Karlsruhe Institute of Technology

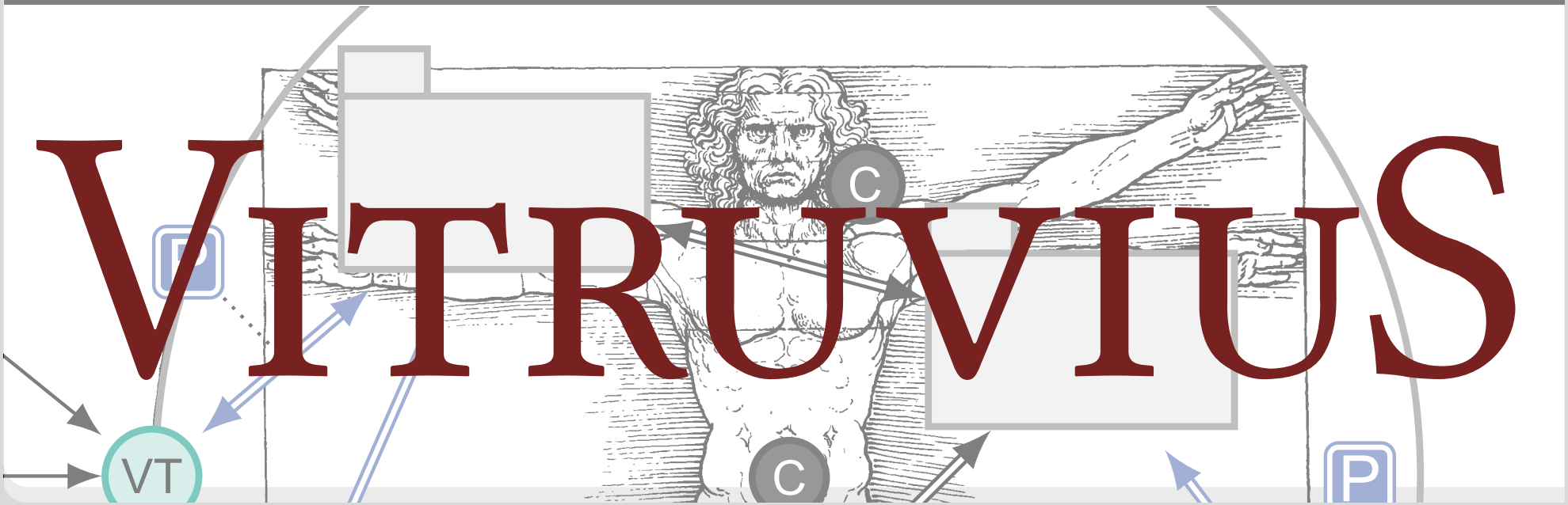
SDQ Software Design and Quality

Synchronizing Heterogeneous Models in a View-Centric Engineering Approach

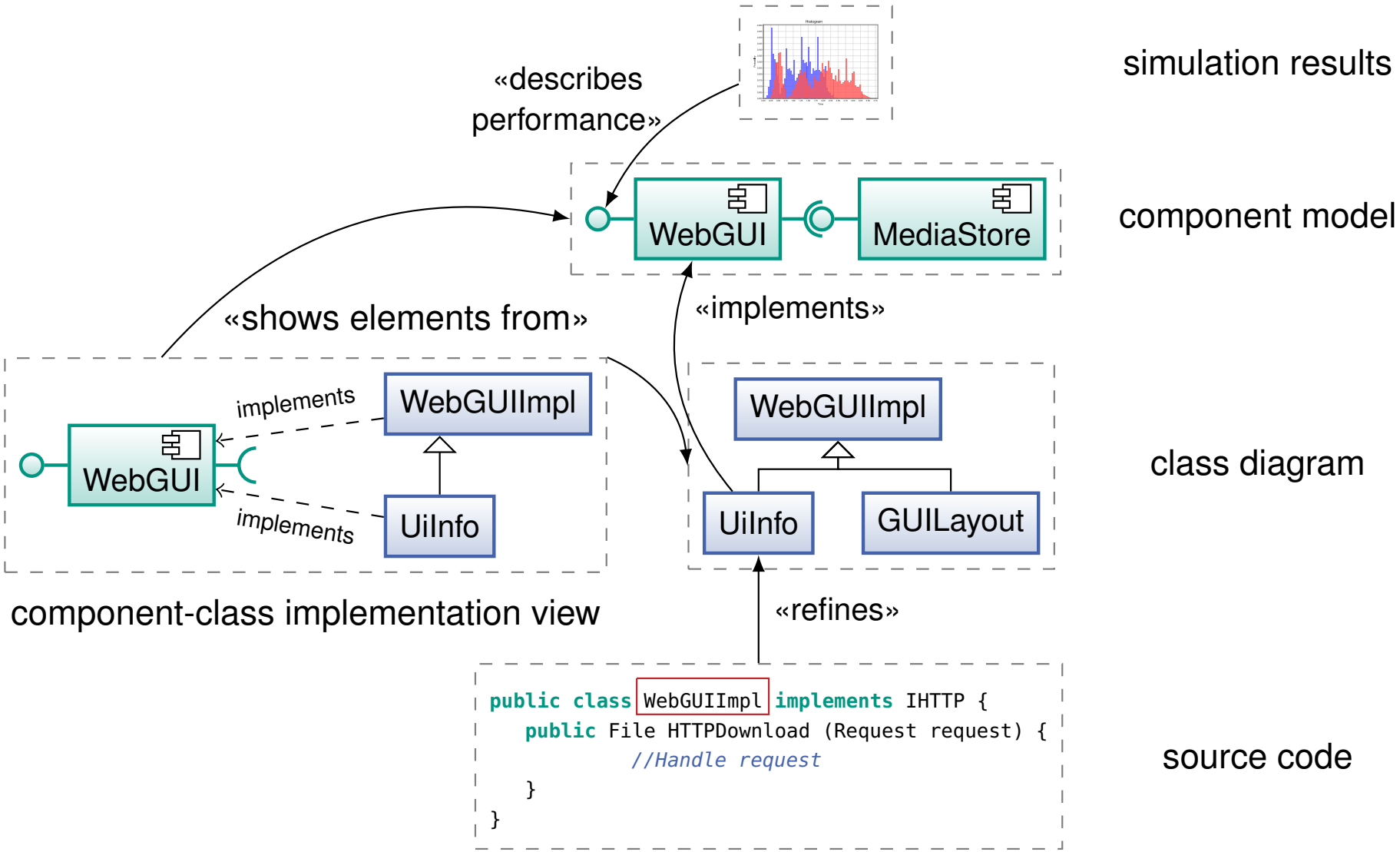
Doctoral Symposium, Conference on Software Engineering

Max E. Kramer | Kiel, Germany | 26.2.2014

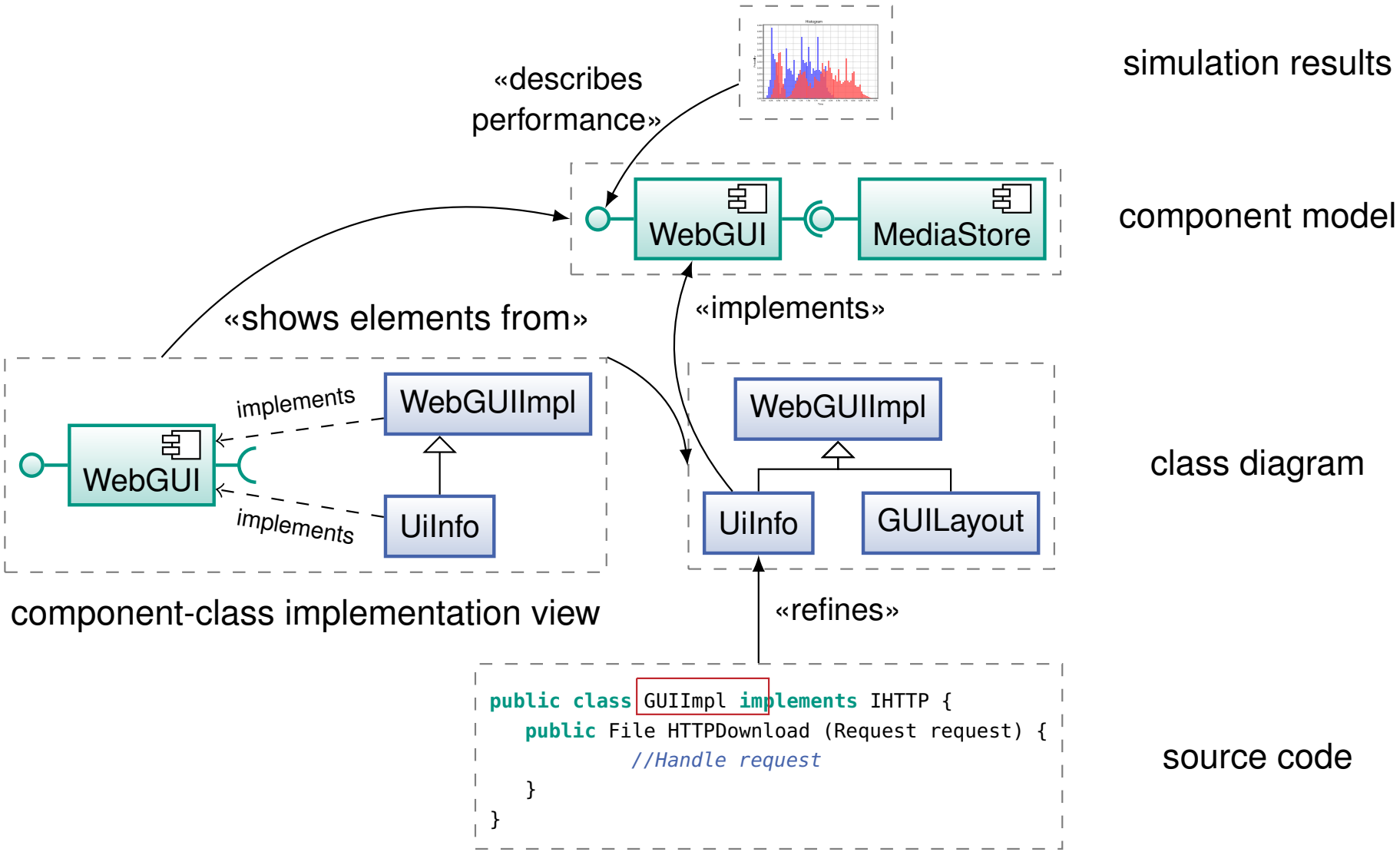
Chair for Software Design and Quality



Motivating Scenario



Motivating Scenario



Problems

Fragmentation

- information is spread over heterogeneous models

Inconsistency

- conflicting redundant information or lost information
- violated constraints may be implicit or even unknown

Complexity

- manual synchronization is time-consuming and error-prone
- bidirectional sync transformations involve accidential complexity

Proposed Solution

Problem

- keeping information consistent across heterogeneous models is hard

Idea

- put models & code in a modular single underlying model (MSUM)
- generate sync transformations from declarative mapping specifications
- synchronize early, synchronize often and listen to *changes of users*

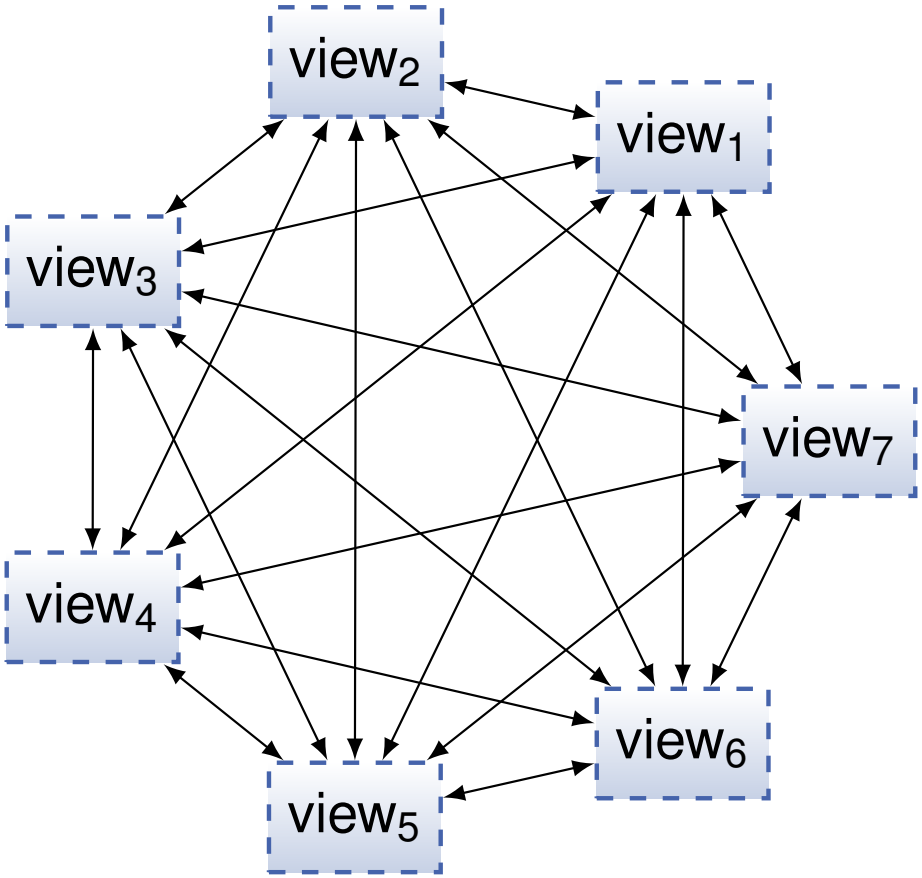
Benefit

- reduce accidental complexity through abstraction (maintain expressivity)

Action

- develop a domain-specific language and generator for sync transformations
- build and integrate it into a framework for view-centric engineering

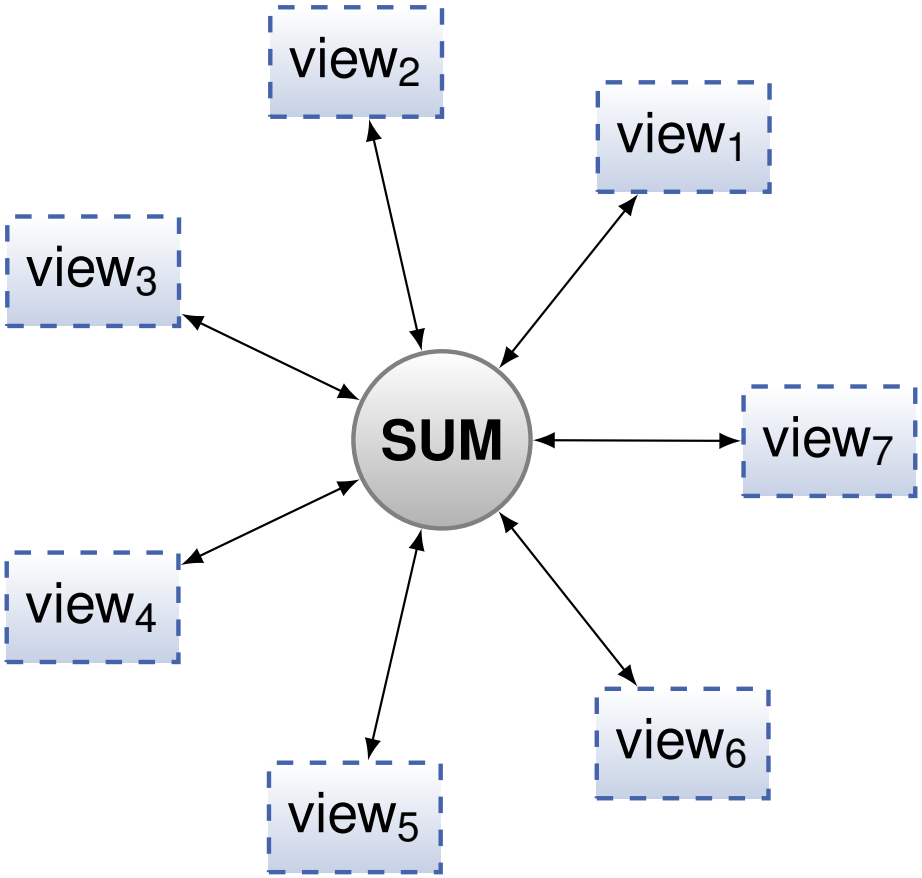
Synthetic vs. Projective Approaches



Synthetic Approaches

- (Eramo et al. 2008, Romero et al. 2009) bidirectional QVT-R transformations for UML
- (Giese et al. 2006, 2009, 2010) incremental synchronization with Triple Graph Grammars (TGGs)
- ...

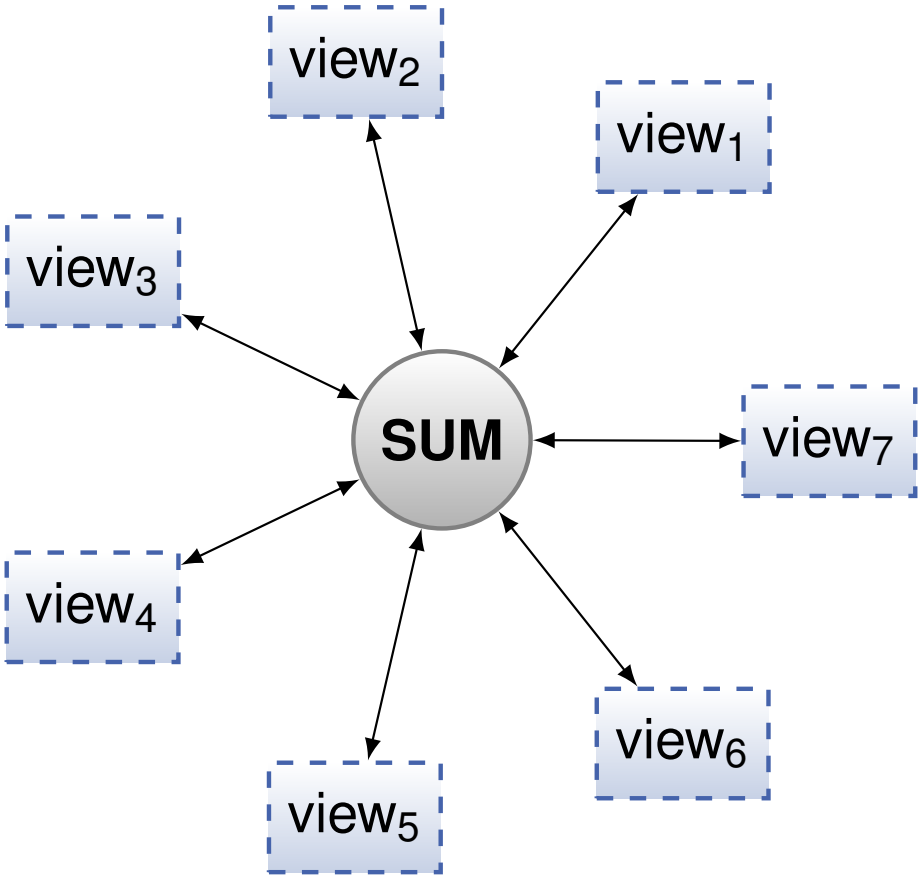
Synthetic vs. Projective Approaches



Projective Approaches

- *Orthographic Software Modeling* (Atkinson et al. 2010)
  - single underlying model (SUM)
  - exclusive manipulation from views
  - project-specific SUM metamodel
- (Cicchetti et al. 2011, 2012) incremental synchronization using HOTs and diffing
- . . .

Synthetic vs. Projective Approaches



Tool Integration

- (Hein et al. 2009)
  ModelBus, tool coupling

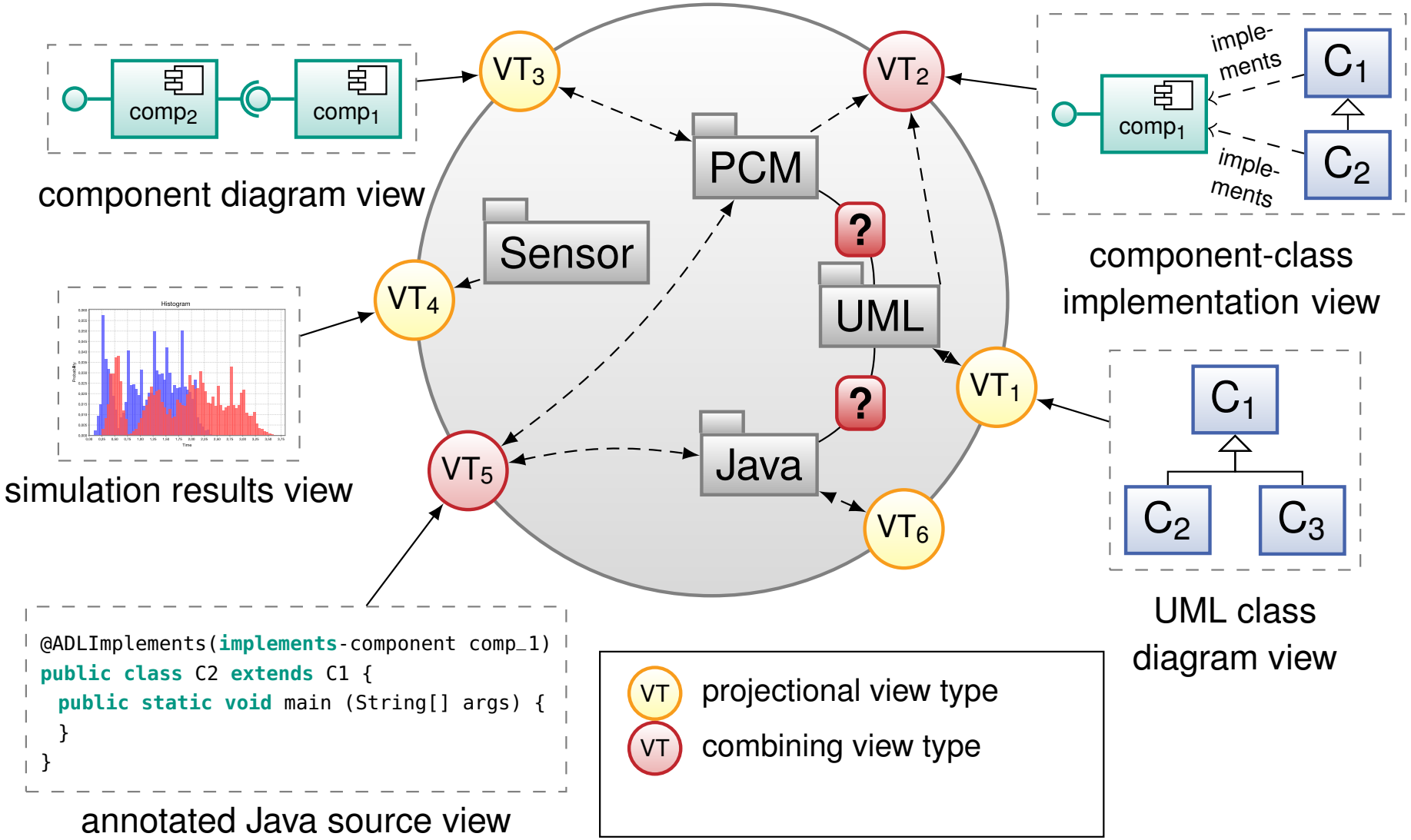
Database Research

- (Bancilhon et al. 1981)
  view-update problem
- (Batini et al. 1986)
  schema integration

Ontologies

- (Doan et al. 2005)
  semantic integration

Modular View-Centric Engineering



Research Goals & Questions

Research Goals

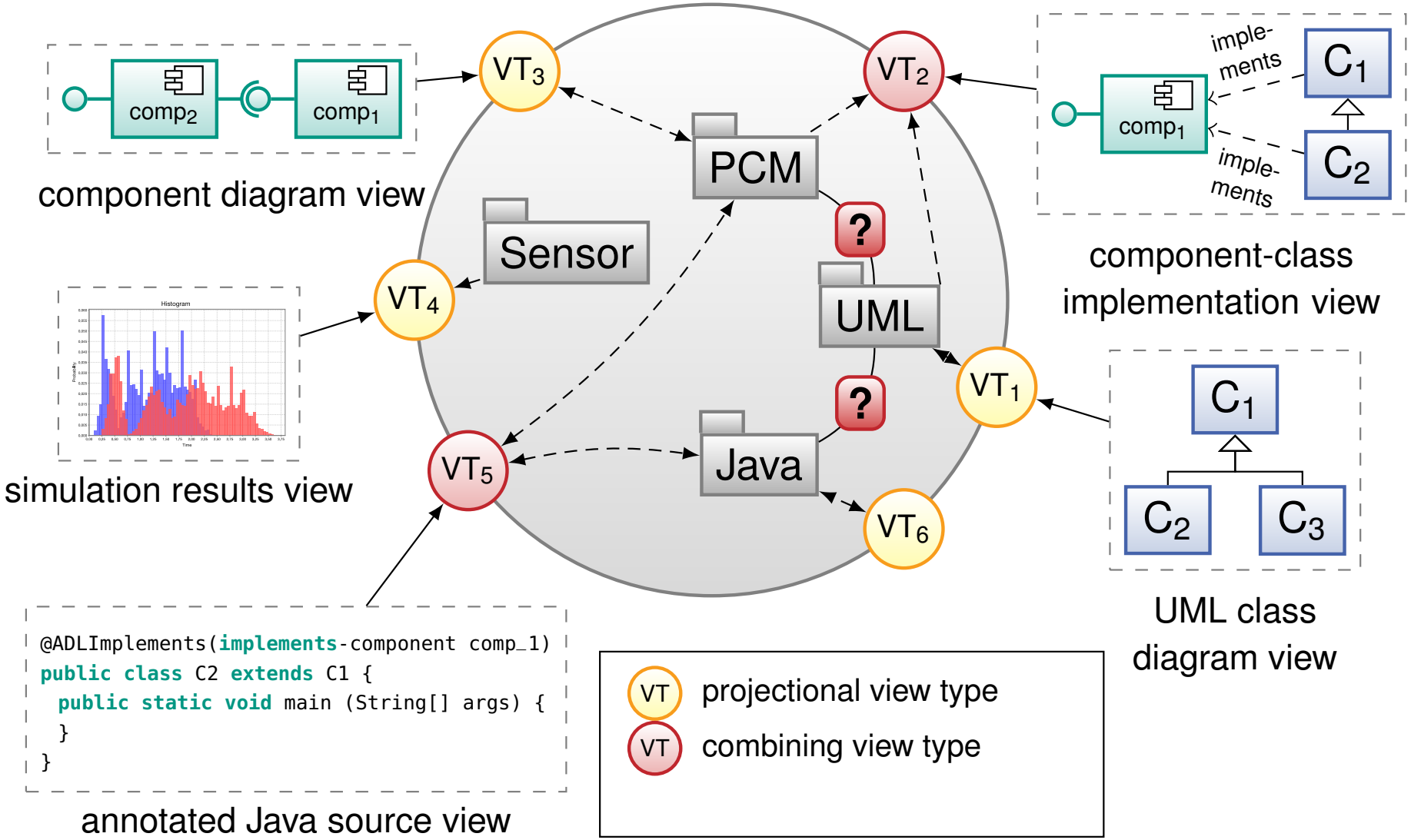
- ease consistent synchronization of heterogeneous models
- support views on these models using sync information and vice-versa

Research Questions

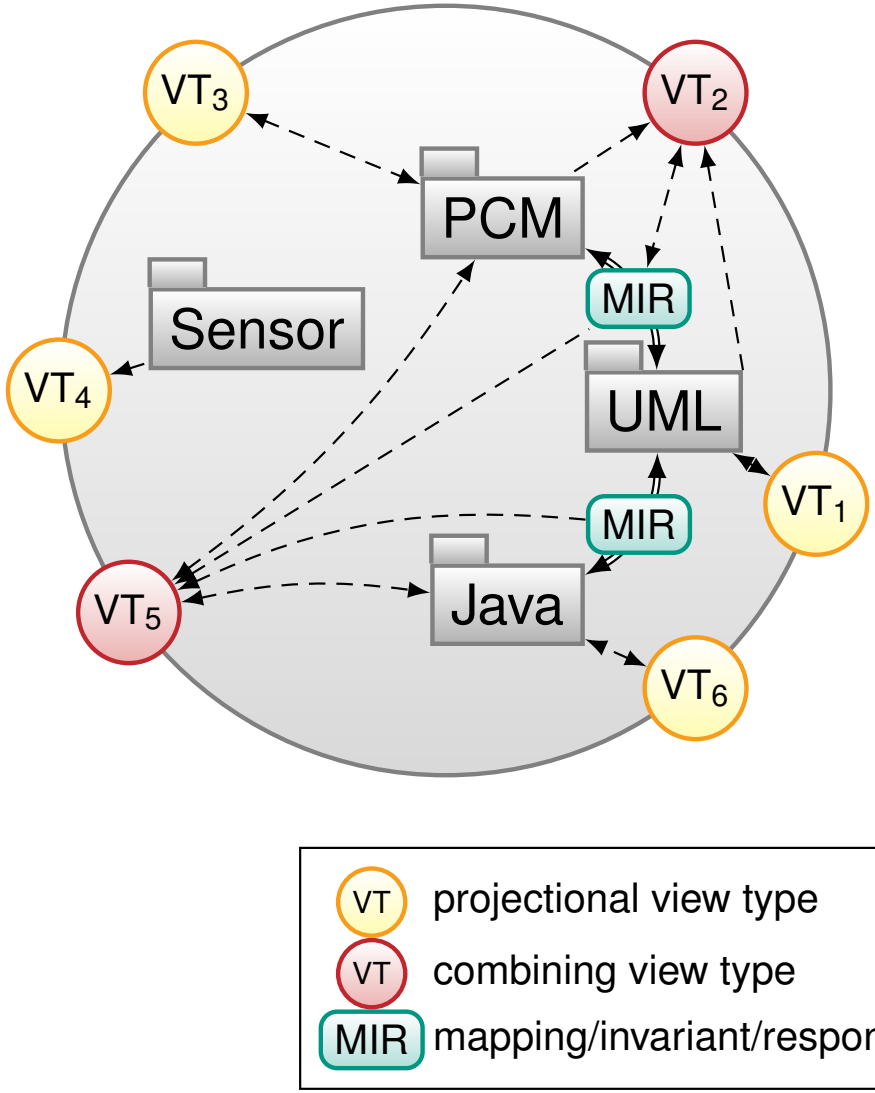
How to *synchronize* heterogeneous models *automatically*?

- Can we *specify* synchronization with a mostly declarative DSL?
- Can we *generate* transform. from mappings and embed custom code?
- Can we *sync* models by triggering these transform. on atomic changes?

Modular View-Centric Engineering



Modular View-Centric Engineering



A DSL for Abstract Consistency

- declarative mappings
- normative invariants
- imperative response actions

```
import "http://sdq.ipd.uka.de/PCM/Repository/5.0" as repo
import "http://www.eclipse.org/uml2/2.1.0/UML" as umlcd

// mappings
map repo::BasicComponent to umlcd::Package {
  when nestingPackage.name = 'Repository'
  with entityName:String to name:String
}

map c:umlcd::Class to bc:repo::BasicComponent {
  when name.endsWith('Comp')
    && visibility = public
    && isAbstract = true
```

A DSL for Abstract Consistency

- declarative mappings
- normative invariants
- imperative response actions

```
// invariants
context repo:Repository
inv uniqueComponentNames(c:repo::BasicComponent, d:repo::BasicComponent):
 self.components->forAll(c,d | c <> d implies c.entityName <> d.entityName)
```

A DSL for Abstract Consistency

- declarative mappings
- normative invariants
- imperative response actions

```
// global variable
var componentNameCount : Map<String,Integer>

// responses
on creation of class:umlcd::Class
restore inv uniqueComponentNames(c:repo::BasicComponent, d:repo::BasicComponent)
by {
   var occurrences = componentNameCount.get(c.entityName)
   occurrences = (occurrences == null) ? 2 : occurrences++
   componentNameCount.put(c.entityName, occurrences)
   class.name = c.entityName + occurrences + 'Comp'
}
```

Evaluation Requirements

What should be evaluated?

- Necessity (Arguments)
  - helps to convince others, optional

What should be validated?

- Functionality (Level I)
  - for a specific case, absolutely required
- Applicability (Level II)
  - for a specific task in different realistic cases, required
- Benefit (Level III)
  - in end-to-end scenarios with realistic users, very difficult

(cf. Böhme et al. 2005)

Evaluation Plans

Necessity: Industrial Poll and Interviews

- collect necessity cases and ensure appropriateness:
  - determine challenges with heterogeneous models, views and sync

Applicability: Case Studies

- assess applicability, generalisability and reusability:
  - PCM architecture, UML Class Diagrams, Java code
  - SysML, MATLAB/Simulink, EAST-ADL (AUTOSAR)
  - Common Workshop Case Study

Benefit: Quasi-Experiment

- try to assess benefits during system development (not end-to-end):
  - small android application
  - either 20 undergrads or 10 graduate students (and some engineers)

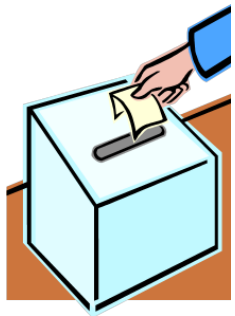
Industrial Poll and Interviews

Scientific Challenges

- choose companies and departments carefully
- adapt to different engineering domains
- uncover approaches that avoid multi-view synchronization

Operational Challenges

- how to reach employees and how to motivate participation?
- anonymous poll or trace-back for interviews?
- telephone interviews instead of an online poll?



Quasi-Experiment

Scientific Challenges

- baseline to compare with?
  - manual synchronization without a tool?
  - general purpose language and manual triggering?
- goal-oriented metrics for effectivity and consistency?

Operational Challenges

- small number of subjects
- restricted randomization

No End-to-End Valdation

- non-recurring effort for the methodologist excluded
- no long-term maintenance and operation

Limitation & Future Work

Limitation

- views have to report sequences of atomic changes

Excluded Work

- concurrent modifications
- versioning
- …

Future Work

- refactoring support (operation-based mappings)
- view-consistency cross-fertilization
- *application to cyber-physical systems*

Conclusions

Problem

- Keep information consistent across heterogeneous models

Idea

- using transformations that are generated from declarative mappings

Benefit

- in order to reduce accidental complexity and efforts

Action

- with a domain-specific synchronization language.

Thank you!

References I

Atkinson, Colin, Dietmar Stoll, and Philipp Bostan. “Orthographic Software Modeling: A Practical Approach to View-Based Development”. In: *Evaluation of Novel Approaches to Software Engineering*. Vol. 69. 2010, pp. 206–219.

Bancilhon, F. and N. Spyratos. “Update semantics of relational views”. In: *ACM Trans. Database Syst.* 6.4 (1981), pp. 557–575.

Batini, C., M. Lenzerini, and S. B. Navathe. “A Comparative Analysis of Methodologies for Database Schema Integration”. In: *ACM Comput. Surv.* 18.4 (1986), pp. 323–364.

Böhme, Rainer and Ralf Reussner. “Validation of Predictions with Measurements”. In: *Dependability Metrics*. Vol. 4909. 2005, pp. 14–18.

Cicchetti, Antonio, Federico Ciccozzi, and Thomas Leveque. “A hybrid approach for multi-view modeling”. In: *Electronic Communications of the EASST* 50 (2011).

— .“Supporting Incremental Synchronization in Hybrid Multi-view Modelling”. In: *Models in Software Engineering*. Vol. 7167. 2012, pp. 89–103.

Codd, E. F. *The relational model for database management: version 2*. Addison-Wesley Longman Publishing Co., Inc., 1990.

Di Ruscio, Davide et al. “Model-Driven Techniques to Enhance Architectural Languages Interoperability”. In: *Fundamental Approaches to Software Engineering*. Vol. 7212. 2012, pp. 26–42.

References II

Diskin, Zinovy. “Algebraic Models for Bidirectional Model Synchronization”. In: *Model Driven Engineering Languages and Systems*. Vol. 5301. 2008, pp. 21–36.

— .“Model Synchronization: Mappings, Tiles, and Categories”. In: *Generative and Transformational Techniques in Software Engineering III*. Vol. 6491. 2011, pp. 92–165.

Diskin, Zinovy, Yingfei Xiong, and Krzysztof Czarnecki. “Specifying overlaps of heterogeneous models for global consistency checking”. In: *Proceedings of the First International Workshop on Model-Driven Interoperability*. 2010, pp. 42–51.

Doan, AnHai and Alon Y. Halevy. “Semantic-integration Research in the Database Community”. In: *AI Mag.* 26.1 (2005), pp. 83–94.

Eramo, Romina, Ivano Malavolta, et al. “A model-driven approach to automate the propagation of changes among Architecture Description Languages”. In: *Software and Systems Modeling* 11 (1 2012), pp. 29–53.

Eramo, Romina, Alfonso Pierantonio, et al. “Change Management in Multi-Viewpoint System Using ASP”. In: *Enterprise Distributed Object Computing Conference Workshops, 2008 12th*. 2008, pp. 433–440.

Foster, J. Nathan et al. “Combinators for bi-directional tree transformations: a linguistic approach to the view update problem”. In: *SIGPLAN Not.* 40.1 (2005), pp. 233–246.

References III

Giese, Holger, Stephan Hildebrandt, and Stefan Neumann. “Model Synchronization at Work: Keeping SysML and AUTOSAR Models Consistent”. In: *Graph Transformations and Model-Driven Engineering*. Vol. 5765. 2010, pp. 555–579.

Giese, Holger and Robert Wagner. “From model transformation to incremental bidirectional model synchronization”. In: *Software and Systems Modeling* 8 (1 2009), pp. 21–43.

— .“Incremental Model Synchronization with Triple Graph Grammars”. In: *Model Driven Engineering Languages and Systems*. Vol. 4199. 2006, pp. 543–557.

Hakimpour, Farshad and Andreas Geppert. “Resolving Semantic Heterogeneity in Schema Integration”. In: *Proceedings of the International Conference on Formal Ontology in Information Systems - Volume 2001*. 2001, pp. 297–308.

Hein, Christian, Tom Ritter, and Michael Wagner. “Model-Driven Tool Integration with ModelBus”. In: *Workshop Future Trends of Model-Driven Development*. 2009.

Hermann, Frank et al. “Concurrent Model Synchronization with Conflict Resolution Based on Triple Graph Grammars”. In: *Fundamental Approaches to Software Engineering*. Vol. 7212. 2012, pp. 178–193.

References IV

Hettel, Thomas, Michael Lawley, and Kerry Raymond. “Model Synchronisation: Definitions for Round-Trip Engineering”. In: *Theory and Practice of Model Transformations*. Vol. 5063. 2008, pp. 31–45.

Kienzle, Jörg, Wisam Al Abed, and Jacques Klein. “Aspect-oriented multi-view modeling”. In: *Proceedings of the 8th ACM international conference on Aspect-oriented software development*. 2009, pp. 87–98.

Lechtenbörger, Jens. “The impact of the constant complement approach towards view updating”. In: *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 2003, pp. 49–55.

Malavolta, Ivano et al. “Providing Architectural Languages and Tools Interoperability through Model Transformation Technologies”. In: *IEEE Transactions of Software Engineering* 36.1 (2010), pp. 119–140.

Romero, José Raúl, Juan Ignacio Jaén, and Antonio Vallecillo. “Realizing Correspondences in Multi-viewpoint Specifications”. In: *Enterprise Distributed Object Computing Conference, 2009. EDOC ’09. IEEE International*. 2009, pp. 163–172.

Xiong, Yingfei et al. “Towards automatic model synchronization from model transformations”. In: *Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering*. 2007, pp. 164–173.