NETMOD Working Group
Internet-Draft
Intended status: Informational
Expires: April 30, 2017

S. Mansfield, Editor
Ericsson Inc.
B. Zeuner
Deutsche Telekom AG
N. Davis
Ciena
Y. Yun
Fiberhome
Y. Tochio
Fujitsu
K. Lam
E. Varma
Nokia

October 27, 2016

Guidelines for Translation of UML Information Model to YANG Data Model

draft-mansfield-netmod-uml-to-yang-03

Abstract

This document defines guidelines for translation of data modeled with UML to YANG including mapping of object classes, attributes, data types, associations, interfaces, operations and operation parameters, notifications, and lifecycle.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as “work in progress”.

This Internet-Draft will expire on April 30, 2017.

Copyright Notice

Table of Contents

1. Introduction

As discussed in draft-lam-teas-usage-info-model-net-topology [5] a Data Model (DM) may be derived from an Information Model (IM). However, in order to assure a consistent and valid data modelling language representation that enables maximum interoperability, translation guidelines are required. A set of translation rules also assists in development of automated tooling.

This draft defines guidelines for translation of data modelled with UML [6] (as constrained by the ONF's UML Modeling Guidelines [7]) to YANG (defined in RFC6020 [1] and YANG 1.1 [4]) including mapping of object classes, attributes, data types, associations, interfaces, operations and operation parameters, notifications, and lifecycle.

2. Terminology

The following terms are defined in RFC6020 [1]

- anydata
- anyxml
- augment
- container
- data node
- identity
- instance identifier
- leaf
- leaf-list
- list
- module
- submodule

The following terms are defined in UML 2.4 [6]

- association
- attribute
- data type
- interface
- object class
- operation
- parameter
- signal (used to model notifications)

3. Overview

This document defines translation rules for all constructs used in a UML based IM to a data model using YANG.

While some mapping rules are straightforward, an IM in UML uses some constructs that cannot be mapped directly to a DM using YANG and conventions are described to make the translation predictable. Additionally, in some cases multiple mapping approaches are possible and selection among these is also necessary to assure interoperability.

Mapping guidelines for these constructs are provided in the following sections.

4. Mapping Guidelines

4.1 Mapping Guideline Considerations

Where "??" is inserted in the table, it means that the specific mapping is for further study as it is either as yet unclear how to map the construct or that there are multiple ways of doing the mapping and a single one needs to be selected.

A table will be included summarizing constructs in UML that do not directly map to YANG and where in this draft the associated guidelines for mapping these constructs will be provided.

4.2 Mapping of Object Classes

Object Class		
- Real object classes having/inheriting at least one attribute identified as "partOfObjectKey" will be mapped to a "list" statement - Real object classes not having/inheriting any attribute identified as "partOfObjectKey" will be mapped to a "container" statement - Abstract object classes used for inheritance will be mapped to a "grouping" statement		
UML Artifact	YANG Artifact	Comment
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
superclass(es)	"grouping" statement	Concrete superclasses are then mapped to container/list which uses these groupings.
abstract	"grouping" statement	It is possible that the superclass or abstract class contains the key attribute for the instantiated subclass, this requires the creation of the grouping but later when the subclass is instantiated the key value must be identified from within the grouping.
object identifier Note: Attributes used as object identifier are defined in UML by the attribute property "partOfObjectKey".	list::"key" substatement	It is possible that the super-class or abstract class contains the key attribute for the instantiated subclass.
object identifier list Does not appear in the UML when mapping to YANG.		The splitting of a list attribute (marked as key) into a single key attribute and an additional list attribute will be done in UML during Pruning and Refactoring. i.e. The mapping tool will never get a list attribute which is part of the object identifier.
objectCreationNotification [YES/NO/NA]	"notification" statement	Goes beyond the simple "a notification has to be sent": a tool can construct the signature of the notification by reading the created object.
objectDeletionNotification	"notification" statement	Goes beyond the simple "a notific-

[YES/NO/NA]		ation has to be sent": a tool can construct the signature of the notification by reading the deleted object. (i.e. not necessary to provide the attributes of the deleted object).
muliplicity >1 on association to the class	list::"min-elements" "max-elements" substatements	min-elements default = 0 max-elements default=unbounded mandatory default=false
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description
Proxy Class XOR	"choice" substatement	
support	"if-feature" substatement	Support and condition belong together. If the "support" is conditional, then the "condition" explains the conditions under which the class has to be supported.
condition		
operation	"action" substatement	YANG 1.0 supports only rpc -> add prefix to the rpc name; i.e. objectClass::rpc; "action" requires YANG 1.1
Conditional PACs	container::presence" substatement	
constraint property	list::"unique" substatement	UML is not able to define a group of attributes to be unique as YANG can do using the "unique" substatement.
{<constraint>}	"when" substatement	

Figure 1: Mapping of Object Classes

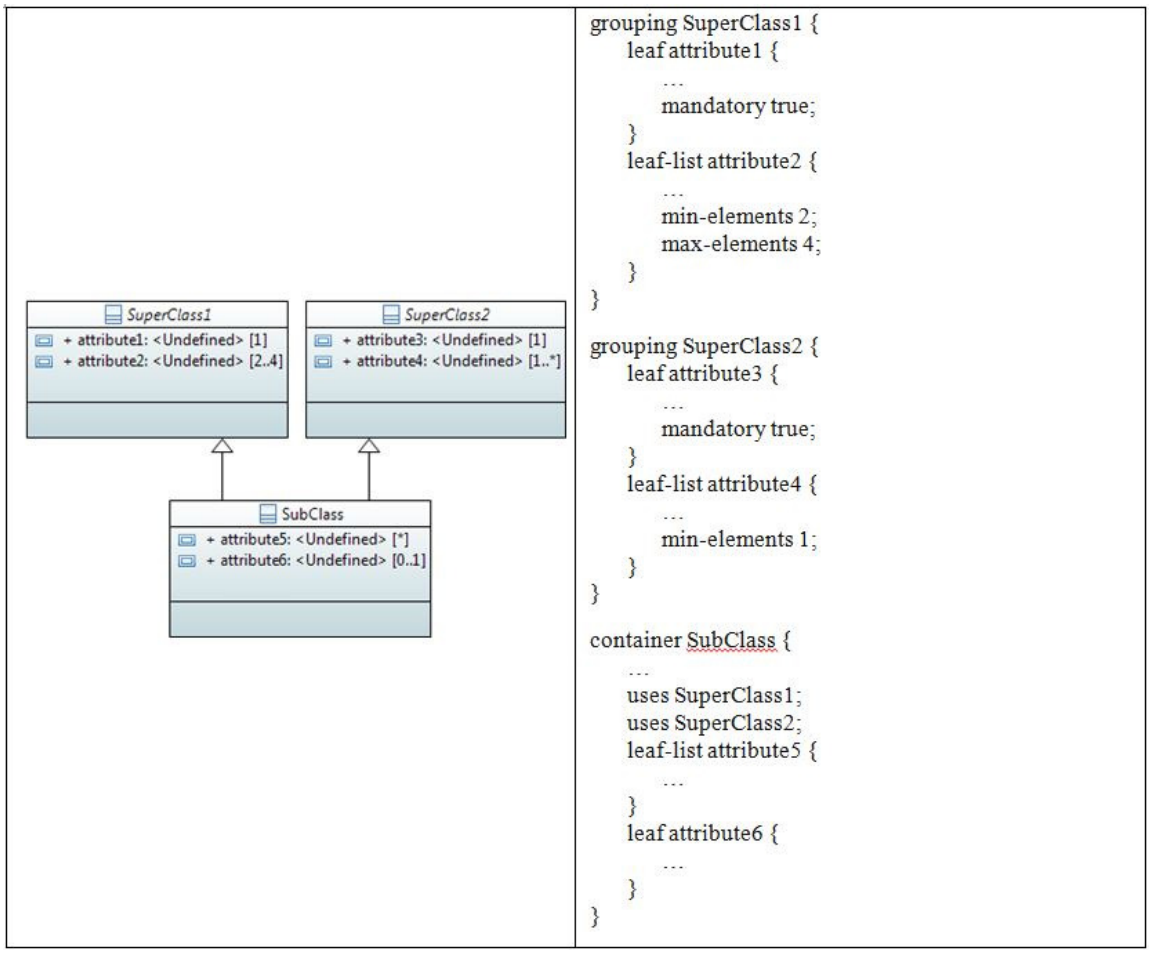
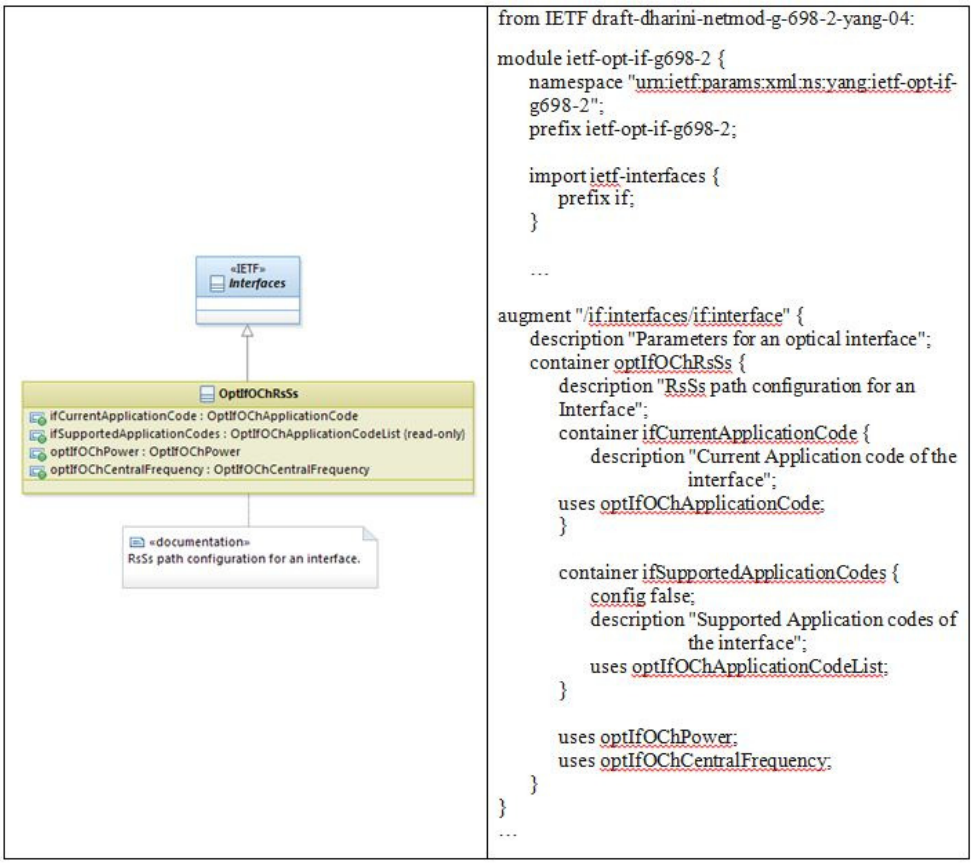


Figure 2: Example of Abstract Object Class Mapping (Available in PDF or HTML versions)




```
from IETF draft-dharini-netmod-g-698-2-yang-04:

module ietf-opt-if-g698-2 {
    namespace "urn:ietf:params:xml:ns:yang:ietf-opt-if-
    g698-2";
    prefix ietf-opt-if-g698-2;

    import ietf-interfaces {
        prefix if;
    }

    ...

augment "/if:interfaces/if:interface" {
    description "Parameters for an optical interface";
    container optIfOChRsSs {
        description "RsSs path configuration for an
        Interface";
        container ifCurrentApplicationCode {
            description "Current Application code of the
                         interface";
        uses optIfOChApplicationCode;
        }

        container ifSupportedApplicationCodes {
            config false;
            description "Supported Application codes of
                         the interface";
            uses optIfOChApplicationCodeList;
        }

        uses optIfOChPower;
        uses optIfOChCentralFrequency;
    }
}
...
```

Figure 3: Example of Non-Abstract Object Class Mapping (Available in PDF or HTML versions)

4.3 Mapping of Attributes

Attribute --> "leaf" (single) or "leaf list" (multiple) statement		
UML Artifact	YANG Artifact	Comment
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
type	"type" substatement (built-in or derived)	
readOnly	"config" substatement (false)	config default = true
isOrdered	"ordered-by" substatement ("system" or "user")	ordered-by default = system
isUnique	No unique sub-statement in leaf-list	Only relevant for multi-valued attributes.
Multiplicity (carried in XMI as lowerValue and upperValue)	"mandatory" or "min-elements" and "max-elements" substatements [0..1]=>no mapping needed; is default substatement=false [1]=>mandatory substatement=true [0..x]=> no mapping needed; is default [1..x]=> min-elements substatement = 1 [0..3]=> max-elements substatement = 3	min-elements default = 0 max-elements default=unbounded mandatory default=false
defaultValue	"default" substatement	If a default value exists and it is the desired value, the parameter does not have to be explicitly config-ured by the user.
isInvariant	"extension" substatement -> ompExt:isInvariant	
valueRange	For string type: "pattern" and/or "length substatement of "type" substatement For integer and decimal type: "range" substatement of "type" substatement For all other typed attributes and for string or integer or decimal typed attributes where the UML definition is not compliant to YANG: "description" substatement	The tool should provide a warning at the output of mapping process notifying when one or more UML valueRange definitions are contained in the description substatement of the corresponding leaf or leaf-list. When the value of "valueRange" is "null", "NA", "See data type", the tool ignores it and doesn't print out "range" substatement.
passedByReference	if passedByReference = true -> type leafref { path "/<object>/<objectidentifier>"} if passedByReference =	Relevant only to attributes that have an object class defined as their type.

	false -> either "list" statement (key property, multiple instances) or "container" statement(single instance)	
partOfObjectKey > 0	list::"key" substatement	It is possible that the (abstract) superclass contains the key attribute for the instantiated subclass.
unit	"units" substatement	
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description
support	For conditional support only:	Support and condition belong together. If the "support" is conditional, then the "condition" explains the conditions under which the class has to be supported.
condition	"if-feature" substatement "when" substatement if condition can be formalized as XPath expression (i.e., it is conditioned by the value of another attribute)	
error notfication??	"must" substatement	
{<constraint>}	"when" substatement	

Figure 4: Mapping of Attributes

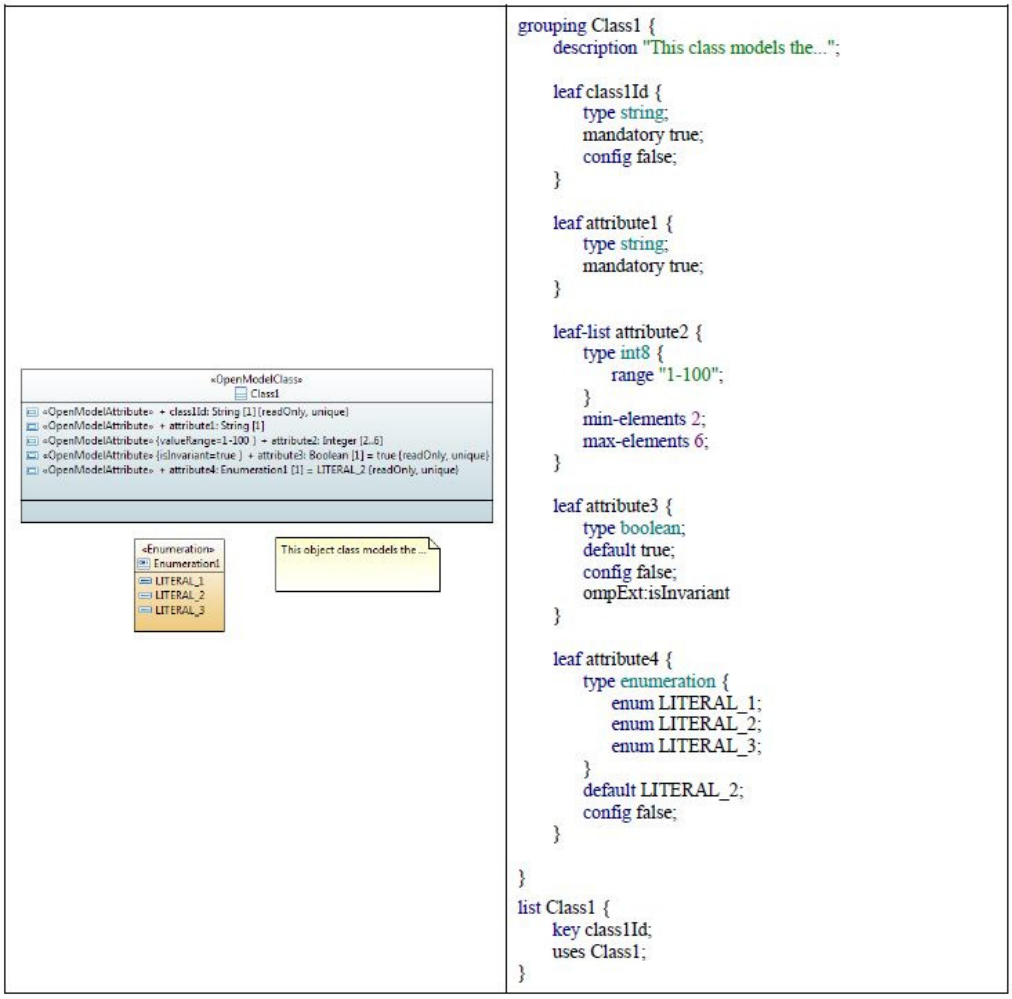


Figure 5: Example of Attribute Mapping (Available in PDF or HTML versions)

4.4 Mapping of Types

Types		
UML Artifact	YANG Artifact	Comment
Primitive Type	Built-In Type if defined	e.g., Integer
Complex Data Type	"grouping" statement	e.g., date-time
Enumeration	"enum" statement	

Figure 6: Mapping of Types

4.4.1 Mapping of Primitive Types

Primitive Type -> "typeDef" statement		
UML Artifact	YANG Artifact	Comment
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
type	"type" substatement (built-in type or derived type)	
defaultValue	"default" substatment	If a default value exists and it is the desired value, the parameter does not have to be explicitly configured by the user. When the value of "defaultValue" is "NA", the tool ignores it and doesn't print out "default" substatement.
unit	"units" substatement	
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description

Figure 7: Mapping of Primitive Types

4.4.2 Mapping of Complex Data Types

Complex Data Type -> "grouping" statement		
UML Artifact	YANG Artifact	Comment
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
not used	"action" substatement	
XOR	"choice" substatement	
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description
complex attribute	"uses" statement	

Figure 8: Mapping of Complex Data Types

Leaf and leaf-list can only use built-in types, typeDef types or enumerations in their type substatement; i.e., not groupings. Complex data types with more than one item (e.g., name value pair) can only be defined using groupings. Groupings can only be used by grouping, container and list statements.

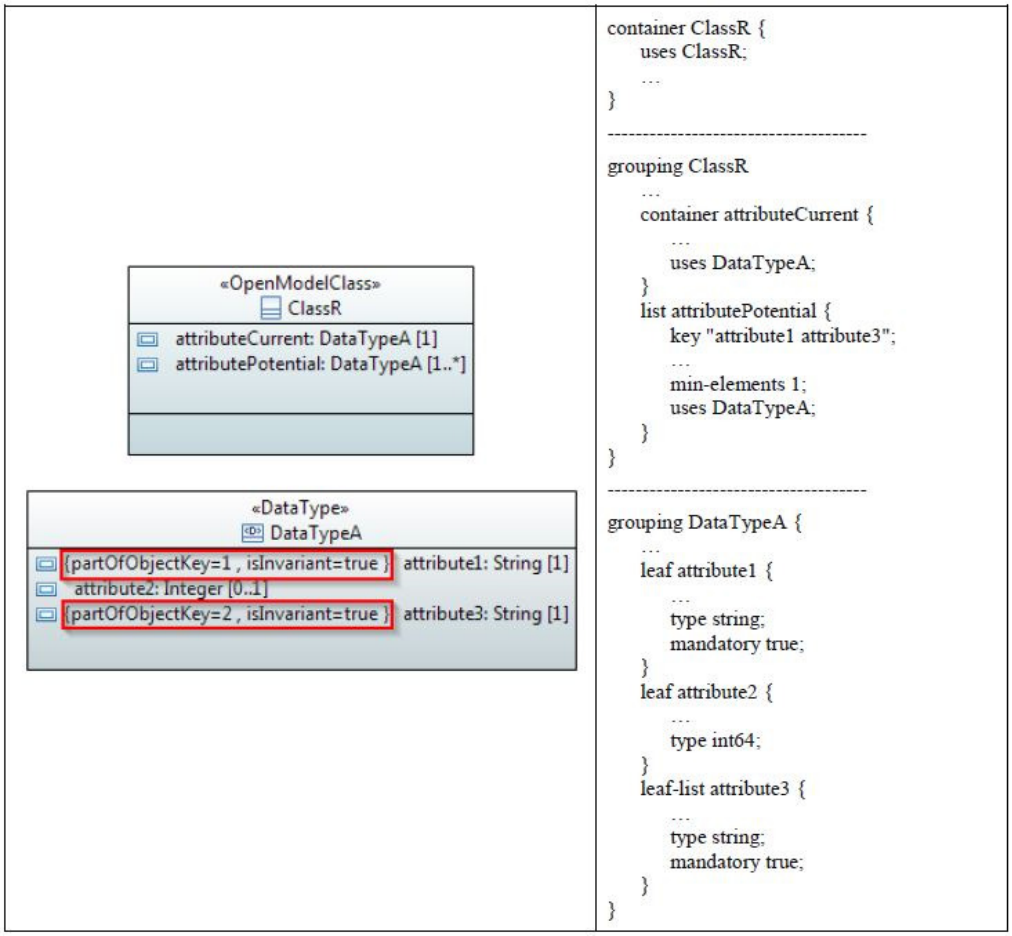
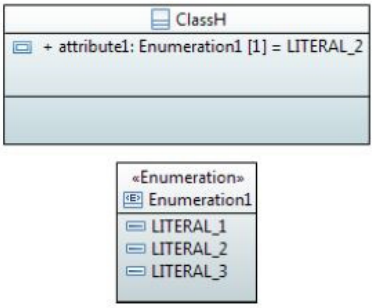


Figure 9: Example of Complex Data Type Mapping (Available in PDF or HTML versions)

4.5 Mapping of Enumeration Types

Enumeration Type -> "enum" statement typedef for reusable (indirect usage) enumerations identity statement?		
UML Artifact	**YANG Artifact**	**Comment**
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
literal name	enum name	
literal integer	"value" substatement	
isLeaf = true isLeaf = false	"enum" substatement "identity"/"base" pattern	UML definition ""
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description

Figure 10: Mapping of Enumeration Types



Using "enumeration"

direct usage (not recommended):

```
container ClassH {
    ...
    leaf attribute1 {
        type enumeration {
            enum LITERAL_1;
            enum LITERAL_2;
            enum LITERAL_3;
        default LITERAL_2;
        mandatory true;
        }
    }
}
typedef Enumeration1 {
    type enumeration {
        enum LITERAL_1;
        enum LITERAL_2;
        enum LITERAL_3;
    }
}
```

indirect usage:

```
container ClassH {
    ...
    leaf attribute1 {
        type Enumeration1;
        default LITERAL_2;
    }
}
```

Using "identity"/"base"

```
// an empty identity value is a "base identity"
// i.e. it provides the reference name for a set of values
identity Enumeration1;
identity LITERAL_1 {
    // the "base" statement qualifies this identity value
    // as belonging to the AdministrativeState set
    base Enumeration1;
}
identity LITERAL_2 {
    base Enumeration1;
}
identity LITERAL_3 {
    base Enumeration1;
}
.....
typedef Enumeration1 {
    type identityref {
        // "identityref" defines the associated set
        base Enumeration1;
    }
}
.....
leaf attribute1 {
    type Enumeration1;
}
```

Figure 11: Example of Enumeration Type Mapping (Available in PDF or HTML versions)

4.6 Mapping of Associations

Associations		
UML Artifact	YANG Artifact	Comment
Inheritance	abstract superclass: "grouping" statement concrete superclass: "augment" statement	Multiple inheritance can also be mapped using "groupings" Need to define when augment is used. Note: Augmentation can be conditional.
Composition with (aggregation= 'composite') "passed by value"	"container" statement containing "list" statement(s) (multiple contained instances) or "container" statement(s) (single contained instances)	How to map "passed by reference"??
Aggregation with (aggregation='shared') "passed by reference"	"leafref" statement	How to map "passed by value"??

Figure 12: Mapping of Associations

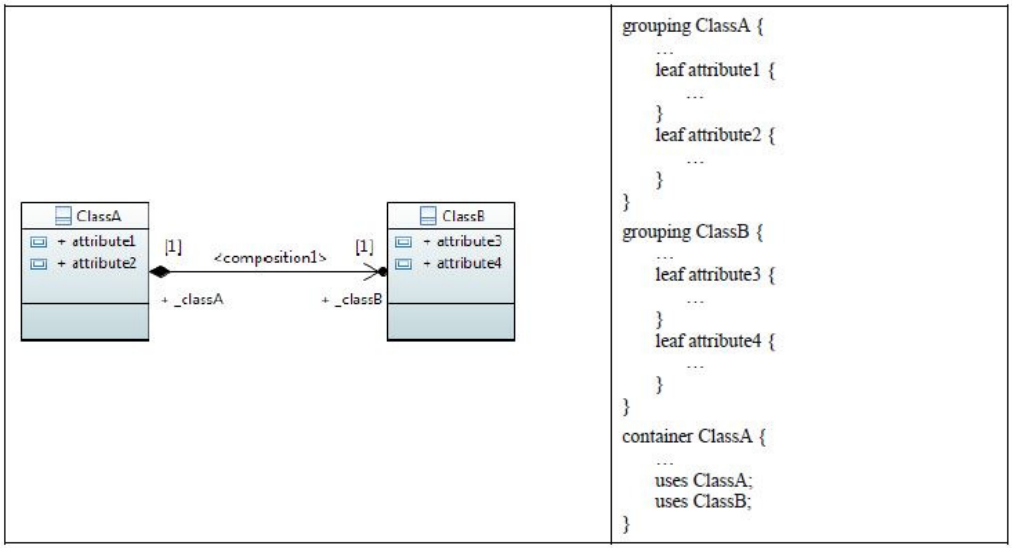
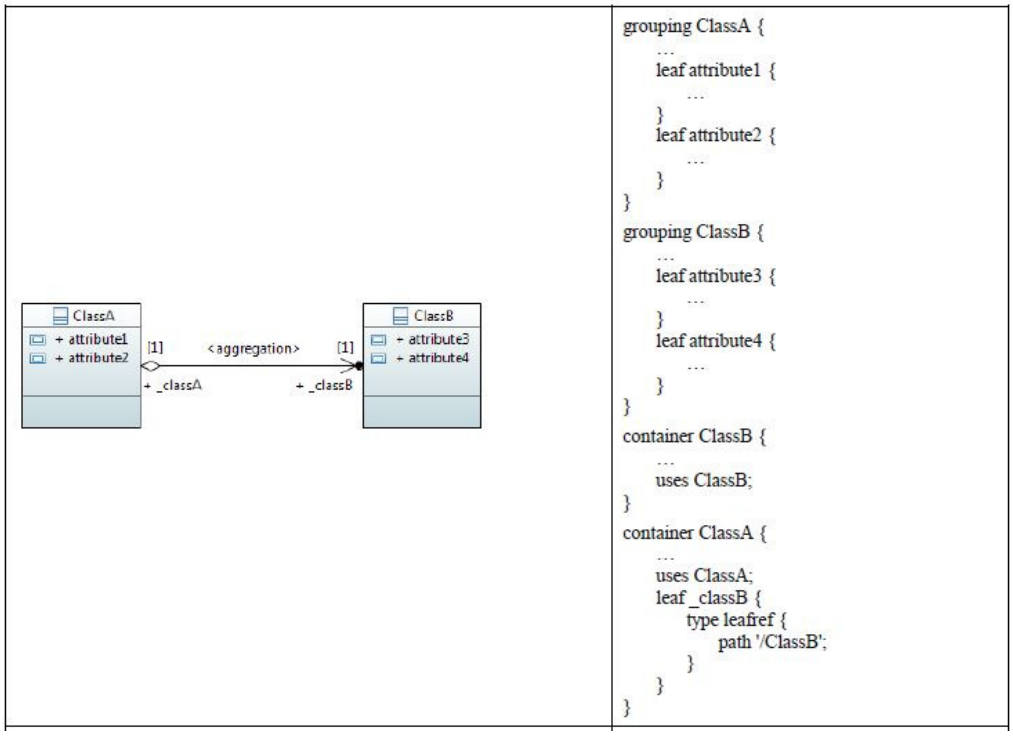
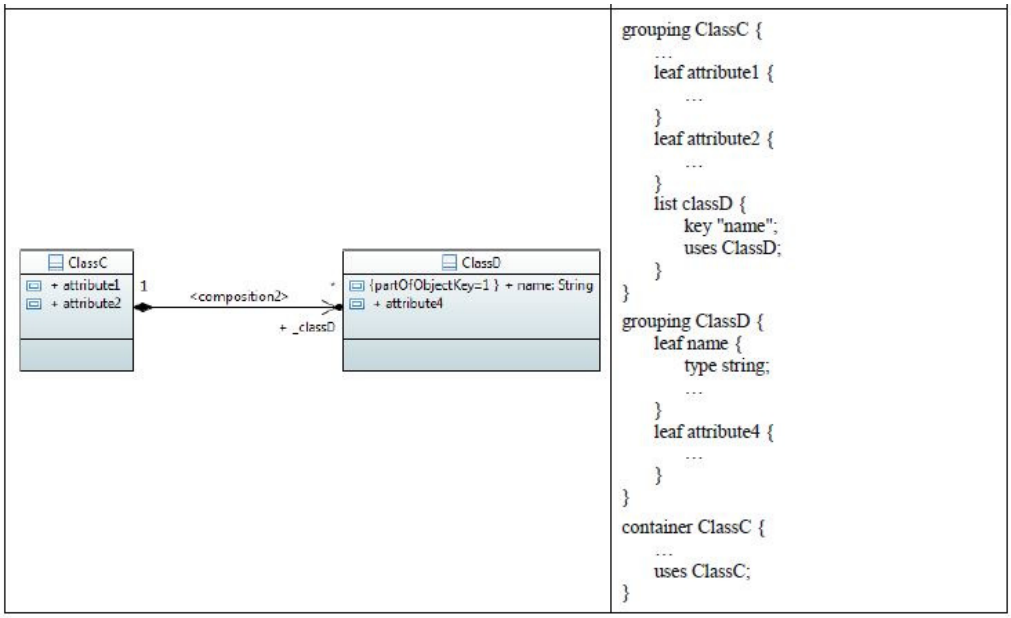


Figure 13: Association Mapping Example 1 (Available in PDF or HTML versions)



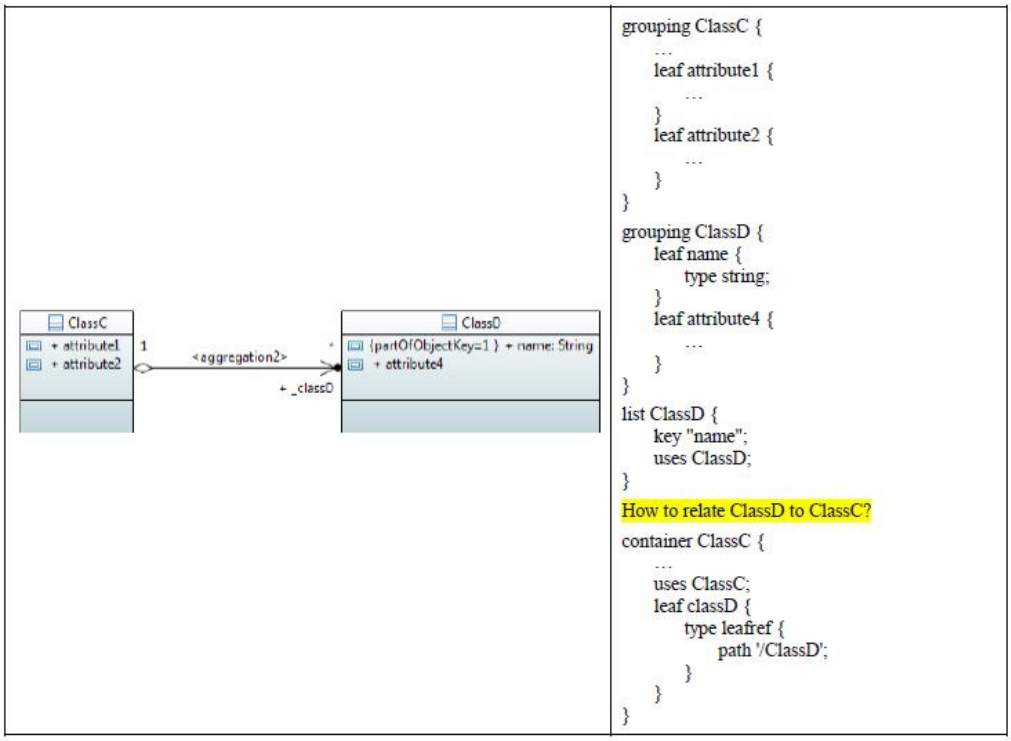

```
grouping ClassA {
    ...
    leaf attribute1 {
        ...
    }
    leaf attribute2 {
        ...
    }
}
grouping ClassB {
    ...
    leaf attribute3 {
        ...
    }
    leaf attribute4 {
        ...
    }
}
container ClassB {
    ...
    uses ClassB;
}
container ClassA {
    ...
    uses ClassA;
    leaf _classB {
        type leafref {
            path '/ClassB';
        }
    }
}
```

Figure 14: Association Mapping Example 2 (Available in PDF or HTML versions)




```
grouping ClassC {
    ...
    leaf attribute1 {
        ...
    }
    leaf attribute2 {
        ...
    }
    list classD {
        key "name";
        uses ClassD;
    }
}
grouping ClassD {
    leaf name {
        type string;
        ...
    }
    leaf attribute4 {
        ...
    }
}
container ClassC {
    ...
    uses ClassC;
}
```

Figure 15: Association Mapping Example 3 (Available in PDF or HTML versions)

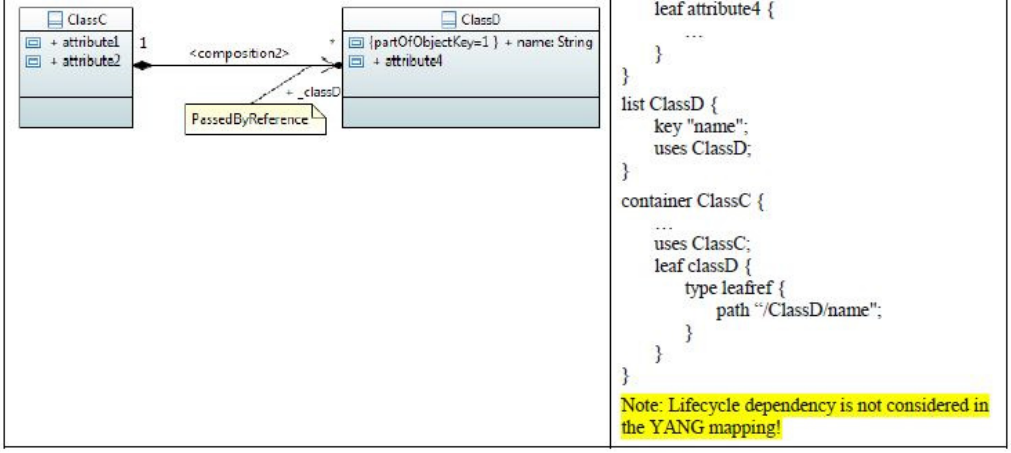



```
grouping ClassC {
    ...
    leaf attribute1 {
        ...
    }
    leaf attribute2 {
        ...
    }
}
grouping ClassD {
    leaf name {
        type string;
    }
    leaf attribute4 {
        ...
    }
}
list ClassD {
    key "name";
    uses ClassD;
}
```

How to relate ClassD to ClassC?

```
container ClassC {
    ...
    uses ClassC;
    leaf classD {
        type leafref {
            path '/ClassD';
        }
    }
}
```

Figure 16: Association Mapping Example 4 (Available in PDF or HTML versions)




```
grouping ClassC {
    ...
    leaf attribute1 {
        ...
    }
    leaf attribute2 {
        ...
    }
}
grouping ClassD {
    leaf name {
        type string;
    }
    leaf attribute4 {
        ...
    }
}
list ClassD {
    key "name";
    uses ClassD;
}
container ClassC {
    ...
    uses ClassC;
    leaf classD {
        type leafref {
            path "/ClassD/name";
        }
    }
}
```

Note: Lifecycle dependency is not considered in the YANG mapping!

Figure 17: Association Mapping Example 5 (Available in PDF or HTML versions)

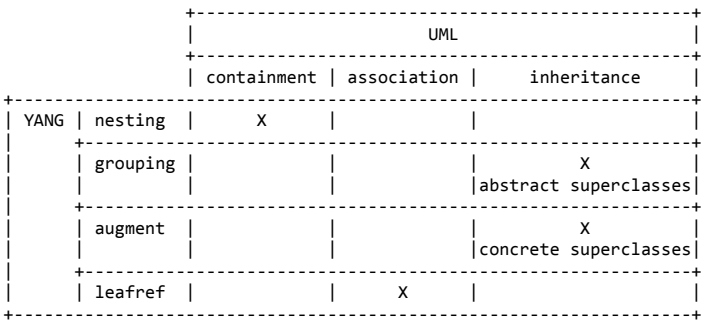


Figure 18: Association Mapping Summary

4.7 Mapping of Interfaces

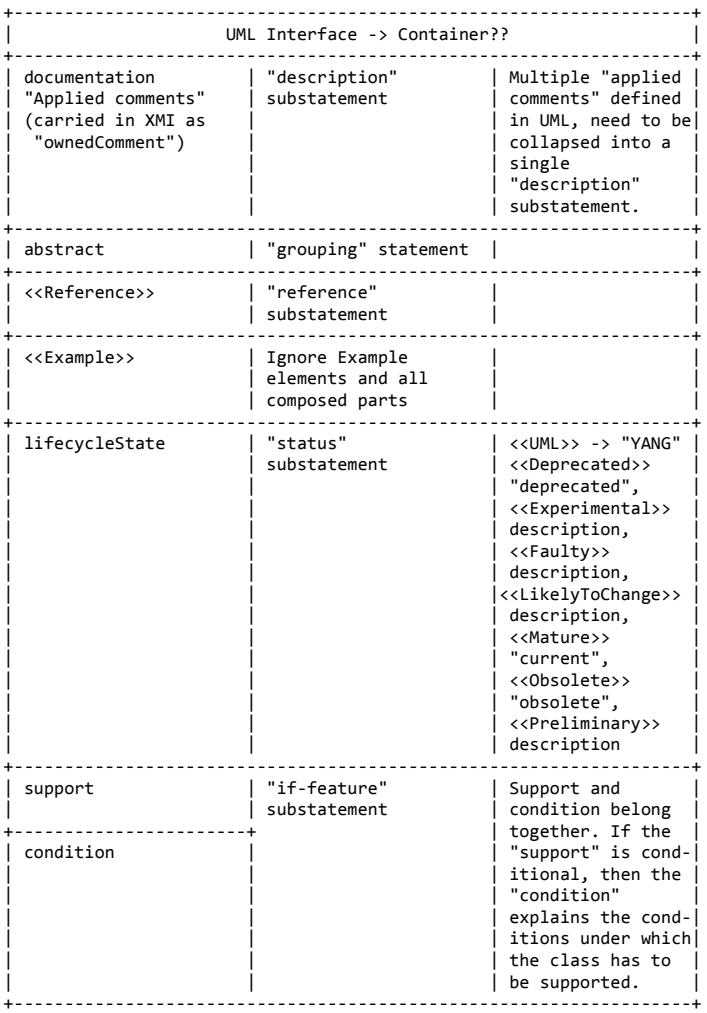
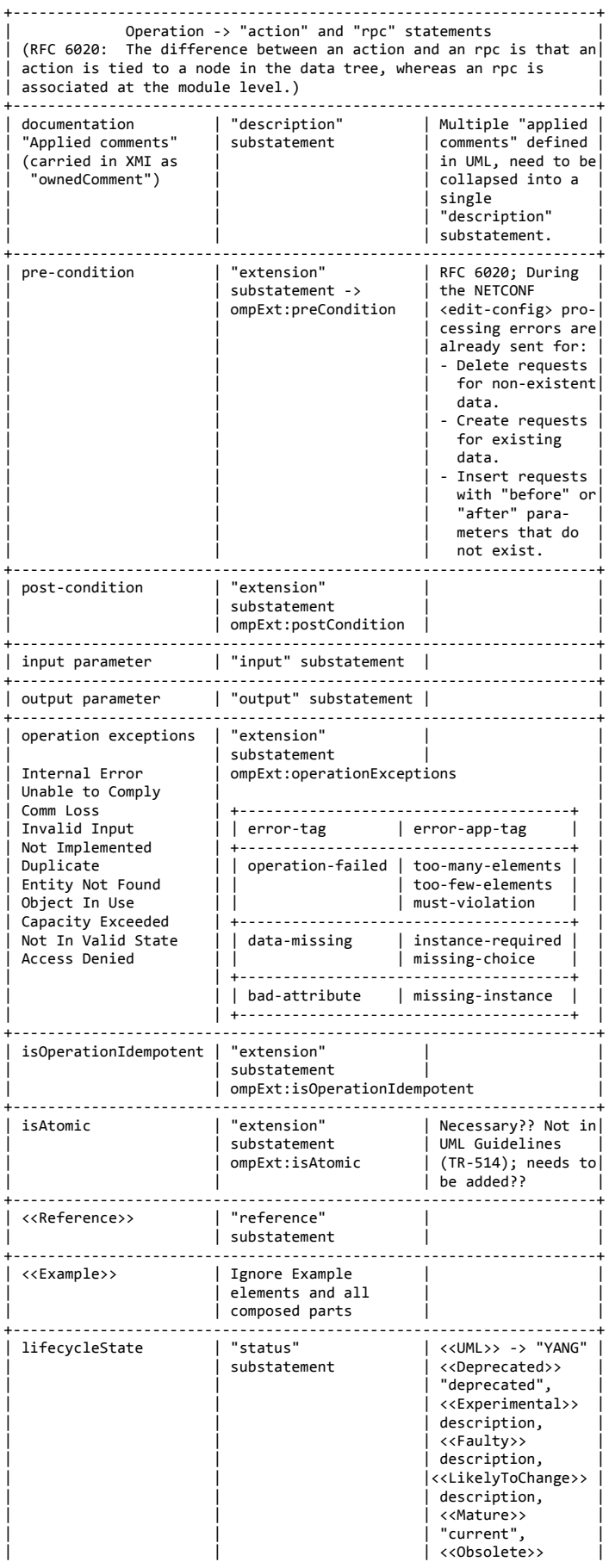


Figure 19: Mapping of Interfaces (grouping of operations

4.8 Mapping of Operations


Operation -> "action" and "rpc" statements
(RFC 6020: The difference between an action and an rpc is that an action is tied to a node in the data tree, whereas an rpc is associated at the module level.)
documentation "Applied comments" (carried in XMI as "ownedComment")
"description" substatement
Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
pre-condition
"extension" substatement -> ompExt:preCondition
RFC 6020; During the NETCONF <edit-config> processing errors are already sent for:
- Delete requests for non-existent data.
- Create requests for existing data.
- Insert requests with "before" or "after" parameters that do not exist.
post-condition
"extension" substatement ompExt:postCondition
input parameter
"input" substatement
output parameter
"output" substatement
operation exceptions
"extension" substatement ompExt:operationExceptions
Internal Error
Unable to Comply
Comm Loss
Invalid Input
Not Implemented
Duplicate
Entity Not Found
Object In Use
Capacity Exceeded
Not In Valid State
Access Denied
error-tag
error-app-tag
operation-failed
too-many-elements
too-few-elements
must-violation
data-missing
instance-required
missing-choice
bad-attribute
missing-instance
isOperationIdempotent
"extension" substatement ompExt:isOperationIdempotent
isAtomic
"extension" substatement ompExt:isAtomic
Necessary?? Not in UML Guidelines (TR-514); needs to be added??
<<Reference>>
"reference" substatement
<<Example>>
Ignore Example elements and all composed parts
lifecycleState
"status" substatement
<<UML>> -> "YANG"
<<Deprecated>> "deprecated",
<<Experimental>> description,
<<Faulty>> description,
<<LikelyToChange>> description,
<<Mature>> "current",
<<Obsolete>

```
|                       |                       | "obsolete",       |
|                       |                       | <<Preliminary>>   |
|                       |                       | description       |
+-----------------------+-----------------------+-------------------+
| support               | "if-feature"          | Support and       |
|                       | substatement          | condition belong  |
+-----------------------+                       | together. If the  |
| condition             |                       | "support" is cond-|
|                       |                       | itional, then the |
|                       |                       | "condition"       |
|                       |                       | explains the cond-|
|                       |                       | itions under which|
|                       |                       | the class has to  |
|                       |                       | be supported.     |
+-----------------------+-----------------------+-------------------+
```

Figure 20: Mapping of Operations

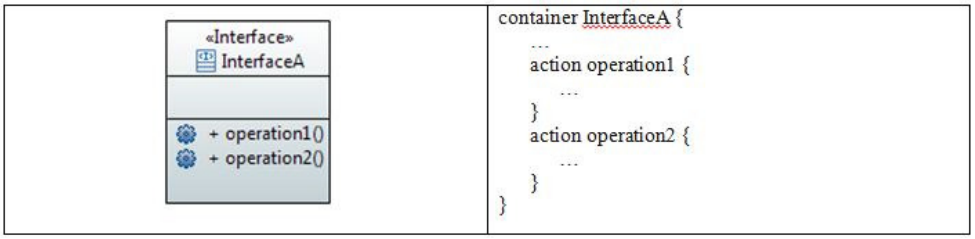


Figure 21: Operation Mapping Example (Available in PDF or HTML versions)

4.9 Mapping of Operation Parameters

Operation Parameters		
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
direction	"input" or "output" substatement	
type isOrdered multiplicity defaultValue valueRange	see mapping of attribute types (grouping, leaf, leaf-list, list, typedef, uses)	
passedByReference	if passedByReference = true -> type leafref {path "/<object>/<objectidentifier>"} if passedByReference = false -> either "list" statement (key property, multiple instances) or "container" statement(single instance)	Relevant only to attributes that have an object class defined as their type.
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description
support condition	"if-feature" substatement not defined for input and output substatements in YANG??	Support and condition belong together. If the "support" is conditional, then the "condition" explains the conditions under which the class has to be supported.
XOR	"choice" substatement	
error notification??	"must" substatement	
complex parameter	"uses" substatement	

Figure 22: Mapping of Operation Parameters

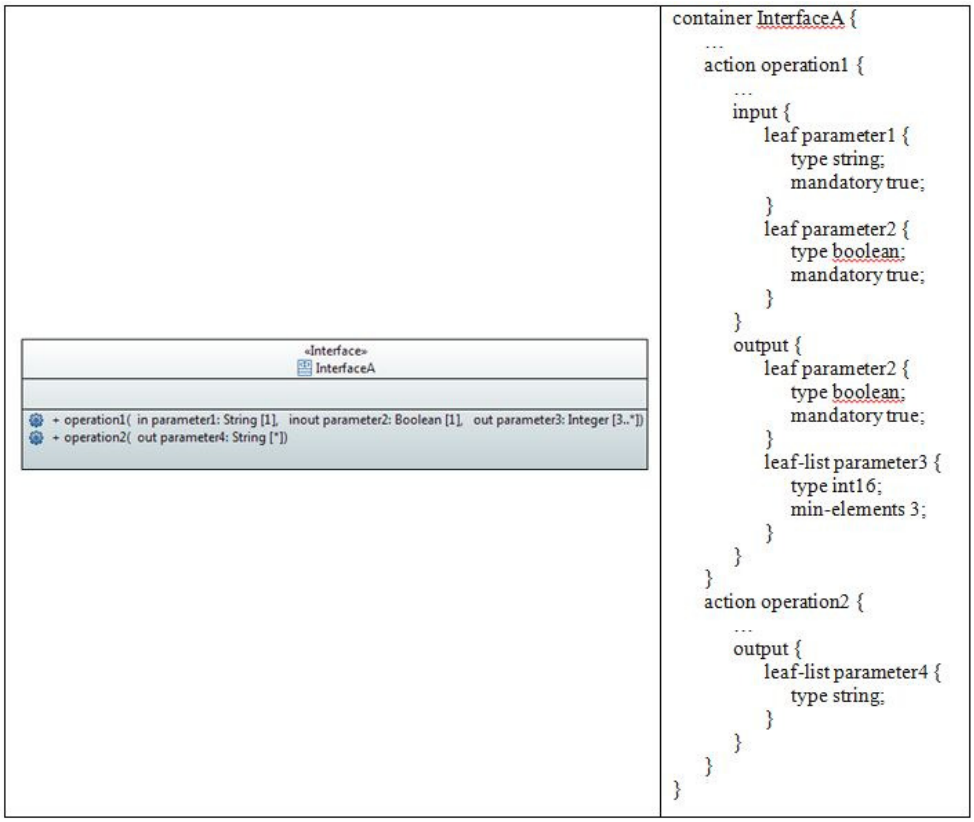
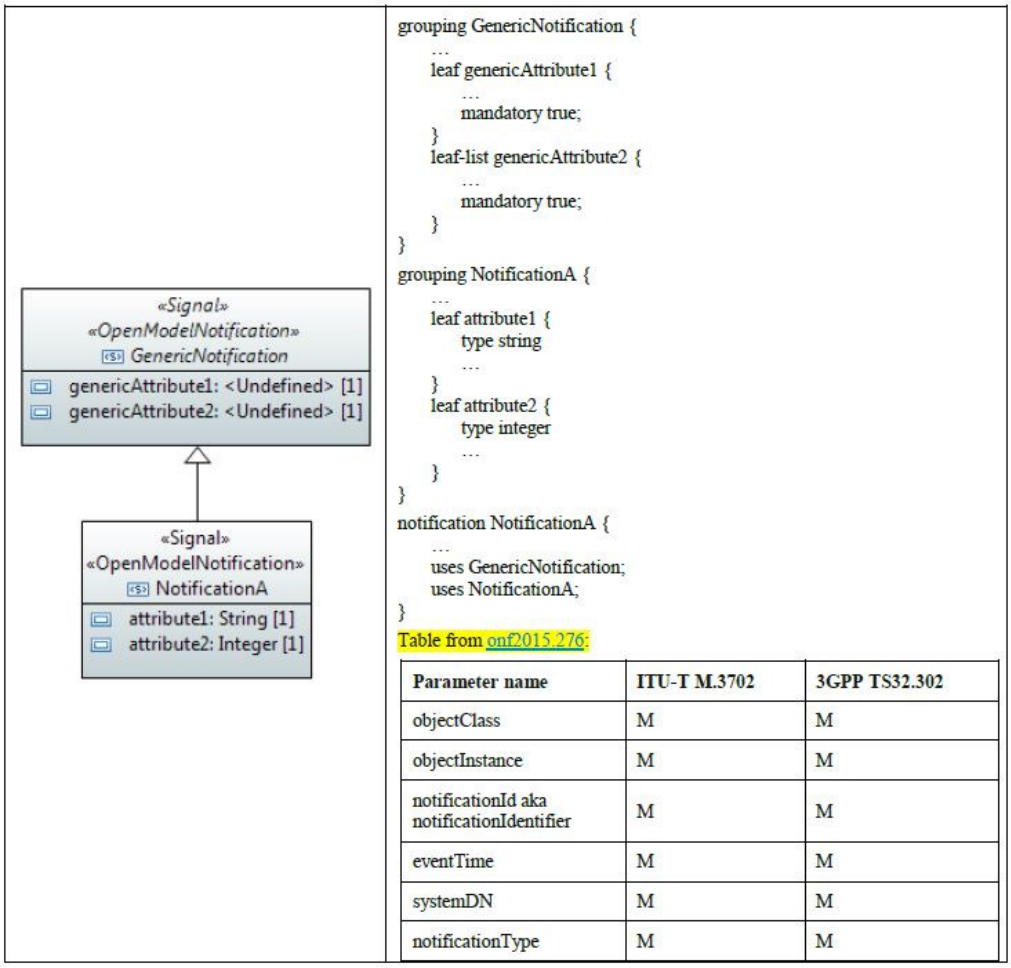


Figure 23: Parameter Mapping Example (Available in PDF or HTML versions)

4.10 Mapping of Notifications

Signal "grouping" statement -> "notification" statement		
documentation "Applied comments" (carried in XMI as "ownedComment")	"description" substatement	Multiple "applied comments" defined in UML, need to be collapsed into a single "description" substatement.
<<Reference>>	"reference" substatement	
<<Example>>	Ignore Example elements and all composed parts	
lifecycleState	"status" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description
support condition	"if-feature" substatement	Support and condition belong together. If the "support" is conditional, then the "condition" explains the conditions under which the class has to be supported.
Proxy Class XOR	"choice" substatement	
error notification??	"must" substatement	
attributes	see mapping of attribute types (grouping, leaf, leaf-list, container, list, typedef, uses)	
complex attribute	"uses" substatement	

Figure 24: Mapping of Notifications




```
grouping GenericNotification {
    ...
    leaf genericAttribute1 {
        ...
        mandatory true;
    }
    leaf-list genericAttribute2 {
        ...
        mandatory true;
    }
}
grouping NotificationA {
    ...
    leaf attribute1 {
        type string
        ...
    }
    leaf attribute2 {
        type integer
        ...
    }
}
notification NotificationA {
    ...
    uses GenericNotification;
    uses NotificationA;
}
```

Table from onf2015.276:

Parameter name	ITU-T M.3702	3GPP TS32.302
objectClass	M	M
objectInstance	M	M
notificationId aka notificationIdentifier	M	M
eventTime	M	M
systemDN	M	M
notificationType	M	M

Figure 25: Notification Mapping Example (Available in PDF or HTML versions)

4.11 Mapping of Lifecycle

UML Lifecycle		
lifecycle stereotypes	"status" substatement or "description" substatement	<<UML>> -> "YANG" <<Deprecated>> "deprecated", <<Experimental>> description, <<Faulty>> description, <<LikelyToChange>> description, <<Mature>> "current", <<Obsolete>> "obsolete", <<Preliminary>> description

Figure 26: Mapping of Lifecycle

5. Mapping Patterns

5.1 UML Recursion

As YANG defines hierarchical data store, any instances that need to store recursive containment will require translation. A mapping between object-oriented store and a hierarchical store is possible; however, there is more than one option:

- Reference Based Approach approach - have a flat list of objects, where the objects are linked into a hierarchy using references. An example of a two-way navigable approach is in RFC7223 [2].
- Assume some specific number of "recursions"; i.e., specify some default number of recursion levels, and define a configurable parameter to allow changing the number of levels.

5.1.1 Reference Based Approach

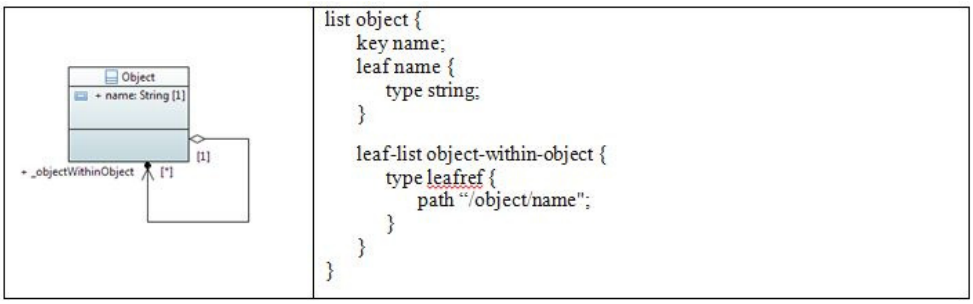


Figure 27: Recursion Mapping Example 1 (Available in PDF or HTML versions)

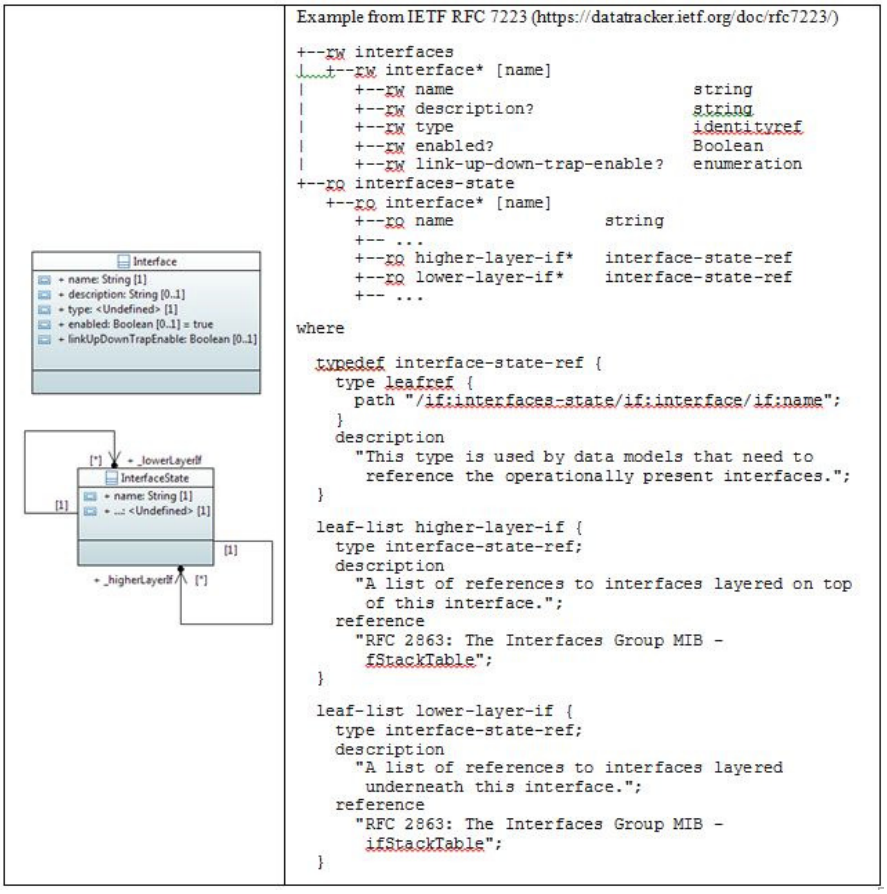


Figure 28: Recursion Mapping Example 2 (Available in PDF or HTML versions)

5.2 UML Conditional Pacs

Use the "presence" property of the container statement?

Note: An example of this usage is given in the "Data nodes for the operational state of IP on interfaces." within ietf-ip.yang RFC7277 [3].

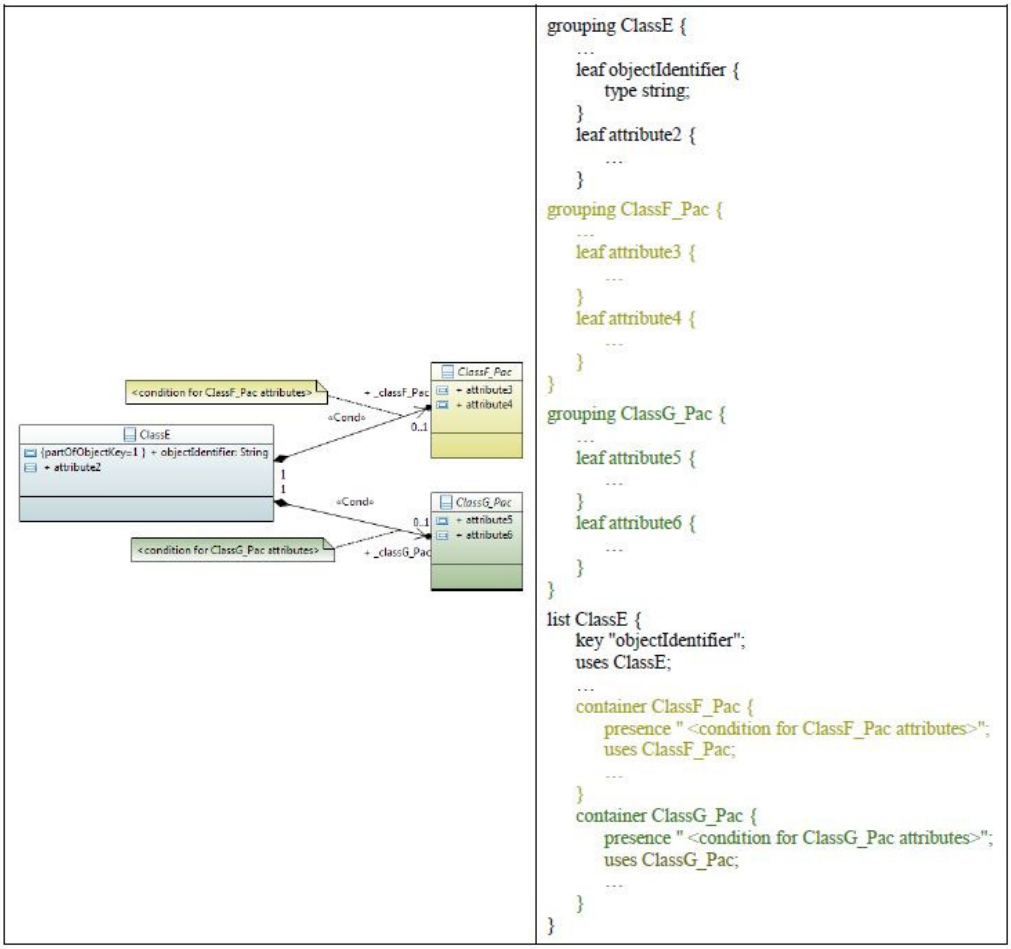
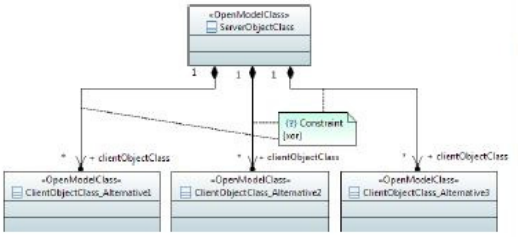


Figure 29: Mapping of Conditional Packages (Available in PDF or HTML versions)

5.3 XOR Relationship

The associations related by the "xor" constraint are mapped to the "choice" property of the container/list statement.

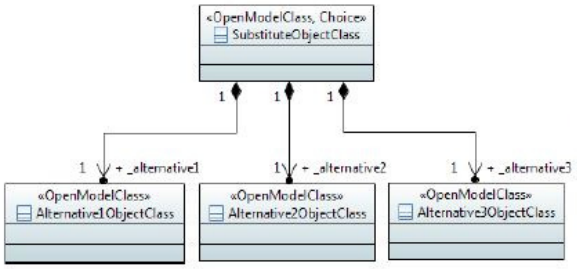



```
grouping ServerObjectClass {
    …
    }
grouping ClientObjectCLass_Alternative1 {
    …
}
grouping ClientObjectCLass_Alternative2 {
    …
}
grouping ClientObjectCLass_Alternative3 {
    …
}
list ClientObjectCLass_Alternative1 {
    key …;
    uses ClientObjectCLass_Alternative1;
    …
}
list ClientObjectCLass_Alternative2 {
    key …;
    uses ClientObjectCLass_Alternative2;
    …
}
list ClientObjectCLass_Alternative3 {
    key …;
    uses ClientObjectCLass_Alternative3;
    …
}
list ServerObjectClass {
    key …;
    …
    choice _clientObjectClass {
        case ClientObjectCLass_Alternative1 {
            leaf ClientObjectCLass_Alternative1 {
                type leafref {
                    path '/ ClientObjectCLass_Alternative1';
                }
            }
        }
        case ClientObjectCLass_Alternative2 {
            leaf ClientObjectCLass_Alternative2 {
                type leafref {
                    path '/ ClientObjectCLass_Alternative2';
                }
            }
        }
        case ClientObjectCLass_Alternative3 {
            leaf ClientObjectCLass_Alternative3 {
                type leafref {
                    path '/ ClientObjectCLass_Alternative3';
                }
            }
        }
    }
}
```

Figure 30: XOR Relationship Mapping Example (Available in PDF or HTML versions)

5.4 Choice Stereotype

The choice sterotype can be associated in UML to a class or a data type. The class or a data type which is annotated with the choice sterotype represents one of a set of classes/data types. This pattern is mapped to the "choice" property of the container/list/grouping statement.




```
grouping SubstituteObjectClass {
    …
    }
grouping Alternative1ObjectClass {
    …
}
grouping Alternative2ObjectClass {
    …
}
grouping Alternative3ObjectClass {
    …
}
container Alternative1ObjectClass {
    uses Alternative1ObjectClass;
    …
}
container Alternative2ObjectClass {
    uses Alternative2ObjectClass;
    …
}
container Alternative3ObjectClass {
    uses Alternative3ObjectClass;
    …
}
list SubstituteObjectClass {
    key …;
    …
    choice __alternative {
        case Alternative1ObjectClass {
            leaf Alternative1ObjectClass {
                type leafref {
                    path '/Alternative1ObjectClass';
                }
            }
        }
        case Alternative2ObjectClass {
            leaf Alternative2ObjectClass {
                type leafref {
                    path '/Alternative2ObjectClass';
                }
            }
        }
        case Alternative3ObjectClass {
            leaf Alternative3ObjectClass {
                type leafref {
                    path '/Alternative3ObjectClass';
                }
            }
        }
    }
}
```

Figure 31: Choice Stereotype Example (Available in PDF or HTML versions)

5.5 Mapping of UML Support and Condition

The UML Modeling Guidelines [7] define support and condition for all UML artifacts (M - Mandatory, O - Optional, C - Conditional, CM - Conditional-Mandatory, CO - Conditional-Optional). Support qualifies the support of the artifact at the management interface. Condition contains the condition for the condition-related support qualifiers.

M - Mandatory maps to the "mandatory" substatement in choice and leaf or to the "min-elements" substatement in leaf-list and list.

O - Optional need not be mapped since the per default the "mandatory" and "min-elements" substaments define optional.

All conditional UML support qualifiers are mapped to the "if-feature" substatement.

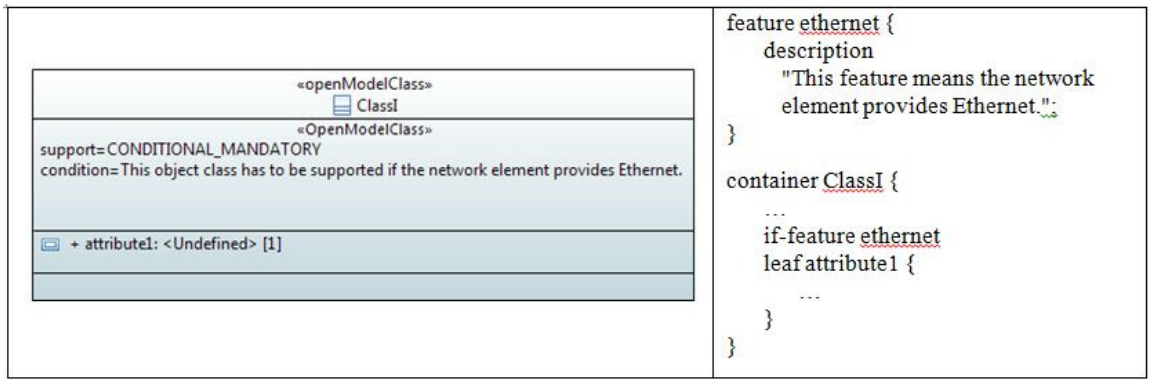


Figure 32: Support and Condition Mapping Example (Available in PDF or HTML versions)

6. Mapping Basics

6.1 UML-YANG or XMI-YANG

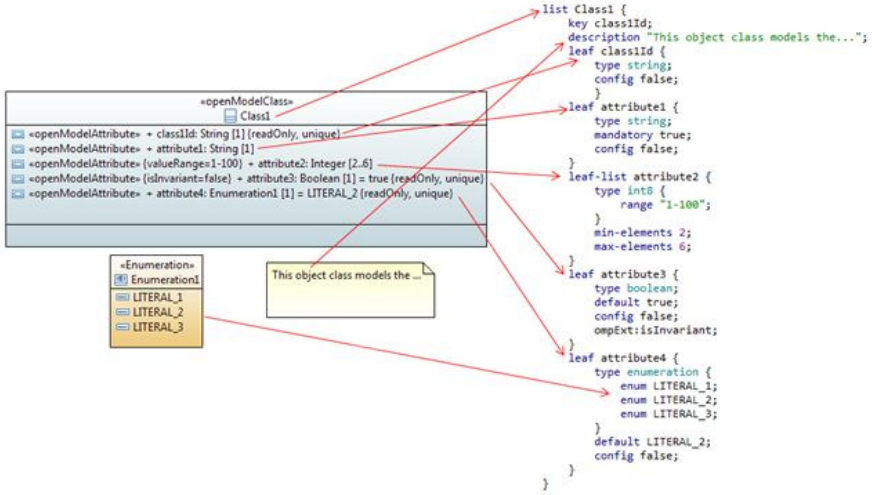


Figure 33: Example UML to YANG Mapping (Available in PDF or HTML versions)

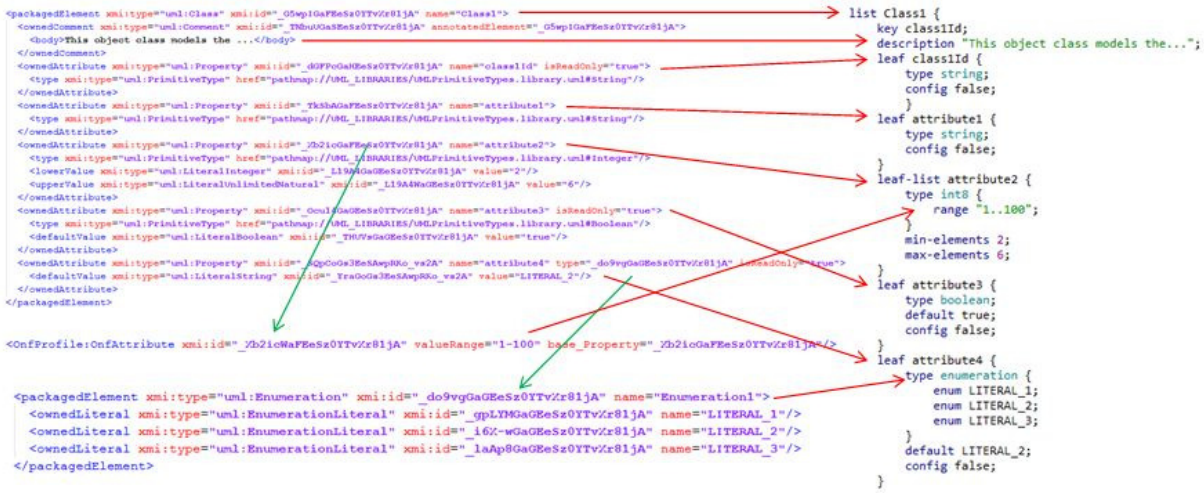


Figure 34: Example XMI (Papyrus) to YANG Mapping (Available in PDF or HTML versions)

7. Acknowledgements

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

This document defines defines guidelines for translation of data modeled with UML to YANG. As such, it doesn't contribute any new security issues beyond those discussed in Sec. 16 of RFC6020 [1].

10. Informative References

[1] Bjorklund, M., Ed., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)", RFC 6020, DOI 10.17487/RFC6020, October 2010, <http://www.rfc-editor.org/info/rfc6020>.

[2] Bjorklund, M., "A YANG Data Model for Interface Management", RFC 7223, DOI 10.17487/RFC7223, May 2014, <http://www.rfc-editor.org/info/rfc7223>.

[3] Bjorklund, M., "A YANG Data Model for IP Management", RFC 7277, DOI 10.17487/RFC7277, June 2014, <http://www.rfc-editor.org/info/rfc7277>.

[4] Bjorklund, M., Ed., "The YANG 1.1 Data Modeling Language", RFC 7950, DOI 10.17487/RFC7950, August 2016, <http://www.rfc-editor.org/info/rfc7950>.

[5] Lam, H., Varma, E., Doolan, P., Davis, N., Zeuner, B., Betts, M., Busi, I., Mansfield, S., Vilata, R., and V. Lopezalvarez, "Usage of IM for network topology to support TE Topology YANG Module Development", Internet-Draft draft-lam-teas-usage-info-model-net-topology-03 (work in progress), May 2016.

[6] OMG, "Unified Modeling Language (UML)", 2011, <http://www.omg.org/spec/UML/2.4/>.

[7] OMG, "ONF TR-514 v1.0 UML Modeling Guidelines", 2015, <https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/UML_Modeling_Guidelines_V1.0.pdf>.

Authors' Addresses

Scott Mansfield (editor)
Ericsson Inc.

USA
Phone: +1 724 931 9316
EMail: scott.mansfield@ericsson.com

Bernd Zeuner
Deutsche Telekom AG
Heinrich-Hertz-Str, 3-7
Darmstadt, 64295
Germany
Phone: +49 6151 58-12086
EMail: b.zeuner@telekom.de

Nigel Davis
Ciena

United Kingdom
EMail: ndavis@ciena.com

Yun Xiang
Fiberhome

China
EMail: yunxig@fiberhome.com.cn

Yuji Tochio
Fujitsu

Japan
EMail: tochio@jp.fujitsu.com

Hing-Kam Lam
Nokia

USA
Phone: +1 732 331 3476
EMail: kam.lam@nokia.com

Eve Varma
Nokia

USA
EMail: eve.varma@nokia.com