Semantics of UML 2.0:

Defining the Semantic Mapping

Manfred Broy (TU Munich)
Michelle Crane (Queen's U)
Juergen Dingel (Queen's U)
Bernhard Rumpe (TU Braunschweig)
Bran Selic (IBM Canada)

Nov 7, 2005

Overview

- **The context**
  - UML Specifications as a predicates
- **The meat (well, an appetizer)**
  - The semantic mapping for class models
- **The issues**
  - Ours and yours

What is a UML Specification?

- **A UML specification *Spec* is a collection of different kinds of models (diagrams)**
- For instance,
  - class model (CM)
  - interaction model (IM)
  - state machine model (SMM)
  - ...

$$Spec \in \mathcal{P}(CM \cup IM \cup SMM \cup ...)$$

What Does a UML Specification Mean?

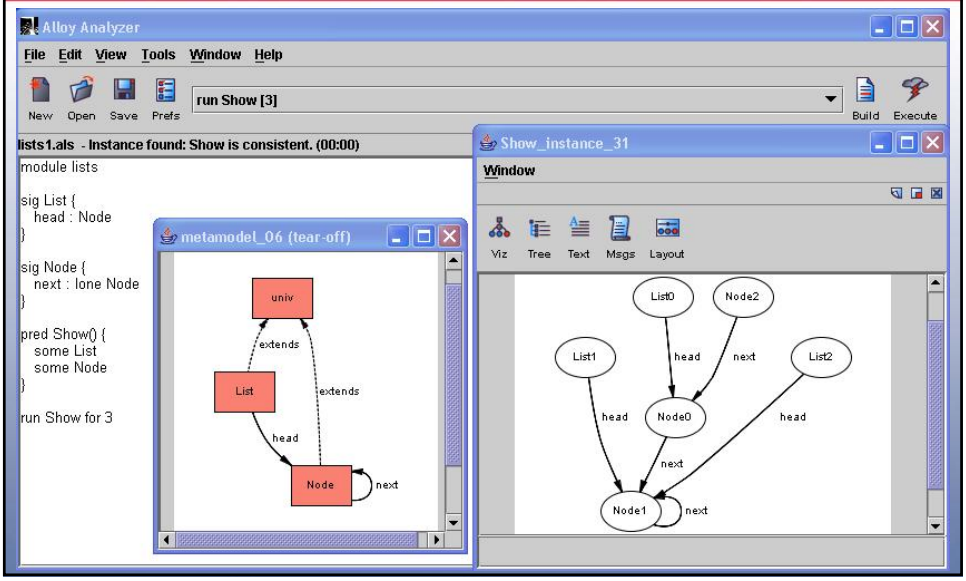
- **The meaning of a UML specification *Spec* is set of all systems satisfying the constraints in *Spec***

$$[\![\_]\!]_{UML} : UML \rightarrow \mathcal{P}(SysModel)$$

$$[\![Spec]\!]_{UML} = \{sm \in SysModel \mid sm \ satisfies \ Spec\}$$

- See, e.g.,
  - [Broy et al]: CIP (1980ties), Spectrum
  - Daniel Jackson's Alloy:
    - class models (object models) as heap invariants

Alloy: "Class Models are Heap Invariants"



What Does a UML Specification Mean?

$$[\![\_]\!]_{UML} : UML \rightarrow \mathcal{P}(SysModel)$$

where

$$UML = \mathcal{P}(CM \cup IM \cup SMM)$$

such that

$$[\![\{cm_i\}_i \cup \{im_j\}_j \cup \{smm_k\}_k]\!]_{UML} = \bigcap_i [\![cm_i]\!]_{CM} \cap \bigcap_j [\![im_j]\!]_{IM} \cap \bigcap_k [\![smm_k]\!]_{SMM}$$

[Rumpe 1996]: "Integration through intersection"

What is the Meaning of a Particular Model?

For instance, class models

$$[\![\_]\!]_{CM} : CM \rightarrow \mathcal{P}(SysModel)$$

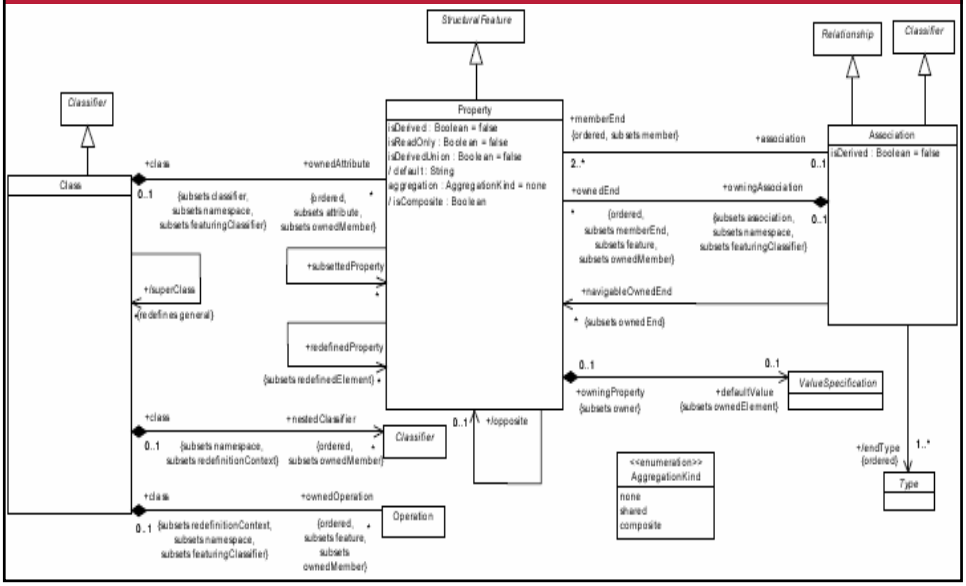
such that

$$[\![\_]\!]_{CM}(cm) = ? \quad for\ all\ cm \in CM$$

Abstract Syntax for Class Models

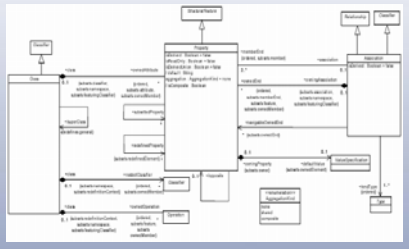
- What exactly makes up an class model (class diagram)?
- Could look into UML Meta Model:

Abstract Syntax for Class Models (Cont'd)



Abstract Syntax for Class Models (Cont'd)

- What exactly makes up a class model?
- Could look into UML Meta Model:

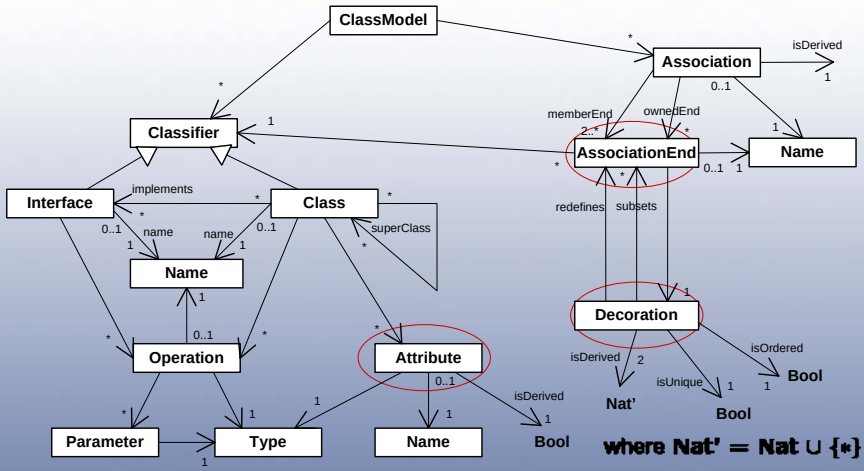


- **2 Problems:**
  - complex
  - difficult to define semantic mapping

Abstract Syntax for Class Models (Cont'd)

- **So, we**
  - simplified the notion of class model
  - defined abstract syntax in tuple notation

Abstract Syntax of Class Models: Simplification



plus context constraints

Abstract Syntax for Class Models: Tuple Notation

$$
\begin{aligned}
ClassModel &= \mathcal{P}(Classifier) \times \mathcal{P}(Association) \\
Classifier &= Class \cup Interface \\
Class &= Name \times \mathcal{P}(Attribute) \times \\
&\quad \mathcal{P}(Operation) \times \mathcal{P}(ClassId) \times \\
&\quad \mathcal{P}(InterfaceId) \\
Attribute &= Name \times Type \times Bool \\
&\vdots
\end{aligned}
$$

What is the Meaning of Class Models?

- **Step 1:**
  - **simplify class models** by expressing certain notions (e.g., associations) **in terms of others** (e.g., attributes) including **possibly some constraints**:

$$simp_{CM} : CM \rightarrow CM_{core} \times FOL$$

- **Step 2:**
  - **map** simplified class models with constraints to system model:

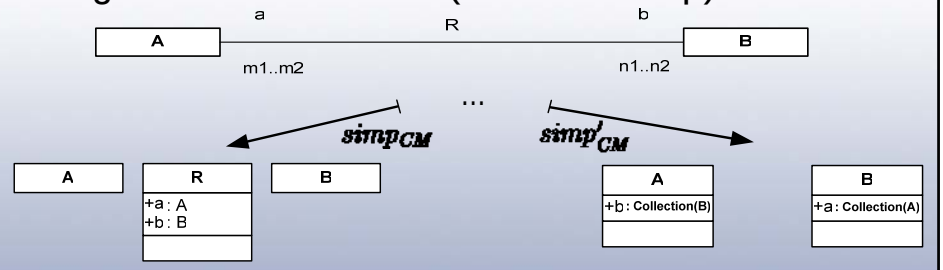
$$[\![\_]\!]_{CM_{core}} : CM_{core} \times FOL \rightarrow \mathcal{P}(SysModel)$$

- **Step 3:**

$$[\![\_]\!]_{CM} = [\![\_]\!]_{CM_{core}} \circ simp_{CM}$$

Meaning of Class Models: Step 1 (Simplification)

Dealing with **associations** (and ownership):



$(\forall a' : A.n_1 \leq |image_R(a')| \leq n_2) \wedge$
$(\forall b' : B.m_1 \leq |image_R(b')| \leq m_2)$
where
$image_R(a') = \{b' : B \mid \exists r : R.\ r.a = a' \wedge r.b = b'\}$
$image_R(b') = \ldots$

Ends owned by R

$(\forall a' : A.n_1 \leq |a'.b| \leq n_2) \wedge$
$(\forall b' : B.m_1 \leq |b'.a| \leq m_2)$

Ends owned by A, B

Meaning of Class Models: Step 2 (Mapping)

$$[\![\_]\!]_{CM_{core}} : CM_{core} \times FOL \rightarrow \mathcal{P}(SysModel)$$

such that for all $(cm, f) \in CM_{core} \times FOL$ we have

$$[\![(cm, f)]\!]_{CM_{core}} =$$
$$\{sm \in SysModel \mid$$
$$(\forall c \in class(cm).$$
$$name(c) \in UTYPE_{sm} \wedge name(c) \in UCLASS_{sm} \wedge$$
$$name(c) \approx_{sm} Ref\ Rec\{a : Loc\ t \mid$$
$$\exists f.(a, t, f) \in attr(c)\} \wedge$$
$$\forall c' \in class(cm).c\ sub_{cm}\ c' \Rightarrow c\ sub_{sm}\ c') \wedge$$
$$\forall a \in asso(cm).name(a) \in UASSOC_{sm} \wedge$$
$$sm \models f\}$$

To Do

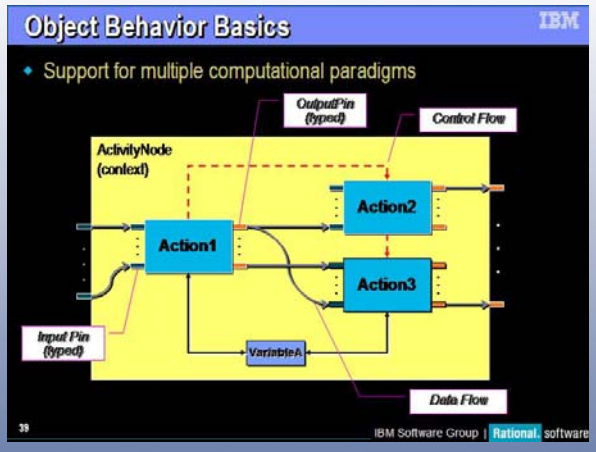
- **Add behaviour:**

$$[\![\_]\!]_{IM} : IM \rightarrow \mathcal{P}(SysModel)$$
$$[\![im]\!]_{IM} = ?$$

$$[\![\_]\!]_{SMM} : SMM \rightarrow \mathcal{P}(SysModel)$$
$$[\![smm]\!]_{SMM} = ?$$

Detour: Actions

- Action: Fundamental unit of behavioural specification
- **Concepts:**
  - typed pins
  - control flow
  - data flow
- **Categories:**
  - communication
  - objects, links
  - etc



Adding Behaviour

One approach: Use Action language

$$[\![\_]\!]_{IM} = [\![\_]\!]_{Action} \circ [\![\_]\!]_{IM_{Act}}$$

where

$$[\![\_]\!]_{IM_{Act}} : IM \rightarrow Action$$

$$[\![\_]\!]_{Action} : Action \rightarrow \mathcal{P}(SysModel)$$

$$[\![\_]\!]_{SMM} = [\![\_]\!]_{Action} \circ [\![\_]\!]_{SMM_{Act}}$$

where

$$[\![\_]\!]_{SMM_{Act}} : SMM \rightarrow Action$$

Issues: Meaning of Class Models

- **Have we simplified / assumed too much?**
  - $simp_{CD}$: associations as attributes
- **Navigability**
  - "being able to **efficiently** access entity on other end"
  - purely a run-time concept
  - currently not modeled
- **Derived properties**
  - express constraint (e.g., x = y+z)
  - may have consequences at run-time
  - but, when to check/enforce?

Issues: Meaning of Class Models (Cont'd)

- **"Associations as maps" vs "associations as tables"**
  - UML (and the community) seems **confused** on that
  - **related, but** impact meanings of **uniqueness and ownership impacted**

As table	As map
Uniqueness: Impossible for only 1 end to be unique	**Uniqueness:** Possible for only one end to be unique
Ownership: Association owns both ends	**Ownership:** 2 notions of ownership

  - **no notation in UML** to choose certain view

Issues

- Any others?