MTOSI Web Services Design Guidelines

Abstract

This supporting document captures all the design guidelines applied to the development of the mTOP DDP Web Services Interface Implementation Specifications (IIS). The intent of this document is to be an mTOP contributor helper's guide for the generation of the MTOSI R2.x WS IIS for any given DDP.

Table of Contents

1 Introduction

1.1 Overview

The purpose of this document is to be the unique design reference for the development of all mTOP WS Interface Implementation Specifications (IIS).

The use of the XML term in this document is generally applicable to all the XML-based artifacts (Ref 1) of the mTOP DDP IIS, such as WSDL and XSD files.

These mTOP Web Services design guidelines use examples that are available In the release distribution.

These XML guidelines are partitioned into the following sections:

- **General Guidelines**

Architecture, naming conventions, documentation, etc

- **WSDL Guidelines**

Usage of the Web Service Description Language (WSDL) specifications

- **XML Schema Guidelines**

Usage of the XML Schema (XSD) specifications

- **XML Guidelines**

Usage of XML documents

1.2 Context

The definition of these mTOP WS guidelines is driven by the development of the MTOSI Release 2.x mTOP product. The writing of these XML guidelines has been made possible based on the experience the mTOP team gained from the previous MTOSI releases (1.0 and 1.1). And, these guidelines are a necessary contribution to ensure the successful development of the mTOP DDP WS IIS for the MTOSI Release 2.x.

1.3 Future

These WS guidelines are not a static document. The following factors will certainly influence the evolution of these guidelines sometime in the future:

- mTOP Methodology: The idea of generating the IIS (WS, IDL) from an information model (UML) is still the targeted goal. The guidelines will therefore have to be adapted to the process/tooling chosen.
- Harmonization/Convergence with other standards (SID, OSS/J, 3GPP, etc) will also lead to an evolution of these guidelines toward common industry-wide (Telecom/IT) WS guidelines.

2 General Guidelines

2.1 Introduction

This section describes all the general mTOP Web Services design guidelines.

The mTOP WS IIS provide an XML-based representation of the mTOP Service Interfaces that can readily be used for implementation. Important architecture aspect covered in this section includes:

- Standard Specifications – description of the XML-based standard specifications the mTOP WS IIS relies upon
- XML Modules – description of the mTOP XML-based components used in the definition of the mTOP Service Interface

2.2 Standard Specifications

The figure below (Figure 1) provides a schematic view of an mTOP WS Service Interface in terms of all the XML standard specifications used in its definition.

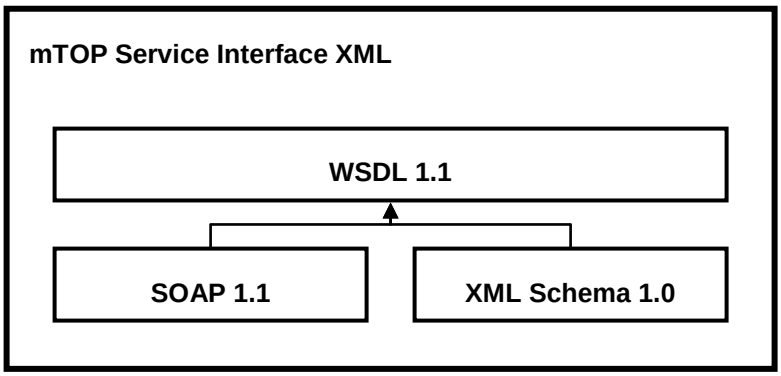

Figure 1 mTOP WS Service Interface XML Specifications View

As with most WS* based specifications (OASIS, etc), the mTOP SI is based on the following XML specifications from the W3C.

- Web Service Description Language version 1.1 (WSDL) [Ref 2] for the mTOP SI description using the Web Services specifications
- Simple Object Access Protocol version 1.1 (SOAP) [Ref 3], which is referenced by the WSDL binding definitions, and
- XML Schema version 1.0 (XSD) [Ref 4] used for the semantic description of the mTOP SI XML message parts.

Note that this document does not include details of the evaluation of all XML-based standard specifications that lead to the choice of the above (early stage of the MTOSI analysis). And, it is clear that WSDL and XML Schema are still the appropriate choice with respect to the industry directions regarding the Web Services (WS) and Service Oriented Architecture (SOA) developments.

Also, the following additional specifications are used as standard helpers/guidelines in the definition of the mTOP WS IIS:

- WS-I Basic Profile 1.2 [Ref 5]
- W3C XML Schema Patterns for Databinding [Ref 6]

2.3 XML Modules (Components View)

The definition of the mTOP WS specifications of an mTOP Service Interface are based on the concept of an XML module. Characteristics of the mTOP XML module follow:

- It is either WSDL or XML Schema
- It is associated with a unique namespace
- It is version controlled independently of other mTOP XML modules
- It is as much as possible a single file
- It encapsulates a (small) set of mTOP WS specifications for a well-defined mTOP data concept (an interface, an object, an event, a grouping of parameters, etc)

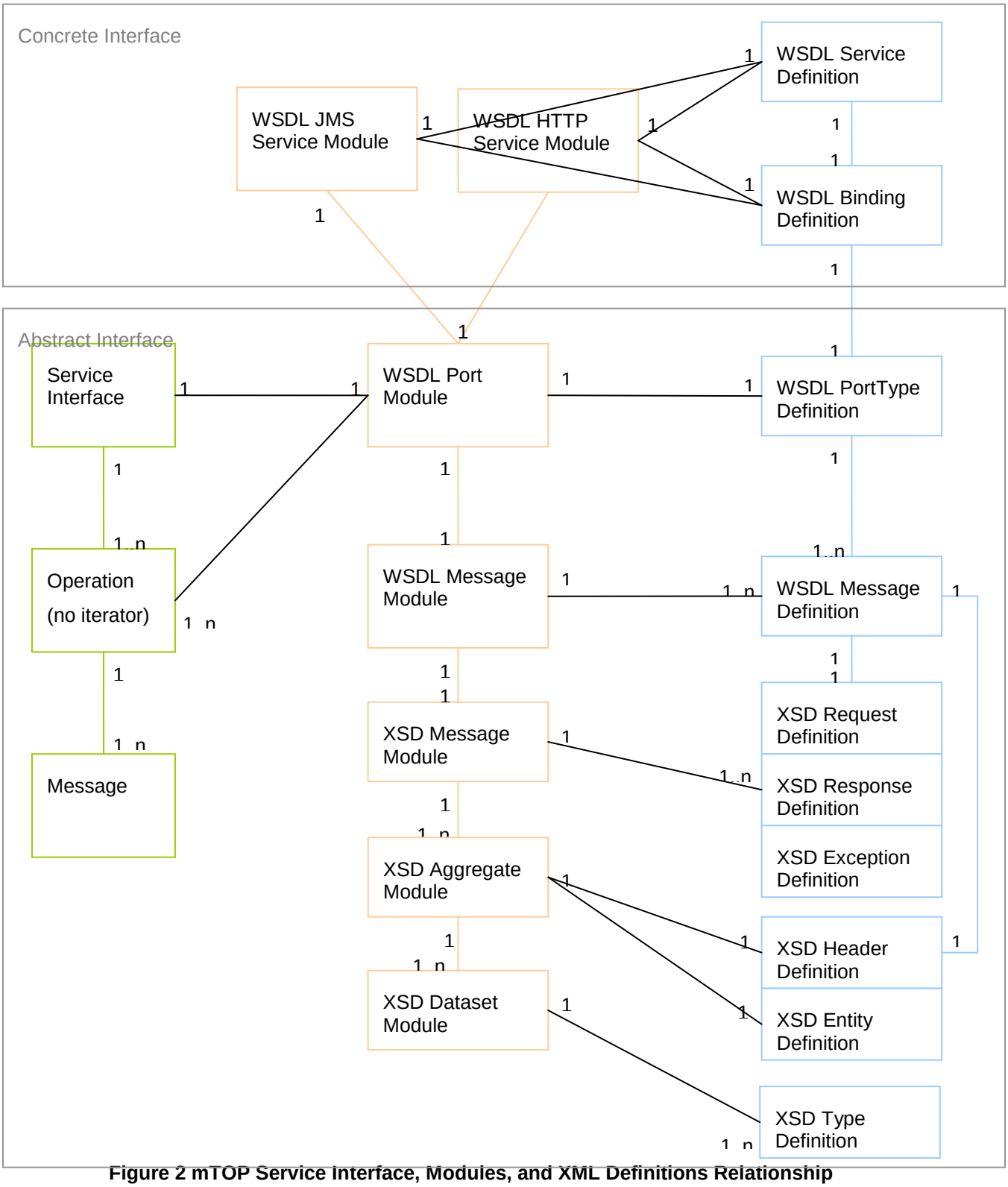

Figure 2 mTOP Service Interface, Modules, and XML Definitions Relationship

Figure 2 is a diagram that highlights the relationship between the basic concept of a service interface and the mTOP XML (WSDL/XSD) module concept and their contents in terms of WS specification definitions.

Figure 3 provides a component's view of the ***ManagedElementRetrieval*** mTOP Service Interface (defined in the ***ManageResourceInventory*** mTOP DDP) in terms of mTOP XML modules, which are defined across various mTOP DDPs.

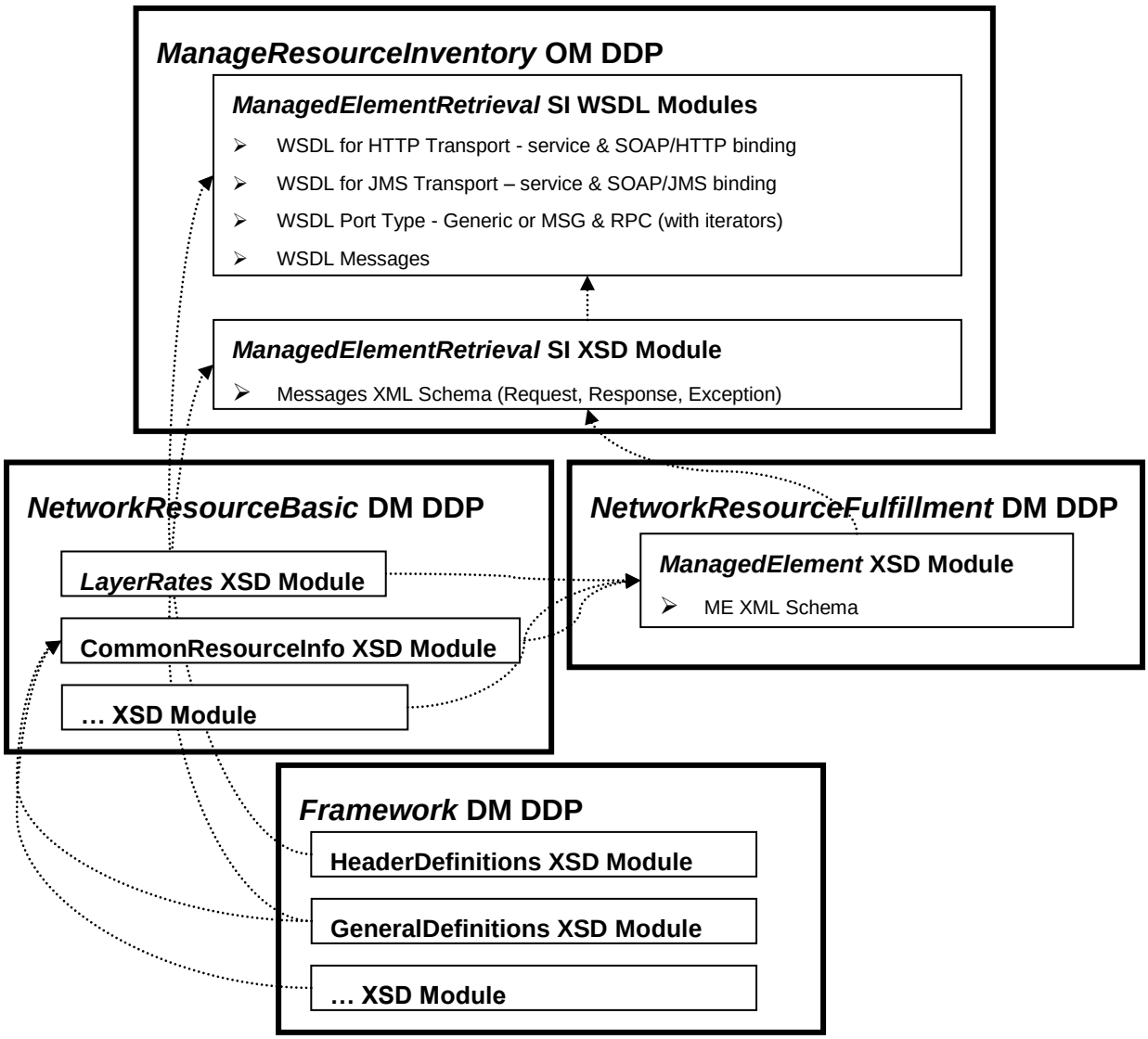

Figure 3 mTOP Service Interface Component's View (XML Modules)

Figure 3 Notes:

- All arrows represent the import (wsdl or xsd) of a specification (follow arrow direction)

- In the case of data retrieval service interfaces using iterator(s), two WSDL port type definitions (RPC & MSG) are required. These two definitions are integrated into the respective WSDL HTTP & JMS service modules.
- The challenge for these WS design guidelines resides in the decomposition of all the mTOP Service Interface WS specifications into XML modules.

2.4 Development View

The development view is established by the mTOP SourceForge subversion repository. All latest DDP's WS IIS artifacts are available from the ***Product/trunk/DDPs***.

2.5 Deployment View (i.e., Specification Release View)

The deployment view is also established by the mTOP SourceForge subversion repository. All DDP's WS IIS artifacts for a given product release are available from the ***Product/tags*** folder.

There is another form of deployment, which is based on a Web deployment of the mTOP DDP specifications (XML modules). The file structure is different from the SourceForge structure as it is more XML module oriented (SourceForge file structure is DDP oriented).

The file structure (address) of the Web deployment shall match the namespace format established for all mTOP XML modules (2.8.4).

2.6 XML Usage Standardization

These mTOP WS Guidelines use the following references to ensure maximum interoperability and integrity of the data produced by interfaces developed from the mTOP WS IIS.

The target is to have:

- *Full compliancy* with the ***Web Services Interoperability*** (WS-I) See Ref 5
- Partial compliancy with the **XML Schema Patterns for Databinding** See Ref 6

2.7 Naming Conventions

Refer to the mTOP naming convention guidelines. give the reference

In this document the following tags are used as variables:

- <DDP_Full> for the full DDP name
- <DDP> for the short DDP name (acronym)
- <ServiceInterface> for the name of an mTOP service interface
- <SI> for the acronym of an mTOP service interface
- <Operation> for the name of an mTOP service interface operation

2.8 Versioning and Extensions

2.8.1 Versioning Concept

Details of the versioning concept including major and minor version changes are captured in the mTOP Framework DDP SD 2-6 [**Error! Reference source not found.**].

Summary of the characteristics of an XML module specification changes captured in major and minor versions follows:

- Changes of the XML module specifications allowed in a minor version are (all changes that are backward compatible):
  - Do not affect the XML specifications footprint (interface & schema)
  - Only extend the XML specifications footprint (add children)
- All other changes of the XML module specifications are associated with a major version.
- Major and minor version numbers are two positive integers. The appropriate number is incremented by one based on the scope of the change (see above) introduced in the mTOP WS specifications in all subsequent mTOP WS IIS releases.
- Note that another marker may be used in future version artifacts to track the WS specifications pre-approved status (i.e “draft” or another number – pre-release issue)

2.8.2 Extensibility Concept

The idea of an mTOP WS specifications extension is about the capability to enhancing the specifications with vendor proprietary definitions.

This capability is also used for a minor version update of an XML module specification (See previous section).

2.8.3 XML Specification Design Requirements

The versioning requirements applicable to designing the mTOP WS specifications are:

1. The namespace of the mTOP XML module shall guarantee the uniqueness of all the specifications contained in the module.
2. The mTOP XML module specifications versioning shall not cover (and be impacted by) the specification from any vendor’s proprietary extensions.
3. Recommendation: All changes between two subsequent versions (major & minor) of an mTOP XML module specifications shall be fully described by XML artifacts (XSLT is the most appropriate artifact type to be used for an accurate description of these changes)..
4. Recommendation: A map of all mTOP XML module versions shall be maintained in association with specific migration guidelines.

Recommendations 3 & 4 would allow the definition of message transformation services, which would be deployed on the CCV to allow forward/backward compatibility between different versions of implemented and deployed mTOP service interfaces. See analysis and latest methodologies with respect to the use of SOA transformation services from references Ref 8.

The following design patterns shall be enhanced as they were applied to the MTOSI v1.x WS IIS:

- The full versioning (major & minor version numbers) of each mTOP service interface Web service description module specifications. The namespace must include major and minor version numbers. The differentiation of the Web service description modules for the same mTOP service interface between two minor versions is mandatory (e.g. a new optional operation is added to the service interface in a new minor versioned XML module specification).
- The use of minor versioning markers in XML documents (SOAP message payload & header) is improved to allow full traceability of changes and simple implementation of pre-processor filters. The design patterns of some XML Schema modules is re-visited accordingly (See XML Schema DG in section **Error! Reference source not found.**).

2.8.4 WS Specifications Design Guidelines

The design of the mTOP WS specifications and use of versioning artifacts are summarized as followed:

- The target namespace for all of the mTOP XML module local specifications must be versioned.
- All extended definitions shall be marked with the appropriate minor version.

Details/examples of the mTOP WS specification versioning artifacts are captured in each one of the guideline sections of this document; WSDL, XML Schema, and XML.

2.9 Namespace

The namespace of all the mTOP XML modules shall be based on the URI notation form as it offers the advantage of also allowing a Web deployment of the mTOP XML modules (See2.5).

Note that the notation used in MTOSI v1.x was mistakenly set as relative path (“tmf854.v1”). The intent was to use the URN notation. And, a proper form would had been “urn:tm854:v1”.

2.9.1 Format

The format used for all mTOP XML module namespaces is as followed:

www.tmforum.org/mTOP/<DDP>/<Type>/<Module>/<Version>

where,

- <DDP> is the acronym of an mTOP DDP
- <Type> is either ***xsd*** or ***wsdl***
- <Module> is the name (or equivalent acronym) of the module
- <Version> is the “v” character followed by the version number. Refer to each specific module type design guidelines for understanding of the version number used in the module specifications namespace.

Notes:

- Some additional markers may be used to indicate pre-approved version specifications (i.e. “<Major>-<Minor>draft”).
- Refer to the mTOP Naming Conventions for acronym/abbreviation details [**Error! Reference source not found.**].

2.9.2 Declaration

The namespace declarations of all definitions of an mTOP XML module shall follow these rules:

- The *targetnamespace* shall be the same as the namespace of the mTOP XML module
- All local definitions (targetnamespace) of an mTOP XML module shall use the “tns” namespace prefix.
- All imported definitions from other mTOP XML modules shall use a consistent namespace prefix in the entire mTOP WS IIS.

2.10 Documentation

The documentation of the mTOP WS IIS uses the 3rd party software WSDLDoc from www.bluetetra.com. This application generates HTML-based documentation (like Javadoc) of all WSDL and XML Schema specifications used for the definition of WS (wsdl:service definition).

Usage of this application for mTOP WS IIS is to generate the HTML (API) documentation for all mTOP service interfaces per DDP. This process can be automated as part of the mTOP loadbuild execution. Also, this is one way of verifying that the WS IIS is correct as the generation would fail for any syntax error found in the WS IIS.

The location for the generated HTML documentation is: ***<DDP>/html***.

An ***index.html*** file provides the home page for all the mTOP service interface WS IIS of the DDP.

3 WSDL Guidelines

3.1 WSDL Modules

3.1.1 Overview

The WS IIS structure of an mTOP service interface in terms of WSDL modules and WSDL definitions is as followed:

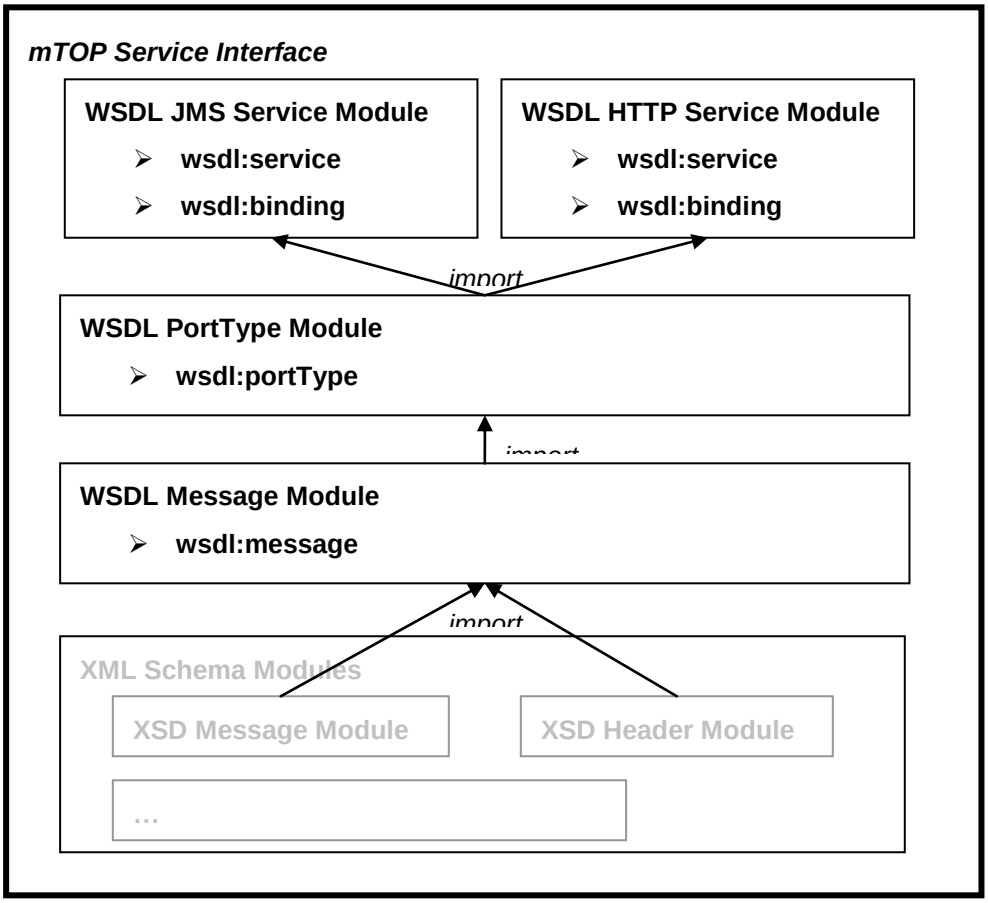

Figure 4 mTOP Service Interface (no iterator) WSDL Modules

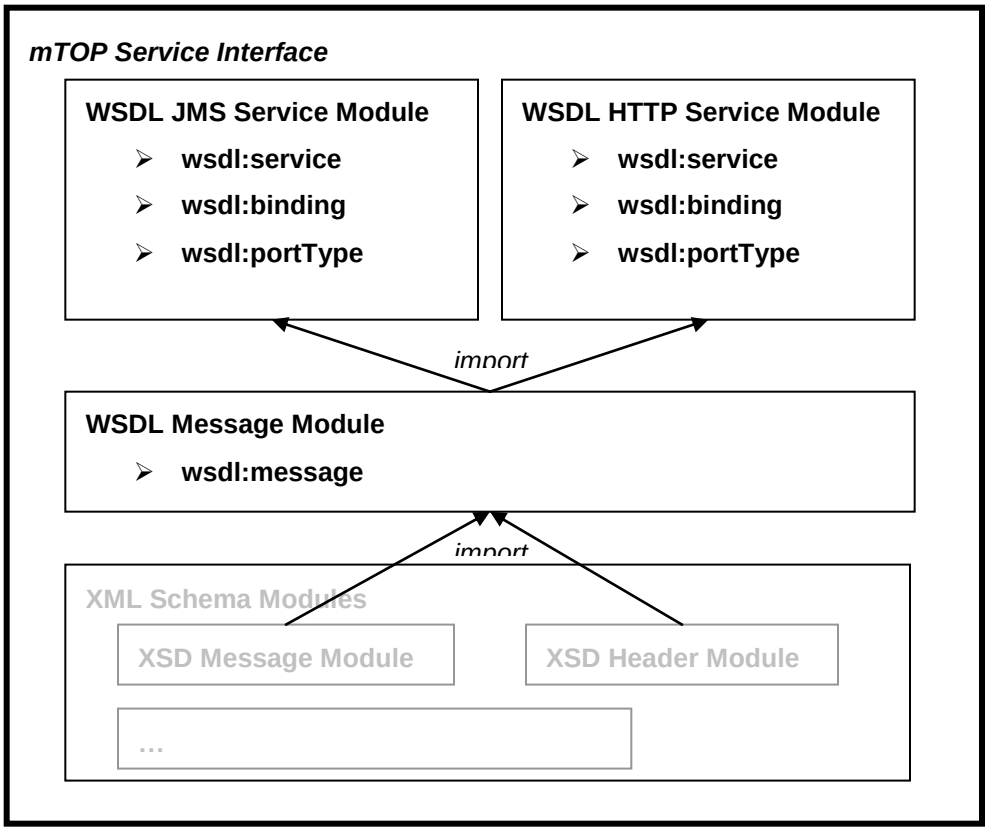

Figure 5 mTOP Service Interface (with iterator) WSDL Modules

The difference between the above two figures relies on the use of the data retrieval iterator. The iterator is necessary for the retrieval of batches of data with an implementation based on the RPC (synchronous) communication pattern. So, two portTypes are defined, one for each binding type (JMS and HTTP). And, they are integrated with the service module for each one of the specific supported transport. See details in section 3.1.3.

3.1.1.1 Design Considerations

- All mTOP service interface WSDL modules have the same targetnamespace
- Two mTOP service interface WSDL modules are offered based on the two supported service profiles:
  - HTTP service WSDL module is the true Web Service (SOAP/HTTP) representation of the mTOP service interface
  - JMS service WSDL module is a Web Service (SOAP/JMS) representation of the mTOP service interface XML messaging using the JMS communication API.
- The WSDL message module is common (wsdl:include) to the two WSDL service profile modules. Also, the WSDL message module use the XSD *import* of the definitions from XSD modules (header and the service interface messages schema)
- The WSDL portType module can also be common (wsdl:include) to the two WSDL service profile modules. It has to be differentiated with two distinct definitions (as seen in Figure 4) as soon as one of the operations is based on the batch retrieval communication style. Then, two WSDL

portType modules are necessary based on the communication pattern; RPC (with iterators) and MSG.

- Ideally, a single WSDL module would have been easier as many WS toolkits have issues with import/include notations. But, the separation into many WSDL modules is necessary to differentiate the two service profile offerings and for efficiency (re-use).
  The following is an example of an alternative solution based on a single WSDL module would include:
  - A WSDL service definition with name="MeRetrieval" and two ports:
    - MeRetrievalHttp based on MeRetrievalSoapHttpBinding
    - MeRetrievalJms based on MeRetrievalSoapJmsBinding
  - Two WSDL binding definitions
    - MeRetrievalSoapHttpBinding as SOAP/HTTP WSDL binding
    - MeRetrievalSoapJmsBinding as SOAP/JMS WSDL binding
  - Two WSDL portType definitions with name="MeRetrieval[Rpc|Msg]"
  - All the necessary WSDL message definitions for the two portType operations

This alternative solution was not accepted as it is blends the two deployment profiles (HTTP & JMS) into a single WSDL specification module.

3.1.1.2 Location

All mTOP service interface WSDL modules are found in: ***<DDP>/IIS/wsdl/<ServiceInterface>***

3.1.1.3 Namespace

All mTOP service interface WSDL modules are assigned the same targetnamespace:
http://www.tmforum.org/mTOP/<DDP>/wsdl/<SI>/<Version>

3.1.1.4 W3C standard specifications

The mTOP WSDL modules are based on:

- WSDL 1.1 [Ref 2]
- SOAP 1.1 [Ref 3]

3.1.1.5 Documentation

WSDL documentation elements shall be used for description of all relevant WSDL-based mTOP module definitions.

Use all examples captured in this section of the document for a case by case reference.

3.1.2 WSDL Message Module

Filename: ***<ServiceInterface>Messages.wsdl***

Content:

- This module includes all the WSDL *message* definitions of an mTOP service interface.
- ***wsdl:definitions*** name = "***mTOP-<ServiceInterface><Version>***" It is the same as for other mTOP service interface WSDL modules
- Refer to section 3.2.1 for details of the ***wsdl:message*** usage.
- This module imports the mTOP service interface XSD message module.
- This module is imported by the mTOP service interface WSDL portType module.

Example:

The following example is extracted the ***MeRetrievalMessages.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- mTOP DDP - OM ManageResourceInventory - Copyright TeleManagement Forum 2007 -->
<wsdl:definitions name="mTOP-ManagedElementRetrievalV1-0"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:mer="http://www.tmforum.org/mtop/mri/xsd/mer/v1"
xmlns:hdr="http://www.tmforum.org/mtop/fmw/xsd/hdr/v1"
xmlns:tns="http://www.tmforum.org/mtop/mri/wsdl/mer/v1-0"
targetNamespace="http://www.tmforum.org/mtop/mri/wsdl/mer/v1-0">
 <wsdl:documentation>
  <p>DDP: Manage Resource Inventory</p>
  <p>Module: ManagedElementRetrieval Service Interface WSDL Message Module</p>
  <p>Description: This module contains all the mTOP ManagedElementRetrieval Service Interface WSDL
message definitions.</p>
 </wsdl:documentation>
 <wsdl:types>
  <xsd:schema>
   <xsd:import namespace="http://www.tmforum.org/mtop/mri/xsd/mer/v1"
schemaLocation="../../xsd/MeRetrievalMessages.xsd"/>
  </xsd:schema>
 </wsdl:types>
  … All wsdl:message definitions here
</wsdl:definitions>
```

Table 1 WSDL Message Module Example

3.1.3 WSDL PortType Module

Filename: ***<ServiceInterface>PortType?.wsdl***

Content:

- This module includes all the WSDL *portType* definitions of an mTOP service interface.
- ***wsdl:definitions*** name = "***mTOP-<ServiceInterface><Version>***" It is the same as for other mTOP service interface WSDL modules
- For a given mTOP service interface a single WSDL portType module is usually necessary. However, two WSDL portType definitions are required if an iterator is used. In which case, the PortType module is no longer necessary as a specific portType definitions can be integrated with the HTTP and JMS module. For example, a WSDL PortType module exists for the OsRetrieval service interface. But, there is no WSDL PortType module for the MeRetrieval service interface as it uses an iterator.
- Refer to section 3.2.3 for details of the wsdl:portType usage.
- This module imports the mTOP service interface WSDL message module.
- This module is imported by the mTOP service interface WSDL HTTP & JMS modules

Example:

The following example is extracted the ***OsRetrievalPortType.wsdl*** module found in the ***IIS/wsdl/OsRetrieval*** folder of the ***Framework*** DDP

An example where we precisely have two PortTypes modules would be more illustrative (e.g. based on the "ResourceInventoryRetrieval" when it is done)

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- mTOP DDP - OM ManageResourceInventory - Copyright TeleManagement Forum 2007 -->
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://www.tmforum.org/mtop/mri/ws/osr/v1"
name="mTOP-OsRetrieval-HTTP" targetNamespace="http://www.tmforum.org/mtop/mri/ws/osr/v1">
  <wsdl:documentation>
    <p>DDP: Manage Resource Inventory</p>
    <p>Module: OsRetrieval Service Interface WSDL PortType Module</p>
    <p>Description: This module contains the mTOP OperationsSystemRetrieval Service Interface PortType
WSDL definitions.</p>
    <p>Version: 1.0</p>
  </wsdl:documentation>
  <!-- ================ -->
  <wsdl:import namespace="http://www.tmforum.org/mtop/mri/ws/os/v1"
location="OsRetrieval_Messages.wsdl"/>
```

```
<!-- ================ -->
<wsdl:portType name="OsRetrieval">
  <wsdl:documentation>
    <p>The OsRetrieval porttype (interface).</p>
  </wsdl:documentation>
  … All wsdl:portType definitions here
</wsdl:definitions>
```

Table 2 WSDL PortType Module Example

3.1.4 WSDL HTTP Module

Filename: ***<ServiceInterface>Http.wsdl***

Content:

- This module includes all the WSDL definitions used for the description of the mTOP service interface when implemented and deployed as a true Web Service (SOAP/HTTP).
- ***wsdl:definitions*** name = "***mTOP-<ServiceInterface><Version>***" It is the same as for other mTOP service interface WSDL modules
- The WSDL HTTP service module includes the following WSDL definitions:
  - WSDL *binding* definitions. Refer to section 3.2.5 for details of the wsdl:binding usage
  - WSDL *service* definitions. Refer to section 3.2.6 for details of the wsdl:service usage
  - WSDL portType definitions if an iterator is required for batch retrieval. Refer to section 3.2.4 for details of the wsdl:portType usage.
- Refer to the Framework SD 2-16 for all the SOAP/HTTP binding implementation details.

Example:

The following example is extracted the ***MeRetrievalHttp.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- mTOP DDP - OM ManageResourceInventory - Copyright TeleManagement Forum 2007 -->
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://www.tmforum.org/mtop/mri/ws/mer/v1" name="mTOP-MeRetrieval-HTTP"
targetNamespace="http://www.tmforum.org/mtop/mri/ws/mer/v1">
  <wsdl:documentation>
    <p>DDP: Manage Resource Inventory</p>
    <p>Module: ManagedElementRetrieval Service Interface WSDL HTTP Module</p>
    <p>Description: This module contains the mTOP ManagedElementRetrieval Service Interface WSDL
```

```
definitions for the Web Service deployment profile with SOAP/HTTP binding and portType.</p>
   <p>Version: 1.0</p>
  </wsdl:documentation>
  <!-- ================ -->
  <wsdl:import namespace="http://www.tmforum.org/mtop/mri/ws/me/v1"
location="MeRetrieval_Messages.wsdl"/>
   … All wsdl:binding ,wsdl:service, and optional wsdl:portType (if iterator is used) definitions are here
</wsdl:definitions>
```

Table 3 WSDL HTTP Module Example

3.1.5 WSDL JMS Module

Filename: ***<ServiceInterface>Jms.wsdl***

Content:

- This module includes all the WSDL definitions used for the description of the mTOP service interface when implemented and deployed with the JMS API.
- ***wsdl:definitions*** name = "***mTOP-<ServiceInterface><Version>***" It is the same as for other mTOP service interface WSDL modules
- The WSDL JMS service module includes the following WSDL definitions:
  - WSDL *binding* definitions. Refer to section 3.2.5 for details of the wsdl:binding usage
  - WSDL *service* definitions. Refer to section 3.2.6 for details of the wsdl:service usage
  - WSDL portType definitions if an iterator is required for batch retrieval. Refer to section 3.2.4 for details of the wsdl:portType usage.

Refer to the Framework SD 2-9 for all the SOAP/JMS binding implementation details.

Example:

The following example is extracted the ***MeRetrievalJms.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- mTOP DDP - OM ManageResourceInventory - Copyright TeleManagement Forum 2006 -->
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://www.tmforum.org/mtop/mri/ws/me/v1"
name="mTOP-MeRetrieval-JMS" targetNamespace="http://www.tmforum.org/mtop/mri/ws/me/v1">
  <wsdl:documentation>
   <p>DDP: Operation Model Manage Resource Inventory</p>
   <p>Module: ManagedElementRetrieval Service Interface WSDL Module</p>
   <p>Description: This module contains the mTOP ManagedElementRetrieval Service Interface WSDL
```

```
definitions for the SOAP/JMS binding based on an MSG style port type (interface).</p>
   <p>Version: 1.0</p>
  </wsdl:documentation>
  <!-- ================ -->
  <wsdl:import namespace="http://www.tmforum.org/mtop/mri/ws/me/v1"
location="MeRetrieval_Messages.wsdl"/>
   … All wsdl:binding ,wsdl:service, and optional wsdl:portType (if iterator is used) definitions are here
</wsdl:definitions>
```

Table 4 WSDL JMS Module Example

3.2 WSDL Definitions Usage

3.2.1 Message

The WSDL message definitions of the mTOP service interface are based on the Message Exchange Pattern (MEP) of the operation they are designed for, which is specified by the WSDL portType definitions (See 3.2.4).

3.2.2 Request-Response Operation Messages

Most of the mTOP service interface operations are associated with the WSDL Request-Response operation MEP definitions. The messages are:

- Request (In) message:
  - Name is <Operation>Request
  - Part#1: "***mtopHeader***" based on the XSD header module (in the Framework DDP)
  - Part#2: "***mtopBody***" based on the XML Schema request message (from the mTOP service interface XSD messages module)
- Response (Out) message :
  - Name is <Operation>Response
  - Part#1: "***mtopHeader***" based on the XML Schema header module (in the Framework DDP)
  - Part#2: "***mtopBody***" based on the XML Schema response message (from the mTOP service interface XSD messages module)
- Exception (Fault) message:
  - Name <Operation>Exception
  - Part: "***mtopBody***" based on the XML Schema exception message (from the mTOP service interface XSD messages module)
  - Note that there is no header part. This is because SOAP v1.1 fault message does not have a SOAP header. The MTOSI header is there fore encapsulate inside the MTOSI exception message payload.
  - The reason to supporting a single generic WSDL fault message per operation is to simplify the description of the operation. The details of the operation exception are prescribed at the XML Schema level. This is to ensure simplicity for the possible evolution of the operation.

But, it is required for the SOAP/JMS implementation profile, which may not rely on the WSDL specifications.

Examples:

The following example is extracted the ***MeRetrievalMessages.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<wsdl:message name="getAllManagedElementsRequest">
  <wsdl:documentation>
    <p>The getAllManagedElements request message.</p>
  </wsdl:documentation>
  <wsdl:part name="mtopHeader" element="hdr:header"/>
  <wsdl:part name="mtopBody" element="mer:getAllManagedElementsRequest"/>
</wsdl:message>
<wsdl:message name="getAllManagedElementsResponse">
  <wsdl:documentation>
    <p>The getAllManagedElements response message.</p>
  </wsdl:documentation>
  <wsdl:part name="mtopHeader" element="hdr:header"/>
  <wsdl:part name="mtopBody" element="mer:getAllManagedElementsResponse"/>
</wsdl:message>
<wsdl:message name="getAllManagedElementsException">
  <wsdl:documentation>
    <p>The getAllManagedElements exception message.</p>
  </wsdl:documentation>
  <wsdl:part name="mtopBody" element="mer:getAllManagedElementsException"/>
</wsdl:message>
```

Table 5 WSDL Request-Response Message Definition Examples

3.2.3 One-Way Operation Messages

A single message is defined for each one of the mTOP service interface operations that are identified in the WSDL portType module based on the WSDL One-Way operation MEP:

- Message:
  - Name is <Operation>
  - Part#1: mtopHeader based on the XSD header module (in the Framework DDP)

- Part#2: mtopBody based on the XML Schema request message (from the mTOP service interface XSD messages module)

Examples:

The following example is extracted the ***NotificationBrokerMessages.wsdl*** module found in the ***IIS/wsdl/NotificationBroker*** folder of the ***Framework*** DDP

```
<wsdl:message name="notify">
  <wsdl:documentation>
    <p>The notify message.</p>
  </wsdl:documentation>
  <wsdl:part name="mtopHeader" element="hdr:header"/>
  <wsdl:part name="mtopBody" element="not:notify"/>
</wsdl:message>
```

Table 6 WSDL One-Way Message Definition Examples

3.2.4 Port Type

3.2.4.1 Description

The WSDL portType definitions include the description of all the operations of an mTOP service interface.

3.2.4.2 RPC vs. MSG

In most cases, a common set of WSDL portType definitions is enough for both implementation and deployment service profiles (HTTP and JMS). These mTOP service interface WSDL portType definitions are all captured in a single WSDL portType module (See 3.1.3).

However, in the case when some operations of the mTOP service interface are based on the Batch Response communication pattern (See **Error! Reference source not found.**) a differentiation of the portType definitions must be established. Then, two sets of portType definitions are necessary:

- RPC with iterator operations
- MSG without iterator operations

The RPC style portType definitions are associated (imported by) with the WSDL HTTP service module. And, the MSG style portType definitions are associated (imported by) with the WSDL JMS service module.

3.2.4.3 Iterator operation

An iterator operation is available by the mTOP service interface for each datatype that can be retrieved using operations based on Batch Response communication pattern. Synopsis of this *iterator* operation follows:

get<DataType>IteratorResponse get<DataType>IteratorRequest(iteratorID)

get<DataType>IteratorException

Where,

- <DataType> is the unit of data retrieved (e.g. ManagedElement object)
- *get<DataType>IteratorResponse* is the response with the next batch of data
- *get<DataType>IteratorRequest* is the request message with
  - *iteratorID* as the input parameter. It is unique for each iterator instance, which was started on successful invocation of the first retrieval operation (e.g. getAllManagedElements). *iteratorID* is handed as a header parameter.
- get<DataType>IteratorException is the exception message

Examples:

The following operations exist in the WSDL portType definitions of the WSDL HTTP module; MeRetrievalHttp.wsdl. They do not exist in the WSDL portType definitions of the WSDL JMS module MeRetrievalJms.wsdl

- getMeIterator
- getMeNamesIterator

3.2.4.4 Naming

The name of the WSDL portType is: ***<ServiceInterface>?<ComPattern>***

Where <ComPattern> is an option with value either:

- ***Rpc*** for synchronous communication pattern
- ***Msg*** for asynchronous communication pattern

All mTOP service interface operations are based on the following two WSDL/SOAP operation MEPs with the following details:

- Request-Response operation
  - A WSDL input definition associated with the operation request WSDL message definition
  - A WSDL output definition associated with the operation response WSDL message definition
  - A WSDL fault definition with:
    - Name = "<OperationName>Exception"
    - Message = "***<OperationName>Exception***" The operation exception WSDL message definition
- One-Way operation
  - A WSDL input definition associated with the operation request WSDL message definition

3.2.4.5 Notes

- The mTOP service interface operations based on the Multiple Batch Response communication pattern (See **Error! Reference source not found.**) and implemented with the asynchronous communication style are defined using the WSDL Request-Response MEP even though multiple responses are expected. This is a limitation of the WSDL MEP, which may be resolved in future version of the WSDL specifications (See Ref 2). Header parameters shall be used (runtime) to describe the mTOP service interface operation Multiple Batch Response MEP (**Error! Reference source not found.**).
- The mTOP service interface operations based on the Bulk Response (File transfer via another communication channel) are also defined using the WSDL Request-Response MEP. One can see the following MEP description limitations:
  - The protocol and details of the file transfer are not part of the WSDL MEP. Header parameters shall be used (runtime) to describe the mTOP service interface operation Bulk Response MEP (**Error! Reference source not found.**), and
  - The description of the notification reporting capability regarding the status of the operation execution is also not described in the WSDL MEP. The notification of the FileTransferStatus event is an option available in the mTOP WS IIS (Framework DDP).

3.2.4.6 Examples

The following example is extracted the ***MeRetrievalHttp.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<wsdl:portType name="MeRetrieval_RPC">
  <wsdl:documentation>
    <p>The ManagedElementRetrieval porttype (interface) for RPC (synchronous) based client/server communication patterns. All SIT operations use the getMeIterator operation to handle multiple responses.</p>
  </wsdl:documentation>
  <!-- ================ -->
  <wsdl:operation name="getAllManagedElementNames">
    <wsdl:documentation>
      <p>This SIT operation has exactly the same behaviour as getAllManagedElements(), but instead of returning the entire object structures, this operation returns their names.</p>
    </wsdl:documentation>
    <wsdl:input message="tns:getAllManagedElementNames"/>
    <wsdl:output message="tns:getAllManagedElementNamesResponse"/>
    <wsdl:fault name="getAllManagedElementNamesException" message="tns:getAllManagedElementNamesException"/>
  </wsdl:operation>
  <!-- ================ -->
  <wsdl:operation name="getAllManagedElements">
```

```
    <wsdl:documentation>
      <p>This SIT operation allows an NMS to request a list of all the known ManagedElements in the
given MD.</p>
    </wsdl:documentation>
    <wsdl:input message="tns:getAllManagedElements"/>
    <wsdl:output message="tns:getAllManagedElementsResponse"/>
    <wsdl:fault name="getAllManagedElementsException"
message="tns:getAllManagedElementsException"/>
  </wsdl:operation>
  <!-- ================ -->
  …
</wsdl:portType>
```

Table 7 WSDL PortType Definition Example

3.2.5 Binding

The WSDL binding definitions of each mTOP service interface include the mapping of the WSDL portType operation definitions to the following two deployment profiles:

- SOAP/HTTP
  - This WSDL binding describe a true Web Service deployment of the mTOP service interface. WSDL binding definition details include:
    - Name = "<ServiceInterface>SoapHttpBinding"
    - Type = Reference to the WSDL portType definition (See 3.2.4)
    - The SOAP binding definition includes:
      - Style = "***document***"
      - Transport = "http://schemas.xmlsoap.org/soap/http"
    - See below for the WSDL operation definitions
- SOAP/JMS
  - This WSDL binding describe a deployment of the mTOP service interface using the JMS API as the message Common Communication Vehicle (CCV).
  - Note that this binding is not directly supported by the WSDL specifications. The URI value of the transport attribute (see below) is not a valid document (See Ref 2)
  - WSDL binding definition details include:
    - name = "<ServiceInterface>SoapHttpBinding"
    - type = Reference to the WSDL portType definition (See 3.2.4)
    - The SOAP binding definition includes:
      - style = "document"
      - transport = "http ://schemas.xmlsoap.org/soap/jms"

- See below for the WSDL operation definitions

- WSDL operation definitions:
  - soap:operation definition
    - Only for the HTTP transport, soapAction = "***<OperationName>***"
    - style = "***document***"
  - One-Way WSDL operation
    - ***wsdl:input*** definition
      - ***soap:header*** (assign the mTOP header part):
        - message = "***<OperationName>***" Reference to the operation request WSDL message
        - part = "***mtopHeader***" reference to the part defined in the operation request WSDL message
        - use = "***litteral***" (literal has a single "t") (no soap encoding of the message payload)
    - ***soap:body*** (assign the mTOP operation message payload):
      - part = "***mtopBody***" reference to the part defined in the operation request WSDL message
      - use = "***litteral***" (no soap encoding of the message payload)
  - Request-Response WSDL operation
    - ***wsdl:input*** definition
      - ***soap:header*** (assign the mTOP header part):
        - message = "***<OperationName>Request***" Reference to the operation request WSDL message
        - part = "***mtopHeader***" reference to the part defined in the operation request WSDL message
        - use = "***litteral***" (no soap encoding of the message payload)
      - ***soap:body*** (assign the mTOP operation message payload):
        - part = "***mtopBody***" reference to the part defined in the operation request WSDL message
        - use = "***litteral***" (no soap encoding of the message payload)
    - ***wsdl:output*** definition
      - ***soap:header*** (assign the mTOP header part):
        - message = "***<OperationName>Response***" Reference to the operation response WSDL message
        - part = "***mtopHeader***" reference to the part defined in the operation response WSDL message
        - use = "***litteral***" (no soap encoding of the message payload)
      - ***soap:body*** (assign the mTOP operation message payload):

- part = "***mtopBody***" reference to the part defined in the operation response WSDL message
- use = "***litteral***" (no soap encoding of the message payload)

- ***wsdl:fault*** definition
  - ***soap:fault***:
    - name = "***<OperationName>Exception***"
    - use = "***litteral***" (no soap encoding of the message payload)

Examples:

The following example is extracted the ***MeRetrievalHttp.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<wsdl:binding name="MeRetrievalSoapHttpBinding" type="tns:MeRetrieval_RPC">
  <wsdl:documentation>
    <p>SOAP HTTP binding definitions of the operations of the MeRetrieval RPC portType (interface)</p>
  </wsdl:documentation>
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <!-- ================ -->
  <wsdl:operation name="getAllManagedElementNames">
    <soap:operation soapAction="getAllManagedElementNames" style="document"/>
    <wsdl:input>
      <soap:header message="tns:getAllManagedElementNames" part="mtopHeader" use="literal"/>
      <soap:body parts="mtopBody" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:header message="tns:getAllManagedElementNamesResponse" part="mtopHeader" use="literal"/>
      <soap:body parts="mtopBody" use="literal"/>
    </wsdl:output>
    <wsdl:fault name="getAllManagedElementNamesException">
      <soap:fault name="getAllManagedElementNamesException" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  …
</wsdl:binding>
```

Table 8 WSDL HTTP Binding Definition Example

The following example is extracted the ***MeRetrievalJms.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<wsdl:binding name="MeRetrievalSoapJmsBinding" type="tns:MeRetrievalMsg">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/jms"/>
  <wsdl:documentation>
    <p>SOAP JMS binding definitions of the operations of the MeRetrieval_MSG portType
(interface)</p>
  </wsdl:documentation>
  <!-- ================ -->
  <wsdl:operation name="getAllManagedElementNames">
    <soap:operation style="document"/>
    <wsdl:input>
      <soap:header message="tns:getAllManagedElementNames" part="mtopHeader" use="literal"/>
      <soap:body parts="mtopBody" use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:header message="tns:getAllManagedElementNamesResponse" part="mtopHeader"
use="literal"/>
      <soap:body parts="mtopBody" use="literal"/>
    </wsdl:output>
    <wsdl:fault name="getAllManagedElementNamesException">
      <soap:fault name="getAllManagedElementNamesException" use="literal"/>
    </wsdl:fault>
  </wsdl:operation>
  <!-- ================ -->
  …
</wsdl:binding>
```

Table 9 WSDL JMS Binding Definition Example

3.2.6 Service

The WSDL service definitions include the deployment description of a given mTOP service interface to one of the two specific supported transport; HTTP or JMS.

Note that the WSDL service definitions are not necessary in these mTOP WS IIS as they usually are generated at the WS deployment time.

Details of the WSDL service definitions based on the deployment profile transport:

- SOAP/HTTP
  - ***wsdl:service*** definitions
    - name = "<ServiceInterface>Http"
    - wsdl:port
      - name = "***<ServiceInterface>SoapHttp***
      - binding = "***<ServiceInterface>SoapHttpBinding***" reference to the WSDL binding definitions (See 3.2.5)
      - soap:address = ***"http://aserver/mtosi/<ServiceInterface> <Version>"***
- SOAP/JMS
  - ***wsdl:service*** definitions
    - name = "<ServiceInterface>Jms"
    - wsdl:port
      - name = "***<ServiceInterface>SoapJms***
      - binding = "***<ServiceInterface>SoapJmsBinding***" reference to the WSDL binding definitions (See 3.2.5)
      - soap:address = "***jms://aserver/mtosi/<ServiceInterface> <Version>***"

Examples:

The following example is extracted the ***MeRetrievalHttp.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<wsdl:service name="MeRetrievalHttp">
  <wsdl:documentation>
    <p>Example of the MeRetrieval WSDL Service definition as a deployment example of this mTOP Service Interface. It is expected to be modified to match the specific deployment configuration in a service provider's CCV.</p>
  </wsdl:documentation>
  <wsdl:port name="MeRetrievalSoapHttp" binding="tns:MeRetrievalSoapHttpBinding">
    <soap:address location="http://aserver/mtosi/MeRetrieval"/>
  </wsdl:port>
</wsdl:service>
```

Table 10 WSDL HTTP Service Definition Example

The following example is extracted the ***MeRetrievalJms.wsdl*** module found in the ***IIS/wsdl/MeRetrieval*** folder of the ***ManageResourceInventory*** DDP

```
<wsdl:service name="MeRetrievalJms">
  <wsdl:port name="MeRetrievalSoapJms" binding="tns:MeRetrievalSoapJmsBinding">
    <wsdl:documentation>
      <p>Example of the MeRetrieval WSDL Service definition as a deployment example of this mTOP Service Interface. It is expected to be modified to match the specific deployment configuration in a service provider's CCV.</p>
    </wsdl:documentation>
    <soap:address location="jms://aserver/mtosi/MeRetrieval"/>
  </wsdl:port>
</wsdl:service>
```

Table 11 WSDL JMS Service Definition Example

3.3 Versioning and Change Policy

The important versioning aspect of the mTOP service interface WSDL specifications are:

- The name of the wsdl:definitions in all mTOP service interface WSDL modules (See 3.1)
- The same namespace is assigned to all these mTOP service interface wsdl:definitions (See 3.1.1.3).

In all of the above, the version is part of the mTOP service interface WS specification name. Note that these definitions do not have runtime constrains. The namespace of the mTOP service interface XML Schema messages establishes the necessary version control.

3.3.1 Major Version Changes

The following mTOP WSDL service interface specification changes are carried out as a new major version of the modules:

- Major version change of the specifications in the XSD message module (Refer to 4.1.2.4)
- An operation is deprecated from the mTOP service interface

3.3.2 Minor Version Changes

The following mTOP WSDL service interface specification changes are carried out as a new minor version of the modules:

- At least one new optional operation is added to the mTOP service interface
- Note that minor changes of the specifications in the XSD message module does not affect the WSDL service interface module

3.3.3 Versioning Identifiers

Versioning identification of the mTOP WSDL specification modules includes (see details in Table 12):

- Name of the WSDL definitions (e.g. name="mTOP-NotificationBrokerV1-0")
- Last chapter of the WSDL definitions documentation element (e.g. Version: 1.0)
- The full version (major & minor numbers) is part of the targetnamespace (e.g. xmlns:tns="http://www.tmforum.org/mtop/fmw/wsdl/notb/v1-0")

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

<!-- mTOP DDP - Framework - Copyright TeleManagement Forum 2007 -->

<wsdl:definitions name="mTOP-NotificationBroker V1-0"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://www.tmforum.org/mtop/fmw/wsdl/notb/v1-0"
targetNamespace="http://www.tmforum.org/mtop/fmw/wsdl/notb/v1-0">

  <wsdl:documentation>

    <p>DDP: Framework</p>

    <p>Module: NotificationBroker Service Interface WSDL JMS Module</p>

    <p>Description: This module contains all the mTOP NotificationBroker Service Interface WSDL
SOAP/JMS binding definitions.</p>

    <p>Version: 1.0</p>

  </wsdl:documentation>

  <!-- ================ -->

  <wsdl:import location="NotificationBroker_PortType.wsdl"/>

   … All wsdl:binding and wsdl:service definitions are here

</wsdl:definitions>
```

Table 12 WSDL Module Versioning

4 XML Schema Guidelines

4.1 XML Schema Modules

4.1.1 Overview

In the mTOP DDP WS IIS, the XML Schema (XSD) modules define the mTOP schema specifications of the following XML parts that are captured in the SOAP message (envelope) (See WSDL binding definitions section 3.2.5):

- The mTOP header, which is captured in the SOAP Header,
- The mTOP operation message (payload), which is captured in the SOAP Body, and
- The mTOP operation exception, which is captured in the SOAP Fault (itself found in the SOAP Body)

4.1.1.1 Naming Conventions

The following rules are naming conventions applicable to all XML Schema definitions:

- Data types: DataNameType
- Group: DataNameGroup
- Element: dataName
- Sequence as list of one parameter (array): DataNameListType (type) or dataNameList (element)

4.1.1.2 Design Considerations

- Each mTOP XSD module has a distinct namespace. This is to allow a maximum flexibility in handling the evolution of the mTOP entities and their specifications defined in the scope of the mTOP XSD module.
- The scope of the schema specifications defined in an XSD module is limited to a logical grouping. The following XSD modules and their distinct patterns are described in these mTOP XSD specifications guidelines:
  - mTOP XSD *message* module for the schema definitions of all operation messages of an mTOP service interface
  - mTOP XSD *aggregate* module for the schema definitions of an mTOP object, event, or any other aggregate data structure
  - mTOP XSD *dataset* module for the schema definitions of a set of mTOP data types (collection of parameters)
- General XSD design guidelines for the mTOP XSD module definitions:
  - Loose definitions to ease WS IIS evolution (backward/forward compatibility)
  - Right balance between:
    - Re-use by choosing to define global data types, and otherwise
    - Lightweight IIS by defining local data types

4.1.1.3 Location

All mTOP XSD modules are found in: ***<DDP>/IIS/xsd***

4.1.1.4 Namespace

All XML Schema definitions of an mTOP XSD module are assigned a unique namespace (targetnamespace), which is based on the following format:
http://www.tmforum.org/mTOP/<DDP>/xsd/<Module>/<MajorVersion>

Where,

- <DDP> is the DDP where the XSD module is found
- <Module> is the distinct XSD module abbreviation/marker
- <MajorVersion> is the major version associated with the XSD module (See 3.3)

4.1.1.5 W3C standard specifications

All the mTOP XSD modules are based on:

- XML Schema 1.0 [Ref 4]

4.1.1.6 Version Identifiers

Versioning identification of the mTOP XSD specification modules includes (see details in Table 13):

- The major version is part of the targetnamespace (e.g. targetNamespace="http://www.tmforum.org/mtop/fmw/xsd/notmsg/**v1**")
- The version attribute of the XSD Schema root element is set with the full version (e.g. **version="1.0"**)

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- mTOP DDP - Framework - Copyright TeleManagement Forum 2007 -->
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.tmforum.org/mtop/fmw/xsd/notmsg/v1"
xmlns:tns="http://www.tmforum.org/mtop/fmw/xsd/notmsg/v1"
xmlns:msg="http://www.tmforum.org/mtop/fmw/xsd/msg/v1"
xmlns:cei="http://www.tmforum.org/mtop/fmw/xsd/cei/v1" attributeFormDefault="unqualified"
elementFormDefault="qualified" version="1.0">
…
</xsd:schema>
```

Table 13 XSD Module Versioning

4.1.1.7 Documentation

XSD annotation and documentation elements shall be used for description of all relevant mTOP XML Schema definitions.

4.1.2 XSD Message Module

4.1.2.1 Purpose

For each mTOP service interface (defined in an OM DDP), there is an XSD message module.

The XSD message module is directly imported in the WSDL message module (See 3.1.2) for the mTOP service interface WSDL message definitions (See 3.2.1).

The XSD message module contains all the XML Schema definitions of the mTOP service interface operation messages.

4.1.2.2 Module Specifications

The format of the XSD message module name is: ***<ServiceInterface>Messages.xsd***

The namespace of the XSD message module is:
http://www.tmforum.org/mTOP/<DDP>/xsd/<SI>/<MajorVersion> (See 4.1.1.4)

The XSD module marker is based on the acronym/abbreviation of the mTOP service interface to which these XML Schema message definitions are associated.

4.1.2.3 XSD Definitions

The XML Schema definitions found in the XSD message module include an element definition for each one of the messages that are described in the WSDL message module (See 3.1.2).

4.1.2.3.1 Request Message

The XML Schema definitions of an mTOP operation request message include:

- Element declaration with name = “<operationName>Request”
- Complex type with a sequence of all the operation input parameters
- Order of the input parameters is given by the IA operation definitions
- All optional input parameters (based on IA definitions) are associated the minOccurs=”0” attribute

The examples below are a bit weak or uncomplete (for instance there is no examples of input parameters); better use another illustration

```
<xsd:element name="getAllManagedElements">
  <xsd:annotation>
    <xsd:documentation>
      <p>Request message structure of the getAllManagedElements operation</p>
      <p>This operation allows an NMS to request a list of all the known ManagedElements in the given MD.</p>
```

```
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="mdName" type="nam:NamingAttributesType">
        <xsd:annotation>
          <xsd:documentation>
            <p>the name of the Management Domain whose contained MEs are being requested</p>
          </xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

Table 14 mTOP operation request definitions in XSD message module

4.1.2.3.2 Response Message

The XML Schema definitions of an mTOP operation response message include:

- Element declaration with name = “<operationName>Response”
- ComplexType with a sequence of all the operation output parameters
- Order of the output parameters is given by the IA operation definitions
- All optional output parameters (based on IA definitions) are associated the minOccurs=”0” attribute

```
<xsd:element name="getAllManagedElementsResponse">
  <xsd:annotation>
    <xsd:documentation>
      <p>Response message structure of the getAllManagedElements operation</p>
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="meList" type="me:ManagedElementListType" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation>
          <p>The list of MEs</p>
        </xsd:documentation>
```

```
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:element>
```

Table 15 mTOP operation response definitions in XSD message module

Note that a local type may be re-used to define the same response message structure of many retrieval operations. See example below extracted from the MeRetrieval service interface.

```
<xsd:element name="getAllManagedElementsResponse" type="tns:MultipleMeObjectsResponseType">
  <xsd:annotation>
    <xsd:documentation>
      <p>Response message structure of the getAllManagedElements operation</p>
    </xsd:documentation>
  </xsd:annotation>
</xsd:element>
<!-- ================== -->
…
<!-- ================== -->
<xsd:complexType name="MultipleMeObjectsResponseType">
  <xsd:annotation>
    <xsd:documentation>
      <p>Response message structure for a set of ManagedElement objects</p>
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="meList" type="me:ManagedElementListType" minOccurs="0">
      <xsd:annotation>
        <xsd:documentation>
          <p>The list of MEs</p>
        </xsd:documentation>
      </xsd:annotation>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

Table 16 Optimized mTOP operation response definitions in XSD message module

4.1.2.3.3 Exception Message

The XML Schema definitions of an mTOP operation exception message include:

- Element declaration with name = “<operationName>Exception”
- Complex type with a choice between elements representing all the operation exception types
- Each exception type element is based on the BaseExceptionMessageType data type from the XSD aggregate module; Framework:MessageDefinitions.xsd
  This

```
<xsd:element name="getAllManagedElementsException">
 <xsd:annotation>
  <xsd:documentation>
   <p>Exception message structure of the getAllManagedElements operation</p>
  </xsd:documentation>
 </xsd:annotation>
 <xsd:complexType>
  <xsd:choice>
   <xsd:element name="exceptionInternalError" type="msg:BaseExceptionMessageType"/>
   <xsd:element name="exceptionTooManyOpenIterator" type="msg:BaseExceptionMessageType"/>
  </xsd:choice>
 </xsd:complexType>
</xsd:element>
```

Table 17 mTOP operation exception definitions in XSD message module

4.1.2.4 Versioning and Change Policy

4.1.2.4.1 Major Version Changes

The following mTOP XSD message specification changes are carried out as a new major version of the module:

- deprecated parameter in at least one message definition,
- modified parameter in at least one message definition,
- new mandatory parameter in at least one message definition

4.1.2.4.2 Minor Version Changes

The following mTOP XSD message specification changes are carried out as a new minor version of the module:

- One or many new optional parameters have been added to one or many messages
- One or many new messages for new optional operations are added

4.1.3 XSD Aggregate Module

4.1.3.1 Purpose

An mTOP XSD aggregate module contains the XML Schema definitions of one mTOP entity. The XSD definitions of this mTOP entity consist in a data type based on a *complexType* with the *sequence* of all the attributes that compose the entity data structure as defined in the IA.

Note that the scope of the XSD aggregate module is usually limited to the XSD definitions of a single mTOP entity. This fragmentation of the mTOP XSD definitions is important to simplify the individual evolution of the mTOP entity in future release of the specifications.

Some exceptions exist in some XSD aggregate modules, where XSD definitions of several mTOP entities may be defined. But, these mTOP entities shall have the same purpose (i.e. the XSD aggregate module; *Framework:MessageDefinitions.xsd* has XSD definitions of many complex mTOP data types, which are all related as they define the basic/core mTOP message data structures).

Finally, the XSD aggregate module may also contain collateral XSD definitions designed for the data management of the specified mTOP entity. For instance, the XSD definition representing the list/array of the mTOP entity is usually also found in the same XSD aggregate module (i.e. the *MeListType* XSD definition as a *sequence* of *ManagedElementType* elements is also found in the *NetworkResourceFulfillment:Me.xsd* module).

4.1.3.2 Module Specifications

There is no special name format for the XSD aggregate module. However, it should include the name of the main entity being defined in it.

The namespace of the XSD message module is:
http://www.tmforum.org/mTOP/<DDP>/xsd/<Module>/<MajorVersion> (See 4.1.1.4)

The XSD module marker is based on the acronym/abbreviation of the XSD aggregate module filename.

4.1.3.3 XSD Definitions

There is a distinct XSD aggregate module for each one of the following mTOP entity types:

- mTOP managed objects and the various sets of common attributes (NetworkResourceBasic:CommonResourceInfoType, NetworkResourceFulfillment:ManagedElementType)
- mTOP events (Framework:CommonEventInfoType, Framework:HeartbeatType)
- other mTOP aggregate definitions utilized for specific complex data structures (e.g. NetworkResourceBasic:TpDataType, NetworkResourceBasic:LayeredParametersType)

Refer to the ComplexType usage guidelines in section 4.2.3.

4.1.3.4 Usage

The use of the XSD definitions of an mTOP entity defined in the XSD aggregate module includes:

- The XSD data type of the mTOP entity is referenced by a local element of an XSD message module definition (by containment as an operation parameter).
- The XSD data type of the mTOP entity is referenced by another XSD aggregate module for the XSD definitions of another mTOP entity (by containment as an object attribute).
- The XSD data type of the mTOP entity is referenced by extension in another XSD aggregate module for the XSD definitions of another mTOP entity (by inheritance of the object attributes). This is the case for the XSD definitions of an mTOP managed object, which extend from another XSD aggregate module defining the common set of the mTOP managed object attributes.
- Finally, the mTOP header XSD element definition from the *Framework:HeaderDefinitions.xsd* module is directly used as the part inserted in the SOAP Header of all request/response messages.

4.1.3.5 Versioning and Change Policy

The following mTOP XSD aggregate specification changes are carried out as a new major version of the module:

- One or many attributes are deprecated (removed/deleted as unused)
- One or many attribute types have changed,
- One or many new mandatory attributes have been added

The following mTOP XSD aggregate specification changes are carried out as a new minor version of the module:

- One or many new optional attributes have been added

4.1.4 XSD Dataset Module

4.1.4.1 Purpose

An mTOP XSD dataset module contains the XML Schema definitions of a set or collection of mTOP parameter data types. The XSD definitions of these mTOP parameters generally consist in a *simpleType* and various forms of *restrictions* as defined in the IA.

As with the XSD aggregate module, the scope of the XSD dataset module is usually limited to the XSD definitions of a set of related mTOP parameters (e.g. All ITU-T parameters are defined in the *NetworkResourceBasic:ITU-T-Definitions.xsd* module).

4.1.4.2 Module Specifications

There is no special name format for the XSD dataset module. However, it should be self explanatory of the mTOP parameters being defined in it.

The namespace of the XSD message module is:
http://www.tmforum.org/mTOP/<DDP>/xsd/<Module>/<MajorVersion> (See 4.1.1.4)

The XSD module marker is based on the acronym/abbreviation of the XSD dataset module filename.

4.1.4.3 XSD Definitions

Examples of XSD dataset modules defined for important sets/collections of mTOP parameters are:

- General mTOP data type definitions such as LocationType found in the *Framework: GeneralDefinitions.xsd* module
- The definition of all the layer rates
- The definition of all the ITU-T parameters
- The definition of all layered parameters (not expected for MTOSI R2.0) with a module per specific technology groupings
- The definition of all alarm probable causes
- The definition of PGP parameters

Refer to the SimpleType usage guidelines in section 4.2.2.

4.1.4.4 Usage

The use of the XSD definitions of an mTOP entity defined in the XSD aggregate module includes:

- The XSD data type of the mTOP entity is referenced by a local element of an XSD message module definition (by containment as an operation parameter).
- The XSD data type of the mTOP entity is referenced by an XSD aggregate module for the XSD definitions of an mTOP entity (by containment as an object attribute).

4.1.4.5 Versioning and Change Policy

The following mTOP XSD dataset specification changes are carried out as a new major version of the module:

- One or many simple data types have been deprecated,
- One or many simple data types have been modified,

The following mTOP XSD dataset specification changes are carried out as a new minor version of the module:

- One or many simple types have been added to the dataset

4.2 XML Schema Definitions Usage

4.2.1 Attribute Definitions

Use of XML Schema attribute definitions is avoided as much as possible in all of the mTOP WS IIS. It is only reserved for some very specific XSD definitions. This is a well-known XSD design principle to allow for the potential extension of the data.

Therefore, all mTOP XML message data are captured as element definitions based on simple or complex types.

4.2.2 Simple Type Definitions

The XML Schema design considerations for all mTOP parameters based on an XSD simpleType definition are as followed:

- Define an XSD simpleType datatype for each mTOP parameter definition that is referenced in many mTOP entity or message definitions (re-use). Otherwise, the mTOP datatype is to be avoided (reduce number of data types – lightweight IIS).
  Note this should also be the same consideration as in the mTOP IA.
- Avoid use of any “advanced” simpleType definitions (See Ref 6).
- All mTOP simpleType definitions should rely on an XSD primitive without any restrictions, such as:
  - A string with no patterns or any other advanced form of restrictions
  - A boolean
  - A basic numerical (integer, double)
  - date, time, and dateTime
  - anyURI
- Refer to the mTOP Enumeration proposal (See Appendix A: Proposal for Versioning and Extension of an mTOP XML Enumeration Entity) regarding the XSD definition of an mTOP parameter that is based on an enumeration of possible values (i.e. layer rates), which supports vendor/proprietary extensions and minor version specification extensions.

4.2.3 Complex Type Definitions

The XML Schema design considerations for all mTOP parameters based on an XSD simpleType definition are as followed:

- Avoid use of any “advanced” simpleType definitions (See Ref 6).
- All mTOP complexType definitions should rely exclusively on the following XSD definitions:
  - The *sequence* of a set of elements
- The following XSD notations and patterns must NOT be used in the mTOP XSD modules:
  - *all* as a generally accepted notation to be avoided
  - *restriction* from another complexType
  - *union* as it is not fully supported by number of data binding toolkits
- The following “advanced” XSD notations are used in the mTOP WS IIS:
  - *choice* for the definition of the operation exception message

4.2.4 Global Element Definitions

The XML Schema design considerations for the XSD definition of any global elements are as followed:

- All XSD definitions imported into WSDL definitions have to be referenced as XSD global elements. That includes:
  - All the mTOP service interface messages (from an XSD message module), and
  - The mTOP header (from an XSD aggregate module).
- All XSD definitions referenced by other XSD definitions for the purposeof:
  - Allowing substitution of the local element base definition (extension with substitution of the data type)

5 XML Guidelines

5.1 Introduction

The purpose of this section is to provide a set of XML usage recommendations, which are not captured by the WS IIS as they are not necessary interoperable specifications. But, they could help, if adopted, by promoting all mTOP users to implement a standardized mTOP XML style. The benefits are:

- Readability of the XML messages as they all have the same style (debugging, integration testing)
- Optimization of the XML messages (performance)

The content of this section shall be part of the mTOP WS IIS user guidelines.

5.2 XML Document

The following recommendations can be applied to the generation of all the mTOP XML messages:

- Namespace declarations
- Default namespace (no prefix)
- mTOP namespace prefixes table
- Indentation
- End of line (Line feed / carriage return)
- Special characters
- Vendor extensions
- Comments

TODO: This section can be completed prior to the MTOSI Release 2.0

6 Change History

The purpose of this section is to summarize the evolution/changes of the mTOP WS design guidelines / best-practices between product releases.

6.1 MTOSI R1.1 to R2.0

TODO: This section can be completed prior to the MTOSI Release 2.0.

The following list includes all the important new or updated features introduced into the development of MTOSI R2.0 WS IIS:

- MTOSI R2.0 is based on the new mTOP DDPs. MTOSI R1.x had two namespaces; ***tmf854.v1*** for all the XML Schema definitions and ***tmf854.v1.ws*** for all the WSDL definitions. The overall MTOSI R2.0 WS IIS is now available as a set of namespaces partitioned based on the following criteria:
  - DDP partitioning,
  - Concept of module based on XML artifact type (XSD or WSDL) and functional scope (service interface definitions, logical entity type definitions grouping)

  See the section 2.1 for the new mTOP information architecture with the new:
  - Development view based on GForge Subversion source control management style
  - Deployment view proposal

  Also, I started updating the mTOP_Guidelines_DDP_Maps.xls with the description of the modules in each one of the DDPs.

  The following entries describe the changes brought to this new major release of MTOSI by going through a top-down review of the mTOP WS specifications.
- Exception handling WS definitions have changed. There is no more generic exception message for all operations, but one dedicated exception message per operation.
- Iterator operation is re-organized based on DDP and service interface concepts. All object retrievals are now re-structured with a dedicated service interface based on object type with their specific object and object name iterator operations.
- Notification service
  - All mTOP notification service interface WS definitions (NotificationProducer, NotificationConsumer, and NotificationBroker) are in the Framework DDP
  - The notify message is defined with a basic event structure
  - All mTOP event schema definitions are extended from this Framework DDP base definition
- WSDL Specifications:
  - Service Interface (WSDL module): Re-structuring of the WSDL definitions in mTOP Service Interface (SI) WSDL module. ().

- XML Schema (XSD) Specifications:
  - Partitioning of the XML Schema definitions based on XSD modules ()
- Other Issues to Resolve
  - Generation of the documentation based on the XSLT from the XSD modules for potential replacement of some of the SD. May be used to also cover a light weight HTML based documentation of the entire WS IIS as replacement of the above HTML documentation (WSDLDoc)
  - Reminder to check IBM WSDLdoc utility at http://awwebx04.alphaworks.ibm.com/ettk/wsdldoc/wsdlUtility.html
  - eBay WS and other XML on-line documentation/guidelines
    - http://developer.ebay.com/DevZone/SOAP/docs/WebHelp/wwhelp/wwhimpl/js/html/wwhelp.htm
    - http://developer.ebay.com/DevZone/SOAP/docs/WSDL/index.htm
    - http://developer.ebay.com/DevZone/SOAP/docs/Schema/index.html

7 References

7.1 External References

Ref 1 W3C Extensible Markup Language (XML) - http://www.w3.org/XML/

Ref 2 W3C Wed Service Description Language version 1.1 (WSDL) – http://www.w3.org/TR/wsdl

Ref 3 W3C Simple Object Access Protocol version 1.1 (SOAP) - http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

Ref 4 W3C XML Schema version 1.0 (XSD) - http://www.w3.org/XML/Schema

Ref 5 Web Services Interoperability WS-I http://www.ws-i.org/

Ref 6 W3C XML Schema Patterns for Databinding - http://www.w3.org/2002/ws/databinding/

Ref 7 W3C XML Schema Patterns for Common Data Structures - http://www.w3.org/2005/07/xml-schema-patterns.html

Ref 8 SOA Transformation Services from Zapthink (SOA web portal) http://www.zapthink.com/search.html?search=transformation & http://www.zapthink.com/report.html?id=ZAPFLASH-2006103

7.2 Internal References

Ref 9 TMF Sourceforge mTOP

Ref 10 MTOSI_DDP_Maps.xls

Ref 11 Framework DDP - SD2-5 Communication Styles

Ref 12 Framework DDP - SD2-2 XML Implementation User Guide

Ref 13 Framework DDP - SD2-6_MTOSI_Versioning_and_Extensibility.doc

Ref 14 Framework DDP - SD0-1_Dictionary

8 Appendix A: Proposal for Versioning and Extension of an mTOP XML Enumeration Entity

8.1 Purpose

The purpose of this section is to capture a methodology for the XML definition of all the mTOP enumerations.

In MTOSI R1.1 (current), the definition of an enumeration has been found difficult to implement because of the XML Schema notation usage (Details are captured in option#1 below).

This document proposes a review of all XML Schema design options to be used for the definition of an XML enumeration data type that supports extensions. And, a solution is required for MTOSI R2.0.

As for all other WS IIS design patterns, the goals are:

- Ensure that the specifications support a full inter-operability (no ambiguities),
- Ensure that the specifications are extensible by vendor extensions (proprietary definitions), and
- Ensure that the mTOP specifications can easily evolve/migrate (promote forward/backward compatibility across releases and converge with other standard patterns)

8.2 Options Evaluation

XML artifacts highlighting an example for each one of the options presented below are captured in the attached ZIP file.

EnumerationOptions.zip

8.2.1 Option#1: Union of Multiple Restriction Types (Current)

Since MTOSI R1.0, the XML Schema of a data type that needs to support any form of extensions is defined as a simple type that is represented by multiple other simple types that are bound together using the xsd:union notation.

Therefore, some MTOSI R1.x enumeration data types use the xsd:union to allow for:

- Minor version extension of the specification (a simple type with pattern-based restriction to allow values with the format: “MINOR_*”), and
- Vendor (proprietary) extension of the specification (a simple type with pattern-based restriction to allow values with the format: “PROP_*”)
- The simple type with all enumeration restrictions as in the specification (option#2).

This option is also described in a W3C paper (Ref 7). Interestingly, it is in contradiction with another paper written by the same author (Ref 6) as it indicates that the use of the xsd:union notation is to not recommended (advanced pattern).

Refer to XML examples:

- option1.xsd as sample schema definition
- option1.xml as sample instance with two supported form of extensions

8.2.2 Option#2: Free String

This option is used in the MTNM (IDL definition) for all enumeration data types that have to support extensions.

Refer to XML examples:

- option2.xsd as sample schema definition
- option2.xml as sample instance with all supported form of extensions

8.2.3 Option#3: Enumeration Restriction

This option is also used in MTOSI R1.x for the enumeration data types that have been found stable (no need to support extensions). This is the typical XML Schema design pattern for a data type representing an enumeration of possible values.

Refer to XML examples:

- option3.xsd as sample schema definition
- option3.xml as sample instance with no extensions (not supported - invalid)

8.2.4 Option#4: Pattern Restriction

This option is similar to the option#1 (current). The difference is that all three simple type definitions are encapsulated into a single pattern restriction.

Refer to XML examples:

- Option4.xsd as sample schema definition
- Option4.xml as sample instance with no extensions (not supported - invalid)
- Option4.1.xsd as a sample schema with a minor version extension (one new possible value)

8.2.5 Option#5: Substitution Group

This option is the recommended approach used by WS guidelines of many standard organizations (OASIS, OSS/J). It is based on the extension by substitution of the enumeration data (base) type.

Refer to XML examples:

- Option5.xsd as sample schema definition
- Option5.1.xsd as a sample schema with a minor version extension (one new possible value)

- Option5.xml as sample instance with the extended enumeration usage

Note that the naming of the enumeration data type must be versioned.

The complete re-definition of an enumeration is possible including the deletion of previously supported terms. So, minor version can only be extensions by addition of new enumerated terms. And, a major version definition is used when some terms are deprecated (removed) in a new specifications release.

The specification data type can also be extended by a vendor with its own data type including additional vendor/proprietary enumerated terms. A naming convention is necessary to distinguish different vendor extensions from the same mTOP enumeration data type definition (i.e. "<EnumBase><MajorVersion>_<MinorVersion>_<VendorPrefix>Type").

Finally, in all two forms of enumeration data type extensions, the validation of the XML enumeration instance requires access to the extended specifications schema (new specification release/version or vendor specifications).

8.2.6 Option#6: Complex Type with Attribute

This last option is another XML Schema design pattern that supports the extension of an enumeration data type. An enumerated term is reserved to indicate the type of the extended value (VENDOR_EXT and MINOR_EXT), which is carried by an attribute in the XML instance. Note that this XML Schema relies on a complex type definition.

Refer to XML examples:

- Option6.xsd as sample schema definition
- Option6.1.xsd as a sample schema with a minor version extension (one new possible value)
- Option6.xml as sample instance with the various extensions usage

8.2.7 Evaluation of all Options

	Pros	Cons
Option#1	Support minor/vendor extensions	Advanced pattern
Option#2	Support all extensions (always forward/backward compatible)	Not normative (anything is possible)
Option#3	The XML Schema enumeration pattern for stable data types (no risk of changing and no requirement for vendor extensions)	No support for extensions
Option#4	Support minor/vendor extensions	Another advanced pattern but simpler than option#1
Option#5	Support minor/vendor extensions Convergence with OSS/J	Also an advanced pattern that requires a naming convention (versioning/vendor definitions) with good documentation
Option#6	Support minor/vendor extensions	Does not rely on an advanced pattern

8.2.8 Solution Adopted

The adopted solution for the XML Schema design pattern of an mTOP XML enumeration data type includes:

- Use option#3 when the data type represents an aspect of the specifications that:
  - will not change in future releases, and
  - does not require to support vendor extensions
- Otherwise, use option#6 as it relies on a simple and efficient pattern. Especially regarding the transparent (non-impact) on handling of minor version extension of the specification definition.

9 Appendix B: Proposal for Versioning and Extension of an mTOP XML Aggregate Entity

9.1 Purpose

The purpose of this section is to study and propose a solution for the XML Schema definitions of an mTOP aggregate (XSD complexType) entity data structure with respect to its versioning and extensions.

Notes:

- Any major changes are not covered here as they include a significant and incompatible re-definition of the mTOP aggregate entity.
- Adding vendor proprietary elements to an mTOP XML aggregate entity is same as in MTOSI R1.x using the dedicated “vendorExtension” element that represents a sequence of generic elements from any other namespace (<xsd:any>)..

9.2 Options Evaluation

XML artifacts highlighting an example for each one of the options presented below are captured in the attached ZIP file.

AggregateExtensions.zip

9.2.1 Option#1: Schema Extension

This option proposes a simple inheritance pattern with the extension of an mTOP aggregate entity to the next minor version (i.e. 1.0 -> 1.1).

Refer to XML examples:

- CommonObjectv1.xsd – XSD aggregate module of the common object base definition in xmlns “http://mtop.coobj1-0”
- ObjectAv1.xsd – XSD aggregate module of the ObjectA (version1.0) and its list/array definitions in xmlns “http://mtop.objа1-0”. ObjectA is an extension of CommonObject.
- ObjectAv1_1.xsd – XSD aggregate module of the ObjectA1 (version1.1) and its list/array definitions in xmlns “http://mtop.objа1-1”. ObjectA1 is an extension of ObjectA.
- listOfObjectAv1.xml – instance list of ObjectA
- listOfObjectAv1_1.xml – instance list of ObjectA1

9.2.2 Option#2: Schema Substitution

This option proposes a substitution pattern of an mTOP aggregate entity to the next minor version.

Refer to XML examples:

- CommonObjectv1.xsd – XSD aggregate module of the common object base definition in xmlns "http://mtop.coobj1-0"
- ObjectAv1.xsd – XSD aggregate module of the ObjectA (version1.0) and its list/array definitions in xmlns "http://mtop.objа1-0". ObjectA is an extension of CommonObject.
- ObjectAv1_1.xsd – XSD aggregate module of the ObjectA1 (version1.1) definition in the same xmlns "http://mtop.objа1-1". ObjectA1 is an extension of ObjectA.
- listOfObjectAv1.xml – instance list of ObjectA
- listOfObjectAv1_1.xml – instance list of ObjectA & ObjectA1

9.2.3 Option#3: Schema Containment (Any)

This option proposes a pattern where an mTOP aggregate entity base XSD definition allows extension by containment of all its attributes added in minor versions.

Refer to XML examples:

- CommonObjectv1.xsd – XSD aggregate module of the common object base definition in xmlns "http://mtop.coobj1-0"
- ObjectAv1.xsd – XSD aggregate module of the ObjectA (version1.0) and its list/array definitions in xmlns "http://mtop.objа1". ObjectA is an extension of CommonObject.
- ObjectAv1_1.xsd – XSD aggregate module with the XSD definitions of minor version (1.1) attributes in the special extended namespace of the mTOP entity xmlns "http://mtop.objа1.ext". ObjectA minor version attributes are confined in the minorExt element, which is defined as a sequence of any elements from the extended namespace.
- listOfObjectAv1.xml – instance list of ObjectA
- listOfObjectAv1_1.xml – instance list of ObjectA & ObjectA1

9.2.4 Option#4: Schema Inclusion (updated)

This option proposes a pattern where the XSD definition of an mTOP aggregate entity is updated to include a new attribute as a minor version change.

Refer to XML examples:

- CommonObjectv1.xsd – XSD aggregate module of the common object base definition in xmlns "http://mtop.coobj1"
- ObjectAv1.xsd – XSD aggregate module of the ObjectA (version1.0) and its list/array definitions in xmlns "http://mtop.objа1". ObjectA is an extension of CommonObject.
- ObjectAv1_1.xsd – XSD aggregate module with the updated ObjectA (version1.1) XSD definitions including the new attribute (attribute2) and its marking with XML attribute (minorVersion set to constant value 1). Note this definition uses the same namespace as in previous version of ObjectA (1.0).
- listOfObjectAv1.xml – instance list of ObjectA (version 1.0)
- listOfObjectAv1_1.xml – instance list of ObjectA (version 1.1)

Notes:

- Version number in namespace only has the major version number as in MTOSI R1.x
- However, the use of *tmf854Version* attribute (as in MTOSI R1.x) associated with the aggregate element (objA in the example) is replaced by an attribute for each extended attribute (minorVersion=1 for new attribute2 added to version 1.1 of the ObjectA XSD specification). It facilitates the identification of all extended attributes of an aggregate.
- The namespace remains the same for the two XSD modules released for version 1.0 and 1.1 of the same ObjectA XML Schema module specifications. Only, the version attribute of the root XML document of the XML Schema module specifications captures the full version.
- Consumer of listOfObjectAv1_1.xml that is built with the version 1.0 of the ObjectA XSD specification must:
  - Use a pre-processor to remove the unknown “attribute2” element prior to validation or binding the XML element with its native object model (e.g. Java using JaxB or XMLBeans), or
  - Relax its validation to ignore the unknown “attribute2” element. XMLBeans runtime databinding has that capabilities of ignoring additional (extended) elements that are not part of the known (compiled) XML Schema.

9.2.5 Evaluation of all Options

	Pros	Cons
Option#1	Class always represent latest updated model by inheritance from previous version (extension)	All up-stream class definitions where modified class is used (all parents up to root) are impacted (require new version) No transparent backward compatibility (require transformation)
Option#2	Class always represent latest updated model by inheritance from previous version (extension) Class reference substitution/overloading (no impact/re-definition on all parents up to root)	No transparent backward compatibility (require transformation)
Option#3	Class footprint remains unchanged in all minor version updates. Transparent forward/backward compatibility between minor versions.	Class does not reflect latest version footprint (hide minor extensions/changes evolution as in additional info)
Option#4	Simple handling of minor version changes (extensions).	All minor version specifications of the same major version (e.g. ObjectA 1.0, 1.1…1.n) have the same namespace. Require special handling for validation or databinding (See above)

9.2.6 Solution Adopted

The solution for MTOSI R2+ is to use option#4:

10 Administrative Appendix

10.1 Document History

Version Number	Date Modified	Description of changes
1.0	May 2008	First consolidated version of this document
1.1	Dec 2009	Minor editorial changes

10.2 Acknowledgments

First Name	Last Name	Company
Michel	Besson	Amdocs
Steve	Fratini	Telcordia

10.3 How to comment on this document

Comments and requests for information must be in written form and addressed to the contact identified below:

Jérôme	Magnet	Nortel
Phone:	1-613-763-1480	
Fax:		
e-mail:	jeromem@nortel.com	

Please be specific, since your comments will be dealt with by the team evaluating numerous inputs and trying to produce a single text. Thus we appreciate significant specific input. We are looking for more input than wordsmith" items, however editing and structural help are greatly appreciated where better clarity is the result.