

# MX-ONE Provisioning Manager and MX-ONE Service Node Manager Web Services

INTERWORKING DESCRIPTION



## NOTICE

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Networks™ Corporation (MITEL®). Mitel makes no warranty of any kind with regards to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes.

No part of this document can be reproduced or transmitted in any form or by any means - electronic or mechanical - for any purpose without written permission from Mitel Networks Corporation.

## TRADEMARKS

The trademarks, service marks, logos and graphics (collectively "Trademarks") appearing on Mitel's Internet sites or in its publications are registered and unregistered trademarks of Mitel Networks Corporation (MNC) or its subsidiaries (collectively "Mitel") or others. Use of the Trademarks is prohibited without the express consent from Mitel. Please contact our legal department at [legal@mitel.com](mailto:legal@mitel.com) for additional information. For a list of the worldwide Mitel Networks Corporation registered trademarks, please refer to the website: <http://www.mitel.com/trademarks>.

© Copyright 2016, Mitel Networks Corporation

All rights reserved

## 1

## GENERAL

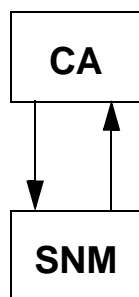
This is a description of the Web Services of the MX-ONE Provisioning Manager (PM) formerly Manager Provisioning (MP), and the MX-ONE Service Node Manager (SNM). The services are formally described in Web Services Description Language (WSDL) files.

This document explains some parts of the WSDL files and the communication behavior for both MX-ONE Service Node Manager and MX-ONE Provisioning Manager.

The usage of Web Services is limited to WSDL and the Simple Object Access Protocol (SOAP) over HTTP/HTTPS. No Universal Description, Discovery and Integration (UDDI), Web Services (WS) security or transaction handling is used.

The figures below show different Web Service scenarios.

In scenario 1, for SNM, the Client Application (CA) sends a request to SNM. SNM then sends a response back to the CA.

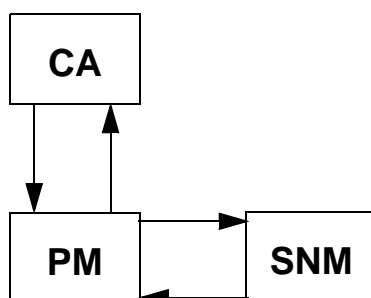


CA: Client Application

SNM: Service Node Manager

**Figure 1: Web Service Scenario 1, SNM**

In scenario 2, for PM, the Client Application (CA) sends a request to PM. PM then sends a request to the SNM. SNM sends back a response to PM and PM sends a response to the CA.



CA: Client Application

SNM: Service Node Manager

PM: Provisioning Manager

**Figure 2: Web Service Scenario 2, PM**

## 2

## WSDL

The WSDL file sets the directives for the interface. It contains the available operations, the valid arguments and in what format the operation call must be in.

The important settings are:

Style:document

Transport:<http://schemas.xmlsoap.org/soap/http>

WSDL files for MX-ONE Provisioning Manager and MX-ONE Service Node Manager can be downloaded from Service Plaza. The files are located under *MX-ONE-> Manager->Service Node Manager* and *MX-ONE-> Manager -> Provisioning Manager*.

## 3 CALL BEHAVIOR

### 3.1 SYNCHRONOUS INTERFACE

The interface is synchronous, which means that each request will receive a reply upon finished execution of the request. A new request can not be sent on the same session/connection until the last one has been responded. A client application needs to have multiple sessions/connections to be able to send several requests simultaneously.

### 3.2 ASYNCHRONOUS INTERFACE

It is possible to have an asynchronous interface, which means that the request will not get any response, and a response can be sent without any request.

### 3.3 SESSION HANDLING

Data and state information is not kept on the Manager between requests, that is, it is stateless. The only thing that can be looked up from previous requests is the last requests response with the `GetLastResponseStatus` command. The web server generates a Session ID at login so that authentication doesn't have to be done on following requests for the same session.

#### 3.3.1 FAILED REQUEST IN MX-ONE PROVISIONING MANAGER

Requests to MX-ONE Provisioning Manager that does not involve interactions with any of its subsystem will return a success or error message according to the success of the operation. A full request rollback will be performed.

If the request involves interactions with a subsystem, such as the MX-ONE Service Node, MiCollab Advanced Messaging, Mitel CMG, etc, then each subsystem will return it's own success statement to MX-ONE Provisioning Manager, which will compose a combined result to the submitter. If a subsystem request fails, that subsystem request will be fully rolled back. If no part of the operation is successful, an error message will be returned. If some subsystem requests are successful and some fails, a partial success message will be returned, identifying the parts that failed.

For the format of the response parameters, see section Response Parameters.

#### 3.3.2 FAILED REQUESTS IN MX-ONE SERVICE NODE MANAGER

For requests on single instances the response is either success or failed, and if it failed a full rollback is performed. Requests that spans over multiple instances are executed individually, and only the failed instance is rolled back. This applies also for requests that comes from MX-ONE Provisioning Manager.

The response object is the same as format as for MX-ONE Provisioning Manager.

## 3.3.3

## FAILED CONNECTION

The connection between the Soap Client and the Soap Server (PM Server) can break after a request has been sent. The SOAP server can also time out for unusually long requests. The client will in that case get an error message – Connection Timed out. Some other error messages that can be send to the Soap Client:

Connection refused – the host exists, nothing is listening for connections on that port, alternatively, a firewall is blocking that port.

Unknown host – the hostname components of the URL is invalid or the IP Phone Server is offline.

404: Not found – no web services at the exact URL.

The connection between the client and Manager can break after a request has been sent. The SOAP server can also time out for unusually long requests. The client will in that case not get a reply and will not know what to display or what actions to take. The Manager will therefore store the latest request response for each session. There is a web service operation that the client can use to read out the latest response when the connection is up again. That operation is called `GetLastRequestStatus`. The identifier in the request is the failed connection's Transaction ID, Session ID and Application ID.

The last request status will only be held in RAM. If the Manager goes down the status will be lost. The client will have to re-issue the request and determine what to do depending on the new response. This operation will be useful for glitches in the network or when the Manager is overloaded and not for system recovery.

Valid response messages from `GetLastRequestStatus` are:

Pending – The last request is still executing and no status can be returned. New `GetLastRequestStatus` requests are required until the execution has finished.

Last response – The last request has finished executing and the response will include a copy of the last response object.

Last response not found – The last request is not found, it probably never reached the Manager. The client should resend the last request.

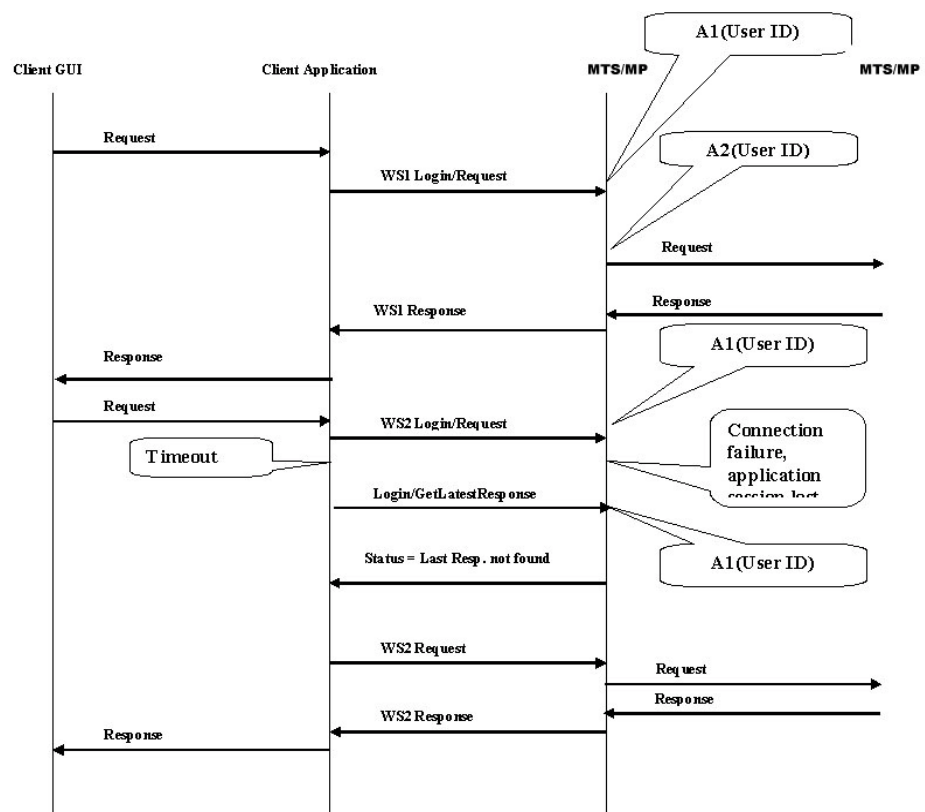
## 4 CALL PROCEDURES

### 4.1 LOGIN PROCEDURE

For MX-ONE Provisioning Manager and MX-ONE Service Node Manager there are no SOAP login operation from the client. For every request the user ID and password has to be sent as Tokens. See Token section under chapter Request Parameters.

### 4.2 REQUEST PROCEDURE

The requests are sent according to the published operations in the WSDL. It needs to include certain parameters and the actual task parameters. When the operation is received on the Manager side, the call is authorized before it is sent to the internal logic (through a façade with logging capabilities). An applicable task unit (Session Enterprise Java Bean) handles the request and passes it on to the Manager if applicable. A request can result in multiple requests to the Manager and also calls to other entities such as databases and other applications. Each request is handled within a transaction that enables roll back if it fails somewhere. A response message is returned to the client with success and/or error messages.



**Figure 3: Request Procedure including login for MX-ONE Service Node Manager**

The requests are sent according to the published operations in the WSDL. It needs to include certain parameters. A response message is returned to the client with success or error message and reason data if applicable).

## 4.3

## GETLASTREQUESTSTATUS PROCEDURE

Alternative procedures:

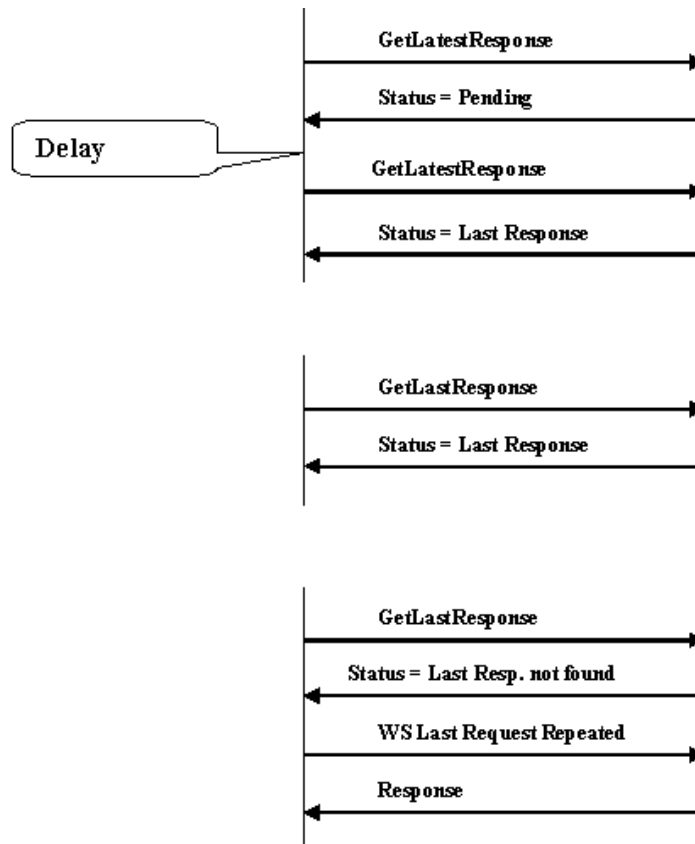


Figure 4: GetLastRequestStatus Procedure



## 5 AVAILABLE OPERATIONS

### 5.1 MX-ONE PROVISIONING MANAGER

All available operations are described in one single WSDL file. In MX-ONE Provisioning Manager it is called manage operation which handles Add, View, Change and Remove actions for each task.

**Note:** A task, for example, User, Department, Administrator, may result in SQL commands.

Example: AddUser

Data contents of the operations:

Input Request

1. Generic Request – type of action (add, change, view or remove)
2. Input requests – the TSA Value Object with:
  - only one task specific value object  
(a Xxx\_VO may only contain one Xxx\_VO)
3. Tokens value object

Output response

1. Generic Response – type of action (add, change, view or remove)
2. Output response – the TSA Value Object with:
  - The generic response value object
  - The same task specific value object as the input one, but probably modified
3. Tokens value object (same as for input)

All available operations are described in the WSDL file. They are set up in the **Manage Data** web service.

### 5.2 MX-ONE SERVICE NODE MANAGER

All available operations are described in one single WSDL file. They are normally set up as an Add, View, Change and Remove operation on each task.

**Note:** A task, for example, IP Extension, CSP, is not mapped directly to a single MML command. A task may result in none, one or several different MML commands, SQL commands and/or other types of commands.

Example: addIPExtension, changeCSP

Data contents of the operations:

Input Request

1. Generic Request – type of action (add, change, view or remove)
2. Input requests – the TSA Value Object with:
  - only one task specific value object

(a Xxx\_VO may only contain one Xxx\_VO)

3. Tokens value object

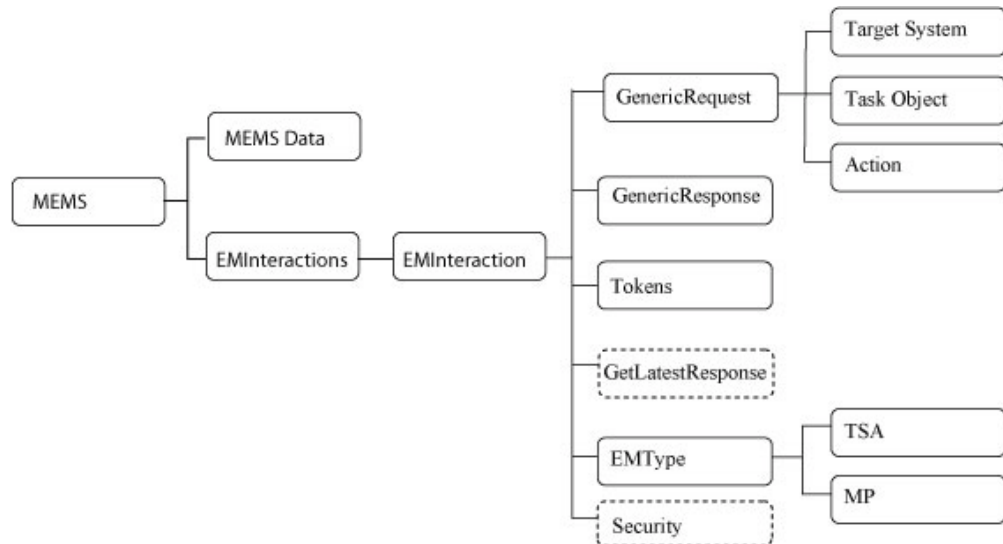
Output response

1. Generic Response – type of action (add, change, view or remove)
2. Output response – the TSA Value Object with:
  - The generic response value object
  - The same task specific value object as the input one, but probably modified
3. Tokens value object (same as for input)

All available operations are described in the WSDL file. They are set up in the **Manage Data** web service.

## 6 REQUEST PARAMETERS

### 6.1 GENERIC REQUEST



**Figure 5: General Request**

This is the top part of the data structure for requests to MX-ONE Provisioning Manager and MX-ONE Service Node Manager. Here the objects GenericRequest, GenericResponse, Tokens and the data containers TSA and PM are shown.

The GenericRequest specifies which target system that is addressed, e.g. TSA (MX-ONE Service Node Manager) or PM (MX-ONE Provisioning Manager).

The Task Object specifies which task that is addressed, e.g. User, IPExtension.

The Action specifies what kind of request it is, e.g. Add, Change, View, Remove.

### 6.2 TOKENS

There is a set of parameters, tokens, that always must be accompanied each request such as User ID, Password, Transaction ID, and so on. See the sections below.

#### 6.2.1 USER ID (LOGIN ONLY)

The User ID is the identity of the client application sending the request to the Manager. The client application must be defined and be assigned a role just as a human system administrator.

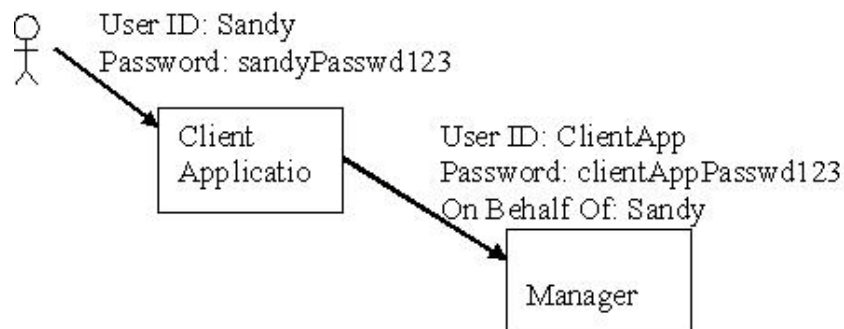
#### 6.2.2 PASSWORD (LOGIN ONLY)

This is the password of the User (User ID) in clear text format.

**Note:** If the SOAP client is considered trusted then the password may be omitted.

### 6.2.3 ON BEHALF OF

The On Behalf Of parameter represents the actual system administrator when the User ID represents a client application. It is needed for logging purposes.



**Figure 6: On Behalf Of parameter illustration**

### 6.2.4 APPLICATION ID

The Application ID is globally unique for each client.

### 6.2.5 SESSION ID (NOT LOGIN)

The Session ID is obtained during the client login procedure. It is passed with every request so that authentication doesn't have to be done on every request.

The Session ID is unique per Application.

### 6.2.6 TRANSACTION ID

The Transaction ID is generated by the client and passed with the request the whole way. The reason for it is to be able to follow the same request through the log files, audit trail, since a request is logged on both the client and the Manager.

The Transaction ID is unique per Session.

## 6.3 MISSING PARAMETERS

This applies only for optional parameters. An optional parameter means that the parameter can be totally omitted in a request and that is what we call a Missing Parameter, that is, it can be null as an object and missing as XML tag in the SOAP request. Missing parameters will be interpreted on the Manager side for the different types of operations like this:

#### **Add operations**

The missing parameter will get the default value on the Manager side if there is one.

#### **Other operations**

The missing parameter will be ignored on the Manager side.

## 6.4 EMPTY PARAMETERS

This applies only for optional parameters. An empty parameter means that it is included in the request but its value is empty/missing, that is, they will exist as an object but its value will be empty or null. Empty parameters shall only be used to clear or delete a value. If you want that parameter to be ignored, please use the Missing Parameter format instead. Note that numbers have the string format and not the integer format in the interface. Empty parameters will be interpreted on the Manager side for the different types of operations like this:

### Add operations

The empty parameter will not be replaced with the default value if there is one. It will remain empty if applicable.

### Change operations

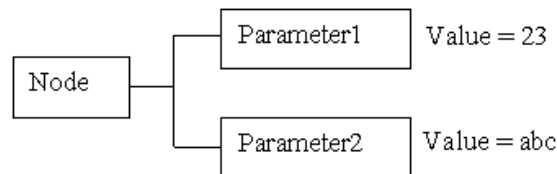
The empty parameter will clear the existing value of the parameter. If the parameter is not allowed to be cleared, do use the missing parameter format for parameters to be ignored.

### Other operations

The empty parameter will be ignored.

### 6.4.1 EXAMPLE

This example is as it appears in the SOAP request.



**Figure 7: Empty parameter**

Ex 1. A request with no empty or missing parameters:

```
<Node> <Parameter1> 23 </Parameter1> <Parameter2> abc </Parameter2> </Node>
```

Ex 2. A request with Parameter1 being a missing parameter, that is, for an add operation it would use the default value if there is one, and for other operations it would be ignored:

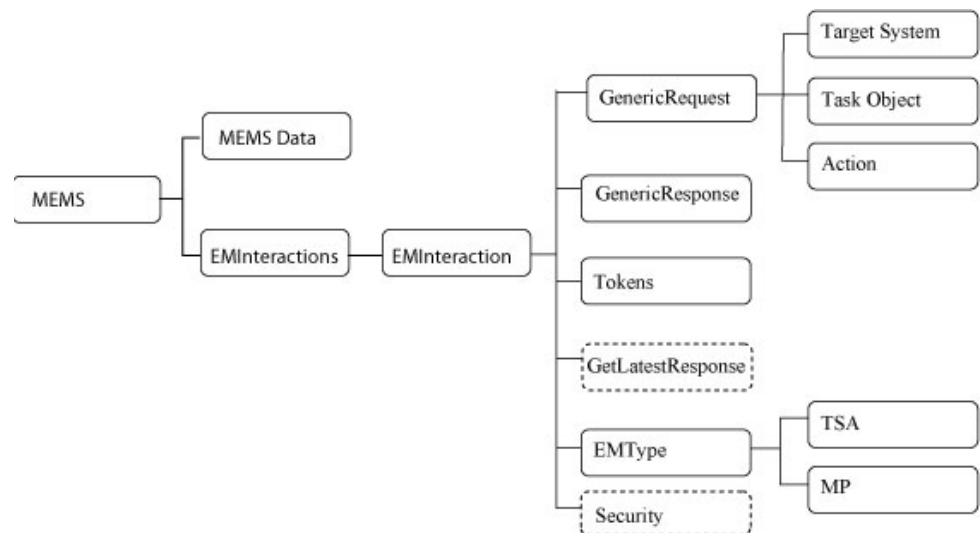
```
<Node> <Parameter2> abc </Parameter2> </Node>
```

Ex 3. A request with Parameter1 being an empty parameter, that is, for an add operation it would remain an empty string and for a change operation the existing value would be cleared:

```
<Node> <Parameter1> </Parameter1> <Parameter2> abc </Parameter2> </Node>
```

## 6.5 REQUEST FORMAT

A request contains two parts; a general part and a specific part. The general part is required for both SNM and PM requests. The general part is followed by a SNM specific or PM specific part.



**Figure 8: General Request**

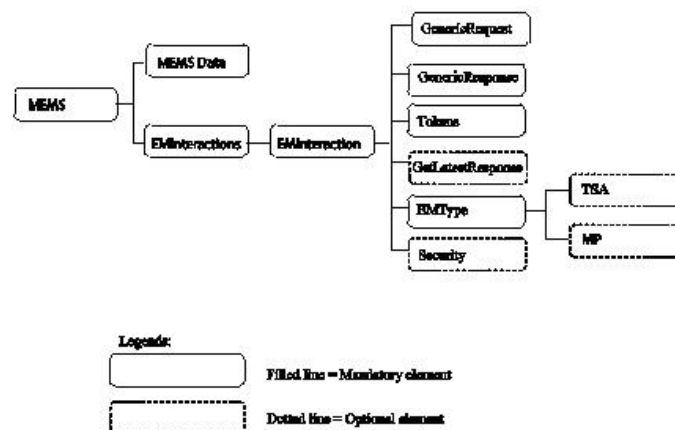
The specific part starts from TSA for SNM and PM for PM. For examples, see 6.5.2 Specific Request Examples, SNM on page 15 for SNM examples and see 6.5.3 Specific Request Examples, MX-ONE Provisioning Manager on page 16 for PM examples.

### 6.5.1

## GENERAL REQUEST

This is the format of the SOAP request.

It is displayed as a graphical view of the XML data structure:



**Figure 9: General Request Format**

Request: Login

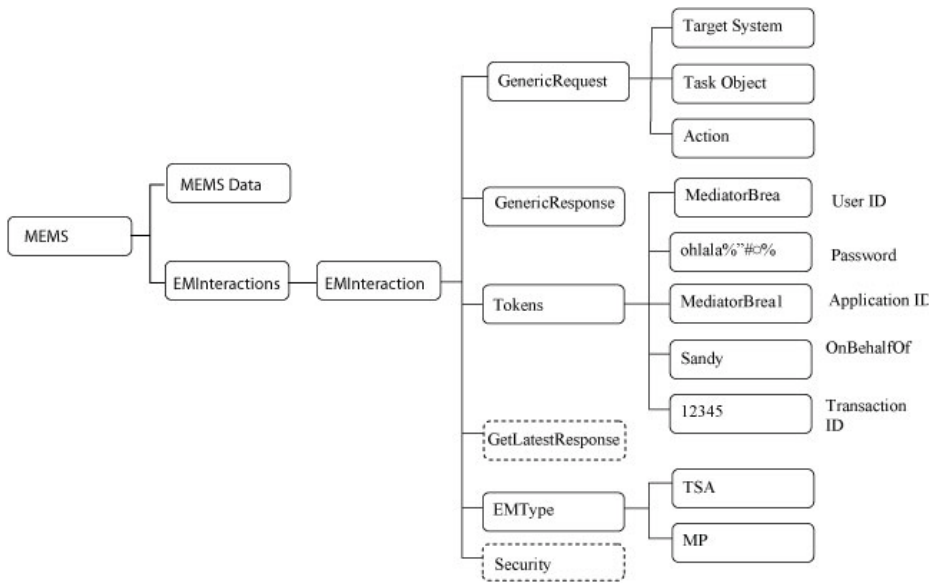


Figure 10: Login

6.5.2

SPECIFIC REQUEST EXAMPLES, SNM

**Note:** All request must start with the general part. The following examples only shows the specific part of the request.

Request: AddIPEExtension

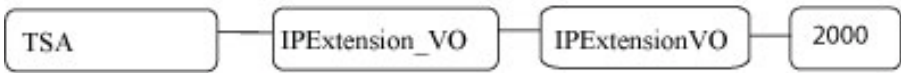


Figure 11: Add Account Code

Request: ViewIPEExtension

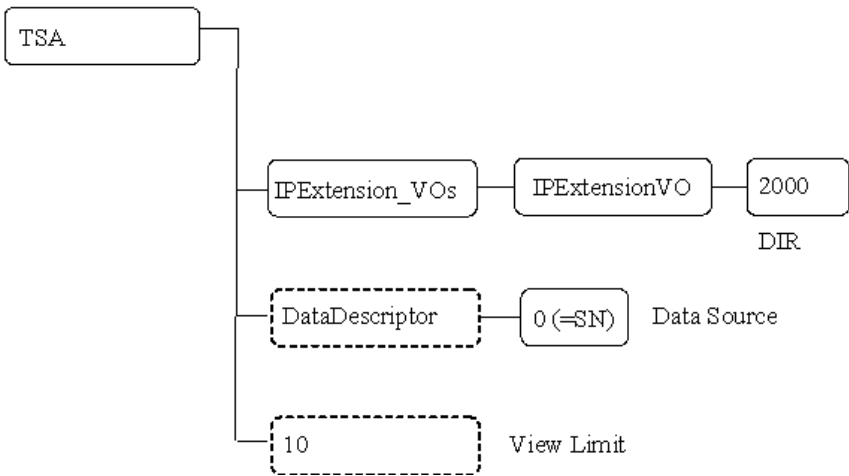


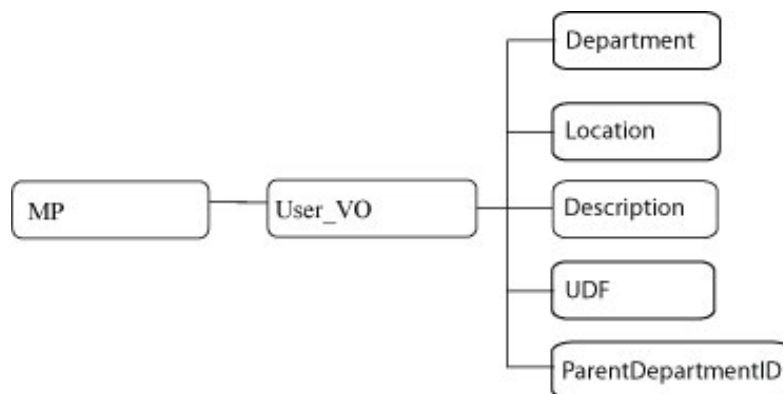
Figure 12: View IP Extension

## 6.5.3

## SPECIFIC REQUEST EXAMPLES, MX-ONE PROVISIONING MANAGER

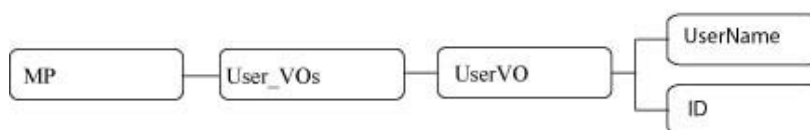
**Note:** All request must start with the general part. The following examples only shows the specific part of the request.

Request: AddUser



**Figure 13: Add User**

Request: ViewUser



**Figure 14: View User**

## 6.6

## MX-ONE PROVISIONING MANAGER

For the View and Remove operations, usually only the identity of the requested instance/instances will be included in the task specific value object. The element that represents the identity of a task is identified in the XML schema file.

For Add operations, the new data will be included in the task specific value object. If a parameter is empty, it will be replaced by the default value. See section Empty Parameters.

For Change operations, only the changed data needs to be included in the task specific value object. It is also allowed to include the complete value object. For more details please see sections Empty Parameters and Missing Parameters.



## 6.7

## MX-ONE SERVICE NODE MANAGER

For the View and Remove operations, usually only the identity of the requested instance/instances will be included in the task specific value object. The element that represents the identity of a task is identified in the XML schema file. For View operations, a limit of the returned instances (called ViewLimit) may also be included for performance reasons.

For View operations it is also possible to include a DataDescriptor in the request, indicating which data to operate on. Possible data sources are the SN, Default Values, Market Templates and Customer Templates. If there is no DataDescriptor in the request, the Manager will operate on the data in the SN. The DataDescriptor handling is currently only supported by the View operation, but will be implemented for all operations.

For Add operations, the new data will be included in the task specific value object. If a parameter is empty, it will be replaced by the default value. See section Empty Parameters.

For Change operations, only the changed data needs to be included in the task specific value object. It is also allowed to include the complete value object. For more details please see sections Empty Parameters and Missing Parameters.

## 7

## RESPONSE PARAMETERS

### 7.1

### TASK SPECIFIC VALUE OBJECT

The only time the task specific value object is returned to the client is in View operations and the searched instances are found. In that case the same task specific value object that is passed in is returned to the client. It will be filled with the return data.

**Note:** The View return task specific value object may be limited to a certain number of instances according to the ViewLimit element set in the request. The client has to repeat the operation with a modified range excluding the already returned instances to get the next set of existing instances.

For Add, Change and Remove operations, no task specific value object is returned.

For more info on task specific value objects see section Task Specific Value Object under section Request Parameters.

### 7.2

### TOKENS

The same Tokens object that is passed in is returned to the client.

**Note:** This could eventually be optimized, but to keep it simple for now the contents stays the same.

For more info see the Tokens section under section Request Parameters.

### 7.3

### GENERIC RESPONSE

A generic response is returned for all requests.

It holds one or more Info elements that each has a set of elements: ID, OK, Reason, Parameter (Field Name, Value and Item), Rollback Status and Message. Generic Response should be read like this (not View):

1. Check the OK for each Info, for example, each instance operated on.
2. If OK is true – The operation was successful for that ID. More info can be found in Message.
3. If OK is false – The operation failed for that ID, see also Reason, Parameter and Rollback Status.

There is a special case for View operations. It will only return one Info item in the Generic Response. It is either an OK or an erroneous Info.

Generic Response for View Single should be read like this:

1. Check OK in Info, there is only one Info.
2. If OK is true – the whole operation was ok, the ID holds the entered ID.
3. If OK is false – the operation failed and it is represented by the Info element. No task specific value object will be returned. With failed operation means that there was a malfunction on the process, syntax error in the input or the instance was not found.

Generic Response for View Range/Multiple should be read like this:

1. Check OK in Info, there is only one Info.
2. If OK is true – the whole operation was ok, the ID holds the entered range.
3. If OK is false – the operation failed on one instance and that one is represented by the Info element. Successful instances will not be returned in the task specific value object. It is an all or nothing scenario. With failed operation means that there was a malfunction on the process or syntax error in the input. No instances found are not considered a fault.

Response is returned for all requests. The following parameters can be included in the Soap Response:

### 7.3.1

#### ID

The ID represents the identity of the instance being operated on. It could be empty if it is not possible to point an ID of an instance to the fault. For example, for IP Extensions it is the directory number, for Login it is the client identity.

In the View operation, the ID holds the range as it was entered in the GUI since the result is returned in only one Info element.

The XML schema file has information in each task which element that acts as identity.

The ID is allowed to be empty if it is not possible to pinpoint an ID of an instance to the fault.

### 7.3.2

#### OK

A Boolean that is true for successful operations and false for failed operations.

If OK is true – The operation was successful for that ID. More info can be found in Message. If OK is false – The operation failed for that ID, see also Reason, Parameter and Rollback Status.

### 7.3.3

#### REASON

A label representing the cause of error and that can be localized on the client side.

It is only allowed to use the following reason values for MX-ONE Service Node Manager:

- TSA.REASON.ALREADY\_INITIATED
- TSA.REASON.CONNECTION\_ERROR
- TSA.REASON.COMBINATION\_NOT\_ALLOWED
- TSA.REASON.EQUIPMENT\_POSITION\_NOT\_EQUIPPED
- TSA.REASON.FEATURE\_INITIATED
- TSA.REASON.FEATURE\_NOT\_ALLOWED
- TSA.REASON.FEATURE\_NOT\_INITIATED
- TSA.REASON.INDIVIDUAL\_AUTHORIZATION\_CODE\_ASSIGNED
- TSA.REASON.INTERNAL\_ERROR
- TSA.REASON.LICENSE\_IS\_NOT\_AVAILABLE
- TSA.REASON.LICENSE\_SERVER\_IS\_BLOCKED
- TSA.REASON.LIM\_BLOCKED

- TSA.REASON.MISSING\_MANDATORY\_OBJECT
- TSA.REASON.MISSING\_MANDATORY\_FIELD
- TSA.REASON.MUST\_BE\_REMOVED\_LAST
- TSA.REASON.NO\_DATA\_FOUND
- TSA.REASON.NO\_FREE\_LICENSE
- TSA.REASON.NO\_FREE\_RESOURCE
- TSA.REASON.NOT\_ALLOWED
- TSA.REASON.NOT\_ALLOWED\_IN\_USE
- TSA.REASON.NOT\_INITIATED
- TSA.REASON.NOT\_IN\_NUMBER\_SERIES
- TSA.REASON.NOT\_MARKET\_SET
- TSA.REASON.NOT\_SUPPORTED\_ACTION
- TSA.REASON.NOT\_SUPPORTED\_OBJECT
- TSA.REASON.OTHER\_EXTENSION\_TYPE
- TSA.REASON.OTHER\_NUMBER\_TYPE
- TSA.REASON.ROLLBACK\_FAILURE
- TSA.REASON.SYNTAX\_ERROR
- TSA.REASON.TEMPORARY\_CONGESTION
- TSA.REASON.WRONG\_HWTYPE
- TSA.REASON.WRONG\_NUMBER\_TYPE
- TSA.REASON.NOT\_IN\_RANGE
- TSA.REASON.LIM\_DOES\_NOT\_EXIST
- TSA.REASON.NOT\_ASSIGNED
- TSA.REASON.EQUIPMENT\_NOT\_ASSIGNED
- TSA.REASON.LIM\_MISSING
- TSA.REASON.ERROR\_READING\_FILE
- TSA.REASON.FEATURE\_ALREADY\_INITIATED
- TSA.REASON.INVALID\_FILE\_FORMAT
- TSA.REASON.FILE\_SIZE\_EXCEEDED
- TSA.REASON.DIRECTORY\_IS\_FULL
- TSA.REASON.FILE\_NOT\_FOUND
- TSA.REASON.MAX\_OP\_IN\_A\_LIM
- TSA.REASON.NOT\_BLOCKED
- TSA.REASON.NOT\_IN\_NUMBER\_SERIES\_OR\_TSA.REASON.WRONG\_NUMBER\_TYPE
- TSA.REASON.MULTI\_PARTY\_CONGESTION
- TSA.REASON.SECURITY\_EXCEPTION
- TSA.REASON.RELOAD\_IN\_PROGRESS

- TSA.REASON.NOT\_ALLOWED\_REMOTE

It is only allowed to use the following reason values for MX-ONE Provisioning Manager:

- MP.REASON.ALREADY\_INITIATED
- MP.REASON.CONNECTION\_ERROR
- MP.REASON.COMBINATION\_NOT\_ALLOWED
- MP.REASON.FEATURE\_INITIATED
- MP.REASON.FEATURE\_NOT\_ALLOWED
- MP.REASON.FEATURE\_NOT\_INITIATED
- MP.REASON.FIELD\_SIZE\_EXCEEDED
- MP.REASON.INTERNAL\_ERROR
- MP.REASON.MISSING\_MANDATORY\_OBJECT
- MP.REASON.MISSING\_MANDATORY\_FIELD
- MP.REASON.NO\_DATA\_FOUND
- MP.REASON.NOT\_ALLOWED
- MP.REASON.NOT\_ALLOWED\_IN\_USE
- MP.REASON.NOT\_INITIATED
- MP.REASON.NOT\_SUPPORTED\_ACTION
- MP.REASON.NOT\_SUPPORTED\_OBJECT
- MP.REASON.SYNTAX\_ERROR
- MP.REASON.NOT\_ASSIGNED
- MP.REASON.FEATURE\_ALREADY\_INITIATED
- MP.REASON.ALREADY\_EXISTS
- MP.REASON.ERROR\_READING\_FILE
- MP.REASON.INVALID\_FILE\_FORMAT
- MP.REASON.ERROR\_MISSING\_FILE
- MP.REASON.NO\_PRIVILEGES
- MP.REASON.NOT\_IN\_NUMBER\_SERIES
- MP.REASON.OTHER\_EXTENSION\_TYPE
- MP.REASON.OTHER\_NUMBER\_TYPE
- MP.REASON.ROLLBACK\_FAILURE
- MP.REASON.AUTHENTICATION\_FAILED

See section Localization of Response for more details on the format of the Reason values.

A label representing the cause of error and that can be localized on the client side. It can return the following reason values:

```
CONNECTION_REFUSED
500_INTERNAL_ERROR
MOST_LIKELY_A_NET_ERROR
302_MOVED
```

NETWORK\_ERROR  
 404\_NOT\_FOUND  
 NOT\_TRANSLATED\_ERROR\_MESSAGE  
 OTHER\_3xx\_RESPONSE  
 UNKNOWN\_HOST  
 WEB\_SERVICE\_ERROR  
 WEB\_SERVICE\_IP\_ADDRESS  
 WEB\_SERVICE\_NOT\_FOUND  
 WRONG\_CONTENT\_TYPE  
 XML\_PARSER\_ERROR

### 7.3.4

## PARAMETER

Parameter holds the following three sub elements:

- **Field Name**, represents the element who's Value was erroneous.
- **Value**, represents the actual value that was entered in the GUI.
- **Item**, represents a certain item of the Value if Value represents multiple items.

It is ok to use either Value or Item or both depending on the situation.

There may be no, one or several Parameter elements. Several Parameter elements mean that the fault is caused by an erroneous combination of field values.

See section Localization of Response for more details on the format of the Field Name string.

Parameter holds three sub elements: Field Name, Value and Item.

The Field Name represents the element who's Value was erroneous. The Value represents the faulty value.

The Item represents a certain item of the Value if Value represents multiple items.

It is allowed to use either Value or Item or both depending on the situation.

There may be no, one or several Parameter elements. Several Parameter elements mean that the fault is caused by an erroneous combination of field values.

### 7.3.5

## MESSAGE

Message is meant as a placeholder for extra information for both successful and erroneous responses. Each message is unique for a task and must be localized.

Message is meant as a placeholder for extra information for both successful and erroneous responses.

### 7.3.6

## ROLLBACK STATUS

Rollback Status indicates whether a rollback was successful or not in case of a failed operation. If no rollback is performed this element will be missing.

Accepted values:

ok – successful rollback

nok – unsuccessful rollback

**Note:** In the future, the rollback status could also include information on the reason why it failed.

7.4 RESPONSE FORMAT

A response contains two parts; a general part and a specific part. The general part is required for both SNM and PM responses. The general part is followed by a SNM specific or PM specific part.

The general part is shown in on page 23.

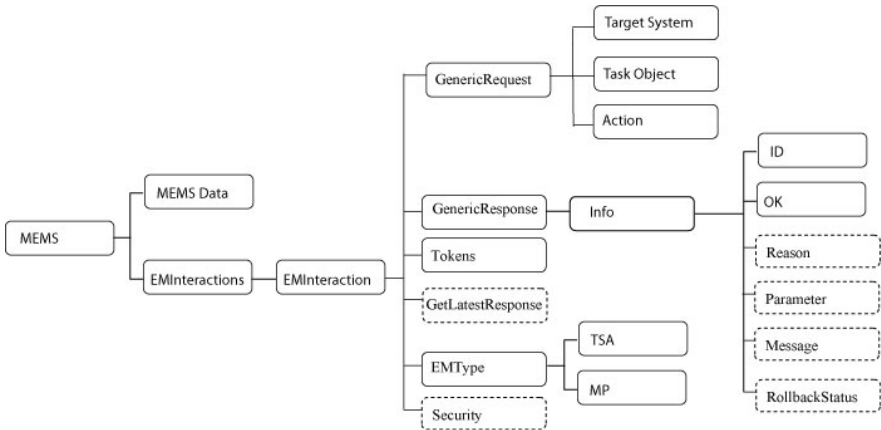


Figure 15: General Response

The specific part starts from TSA for SNM and PM for PM. For examples, see 7.4.2 Response Examples, MX-ONE Service Node Manager on page 24 for SNM examples and see 7.4.3 Response Examples, MX-ONE Provisioning Manager on page 25 for PM examples.

7.4.1 GENERIC RESPONSE

This is the format of the structure of the SOAP response.

Generic Response, a graphical view of the XML data structure:

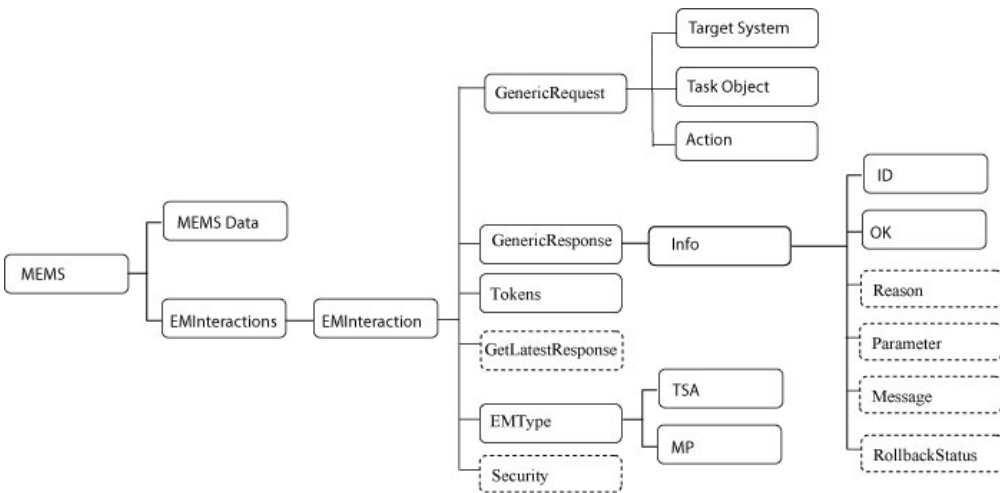


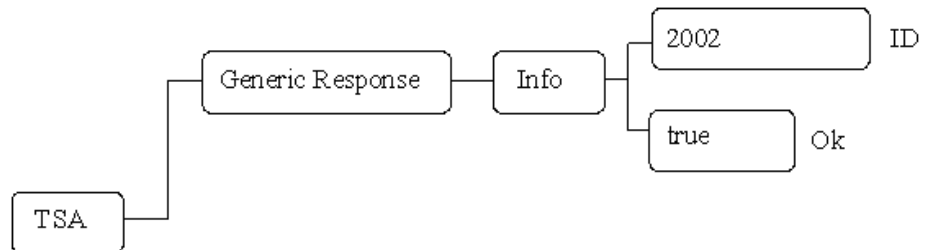
Figure 16: General Response Format

## 7.4.2

## RESPONSE EXAMPLES, MX-ONE SERVICE NODE MANAGER

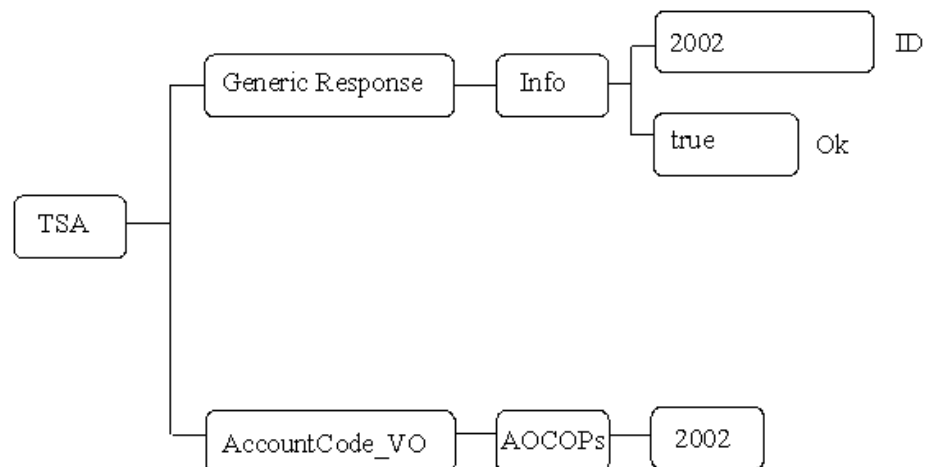
**Note:** All responses must start with the general part. The following examples only shows the specific part of the response.

Successful task response for Add, Change and Remove:



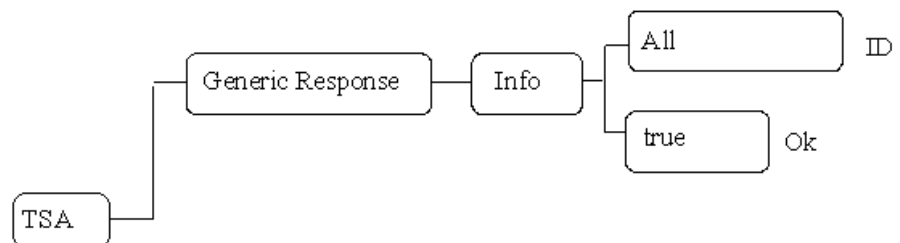
**Figure 17: Task Response for Add, Change and Remove**

Successful task response for View:



**Figure 18: Task Response for View**

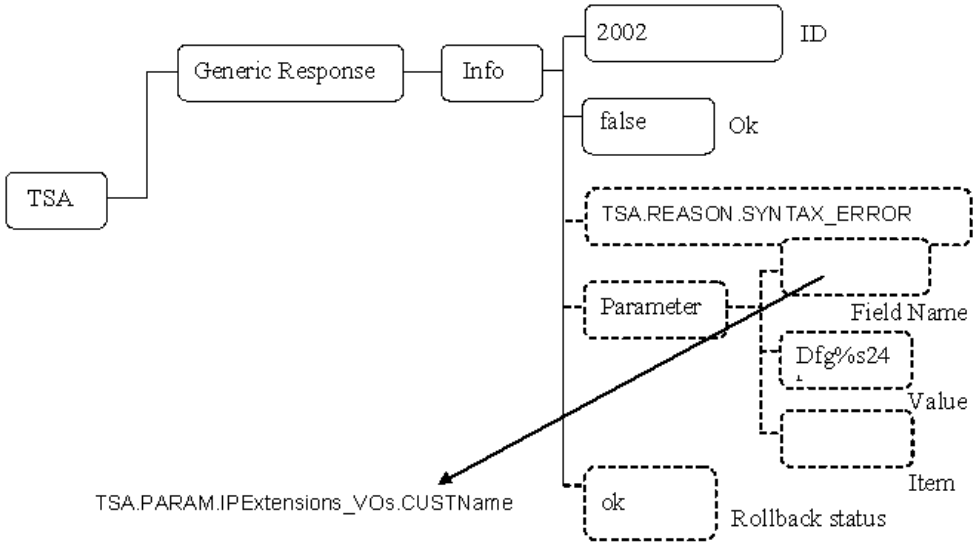
Successful task response for View All when no existing data:



**Figure 19: Task Response View All**

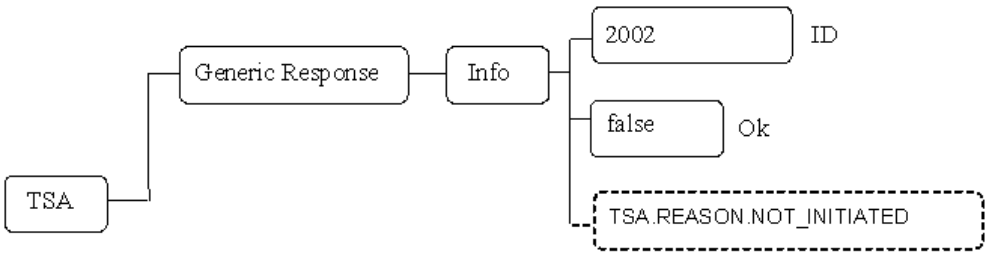
Erroneous task response for Add, Change, Remove:





**Figure 20: Erroneous Task Response Add, Change and Remove**

Erroneous task response for View:



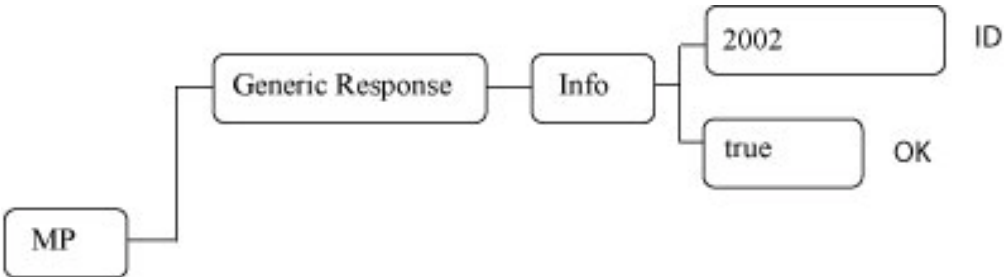
**Figure 21: Erroneous Task Response View**

7.4.3

RESPONSE EXAMPLES, MX-ONE PROVISIONING MANAGER

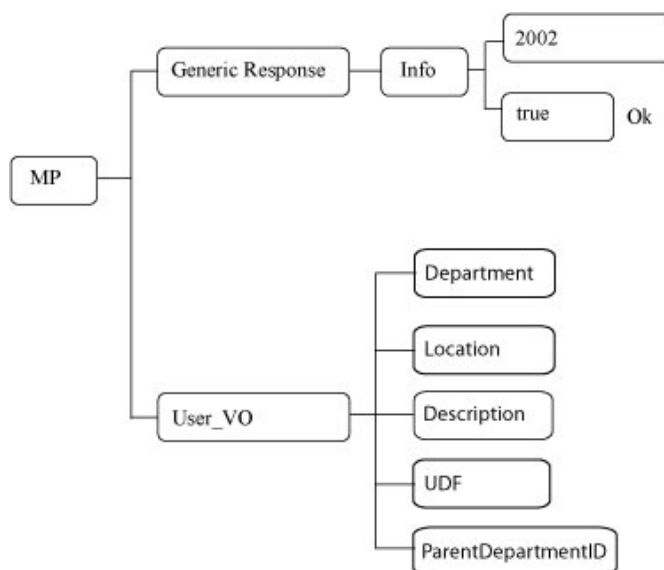
**Note:** All responses must start with the general part. The following examples only shows the specific part of the response.

Successful task response for Add, Change and Remove:



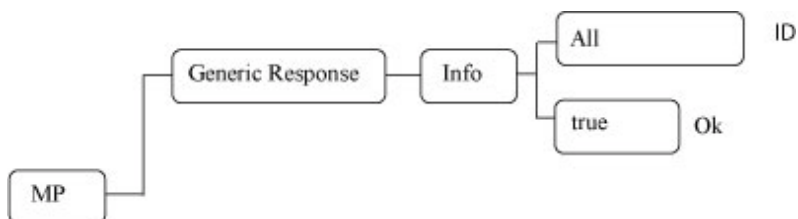
**Figure 22: Successful Task Response Add, Change and Remove**

Successful task response for View:



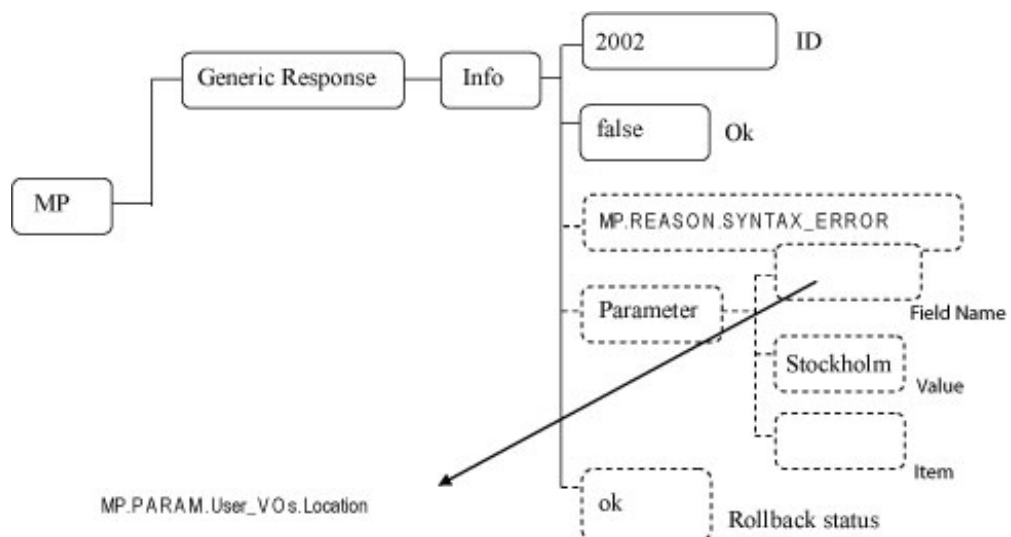
**Figure 23: Successful Task Response View**

Successful task response for View All when no existing data:



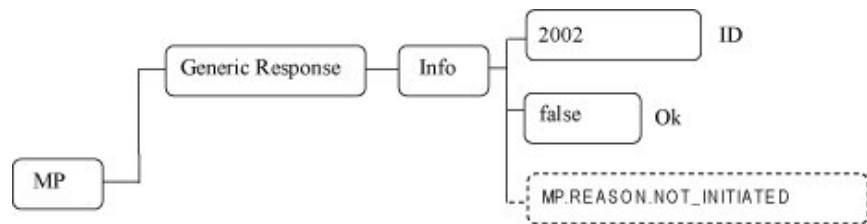
**Figure 24: Successful Task Response View All**

Erroneous task response for Add, Change, Remove:



**Figure 25: Erroneous Task Response Add, Change and Remove**

Erroneous task response for View:



**Figure 26: Erroneous Task Response View**

## 7.5

## LOCALIZATION OF RESPONSE

Localization is to be done by the client application. To simplify localization there are some rules that the Manager apply:

Format of elements that should be localized:

- Plain text
- Descriptive name, that is, not a number
- No spaces
- Prefixed by name-spaces

Elements that are formatted to be localized:

|            |                 |
|------------|-----------------|
| Reason     | prefix: REASON  |
| Field Name | prefix: PARAM   |
| Message    | prefix: MESSAGE |

Everything is prefixed by the application type, for example, TSA for MX-ONE Service Node Manager and PM for MX-ONE Provisioning Manager.

## 8

# REQUEST PARAMETERS AND FORMAT, FOR MX-ONE PROVISIONING MANAGER

The format of the parameters of all tasks that are published are described in XML schema structure and included in the WSDL file.

A SOAP message is sent to the PM Server. This is the format of the SOAP request. It is displayed as a graphical view of the XML data structure

### 8.1

## MANAGE DATA

The following actions can be performed:

- Add
- Change
- Remove
- View

For information on how to manage data, see the descriptor files.