XMI and MOF:

a mini-tutorial

Perdita Stevens

Division of informatics

University of Edinburgh

http://www.dcs.ed.ac.uk/home/pxs

UML + XML = ?

This tutorial is about the combination of two major buzzwords: UML and XML.

My assumption is that everyone knows what UML is:

- the Unified Model(l)ing Language

- a diagrammatic language for recording the design of (object-oriented software) systems

but that perhaps not everyone knows how it’s defined?

How UML is defined

Two main documents:

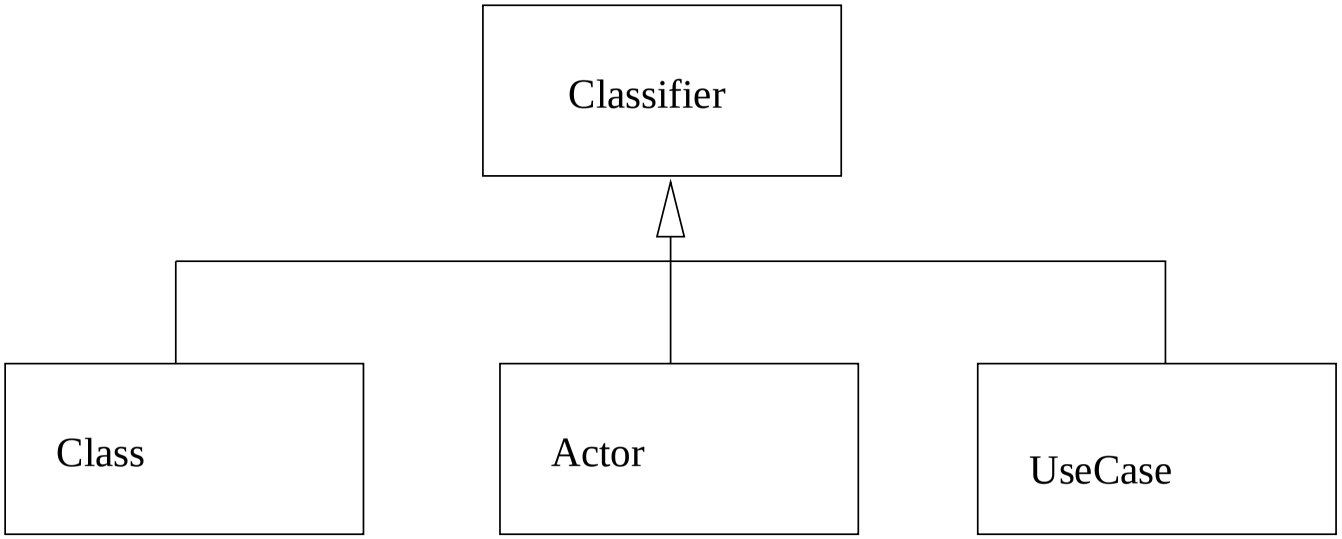
- Notation Guide : informal explanation of notation (concrete syntax) and its connection to abstract syntax.
- Semantics : *semi-formal specification of abstract syntax*, plus further explanation of semantics.

Semantics takes precedence over Notation Guide in cases of conflict – theoretically.

Plus: definition of the Object Constraint Language (OCL).

UML semantics document: syntax

The **abstract syntax** describes (in UML!) the relationships between kinds of UML model elements (the *metamodel*). For example, use cases, actors and classes are all said to be examples of classifiers:



Back in the days of the method wars...

... people wanted to be able to *switch tools*, which meant switching notations. But the different OOMLs recorded different information.

One major reason for UML success... but still not fully realised in practice.

Hmm... structured data (UML models) ... standard way to exchange it between tools.... yup.

XML may be the key to getting real value from UML tools

But how exactly to use XML?

XMI

XMI = XML Metadata Interchange

an OMG standard.

Most importantly,

XMI is a way to save UML models in XML.

In fact it’s more general.

UML is a MOF-based metamodel;

XMI shows how to save any MOF-based metamodel in XML

MOF 1

MOF = Meta Object Facility

an OMG standard

MOF is a simple language for defining languages, e.g., UML.

(Strictly UML's metamodel does not fit MOF exactly, so UML also has a *interchange metamodel* which does – hack!)

MOF looks remarkably like a small subset of UML; notated with UML.

OMG 4 level metamodel architecture

META-LEVEL	MOF TERMS	EXAMPLES
M3	meta-metamodel	“MOF Model”
M2	meta-metadata metamodel	UML Metamodel
M1	metadata model	UML Models
M0	data	Modelled systems

Why isn’t MOF just the same as UML?

From OMG-MOF1.3

> The key differences are due to different usage scenarios of MOF and UML. The MOF needs to be simpler, directly implementable, and provide a set of CORBA interfaces for manipulating meta objects. The UML is used as a general-purpose modeling language, with potentially many implementation targets.

MOF provides an “open-ended information modeling capability.”

Back to XMI

So, XMI is an attempt to take full advantage of

- UML's world dominance
- UML's definition in MOF
- XML's world dominance

and produce *a standard way to save UML models in XML*, in order to provide modellers with the ability to *move UML models between tools*.

(Extras for free: e.g. same for any other MOF-based language.)

What's in XMI1.1?

- Design principles.
- A set of XML Document Type Definition (DTD) production rules for transforming MOF based metamodels into XML DTDs.
- A set of XML Document production rules for encoding and decoding MOF based metadata.
- (Concrete DTDs for UML and MOF.)

Does allow for extensions, incomplete models etc.

And that means for UML...?

Because you can look up or down the metamodel architecture,

UML can be regarded as:

- an XML document that conforms to a DTD describing MOF;
- an XML DTD to which UML models must conform.

The latter is the more important for most people.

Getting concrete

Let’s see an example of a UML model saved as an XML document which matches a UML DTD (“saved as XMI”)

and an example of how such a thing can be used.

Example: produce HTML from XMI

Start with your UML model saved as an XMI file.

In XSL (XML Style Language) write a pattern-matching style sheet saying what information to extract from the XMI document and how to turn it into HTML.

For example, produce automatically updated web pages detailing the attributes and operations of each class.

A full tutorial description of this is at

`http://www.objectsbydesign.com`.

So who does need to know about MOF?

If you define your own modelling language M and need to save models in XML, in principal you might define your M as an instance of the MOF and use XMI to develop a DTD for M models.

But often an easier alternative might be to make your language an explicit subset of UML, and just use the UML DTD or some cutdown variant of it.

Are there cases where the latter is not possible but the former is sensible? I don't know.

Caveats

1. XMI/UML are co-evolving and settling down.

   Different combinations of UML versions and XMI versions exist: only an exact match will enable tool-to-tool interchange.

   XMI1.0, 1.1

   UML1.1, 1.3, (1.4)

2. XMI doesn't (yet) specify how to record graphical information; tools do this differently.

More information

WWW: Lots of pointers at “The XMI Hackers’ Homepage”

`http://www.dcs.ed.ac.uk/home/pxs/XMI/`

More things are gradually appearing there too...

Books:

- David Carlson, Modeling XML Applications with UML
- Steve Brodsky’s forthcoming book on XMI in Java
- mine on using XMI to get more value from UML tools