How and Why is Change Modeled? – A Scoping Literature Review

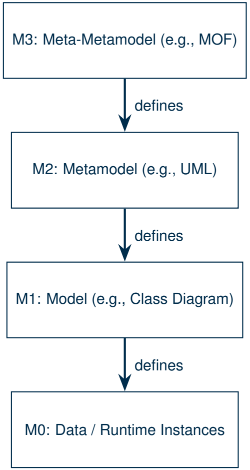
Thomas Weber, Johan Cederbladh, Sebastian Weber, Arne Lange, Antonio Cicchetti, Ralf Reussner
KASTEL-DSiS
07/10/2025

Introduction

Change

- Change is central to development in computer science [1]
- ‘*The only constant is change.*’ — Heraclitus[a]

[a] Attributed to Heraclitus, based on fragments of his writings. See: https://www.reference.com/world-view/said-only-thing-constant-change-d50c0532e714e12b


M3: Meta-Metamodel (e.g., MOF)
defines
M2: Metamodel (e.g., UML)
defines
M1: Model (e.g., Class Diagram)
defines
M0: Data / Runtime Instances

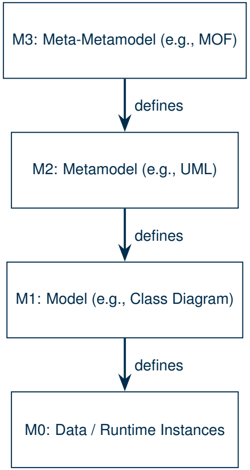
Introduction

Change

- Change is central to development in computer science [1]
- ‘*The only constant is change.*’ — Heraclitus[a]

Focus

- Concrete artifacts: models and metamodels
- Shared conceptual core of change across domains


M3: Meta-Metamodel (e.g., MOF)
defines
M2: Metamodel (e.g., UML)
defines
M1: Model (e.g., Class Diagram)
defines
M0: Data / Runtime Instances

[a] Attributed to Heraclitus, based on fragments of his writings. See: https://www.reference.com/world-view/said-only-thing-constant-change-d50c0532e714e12b

Introduction

Change

- Change is central to development in computer science [1]
- ‘*The only constant is change.*’ — Heraclitus[a]

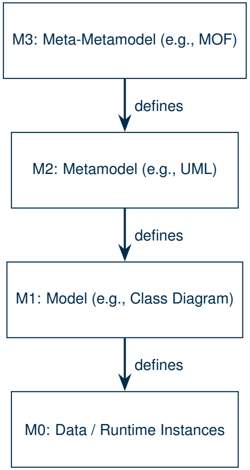
Focus

- Concrete artifacts: models and metamodels
- Shared conceptual core of change across domains

Scope

Model-Based Engineering

Excludes other dimensions, e.g., organizational change

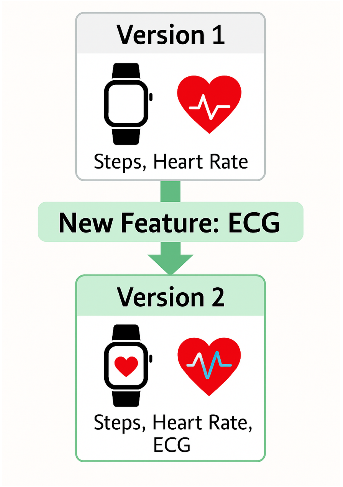

M3: Meta-Metamodel (e.g., MOF)
defines
M2: Metamodel (e.g., UML)
defines
M1: Model (e.g., Class Diagram)
defines
M0: Data / Runtime Instances

[a] Attributed to Heraclitus, based on fragments of his writings. See: https://www.reference.com/world-view/said-only-thing-constant-change-d50c0532e714e12b

Motivation

Why Model Change?

Isn’t dealing with change enough?

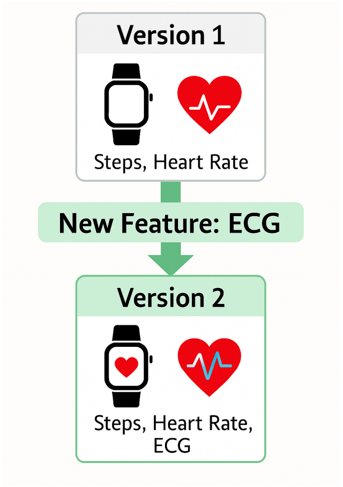


Motivation

Why Model Change?

Isn’t dealing with change enough?

We need to embrace the change!



Motivation

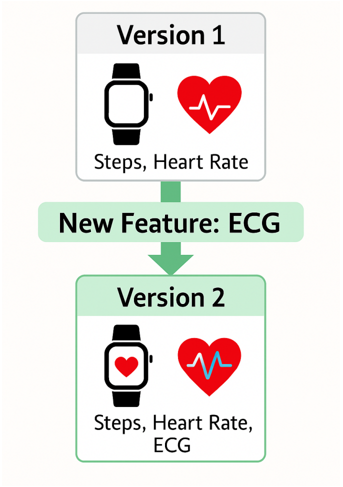
Why Model Change?

Isn't dealing with change enough?

We need to embrace the change!

Rationale

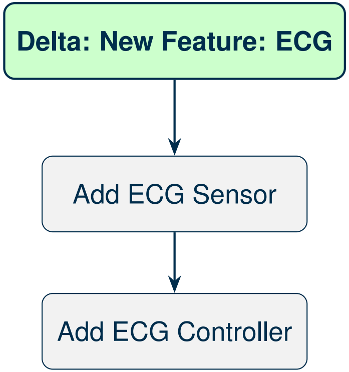
- Models evolve over time
- Changes must be tracked and understood
- States alone may lose information [1]
- Example: rename vs delete + re-add



Delta Concept

What is a Delta?

- Delta, i.e., description of change
- Artifact structure affects how change is defined and modeled



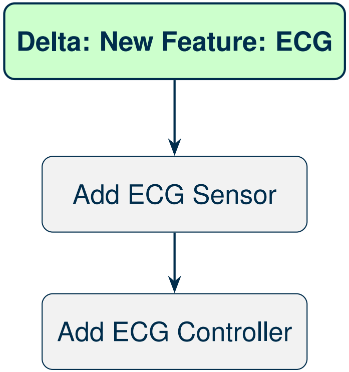
Delta Concept

What is a Delta?

- Delta, i.e., description of change
- Artifact structure affects how change is defined and modeled

Semantics of a Delta

- Artifacts range from semantically meaningful (metamodels) to syntactic (e.g., Git blobs)
- Metamodel-level changes can be interpreted as domain-specific or domain-agnostic



Scoping Literature Review

What is a Scoping Review?

- Maps key concepts and definitions in broad field
- Identifies characteristics or factors related to a concept
- More structured than traditional reviews, less rigid than systematic ones

Scoping Literature Review

What is a Scoping Review?

- Maps key concepts and definitions in broad field
- Identifies characteristics or factors related to a concept
- More structured than traditional reviews, less rigid than systematic ones

Why Use a Scoping Review?

- Suitable for vast, interconnected literature
- Helps clarify the concept of change
- Enables focused inclusion/exclusion criteria

Scoping Literature Review

What is a Scoping Review?

- Maps key concepts and definitions in broad field
- Identifies characteristics or factors related to a concept
- More structured than traditional reviews, less rigid than systematic ones

Why Use a Scoping Review?

- Suitable for vast, interconnected literature
- Helps clarify the concept of change
- Enables focused inclusion/exclusion criteria

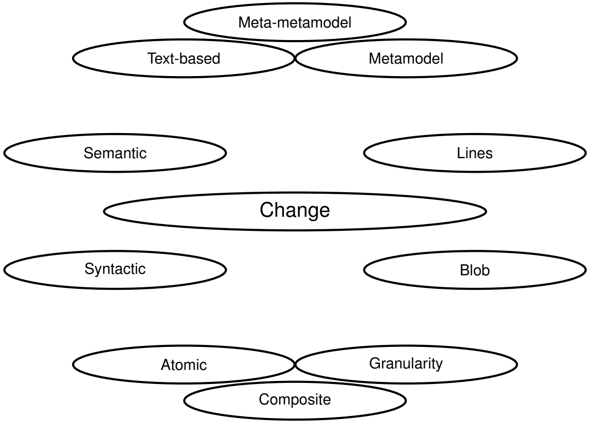
This Review

- Based on Munn et al.'s guidelines. [2]
- Inclusion: MDE context, concept & definition of change, model/metamodel focus
- Exclusion: Non-English, not peer-reviewed, no full text
- Sources: Scopus, IEEE, ACM, Google Scholar
- Final set: 41 core papers selected

RQ1: How is Change Modeled?

Modeling Change

- Change defined by consequence, i.e., difference before and after
- Requires structured artifacts to assess and model change
- Changes are modeled as modifications to models and metamodels



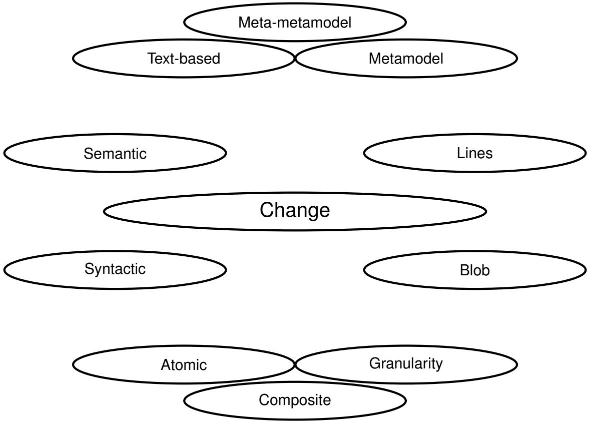
RQ1: How is Change Modeled?

Modeling Change

- Change defined by consequence, i.e., difference before and after
- Requires structured artifacts to assess and model change
- Changes are modeled as modifications to models and metamodels

Structure

- Structure enables modeling at different abstraction levels
- Granularity ranges from blackbox blobs to, e.g., line-based divisions



RQ2: Purpose of Change Metamodel

Why Purpose Matters

- Purpose defines goals and requirements of change metamodels
- Different stakeholders — different motivations for change
- Use cases influence design priorities, e.g., completeness or performance

Describe Changes

Analyze Evolution

Enable Automation

Support Consistency

RQ2: Purpose of Change Metamodel

Why Purpose Matters

- Purpose defines goals and requirements of change metamodels
- Different stakeholders — different motivations for change
- Use cases influence design priorities, e.g., completeness or performance

Purpose and Trade-offs

- Supports versioning, synchronization, transformation
- Different purposes imply different requirements
- Traceability vs. performance trade-offs
- Stakeholder diversity affects modeling needs

Describe Changes

Analyze Evolution

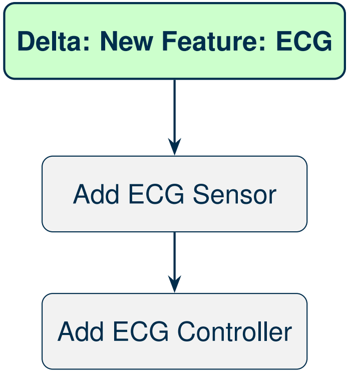
Enable Automation

Support Consistency

RQ3: Atomic and Composite Changes

Modeling Change Types

- Atomic: complete coverage of change types
- Enables rollback, traceability, automation



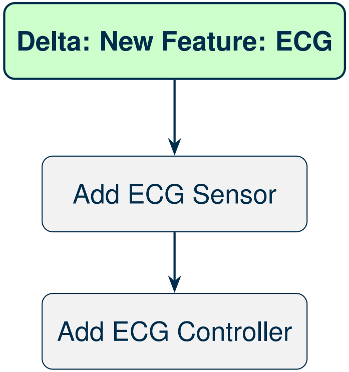
RQ3: Atomic and Composite Changes

Modeling Change Types

- Atomic: complete coverage of change types
- Enables rollback, traceability, automation

Composite Changes

- Composite changes, i.e., finite atomic groupings, but in-complete in regard to possible changes
- Composite: user intent [3], productivity, understanding

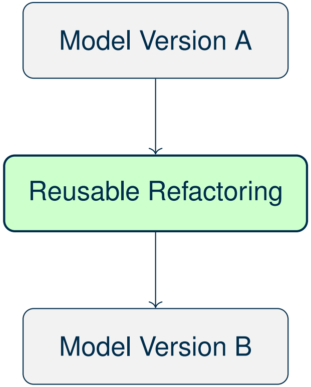


Example: EDelta: Reusable Metamodel Refactorings [4]

EDelta Overview

- Supports atomic and composite metamodel changes
- Enables reusable refactoring catalog

- Built on Xtext and EMF
- Purpose: enable safe and automated evolution of metamodels

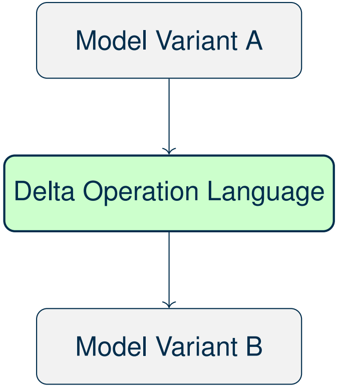


Example: Delta Operation Language [5]

Delta Operations

- Software Product Line Engineering
- Metamodel-independent approach, deriving metamodel-specific change metamodels

- Models differences via delta operations
- Supports complete atomic change modeling
- Purpose: Enables product derivation with deltas

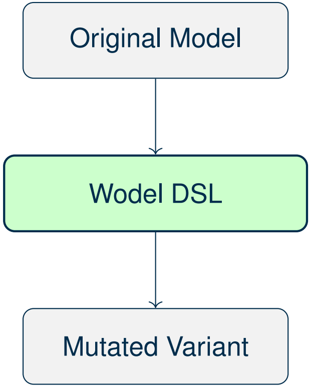


Example: Wodel – DSL for Model Mutation [6]

Mutation DSL

- Domain-independent DSL for model mutation
- Supports atomic and composite changes

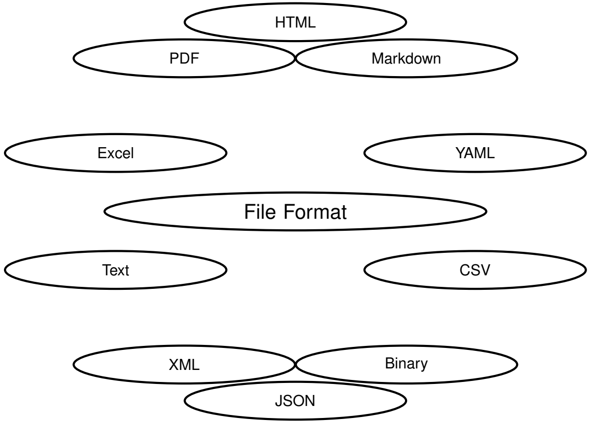
- Enables programmatic generation of model variants
- Purpose: testing, variant creation, mutation analysis



Artifact Structure

Structure

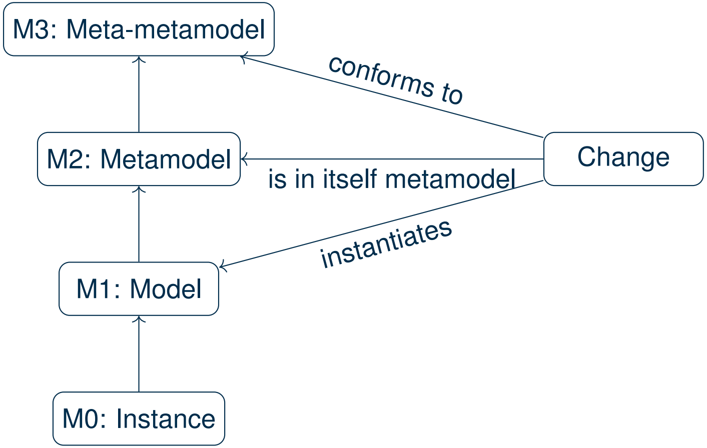
- Structure can be semantic or syntactic
- Metamodels: domain-specific, strict structure
- Text-based artifacts: less struc-tured
- Structure affects change model-ing capability



Meta-Metamodel Level

Cross-Domain Flexibility

- Changes at meta-metamodel level are domain-agnostic
- Metamodel defines domain-specific semantics
- Enables reuse across domains
- Supports flexible change modeling



RQ1: How is Change Modeled?

Structural Scope

- Artifacts range from blackbox blobs to structured models [7]
- More structure enables finer change modeling, but reduces generality [8]

RQ1: How is Change Modeled?

Structural Scope

- Artifacts range from blackbox blobs to structured models [7]
- More structure enables finer change modeling, but reduces generality [8]

Metamodeling Approaches

- Most approaches build on EMF [9], with exceptions [10, 11, 12, 13, 14]
- Change metamodels defined at metamodel or meta-metamodel level [15, 16, 17]
- Parametrized metamodels balance generality and specificity [16]

RQ1: How is Change Modeled?

Structural Scope

- Artifacts range from blackbox blobs to structured models [7]
- More structure enables finer change modeling, but reduces generality [8]

Metamodeling Approaches

- Most approaches build on EMF [9], with exceptions [10, 11, 12, 13, 14]
- Change metamodels defined at metamodel or meta-metamodel level [15, 16, 17]
- Parametrized metamodels balance generality and specificity [16]

Advanced Concepts

- Semantic vs. syntactic dimensions influence applicability [18]
- Terminology varies: change, delta, operation, event, mutation [19, 6]

RQ2: What is the Purpose of the Change Metamodel?

Describing and Reusing Change

- Central purpose: describe modifications between model versions [20, 21, 22, 5]
- Reusable change formats enable reuse across models [23]

RQ2: What is the Purpose of the Change Metamodel?

Describing and Reusing Change

- Central purpose: describe modifications between model versions [20, 21, 22, 5]
- Reusable change formats enable reuse across models [23]

Versioning and Collaboration

- Changes can represent entire model states for versioning [24, 25, 26, 1]
- Support for collaborative modeling, conflict management, and live modeling [27, 28, 11, 12, 13, 14]

RQ2: What is the Purpose of the Change Metamodel?

Describing and Reusing Change

- Central purpose: describe modifications between model versions [20, 21, 22, 5]
- Reusable change formats enable reuse across models [23]

Versioning and Collaboration

- Changes can represent entire model states for versioning [24, 25, 26, 1]
- Support for collaborative modeling, conflict management, and live modeling [27, 28, 11, 12, 13, 14]

Consistency and Evolution

- Enables consistency preservation and model repair [29, 30, 16, 31]
- Supports co-evolution, variant derivation, and semantic reasoning [32, 33, 34, 17, 18]

RQ3: To What Extent Can Atomic and Composite Changes Be Modeled?

Atomic Changes

- Most approaches define complete atomic change metamodels [6]
- Atomic changes serve as the foundation for model variant generation and fine-grained evolution

RQ3: To What Extent Can Atomic and Composite Changes Be Modeled?

Atomic Changes

- Most approaches define complete atomic change metamodels [6]
- Atomic changes serve as the foundation for model variant generation and fine-grained evolution

Composite Changes

- Composite changes extend atomic models via refactoring catalogs or user-defined constructs [23, 16]
- Used to support users with grouped changes and traceability [3]

RQ3: To What Extent Can Atomic and Composite Changes Be Modeled?

Atomic Changes

- Most approaches define complete atomic change metamodels [6]
- Atomic changes serve as the foundation for model variant generation and fine-grained evolution

Composite Changes

- Composite changes extend atomic models via refactoring catalogs or user-defined constructs [23, 16]
- Used to support users with grouped changes and traceability [3]

Granularity and Completeness

- Granularity affects understandability and correctness of change modeling
- Incomplete metamodels or heuristic-based grouping may lead to erroneous co-evolution [16]

Discussion: Designing and Applying Change Metamodels

Metamodel Selection

- Choose between existing, derived, or custom metamodels based on purpose and domain [35]
- Domain-specificity and expressive power guide the selection process

Discussion: Designing and Applying Change Metamodels

Metamodel Selection

- Choose between existing, derived, or custom metamodels based on purpose and domain [35]
- Domain-specificity and expressive power guide the selection process

Granularity and Intent

- Higher abstraction captures developer intent but risks incompleteness
- Define granularity and change acquisition method (recorded vs. derived)

Discussion: Designing and Applying Change Metamodels

Metamodel Selection

- Choose between existing, derived, or custom metamodels based on purpose and domain [35]
- Domain-specificity and expressive power guide the selection process

Granularity and Intent

- Higher abstraction captures developer intent but risks incompleteness
- Define granularity and change acquisition method (recorded vs. derived)

Combining Metamodels

- Use multiple metamodels for different tasks (e.g., consistency vs. user display)
- Annotate changes with intent or standards [3]

Conclusion

Key Insights

- Presented scoping literature review change metamodels
- Addressed three RQ modeling, purpose, and granularity
- Identified diverse approaches in change modeling
- Concept reuse can improve efficiency

Conclusion

Key Insights

- Presented scoping literature review change metamodels
- Addressed three RQ modeling, purpose, and granularity
- Identified diverse approaches in change modeling
- Concept reuse can improve efficiency

Future Work

- Build a systematic literature review based on this study
- Explore combining metamodels for complex use cases
- Align abstraction levels with user needs and consistency mechanisms

References I

J. W. Wittler, T. Saglam, and T. Kühn, "Evaluating model differencing for the consistency preservation of state-based views." *J. Object Technol.*, vol. 22, no. 2, pp. 1–14, 2023.

Z. Munn, M. D. Peters, C. Stern, C. Tufanaru, A. McArthur, and E. Aromataris, "Systematic review or scoping review? guidance for authors when choosing between a systematic or scoping review approach," *BMC medical research methodology*, vol. 18, pp. 1–7, 2018.

J. Cederbladh, E. Kamburjan, D. A. Manrique-Negrin, R. Mittal, and T. Weber, "Traceability support for engineering reviews of horizontal model evolution," *Systems Engineering*, vol. n/a, no. n/a, p. e70001, 2025. [Online]. Available: https://incose.onlinelibrary.wiley.com/doi/abs/10.1002/sys.70001

L. Bettini, D. Di Ruscio, L. Iovino, and A. Pierantonio, "Edelta 2.0: supporting live metamodel evolutions," in *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, 2020, pp. 1–10.

References II

- D. Kuryazov and A. Winter, “Representing model differences by delta operations,” in *2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations*. IEEE, 2014, pp. 211–220.
- P. Gómez-Abajo, E. Guerra, and J. De Lara, “Wodel: a domain-specific language for model mutation,” in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, 2016, pp. 1968–1973.
- N. N. Zolkifli, A. Ngah, and A. Deraman, “Version control system: A review,” *Procedia Computer Science*, vol. 135, pp. 408–415, 2018.
- T. T. Nguyen, H. A. Nguyen, N. H. Pham, and T. N. Nguyen, “Operation-based, fine-grained version control model for tree-based representation,” in *International Conference on Fundamental Approaches to Software Engineering*. Springer, 2010, pp. 74–90.
- D. Steinberg, F. Budinsky, E. Merks, and M. Paternostro, *EMF: eclipse modeling framework*. Pearson Education, 2008.

References III

- J. Exelmans, C. Teodorov, R. Heinrich, A. Egyed, and H. Vangheluwe, “Collaborative live modelling by language-agnostic versioning,” in *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE, 2023, pp. 364–374.
- J. Exelmans, J. Pietron, A. Raschke, H. Vangheluwe, and M. Tichy, “A new versioning approach for collaboration in blended modeling,” *Journal of Computer Languages*, vol. 76, p. 101221, 2023.
- J. Exelmans, C. Teodorov, and H. Vangheluwe, “Operation-based versioning as a foundation for live executable models,” *Software and Systems Modeling*, pp. 1–19, 2024.
- I. David, E. Syriani, and C. Masson, “Extensible conflict-free replicated datatypes for real-time collaborative software engineering,” in *2022 17th Conference on Computer Science and Intelligence Systems (FedCSIS)*. IEEE, 2022, pp. 849–853.

References IV

- I. David and E. Syriani, “Real-time collaborative multi-level modeling by conflict-free replicated data types,” *Software and Systems Modeling*, vol. 22, no. 4, pp. 1131–1150, 2023.
- E. Burger and B. Gruschko, “A change metamodel for the evolution of mof-based metamodels,” in *Modellierung 2010*. Gesellschaft für Informatik eV, 2010, pp. 285–300.
- H. Klare, M. E. Kramer, M. Langhammer, D. Werle, E. Burger, and R. Reussner, “Enabling consistency in view-based system development–the vitruvius approach,” *Journal of Systems and Software*, vol. 171, p. 110815, 2021.
- C. Seidl, I. Schaefer, and U. Aßmann, “Deltaecore-a model-based delta language generation framework,” in *Modellierung 2014*. Gesellschaft für Informatik eV, 2014, pp. 81–96.
- A. Cicchetti and F. Ciccozzi, “Towards a novel model versioning approach based on the separation between linguistic and ontological aspects.” in *ME@ MoDELS*. Citeseer, 2013, pp. 60–69.

References V

- G. Bergmann, I. Dávid, Á. Hegedüs, Á. Horváth, I. Ráth, Z. Ujhelyi, and D. Varró, “Viatra 3: A reactive model transformation platform,” in *Theory and Practice of Model Transformations: 8th International Conference, ICMT 2015, Held as Part of STAF 2015, L'Aquila, Italy, July 20-21, 2015. Proceedings 8*. Springer, 2015, pp. 101–110.
- J. E. Rivera and A. Vallecillo, “Representing and operating with model differences,” in *International Conference on Objects, Components, Models and Patterns*. Springer, 2008, pp. 141–160.
- C. Brun and A. Pierantonio, “Model differences in the eclipse modeling framework,” *UPGRADE, The European Journal for the Informatics Professional*, vol. 9, no. 2, pp. 29–34, 2008.
- A. Cicchetti, D. Di Ruscio, and A. Pierantonio, “Model patches in model-driven engineering,” in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2009, pp. 190–204.

References VI

- L. Bettini, D. Di Ruscio, L. Iovino, A. Pierantonio *et al.*, “Edelta: An approach for defining and applying reusable metamodel refactorings.” in *MODELS (satellite events)*, 2017, pp. 71–80.
- X. Blanc, I. Mounier, A. Mougenot, and T. Mens, “Detecting model inconsistency through operation-based model construction,” in *Proceedings of the 30th international conference on Software engineering*, 2008, pp. 511–520.
- A. Yohannis, D. Kolovos, and F. Polack, “Turning models inside out,” in *CEUR Workshop Proceedings 1403*. York, 2017, pp. 430–434.
- M. Herrmannsdoerfer and M. Koegel, “Towards a generic operation recorder for model evolution,” in *Proceedings of the 1st International Workshop on Model Comparison in Practice*, 2010, pp. 76–81.
- M. Koegel, M. Herrmannsdoerfer, Y. Li, J. Helming, and J. David, “Comparing state-and operation-based change tracking on models,” in *2010 14th ieee international enterprise distributed object computing conference*. IEEE, 2010, pp. 163–172.

References VII

- G. Taentzer, C. Ermel, P. Langer, and M. Wimmer, “A fundamental approach to model versioning based on graph modifications: from theory to implementation,” *Software & Systems Modeling*, vol. 13, no. 1, pp. 239–272, 2014.
- G. Taentzer, M. Ohrndorf, Y. Lamo, and A. Rutle, “Change-preserving model repair,” in *International conference on fundamental approaches to software engineering*. Springer, 2017, pp. 283–299.
- L. Marchezan, R. Kretschmer, W. K. Assunção, A. Reder, and A. Egyed, “Generating repairs for inconsistent models,” *Software and Systems Modeling*, vol. 22, no. 1, pp. 297–329, 2023.
- X. Blanc, A. Mougenot, I. Mounier, and T. Mens, “Incremental detection of model inconsistencies based on model operations,” in *International Conference on Advanced Information Systems Engineering*. Springer, 2009, pp. 32–46.

References VIII

- M. Herrmannsdoerfer, “Cope–a workbench for the coupled evolution of metamodels and models,” in *International conference on software language engineering*. Springer, 2010, pp. 286–295.
- A. Cicchetti, D. Di Ruscio, R. Eramo, and A. Pierantonio, “Automating co-evolution in model-driven engineering,” in *2008 12th International IEEE enterprise distributed object computing conference*. IEEE, 2008, pp. 222–231.
- I. Schaefer, L. Bettini, V. Bono, F. Damiani, and N. Tanzarella, “Delta-oriented programming of software product lines,” in *Software Product Lines: Going beyond: 14th International Conference, SPLC 2010, Jeju Island, South Korea, September 13-17, 2010. Proceedings 14*. Springer, 2010, pp. 77–91.
- H. Vangheluwe, J. De Lara, and P. J. Mosterman, “An introduction to multi-paradigm modelling and simulation,” in *Proceedings of the AIS’2002 conference (AI, Simulation and Planning in High Autonomy Systems), Lisboa, Portugal*, vol. 21, no. 1, 2002.