Issues and Challenges in Model Versioning

Tutorial at iiWAS2009/MoMM2009
(Kuala Lumpur, Malaysia)

Dipl.-Ing. Kerstin Altmanninger
(Presenter)
Dipl.-Ing. Kerstin Altmanninger

Johannes Kepler University Linz
Department of Telecooperation (TK)
Altenberger Straße 69, 4040 Linz, Austria

Phone: +43 (0)70/2468-9262
Fax: +43 (0)70/2468-9829
Email: kerstin.altmanninger@jku.at
WWW: http://www.tk.jku.at

Projects:
- ModelCVS: A Semantic Infrastructure for Model-based Tool Integration
- AMOR – Adaptable Model Versioning

PhD thesis:
Models in Conflict – A Semantically Enhanced Version Control System for Model Artifacts
Outline

- Introduction to Model-driven Engineering
- Features of Version Control Systems
  - Top-level Features
  - 3-way Merge Features
- Challenges in Model Versioning
  - Existing Systems
  - Requirements
  - Issues
- Discussion
- AMOR Project
- Literature
Introduction to
Model-driven Engineering
Problem #1: Problem Area vs. Solution Area

**Problem**
- How do analyze models associate with code?
- How does changes in the code affect the model?
- Are models consistent with the code?
- …

Traditional Development

Analysis ➔ Design ➔ Implementation ➔ Test ➔ ...

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger
Problem #2: Modeling as a Necessary Evil

- »When it comes down to it, the real point of software development is cutting code«
- »Diagrams are, after all, just pretty pictures«
- »No user is going to thank you for pretty pictures; what a user wants is software that executes«

Introduction to Model-driven Engineering

Solution: Model-driven Engineering (MDE)

Model-driven Engineering

Model-Driven Architecture (MDA)

Platform Independent Model (PIM)

Platform Specific Model (PSM)

Analysis

Design

Implementation

Domain-Specific Modeling (DSM)

Model

Code

OMG, ArcStyler, OptimalJ, Executable UML, AndroMDA, …

MetaCase, Eclipse Modeling Framework, OSLO, Software Factories, WebRatio, …

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger
Effect #1: Evolution of Modeling

- **Models as sketch**
  - Communication of ideas/alternatives
  - Goal: Modeling

- **Models as templates**
  - Documentation of design decisions
  - Goal: Implementation guide

- **Models as programs**
  - Generation of applications automatically
  - Goal: Models are source code and vice versa
Effect #2: Model as the Central Artifact

- Static Analysis
- Dynamic Analysis
- Documentation
- Rapid Prototyping
- Refactorings/Transformations
- Code Generation
- Versioning
- Automatic Testing

Introduction to Model-driven Engineering
Effect #3: Models are more than just Pictures

- Foundation: Formal language definition
  - Modeling languages: MOF, Ecore, ...

- Automatic model processing (e.g., PIM -> PSM, refactoring)
  - Model transformation languages: QVT, ATL, ...

- From the model to software
  - Code generation languages: JET, Xpand, ...

- Persistent model storage and model exchange
  - Model serialization/deserialization: XML
Features of Version Control Systems
Overview of Version Control Systems

- Part of Software Configuration Management

- **Basic components:**
  - Repository
  - (Software) Artifacts => versions (revisions vs. variants)

- **Purpose:**
  - Historical archiving and management of changes in (software) artifacts
    - retrieving old revisions
    - keeping track of changes
  - Supports collaboration
Top-level Features

Version Control System

- Collaboration
  - Pessimistic
  - Optimistic

- Merge
  - Raw
  - Two-way
  - Three-way

- Repository Architecture
  - Centralized
  - Distributed

- Branching
  - Implicit
  - Explicit

- Comparison
- Conflict Detection
- Conflict Resolution
- Merge
Pessimistic Versioning

- Lock-modify-unlock paradigm

**Disadvantages:**
- Prohibits parallel working
- Administration overhead
- False sense of security
Features of Version Control Systems

Optimistic Versioning

- **Copy-modify-merge paradigm**
  - **Advantage:** Saves time
  - **Disadvantage:** Result depends on quality of the merge
Features of Version Control Systems

**Raw vs. Two-way vs. Three-way**

(a) Raw

(b) Two-way

(c) Three-way

---

**Issues and Challenges in Model Versioning**

Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger

16/77
Centralized vs. Distributed

- **Centralized**
  - Users access a master repository via a client
  - Edited working copies must be committed before they can be propagated to other VCS users
  - Version history is only available in the master repository

- **Distributed**
  - Each user has an own repository
  - Repositories can be synchronized optionally with others
  - Version history is distributed on the different repositories
Branching Variants

- Implicit/Explicit
- Used due to different intentions
  - Version Management
    - software product releases
  - Variant Management
    - software product variants
    - common core
  - Decentralized Version Control Systems
  - Resolution Support
    - storage of the intermediate versions before a merge
    - to retrieve information about differences, conflicts and resolution patterns
3-way Merge Features

Features of Version Control Systems

Comparison

Conflict Detection

Conflict Resolution

Merge

Conflict Resolution

Comparison

Merge

Conflicts Detection

Features of Version Control Systems

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger

19/77
Comparison Phase

Features of Version Control Systems

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger

20/77
State vs. Change

State-based

- **Input:**
  - Two versions of an artifact

- **Advantage:**
  - Independent of a development environment / editor

- **Disadvantage:**
  - Matching techniques needed

- **Conflict detection** is computed on the resulting difference sets

Change-based

- **Input:**
  - Two versions of an artifact
  - Change protocol (log file)

- **Advantage:**
  - Composite operations available

- **Disadvantage:**
  - Bounded to a development environment / editor

- **Conflict detection** is computed on the logged changes
Features of Version Control Systems

UUIDs vs. Heuristics

- Matching approaches particularly important for state-based versioning
- Needed to identify the artifact’s elements occurring in both versions

**Universally Unique Identifiers (UUIDs)**

- All elements in an artifact need to provide a UUID
- Newly created elements must be assigned with a UUID
- UUIDs must not be modified until the deletion of the element
- Two elements occurring in different versions with the same UUID are considered as identical

**Advantages:**
- Easiest approach to realize a matching
- Elements can still be matched, although they have been changed fundamentally
- Unique matching

**Disadvantage:**
- Assignment of UUIDs needed
### UUIDs vs. Heuristics

**Heuristics**
- May be based on structural similarities of two artifact versions
- Often use metrics from information theory
- **Advantage:**
  - flexible
- **Disadvantage:**
  - matching might not be unique
  - performance may be an issue

**UUIDs and Heuristics can be combined!**
Line vs. Tree vs. Graph

Line-based
- Any kind of artifacts

Tree-based
- HTML, XML file
- Latex document

Graph-based
- Model artifacts
- Program source code

Features of Version Control Systems
- Basis
- Match
- Representation
- Granularity
- Differences
Fixed vs. Configurable Granularity

- Also known as version granularity, delta granularity or unit of conflicts
- Can be applied on any granularity level

Features of Version Control Systems

- Basis
- Match
- Representation
- Granularity
- Differences
Outcome of the Comparison Phase

Resulting set of difference can be classified according to two orthogonal dimensions

- Dependency on an underlying language
- Divisibility of a change operation

Features of Version Control Systems

• Basis
• Match
• Representation
• Granularity

Differences
Features of Version Control Systems

Conflict Detection Phase

- Textual
- Graphical

Conflict Detection

Conflicts

Visualization

Presentation

Textual

Graphical

Grouping
Conflic Detection Phase

Features of Version Control Systems

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Contradicting Intention

Same Intention

Text
Syntax
Semantics

Static
Behavioral
Linguistic
Equivalence

ADD
DEL
UPD

ADD n.a. n.a.
DEL n.a. –
UPD n.a.

Semantics

Equivalence
Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Features of Version Control Systems

Textual Conflict

- Syntax
- Semantics
- Equivalence
Syntactic Conflict

- Detected by a structural comparison of artifacts, which are represented either in a tree or graph reflecting the syntax of the artifact’s language.«

- Syntactic conflicts are resulting from modifications, which produce a violation of the language’s syntax when merged.«
Semantic Conflicts

- Some conflicts cannot be detected by a structural comparison due to:
  - Semantic meaning of linguistic expressions
  - Syntactic different parts have been edited which may
    - violate language constraints or
    - the execution behavior if merged

- Formal semantics are needed to detect such kinds of conflicts

- Formal semantics consists of three parts:
  - Syntax of a language
  - Semantic domain
  - Semantic mapping
Static Semantic Conflict

- Occurs due to violation of language constraints
- In modeling e.g., by violation of relationships, context conditions, consistency or violation of integrity
Behavioral Semantic Conflict

- Occurs due to concurrent changes of the execution behavior

- E.g., concurrent changes on the data or control flow may yield to behavioral side effects

Features of Version Control Systems

- Text
- Syntax
- Semantics
- Equivalence
Linguistic Semantic Conflict

Occurs e.g., if two VCS users edit the same artifact with the same intentions but utilized different literal designations.
Conflict due to Equivalence

May occur due to utilization of
- semantically equivalent concepts
- refactorings
Support for semi-automatic resolution

- Dependency analysis to identify resolution orders
- Resolution patterns/policies
Challenges in Model Versioning
Existing Systems

Alienbrain
OpenCVS
AllChange
LibreSource
git
Bazaar
AllFusion Harvest
EVS
Darcs

Audodesk Vault
GNU arch
CVS
Aldon
FileHamster
SVK
Aegis
Codeville
CVSNT

Subversion
FirePublish
Monotone
Mercurial
CollabNet
DVCS
Evolution
ClearCase
Fossil

BitKeeper

Challenges in Model Versioning

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger
# Survey Result about available Model VCSs

| Approaches         | Generic | State | Change | UUID | Heuristics | Line | Tree | Graph | Configurable | Representation | Granularity | Conflicts | Textual | Syntactic | Composite Op. | Equivalence | Static | Behavioral | Textual | Graphical | Grouping | Visualization | Presentation | Automation | Dependencies | Recommendations | Validation |
|--------------------|---------|-------|--------|------|------------|------|------|-------|-------------|---------------|-------------|------------|---------|---------|---------|-------------|-------------|--------|------------|---------|----------|----------|---------------|--------------|------------|-------------|---------------|------------|
| Subversion         | +       | +     | −      | +    | −          | +    | −    | −     | +           | −             | −           | +          | +        | −        | −        | −           | −           | +      | +          | −        | −         | −        | −             | −            | −          | −           | −             | −          |
| RSA                |         | −     | −      | +    | +          | −    | −    | −     | −           | −             | −           | +          | −        | −        | −        | −           | −           | +      | +          | −        | −         | −        | −             | −            | −          | +           | +             | −          |
| EMF Comp.          |         | +     | +      | +    | −          | −    | −    | −     | −           | −             | −           | −          | +        | −        | −        | −           | −           | +      | −          | −        | −         | −        | −             | −            | −          | +           | +             | −          |
| Alanen & Porres    | +       | +     | −      | +    | −          | +    | −    | −     | −           | −             | −           | +          | −        | −        | +        | −           | −           | +      | −          | −        | −         | −        | −             | −            | −          | +           | +             | −          |
| Unicase            | −       | −     | +      | +    | −          | −    | −    | −     | −           | −             | −           | +          | −        | −        | −        | −           | −           | +      | −          | −        | −         | −        | −             | −            | −          | +           | +             | −          |
| Oda & Saeki        |         | +     | +      | +    | −          | −    | −    | +     | −           | +             | −           | −          | −        | −        | −        | −           | −           | +      | −          | −        | −         | +        | −             | −            | −          | −           | −             | −          |
| CoObrA             |         | +     | +      | +    | +          | −    | +    | −     | −           | +             | −           | +          | −        | −        | −        | −           | −           | +      | −          | −        | −         | −        | −             | +            | −          | −           | −             | −          |
| Cicchetti et al.   |         | −     | +      | +    | −          | −    | +    | −     | −           | +             | −           | −          | −        | −        | −        | −           | −           | +      | −          | −        | −         | −        | −             | −            | −          | −           | −             | −          |
| Odyssey-VCS        |         | +     | −      | −    | −          | −    | −    | −     | −           | +             | −           | −          | −        | −        | −        | −           | −           | +      | −          | −        | −         | −        | −             | −            | −          | −           | −             | −          |
| SMoVer             | +       | +     | +      | −    | −          | +    | +    | −     | −           | +             | −           | +          | −        | −        | −        | −           | −           | +      | −          | −        | −         | −        | −             | −            | −          | −           | −             | −          |

**Legend:**
- **Supported**
- **partly supported**
- **not supported**
- **not applicable**
Overall Challenges

- **General**
  - Mismatch in representation
  - Bounded to modeling environment
  - Inflexible in modeling language
  - Too generic
  - Various serialization variants
  - Disregard of dependencies between artifacts

- **Comparison Phase**
  - Non-configurable

- **Conflict Detection**
  - Erroneous conflict detection
    - Missing interpretation of semantics of artifacts
    - Missing information about edit operations in state-based systems
  - Unsatisfactory conflict visualization

- **Unsupportive Conflict Resolution**

- **Inconsistent Merged Model Versions**
Why not using Subversion with Unix diff? (1/2)
Why not using Subversion with Unix diff? (2/2)

Mismatch in representation!
Specific vs. Generic VCSs for Model Artifacts

- How can a standalone VCS for model artifacts be realized without depending on the traces of changes on the artifact, performed by modelers?
- How can a VCS for model artifacts be realized without depending on a concrete modeling language?
- How can such a system be generic but provides accurate conflict detection and supportive conflict resolution support?
For standalone VCSs the versions of model artifacts need to be exchanged between the modeling environment and the VCS

Modeler may use different modeling environments to edit model artifacts from the repository

**Problems:**
- Modeling environments may export different XMI variants (those XMI representations of a model artifact may vary considerably)
- Modeling environments may utilize divergent metamodels of a modeling language (e.g., various UML implementations exist)

Summing up those cannot be utilized in the VCS right away

**Solution:** Adapters to import different XMI representations are indispensable
Dependencies between Artifacts

- Artifacts in a repository may depend on each other (may have communalities)
- If e.g., a class is renamed in a UML class diagram this has an effect on the UML Sequence Diagram instantiating this class
Comparison Phase

- On which granularity level should the comparison be performed?
- For generic VCSs the granularity level should be configurable for specific languages and by VCS users. How can this be realized?
- How can composite operations (like refactorings) be detected in state-based systems?
- Which information is needed for the following phases conflict detection and resolution?
Conflict Detection Phase

- How can accurate conflict detection be achieved?
  - avoid falsely indicated conflicts
  - find undetected conflicts
- How can conflicts be presented?
- What should be presented to the user?
  - Tentative merge
    (apply only non-conflicting changes)
  - Left change, right change, resolution patterns
Conflict Resolution Phase

- Which influence has the resolution order, in which conflicts are resolved, to the overall effort to combine two versions?
- How can a VCS user be assisted in the resolution of conflicts by the system?
- Can recurring conflict resolution activities by users be prevented?
- If VCSs can learn from conflict resolution activities, how can rules be applied on similar conflicting situations?
Merge Phase

- How can it be ensured that only valid models are saved in the repository?
- How can it be prevented that information gets lost after the merge of two artifacts?
Discussion
Discussion

Questions

- Are there more challenges and issues?
- What are your expectations on Version Control Systems for models?
- Do we really need Version Control Systems dedicated for model artifacts?
- Versioning habits?
- ...
Questionnaire

http://www.modelversioning.org

Background

Which role do you mainly play in projects?
- manager
- developer
- architect
- tester
- others

Which other roles do you occasionally have in projects?
- manager
- developer
- architect
- tester
- others

How is your team geographically distributed?
- in the same building
- in the same town
- in the same time zone
- all over the world

How many people typically participate in your projects?
- up to 5 persons
- up to 20 persons
- up to 100 persons
- more

Versioning Habits

What version strategy do you apply?
- pessimistic (lock/modify/unlock)
- optimistic (modify/merge)
- none at all
AMOR Project
Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger

Research Objectives

- **General**
  - Mismatch in representation
  - Bounded to modeling environment
  - Inflexible in modeling language
  - Too generic
  - Various serialization variants
  - Disregard of dependencies between artifacts
- **Comparison Phase**
  - Non-configurable
- **Conflict Detection (CD)**
  - Erroneous conflict detection
    - Missing interpretation of semantics of artifacts
    - Missing information about edit operations in state-based systems
  - Unsatisfactory conflict visualization
- **Unsupportive Conflict Resolution (CR)**
- Inconsistent Merged Model Versions
Partners

- AMOR: Adaptable Model Versioning
  - FFG FIT-IT Semantic Systems Project

Industrial Partner:

Academic Partners:

- Business Informatics Group (BIG)
  Institut für Software Technik und Interaktive Systeme

- Arbeitsgruppe Information Systems (IFS)
  Institut für Bioinformatik

- Institut für Telekooperation (TK)
Conceptual Architecture of AMOR

**Front-End**
- **Modeling Tools**
  - UML
  - Metamodel
  - Model
- **Enterprise Architect**
- **Modeler**
- **Versioning Assistant**
  - Stand-alone (generic)
  - Plug-in (optional)
- **Editing Observer** (optional)
- **Editing-Log**
- **AMOR Admin**

**Back-End**
- **Components**
  - Model Repository
  - Conflict Resolution Reasoner
    - Resolution Operation Storage
    - Pattern Miner
- **Advanced Conflict Detector**
  - Semantic View Editor
  - Semantic View Base
  - Operation Conflict Rule Editor
  - Operation Conflict Base
- **Conflict Resolution**
- **Process and Artifact**
  - New Version
  - Merge
    - Generic Merge Algorithm
    - (all conflicts resolved)
  - Resolution Decisions
    - Possibilities: keep mine, use other, ignore, update
  - Conflict Report
    - Semantic View Reports (optional)
    - Operation-based Report (optional)
  - Change Report
    - Semantic-based (optional)
    - Operation-based (optional)
    - State-based (generic)

**AMOR Project**

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

56/77

Dipl.-Ing. Kerstin Altmanninger
Configurable Comparison – Motivation

Advantages:
- Configurable for specific user desires
- Configurable for specific modeling languages
By inspecting the structural features, namely
  • the attributes and
  • references of a model element
one can determine whether the model element as a whole has been updated

Particularly four different strategies for the detection of structural changes in a graph that are of interest for conflict detection can be identified
  • Attribute update (ATT)
  • Reference update (REFS)
  • Role update (ROL)
  • Referenced element update (REF)

By setting of those strategies on model elements and properties configurable comparison can be achieved
Architecture of Semantic Conflict Detection

Basic VCS Phases

Model Versions

(Variable) Modeling Language Specification

Conceptual Realization

Basic VCS

Model $V^*_{syn}$

Model $V^*_{sem}$

Model $V''_{sem}$

Model $V''_{syn}$

Comparison

Conflict Detection

Conflict Resolution

Comparison

Semantic Views

Metamodel

Abstract Syntax

Semantic Aspects

M3 (Ecore)

Equivalent Concepts

Static Semantics

Behavioral Semantics

Transformation

View Definition Metamodel

Semantic View Definitions

Semantic Enhancement

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger
Example of Semantic Conflict Detection
Operation Recorder – Motivation

- Detection of **composite operations** and **refactorings** enables…
  - More accurate conflict detection
  - Smart conflict resolution
- Composite operations in modeling environments (accomplished by experts)
  → Fixed, pre-defined composite operations
- Hardly possible to predefine all relevant composite operations
  → Highly valuable to enable “usual modelers” to specify composite operations
- Modelers have domain knowledge
  → But, usually no knowledge on model transformation techniques
Issues and Challenges in Model Versioning

Operation Recorder

EMF Ecore based models

EMF Compare

EMF Ecore based models

Any Editor

Extended OCL

EMF Ecore based models
Supportive Conflict Resolution

- **Goal**
  - Accelerate conflict resolution
  - Minimize errors

- **Proposed Solution**
  - Infer general resolution strategies from accumulated historical data
  - Provide enhanced resolution support for repetitive conflict situations
  - Apply collaborative merge in order to capture all user intentions
Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

Architecture for Conflict Resolution

AMOR Project

Dipl.-Ing. Kerstin Altmanninger
Conflict Resolution: Motivation Example

V* (Merged Version)
... but sometimes this will not be enough

... then Lisa and Bart must resolve the conflict together
Why Collaborative Conflict Resolution?

Automatic resolution is not possible here!
Why Collaborative Conflict Resolution?

→ Higher semantics
→ Better quality
Collaborative Conflict Resolver

1. Validation: Conflict - Multiplicity of composition "And" may not be.
2. Resolution Strategies:
   - Apply all changes
   - Apply my changes
   - Custom rule to merge

3. Resolution Advisor
4. Resolve
5. Accept and Commit

AMOR Project

Issues and Challenges in Model Versioning Tutorial@iiWAS2009/MoMM2009

Dipl.-Ing. Kerstin Altmanninger
Summary: AMOR Workflow (1/2)

Issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

AMOR Project

Comparison

Conflict Detection

V'
V
V''

Semantic View Definition Repository

Semantic View Definitions

Semantic View Specification

Operation Repository

Operations

Operation Specification
Summary: AMOR Workflow (2/2)

issues and Challenges in Model Versioning
Tutorial@iiWAS2009/MoMM2009

AMOR Project

Conflict Resolution

Merge

Resolution Lookup

Recommendations

Pattern Storage

Resolution Pattern

Resolution Reasoning

Resolution

Single

Semi-automatic Resolution

Manual Resolution

Collaborative

Semi-automatic Resolution

Manual Resolution

\[ \text{V}^* \]
Literature
Model Driven Development

- **Books**
  - IEEE Computer, Special Issue on Model Driven Engineering. February 2006
  - Cover Feature: Model Driven Engineering by Douglas Schmidt, Vanderbilt University

- **Websites**
  - [www.codegeneration.net/](http://www.codegeneration.net/)
  - [www.metacase.com](http://www.metacase.com)
  - [www.planetmde.org](http://www.planetmde.org)
  - [www.omg.org/mda/](http://www.omg.org/mda/)
  - [www.modelware-ist.org](http://www.modelware-ist.org)
Versioning (1/2)

- Articles
Versioning (2/2)

- Conference & Workshop Papers
AMOR Project


Thank you for your attention!

http://smover.tk.uni-linz.ac.at

http://www.modelversioning.org