Systems and Virtualization Management 2007

October 23, 24

UML Modeling for CIM

UML Modeling subgroup in DMTF Architecture WG

Andy Maier, IBM (maiera@de.ibm.com)

Robert Kieninger, IBM (kieningr@de.ibm.com)

Last updated: 10/12/2007

Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change. The Standard Specifications remain the normative reference for all information.
- For additional information, see the Distributed Management Task Force (DMTF) Web site.

Content of this session

- Motivation for using UML for CIM modeling
- Overview on the proposed CIM mapping to UML
- Strategic direction with UML in CIM
- Simple example with OCL demo

Why CIM modeling using UML

- CIM modeling is currently done by writing :
  - CIM MOF (the normative definition)
  - Accompanying Visio diagrams
  - Older concept: White Papers - some overlap with mgmt profiles
  - New concept: Management Profiles

- There are no modeling tools that support CIM MOF

- UML is a well established modeling approach
  - Broad tools support (both UML 1.x and 2.0)
  - Open Source tooling environments (e.g. Eclipse)

- UML provides capabilities beyond just CIM MOF and current Visio diagrams
  - Automatic consistency between class definition and class diagram
  - Additional types of diagrams (e.g. Object, Interaction, State, Use Case)

-> Idea: Use UML to do CIM modeling

What's needed

- "Mapping" between CIM MOF and UML
  - Must work in both directions -> round trip engineering
  - At least for the information that is in CIM MOF

- UML terminology: "UML Profile for CIM"
  - A document that defines the mapping (DMTF spec DSP0219)
  - A set of files (UML profiles and UML type libraries)
  - Defines how to represent CIM models (i.e. CIM Schema) in UML
  - Defines how to import CIM MOF into UML and export again to CIM MOF
  - Mapping is model agnostic, i.e. it allows to represent any CIM Schema / Model in UML
  - Note: Do not confuse "DMTF Mgmt Profile" with "UML Profile for CIM"

- UML is often seen as a drawing tool

  ... but that is only one aspect of it....
  - UML is defined very precisely in terms of UML metaclasses (e.g. Class, Property)
  - This allows UML to be used as a representation of a model, not just as a drawing tool.

File representation of UML

- CIM MOF has a clearly defined file representation, what is it for UML ?
- OMG defines a representation format for UML, called XMI
- Today (UML 2.0), XMI is able to represent the model aspects of UML, but not (yet) the diagram aspects
- UML tools tend to use their own storage formats in order to support their extensions
- -> XMI is positioned as an interchange format, not as "the" representation format

Early tools support

- Some tools have implemented earlier versions of a UML Profile for CIM:
- "ECUTE" plugin for Rational Rose / Rational Software Architect (IBM)
  - Part of SBLIM Open Source Project www.sourceforge.net/projects/sblim
  - Allows to import/export CIM MOF from/to these UML tools
  - Supports round trip engineering
- "Model Wizard" (Cisco)
  - aka "CIM Wizard"
  - http://modelwizard.sourceforge.net
  - Allows to easily learn about the CIM Schema and pick areas needed for your model

Strategic Direction

- Modeling in UML starts out as an alternative to defining CIM models in CIM MOF
  - Used because it provides additional value (e.g. diagrams, visual modeling)
  - Supports round trip engineering, so CIM MOF can stay the normative definition
  - Requires unambiguous bidirectional mapping between UML and CIM MOF
- Because it is a precise definition of a CIM Model, we have the option to transition the normative definition of the CIM Schema from CIM MOF to UML
  - Requires that some more experience with using UML for CIM is gained
    - Experience with early tools very positive, but they used an early mapping
  - Requires that some issues around XMI are resolved
    - Representation format vs. just exchange format
    - Need to gain experience with using XMI across tools
    - How to separate normative stuff in XMI (e.g. model) from additions (e.g. use cases)
    - How to write DMTF CRs in terms of XMI
  - Requires that DMTF embraces UML as the normative definition for CIM
- UML option is on the table, decision on its usage in DMTF is to be made

UML Modeling subgroup

- "UML Modeling" is a subgroup under the DMTF Architecture WG
- Consists of both DMTF and OMG members, under the DMTF/OMG alliance

- Work products:
  - DMTF Spec DSP0219 "UML Profile for CIM"
    - Based upon UML 2.1
    - Based upon CIM V2 (i.e. CIM Infrastructure Spec 2.3 with some additional CRs)
  - Intention is that OMG embraces that spec, resulting in an OMG document
    - through the OMG Fast Track process (ca. 3 months to get through)
  - Documented set of issues against CIM and UML

- Current status:
  - DSP0219 V0.5.0 was publicly released as a Work in Progress in 9/2006
  - Current WG internal version V0.5.10 is fairly complete
  - All known issues against UML have been raised as part of the UML 2.1 review process
    - A few more should be expected to come up as remaining TBDs are discussed
    - most of them expected to be addressed in UML 2.2
    - UML 2.2 is not in time for V1.0 of DSP0219, but no real show stoppers in sight

CIM Profile for UML - Overview

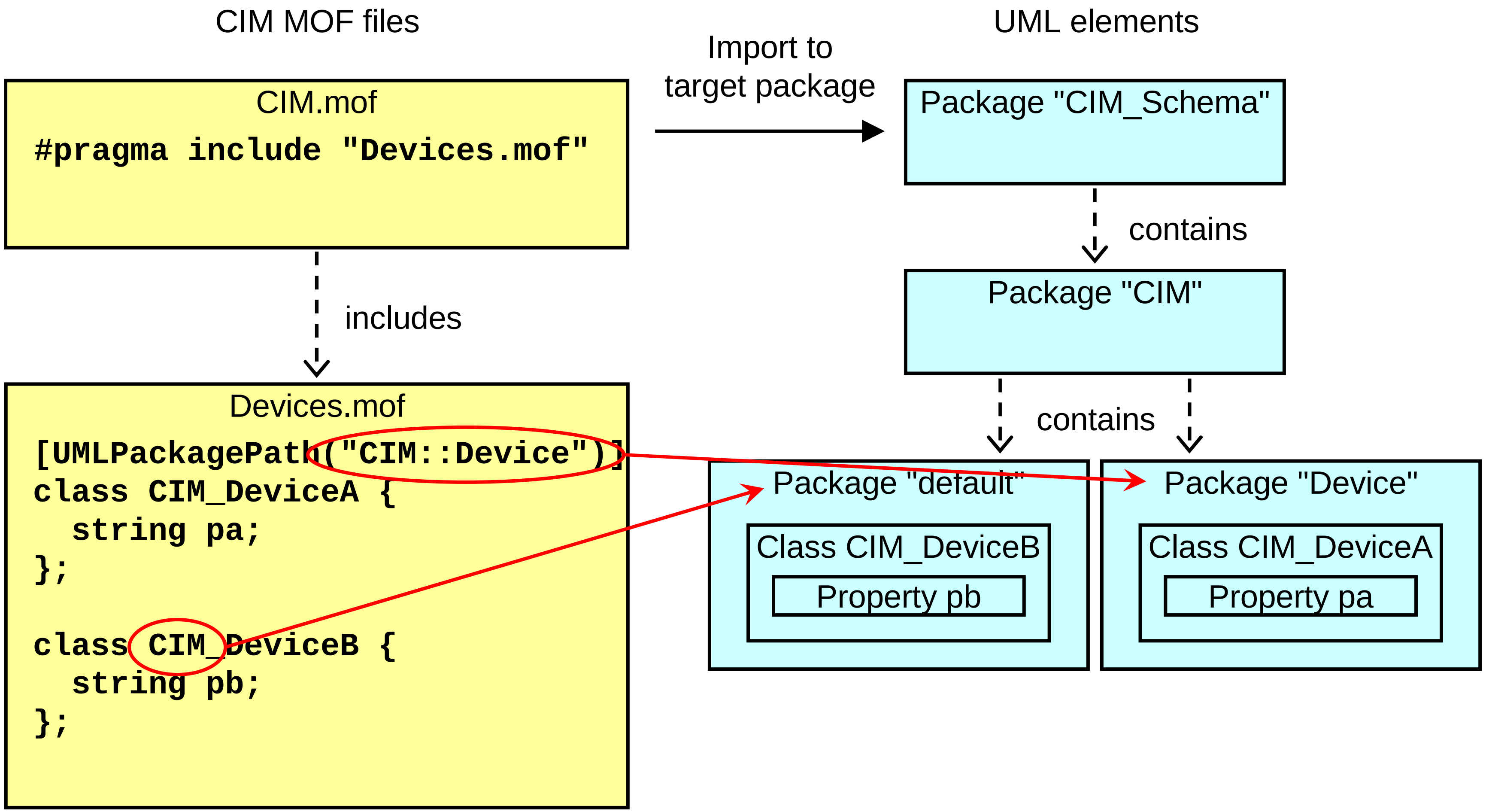
CIM element	UML construct
Class	*Class* metaclass
Indication	*Class* metaclass
Association	*AssociationClass* metaclass
Reference	*Property* metaclass (used as association end)
Property	*Property* metaclass (used as normal attribute)
Method	*Operation* metaclass
Parameter	*Parameter* metaclass
Schema	Top level package with stereotype CIM_Schema
Instance	*InstanceSpecification* metaclass instances
Datatype	*PrimitiveType* metaclass instances
Qualifier Type	*Stereotype* metaclasses and native UML constructs
UmlPackagePath Qualifier	Package hierarchy
OCL Constraint Qualifiers	*Constraint* metaclass
Trigger	not mapped (not used in CIM)
Namespace	not mapped (it is a runtime artifact in CIM servers, not a modeling element)

Bottom line: Very natural and straight forward in most areas

CIM Profile for UML - Datatypes

CIM datatype	UML construct
uint8	DataType instance named "uint8"
sint8	DataType instance named "sint8"
uint16	DataType instance named "uint16"
sint16	DataType instance named "sint16"
uint32	DataType instance named "uint32"
sint32	DataType instance named "sint32"
uint64	DataType instance named "uint64"
sint64	DataType instance named "sint64"
string	DataType instance named "string"
string [] (with Octetstring)	DataType instance named "octetstring“ with *lowerValue* / *upperValue* indicating the array
uint8 [] (with Octetstring)	DataType instance named "octetstring"
boolean	DataType instance named "boolean"
real32	DataType instance named "real32"
real64	DataType instance named "real64"
datetime	DataType instance named "datetime"
<classname> ref	UML association end, typed to the referenced class
char16	DataType instance named "char16"
arrays	CIM datatype without the array is mapped as defined in this table. The array characteristic is represented in the *lowerValue* and *upperValue* attributes (of the UML metaclasses owning the array)

Example for Package Path Mapping



CIM MOF files

UML elements

Import to target package

CIM.mof

```
#pragma include "Devices.mof"
```

includes

Devices.mof

```
[UMLPackagePath("CIM::Device")]
class CIM_DeviceA {
  string pa;
};

class CIM_DeviceB {
  string pb;
};
```

Package "CIM_Schema"

contains

Package "CIM"

contains

Package "default"

Class CIM_DeviceB

Property pb

Package "Device"

Class CIM_DeviceA

Property pa

Note: UMLPackagePath qualifier was introduced with DMTFCR01592 / ARCHCR00037 and has been applied to the CIM Schema in 2.13

Two Kinds of Qualifier Types

- **Well-known Qualifiers**
  - Subset of the DMTF defined qualifier types
  - Mapped to specific UML constructs
  - Example: *Abstract* qualifier is mapped to *isAbstract* attribute in UML *Class* metaclass

- **Generically mapped Qualifiers**
  - Remaining DMTF defined qualifier types, plus vendor defined
  - Mapped to UML Stereotypes through a generic mapping rule
  - Supports qualifiers unknown at the time of writing the mapping spec

-> Set of qualifier types remains extensible, as it is in CIM MOF

Mapping of Well-known Qualifier Types (1)

CIM Qualifier Type					UML Construct
Name	**Datatype**	**Default**	**Flavors**	**Scopes**	
Abstract	boolean	FALSE	Restricted	Class Association Indication	*isAbstract* attribute of the *Class* metaclass
Association	boolean	FALSE	ToSubclass DisableOverride	Association	*AssociationClass* with stereotype *CIM_Association*
Aggregation	boolean	FALSE	ToSubclass DisableOverride	Association	*None/Shared/Composite* values of *aggregation* attribute on the association end
Composition	boolean	FALSE	ToSubclass DisableOverride	Association	
Aggregate	boolean	FALSE	ToSubclass DisableOverride	Reference	determines on which association end the *aggregation* attribute is set
ArrayType	string	"Bag"	ToSubclass DisableOverride	Property Parameter	(1) *isOrdered* attribute of the *Property* or *Parameter* metaclasses (2) Also mapped generically to represent all three values
Deprecated	string array	NULL	Restricted	any	"Marker" stereotype
Description	string	NULL	ToSubclass EnableOverride	any	*Comment* metaclass associated to the qualified element, with tools specific way to mark the comment as description, e.g. *Documentation* stereotype for RSA

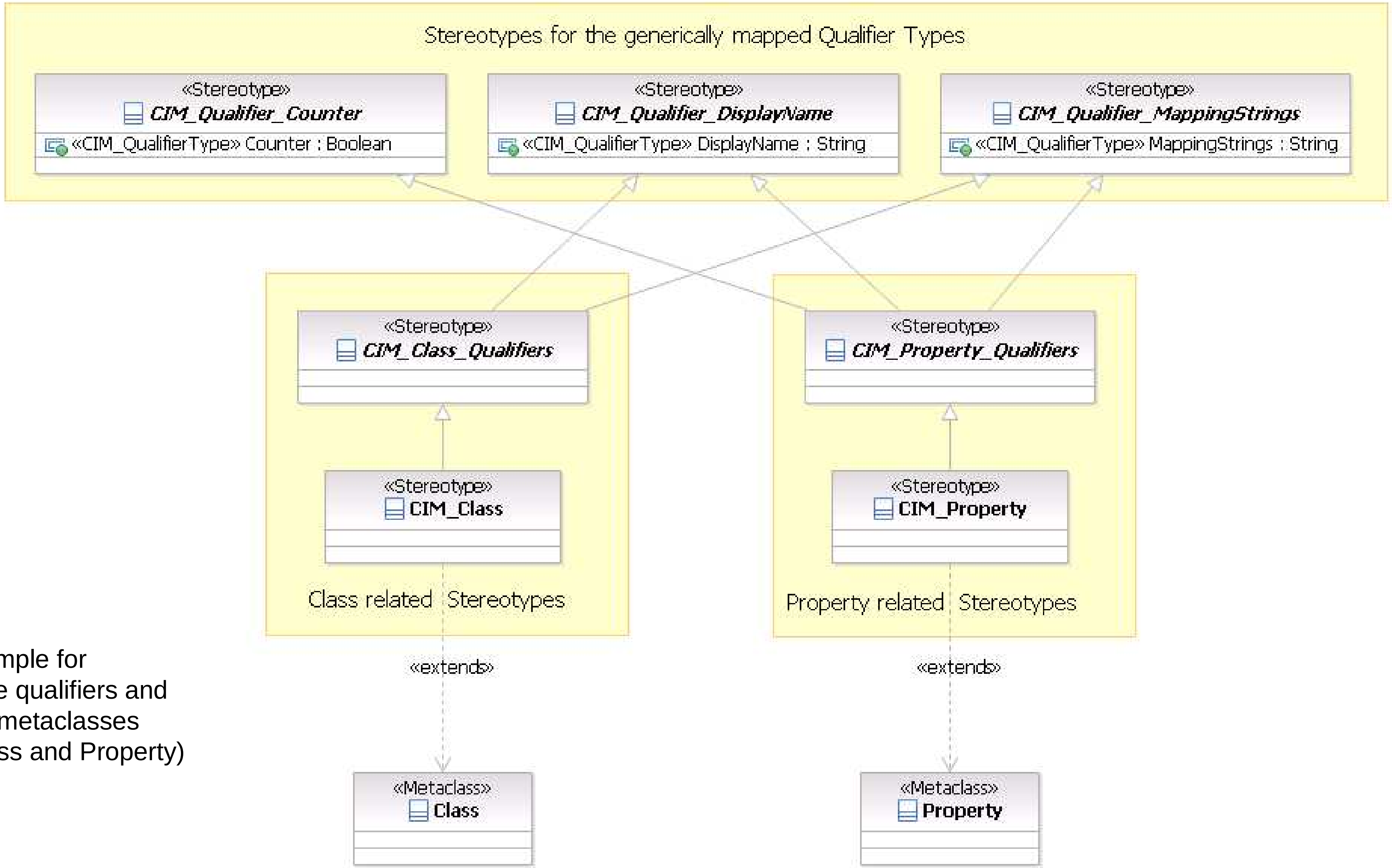
Mapping of Well-known Qualifier Types (2)

CIM Qualifier Type					UML Construct
Name	**Datatype**	**Default**	**Flavors**	**Scopes**	
Experimental	boolean	FALSE	Restricted	any	"Marker" stereotype
Indication	boolean	FALSE	ToSubclass DisableOverride	Class Indication	*Class* with stereotype *CIM_Indication*
In	boolean	TRUE	ToSubclass DisableOverride	Parameter	*direction* of the *Parameter* metaclass
Out	boolean	FALSE	ToSubclass DisableOverride	Parameter	
Key	boolean	FALSE	ToSubclass DisableOverride	Property Reference	"Marker" stereotype
Max	uint32	NULL	ToSubclass EnableOverride	Reference	*upperValue* of the *Property* metaclass
Min	uint32	0	ToSubclass EnableOverride	Reference	*lowerValue* of the *Property* metaclass

Mapping of Well-known Qualifier Types (3)

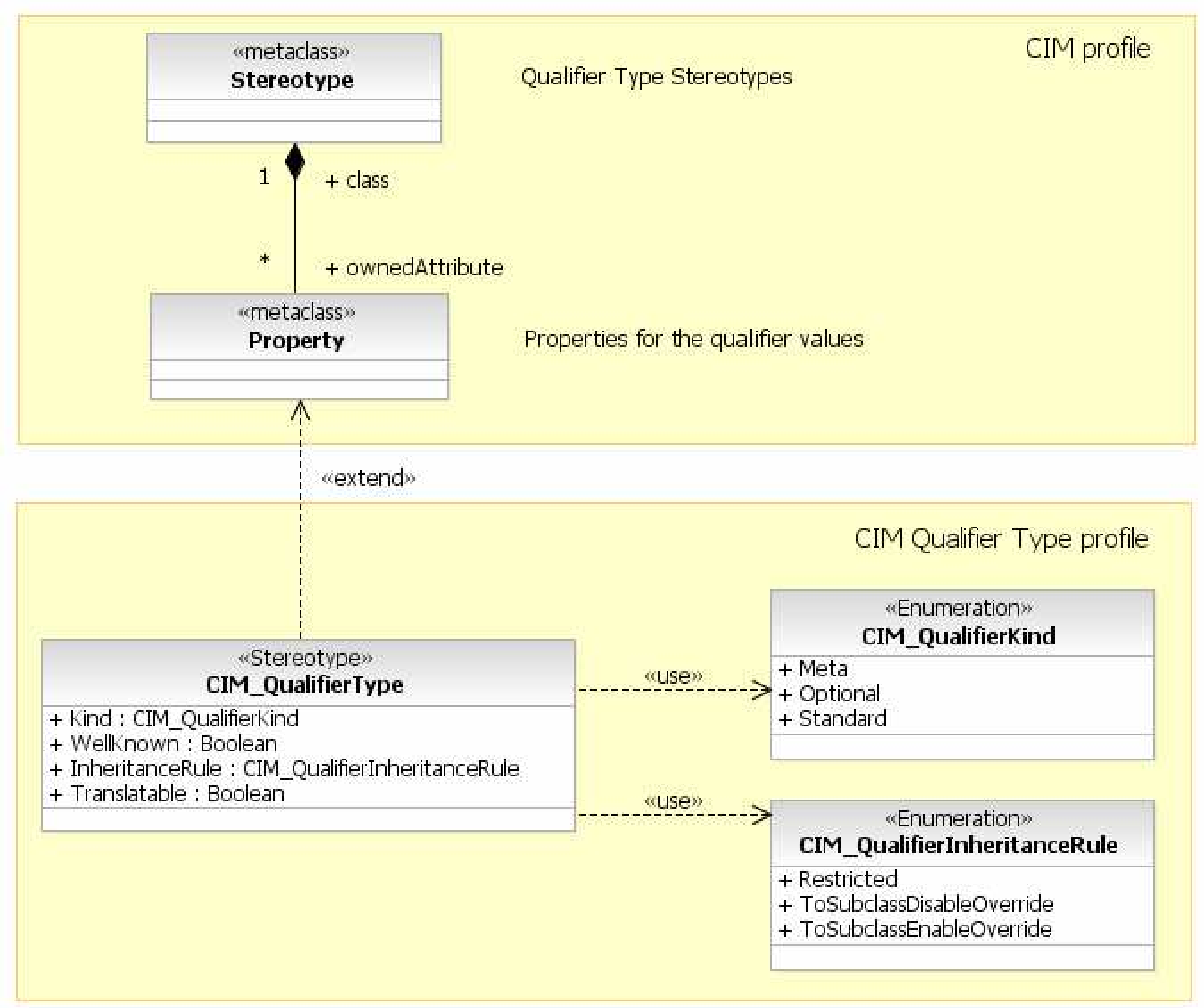
CIM Qualifier Type					UML Construct
Name	Datatype	Default	Flavors	Scopes	
ClassConstraint, MethodConstraint, PropertyConstraint	string array	NULL	ToSubclass EnableOverride	Class Association Indication Method	"inv:" constraints: *ownedRule* association of the *Class* metaclass "body:" constraints: *ownedRule* association of the *Operation* metaclass "pre:" constraints: *precondition* association of the *Operation* metaclass "post:" constraints: *postcondition* association of the *Operation* metaclass
Octetstring	boolean	FALSE	ToSubclass DisableOverride	Property Method Parameter	*PrimitiveType* "octetstring" or array thereof
Override	string	NULL	Restricted	Property Reference Method	*redefinedProperty* or *redefinedOperation* of the *Property* or *Operation* metaclasses
Static	boolean	FALSE	ToSubclass DisableOverride	Property Method	*isStatic* of the *Property* or *Operation* metaclasses
Terminal	boolean	FALSE	ToSubclass EnableOverride	Class Association Indication	*isLeaf* of the *Class* metaclass
Write	boolean	FALSE	ToSubclass EnableOverride	Property	*isReadOnly* of the *Property* metaclass

Generic Mapping of Qualifier Types (1)


Stereotypes for the generically mapped Qualifier Types
«Stereotype»
CIM_Qualifier_Counter
«CIM_QualifierType» Counter : Boolean
«Stereotype»
CIM_Qualifier_DisplayName
«CIM_QualifierType» DisplayName : String
«Stereotype»
CIM_Qualifier_MappingStrings
«CIM_QualifierType» MappingStrings : String
«Stereotype»
CIM_Class_Qualifiers
«Stereotype»
CIM_Property_Qualifiers
«Stereotype»
CIM_Class
«Stereotype»
CIM_Property
Class related Stereotypes
Property related Stereotypes
«extends»
«extends»
«Metaclass»
Class
«Metaclass»
Property

Example for
three qualifiers and
two metaclasses
(Class and Property)

Generic Mapping of Qualifier Types (2)

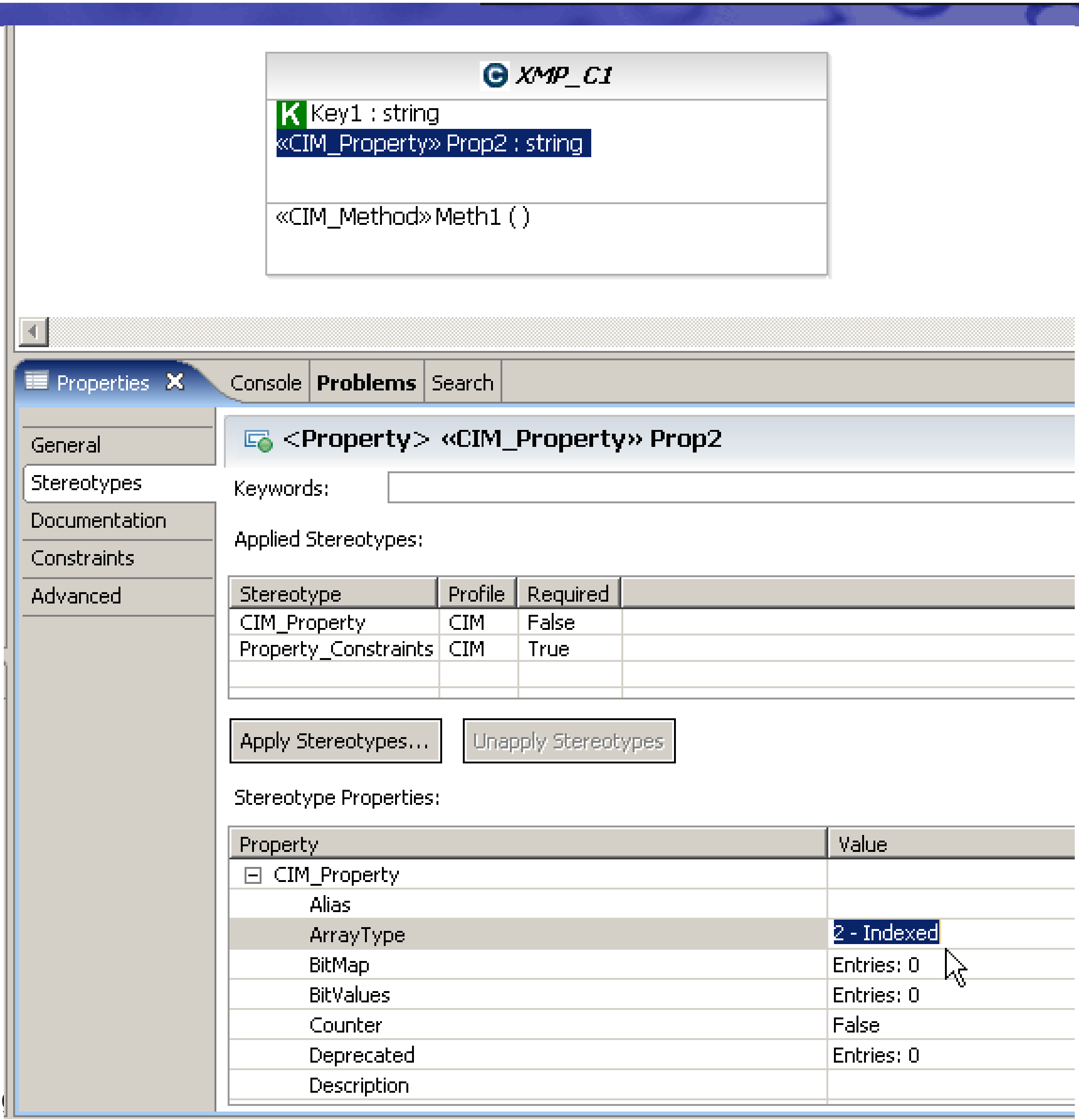


Simple Example
(CIM MOF)

```
[Abstract, Version("2.14.0")]
class C1 {

    [Key, MaxLen(256)]
    uint32 Key1;

    [ArrayType("Indexed")]
    string Prop2 [];

    [Units("Seconds")]
    uint32 Meth1 (
        [In, MaxValue(16000)]
        uint16 Parm1);
};
```

-> There are two errors in this MOF ...

Simple Example
(UML Tool)



XMP_C1

K Key1 : string

«CIM_Property» Prop2 : string

«CIM_Method» Meth1 ( )

<Property> «CIM_Property» Prop2

Keywords:

Applied Stereotypes:

Stereotype	Profile	Required
CIM_Property	CIM	False
Property_Constraints	CIM	True

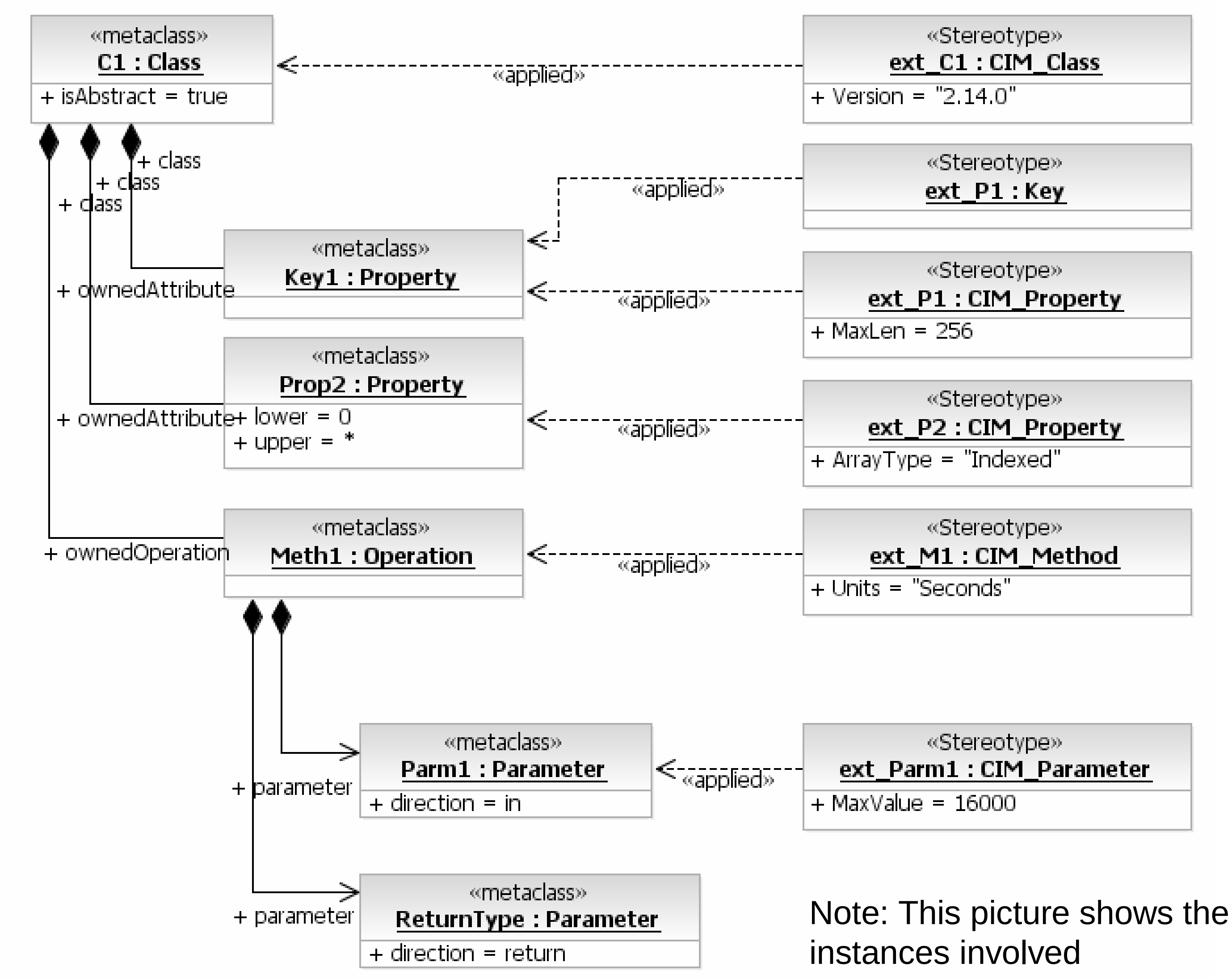
Stereotype Properties:

Property	Value
⊟ CIM_Property	
Alias	
ArrayType	2 - Indexed
BitMap	Entries: 0
BitValues	Entries: 0
Counter	False
Deprecated	Entries: 0
Description	

Class diagram

Properties view, showing the applied stereotypes

Simple Example
(UML metaclass instances)



Note: This picture shows the UML metaclass instances involved

OCL Constraints

- OMG defines Object Constraint Language (OCL), currently at V2.0
- OCL allows to define constraints at meta model and at user model levels

- DSP0219 defines 306 OCL constraints at the meta model level
  - UML profiles for specific UML tools will implement these constraints

  -> Ensures that a UML modeler stays within the bounds defined by CIM

  -> Relieves the UML modeler from memorizing all these rules

- New OCL qualifiers allow to specify OCL constraints for CIM Schema
  - ARCHCR00057 added OCL related qualifiers:
    ClassConstraint
    MethodConstraint
    PropertyConstraint
  - Will be in CIM Infrastructure Specification V2.4, currently rolling out as preliminary standard
  - Sample usage scenarios:
    - Verify that OtherDescription property is set when ValueMap property specifies "Other"
    - Verify consistency if one property value depends on another property value (even in another class)
    - Verify that an associated class has an instance if a property in a Capabilities advertises a capability

  -> Allows to express model relationships much better than ModelCorrespondence qualifier

Current spec status

- DMTF DSP0219 spec
  - V1.0 has been published as Preliminary Standard in 8/2007
  - http://www.dmtf.org/standards/published_documents/DSP0219.pdf
  - Sample implementation is available to DMTF and in the process for publishing
    - DMTF currently in the process of deciding whether sample implementations can be published

- OMG standardization
  - Agreement exists between DMTF and OMG to also publish DSP0219 as an OMG standard
  - OMG provides “Fast track” process for this (3..6 months)
  - OMG needs to change its processes in order to ensure that OMG spec releases and DMTF spec release do not deviate in the future.
    - Currently (10/2007) being discussed at OMG

Issues discovered

- OMG Issues being worked within OMG Arch. Board
  - Target: UML 2.2
- CIM Issues being raised as CRs
  - Target: DSP0004 V2.4

Outlook: Mgmt Profiles in UML

- End to end model interoperability is achieved with DMTF Management Profiles

- Mgmt Profiles today are specification documents
  - Compliant to "Profile Usage Guide" (DSP1001)

- DMTF has set up a new working group “Machine Readable Profiles”
  - a joint subgroup of the Architecture WG and the WBEM Infrastructure and Protocols (WIP) WG

- WG goal is to define a concept and format for machine readable Mgmt Profiles

- This should allow to map that format also to a UML representation of mgmt profiles

- UML tools can then be used to
  - Ensure consistency between CIM Schema and Mgmt Profiles
  - Define / edit Mgmt Profiles
  - Generate various outputs:
    - implementation guides / documentation
    - CIM providers for various provider APIs
    - Test drivers to test providers for compliance to mgmt profile
    - Java proxy objects for the CIM objects on the client side
    - Web Services wrappers for the CIM objects

Backup

OMG / UML Issues raised (1)

Req #	Description	Resolution
UML001	OCL Syntax in expressions unclear	OMG Issue #9886
UML002	No mechanism for description of elements	Done - addressed by the more general issue UML011
UML003	Optional values and evaluation of defaults	OMG Issue #9887
UML004	No mechanism to indicate that an element is experimental	Will not be raised with OMG, because OMG has the SPEM approach (Software Process Engineering Metamodel) as an answer to that. Mapping Experimental qualifier generically for V1 of the spec.
UML005	No mechanism to indicate that a property is a key	OMG Issue #9888
UML006	No mechanism to indicate that an element has been deprecated	Will not be raised with OMG, because OMG has the SPEM approach (Software Process Engineering Metamodel) as an answer to that. Mapping Experimental qualifier generically for V1 of the spec.
UML007	Representation of the special NULL value on defaultValue of UML Properties	Already covered by the OMG issue #9700 raised as a result of UML009
UML008	Clarify the aggregation side in a UML Association	OMG Issue #9889
UML009	UML's support for null values and semantics is unclear	OMG Issue #9700
UML010	Unnecessary restriction on aggregations being binary	OMG Issue #9701
UML011	No way of specifying element documentation	OMG Issue #9702 Addresses the more specific UML002
UML012	Unclear usage of LiteralExpression::type	OMG Issue #9703
UML013	"Property::lowerValue" is not a good name	OMG Issue #9704

Source: DSP0219 Issues+Reqs document, V0.3.2

OMG / UML Issues raised (2)

Req #	Description	Resolution
UML014	ValueSpecification::isComputable() introduces conformance requirements not captured in compliance points	OMG Issue #9705
UML015	Definition of stereotype placement requires a name	OMG Issue #9706
UML016	The default for a Property should not be inconsistent with its type	OMG Issue #9622
UML017	All associations ends in the UML2 metamodel itself should be navigable	OMG Issue #9371
UML018	Show an example of correct notation for the metamodel	OMG Issue #9372
UML019	Use the new 'dot' notation in examples	OMG Issue #9373
UML020	AssociationClass is severely underspecified	OMG Issue #9374
UML021	Automatic application of stereotypes on attributes and association end points	Not being raised. Option 1 is quite incompatible to do, and option 2 introduces unwanted dynamics. One possible solution is to get tools to offer a stereotype palette from which modeling elements can be created, as opposed to create the native elements. Todo: email the issue ot the OMG mailing list to get awareness.
UML022	Meaning of isAbstract attribute on stereotypes	OMG issue #9830
UML023	Missing isLeaf attribute on stereotypes	OMG issue #9831
UML024	No construct to represent indexed arrays	TBD: This was issue CIM003. Discuss.
UML025	Clarify isRequired	OMG Issue #9890

Source: DSP0219 Issues+Reqs document, V0.3.2

CIM Issues raised

Req #	Description	Resolution
CIM001	The *Read* qualifier is not representable in UML	Closed - Generically mapped for now, drop the *Read* qualifier in CIM V3
CIM002	The *Write* qualifier is not natively representable in UML. Discussion in Arch/WIP WGs revealed that there is disagreement in what the qualifier defines	ToDo: Wait for resolution of *Write* qualifier discussion in Arch/WIP WGs, then resume here.
CIM003	The CIM *ArrayType* qualifier has three values, two of which can be mapped to the UML *isOrdered* attribute	Closed - ARCHCR00081 clarifies arrays in CIM. It turns out that all three array types make sense, and therefore this issue is no longer a CIM issue, but an OMG issue (OMG0023).
CIM004	The *Min* qualifier description does not exclude the value of NULL. However, it is believed that this value was always meant to be invalid for this qualifier.	Closed - ARCHCR00085 clarifies that NULL is not allowed for Min
CIM005	The *Max* qualifier description does not define the meaning of the value NULL. It is believed that NULL means "unlimited", i.e. "*" in UML notation.	Closed - ARCHCR00085 clarifies that NULL means "unlimited" for Max
CIM006	In UML, aggregations and compositions are defined only for binary associations. In CIM, no such restriction is defined currently. An according restriction should be defined in CIM.	Closed – Issue UML010 attempts to remove this restriction from UML

Source: DSP0219 Issues+Reqs document, V0.3.2