Åbo Akademi University

Difference and Union of Models, 10 years later

Ivan Porres
Åbo Akademi University
ivan.porres@abo.fi

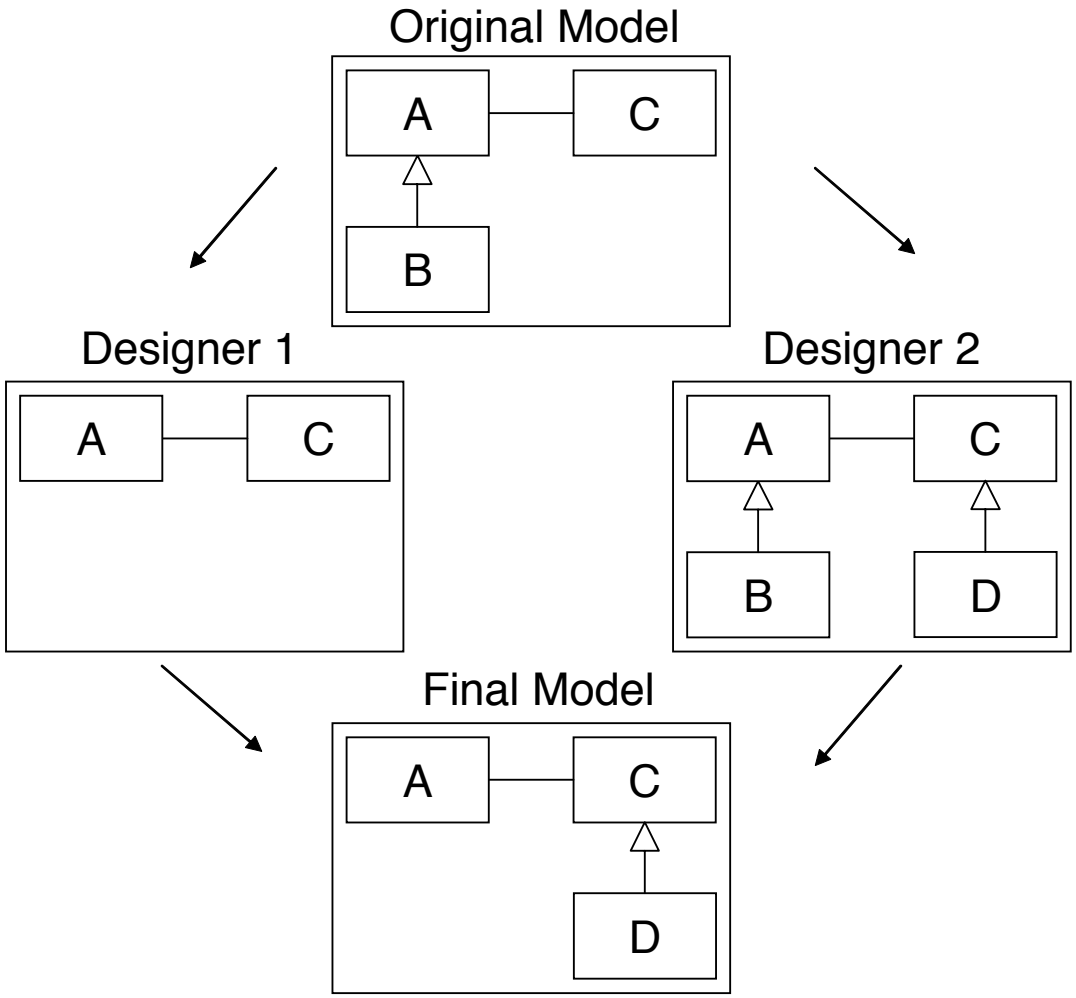
MODELS 2013 Conference
Miami October 3, 2013

Difference and Union of Models

Marcus Alanen and Ivan Porres

UML 2003 Conference, San Francisco

Motivation: Fine-grained version control with optimistic locking



The Context: 2002-2003

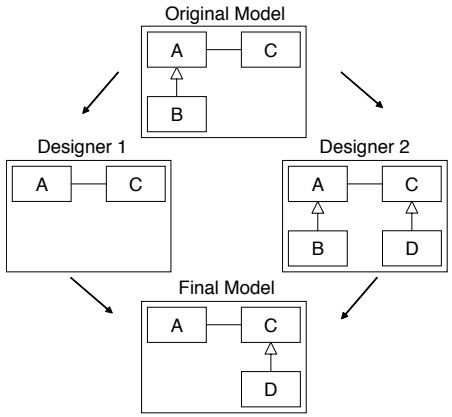
- Martin Fowler: What Is the Point of the UML?, UML 2003
  - "Models as sketches vs formal designs"
- Jean Bézivin, MDA: From Hype to Hope, and Reality, UML 2003
  - "Everything is a model"
- Stuart Kent, "Model Driven Engineering", IFM 2002
  - Defines MDE, identifies the need for families of languages, transformation, aligns metamodelling with formal language engineering

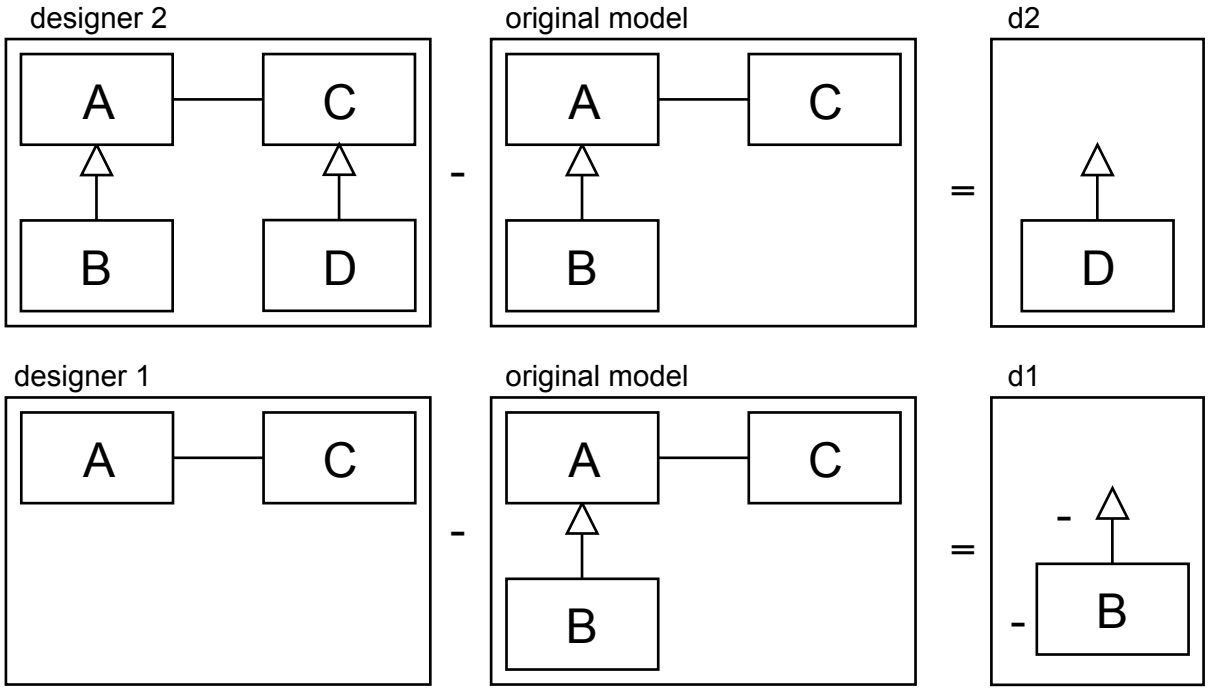
The Problem

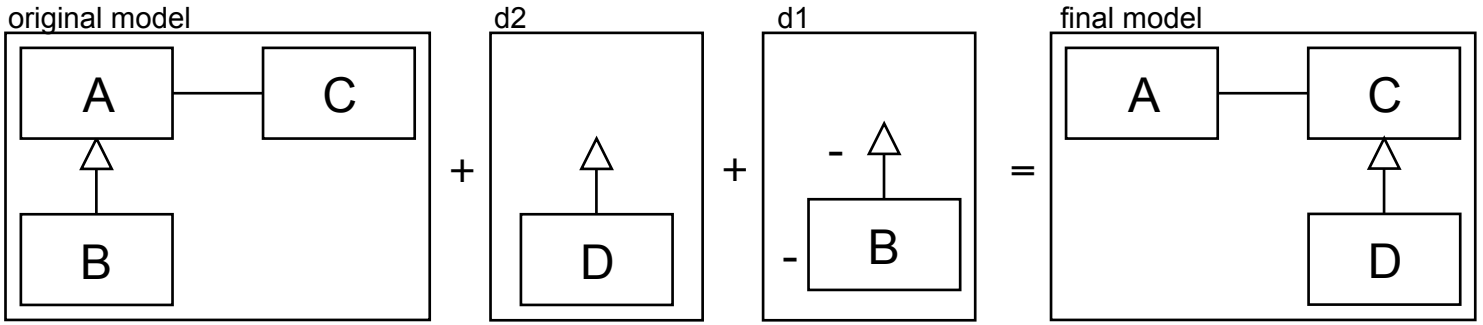
- Difference of two models: $M_2 - M_1 = \Delta$
- Merge of a model and a model difference $M_1 + \Delta = M_2$
- Union of Models: $M_f = M_0 + (M_1 - M_0) + (M_2 - M_0)$

- For models in any MOF-based language

The Problem


Original Model
A
C
B
Designer 1
A
C
Designer 2
A
C
B
D
Final Model
A
C
D


designer 2
A
C
B
D
-
original model
A
C
B
=
d2
D
designer 1
A
C
-
original model
A
C
B
=
d1
-
-
B


original model
A
C
B
+
d2
D
+
d1
-
-
B
=
final model
A
C
D

String-based comparison

- Same model, serialized as two different strings

```
<UML:Model xmi.id = '122' name = 'Example Model'/>
  ...

<UML:Model name = 'Example Model' xmi.id = '122'/>
  ...
```

XML-based comparison

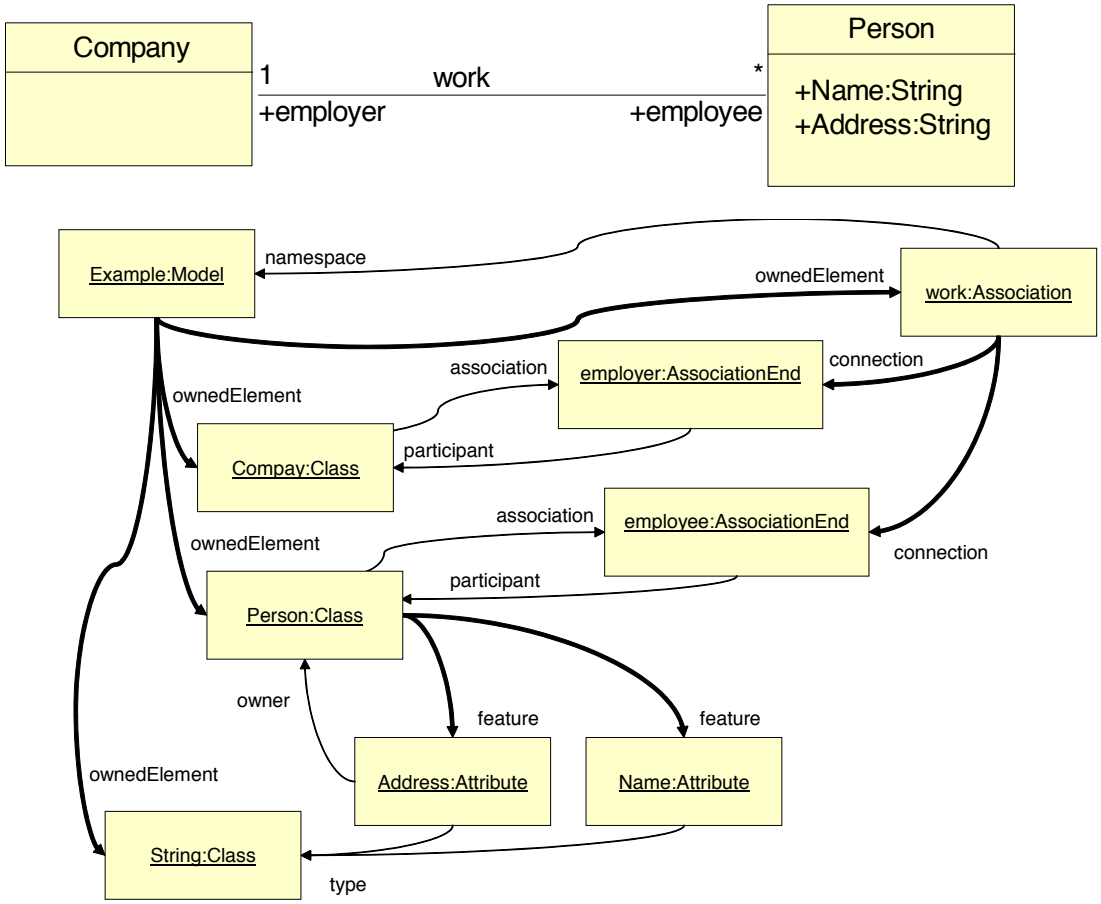
- Same model, serialized as two different XML documents

```
<UML:Model xmi.id = '122' name = 'Example Model'>
 <UML:Namespace.ownedElement>
  <UML:Class xmi.id = '123' name = 'Customer'>
  <UML:Class xmi.id = '124' name = 'Product'>
  ...
```

```
<UML:Model xmi.id = '122' name = 'Example Model'>
 <UML:Namespace.ownedElement>
  <UML:Class xmi.id = '124' name = 'Product'>
  <UML:Class xmi.id = '123' name = 'Customer'>
  ...
```

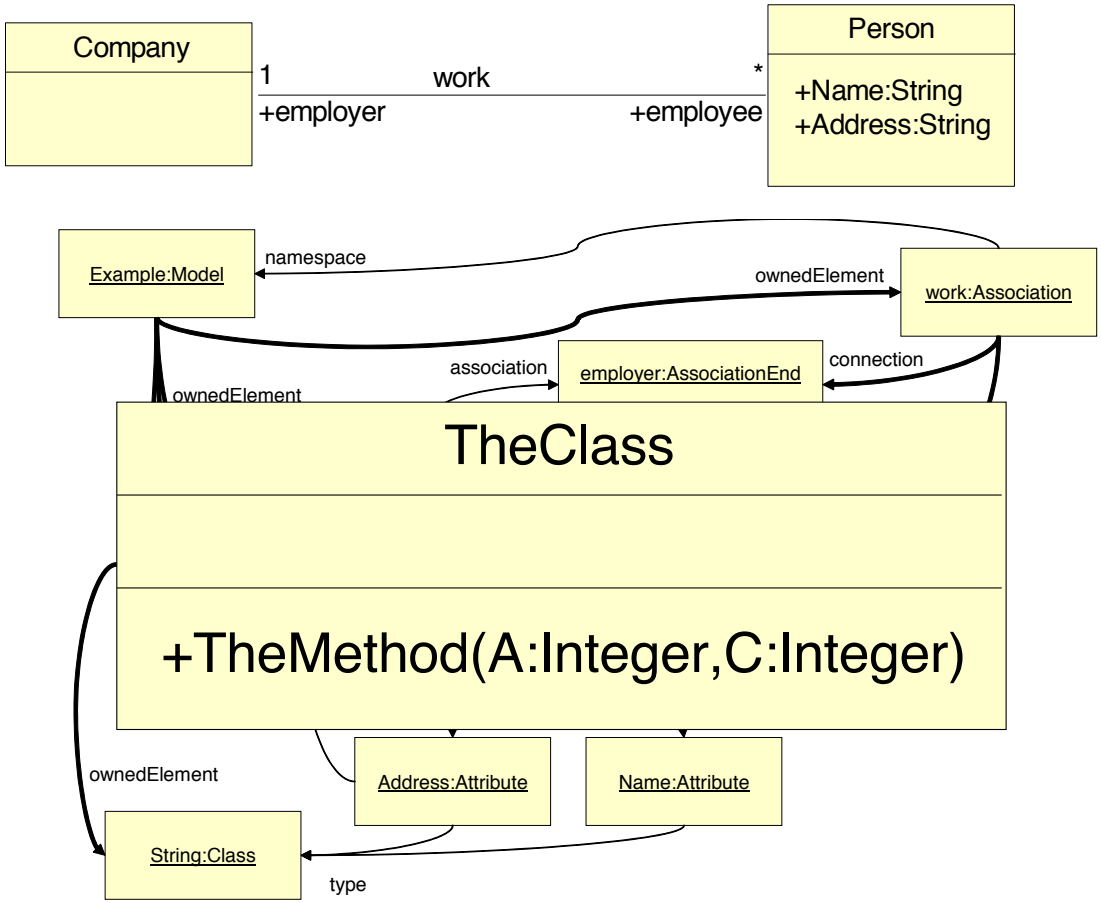
Underlying Model Structure

- Typed model elements, attributes store primitive values, features store links to other model elements



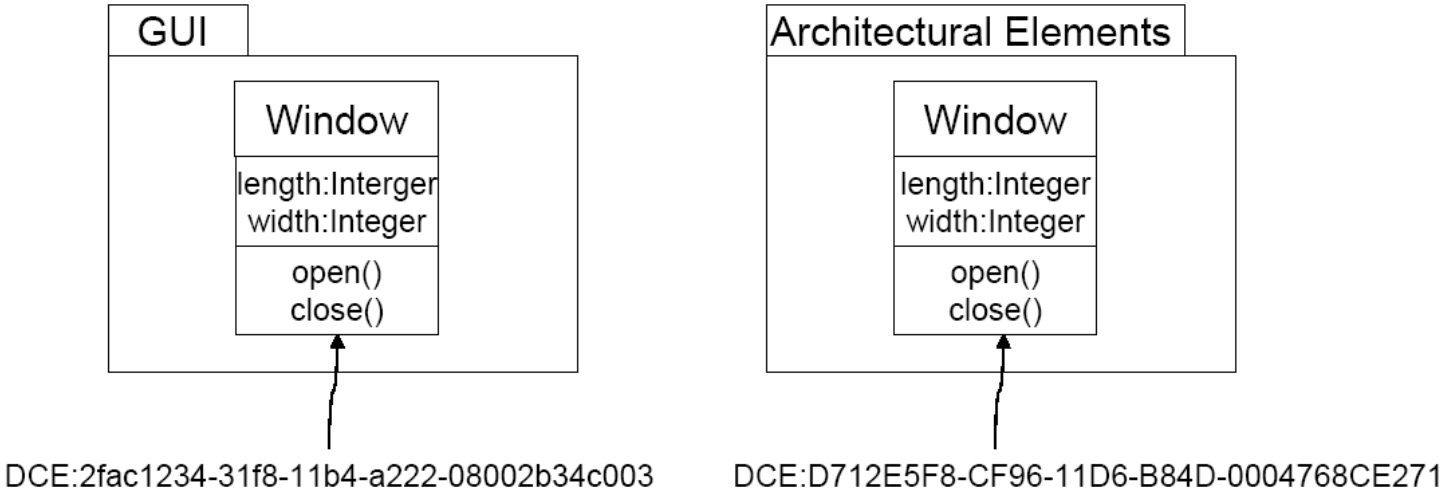
Underlying Model Structure

- Features may be ordered (sequences) or unordered (sets)



Identification Elements

- We assume that each model element has a universally unique, stable identifier
- We match elements by identity, not similarity



Representing Differences as Elementary Operations

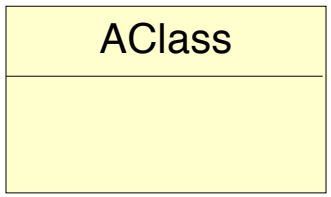
- The difference of two models is not a model!

Operation	Dual
new(e,t)	del(e,t)
del(e,t)	new(e,t)
set(e,a,v_0,v_1)	set(e,a,v_1,v_0)
insert(e_0,f,e_1)	remove(e_0,f,e_1)
remove(e_0,f,e_1)	insert(e_0,f,e_1)
insertAt(e_0,f,e_1,i)	removeAt(e_0,f,e_1,i)
removeAt(e_0,f,e_1,i)	insertAt(e_0,f,e_1,i)

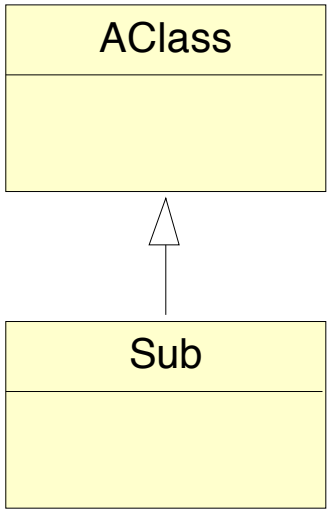
Basic Difference Algorithm

- M_{old}-M_{new}=Δ=[N,U,D]
- 1. Define a mapping from elements in M_{old} to M_{new}
- 2. Define a new operation for each element in M_{new} that does not exist in M_{old}
- 3. Define Update Operations
  - attributes: define a set operation
  - unordered features: insert,remove
  - ordered features: Longest Common Subsequence algorithm
- 4. Define a delete operation for each element in M_{old} that does not exist in M_{new}

Example

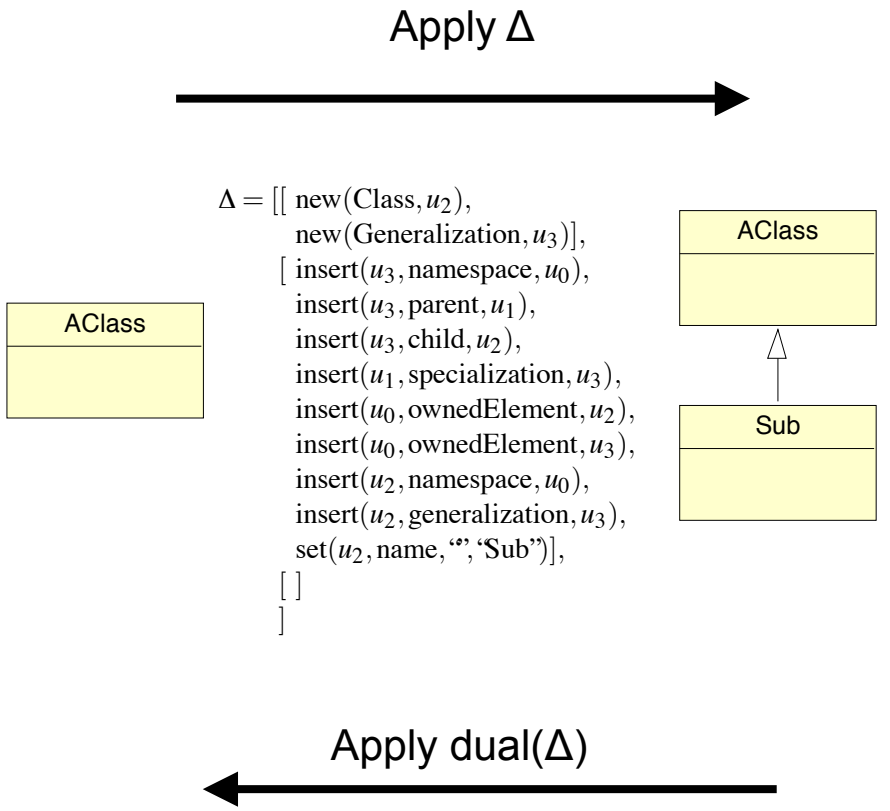


$$
\begin{aligned}
\Delta = [[\ & \mathrm{new}(\mathrm{Class}, u_2),\\
& \mathrm{new}(\mathrm{Generalization}, u_3)],\\
[\ & \mathrm{insert}(u_3, \mathrm{namespace}, u_0),\\
& \mathrm{insert}(u_3, \mathrm{parent}, u_1),\\
& \mathrm{insert}(u_3, \mathrm{child}, u_2),\\
& \mathrm{insert}(u_1, \mathrm{specialization}, u_3),\\
& \mathrm{insert}(u_0, \mathrm{ownedElement}, u_2),\\
& \mathrm{insert}(u_0, \mathrm{ownedElement}, u_3),\\
& \mathrm{insert}(u_2, \mathrm{namespace}, u_0),\\
& \mathrm{insert}(u_2, \mathrm{generalization}, u_3),\\
& \mathrm{set}(u_2, \mathrm{name}, \text{“”}, \text{“Sub”})],\\
[\ &]\\
]
\end{aligned}
$$

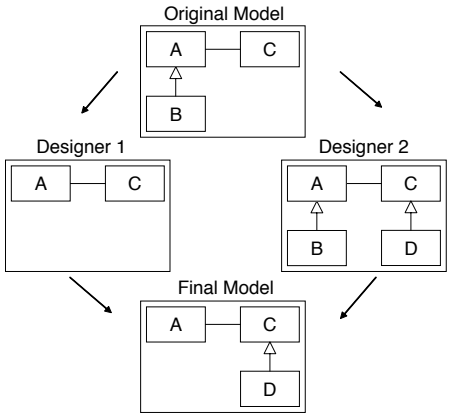


Basic Merge Algorithm

- Δ=[N,U,D], $M_1 + \Delta = M_2$
  - 1. Apply all new operations
  - 2. Apply all update operations
  - 3. Apply all delete operations



Union of Two Models



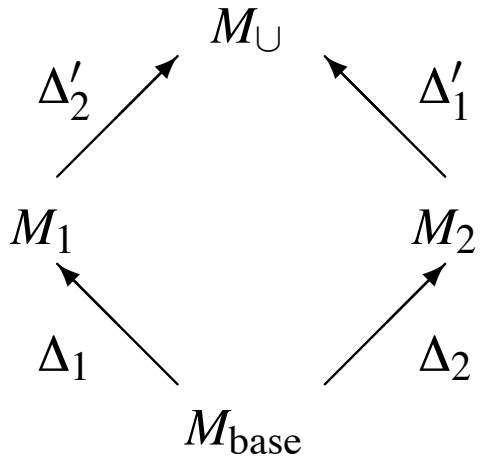
- Assume two developers update a model simultaneously
  - M_1-M_{base} = Δ_1, M_2-M_{base} = Δ_2
- We want to merge both changes into the base model- However, the order of application of differences matters
  - $(M_{base} + \Delta_1) + \Delta_2 \neq (M_{base} + \Delta_2) + \Delta_1$

Example

$$
\begin{array}{ccc}
 & [A,C] & \\
F_1 = [\text{insertAt}(B,1)] \swarrow & & \searrow F_2 = [\text{insertAt}(D,2)] \\
[A,B,C] & & [A,C,D] \\
F_2 = [\text{insertAt}(D,2)] \downarrow & & \downarrow F_1 = [\text{insertAt}(B,1)] \\
[A,B,D,C] & & [A,B,C,D]
\end{array}
$$

Union of Two Models

- Solution: a difference minimization operator ⊗
  - M_1-M_{base} = Δ_2, M_2-M_{base} = Δ_2
  - Δ_2' = ⊗(Δ_2, Δ_1), Δ_1' = ⊗(Δ_1, Δ_2)
  - Conflict resolution: (M_{base} + Δ_1) + Δ_2' = (M_{base} + Δ_2) + Δ_1'
  - Δ_1' may contain less operations than Δ_1



Conflict Resolution

- Model Elements
  - new,del: not possible since ids are unique
  - del,del: automatic resolution (no actual conflict)
  - del, update: manual or metamodel specific resolution
- Attributes
  - set,set: manual or metamodel specific resolution if values are different
- Features
  - update, update: automatic resolution in some cases based on longest common subsequence, manual or metamodel specific if update breaks multiplicity constraints

Ordered Features

$$
\begin{array}{ccccc}
 & & [A,C] & & \\
F_1 = [\text{insertAt}(B,1)] & \swarrow & & \searrow & F_2 = [\text{insertAt}(D,2)] \\
 & & [A,B,C] \quad [A,C,D] & & \\
F_2' = [\text{insertAt}(D,3)] & \searrow & & \swarrow & \\
 & & [A,B,C,D] & &
\end{array}
$$

A Complete Model Union System

1. Model Difference calculation
2. Difference minimization
   - Automatic metamodel-independent conflict resolver
     - Algorithm presented in the article
   - Automatic metamodel-dependent conflict resolver
     - Uses knowledge specific to each modeling language
   - Manual resolution
     - Requires a proper difference visualization and inspection
3. Model union calculation

Limitations

- Unique, stable, universal model element identifiers across model revisions
- Alternative: element matching based on similarity
  - U. Kelte, J. Wehren, J. Niere, A Generic Difference Algorithm for UML Models

Limitations

- Metamodel is assumed to be static
- This is not a realistic assumption, specially for domain-specific modeling languages
  - B. Gruschko, D. Kolovos, R. Paige, Towards synchronizing models with evolving metamodels

Limitations

- Metamodel independent algorithms are generally applicable but do not exploit useful knowledge of the modeling language in the mapping, conflict resolution and merge steps
  - S. Nejati et al., Matching and Merging of Statecharts Specifications
  - J. Küster, C Gerth, A. Förster, G. Engels, Detecting and Resolving Process Model Differences in the Absence of a Change Log

Limitations

- Standardized representation of differences is not discussed
  - A. Cicchetti, D. Di Ruscio, and A. Pierantonio, A Metamodel Independent Approach to Difference Representation
- Manual conflict resolution requires inspection of base model and differences
  - H. Störrle, Making sense to modelers: Presenting UML class model differences in prose

Limitations

- No actual tools are discussed
  - Y. Lin, J.Gray, F. Jouault, DSMDiff: a differentiation tool for domain-specific models
  - C. Brun, A. Pierantonio, Model Differences in the Eclipse Modelling Framework, Eclipse EMF Compare

Future work (in 2013)?

- Better conflict detection
- Better difference reporting
- Better tool support for multiple languages
- Better conflict resolution

- K. Altmanninger, P. Brosch, G. Kappel, P. Langer,M. Seidl, K. Wieland, M. Wimme, Why Model Versioning Research is Needed!? An Experience Report

10 year most influential paper?

- It describes a problem and solves it
  - The problem is relevant and general
- It also describes, sometimes implicitly, many other interesting problems to solve
  - It brings the problem to the attention of a research community

10 year most influential paper?

- The right time
  - Anybody could have stumbled into and solved the problem before us
- The right forum
  - Alanen and Porres, Difference and union of models, Proc. of UML 2003, Springer: **240** citations
  - Alanen and Porres, Version control of software models, Chapter in Advances in UML and XML-Based Software Evolution, 2004, Idea Group Publishing: **9** citations

10 year most influential paper?

- Would it have been accepted to MODELS 2013?
  - Of course not, it was published 10 years ago
  - No validation, experimental tool, evaluation results

The 2003 debate, 10 years later

"Models are sketches" Martin Fowler: What is the Point of the UML?	vs	"Models are formal artifacts" Jean Bézivin, MDA: From Hype to Hope, and Reality

- We were right (all of us)!
  - UML is the lingua franca for software design sketches
  - Empirical studies show great success in the use of UML and domain-specific languages in the industry.

Acknowledgments

- Dr. Marcus Alanen
- Prof. Ralph Johan Back, Prof. Johan Lilius
- The MODELS community