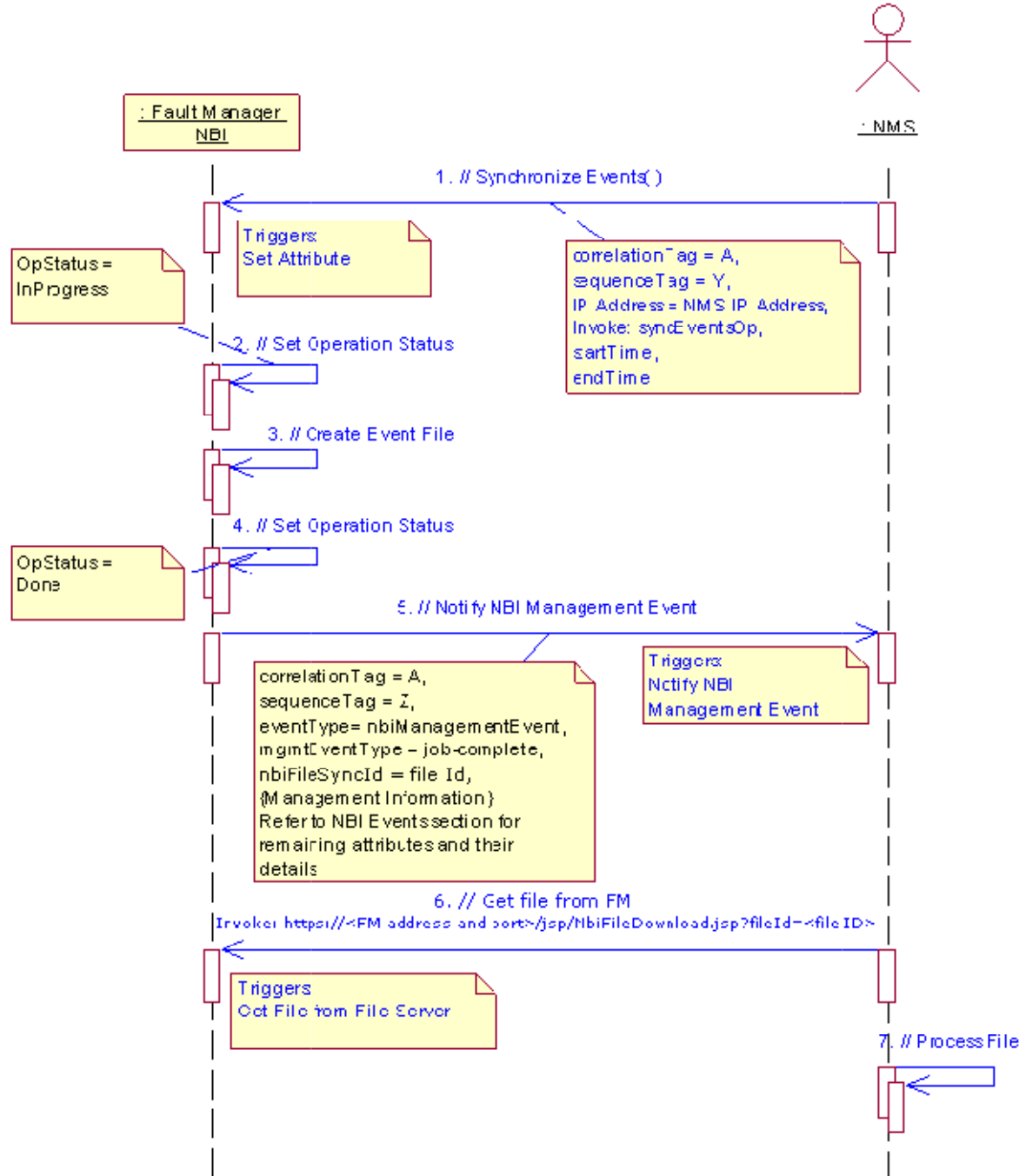


Figure 6 Synchronize Events - File Based Synchronization

Figure 7 shows the expected flow of events when NMS times out waiting for management event from FM that denotes the completion of file generation. The key features shown in this figure is the availability of Operation Status MIB which can be used to track the status of requested operation even if NMS fails to receive management event.

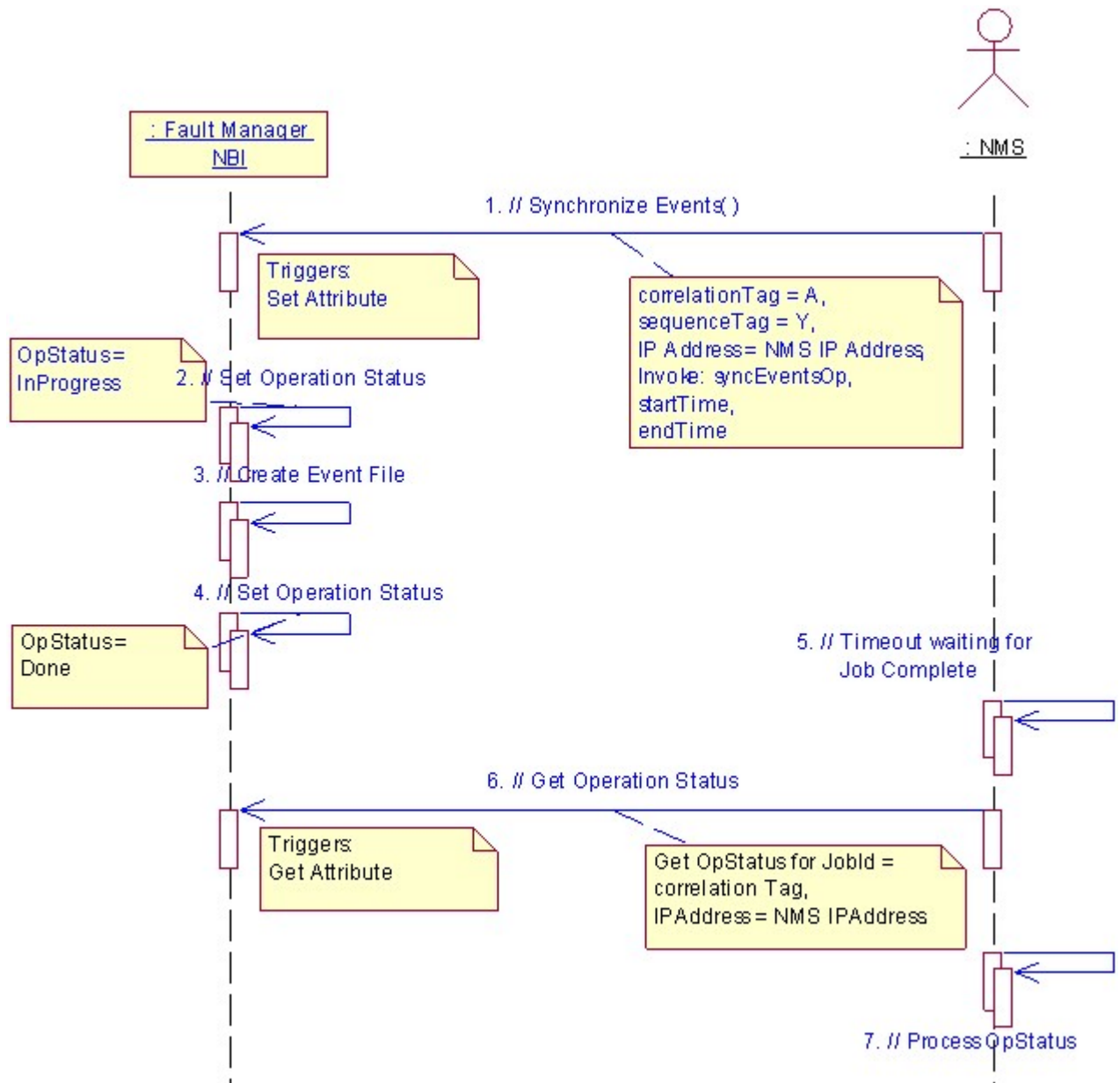


Figure 7 Synchronize Events - File Based Synchronization - Job Completion Timeout

Figure 8 shows the flow of events for a failed File Based Synchronization operation. The failure is denoted by the management event with event type as Job-Aborted and the reason code as the reason for failure. For other possible failure reasons, refer to Appendix B.3. Refer to Appendix B.4 on information on all operations, their possible failure and associated reason codes.

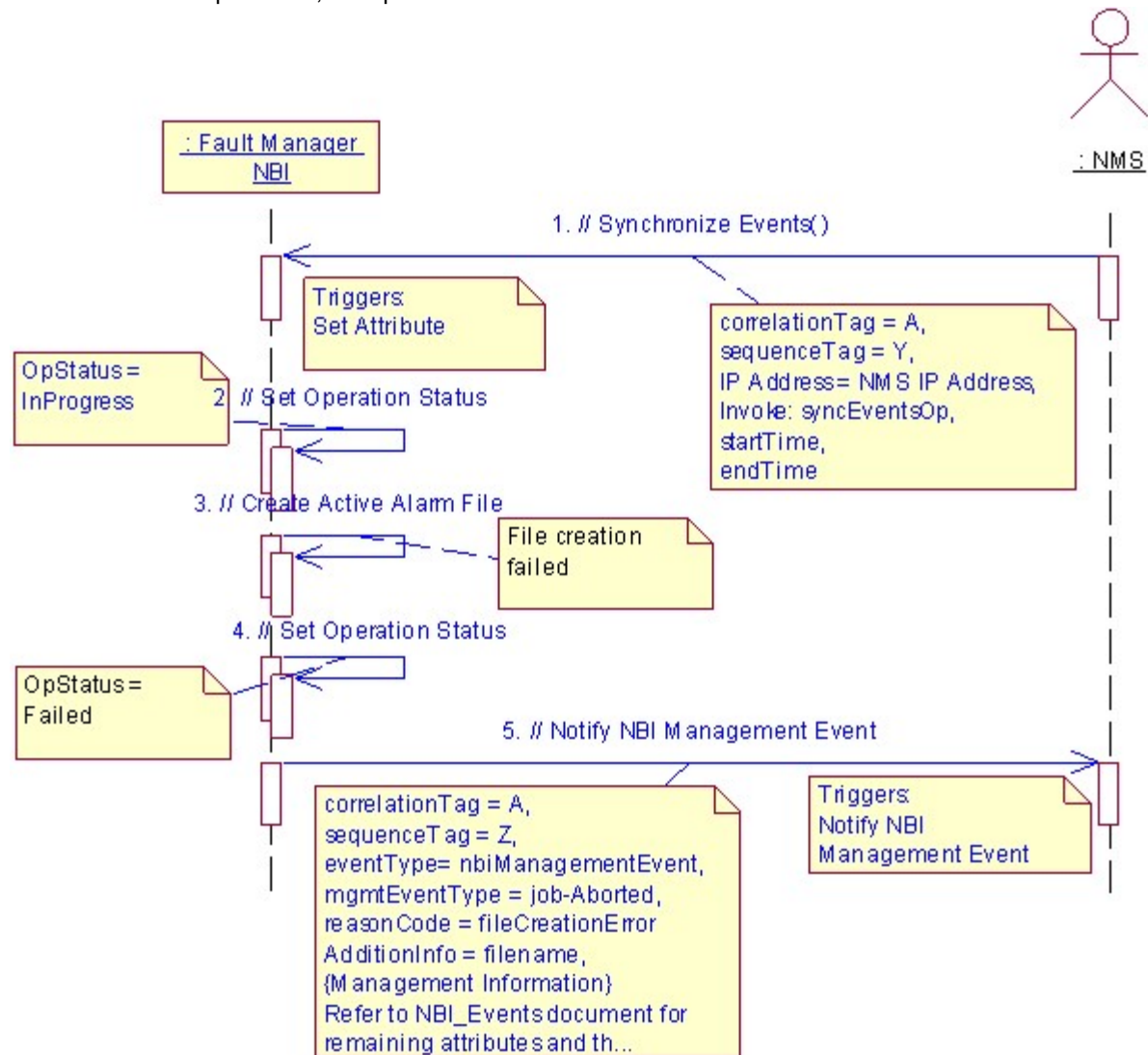


Figure 8 - File Based Synchronization- Job Failed

4.3.3. Alarm Synchronization

FM NBI supports collection and generation of alarm summary file containing alarms that are currently active on the FM. The active alarms in the generated file do not contain alarms that are cleared (CLEAR severity). FM notifies the requestor (NMS), using management events, of the completion of the file creation providing information about file Id.

NOTE:

The file ID of the file containing generated alarms is returned by the Management Operation Event Notification as the value of the **nbiSyncFileToken** attribute. The requestor (NMS) can fetch the file invoking the URL in format described below:

`http(s)://<FM address and port>/jsp/NbiFileDownload.jsp?fileId=<file ID>`

Deprecated: File is accessible directly on FM server, full path to the location is available in returned by the Management Operation Event Notification as the value of the **nbiAdditionalInformation** attribute.

Structure of the file is described in appendix C. If requestor (NMS) download mechanism honors "Content-Disposition" http header file will be saved according to file naming convention described in Appendix C-1.

NOTE:

The default path where generated alarm synchronization files are stored on FM is **/opt/Motorola/nms/data/NBI/alarmsdir** and can be changed by invoking script: **/opt/Motorola/nms/bin/configureNbi.sh** on FM server.

This does not apply to Dimetra FM.

This alarm summary file generation capability is used by the NMS for alarm synchronization. It is the responsibility of the NMS to transfer the generated file from FM to the desired destination. By default, the FM guarantees the generated file to be available for 30 minutes after which the file may be automatically purged. The file availability time can be configured on the FM by modifying the **NbiFilePurgePolicy** using the FM Policy Administration Tool. Refer to the Fault Manager manual for details on FM Policies.

For details on SNMP interface definition, refer to section 4.5.2.

Figure 9 shows the flow of events between FM and NMS for a File Based Alarms Synchronization.

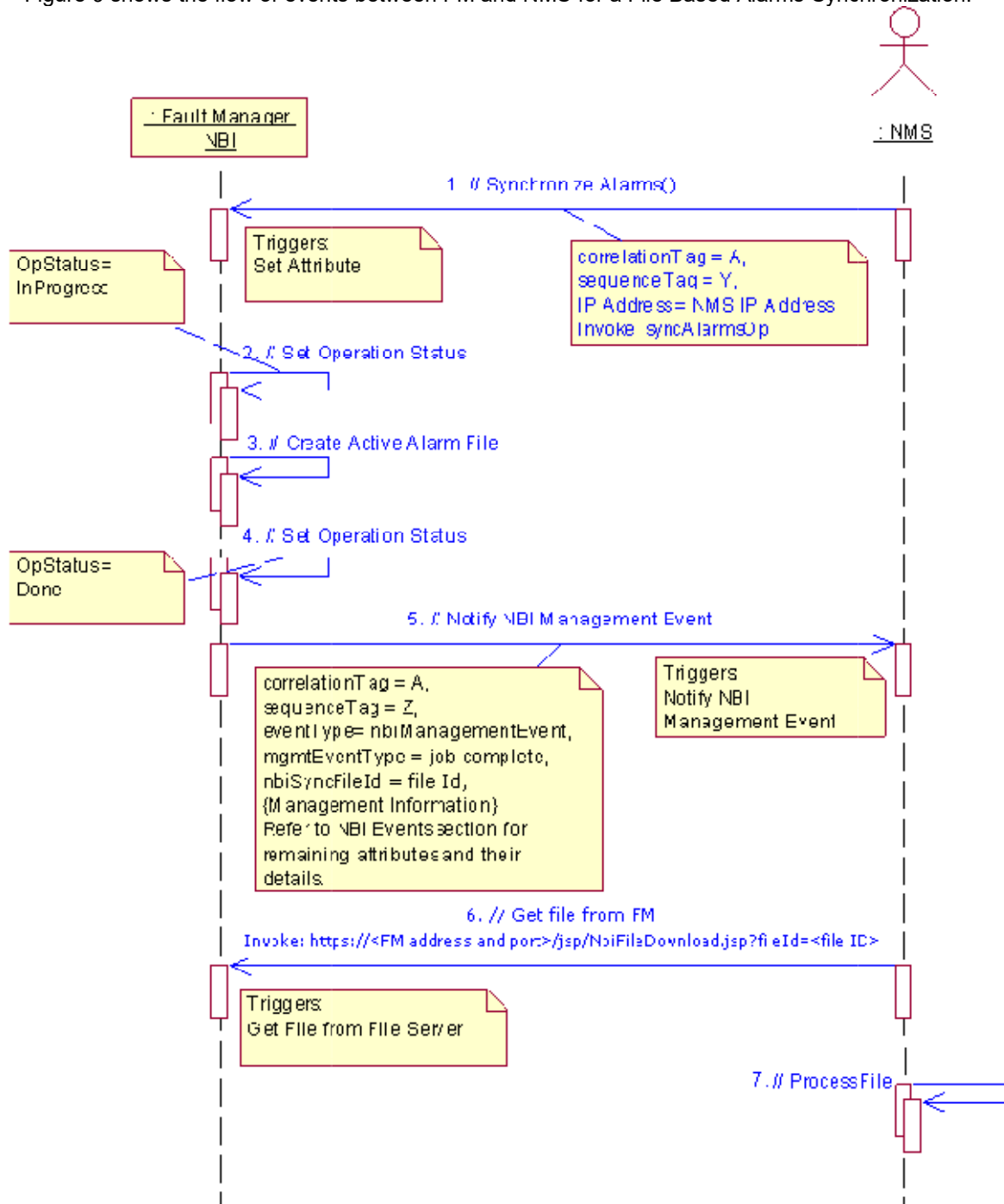


Figure 9 Synchronize Alarms - File Based Synchronization

4.3.4. FM-NMS Communication Link Management

FM-EMS communication link will be managed by the NMS using periodic polling of FM data described in the Table 2. FM will support access to FM system uptime, the latest notification sequence number

currently assigned to a specified NMS and latest NBI Event Record Identifier (if supported). Access to these attributes shall be via the SNMP interface. NMS can poll these attributes periodically with failure to access these attributes termed as management link-down.

Table 2 OIDs to be polled

FIELD NAME/OID	MIB FILENAME	DESCRIPTION
latestSequenceNumber .1.3.6.1.4.1.161.3.10.106.1.1.3	NBI-FM-MIB	This is the unique and the latest sequence number which is present in each nms. This gives the latest nms sequence number which is used for the synchronization purpose.
latestEventRecordId .1.3.6.1.4.1.161.3.10.106.1.1.4	NBI-FM-MIB	This is the unique and the latest event record identifier which is used for the synchronization purpose.
nbiUpTime .1.3.6.1.4.1.161.3.10.1.7	NBI-FM-MIB	This variable states the duration for which the NMS Server has been running

Apart from the simple link-down detection, the link management feature gives the NMS a possibility of building robust communication supervision.

On the basis of the **nbiUpTime** field you can determine how long the FM server has been working and lets you detect restarts if the time is shorter than you expected. In such cases NMS can request events/alarms synchronization to be in sync with the FM.

The **latestSequenceNumber** or **latestEventRecordId** are the numbers of the last notification sent to the NMS. This information can be used to confirm synchronization between FM and NMS.

Figure 10 below shows the supported SNMP request handling on FM for FM Server start time and latest NBI notification sequence number. These requests can be sent periodically to supervise the NMS-FM Management link.

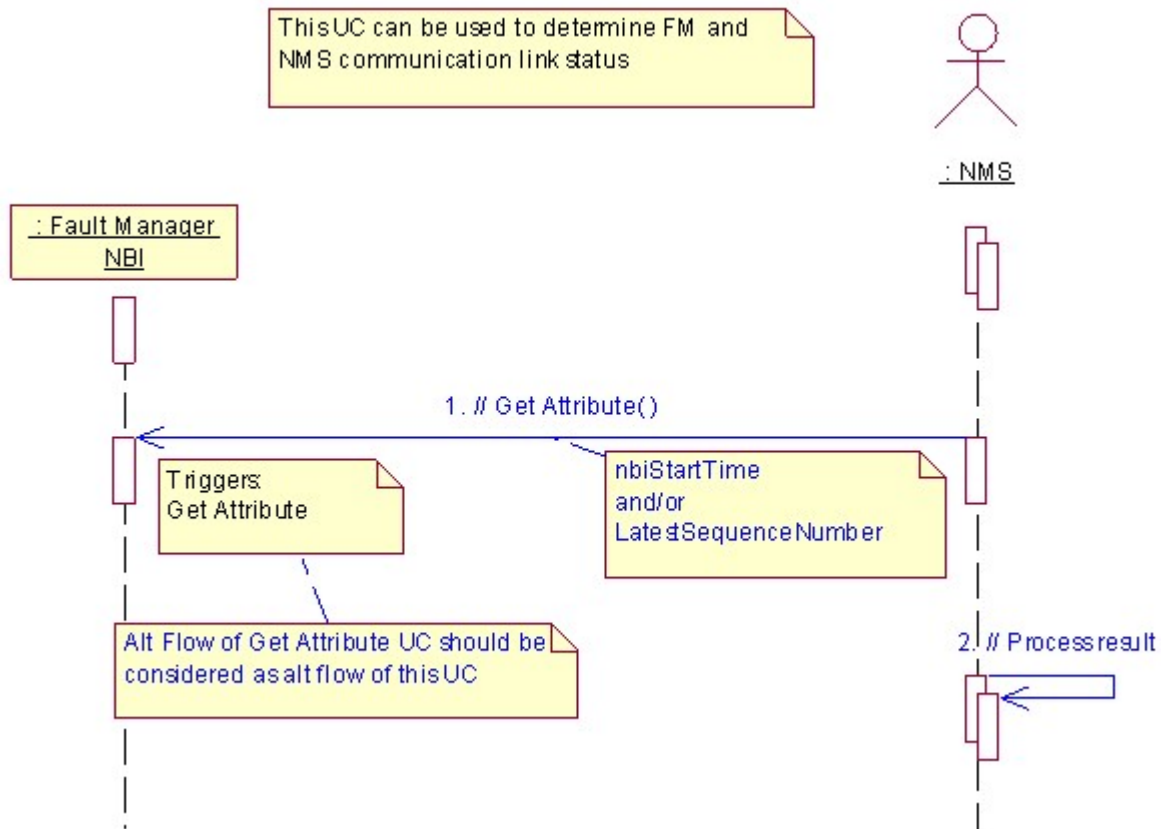


Figure 10 Supervise FM-NMS Communication Link

4.4. SNMP Definition - NBI Notifications

The NBI sends two types of SNMP notifications to the registered NMS, '**nbiEventRecord**' (*Network Event Notification*) and '**nbiManagementEvent**' (*Management Event Notification*). These notifications are defined in the NBI-FM-MIB. Refer to NBI-FM-MIB for details.

4.4.1. NBI Network Event Trap Definition

This trap is generated whenever an event is raised on the FM. SNMP varbinds for the NBI notifications are specified in the table 3. Most of the varbinds in this trap map to FM's internal event attributes which is shown in the table 4.

Table 3 Network Event Trap attributes

NAME	TYPE	ACCESSABILITY	DESCRIPTION
nbiNotiSequenceNum(.1.3.6.1.4.1.161.3.10.105.3)	Unsigned32	accessible-for-notify	This attribute specifies sequence number of the notification. Each notification sent to a receiver is assigned a contiguous sequence number. The sequence number is maintained for every receiver of the notification. This attribute is persisted and hence the sequence is maintained across restarts.
nbiEventId(.1.3.6.1.4.1.161.3.10.105.4)	DisplayString	accessible-for-notify	This attribute specifies the class identifier of the event.
nbiEventType(.1.3.6.1.4.1.161.3.10.105.5)	NbiEventType	accessible-for-notify	This attribute specifies the type of event.
nbiEntityId(.1.3.6.1.4.1.161.3.10.105.6)	DisplayString	accessible-for-notify	This attribute specifies the Identifier of the managed entity having the problem or the condition that initiated the event/alarm.
nbiEntityName(.1.3.6.1.4.1.161.3.10.105.7)	DisplayString	accessible-for-notify	This attribute specifies the User defined name of the managed entity having the problem or the condition that initiated the event/alarm.
nbiManagedObjectName(.1.3.6.1.4.1.161.3.10.105.8)	DisplayString	accessible-for-notify	This attribute specifies the User defined name of the source managed resource that is managing the Entity reporting the event/alarm. This attribute is set to NULL if not used.
nbiSeverity(.1.3.6.1.4.1.161.3.10.105.9)	INTEGER	accessible-for-notify	This attribute indicates the severity level of this event/alarm.
nbiSourceDevice(.1.3.6.1.4.1.161.3.10.105.10)	DisplayString	accessible-for-notify	This attribute specifies the managed resource, that is managing the Entity, reporting the event/alarm.
nbiDetectionEventTime(.1.3.6.1.4.1.161.3.10.105.11)	DateAndTime	accessible-for-notify	This attribute contains the time when the event was detected on the manager.
nbiSourceEventTime(.1.3.6.1.4.1.161.3.10.105.12)	DateAndTime	accessible-for-notify	This attribute contains the time when the event was sourced at the NE. This attribute is set to NULL if not used.
nbiText(.1.3.6.1.4.1.161.3.10.105.13)	DisplayString	accessible-for-notify	This parameter if present specifies a specific event message. This attribute is set to NULL if not used.

NAME	TYPE	ACCESSABILITY	DESCRIPTION
nbiAdditionalInformation(.1.3.6.1.4.1.161.3.10.105.14) (deprecated)	DisplayString	accessible-for-notify	This parameter when present allows the inclusion of a set of additional information in the event report. This attribute is set to NULL if not used.
nbiEventIdentifier(.1.3.6.1.4.1.161.3.10.105.17) (deprecated)	DisplayString	accessible-for-notify	Unique event identifier Note: Available in Dimetra FM Only
nbiEventRecordId(.1.3.6.1.4.1.161.3.10.105.19)	DisplayString	accessible-for-notify	Unique event identifier
nbiSourceAgentIP(.1.3.6.1.4.1.161.3.10.105.20)	DisplayString	accessible-for-notify	This attribute specifies the IP Address of the agent (device) reporting the event/alarm.
nbiCorrelationTag(.1.3.6.1.4.1.161.3.10.105.1)	Unsigned32	accessible-for-notify	This unsigned integer is used to bind all activity associated with a 'cause' or 'root'(either a manager request or an NMS-initiated event). Correlation tag values are unique but tag sequence is not guaranteed (i.e. They may not be received in the order sent). No history is kept of recently used tags – only active tags are tracked. The following non-overlapping ranges of tags are reserved for assignment by the NMS and the FM: 00000000 - 7FFFFFFF allocated for FM-initiated transactions 80000000 - FFFFFFFF allocated for NMS-initiated transactions.

Table 4 Mapping between FM and Network Event Trap event attributes

NBI	FM	Comment	Example
nbiSequenceTag	-	no counterpart on FM exists for this property	-
nbiCorrelationTag	-	no counterpart on FM exists for this property	-
nbiSequenceNum	-	no counterpart on FM exists for this property	-
nbiEventId	Identifier		nm_common_application_fault.2.2
nbiEventType	Category	Alarm vs. EventInformation	Equipment Alarm * - nbi receives severity as a digit, eg. 4
nbiEntityId	Failure Object	Managed Object Level (entity - managed object information)	10.3.233.20:nm_common_application_fault:1
nbiEntityName	EntityName		Application
nbiManagedObjectUserName	Managed Resource		fmHostname.fmDomainName
nbiSourceDevice	Source	Managed Resource Level (device identification information)	10.3.233.20:NMServer

nbiSeverity	Severity		Clear * - nbi receives severity as a digit, eg. 6
nbiDetection EventTime	NeTime Stamp		Nov 16, 2010 13:34:53 +0000
nbiSource EventTime	Date/Time		Nov 16, 2010 13:34:53 +0000
nbiText	Message		ENABLED, USER REQUESTED
nbiAdditional Information (deprecated)	-	no counterpart on FM exists for this property	-
nbiEventIdentifier (deprecated)	Event ID	FM unique event identifier Note: Dimetra Only	141
nbiEventRecordId	Event ID	FM unique event identifier	141
nbiSourceAgentIP	Reporting Agent	IP address of NE which sourced an event	10.3.233.20

Two important attributes – such as **severity** and **event type** – are forwarded to the registered NMS as digits, whereas on the FM they exist as human readable text values. To understand the type of event (alarm or event) as well as severity of a Network Event which is received by the NMS the following translation should be done:

- for the event type - refer to the table 5 or the NbiEventType field definition in the NBI-FM-MIB mib file,
- for the severity - refer to the table 6 or the nbiSeverity field definition in the NBI-FM-MIB mib file.

NOTE:

On the basis of the two aforementioned attributes you can determine whether the received event is an alarm or not and how important it is.

Table 5 Event Type Conversion matrix

nbiEventType numerical value	FM Category
1	Attribute-Value-Change Event
2	Communication Alarm
4	Equipment Alarm
6	Object-Creation Event
7	Object-Deletion Event
11	Quality-of-Service Alarm
13	Security Violation
255	NBI_Management Event
256	Informational Event
257	Management Event

Table 6 Severity Conversion matrix

nbiSeverity numerical value	FM Severity
1	CommFailure
2	Critical
3	Major
4	Minor
5	Warning
6	Clear
7	Info
8	Unknown

4.4.2. NBI Management Event Trap Definition

This SNMP trap is sent after the completion of event or alarm sync operation. This event denotes both failure and success conditions. The event will be sent with the appropriate event type varbind and reason code that are used to determine the failure and success condition. The Management Event Trap contains most of the varbinds present for the 'nbiEventRecord' – section 4.4.1 - and additional varbinds enumerated in the Table 7 below:

Table 7 Management Event Trap attributes

NAME	TYPE	ACCESSABILITY	DESCRIPTION
nbiSequenceTag (.1.3.6.1.4.1.161.3.10.10 5.2)	Unsigned32	accessible-for-notify	This attribute specifies sequence number of the notification. The sequence number is maintained for every receiver of the notification. This attribute is persisted and hence the sequence is maintained across restarts/reboots
nbiMgmtEvent Type (.1.3.6.1.4.1.161.3.10.10 5.15)	INTEGER	accessible-for-notify	This parameter indicates the type of management event being communicated. Following are the management information types and corresponding values: <ul style="list-style-type: none"> • transfer-Complete (2) • data- Committed (3) • job-Aborted (4) • job-Complete (7) • file-Transfer- Failed(11)
nbiMgmtReasonCode (.1.3.6.1.4.1.161.3.10.10 5.16)	NbiReasonCode	accessible-for-notify	This parameter gives the status of the operation, for example whether the operation is successful failure waiting invalid or failure due to some other reason. Following are the management reason codes and corresponding values: <ul style="list-style-type: none"> • NotUsed (0) • UnKnown (1) • dataNotAvailable (2) • dataExtractionError (3) • checksumError (4) • fileCreationError (5) • requestError (6) • operationConcurrencyError (7)
nbiSyncFileId (.1.3.6.1.4.1.161.3.10.10 5.17) (deprecated)	DisplayString	accessible-for-notify	It gives the file ID for NBI file synchronization. If synchronization process fail or there is no events /alarms to synchronization this attribute has null value. Note: Available in Common FM Only

nbiSyncFileToken (.1.3.6.1.4.1.161.3.10.10 5.18)	DisplayString	accessible-for- notify	It gives the file ID for NBI file synchronization. If synchronization process fail or there is no events /alarms to synchronization this attribute has null value.
--	---------------	---------------------------	---

NOTE:

Refer to Appendix B for more details on NBI notification types and the exact varbinds accompanying each type of notification type.

4.5. SNMP Definition - Management Operations

4.5.1. Quick Synchronization SNMP Interface Definition

The NMS can request from the FM a quick synchronization to retrieve recent lost trap(s). This can be achieved by performing an SNMP SET request on the appropriate element from the **nbiQuickSyncTable** or **nbiEventRecordSyncTable** table supported in the **NBI-FM-MIB**. Once the SNMP SET request is received by the NBI layer of FM, the lost trap(s) shall be re-sent to the requestor (NMS).

To trigger a quick synchronization an appropriate SNMP SET should be sent to the FM. There are two equivalent SET methods based on the **nmsCorrelationId** or **nmsEventRecordCorrelationId** OID. The general syntax for the quick synchronization operation is as follows:

```
set <nmsCorrelationId >.< nmsIpAddress >.< nmsPortNumber >.< nmsSequenceNumber> <intvalue>
```

or

```
set <nmsEventRecordCorrelationId >.< nmsIpAddress >.< nmsPortNumber >.  
<nmsEventRecordSequenceNumber> <intvalue>
```

First of all, you have to specify how many and which notifications will be re-sent. As the FM maintains last "N" events per each registered NMS, not only must you specify what to re-sent (**nmsSequenceNumber** or **nmsEventRecordSequenceNumber** parameter) but also whom this information will be intended to (**nmsIpAddress** and **nmsPortNumber** parameters). The **intvalue** is an integer value of the **nmsCorrelationId** or **nmsEventRecordCorrelationId** which is being set. This parameter is explained in the section 4.1.1.

NOTE:

For the details related to the individual elements of this syntax, please refer to the Table 8 - *Elements of the syntax for the Quick Synchronization Operation* below.

The correlation ID or event record correlation ID as well as the NMS IP address and port number can be taken immediately to prepare an SNMP set request, without any conversion. The same, however, is not true for the **nmsSequenceNumber** or **nmsEventRecordSequenceNumber**. Those parameters have been defined as **DisplayString** which means that their translated form must be appended to the SNMP set request.

The simple procedure for conversion into a DisplayString is as follows:

- Treat an **nmsSequenceNumber** or **nmsEventRecordSequenceNumber** as a composition of characters.
- For each character determine its ASCII code (refer to the 7-bit ASCII codes table available in the Appendix E). As you can provide both a list of particular notifications separated by commas or a range of notifications – keep in mind that not only ASCII codes for digits will be used but also for a comma or a hyphen.
- Prepare the **nmsSequenceNumber** or **nmsEventRecordSequenceNumber** display string equivalent by concatenating:
 - a number of characters converted into the ASCII code (a length-like value)
 - dot
 - ASCII codes for each character separated by dots

NOTE:

The procedure above assumes you prepare an SNMP set request by means of a command-line-like tool. If you use a third party library for SNMP support, please consult your vendor's documentation to get familiar how to set a value of DisplayString.

NOTE:

It is recommended that the number of trap requested to be synchronized (re-sent) doesn't exceed 20. It is recommended to use file based event synchronization if more than 20 events need to be synchronized.

See the examples below to understand the quick synchronization in practice.

Table 8 Elements of the syntax for the Quick Synchronization Operation

NAME/OID	TYPE	ACCESS	DESCRIPTION
nmsIpAddress .1.3.6.1.4.1.161.3.10.106.1.1.1	IpAddress	Not Accessible	The first part of the index of nbiQuickSyncTable table. This is an IP address of the NMS which has been registered in the FM.
nmsPortNumber .1.3.6.1.4.1.161.3.10.106.1.1.2	Integer32	Not Accessible	The second part of the index of nbiQuickSyncTable table. This is a port at which NMS is listening for SNMP traps. The value should correspond to the port number which has been registered in the FM
nmsSequenceNumber .1.3.6.1.4.1.161.3.10.106.2.1.1	Display String	Not Accessible	<p>The third part of the index of nbiQuickSyncTable table. By means of this parameter you determine how many and which notifications will be re-sent The sequence number field can contain:</p> <ul style="list-style-type: none"> - comma separated list of notification sequence Ids to be sent to NMS. - a range expressed as x-y of notifications to be re-sent to NMS, where "x" is the starting sequence Id of the notification and "y" is the end sequence Id of the notification. - both comma separated values and a range <p>Example: Sequence number field of 1,2,4-6,10 will enable NBI to send notification with sequence Ids 1,2,4,5,6,10.</p> <p>NOTE: This field is of type DisplayString which means that it must be appropriately converted to be used in the SNMP SET request</p>
nmsCorrelationId .1.3.6.1.4.1.161.3.10.106.2.1.2	Integer32	Read-Write	This is the OID which is going to be set by the issued SNMP SET operation.
nmsEventRecordSequenceNumber .1.3.6.1.4.1.161.3.10.106.3.1.1	Display String	Not Accessible	<p>The third part of the index of nbiEventRecordSyncTable table. By means of this parameter you determine how many and which notifications will be re-sent The sequence number field can contain:</p> <ul style="list-style-type: none"> - comma separated list of notification Event Record Ids to be sent to NMS. - a range expressed as x-y of notifications to be re-sent to NMS, where "x" is the starting sequence Id of the notification and "y" is the

			<p>end sequence Id of the notification. - both comma separated values and a range</p> <p>Example: Sequence number field of 1,2,4-6,10 will enable NBI to send notification with event record Ids 1,2,4,5,6,10.</p> <p>NOTE: This field is of type DisplayString which means that it must be appropriately converted to be used in the SNMP SET request</p>
nmsEventRecordCorrelationId .1.3.6.1.4.1.161.3.10.106.3.1.2	Integer32	Read-Write	This is the OID which is going to be set by the issued SNMP SET operation.

NOTE:

If you request non-existing event(s) then the following error will be reported (event with the id=98 in the example):

Error in packet.

Reason: (noSuchName) There is no such variable name in this MIB.

Failed object: SNMPv2-SMI::enterprises.161.3.10.106.2.1.2.10.7.233.187.700.2.57.56

The same is truth if you specify a range of events and some of them don't exist.

Example 1:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform a quick synchronization resulting in the notification with the id "1" to be re-sent.

Steps:

The **nmsSequenceNumber** – expressed as a human readable string – is "1". This format, however, is not acceptable in terms of SNMP SET operation.

To convert this value into a **DisplayString** format we need to find an ASCII representation for the digit "1" which is **49**. As we have used only one ASCII code, the 49 will be preceded with the value 1 (this value doesn't undergo ASCII conversion). So the **nmsSequenceNumber** ready to attach to the SNMP SET looks like this: **1.49** and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
.1.3.6.1.4.1.161.3.10.106.2.1.2.10.1.233.186.162.1.49      i      1
```

nmsCorrelationId OID
NMS IP address
NMS port
Re-sent notification with ID 1
int
Correlation ID to be set

Result:

Trap containing message with the Sequence number 1 is being re-sent to the NMS.

Example 2:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform a quick synchronization resulting in the notifications with the IDs "1","20","208 to be re-sent.

Steps:

The **nmsSequenceNumber** – expressed as a human readable string – is "**1,20,208**". According to the rules demonstrated in the example 1 we must find ASCII equivalents for digits and a comma which are:

- ASCII code of "0" is 48
- ASCII code of "1" is 49
- ASCII code of "2" is 50
- ASCII code of "8" is 56
- ASCII code of "," is 44

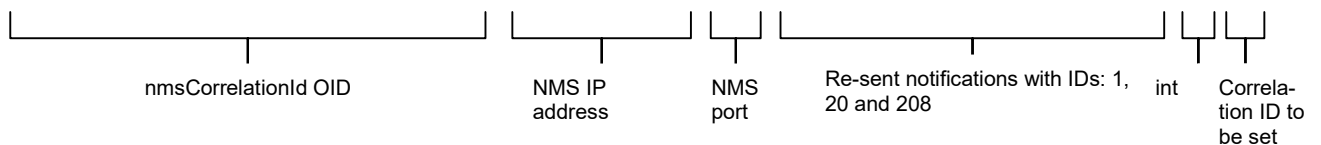
NOTE:

Eight ASCII characters are required to express "1,20,208". That's why the **nmsSequenceNumber** starts with the value 8.

The **nmsSequenceNumber** ready to attach to an SNMP SET looks like this: **8.49.44.50.48.44.50.48.56** and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
```

```
.1.3.6.1.4.1.161.3.10.106.2.1.2.10.1.233.186.162.8.49.44.50.48.44.50.48.56 i 2
```

**Result:**

Traps containing messages with the Sequence numbers: 1, 20 and 208 are being re-sent to the NMS.

Example 3:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform a quick synchronization resulting in the notifications with the IDs "33", "40", "50", "51" and "52" to be re-sent. We want to specify individual IDs as well as a range of notifications in the following form: **"33,40,50-52"**.

Steps:

The **nmsSequenceNumber** – expressed as a human readable string – is **"33,40,50-52"**. According to the rules demonstrated in the examples 1 and 2 we must find ASCII equivalents for digits, a comma and a hyphen which are:

- ASCII code of "0" is 48
- ASCII code of "2" is 50
- ASCII code of "3" is 51
- ASCII code of "4" is 52
- ASCII code of "5" is 53
- ASCII code of "," is 44
- ASCII code of "-" is 45

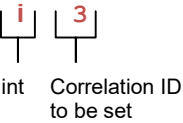
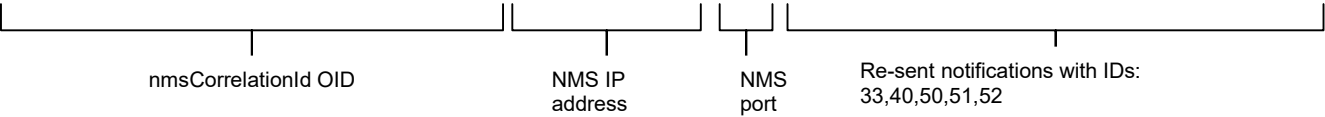
NOTE:

Eleven ASCII characters are required to express "33,40,50-52". That's why the **nmsSequenceNumber** starts with the value 11.

The **nmsSequenceNumber** ready to attach to an SNMP SET looks like this:

11.51.51.44.52.44.53.48.45.53.50 and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
.1.3.6.1.4.1.161.3.10.106.2.1.2.10.1.233.186.162.11.51.51.44.52.48.44.53.48.45.53.50
```



Result:

Traps containing messages with the Sequence numbers: 33, 40, 50, 51 and 52 are being re-sent to the NMS.

Example 4:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform a quick synchronization resulting in the notification with the Event Record ID “1” to be re-sent.

Steps:

The **nbiEventRecordId** – expressed as a human readable string – is “1”. This format, however, is not acceptable in terms of SNMP SET operation.

To convert this value into a **DisplayString** format we need to find an ASCII representation for the digit “1” which is **49**. As we have used only one ASCII code, the 49 will be preceded with the value 1 (this value doesn’t undergo ASCII conversion). So the **nbiEventRecordId** ready to attach to the SNMP SET looks like this: **1.49** and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
.1.3.6.1.4.1.161.3.10.106.3.1.2.10.1.233.186.162.1.49      i      4
```

nmsEventRecordCorrelationIdOID
NMS IP address
NMS port
Re-sent notification with ID 1
int
Correlation ID to be set

Result:

Trap containing message with the Event Record Identifier 1 is being re-sent to the NMS.

Example 5:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform a quick synchronization resulting in the notifications with the Event Record IDs "33", "40", "50", "51" and "52" to be re-sent. We want to specify individual IDs as well as a range of notifications in the following form: **"33,40,50-52"**.

Steps:

The **nbiEventRecordId** – expressed as a human readable string – is **"33,40,50-52"**. According to the rules demonstrated in the examples 1 and 2 we must find ASCII equivalents for digits, a comma and a hyphen which are:

- ASCII code of "0" is 48
- ASCII code of "2" is 50
- ASCII code of "3" is 51
- ASCII code of "4" is 52
- ASCII code of "5" is 53
- ASCII code of "," is 44
- ASCII code of "-" is 45

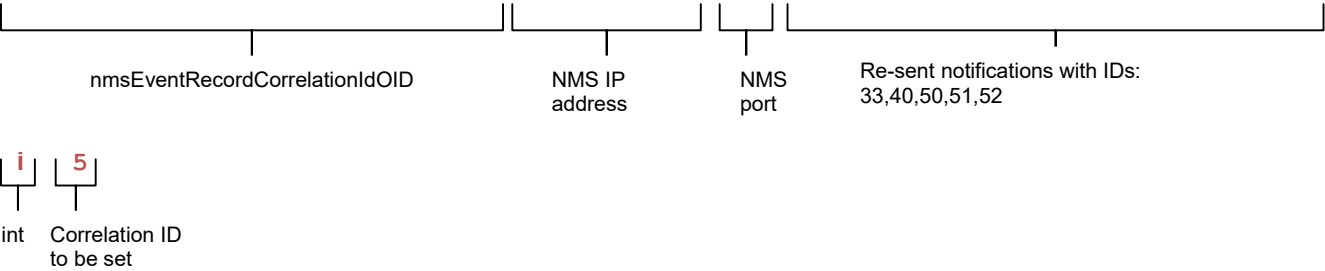
NOTE:

Eleven ASCII characters are required to express "33,40,50-52". That's why the **nbiEventRecordId** starts with the value 11.

The **nbiEventRecordId** ready to attach to an SNMP SET looks like this:

11.51.51.44.52.48.44.53.48.45.53.50 and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
.1.3.6.1.4.1.161.3.10.106.3.1.2.10.1.233.186.162.11.51.51.44.52.48.44.53.48.45.53.50
```



Result:

Traps containing messages with the Event Record Identifiers equal: 33, 40, 50, 51 and 52 are being re-sent to the NMS.

4.5.2. Alarm Synchronization SNMP Interface Definition

NMS can request synchronization of alarms from the FM. This alarm sync operation can be triggered on the FM by sending an SNMP SET request to the SNMP agent, which runs as part of the NBI layer of the FM. On receiving the SNMP SET request, the SNMP agent shall start to collect all **active alarms** (alarms with a severity different from CLEAR) from the FM DB and write it to the Alarm Summary XML file. After the creation of the XML file, the SNMP agent shall send out the '**nbiManagementEvent**' event (see Appendix B) to the NMS, giving the file Id which identifies specific xml file.

NOTE:

Refer to Appendix C.3 for an example Alarm Summary XML file.

To trigger an alarm synchronization operation, an appropriate SNMP SET regarding **nbiOperationsTable** (refer to the NBI-FM-MIB file) - with the minimum two varbinds mentioned in the Table 9 - should be sent to the FM:

Table 9 Alarm Synchronization request varbinds

NAME	OID	Description / Value
nbiSequenceId.<nbiNmsId>. <nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.5. <nbiNmsId>.<nbiOperationId>	For the initial alarm synchronization request the nbiSequenceId will be 1 . NOTE: This field is OPTIONAL
nbiOperationsType.<nbiNmsId>. <nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.6. <nbiNmsId>.<nbiOperationId>	2
nbiOperationRow Status.<nbiNmsId>. <nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.7. <nbiNmsId>.<nbiOperationId>	4

NOTE:

For the details related to the individual elements of this syntax, please refer to the Table 10 - *Elements of the syntax for the Alarms/Events Synchronization Operation* below.

The only parameter that needs conversion is the **nbiNmsId**. The rest of the parameters can be used directly. The **nbiNmsId** has been defined as **DisplayString** which means that its translated form must be appended to the SNMP set request.

The simple procedure for conversion into a DisplayString is as follows:

- Treat an **nbiNmsId** as a composition of characters.
- For each character determine its ASCII code (refer to the 7-bit ASCII codes table available in the Appendix E). As this field is supposed to hold a textual representation of an IP address, all you need are ASCII codes for a dot and digits.
- Prepare the **nbiNmsId** display string equivalent by concatenating:
 - a number of characters converted into the ASCII code (a length-like value)
 - dot
 - ASCII codes for each character separated by dots

NOTE:

The procedure above assumes you prepare an SNMP set request by means of a command-line-like tool. If you use a third party library for SNMP support, please consult your vendor's documentation to get familiar how to set a value of DisplayString.

Table 10 Elements of the syntax for the Alarms/Events Synchronization Operation

NAME/OID	TYPE	ACCESS	DESCRIPTION
nbiNmsId .1.3.6.1.4.1.161.3.10.102.1.1.1	Display String	Read Only	The first part of the index of nbiOperationsTable table. This is an IP address of the NMS which has been registered in the FM. NOTE: This field is of type DisplayString which means that it must be appropriately converted to be used in the SNMP SET request
nbiOperationId .1.3.6.1.4.1.161.3.10.102.1.1.2	Integer32	Read Only	The second part of the index of nmbiOperationsTable . This is a Correlation Tag specified in the initial operation request NOTE: This value must be unique per each request
nmbiStartTime .1.3.6.1.4.1.161.3.10.102.1.1.3	Date And Time	Read Create	Time stamp that is used to collect events whose Event Time is later or equal to this time. This parameter is expressed in UTC time NOTE: This field is not applicable for SyncAlarms operation type
nmbiEndTime .1.3.6.1.4.1.161.3.10.102.1.1.4	Date And Time	Read Create	Time stamp that is used to collect events whose Event Time is later or equal to this time. This parameter is expressed in UTC time NOTE: This field is not applicable for SyncAlarms operation type
nbiSequenceld .1.3.6.1.4.1.161.3.10.102.1.1.5	Integer32	Read Create	The nbiSequenceld is a message sequence identifier for the event synchronization operation. For the initial event synchronization request, the nbiSequenceld will be 1. NOTE: This field is optional
nbiOperationsType .1.3.6.1.4.1.161.3.10.102.1.1.6	Nbi Operation Type	Read Create	The operation identifier. The valid operations are: - SyncAlarms(2), - SyncEvents(3)

NAME/OID	TYPE	ACCESS	DESCRIPTION
nbiOperationRow Status .1.3.6.1.4.1.161.3.10.102.1.1.7	RowStatus	Read Create	This specifies the process by which the data have to be entered. The following values are possible: - active (1) - notInService (2) - notReady (3) - createAndGo (4) - createAndWait (5) - destroy(6)

NOTE:

You cannot issue an identical request more than once, as it results in the following error:

Error in packet.

Reason: inconsistentValue (The set value is illegal or unsupported in some way)

Failed object: SNMPv2-SMI::enterprises.161.3.10.102.1.1.7.12.49.48.46.55.46.50.51.51.46.49.56.55.32

This is due to the fact that each request must use a unique **nbiOperationId** value.

See the examples below to understand the alarm synchronization in practice.

Example 1:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- NMS IP address: **10.7.233.187**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform file-based alarms synchronization.

Steps:

To prepare a valid request you are supposed to provide all mandatory varbinds mentioned in the Table 9 *Alarm Synchronization request varbinds*. With the exception of the **nbiNmsId**, all the values can be used directly, without any conversion.

The **nbiNmsId** – expressed as a human readable string is - according to the assumption made at the beginning of this example - “**10.7.233.187**”. This format, however, is not acceptable in terms of SNMP SET operation. To convert this value into a DisplayString format we need to find an ASCII representation for all the digits and dots comprising of the NMS IP address, which are:

- ASCII code of “0” is 48
- ASCII code of “1” is 49
- ASCII code of “2” is 50
- ASCII code of “3” is 51
- ASCII code of “4” is 52
- ASCII code of “7” is 55
- ASCII code of “8” is 56
- ASCII code of “.” is 46

Twelve ASCII characters are required to express IP address “10.7.233.187”. That’s why the **nbiNmsId** starts with the value 12.

The **nbiNmsId** ready to attach to an SNMP SET looks like this: **12.49.48.46.55.46.50.51.51.46.49.56.55** and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
.1.3.6.1.4.1.161.3.10.102.1.1.6.12.49.48.46.55.46.50.51.51.46.49.56.55.1 i 2
.1.3.6.1.4.1.161.3.10.102.1.1.7.12.49.48.46.55.46.50.51.51.46.49.56.55.1 i 4
```

6 – nbiOperationsType 7 - nbiOperationRowStatus	NMS IP address 10.7.233.187 expressed as DisplayString	Nbi Op .Id	int	Sync Alarms and create and go
--	---	------------------	-----	---

Result:

Trap containing the results of the operation will be sent to the NMS. The most important fields of the trap's content are:

- **nbiSyncFileToken** – a file Id which identifies specific alarms synchronization xml file,
- **nbiAdditionalInformation** - a full path to the location where the alarms file is placed (deprecated),
- **nbiMgmtEventType** – type of the management event being communicated,
- **nbiMgmtReasonCode** – status of the operation.

NOTE:

For the complete varbinds in the trap refer to the sections 4.4.2 and 4.4.1.
For event types and reason codes please refer to the appendix B.

4.5.3. Event Synchronization SNMP Interface Definition

NMS can request synchronization of events from the FM. This events sync operation can be triggered on the FM by sending an SNMP SET request to the SNMP agent, which runs as part of the NBI layer of the FM. On receiving the SNMP SET request, the SNMP agent shall start to collect all events or only events satisfying specified time constraints and write them to the Event Summary XML file. After the creation of the XML file, the SNMP agent shall send out the '**nbiManagementEvent**' event (see Appendix B) to the NMS, giving the file Id which identifies specific xml file.

To trigger an alarm synchronization operation, an appropriate SNMP SET regarding **nbiOperationsTable** (refer to the NBI-FM-MIB file) - with the minimum two varbinds mentioned in the Table 11 below - should be sent to the FM. In such case all events will be fetched.

Additional filtering can be applied if at least one of the optional time-restriction or event record identifier parameter has been defined.

Table 11 Event Synchronization request varbinds

NAME	OID	Description / Value
nbiSequenceId.<nbiNmsId>.<nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.5.<nbiNmsId>.<nbiOperationId>	For the initial alarm synchronization request the nbiSequenceId will be 1. NOTE: This field is OPTIONAL
nbiStartTime.<nbiNmsId>.<nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.3.<nbiNmsId>.<nbiOperationId>	Time the NMS is requesting to collect events with the Event Time later or equal to this time. NOTE: This field is OPTIONAL
nbiEndTime.<nbiNmsId>.<nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.4.<nbiNmsId>.<nbiOperationId>	Time the NMS is requesting to collect events with the Event Time earlier or equal to this time. NOTE: This field is OPTIONAL
nbiStartRecordId.<nbiNmsId>.<nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.8.<nbiNmsId>.<nbiOperationId>	Record Identifier that is used to collect events whose EventRecordId is greater than or equal to this ID NOTE: This field is OPTIONAL
nbiEndRecordId.<nbiNmsId>.<nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.9.<nbiNmsId>.<nbiOperationId>	Record Identifier that is used to collect events whose EventRecordId is lesser than or equal to this ID NOTE: This field is OPTIONAL
nbiOperationsType.<nbiNmsId>.<nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.6.<nbiNmsId>.<nbiOperationId>	3

nbiOperationRow Status.<nbiNmsId>. <nbiOperationId>	.1.3.6.1.4.1.161.3.10.102.1.1.7. <nbiNmsId>.<nbiOperationId>	4
---	---	---

NOTE:

The syntax is remarkably similar to the Alarm Synchronization and the details related to the individual elements of this syntax can be found in the Table 10 - *Elements of the syntax for the Alarms/Events Synchronization Operation*.

If the filtering event is not used then the only parameter that needs conversion is the **nbiNmsId** and the rest of the parameters can be used directly. The conversion is described in the section 4.5.2. In case of applying time constraints the NMS is expected to attach to the SNMP SET request converted start or/and end time value(s).

4.5.3.1. Filtering events – time constraints

Both start time and end time are of type DateAndTime. According to the RFC 2579 DateAndTime syntax is based on the octet string, length 8 or 11 and is used to express UTC time formatted according to the following pattern (display hint definition from the RFC2579):

```
"2d-1d-1d,1d:1d:1d.1d,1a1d:1d"
```

For example, **2010-12-20,19:00:00.0,+5.30**, **2010-12-20,19:00:00.0** are the RFC2579 compliant UTC time definitions.

The meaning of the individual octets is explained in the table 12 below:

Table 12 Octets of the DateAndTime type

Field	Octets	Contents	Range	Comment
1	1-2	year	0-65536	
2	3	month	1-12	
3	4	day	1-31	
4	5	hour	0-23	
5	6	minutes	0-59	
6	7	seconds (use 60 for leap-second)	0-60	
7	8	deci-seconds	0-9	
8	9	direction from UTC	+' / '-'	optional
9	10	hours from UTC	0-13	optional
10	11	minutes from UTC	0-59	optional

NOTE:

To convert the **direction from UTC** parameter, first you are supposed to find an ASCII value for the "+" or "-" sign and then use the hexadecimal equivalent of that ASCII value.

You can define only one time constraint. For instance, if only **nbiStartTime** is defined then the **nbiEndTime** is set to the current time. Similarly, if only **nbiEndTime** has been specified then **nbiStartTime** is assumed to be zero.

To prepare time constraints for start and/or end time events filtering, the time must be first formatted according to the pattern "2d-1d-1d,1d:1d:1d.1d,1a1d:1d" and then each octet must be translated into a hex value. Once this is done an **eight** or **eleven-length** sequence of hexadecimal symbols can be attached to the SNMP SET request as the value of the **nbiStartTime** and/or **nbiEndTime**.

When preparing time constraints pay attention that there are two possible formats of DateAndTime. One contains information regarding Time Zone offset (11 bytes) and the other doesn't. If the FM receives only 8 bytes then the FM assumes that the provided start and/or end time constraints relate to the UTC time (GMT+0000).

The recommended approach is to use an 11 bytes (full) format, consisting Time Zone offset information. In such cases you can express time of events by means of your local time (NMS time), no matter what the Time Zone of the server is.

NOTE:

For a third party library supporting SNMP, please consult your vendor's documentation to get familiar how to set a value of DateAndTime.

See the examples below to understand the event synchronization in practice and how to prepare time constraints when events filtering needs to be applied.

Example 1:**Assumptions:**

- The Time Zone of the FM server is **CST (-0600)**
- The Current date and time on the FM server is:
Tue Jan 11 06:35:06 CST 2011
- The Time Zone of the NMS is **GMT+0100** (and automatic daylight savings changes are enabled)
- The Current date and time on the NMS is:
13:35 GMT+0100 Warsaw DST

Goal:

We want to specify time constraint so that we could choose events dated to

- **Jan 11,2011 13:05:34 +0100** (time of an event according to the NMS time)
- **Jan 11,2011 06:05:34 -0600** (the same event but seen according to the FM time)

Steps:

To issue a request for events synchronization and to apply some filtering against the time of events we have a couple of possibilities how to specify the start and/or end time. The same event can have a "different" time depending on the Time Zone - "different" refers only to the fact that the same time is formatted according to the Local Time Zone and thus its textual representation is different depending on the Time Zone.

Probably the most convenient way is to use our local time (in the example the local time is the time of the NMS because we send and SNMP request from the NMS to FM).

So, we will use

Jan 11,2011 13:05:34.0 +0100 → hex → "07 DB 01 0B 0D 05 22 00 2B 01 00"

Another possibility is to use FM time – but this approach requires knowledge of the FM server configuration which is not always available. To express the same condition we would have to specify:

Jan 11,2011 06:05:34 -0600 → hex → "07 DB 01 0B 06 05 22 00 2D 06 00"

In a similar way we can express the same time in a variety of ways as long as we can correctly translate the time into a specific Time Zone.

Example 2:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- NMS IP address: **10.7.233.187**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP command line utility**

Goal:

We want to perform file-based events synchronization. We don't want to apply any filters.

Steps:

To prepare a valid request you are supposed to provide all mandatory varbinds mentioned in the Table 11 *Event Synchronization request varbinds*. With the exception of the **nbiNmsId**, all the values can be used directly, without any conversion.

The **nbiNmsId** – expressed as a human readable string is - according to the assumption made at the beginning of this example - “**10.7.233.187**”. This format, however, is not acceptable in terms of SNMP SET operation. To convert this value into a DisplayString format we need to find an ASCII representation for all the digits and dots comprising of the NMS IP address, which are:

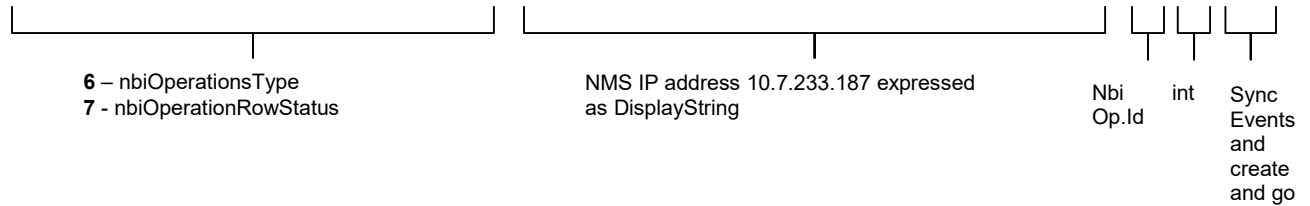
- ASCII code of “0” is 48
- ASCII code of “1” is 49
- ASCII code of “2” is 50
- ASCII code of “3” is 51
- ASCII code of “4” is 52
- ASCII code of “7” is 55
- ASCII code of “8” is 56
- ASCII code of “.” is 46

Twelve ASCII characters are required to express IP address “10.7.233.187”. That's why the **nbiNmsId** starts with the value 12.

The **nbiNmsId** ready to attach to an SNMP SET looks like this: **12.49.48.46.55.46.50.51.51.46.49.56.55** and the complete SET request is presented below:

```
snmpset -v 3 -l noAuthNoPriv -u MotoNorth 10.1.233.20:8001
```

```
.1.3.6.1.4.1.161.3.10.102.1.1.6.12.49.48.46.55.46.50.51.51.46.49.56.55.1 i 3
.1.3.6.1.4.1.161.3.10.102.1.1.7.12.49.48.46.55.46.50.51.51.46.49.56.55.1 i 4
```

**Result:**

Trap containing the results of the operation will be sent to the NMS. The most important fields of the trap's content are:

- **nbiSyncFileToken** – a file Id which identifies specific events synchronization xml file,
- **nbiAdditionalInformation** - a full path to the location where the events file is placed (deprecated),
- **nbiMgmtEventType** – type of the management event being communicated,
- **nbiMgmtReasonCode** – status of the operation.

NOTE:

For the complete varbinds in the trap refer to the sections 4.4.2 and 4.4.1.

For event types and reason codes please refer to the appendix B.

Example 3:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- NMS IP address: **10.7.233.187**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP**

Goal:

We want to perform file-based events synchronization. We want to apply a filter to choose only events dated from **2010-12-20,9:56:00** to **2010-12-21,9:36:00**.

Steps:

To prepare a valid request you are supposed to provide all mandatory varbinds mentioned in the Table 11 *Event Synchronization request varbinds* as well as **nbiStartTime** and **nbiEndTime**.

The **nbiNmsId** ready to attach to an SNMP SET looks like this: **12.49.48.46.55.46.50.51.51.46.49.56.55**, see the *example 1* to understand how the conversion was done.

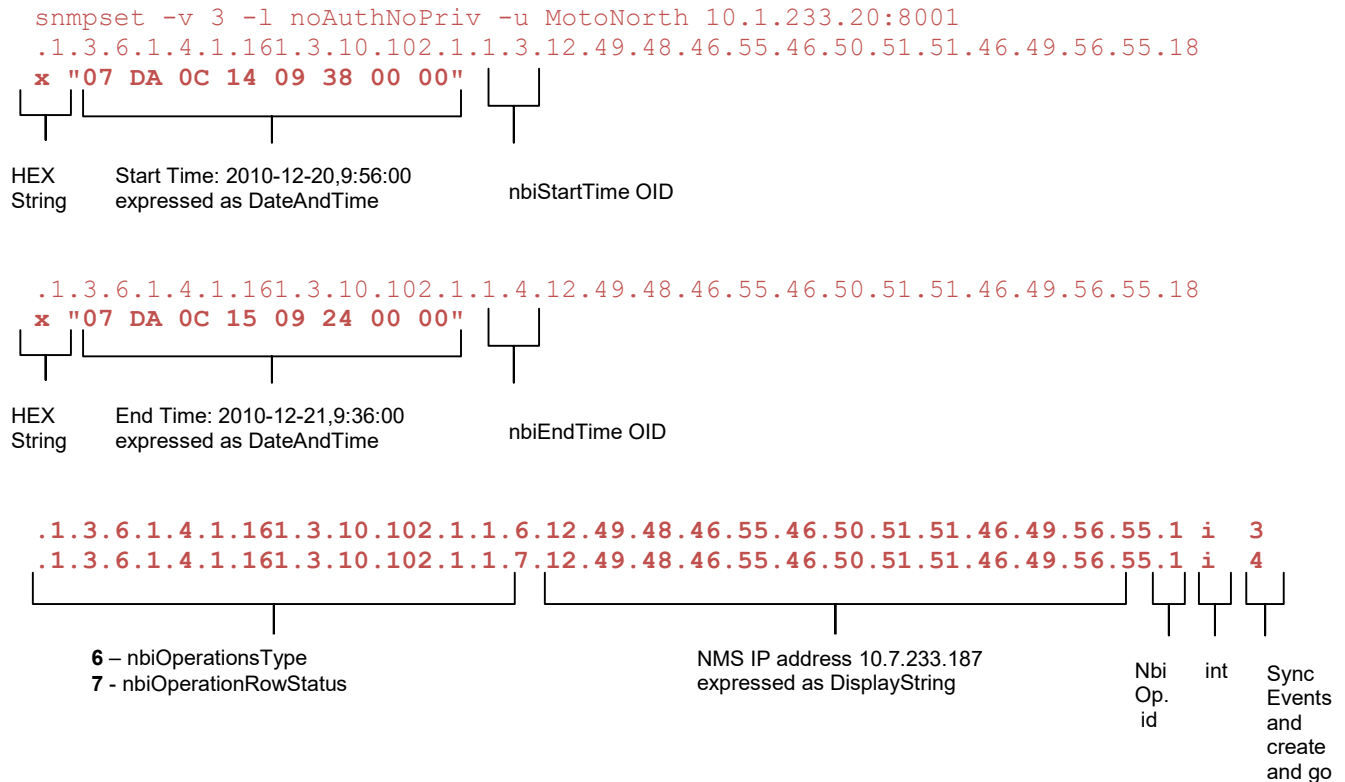
The hexadecimal, an eight-octet-length **start time** is: **07 DA 0C 14 09 38 00 00**

- 07 DA = Year (2010)
- 0C = Month (December)
- 14 = Day (20)
- 09 = Hour (9)
- 38 = Minute (56)
- 00 = Seconds (0)
- 00 = Tenths (0)

Similarly, the hexadecimal, an eight-octet-length **end time** is: **07 DA 0C 15 09 24 00 00**

- 07 DA = Year (2010)
- 0C = Month (December)
- 15 = Day (20)
- 09 = Hour (9)
- 24 = Minute (36)
- 00 = Seconds (0)
- 00 = Tenths (0)

The complete SET request is presented below:



Result:

Trap containing the results of the operation will be sent to the NMS. The most important fields of the trap's content are:

- **nbiSyncFileToken** – a file Id which identifies specific events synchronization xml file,
- **nbiAdditionalInformation** - a full path to the location where the events file is placed (deprecated),
- **nbiMgmtEventType** – type of the management event being communicated,
- **nbiMgmtReasonCode** – status of the operation.

NOTE:

For the complete varbinds in the trap refer to the sections 4.4.2 and 4.4.1.

For event types and reason codes please refer to the appendix B.

Example 4:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- NMS IP address: **10.7.233.187**
- SNMP protocol version: **v3**
- Port number on the FM listening for NBI requests: **8001**
- FM account name for the NBI service: **MotoNorth**
- Security settings for the NBI account: **NoAuth, NoPriv**
- SNMP tool: **Net-SNMP**

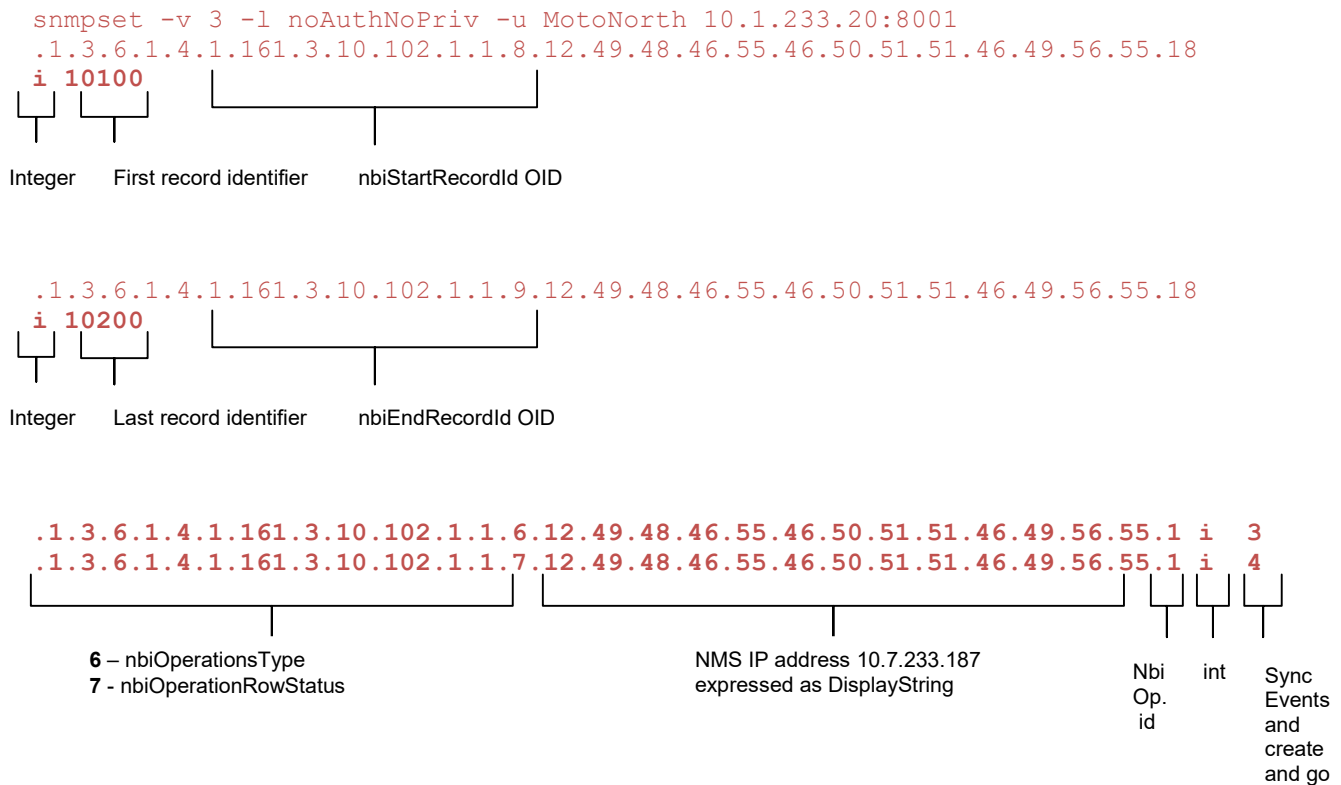
Goal:

We want to perform file-based events synchronization. We want to apply a filter to choose only events with identifiers between 10100 and 10200.

Steps:

To prepare a valid request you are supposed to provide all mandatory varbinds mentioned in the Table 11 *Event Synchronization request varbinds* as well as **nbiStartRecordId** and **nbiEndRecordId**. The **nbiNmsId** ready to attach to an SNMP SET looks like this: **12.49.48.46.55.46.50.51.51.46.49.56.55**, see the *example 1* to understand how the conversion was done.

The complete SET request is presented below:



Result:

Trap containing the results of the operation will be sent to the NMS. The most important fields of the trap's content are:

- **nbiSyncFileToken** – a file Id which identifies specific events synchronization xml file,
- **nbiAdditionalInformation** - a full path to the location where the events file is placed (deprecated),
- **nbiMgmtEventType** – type of the management event being communicated,
- **nbiMgmtReasonCode** – status of the operation.

NOTE:

For the complete varbinds in the trap refer to the sections 4.4.2 and 4.4.1.

For event types and reason codes please refer to the appendix B.

5. REST API (not supported in MOTOTRBO)

Fault Manager provides REST API as a part of North Bound Interface. REST API uses encrypted HTTPS as a transport protocol and JSON as document format in a HAL structure. SSL/TLS X.509 certificates are used for HTTPS communication (certificates should be loaded to the client application trust store for successful communication). Requests and responses are encoded in UTF-8. All requests going to NBI REST API will follow the base URL:

https://<UEM_IP>:<UEM_PORT>/nbi/v1

All queries described below are relative to this base URL. The API version is included in base URL. The default port for REST API is set to **49624**. List of error codes is defined in appendix **G**. Certificates used for communication over HTTPS

5.1. Authentication

User of this API will be authenticated using Basic HTTP authentication mechanism (user and password in a format `user:password` shall be Base64 encoded and provided in every https request in the header, e.g.

Authorization: Basic QWxhZGRpbjpvYVUHNlc2FtZQ==

User of this API shall be provisioned as a local user in the UEM with assigned permission "NBI Rest".

5.2. Inventory service

REST API provides inventory service. This service allows client to retrieve full or filtered list of managed Network Elements from FM. Response JSON documents contains fields from table below. For details see JSON schema documents attached on NBI Developers Guide CD.

Managed Resource fields	Obligatory	GMR	DMR	LMR	Generic Node
nbiSourceDevice	x			x	
nbiManagedObjectUserName	x			x	
nbiSeverity	x			x	
nbiManaged	x			x	
nbiClass	x			x	
nbiType	x			x	
nbiSynchronization	x	-	x	-	x
nbiSupervision	x	-	x	-	x
nbiDeviceCategory	Astro, LTE			x	
nbiIpAddress	x	-	x	-	x
nbiClusterWideStatus	Dimetra				
nbiRegion	Dimetra				
nbiSitelid	Dimetra	-	-	x	-

5.2.1. Inventory query

Provides all Managed Resources data with all properties and links to the detailed information for every Managed Resource

GET /inventory

The query result might be limited to the properties of each Managed Resource by using parameter `fields`. Comma is used as a delimiter for the properties required in the query result, e.g.

GET /inventory?fields=nbiSourceDevice,nbiClass

5.2.2. Detailed Managed Resource query

Provides individual Managed Resource data. This query provides all information as used in the general inventory query, but just for specific Managed Resource

GET /{nbiClass}/{nbiSourceDevice}

or

GET /inventory/{nbiSourceDevice}

where:

`nbiClass` - GenericNode, DeviceManagedResource, LogicalManagedResource, GroupManagedResource

`nbiSourceDevice` - unique source identifier of Managed Resource

5.2.3. Search query

There are 4 kinds of inventory search queries listed below.

5.2.3.1. Search based on class name `nbiClass`

GET /inventory?nbiClass={nbiClass}

or

GET /{nbiClass}

where:

`nbiClass` is one of GenericNode, DeviceManagedResource, LogicalManagedResource, GroupManagedResource,

e.g. **GET /inventory?nbiClass=GenericNode** or **GET /GenericNode**

5.2.3.2. Search based on parent identifier `nbiSourceDevice`

This query returns a list of all managed resources having the specified parent managed resource, e.g. query with parent Device Managed Resource will result with all Logical Managed Resources assigned to this Device Managed Resource.

GET /inventory?parents.nbiSourceDevice={nbiSourceDevice}

where :

`nbiSourceDevice` is the unique source identifier of Managed Resource,

e.g. **GET /inventory?parents.nbiSourceDevice=10.1.233.10%3AUEMSRVR**

5.2.3.3. Search based on resource type nbiType

GET /inventory?nbiType={nbiType}

where:

nbiType is a resource type e.g. Repeater Site, RF Site, etc.

e.g. **GET /inventory?nbiType=Motorola%20Unified%20Event%20Manager**

5.2.3.4. Search for resources which lost communication or had successful synchronization some time ago or in specified time range

The following post-fixes might be used with specific attribute to specify the time range:

- gt greater than
- ge greater or equal than
- lt lower than
- le lower or equal than

GET /inventory?{attribute}_lt={startTime}

GET /inventory?{attribute}_gt={startTime}&{attribute}_lt={endTime}

GET /inventory?{attribute}_ge={startTime}&{attribute}_le={endTime}

where:

attribute is nbiSynchronization or nbiSupervision

e.g. Search for all Managed Resources which lost communication at the specified time

GET /inventory?nbiSupervision_lt=2015-12-10T12:01:19.000%2B0100

5.2.4. MOSCAD RTU entities inventory and management

RTU Logical Managed Resource (LMR) corresponds to the group of devices connected to the Motorola SDM3000 RTU. The queries provides Fault Objects (entities) information to the LMRs attached to the RTU, e.g.: Digital Input entities, Digital Output entities, Analog Input entities, etc.

Note: The Fault Objects (entities) provided on this interface are for RTUs managed by UEM using SCADA over SNMP interface.

LMR Fault Object (entity) values reported via the UEM NBI reflect the most recent values sent by the SDM3000 to the UEM. The UEM does NOT initiate a real-time poll of current LMR values when an NBI query is initiated. Changes to Digital I/O values are always sent from the SDM to the UEM, however changes to Analog Inputs are only sent to the UEM when the configured threshold or delta change is triggered. The accuracy of the Analog Input values reported through the UEM NBI depends on how the SDM3000 is configured to send threshold and delta change updates to the UEM. For example, if the SDM3000 is configured to send an update to the UEM on a delta change of 1 for a temperature analog input, the value reported by the UEM will be within 1 degree accuracy of the current temperature at a site

5.2.4.1. Get specific entities from MOSCAD RTU LMR

Provides all information about every entity (Fault Object) in the specific MOSCAD RTU LMR. Below link might be taken from the json response to the query listed in section 5.2.3.1 with the class LogicalManagedResource. LMRs received with the entities json object are RTUs LMRs to which the link is provided as below:

GET /LogicalManagedResource/{nbiSourceDevice}/entities

where:

nbiSourceDevice is the unique source identifier of Managed Resource (LMR)

5.2.4.2. Get specific entity from MOSCAD RTU LMR

Provides information about one specific entity (Fault Object) of the MOSCAD RTU LMR. The reference is to different types of entities received in the response to the query in section 5.2.4.1

```
GET /DigitalOutputEntity/{nbiEntityId}
```

or

```
GET /DigitalInputEntity/{nbiEntityId}
```

or

```
GET /AnalogInputEntity/{nbiEntityId}
```

or

```
GET /ValueEntity/{nbiEntityId}
```

Note: Gives metering entity only information

or

```
GET /ManagedEntity/{nbiEntityId}
```

Note: Gives other entity information for types other than mentioned above

where:

`nbiEntityId` is the unique identifier of Fault Object in the RTU LMR

5.2.4.3. Update value for Digital Output Fault Object in RTU

This update changes the state of Digital Output entity. It enables/disables device connected to the RTU. Note that this is request to set Digital Output to the proper state. The actual response if the device state was changed might be received by using request described in previous section 5.2.4.2. It is mainly due to the asynchronous nature of setting entities over RTU.

```
PUT /DigitalOutputEntity/{nbiEntityId}
```

where:

`nbiEntityId` is the unique identifier of Fault Object in the RTU LMR

header:

```
Content-Type : application/json
```

body:

```
{
  "enabled" : {value}
}
```

where:

`value` is true (enable) or false (disable)

Example 1:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- Port number on the FM listening for REST requests: **49624**
- FM account name for the REST service: **root**
- HTTP tool: **CURL**

Goal:

We want to retrieve full list of FM managed Network Elements

The complete CURL request is presented below:

```
curl -k --user 'root:password' https://10.1.233.20:49624/nbi/v1/inventory
```



Example 2:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- Port number on the FM listening for REST requests: **49624**
- FM account name for the REST service: **root**
- HTTP tool: **CURL**

Goal:

We want to retrieve list of *ZC Site* LMRs from FM

The complete CURL request is presented below:

```
curl -k --user 'root:password' https://10.1.233.20:49624/nbi/v1/inventory?nbiType=ZC%20Site
```

Diagram illustrating the components of the CURL request:

- `root`: user name
- `password`: password
- `10.1.233.20`: FM IP address
- `49624`: REST port

Example 3:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- Port number on the FM listening for REST requests: **49624**
- FM account name for the REST service: **root**
- HTTP tool: **CURL**

Goal:

We want to retrieve list of Device Managed Resources managed by FM

The complete CURL request is presented below:

```
curl -k --user 'root:password' https://10.1.233.20:49624/nbi/v1/DeviceManagedResource/
```

The diagram illustrates the components of the curl command. It shows four boxes with lines pointing to specific parts of the command: 'user name' points to 'root', 'password' points to 'password', 'FM IP address' points to '10.1.233.20', and 'REST port' points to '49624'.

Example 4:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- Port number on the FM listening for REST requests: **49624**
- FM account name for the REST service: **root**
- HTTP tool: **CURL**

Goal:

We want to retrieve list of entities for Logical Managed Resource with Source Id 10.3.233.33:DO:1 managed by FM.

The complete CURL request is presented below:

```
curl -k --user 'root:password'  
https://10.1.233.20:49624/nbi/v1/LogicalManagedResource/10.3.233.33%3ADO%3A1/entities
```

Example 5:**Pre-conditions:**

Suppose the following configuration:

- IP address of the FM: **10.1.233.20**
- Port number on the FM listening for REST requests: **49624**
- FM account name for the REST service: **root**
- HTTP tool: **CURL**

Goal:

We want to enable the Digital Output entity with id 10.3.233.33:DO:1:IOAPPL_DO_OBJ_ID:LOW:0:57 managed by FM.

The complete CURL request is presented below:

```
curl -k --user 'root:password' -X PUT -H "Content-Type: application/json" -d
'{"enabled":true}'
https://10.1.233.20:49624/nbi/v1/DigitalOutputEntity/10.3.233.33%3ADO%3A1%3AIOAPPL_DO_OBJ
_ID%3ALOW%3A0%3A57
```

This page is intentionally blank

Appendix A NBI Management Operation Return Status Code

A.1 SNMP SET and GET Request Response Status Code

This table specifies the various SNMP error codes supported on the FM NBI.

Status String name	Integer Value	Description
noError	0	No error occurred. This code is also used in all request PDUs, since they have no error status to report.
tooBig	1	The size of the Response-PDU would be too large to transport.
noSuchName	2	The name of a requested object was not found.
badValue	3	A value in the request didn't match the structure that the recipient of the request had for the object. For example, an object in the request was specified with an incorrect length or type.
readOnly	4	An attempt was made to set a variable that has an Access value indicating that it is read-only.
genErr	5	An error occurred other than one indicated by a more specific error code in this table.
noAccess	6	Access was denied to the object for security reasons.
wrongType	7	The object type in a variable binding is incorrect for the object.
wrongLength	8	A variable binding specifies a length incorrect for the object.
wrongEncoding	9	A variable binding specifies an encoding incorrect for the object.
wrongValue	10	The value given in a variable binding is not possible for the object.
noCreation	11	A specified variable does not exist and cannot be created.
inconsistentValue	12	A variable binding specifies a value that could be held by the variable but cannot be assigned to it at this time.
resourceUnavailable	13	An attempt to set a variable required a resource that is not available.
commitFailed	14	An attempt to set a particular variable failed.
undoFailed	15	An attempt to set a particular variable as part of a group of variables failed, and the attempt to then undo the setting of other variables was not successful.
authorizationError	16	A problem occurred in authorization.
notWritable	17	The variable cannot be written or created.
inconsistentName	18	The name in a variable binding specifies a variable that does not exist.

A.2 SNMP Request Validation Return Codes

This table specifies the SNMP Error Codes (defined in A1) returned on successful or failed validation of incoming SNMP requests

Description	SNMP mapped Error codes
Operation is successful	NoError
Managed object instance already exists.	noCreation
Managed object instance is not found.	noSuchName
Error to process attribute name list	wrongType, wrongLength, wrongEncoding
Error to process attribute value list	wrongType, wrongLength, wrongEncoding, wrongValue
The requesting client does not have permission to access operation.	noAccess
The managed object to be changed is read only.	notWritable
The requested operation is rejected due to some in-progress operation as defined operation concurrency matrix	genErr
Memory allocation error.	resourceUnavailable

Appendix B NBI Notifications

Please note that all un-supported events will be marked with gray letters.

B.1 NBI Supported Events

Event Type	Content
nbiEventRecord (Network Event Notification)	nbiNotiSequenceNum nbiEventId nbiEventType nbiEntityId nbiEntityName nbiManagedObjectUserName nbiSeverity nbiSourceDevice nbiDetectionEventTime nbiSourceEventTime nbiText nbiEventRecordId nbiSourceAgentIP
nbiManagementEvent (Management Event Notification)	nbiSequenceTag nbiCorrelationTag nbiEventId nbiEventType nbiEntityId nbiEntityName nbiManagedObjectUserName nbiSeverity nbiSourceDevice nbiDetectionEventTime nbiSourceEventTime nbiText nbiAdditionalInformation nbiMgmtEventType nbiMgmtReasonCode nbiSyncFileToken

where:

Integer Value	nbiEventType
1	Atribute-Value-Change Event
2	Communication Alarm
4	Equipment Alarm
6	Object-Creation Event
7	Object-Deletion Event
11	Quality-of-Service Alarm
13	Security Violation
255	NBI_Management Event
256	Informational Event
257	Management Event

Integer Value	nbiSeverity
1	CommFailure
2	Critical
3	Major
4	Minor
5	Warning
6	Clear
7	Info
8	Unknown

B.2 NBI Supported Management Events

This table specifies the supported event types used in management events to denote success or failure of the management operation. Receivers of this event (NMS) can use this event type to determine if the requested operation is a success or a failure.

nbiMgmtEventType	Integer Value	Description
transfer-Complete	2	
data-Committed	3	
job-Aborted	4	It is used for all failed operations with appropriate reason codes indicating the reasons of failures.
job-Complete	7	Successful Operations including file creations.
file-Transfer-Failed	11	

B.3 Reason Code of NBI Management Notifications

This table specifies various reason codes used when sending management event to NMS to denote success or failed management operation.

nbiMgmtReasonCode	Integer Value	Description
notUsed	0	Used when the operation is successful
unknown	1	
dataNotAvailable	2	Used when the data for processing an operation request is not available. For example, if no events are found within the time interval specifies in the sync events request, this reason code is used.
dataExtractionError	3	Successful Operations including file creations.
checksumError	4	
fileCreationError	5	
requestError	6	Used when management operation request is invalid.
operationConcurrencyError	7	Operation failure due to concurrency restrictions.

B.4 Additional Information of NBI Management Notifications

This table provides the possible event type, reason code, additional information that FM will use in the management event to denote operation status (success or failure).

nbiMgmt EventType	Mgmt Operation	nbiMgmt ReasonCode	nbiAdditional Information(deprecated)	Description
job- Complete	Sync Events	Not Used (0)	<opName><fileName>	Sent when sync operation is successful with event file generated. <filename> is the fully qualified file path containing the name of the generated file.
	Sync Alarms	Not Used (0)	<opName><fileName>	Sent when sync operation is successful with alarm file generated. <filename> is the fully qualified file path containing the name of the generated file.
job- Aborted	Sync Events/ Sync Alarms	fileCreation Error (5)	<opName> <specificReason>	Failure during file creation. <specificReason> can be one of the following <ul style="list-style-type: none"> • unKnown • errorInPreparation • hardDiskFull • hardDiskFailure • tooManyFiles • collectionTimeOut • incompleteTruncatedFile • corruptedFile • lowMemory
	Sync Events/ Sync Alarms	dataNot Available (2)	<opName>	Used when the data not found.
	Sync Events/ Sync Alarms	data Extraction Error (3)	<opName>	Used when unable to extract events and alarms from the database.
	Sync Events/ Sync Alarms	operation Concurrency Failure (7)	<opName>	Used when the operation was rejected due to conflict with another parallel operation.
	Sync Events/ Sync Alarms	requestError (6)		Used when the management operation request is invalid

¹ This column defines the names of the fields included in nbiAdditionalInformation (deprecated) for each notification. The format of nbiAdditionalInformation (deprecated) is one or more field values in XML tag format. For an example, the nbiAdditionalInformation (deprecated) of a job-complete notification is <opName>syncEvents</opName><fileName>/EMS/syncEvents/AL20021224.170000-20021224.174500-1130_EMSId </fileName>.

B.5 HTTP status codes using during synchronization file download

This table provides the possible HTTP status codes and information what was wrong during download xml file using NbiFileDownload.jsp script.

HTTP status code	Action	Error
200 (OK)	File has been downloaded	-
400 (Bad Request)	File has not been downloaded	There is no fileId parameter in NbiFileDownload.jsp or fileId parameter is empty.
		fileId parameter in NbiFileDownload.jsp script format is wrong.
		There is more than one file assigned to fileId passed to NbiFileDownload.jsp script.
404 (Not Found)	File has not been downloaded	There is no file assigned to fileId passed to NbiFileDownload.jsp script.
		File assigned to fileId parameter passed to NbiFileDownload.jsp cannot be read.

This page is intentionally blank

Appendix C File Format

The summary event/alarm file is an XML file whose structure is described by the xml schema defined in the section C.2. Both files have the same format and are differentiated by the **FileType** attribute. Also the file name can be used to determine if the file is an event summary file or an alarm summary file. The file naming convention is described in section C.1 below. If requestor (NMS) download mechanism honors "Content-Disposition" HTTP header, downloaded file will be saved to file which name will be according to this naming conventions.

C.1 File Name Convention

The following convention shall be applied for event/alarm summary file naming:

<managementData_type><Startdate>.<Starttime>-<Enddate>.<Endtime>[_<UniqueId>][_<RC>]

where the meaning of the fields is as follows:

Field	Description	Value
managementData_type	Type of the management data contained in the file	<ul style="list-style-type: none"> - "AL" for files containing Alarm data; - "EV" for files containing Event data;
Startdate / Enddate	Indicates the date when data collection on the FM started (finished - for Enddate)	YYYYMMDD where: <ul style="list-style-type: none"> - YYYY is the year in four-digit notation; - MM is the month in two digit notation (01 - 12); - DD is the day in two-digit notation (01 - 31)
Starttime / Endtime	Indicates the time when data collection on the FM started (finished - for Endtime)	HHMMshhmm where: <ul style="list-style-type: none"> - HH is the two-digit hour of the day (local time), based on 24-hour clock (00 - 23); - MM is the two digit minute of the hour (local time); - SS is the two digits second of the minute (local time); - s is the sign of the local time differential from UTC (+ or -), in case the time differential to UTC is 0 then the sign may be arbitrarily set to "+" or "-"; - hh is the two-digit number of hours of the local time differential from UTC (00-23); - mm is the two digit number of minutes of the local time differential from UTC (00-59).
UniqueId	This is the name of the FM or domain. The field may be omitted only if the distinguished name is not available	

Field	Description	Value
RC	The RC parameter is a running count, starting with the value of "1", and shall be appended only if the filename is not unique, i.e. more than one file is generated and all other parameters of the file name are identical. Notice that the delimiter for this field, __, is an underscore character (_), followed by a minus character (-), followed by an underscore character (_).	

Example:

AL20021224.170000-20021224.174500-1130_EMSId.xml

- file containing events,
- produced by EMS <EMSId> on December 24, 2002, for the time 17:00:00 local to 17:30:00 local, with a time differential of –11:30 hours against UTC.

C.2 XML Schema

This section captures the XML schema file (*eventSummaryFile.xsd*) for the Event Summary File. Alarm summary file follows the same schema with references to “eventSummaryFile” replaced with “alarmSummaryFile” in the schema file.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified">

  <xs:element name="eventSummaryFile">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Header" />
        <xs:element ref="eventRecord" minOccurs="0" maxOccurs="unbounded" />
        <xs:element ref="Footer" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="Header">
    <xs:complexType>
      <xs:attribute name="FileFormatVersion" type="xs:string"
        use="required">
        <xs:annotation>
          <xs:documentation>Version of the file format. Initial Version is
            "1.0". This version string will be updated if the format of the
            file changes.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>

      <xs:attribute name="SenderDN" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>Distinguished Name of the sender of this file.
            Typically this is the configured name of the Fault Manager sending
            the file.</xs:documentation>
        </xs:annotation>
      </xs:attribute>

      <xs:attribute name="SenderType" type="xs:string" use="required">
        <xs:annotation>
          <xs:documentation>Type of the sender. For This is defaulted to
            "EMS", if sender Type is not available.</xs:documentation>
        </xs:annotation>
      </xs:attribute>

      <xs:attribute name="StartDateTime" type="xs:dateTime"
        use="required">
        <xs:annotation>
          <xs:documentation>Date and Time of Start of event/alarm
            collection. Value will be in UTC dateTime format.
            Format: YYYY-MM-DDTHH:MM:SS+/-UTC offset.
            (eg . 2000-03-01T14:00:00+02:00)
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>

      <xs:attribute name="SoftwareVersion" type="xs:string"
        use="required">
        <xs:annotation>
          <xs:documentation>Version of the FM software
            associated with the file.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>

      <xs:attribute name="FileType">
        <xs:annotation>
          <xs:documentation>Specifies the type of the file. Since alarm and
            event summary files follow the same schema, the distinguishing
            attribute between the two files is this FileType attribute. For
```



```

    e.g. FileType is set to "alarmSummary" for alarm file and set to
    "eventSummary" for event file.</xs:documentation>
  </xs:annotation>
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="alarmSummary" />
      <xs:enumeration value="eventSummary" />
    </xs:restriction>
  </xs:simpleType>
</xs:attribute>
</xs:complexType>
</xs:element>

<xs:element name="eventRecord">
  <xs:complexType>
    <xs:attribute name="EventId" type="xs:string">
      <xs:annotation>
        <xs:documentation>Unique Class Identifier of the alarm/event
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="EventType" type="xs:string">
      <xs:annotation>
        <xs:documentation>Type of event/alarm. For e.g. state-cause
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="Severity">
      <xs:annotation>
        <xs:documentation>Severity of the event/alarm. Seveirity could be
        one of Major, Critical, Minor, Warning, Clear, Info.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="CommFailure" />
          <xs:enumeration value="Critical" />
          <xs:enumeration value="Major" />
          <xs:enumeration value="Minor" />
          <xs:enumeration value="Clear" />
          <xs:enumeration value="Warning" />
          <xs:enumeration value="Info" />
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>

    <xs:attribute name="SourceNEIP" type="xs:string">
      <xs:annotation>
        <xs:documentation>Managed Resource that is managing the Entity, reporting the
        event/alarm
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="EntityId" type="xs:string">
      <xs:annotation>
        <xs:documentation>Identifier of the managed entity having the
        problem or the condition that intiated the event/alarm
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="EntityUserName" type="xs:string">
      <xs:annotation>
        <xs:documentation>User defined name of the managed entity having
        the problem or the condition that intiated the event/alarm
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="SourceUserName" type="xs:string">
      <xs:annotation>

```

```

        <xs:documentation>User defined name of the source managed resource
        reporting the alarm</xs:documentation>
      </xs:annotation>
    </xs:attribute>

    <xs:attribute name="NEEventTime" type="xs:dateTime">
      <xs:annotation>
        <xs:documentation>Time of Event/Alarm generation at the network
        element. Follows the dateTime format. If NE does not report the
        time, this attribute will be set to "". For example, Feb 10, 2006
        at 11.45.40 AM EDT will be displayed as 2006-02-10T11:45:40.0-4:0.
        The last 4 digits are optional and represent timezone as difference from
        UTC. EDT is 4 hours behind Coordinated Universal Time (UTC).
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>

  <xs:attribute name="MgrEventTime" type="xs:dateTime">
    <xs:annotation>
      <xs:documentation>Time of Event/Alarm Detection on the manager. The
      format of the attribute follows the dateTime format. For example,
      Feb 10, 2006 at 11.45.40 AM EDT will be displayed as 2006-02-10T11:45:40-4:00.
      The last 4 digits are optional and represent timezone as difference
      from UTC. EDT is 4 hours behind Coordinated Universal Time (UTC).
    </xs:documentation>
  </xs:annotation>
</xs:attribute>

  <xs:attribute name="Text" type="xs:string">
    <xs:annotation>
      <xs:documentation>Description of the alarm</xs:documentation>
    </xs:annotation>
  </xs:attribute>

  <xs:attribute name="EventIdentifier" type="xs:string">
    <xs:annotation>
      <xs:documentation>Unique event identifier</xs:documentation>
    </xs:annotation>
  </xs:attribute>

  <xs:attribute name="ReportingAgent" type="xs:string">
    <xs:annotation>
      <xs:documentation>This attribute specifies the IP Address of the agent (device)
      reporting the event</xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
</xs:element>

<xs:element name="Footer">
  <xs:complexType>
    <xs:attribute name="EndDateTime" type="xs:dateTime" use="required">
      <xs:annotation>
        <xs:documentation>Date and Time of End of event/alarm collection.
        Value will be in UTC dateTime format. Format - YYYY-MM-DDTHH:MM:SS+/-UTC
        offset. (eg . 2000-03-01T14:30:00+02:00)
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
</xs:element>
</xs:schema>

```

C.3 Sample Alarm Summary XML File

```
<?xml version="1.0" encoding="UTF-8"?>

<eventSummaryFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="AlarmSummaryFile.xsd">

  <Header SenderType="EMS" FileFormatVersion="1.0"
    SoftwareVersion="Fault Manager 1" SenderDN="fmHostname"
    StartDateTime="2010-12-20T12:54:53+00:00" FileType="alarmSummary" />

    <eventRecord Severity="Critical"
      MgrEventTime="2010-12-20T10:09:39+00:00"
      SourceNEIP="10.1.1.0:Hpd_Site"
      Text="MALFUNCTIONED, NO REASON - scCommonChlText"
      EntityUserName="Channel 1"
      NEEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="scSiteName"
      EventType="Quality-of-Service_Alarm"
      EventId="hpd_channel_fault.5.1"
      EntityId="10.1.1.0:hpd_channel_fault:1" />

    <eventRecord Severity="Critical"
      MgrEventTime="2010-12-20T10:06:22+00:00"
      SourceNEIP="10.1.1.1:MotHpdSiteCtrl"
      Text="7 Unknown state, HARDWARE FAILURE - UNKNOWN severity, default
      to CRITICAL, scCommonFanText"
      EntityUserName="Fan"
      NEEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="10.1.1.1"
      EventType="Equipment_Alarm" EventId="hpd_sc_fan_fault.7.2"
      EntityId="10.1.1.1:hpd_sc_fan_fault:1.1" />

    <eventRecord Severity="Critical"
      MgrEventTime="2010-12-20T10:02:22+00:00"
      SourceNEIP="10.1.1.0:Hpd_Site"
      Text="SITE OFF, NO VOICE CHANNEL - scCommonSiteText"
      EntityUserName="Site"
      NEEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="scSiteName"
      EventType="Quality-of-Service_Alarm"
      EventId="hpd_site_fault.21.4"
      EntityId="10.1.1.0:hpd_site_fault:1" />

    <eventRecord Severity="Critical"
      MgrEventTime="2010-12-20T10:09:39+00:00"
      SourceNEIP="10.1.1.1:MotHpdSiteCtrl"
      Text="CRITICAL MALFUNCTION, MULTIPLE SCs WITH SAME scNUM -
      scCommonBoxText"
      EntityUserName="Site Controller"
      NEEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="10.1.1.1"
      EventType="Equipment_Alarm"
      EventId="hpd_site_controller_fault.6.4"
      EntityId="10.1.1.1:hpd_site_controller_fault:1" />

    <eventRecord Severity="Critical"
      MgrEventTime="2010-12-20T09:56:41+00:00"
      SourceNEIP="10.1.1.0"
      Text="At least one node in this subnet is in failed state."
      EntityUserName="10.1.1.0"
      NEEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="10.1.1.0"
      EventType="Equipment_Alarm"
      EventId="NULL"
      EntityId="10.1.1.0" />

  <Footer EndDateTime="2010-12-20T12:54:53+00:00" />
</eventSummaryFile>
```

C.4 Sample Event Summary XML File

```
<?xml version="1.0" encoding="UTF-8"?>

<eventSummaryFile xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="EventSummaryFile.xsd">

  <Header SenderType="EMS" FileFormatVersion="1.0"
    SoftwareVersion="Fault Manager 1" SenderDN="fmHostname"
    StartDateTime="2010-12-31T12:43:20+00:00" FileType="eventSummary" />

    <eventRecord Severity="Info"
      MgrEventTime="2010-12-31T12:00:19+00:00"
      SourceNEIP="10.7.233.20:NMServer"
      Text="Discovered Device Managed Resource - fmHostname.fmDomainName
        (10.7.233.20:Motorola Network Management General Purpose Server)."
      EntityUserName="Discovery"
      NEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="fmHostname.fmDomainName"
      EventType="Object-Creation_Event"
      EventId="DiscDeviceSuccess"
      EntityId="10.7.233.20:NMServer:DiscEntity"
      EventIdentifier="1"
      ReportingAgent="10.7.233.20 " />

    <eventRecord Severity="CommFailure"
      MgrEventTime="2010-12-31T12:00:19+00:00"
      SourceNEIP="10.7.233.20:NMServer"
      Text="Fault information for this device may be out of date. Reason:
        there may be missing event information, a loss of communication, or a
        failure to synchronize with the device."
      EntityUserName="Synchronization"
      NEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="fmHostname.fmDomainName"
      EventType="Communication_Alarm"
      EventId="OutOfSyncStatus"
      EntityId="10.7.233.20:NMServer:DeviceSync"
      EventIdentifier="2"
      ReportingAgent="10.7.233.20 " />

    <eventRecord Severity="CommFailure"
      MgrEventTime="2010-12-31T12:00:19+00:00"
      SourceNEIP="10.7.233.0" Text="At least one node in this subnet is in
        failed state."
      EntityUserName="10.7.233.0"
      NEEventTime="1970-01-01T00:00:00-00:00"
      SourceUserName="10.7.233.0"
      EventType="Equipment_Alarm"
      EventId="NULL"
      EntityId="10.7.233.0"
      EventIdentifier="3"
      ReportingAgent="10.7.233.20 " />

    <eventRecord Severity="Clear"
      MgrEventTime="2010-12-31T12:00:19+00:00"
      SourceNEIP="10.7.233.20:NMServer"
      Text="ENABLED, USER REQUESTED"
      EntityUserName="Application"
      NEEventTime="2010-12-31T12:00:19+00:00"
      SourceUserName="fmHostname.fmDomainName"
      EventType="Equipment_Alarm"
      EventId="nm_common_application_fault.2.2"
      EntityId="10.7.233.20:nm_common_application_fault:1"
      EventIdentifier="4"
      ReportingAgent="10.7.233.20 " />

    <eventRecord Severity="Clear"
      MgrEventTime="2010-12-31T12:00:19+00:00"
      SourceNEIP="10.7.233.20:NMServer"
      Text="Fault Manager received Synchronization Complete notification
        from the device."
      EntityUserName="Synchronization" NEEventTime="2010-12-
        31T12:00:19+00:00"
      SourceUserName="fmHostname.fmDomainName"
      EventType="Communication_Alarm"
```

```
        EventId="NMASyncComplete"  
        EntityId="10.7.233.20:NMServer:DeviceSync"  
        EventIdentifier="5"  
        ReportingAgent="10.7.233.20 " />  
  
<Footer EndDateTime="2010-12-31T12:43:20+00:00" />  
  
</eventSummaryFile>
```

Appendix D SNMP v3 USM Configuration

FM NBI supports a pre-defined set of SNMPv3 users for communication with the NMSs. Following table denotes the default USM configuration supported by the FM NBI.

	Motorola NMS User Security Model	Non Motorola NMS User Security Model
User Name	MotoNorthMotorola	MotoNorth
Context Name	""	""
SNMP Engine ID	Derived by FM based on MAC Address of the Ethernet interface over which the NBI is supported	Derived by FM based on MAC Address (by default) of the Ethernet interface over which the NBI is supported
Security Level	<ul style="list-style-type: none"> • without Authentication • without Privacy or Authentication • without Privacy or Authentication • with Privacy 	<ul style="list-style-type: none"> • without Authentication • without Privacy or Authentication • without Privacy or Authentication with Privacy • with Privacy
Authentication Protocol	Not Applicable	Not Applicable
Privacy Protocol	Not Applicable	Not Applicable

NOTE: If Authentication and/or Privacy is enabled, FM support the following protocols

Authentication Protocol: HMAC-SHA-96

Privacy Protocol: AES-128

Appendix E ASCII codes (7-bit) table

ASCII code	Character	ASCII code	Character	ASCII code	Character	ASCII code	Character
1		33	!	65	A	97	a
2		34	"	66	B	98	b
3		35	#	67	C	99	c
4		36	\$	68	D	100	d
5		37	%	69	E	101	e
6		38	&	70	F	102	f
7		39	'	71	G	103	g
8		40	(72	H	104	h
9		41)	73	I	105	i
10		42	*	74	J	106	j
11		43	+	75	K	107	k
12		44	,	76	L	108	l
13		45	-	77	M	109	m
14		46	.	78	N	110	n
15		47	/	79	O	111	o
16		48	0	80	P	112	p
17		49	1	81	Q	113	q
18		50	2	82	R	114	r
19		51	3	83	S	115	s
20		52	4	84	T	116	t
21		53	5	85	U	117	u
22		54	6	86	V	118	v
23		55	7	87	W	119	w
24		56	8	88	X	120	x
25		57	9	89	Y	121	y
26		58	:	90	Z	122	z
27		59	;	91	[123	{
28		60	<	92	\	124	
29		61	=	93]	125	}
30		62	>	94	^	126	~
31		63	?	95	_	127	
32		64	@	96	`		

Appendix F NBI Fields mapping

The table below defines mapping between NBI Network Event, NBI Event Summary File, Event Archive File and FM Event field names. Only fields that exist in NBI Network Event are mapped.

NBI Network Event	NBI Event Summary File	Event Archive File	FM Event Field
nbiSequenceTag	-	-	-
nbiCorrelationTag	-	-	-
nbiNotiSequenceNum	-	-	-
nbiEventId	EventId	Identifier	Identifier
nbiEventType	EventType	Category	Category
nbiEntityId	EntityId	FailureObject	Failure Object
nbiEntityName	EntityUserName	Entity	EntityName
nbiEventIdentifier	EventIdentifier	Event Id	Event ID
nbiEventRecordId	EventIdentifier	Event Id	Event ID
nbiSourceAgentIP	ReportingAgent	ReportingAgent	Reporting Agent
nbiManagedObjectUserName	SourceUserName	ManagedResource	Managed Resource
nbiSeverity	Severity	Severity	Severity
nbiSourceDevice	SourceNEIP	Source	Source
nbiDetectionEventTime	MgrEventTime	NeTimeStamp	NeTimeStamp
nbiSourceEventTime	NEEventTime	Date/Time	Date/Time
nbiText	Text	Message	Message
nbiAdditionalInformation	-	-	-

Appendix G REST Error Codes

It is assumed that all operations share the same error codes and error response format.

HTTP status code	Error
200 (OK)	-
400 (Bad Request)	invalid parameters or invalid data in request
401 (Not authorized)	basic user/password authentication failed
403 (Forbidden)	" <i>NBI Rest</i> " permission for user is required
404 (Not Found)	request not supported URL
503 (Service Unavailable)	Missing NBI license

This page is intentionally blank

