Towards Rethinking Consistency Management in Collaborative Modeling Through Organizational Dimensions

Luciano Marchezan
lucianomarchp@gmail.com
Université de Montréal
Montreal, QC, Canada

Hyacinth Ali
hyacinth.ali@umontreal.ca
Université de Montréal
Montreal, QC, Canada
University of Alberta
Edmonton, AB, Canada

Eugene Syriani
syriani@iro.umontreal.ca
Université de Montréal
Montreal, QC, Canada

Abstract

Collaborative modeling requires teams to maintain consistency across artifacts while coordinating work, communicating, and planning decisions. Existing consistency management approaches provide mechanisms for detecting, preventing, and resolving inconsistencies, but primarily focus on technical aspects and provide limited alignment to the organizational perspective. In practice, factors such as decision authority, collaboration structure, ownership, and tolerance for temporary inconsistencies influence how inconsistencies arise and should be handled. This requires consistency management to consider these organizational dimensions. In this paper, we propose a scenario-aware perspective for collaborative modeling based on organizational dimensions. For that, we: (i) introduce organizational dimensions for collaborative consistency management; (ii) define different collaborative scenarios based on these dimensions; and (iii) map these scenarios to different consistency management strategies. We also discuss research challenges and opportunities for scenario-aware consistency management.

CCS Concepts

• **Software and its engineering** → **Consistency**; *Abstraction, modeling and modularity*.

Keywords

Consistency Management; Organizational Dimensions; Collaborative Modeling

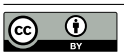
ACM Reference Format:
Luciano Marchezan, Hyacinth Ali, and Eugene Syriani. 2026. Towards Rethinking Consistency Management in Collaborative Modeling Through Organizational Dimensions. In *ACM/IEEE 29th International Conference on Model Driven Engineering Languages and Systems (MODELS Companion 2026), October 04–09, 2026, Málaga, Spain.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3837062.3838722

1 Introduction

Keeping artifacts consistent is a challenge in collaborative modeling, where stakeholders, including developers, modelers, and domain experts, jointly create and evolve shared models and related artifacts[1] [7]. In such settings, inconsistencies naturally emerge due to distributed work, diverging assumptions, partial communication, and concurrent modifications across interconnected artifacts [28]. A wide range of consistency management approaches has been proposed, including rule-based checking, constraint evaluation, trace-based analysis, and model transformation techniques [15, 16]. Complementary approaches focus on repair through recommendations, repair trees, and automated transformations [20, 23, 26], while action-driven consistency prevents inconsistencies by design through API-level enforcement [1]. Although effective, these approaches are largely designed from a technical perspective and differ significantly in their assumptions about artifacts, automation, and decision-making.

In collaborative modeling, however, consistency is also defined by organizational and social factors such as responsibilities, ownership, decision authority, and collaboration practices [14, 32]. Thus, models often serve as boundary objects that support communication and negotiation among stakeholders with different goals [17]. As a result, real-world collaborative scenarios,[2] can vary considerably because organizations adopt different collaboration practices and governance structures. For example, we can consider a cross-organizational modeling project in which domain experts from different institutions collaboratively design a business process [27]. Rather than using specialized modeling tools, participants contribute asynchronously by editing PowerPoint diagrams, lowering the technological barrier to participation. A designated mediator periodically consolidates these contributions into a shared process model and coordinates discussions whenever conflicting changes arise. In this setting, an inconsistency (e.g., two stakeholders proposing different versions of the same activity) is not necessarily treated as an error requiring immediate repair. Instead, it may intentionally remain unresolved while stakeholders negotiate alternatives or await additional information. Such collaborative practices have been shown to increase trust, ownership, and cross-organizational alignment, emphasizing that modeling serves purposes beyond technical correctness [12].

By contrast, in a software development project where multiple engineers concurrently edit an executable system model, the same inconsistency might need to be detected and repaired immediately to avoid build failures or invalid downstream artifacts [1]. These distinct cases illustrate that the appropriate consistency management strategy depends not only on the technical characteristics



MODELS Companion 2026, Málaga, Spain

ACM ISBN 979-8-4007-2903-4/2026/10
https://doi.org/10.1145/3837062.3838722

[1] We use "artifact" to refer to any element relevant to consistency management.
[2] A scenario is a combination of different organizational factors.

of the artifacts but also on organizational factors such as inconsistency tolerance (e.g., whether inconsistencies must be resolved immediately), repair authority (i.e., who is responsible for resolving them), consistency scope (i.e., which artifacts are relevant for maintaining consistency), coordination mechanisms, and stakeholder responsibilities [9].

Despite this variability, existing work provides limited support for systematically aligning consistency management strategies with organizational context. Most existing approaches implicitly assume a particular way of organizing collaboration, defining responsibilities, and resolving inconsistencies. Therefore, they provide limited guidance on selecting an appropriate strategy for different collaborative scenarios. To address this gap, we propose a scenario-aware perspective on consistency management that explicitly considers organizational dimensions. We argue that these dimensions define distinct collaboration scenarios that determine how inconsistencies should be handled. Specifically, the contributions of this paper are: (i) the identification of key organizational dimensions for collaborative modeling scenarios; (ii) the characterization of scenarios as configurations of these dimensions; and (iii) the mapping of these scenarios to existing consistency management strategies.

The remainder of this paper is organized as follows. Section 2 summarizes families of consistency management approaches. Section 3 introduces the organizational dimensions, and Section 4 presents the scenario mapping. Finally, Section 5 discusses research challenges and future directions.

2 Background and Motivation

Consistency management has been studied from several complementary perspectives, primarily focusing on technical mechanisms for preventing or resolving inconsistencies. While these approaches differ in their features, they are typically evaluated according to technical criteria such as correctness, performance, scalability, and automation [30]. A first family of approaches concentrates on *checking*. These approaches define rules or constraints over models and other artifacts and report inconsistencies when expected relationships are no longer satisfied [18, 28, 29]. Such techniques are particularly important for maintaining consistency across different abstraction levels and heterogeneous artifacts [10, 15, 16].

A second family focuses on *incrementality and scalability*. Incremental checking avoids re-evaluating all rules after each change and is therefore well suited to interactive and collaborative environments [4, 11]. While these approaches improve feedback timing, they provide limited guidance on how inconsistencies should be managed once detected. A third family addresses *repair* by generating recommendations, repair trees, or automated transformations to help resolve inconsistencies [2, 19, 20, 22, 23]. Although effective in many situations, repair mechanisms often assume particular decision-making processes regarding who evaluates and applies the proposed changes [21]. In practice, however, inconsistency resolution is rarely deterministic as multiple alternative repairs may be valid, and selecting among them may depend on contextual, expert-based criteria. This introduces a fundamental research challenge of managing uncertainty in consistency evolution, where the system cannot unambiguously determine a single "correct" resolution. This challenge is analogous to issues observed in model transformation research, where non-deterministic transformations can produce multiple valid outputs and require explicit strategies to control or interpret alternative outcomes [3].

A fourth family focuses on *collaborative and global consistency*. These approaches make consistency information visible across users, artifacts, and versions, allowing teams to identify and track inconsistencies in shared modeling environments [10, 31]. These include prevention-oriented approaches that aim to avoid inconsistencies in the first place [1, 16].

Collaborative and participatory modeling research shows that factors such as domain-expert responsibilities, ownership structures, decision authority, and collaboration practices influence how inconsistencies (and repairs) are managed [17, 32]. Evidence from collaborative and inter-organizational contexts further highlights that consistency depends not only on technical specifications but also on organizational arrangements governing collaboration [5]. For example, domain experts may collaborate with modeling specialists who then refine and revise artifacts, making transparency and ownership central concerns in evaluating modeling outcomes [12]. Moreover, some approaches seek to reduce the separation between domain experts and modeling specialists by allowing end users to participate directly in the creation of domain-specific modeling languages through example-based language definition [6]. In such settings, responsibilities for creating and evolving modeling artifacts are distributed differently, and technical barriers for participation are reduced. These examples illustrate how collaboration structures, ownership, and decision authority can vary across modeling scenarios. As a result, inconsistencies may emerge and be resolved under very different organizational dimensions, motivating the need for consistency management approaches that are aware of the dimensions of the scenario in which they operate.

3 Organizational Dimensions for Collaborative Modeling

To make organizational characteristics explicit, we propose a set of dimensions that characterize collaborative modeling scenarios and influence consistency management requirements. These dimensions are not intended as a definitive taxonomy, but rather as a framework for reasoning about when different consistency management strategies may be appropriate. Furthermore, these dimensions are not mutually exclusive, as a given scenario may involve multiple characteristics that interact in complex ways. Also, these dimensions affect all families of consistency management approaches, including checking, incremental analysis, repair, collaborative monitoring, and prevention, as they determine how inconsistencies are detected, tolerated, communicated, and resolved in practice. The dimensions include:

(i) **Tolerance of inconsistency.** Captures the extent to which a scenario can accept the presence of temporary inconsistencies. Some settings require immediate consistency because inconsistencies may violate safety, policies, or release constraints. Other settings can tolerate temporary inconsistencies because modelers are exploring alternatives, working asynchronously, or negotiating design decisions; (ii) **Timing of intervention.** This dimension describes when consistency management activities are performed relative to the evolution of artifacts. Intervention may occur proactively

before inconsistencies are introduced, immediately after changes are made, at synchronization points, or during dedicated review and integration phases; (iii) **Repair authority.** This dimension defines who is responsible for resolving inconsistencies once they are detected. Responsibility may lie with the author of the change, a designated model owner, a specific role (e.g., quality engineer), a project lead, the development team collectively, or an automated tool or agent that suggests or performs repairs; (iv) **Collaboration structure.** This dimension captures how many stakeholders participate in modeling activities and how they interact over time. Modeling may involve individual contributors or large groups, and it may occur synchronously or asynchronously. These characteristics influence communication needs, feedback expectations, and coordination; (v) **Consistency dependency across artifacts.** This dimension describes which artifacts are considered relevant for consistency management and how dependent they are. Consistency may be limited to a single artifact, extend across multiple related views, or span different abstraction levels such as requirements, models, design artifacts, and source code. It therefore determines the boundaries of what must be checked; and (vi) **Repair risk and side effects.** This dimension captures the potential impact of resolving inconsistencies. Some repairs are local and reversible, affecting only a small part of an artifact, whereas others may propagate across multiple artifacts, affect multiple stakeholders, or modify important design decisions with long-term consequences.

Together, these dimensions highlight that consistency management requirements emerge not only from the artifacts being modeled but also from the way collaboration is organized. These dimensions also interact with the development lifecycle, as the appropriate consistency management strategy may evolve with project maturity. For example, early development phases often tolerate greater inconsistency to support exploration, whereas later phases typically require stricter consistency as uncertainty decreases. Hence, a consistency management method that is appropriate in one scenario may be ineffective or even disruptive in another. For example, preventive inconsistencies may be desirable when ownership and decision authority are clearly defined, whereas tolerating inconsistencies may be preferable in highly collaborative scenarios.

4 Mapping Dimensions to Modeling Scenarios

The dimensions introduced define a set of organizational and contextual factors that shape consistency management requirements in collaborative modeling. In this section, we map these dimensions to representative collaborative modeling scenarios. Each scenario can therefore be understood as an instantiation of a particular combination of consistency dimensions, i.e., *tolerance for inconsistency*, *timing of intervention*, *repair authority*, *collaboration structure*, *consistency scope across artifacts*, and *repair risk and side effects*.

Rather than advocating a single best strategy for a particular modeling scenario, we argue that different configurations of the dimensions favor different consistency management strategies. Table 1 summarizes this perspective by mapping representative collaborative modeling scenarios to consistency dimensions, preferred strategies, and key challenges. This mapping should be interpreted as a design guide rather than a definitive classification. Also, noticed that due to space limitation we only reference a few of the approaches available in literature that align with each scenario. These are the proposed scenarios:

Exploratory co-design. Exploratory co-design corresponds to a modeling scenario characterized by high tolerance for inconsistency, low repair risk, and shared decision-making under synchronous collaboration. In such settings, inconsistencies may represent open design questions, different correct alternatives, or incomplete decisions rather than defects. Consequently, the emphasis is less on the *repair* family of approaches and more on the *collaborative and global consistency* family [1, 10, 16, 31], which can make inconsistencies visible and support awareness without forcing immediate resolution. Lightweight *checking* mechanisms [10, 13, 15, 16] may still be useful to identify emerging conflicts, but their role is primarily informational rather than corrective. The primary requirement is therefore not only to detect inconsistencies but also to preserve the context needed for future discussion and resolution.

Asynchronous individual contribution. This modeling scenario is characterized by moderate tolerance for inconsistency, delayed coordination, and localized repair authority, combined with asynchronous collaboration and weak artifact dependency. Here, the *incrementality and scalability* family is particularly relevant because contributors benefit from fast local feedback without requiring global re-analysis after every modification [4, 8, 11]. These incremental mechanisms are typically complemented by traditional *checking* approaches that periodically assess consistency across the broader project [31]. The key challenge is distinguishing *work-in-progress* inconsistencies from violations that genuinely require intervention, while balancing local autonomy and eventual global consistency.

Centralized integration. Centralized integration corresponds to scenarios with low tolerance for inconsistency, concentrated repair authority, and high-impact changes across coupled artifacts. In these settings, dedicated integration phases typically require comprehensive *checking* to identify inconsistencies across artifacts. Once inconsistencies are detected, the *repair* family becomes particularly important, as repair recommendation approaches can support informed decision-making while preserving human control over the final resolution [2, 20, 22, 23, 25]. Rather than emphasizing fully automated repair, consistency management should focus on ranking alternatives, exposing trade-offs, and explaining the potential consequences of each repair option.

Collaborative repair session. Collaborative repair emerges in situations where repair authority is distributed, collaboration is a higher priority, and artifacts exhibit strong dependency across multiple artifacts. In such contexts, the *repair* and the *collaborative and global consistency* families become tightly interconnected. Repair decisions require negotiation among participants, while shared awareness mechanisms help stakeholders understand the broader consistency state and coordinate their actions. Consistency management tools should therefore support ownership assignment, discussion, negotiation, and visibility of dependencies among inconsistencies and repairs. Work on relationships among inconsistencies suggests that understanding such dependencies can improve repair effectiveness and coordination [20].

Strict release or safety gate. Strict consistency gates correspond to scenarios with very low tolerance for inconsistency, strict timing of intervention, and centralized or highly controlled repair

Table 1: Mapping between collaborative modeling scenarios, organizational dimensions, and consistency management strategies.

Modeling Scenario	Key dimensions	Suitable strategy	Main challenge
Exploratory co-design (many authors, synchronous collaboration)	High inconsistency tolerance; low repair risk; synchronous collaboration; weak artifact dependency	Tolerate immediate detection with lightweight awareness	Avoid interrupting creativity while preserving knowledge about unresolved inconsistencies.
Asynchronous individual contribution with staged review	Medium tolerance; deferred intervention; low repair authority distribution; moderate artifact dependency	Incremental local checking plus deferred global checking	Distinguish work-in-progress inconsistencies from integration-relevant inconsistencies.
Centralized integration by a designated architect or integrator	Low tolerance; high repair authority concentration; high-risk repairs; cross-artifact dependency	Global checking with repair recommendations and prioritized repairs	Support informed decision-making without hiding trade-offs among possible repairs.
Collaborative repair session with shared decision-making among domain experts	Medium tolerance; shared repair authority; high collaboration intensity; strong artifact dependency	Shared inconsistency dashboard and negotiated repair alternatives	Make dependencies among inconsistencies and repairs understandable to all participants.
Strict release or safety gate with controlled changes and centralized authority	Very low tolerance; strict timing; centralized authority; high-risk repairs	Prevention, mandatory checking, and traceable repair execution	Balance assurance with the need to explain why edits are rejected or enforced.
Continuous model-code co-evolution across multiple artifacts and abstraction levels	Low tolerance; continuous intervention; distributed authority; strong vertical artifact dependency	Vertical consistency checking and change propagation	Prevent repairs in one artifact from unintentionally violating another artifact.

authority. These settings are typical in release processes, regulated environments, or safety-critical systems where inconsistencies cannot be tolerated even temporarily. Consequently, strong *checking* mechanisms are required to ensure compliance with consistency constraints, while prevention-oriented approaches from the *collaborative and global consistency* family can help avoid the introduction of invalid states altogether [1, 16, 33]. The primary challenge is balancing enforcement with usability by providing understandable explanations and acceptable alternatives when changes are rejected due to creating inconsistencies.

Continuous model-code co-evolution. Reflects scenarios characterized by strong artifact dependency across abstraction levels, continuous intervention, and distributed repair authority. Such environments require a combination of the *checking*, *incrementality and scalability*, and *repair* families. Continuous and often incremental consistency checking is necessary to maintain alignment across requirements, models, and source code, while repair and change-propagation mechanisms help restore consistency when violations occur [10, 16]. In addition, aspects of the *collaborative and global consistency* family are relevant because inconsistencies frequently span multiple artifacts and stakeholders.

5 Conclusion and Research Challenges

In this paper, we introduce a scenario-aware perspective on consistency management that considers the organizational context in which collaborative modeling takes place. While this perspective helps structure existing approaches, it also raises several open research challenges for developing consistency management techniques that better align technical mechanisms with organizational collaboration needs.

Flexibility in rule definitions. Existing approaches define consistency rules and checking mechanisms, but provide limited support for expressing the collaborative context in which those rules operate. Scenario-aware consistency management requires mechanisms for associating rules with organizational characteristics such as ownership (who is responsible for a rule), decision authority (who can approve or reject changes), collaboration structure (how stakeholders interact), and tolerance for temporary inconsistencies (how long inconsistencies can be tolerated). One direction is to leverage research on modeling organizational structures to better integrate these aspects into consistency specifications.

Informed repair decisions. While repair techniques can generate technically valid solutions, collaborative modeling requires decision support rather than isolated repair suggestions. Approaches should therefore focus on explaining who is affected, which artifacts change, whether repairs are reversible, and which stakeholders are responsible for approval. A promising direction is to leverage inconsistency dependency analysis to better characterize the impact and propagation of repairs, as well as to use natural language generation techniques, including large language models, to support human understanding of repair consequences.

Dynamic collaboration. Collaborative scenarios are not static. Teams may shift between exploratory design, structured review, asynchronous development, and tightly controlled integration. Consistency management approaches should therefore adapt to evolving collaboration contexts rather than assuming a single fixed operating mode. This motivates research on adaptive consistency mechanisms and dynamic configuration of consistency strategies across the project lifecycle.

Global awareness. Consistency information is often presented at the level of individual users, even when inconsistencies involve multiple stakeholders. Future approaches should support shared awareness by making relevant inconsistencies, dependencies, ownership information, and repair decisions visible across the team, while avoiding information overload. This requires context-sensitive filtering mechanisms that adapt the presentation of consistency information to roles, responsibilities, and current tasks.

From technical to organizational evaluation. Consistency management approaches are typically evaluated in terms of correctness, scalability, performance, or repair quality. A scenario-aware

perspective introduces additional evaluation concerns such as coordination effort, decision latency, stakeholder satisfaction, and collaboration effectiveness. This motivates the need for empirical evaluation methods, including controlled experiments with human participants and industrial case studies, despite the inherent challenges of accessing real-world collaborative settings.

Validating the proposed scenarios. The dimensions and scenarios presented in this paper should be interpreted as an initial conceptual framework rather than an exhaustive or systematically derived classification. Their purpose is to stimulate discussion on how organizational context influences consistency management and to make explicit assumptions that are often only implicit in existing approaches. As future work, we plan to conduct a systematic literature review to identify, validate, and refine these dimensions and scenarios based on evidence from existing consistency management research. In addition, we plan to investigate how the proposed dimensions can be integrated with existing collaboration and versioning feature frameworks [24].

Acknowledgments

This work was partially funded by the Natural Sciences and Engineering Research Council of Canada (NSERC), grant number RGPIN-2026-06532.

References

[1] Hyacinth Ali, Gunter Mussbacher, and Jörg Kienzle. 2020. Action-Driven Consistency for Modular Multi-Language Systems with Perspectives. In *Proceedings of the 12th System Analysis and Modelling Conference*. New York, NY, USA, 95–104. doi:10.1145/3419804.3420270

[2] Angela Barriga, Rogardt Heldal, Adrian Rutle, and Ludovico Iovino. 2022. PARMOREL: a framework for customizable model repair. *SoSym* (2022), 1–24.

[3] Antonio Bucchiarone, Jordi Cabot, Richard F Paige, and Alfonso Pierantonio. 2020. Grand challenges in model-driven engineering: an analysis of the state of the research. *Software and Systems Modeling* 19 (2020), 5–13. Issue 1.

[4] Jordi Cabot and Ernest Teniente. 2009. Incremental integrity checking of UML/OCL conceptual schemas. *Journal of Systems and Software* 82, 9 (2009), 1459–1478. doi:10.1016/j.jss.2009.03.009

[5] Luis M Camarinha-Matos and Hamideh Afsarmanesh. 2007. A comprehensive modeling framework for collaborative networked organizations. *Journal of Intelligent Manufacturing* 18, 5 (2007), 529–542.

[6] Hyun Cho, Jeff Gray, and Eugene Syriani. 2012. Creating visual Domain-Specific Modeling Languages from end-user demonstration. In *2012 4th International Workshop on Modeling in Software Engineering (MISE)*. 22–28. doi:10.1109/MISE.2012.6226010

[7] Istvan David, Kousar Aslam, Sogol Faridmoayer, Ivano Malavolta, Eugene Syriani, and Patricia Lago. 2021. Collaborative Model-Driven Software Engineering: A Systematic Update. *International Conference on Model Driven Engineering Languages and Systems*, 273–284. doi:10.1109/MODELS50736.2021.00035

[8] István Dávid, Joachim Denil, and Hans Vangheluwe. 2018. Process-oriented inconsistency management in collaborative systems modeling. In *16th International Industrial Simulation Conference*. 54–61.

[9] Istvan David, Hans Vangheluwe, and Eugene Syriani. 2023. Model consistency as a heuristic for eventual correctness. *Journal of Computer Languages* 76 (2023), 101223. doi:10.1016/j.cola.2023.101223

[10] Zinovy Diskin, Yingfei Xiong, and Krzysztof Czarnecki. 2010. Specifying Overlaps of Heterogeneous Models for Global Consistency Checking. *International Workshop on Model-Driven Interoperability*, 42–51. doi:10.1145/1866272.1866279

[11] Iris Groher, Alexander Reder, and Alexander Egyed. 2010. Incremental Consistency Checking of Dynamic Constraints., David S Rosenblum and Gabriele Taentzer (Eds.). *FASE* 6013, 203–217.

[12] Anne Gutschmidt, Birger Lantow, Ben Hellmanzik, Ben Ramforth, Matteo Wiese, and Erko Martins. 2023. Participatory Modeling from a Stakeholder Perspective: On the Influence of Collaboration and Revisions on Psychological Ownership and Perceived Model Quality. *Software and Systems Modeling* 22, 1 (2023), 13–29. doi:10.1007/s10270-022-01036-7

[13] R Jongeling. 2019. How to Live with Inconsistencies in Industrial Model-Based Development Practice. *International Conference on Model Driven Engineering Languages and Systems Companion*, 642–647. doi:10.1109/MODELS-C.2019.00098

[14] Robbert Jongeling, Federico Ciccozzi, Jan Carlson, and Antonio Cicchetti. 2022. Consistency management in industrial continuous model-based development settings: a reality check. *Software and Systems Modeling* 21 (8 2022), 1511–1530. Issue 4. doi:10.1007/s10270-022-01000-5

[15] Vasanthi Kaliappan and Norhayati Mohd Ali. 2018. Improving consistency of UML diagrams and its implementation using reverse engineering approach. *Bulletin of Electrical Engineering and Informatics* 7 (2018), 665–672. Issue 4.

[16] Djamel Eddine Khelladi, Benoit Combemale, Mathieu Acher, Olivier Barais, and Jean-Marc Jézéquel. 2020. Co-Evolving Code with Evolving Metamodels. In *International Conference on Software Engineering*. ACM, 1496–1508. doi:10.1145/3377811.3380324

[17] Elias Kuiter, Sebastian Krieter, Jacob Krüger, Gunter Saake, and Thomas Leich. 2021. variED: an editor for collaborative, real-time feature modeling. *Empirical Software Engineering* 26, 2 (2021), 24.

[18] Francisco J Lucas, Fernando Molina, and Ambrosio Toval. 2009. A systematic review of UML model consistency management. *Information and Software Technology* 51 (2009), 1631–1645. Issue 12. doi:10.1016/j.infsof.2009.04.009 Quality of UML Models.

[19] Nuno Macedo, Tiago Jorge, and Alcino Cunha. 2017. A Feature-Based Classification of Model Repair Approaches. *IEEE Transactions on Software Engineering* 43 (2017), 615–640. Issue 7. doi:10.1109/TSE.2016.2620145

[20] Luciano Marchezan, Wesley KG Assunção, Edvin Herac, Saad Shafiq, Cristian Knebel, and Alexander Egyed. 2026. Dependency-aware model repair: prioritizing repairs through consistency rule and inconsistency analysis. *Software and Systems Modeling* (2026), 1–31.

[21] Luciano Marchezan, Wesley K. G. Assunção, Gabriela Karoline Michelon, and Alexander Egyed. 2023. Do Developers Benefit from Recommendations When Repairing Inconsistent Design Models? A Controlled Experiment. In *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering* (Oulu, Finland) *(EASE '23)*. Association for Computing Machinery, New York, NY, USA, 131–140. doi:10.1145/3593434.3593482

[22] Christian Nentwich, Licia Capra, Wolfgang Emmerich, and Anthony Finkelstein. 2002. xlinkit: a consistency checking and smart link generation service. *ACM Trans. Internet Techn.* 2 (2002), 151–185. Issue 2.

[23] Manuel Ohrndorf, Christopher Pietsch, Udo Kelter, Lars Grunske, and Timo Kehrer. 2021. History-Based Model Repair Recommendations. *ACM Trans. Softw. Eng. Methodol.* 30 (1 2021). Issue 2. doi:10.1145/3419017

[24] Jakob Pietron, Alexander Raschke, Joeri Exelmans, and Matthias Tichy. 2023. Collaboration and versioning framework–a systematic top-down approach. In *2023 ACM/IEEE international conference on model driven engineering languages and systems companion (MODELS-c)*. IEEE, 767–777.

[25] Alexander Reder and Alexander Egyed. 2012. Incremental Consistency Checking for Complex Design Rules and Larger Model Changes., Robert B France, Jürgen Kazmeier, Ruth Breu, and Colin Atkinson (Eds.). *MoDELS* 7590, 202–218.

[26] Jan Scheffczyk, Peter Rödig, Uwe Borghoff, and Lothar Schmitz. 2004. Managing inconsistent repositories via prioritized repairs., Ethan V Munson and Jean-Yves Vion-Dury (Eds.). *ACM Symposium on Document Engineering*, 137–146.

[27] Harald Störrle. 2024. Engaging End-User-Modelers: An Action Research Study. In *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems* (Linz, Austria) *(MODELS Companion '24)*. Association for Computing Machinery, New York, NY, USA, 630–639. doi:10.1145/3652620.3688555

[28] Damiano Torre, Yvan Labiche, and Marcela Genero. 2014. UML Consistency Rules: A Systematic Mapping Study. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, 1–10. doi:10.1145/2601248.2601292

[29] Damiano Torre, Yvan Labiche, Marcela Genero, and Maged Elaasar. 2018. A systematic identification of consistency rules for UML diagrams. *Journal of Systems and Software* 144 (10 2018), 121–142. doi:10.1016/j.jss.2018.06.029

[30] Damiano Torre, Yvan Labiche, Marcela Genero, Maged Elaasar, and Claudio Menghi. 2020. UML Consistency Rules a Case Study with Open-Source UML Models. *Proceedings of the 8th International Conference on Formal Methods in Software Engineering*, 130–140. doi:10.1145/3372020.3391554

[31] M.A. Tröls, L. Marchezan, A. Mashkoor, and A. Egyed. 2022. Instant and global consistency checking during collaborative engineering. *Software and Systems Modeling* 21 (2022). Issue 6. doi:10.1007/s10270-022-00984-4

[32] Birgit Vogel-Heuser, Markus Böhm, Felix Brodeck, Katharina Kugler, Sabine Maasen, Dorothea Pantförder, Minjie Zou, Johan Buchholz, Harald Bauer, Felix Brandl, et al. 2020. Interdisciplinary engineering of cyber-physical production systems: highlighting the benefits of a combined interdisciplinary modelling approach on the basis of an industrial case. *Design Science* 6 (2020), e5.

[33] XC Zhang and Johannes F Broenink. 2013. A concurrent design approach and model management support to prevent inconsistencies in multidisciplinary modelling and simulation. In *19th European Concurrent Engineering Conference, ECEC 2013*. EUROSIS-ETI Publication, 21–28.