Vérification de la cohérence de modèles UML

Workshop Consistency Problems in UML-based Software Development

Jean Louis Sourrouille

Origine des problèmes de cohérence de modèles

- Processus de développement
  - Multi-vues, Dépendances, Niveaux d'abstraction/Raffinement
- Sémantique d'UML imprécise

Diagramme:
- Système
- Vue
- Modèle, $Model_{in}$
- Cohérence intra-modèle
- Propriété du modèle
- Relation entre modèles
- Cohérence inter-modèles
### Questions relatives à la cohérence de modèles

- **Définition de la cohérence**
  - Règles : ne pas violer de contrainte (complétude ?)
  - « Translationnelle » : la traduction du modèle dans un autre langage vérifie des propriétés (vérification partielle ?)

- **Cohérence et processus de développement**
  - Règles de raffinement
    (Object Z, CSP..., profil UML comportant des règles de transformation)
  - Méthodologie de développement (guide de style...)

- **Outils de vérification**
  - Vérifier des propriétés directement sur le modèle en UML
    Contraintes en OCL, Règles de production
  - Traduire le modèle dans un autre langage

---

### A Pragmatic View about Consistency Checking of UML Models

*Jean Louis Sourrouille*
A model of modeling

Questions

Let m(S)/f a model of a system S in a formalism f

- Is the UML model m(S)/UML consistent ?
  - The UML syntax and semantics should be completely specified
  - There is mm(UML)/F : F is a formal language (or mm is formal)
    - UML Metamodel: mm(UML)/MOF, neither precise nor formal
    - It is not possible to check definitely UML models
- Is it possible to formally define a UML ?
  - It is ! (yet another formal language ?)
- Is it advisable to define very precisely a UML ?
  - If all the possible model interpretations should be known, no!
- How to deal with consistency checks in this context ?
  - Intra-consistency only (horizontal)
Basics

- **Syntactic rules:** is this expression valid?
  - Defined by the metamodel structure and wellformedness rules
    - Include static semantics, e.g., circular inheritance is forbidden
  - *By extension,* include all formal rules (in OCL)

- **Semantics:** is this expression true (has a meaning)?
  - Language constructions are associated with *notions in a semantic domain*
  - Natural language in the UML description *(Semantics)*

*Remark:* Constraints related to the modeled domain are also called *«semantic constraints»*

- Does the model conform to the modeled system?
- *To separate from consistency issues*

---

**UML semantic domain = modeling domain**

- Consistency checking implies to agree about the modeling domain semantics *(semantics induces constraints)*
Interpretation

Development: From abstract (no details) to formal model (code)
⇒ Multiple interpretation is useful to keep abstract models
The real question is: What is the set of licit interpretations?

A consistent model has at least one licit interpretation

1. Checking syntactic constraints

Assuming that the UML syntax has all the required properties:
- Tools should check the syntax
- Tools should implement the whole UML

In practice users have to verify manually the syntax
Strangely, even non valid expressions seem to have a meaning!
2. Checking semantic constraints

Manually checked by definition: What can be done?

1- To reduce the number of semantic constraints
   – Increasing the expressive power of OCL
   – Improving constraint description

2- To define the set of licit interpretations within a community

Interpretations in the Modeling domain

Licit interpretations

Licit interpretations within a community

How to define the set of licit interpretations?

– Adding constraints easy to express (and to check)
  Restrict the UML, e.g., one trigger event only in a sequence
  ⇒ Sequence diagram order becomes a total order

– Using a style guide
  Including development process guidelines

Where serious errors come from?
Does Multiview add consistency issues?

The system model is the merging of views

or

Views represent system model parts

A ModelElement has the same properties in all the views

Going deeper to detect inconsistencies

- Most errors are syntactic (some are not expressed in the UML)
  e.g., within a namespace, any used name (library) should be known
- The metamodel should provide a path between linked elements
  e.g., from Borrow() to Loan()
- Going further requires to express more constraints
  OCL ? Formal languages ? Specific checking program ?
  Natural language (manual check) ?
Constraints in natural language

- ... May be precise
  « A guard should not have side effects »
  OCL: self.transition->stateMachine->notEmpty implies
  post: (self.transition.stateMachine-> context = self.transition.stateMachine->context)
  Where context is the ModelElement associated with the StateMachine.

- ... May be easier to express
  - « Any name should be visible in the namespace it is used »

- ... and in practice, are still very useful
  Not to reject because it is not « formal »
  (Formal languages are defined using natural language!)

Translation of the model into code

- Programming languages are formal
  i.e., « only one » interpretation (equivalence class in fact)
- Which interpretation to choose among the licit ones?
  Meaning defined by the target the language
Translation of expressions into formal languages

- **Translational approach**
  - \texttt{m(expr)/UML} $\rightarrow$ \texttt{m'(expr)/F}, where \texttt{F} is a formal language
  - \texttt{F} is chosen according to the model properties to verify
  - Allows reusing tool, knowledge, skills
  - Widely proposed (B, object Z, CSP, etc.)

- **Going to implementation step**

Dealing with the interpretation choice

- **To define a specific profile adding constraints**
  - Not always possible using existing tools (lack of parameters)

- **To generate the code from \texttt{F}**
  - The whole model is translated into \texttt{F}
  - More than executable UML (properties proof)
Translation into a rule based language

- All the interpretations are kept
- Additional checks
  - Semantic rules becoming syntactic (metalevel access, etc.)
  - Semantic rules within a community

Feedback easier than with OCL

Conclusion

- UML semantic domain not formally defined
- Proposal (UML compatible)
  - To define the set of licit interpretations within a community
    According to the development process and the context (tools)
- UML should be improved to reduce the developer burden
  - Greater expressive power of the meta-metalanguage
  - More precise metamodel
  - More complete (paths to check element dependencies)
- Tools should
  - Check syntactic constraints
  - Support the whole UML
  - Improve code generation
Z. Huzar, L. Kuzniarz, G. Reggio, J.-L. Sourrouille,  
"Consistency Problems in UML-based Software Development",  
in *UML Modeling Languages and Applications: 2004 Satellite Activities*,  
LNCS 3297, 2005, pp1-12

Jean-Louis Sourrouille, Guy Caplat,  
"A Pragmatic View about Consistency Checking of UML Model",  
Workshop on Consistency Problems in UML-Based Software Development,  
2003, pp43-50  
http://www.ipd.bth.se/consistencyUML/Consistency_Problems_in_UML_II.pdf

**Syntactic vs. Semantic Constraints**

<table>
<thead>
<tr>
<th>Clause</th>
<th>Definition</th>
<th>Check</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>adjective</em>, <em>commonNoun</em></td>
<td>Syntax: adjective, commonNoun</td>
<td>Formal language</td>
</tr>
<tr>
<td><em>commonNoun</em> :</td>
<td>- <code>man</code></td>
<td>Manual</td>
</tr>
<tr>
<td><em>adjective</em> :</td>
<td>- <code>happy</code>, <code>old</code>, <code>mellow</code></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clause</th>
<th>Definition</th>
<th>Check</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>clause</em></td>
<td></td>
<td>Automatic</td>
</tr>
<tr>
<td><em>clauseHum</em> :</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>adjHum</em>, <em>comNounHum</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>adjectiveHum</em> :</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>comNounHum</em> :</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>adjectiveObj</em> :</td>
<td></td>
<td></td>
</tr>
<tr>
<td><em>comNounObj</em> :</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

« Happy apple »

- syntactically correct
- syntactically incorrect

**Definitions**

<table>
<thead>
<tr>
<th>Syntactic constraint</th>
<th>Semantic constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Check</td>
</tr>
<tr>
<td>Formal language</td>
<td>Automatic</td>
</tr>
<tr>
<td>Natural language</td>
<td>Manual</td>
</tr>
</tbody>
</table>