The Unified Modeling Language
Opportunities and Challenges for Formal Methods

Stuart Kent

University of Kent @ Canterbury, UK
www.cs.ukc.ac.uk

Outline

- An update on UML
  - A little history
  - UML and other OMG standards
  - The OMG revision process
  - UML related RFP’s
  - UML profile RFP’s
  - UML 2.0
- Challenges & opportunities - Language definition
- Challenges & opportunities - Tools
- A precise OO meta-modeling facility - MMF

A little history

- UML began with Rational and the 3 amigos
- OMG put out a request for proposals for a standard object modeling language
- A number of consortiums put in initial submissions
- Initial submissions were combined into the eventual final submission, under one consortium
  - exception: the OPEN group
- UML was accepted by and passed over to the OMG
- Revision of UML is now in the hands of the OMG
- Possibility of ISO standardisation
UML & other OMG standards

- MOF
  - Language for describing other languages
  - Meta-models in MOF used to generate IDL and XMI

- CORBA (IDL)
  - Generate IDL from meta-model in MOF
  - IDL represents set of interfaces to a repository supporting that meta-model
  - UML could be one such meta-model
  - Generate IDL from (certain) UML models

- XMI (XML interchange)
  - Dictates how UML and MOF models are interchanged in XML
  - Based on MOF technology

OMG processes

- Major Revision
  - Request For Information (RFI)
  - Request For Proposals (RFP)
  - Initial Submissions
  - Final Submissions
  - Voting
  - Accept Major Revision x.0

- Minor Revision
  - Issues submitted
  - Revision Task Force (RTF) consider
  - Voting
  - Reject
  - Accept Minor Revision n.x

UML related RFP’s

- www.omg.org/techprocess/meetings/schedule/
- Action Semantics
- Various Profiles
- UML 2.0
  - 4 RFP’s
  - RFP’s just issued; awaiting initial submissions

Action Semantics

- Goal is to sort out
  - abstract syntax
  - semantics

  For an action language for UML

- Initial submission received
- Adopts a meta-modeling approach
- Semantics expressed as a mapping from abstract syntax to possible executions
UML Profile RFP's

- EDOC – Enterprise Distributed Object Computing
  - Working on merging 2 initial submissions
  - Component architecture, viewpoints & whole lifecycle
  - Closely tied to EAI (intersecting teams)
- Textual language for EDOC profile
  - Awaiting initial submissions
- CORBA profile
  - Use UML to write IDL (in a nutshell)
  - Revised submission received
- Scheduling profile
  - Real-time profile for UML (in a nutshell)
  - Awaiting initial submissions
- Event Based Architecture in Enterprise Application Integration (EAI) profile
  - Working a single joint revised submission
  - Closely tied to EDOC

4 RFP’s

- Infrastructure
- Superstructure
- Object Constraint Language (OCL)
- Diagram interchange (planned)

UML 2.0 Working Group
www.celigent.com/omg/adptf/wgs/uml2wg.htm

Infrastructure

- Alignment with MOF
- Family of languages
  - Notation mix
  - Profiles
  - Language Evolution
  - Patterns
- Comprehensive definition + separation of concerns

Outline

- An update on UML
- Language definition
  - Challenges
  - Opportunities
- Tools
- A precise OO meta-modeling facility - MMF

Comprehensive definition + separation of concerns

XMI

- Syntax(es)
- Model
- UML 1.3
- Concepts
- Instance

The precise UML group & IBM have done a feasibility study on this. Discussed later. Also see www.puml.org
Language definition - challenges

- UML is a family of languages
  - Profiles
  - Continual evolution – product lines?
- Accessibility of the definition and tools used to produce it
  - Who are the stakeholders?
- Backwards compatibility with the existing definition
  - What does this mean, exactly?
  - Who are the stakeholders?
- Separation of concerns (syntax, semantics etc.)
  - Visual languages?

Language definition - opportunities

- Formal language definition is at the heart of FM
- Two approaches (at least):
  - **Translate UML to a FM**
    - E.g. UML to PVS or Object Z
    - Makes use of abstractions from semantic domain already captured by FM
    - Definition can be colored by FM
    - May not address
      - Family issues
      - Accessibility of definition
      - Backwards compatibility
      - Visual language issues
  - **Adapt FM techniques to UML**
    - Address challenges directly, borrowing from FM as appropriate
      - Borrow from FM separation of concerns
        - Concrete/abstract syntax
        - Model-theoretic/axiomatic semantics
      - Learn from graph grammar work in defining diagrammatic syntax
      - Example is pUML feasibility study

Outline

- An update on UML
- Language definition
- **Tools**
  - Challenges
  - Opportunities
- A precise OO meta-modeling facility - MMF

Tools - challenges

- Tools that allow a model to be exercised and reasoned with
  - Testing models – instance generation and instance checking
  - Reasoning
  - Feedback is given through the modeling language
- Specialized (visual) language editors
  - VL = diagrams + editor
  - 3D?
- Support for language families
  - Federated architecture
  - Automatic generation of tools from definitions
Tools - opportunities

★ FM delivers techniques for exercising models:
  ○ Automated reasoning
  ○ Model checking
  ○ Animation (in presence of partial models)

★ 2 approaches to utilizing these techniques:
  
  **Translate UML to a FM**
  
  - Can make use of existing tools
  - Feedback is the challenge, e.g.
  
  1. translate model
  2. Perform check
  3. translate result & debugging info

  **Adapt FM techniques to UML**
  
  - Can make use of existing tools
  - Feedback is the challenge, e.g.
  
  1. translate model
  2. initiate reasoning step
  3. Perform step
  4. translate result & debugging info

Outline

★ An update on UML
★ Language definition
★ Tools
★ A precise OO meta-modeling facility - MMF

Goals of MMF

To provide a **rearchitected definition** of UML which:

★ is **precise** to the degree that
  ○ conformance can be checked systematically, without argument and, preferably automatically
  ○ self-consistency of the definition can be established.

★ is **comprehensive**, covering syntax, both concrete and abstract, and semantics. On the other hand, redundant and overlapping concepts should be kept to a minimum.

★ accepts that UML is a **family** of languages, providing mechanisms which
  ○ allow profiles and language extensions to be defined in a controlled and managed way
  ○ make the relationships of profiles and extensions to existing language fragments explicit and unambiguous.

★ is **accessible** to tool builders and those involved in the standardization of the language.
Results

Intellectual Property
- A Meta-Modeling Facility (MMF)
- Fragments of a rearchitected definition of UML

Concrete Artefacts
- A report describing the facility and UML fragments
- A tool supporting the facility
- Available from www.puml.org

Conclusion
- Rearchitecting UML using pOOMM approach is entirely feasible

Meta-Modeling Facility (MMF)
- MMF = MM Language (MML) + MM Tool (MMT)
- MML is used to define the UML family
- MML is a member of the UML family
- MML has facilities for componentizing language definitions, maintaining a separation between
  → Language aspect (model–instance, syntax–concepts)
  → Subject area (static core, constraints, model management, etc.)
- MMT supports
  - checking
  - reflection
  - constraint execution

Precision – a confidence trick
1. Construct a MML model of MML
   - based on intuitive semantics of MML
   - class/package diagrams + constraints
2. Implement tool based on that model; define MML in tool
3. Learn from tool to improve model
4. Implement changes in tool and definition in tool
   - Confidence increases the more one cycles through 3, 4

UML Architecture: subject areas

Arrow represents package generalization – cannot produce between packages in Rose
### The staticCore.instance.concepts package

```
InstanceElement
  + elements
  + value
  ++ slots
```

### The staticCore.semantics package

```
Classifier + of + instances
  + of + instances
  + of + instances

Instance + of + instances
  + of + instances

Class + of + instances

Object
```

### Concepts (abstract syntax)  Semantic domain

### OCL constraints for semantic mappings

```
context.uml.staticCore.semantics.Instance inv:
satisfies(c : Classifier) : Boolean
if self.of = c then
  of.allContents() -> forall(e1 |
    elements -> exists(e2 |
      e2.name = e1.name and
      e2.satisfies(e1)))
  else false
endif
```

### A systematic method for extending MML

1. determine whether the model element is a subclass of Classifier, i.e. exhibits the properties of a generalisable container;
2. if so, subclass the model element from Classifier in the model.concepts package;
3. constrain the model element’s contents to be those of the attribute ‘elements’;
4. in the instance.concepts package, identify or add a new instance subclass which is an instance of the classifier;
5. in the semantics package, link them by subclassing the ‘of/instances’ association;
6. for each element of the new model element repeat steps 4-5;
7. determine any dependencies between instances and their elements, and specify these using appropriate constraints.
MML includes a model of OCL

Current architecture of UML / MOF

A possible new architecture for UML / MOF

MMT architecture
Advantages of MMT

- Meta-models, models and instances can be checked for correctness against their definition
- Tool to support MML comes from executing MML meta-model in virtual machine
  - Changes are easy to implement
  - As well as ability to check a M-M against MML, the M-M can be executed providing direct tool support for the M-M (CASE tool generation)
- Because of comprehensive M-M, this should also support e.g. syntax mappings
- User only needs to deal with MML and languages defined in MML

Conclusions

- FM has a lot to offer UML, specifically:
  - Patterns and techniques for formal language definition
  - Semantically sophisticated tools
- There is an opportunity now to make UML into a formal language. Key challenges are:
  - To treat UML as a family of languages
  - To provide accessible definitions
  - To deal with visual syntax
  - To rework all the stuff for dynamic modeling
  - To deal with backwards compatibility issues
- Users could really do with tools that allow models to be exercised. Key challenge is:
  - To feedback results and debugging information through UML
- Reworking the UML infrastructure using a precise OO meta-modeling approach should (at least) provide a better platform for FM folks to work from