JOT
JOURNAL OF
OBJECT TECHNOLOGY

Journal of Object Technology | **RESEARCH ARTICLE**

Investigating Novice Modelers' Intuitive Consistency Notions for the Case of Compositional Models

Karl Kegel[*], **Kevin Feichtinger**[†], **Terru Stübinger**[†], **Romain Pascual**[‡], **Bernhard Beckert**[†], **Ralf Reussner**[†], **and Uwe Aßmann**[*]

[*]Dresden University of Technology, Germany
[†]Karlsruhe Institute of Technology (KIT), Germany
[‡]CentraleSupélec, Université Paris-Saclay, France

ABSTRACT The terms "consistency" and its counterpart, "inconsistency", are omnipresent in model-driven software engineering. Particularly during model evolution, consistency assurance is a central concern. Technically, consistency can be defined from various viewpoints. However, it is unclear how these definitions of consistency align with practitioners' mental models. Understanding how practitioners assess and respond to different kinds of consistency problems in their own minds is required for realizing successful consistency assurance. An important aspect towards this goal is the investigation of whether there is a shared mental model of consistency among an evenly experienced group of practitioners. The current state of the art gives no answer to this. In this work, we present a model dataset of 62 consistency problems based on simple, compositional models. For this dataset, we collected 452 consistency ratings from a student group and one rating per problem from an expert group. The results show that there is no common understanding, i.e., there is no shared mental model of consistency within a group or between groups. Still, common patterns in the ratings are observable, such as a preference for rating consistency on a fine-grained scale rather than as a boolean property, or that syntactic contradictions are much less relevant to ratings than subjective semantic interpretations. All developed tools, models, and collected data are openly available for future research.

KEYWORDS Software Modeling, Model Consistency, Model Evolution, Human Factors in Modeling

1. Introduction

Research on *consistency* is a long-standing topic in computer science. A first prominent cluster of research using the term *consistency* stems from database engineering. In database systems, consistency needs to be assured on many occasions, e.g., keeping a database consistent during transactional updates (Ellis 1977; Traiger et al. 1982), or assuring the compliance to integrity constraints (Bertossi 2006), naming just a few examples. But also in software engineering, the need for consistency assurance is inevitable (Nuseibeh 1996). Particularly in *Model-Driven Software Engineering (MDSE)*, *consistency* research became prominent with the rise of the *Unified Modeling Language (UML)*. In UML, multiple models can be defined to describe views of the same system. These views, or models, must be consistent to assure a coherent system design. Approaches for assuring consistency in UML have been surveyed, e.g., by Usman et al. (Usman et al. 2008) and Knapp and Mossakowski (Knapp & Mossakowski 2018). State-of-the-art approaches (Klare et al. 2021; Stevens 2018; Marchezan et al. 2023) aim to enable automated consistency checking, assurance, and repair within a coding or modeling project. In short, we refer to such tools and approaches as *Consistency Management Systems (CoMS)*.

But what exactly does *consistency*, *inconsistency*, or *being consistent* mean? Finkelstein once wrote *"The everyday concept of inconsistency is easy to grasp. It is simply saying something in one place and another contradictory thing in another place"* (Finkelstein 2000), which emphasizes contradictions being both effects and causes of inconsistency. A consequential step is to interpret these contradictions as a form of distance – which, in turn, makes *consistency* a form of similarity or com-

JOT reference format:
Karl Kegel, Kevin Feichtinger, Terru Stübinger, Romain Pascual, Bernhard Beckert, Ralf Reussner, and Uwe Aßmann. *Investigating Novice Modelers' Intuitive Consistency Notions for the Case of Compositional Models.* Journal of Object Technology. Vol. 25, No. 3, 2026. http://dx.doi.org/10.5381/jot.2026.25.3.a12

patibility.

Technical approaches for consistency checking and assurance typically rely on rules or constraints, e.g., *integrity constraints* (Bertossi 2006), *consistency specifications* (Klare et al. 2021), or *conforms* relations (Stevens 2018). In other words, an engineer constructively defines what *consistency* means in a given project or context. However, if consistency assurance and repair tools become established, developers and modelers with different technical backgrounds may be exposed to consistency rules and constraints that do not align with their expectations. This discrepancy has the potential to impact usability and acceptance. Therefore, we require empirical knowledge about how users think about consistency, i.e., what their *mental models* (Carroll & Olson 1988) of the term *consistency* are. However, as the current state of the art does not yet tackle this topic at all, we need to start our investigation at an early point. Before productively determining concrete mental models, we need to know how much of a shared *shared mental model* (Cannon-Bowers et al. 1993) exists among software engineers.

1.1. Context and Problem Statement

This work aims to investigate *human factors* (Wickens et al. 2004), specifically *mental models* (Carroll & Olson 1988), for the design and communication of consistency assessment methods in MDSE. In general, a CoMS in MDSE manages a set of models. The *consistency assessment method* realized by a CoMS determines whether its managed set of models is consistent. A *user* is the person who uses a CoMS during the development of their system. A user's *intuitive consistency assessment* is what they think about a consistency problem without being told or educated about a specific method.

For both educators and engineers, it is essential to understand what users mean by the term *consistency*. However, before one can search for an empirical definition of consistency, a prior question must be investigated. That is, if there can exist one single, intuitive definition of consistency at all. In other words, do people think generally the same about consistency? Do they have a shared opinion about what is consistent and what is not? To the best of our knowledge, no studies have investigated these questions about consistency in MDSE.

1.2. Objective

The objective of this work is to design and execute a study to investigate mental models of consistency in MDSE, specifically the existence of a shared mental model of consistency among software engineers. Because MDSE is a large field with many possible scenarios in which consistency can be investigated, we focus on simple model evolution and co-evolution cases. We ask the primary research question: ***RQA:* Do software engineers have a shared mental model of consistency when assessing model evolution and co-evolution scenarios?** We ask the secondary research question: ***RQB:* What are key observations when software engineers assess consistency? What are observable commonalities, differences, and contradictions?**

1.3. Approach & Scope

The following paragraphs outline the method, focus, and restrictions of this work.

Which kind of models do we present to the participants? We investigate consistency assessments for compositional models, i.e., models based on "composing" relationships. Such relationships may be *composed-of*, *part-of*, *contains*, or *subset*. We aim to keep the studied models small and comprehensible.

What kind of consistency problems do we present? We focus on inconsistencies resulting from model evolution. We investigate two general types of model evolution scenarios: two-model and three-model scenarios. A two-model evolution scenario is the simple evolution of a model A into a model A'. A three-model evolution scenario is the evolution of model A into two models, A'_1 and A'_2. We furthermore investigate the more complex case of two-model scenarios with linked models. Figure 1 illustrates these scenario types.

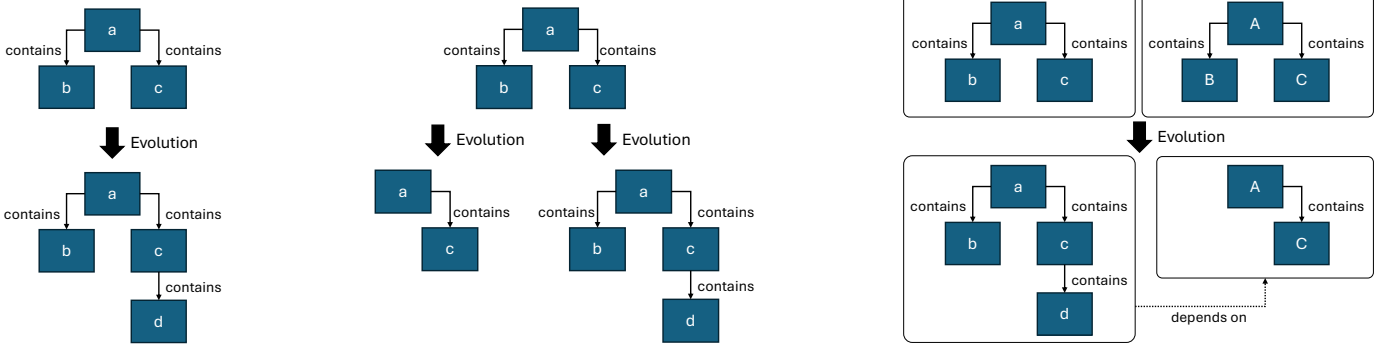
What is our target group? We focus on investigating the mental models of software engineering students (student group) who are familiar with software modeling but have not yet been influenced by a domain-specific work environment. In addition, we question a small group of modeling experts from MDSE research (the expert group) for comparison and reference.

What do we investigate in detail? Towards answering RQA and RQB, we investigate the following questions:

- Q1 Does the consistency assessment happen among groups in unison, or is it scattered?
- Q2 Is the consistency assessment rather a Boolean decision or a gradual decision?
- Q3 Do modeling experts and modeling novices assess consistency equally or differently?
- Q4 Is the consistency assessment identical or different for reversed operations in two-model scenarios, i.e., is it symmetrical?
- Q5 What is the influence of the types of model evolution operations on the consistency assessment?
- Q6 Is there an observable influence of subjective interpretations on the consistency assessment?
- Q7 Does the assessment differ in the case of more complex models, e.g., linked compositional models?

1.4. Method & Contribution

The remainder of this work starts with the presentation and discussion of background topics and related works in Section 2. We proceed with the design of a model dataset of consistency problems in Section 3. Section 4 presents the design of the study and proposes evaluation criteria for questions Q1-Q7. The results of the study are presented and discussed in Section 5. We close this work with a summary and conclusion in Section 7. We contribute: A manually authored open dataset of 62 model evolution problems; 452 collected consistency assessments of the aforementioned problems; A list of general conclusions and findings from evaluating the gained results. All tools and data are openly available in our supplementary material (7).



(a) Two-model evolution scenario. A schematic example of a simple model evolution.

(b) Three-model evolution scenario. A schematic example of two alternative evolutions of the same base model.

(c) Two-model evolution scenario with linked models. A schematic example of the evolution of two linked models.

Figure 1 Abstract representations of the three types of model evolution scenarios we investigate. The depicted abstract diagrams illustrate what we define as two-model, three-model, and linked model evolution scenarios.

2. Background & Related Works

2.1. Human Factors & Mental Models

"Human factors issues arise in every domain in which humans interact with the products of a technological society" (Carroll & Olson 1988). To design a good system, an engineer must always consider users' behavior and expectations beyond just technical requirements. A way to capture and describe a human's "way of thinking" is through mental models. Thus, understanding mental models has long been an important aspect of human factors research (Carroll & Olson 1988; Wickens et al. 2004).

For a historical overview of mental model research, we refer to the comprehensive article by Johnson-Laird (Johnson-Laird 2004). Right at the beginning of his work, Johnson-Laird presents the following quote by Craik (1954): *"If the organism carries a "small-scale model" of external reality and of its own possible actions within its head, it is able to try out various alternatives, conclude which is best of them, [...], utilize the knowledge of past events in dealing with the present and the future, [...]."* which well-emphasizes the concept of mental models as experience-based recipes for reasoning. Littman et al. empirically investigated which strategies developers use to understand existing code, i.e., how they construct their mental models of a system, to maintain it successfully (Littman et al. 1987). Latoza et al. survey the effort required to build and maintain mental models of a development project for individuals and teams. They found that understanding a piece of code (building a mental model of it) is considered a serious problem among developers (LaToza et al. 2006). Yu and Petter investigate the building and benefits of *shared mental models* (Cannon-Bowers et al. 1993) during agile software development (Yu & Petter 2014). A shared mental model describes the overlap among individuals' mental models within a team.

2.2. Consistency in MDSE

Consistency is a highly important topic in software engineering, particularly in MDSE. A practice in software engineering is the separation of concerns while building complex systems, as summarized in the *Viewpoints Framework* by Finkelstein et al. (Finkelstein et al. 1992). In MDSE, different specialized models, called views, are created for the different aspects of a system that must be consistent. An example is the consistency assurance across different UML diagram types, as surveyed by Usman et al. (Usman et al. 2008), and Knapp and Mossakowski (Knapp & Mossakowski 2018).

Recently, the understanding of consistency in interdisciplinary contexts, such as the model-driven engineering of *Cyber-Physical Systems (CPS)*, has attracted greater attention from researchers. CPS require a deeper understanding of consistency as the engineering domains between the viewpoints may strongly differ (Feichtinger et al. 2024). Consistency in CPS engineering is a multidimensional problem, where consistency relates model elements beyond the syntactical viewpoint, but also involves the semantics of the models (Pascual et al. 2025; Färber et al. 2026). Further, consistency may not be achieved immediately, and thus, may require allowing temporary inconsistencies during development, and use other notions, e.g., descriptive or quantitative notions of consistency to achieve overall consistency among models (Färber et al. 2026; Feichtinger et al. 2024). Additionally, dependencies and relations between different models and their elements need to be well communicated (Feichtinger et al. 2022; Voelk et al. 2025) to avoid inconsistencies. For that, the perception of consistency in the involved domains is a key factor. When discussing communication in interdisciplinary settings, differences in the semantics of terms and practices (Voelk et al. 2025) and various types of inconsistencies (Albers et al. 2024) have been found.

2.3. Consistency and Evolution

This work focuses on evolving models when investigating modelers' mental models of consistency. In other words, we look at how model transformations introduce inconsistency from a perceptual standpoint. For a broad overview on model transformations, i.e., model evolution, we refer to the taxonomy by

Mens and Van Gorp (Mens & Van Gorp 2006). Using their terminology, this work considers horizontal and vertical endogenous transformations, as well as horizontal exogenous transformations to a smaller extent. That model transformations can introduce inconsistencies is nothing new. But understanding, formalizing, and measuring how certain transformations affect a system's consistency is ongoing research. Kosiol et al. consider the consistency of graph models as a gradual property. They consider transformations to be either consistency-improving or consistency-sustaining, and formalize the properties of such transformations (Kosiol et al. 2020). A different case of the evolution problem is parallel work on different versions or variants of the same artifact, i.e., multiple lines of development. Also in this case, the increase and decrease in consistency can be measured as a gradual property (Kegel et al. 2025).

3. The CoModE Dataset

This section introduces the *Consistency in Model Evolution (CoModE)* dataset, which poses an independent contribution of this work. The CoModE dataset comprises data points, along with accompanying tools and artifacts. A *data point* is an example of a consistency problem, which comprises one or more models, as well as metadata describing the scenario. *Artifacts* beyond the set of data points are the documentation, metamodels, transformation, and visualization tools used and developed. This work uses the CoModE dataset *version 1 (v.1)*. For further information on the data set and the data creation process, we refer to the repository referenced in the supplementary material of this work (see Sec. 7).

We distinguish the terms *scenario*, *operation-combination*, and *data-point*. A scenario describes a specific type of evolution problem. For each scenario, we create a set of data points, i.e., concrete problems. Each data point uses a specific operation combination. The operation-combination describes which model evolutions are performed.

3.1. Design

This section summarizes the choices made in the design process of the CoModE dataset. We acknowledge that no single ideal dataset exists. Each design decision may have valid alternatives.

Open Access We publish the code and software artifacts under the open-source *GNU GPL-v3*[1] license. We publish non-code artifacts under the *Creative Commons CC-BY 4.0*[2] license.

Reusability / Open Standards We use XML Schema Definition (XSD) to define the required metamodels. The models are denoted as XML files. PlantUML visualizations are generated from XML files via transformations.

Comprehensibility We keep the data points general and simple by using directed graph models. Graphs are part of the curriculum of most software engineering courses and the syntactic foundation of many modeling languages. We visualize the models in a simple graphical representation. We restrict the graph models to only contain elements that can also be visualized.

Extensibility The CoModE repository contains all tools and documentation to create new data points. The provided XSD schemata verify the correctness and conformity of newly created XML models. We provide a template for creating new data points in the repository.

3.2. Technology & Process

The data points in the CoModE dataset are created by hand. The developed tools support this process. The following steps explain the process of creating a new data point. We refer to the creator of data points as *maintainer*. For CoModE v.1, we, the authors of this work, are the maintainers of all data points.

1. The maintainer creates a base model as an XML file in accordance with the provided XSD graph metamodel. The base model shows the initial state of a model before change operations are applied.
2. The maintainer creates one or more change models as XML files. We provide a change metamodel in XSD. Simply put, a change model is a structured log of change operations representing a model transformation/evolution.
3. The maintainer runs the provided transformation tools to generate the evolved models and their visualizations. Committing the data point to the CoModE repositories automates this step via continuous integration. First, the change models are applied to the base model, producing the evolved models. The base model and the evolved models are then transformed into a graphical representation. *PlantUML (PUML)* serves as the intermediate format. The PNG images showing the model evolution cases are stored in the output directory.

In consequence, the maintainer creates base models and transformation models in XML. The participants receive rendered images of the base model and the evolved model(s). The participants do not receive the explicit transformation models.

3.3. Metamodels

The CoModE dataset is based on a graph metamodel and a transformation (change) metamodel. We provide both metamodels in our supplementary material (see Sec. 7).

The models that form the model evolution cases are graph-based models. The graph metamodel is defined as an XML Schema Definition (XSD). A *graph* consists of *groups*, *nodes*, and directed *edges*. A group is a collection of nodes. The edges are labeled and directed. Nodes may contain attributes. We do not restrict instances of the metamodel to reside on a specific level of abstraction. Instances can be object nets, class structures, automatons, etc.

To represent changes, we require a change metamodel. We define this metamodel using XSD. The change operations part of an evolution are grouped as *semantic edits*. A semantic edit is an indexed list of change operations that are annotated with an evolution category. For this work, we rely on the following

[1] https://www.gnu.org/licenses/gpl-3.0.en.html
[2] https://creativecommons.org/licenses/by/4.0/

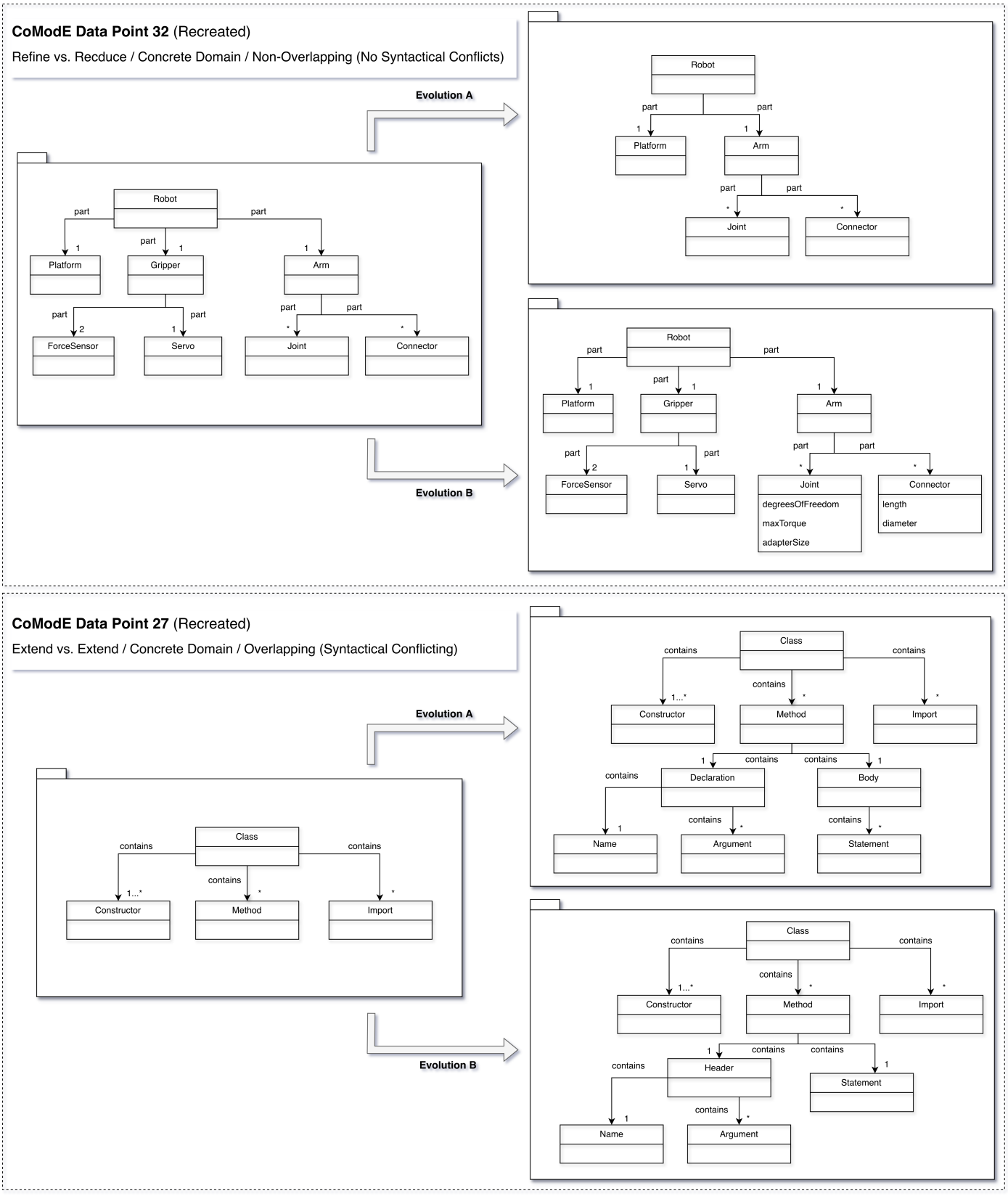


Figure 2 Two exemplary data points from the CoModE dataset Both data points (model evolution scenarios) present three-way evolutions. The top case shows the evolution in two variants with no syntactical conflicts. The evolutions in the bottom case would have syntactic conflicts, if merged. Both cases are recreated and reformatted, the original models, as presented to our participants, are visualized using PlantUML. We will refer to these examples again for reference when discussing the study results in Section 5.

evolution categories: *Refine*, *Simplify*, *Extend*, *Simplify*, *Reduce*, *Transfer* (change domain), and *Rename*. Simply put, a change model is a structured log of change operations, annotated with the reasons for each change.

3.4. Data Points

For answering Q1-Q7 (cf. Section 1.3), we require a set of data points that covers the different aspects of interest and is sufficiently diverse but remains manageable. For an overview of the different evolution scenarios, we again refer to Figure 1. Table 1 lists the data points included in the CoModE v.1 dataset. For the model's XML files and visualizations, we refer to the supplementary material (see Sec. 7). Figure 2 presents two data points, i.e., model evolution cases, from the CoModE dataset. Both points are included in the study. The visualizations shown are recreations of the original PlantUML diagrams. In the study, the participants will receive each data point in a similar format, with the base model on the left and the evolved models on the right.

Domains We create models in both abstract and concrete domains. In the abstract domain, we use alphanumeric names for the *nodes*. For the concrete cases, we use the domains of cyber-physical systems, software engineering, and the general known world (animals, family relations, etc.). For each scenario and operation combination, we create data points across all four domains, except for the renaming operation, for which we omit the abstract domain.

Scenarios We distinguish between two-model and three-model scenarios as presented in Fig. 1. In a two-model scenario, a model evolution operation is applied to transform a model A into a model A'. For each two-model data point, we also consider the reverse as a data point. Additionally, we create a small set of data points of two-model scenarios with linked models. These data points (71-74) present more complex evolution cases where each model consists of two linked sub-models. In a three-model scenario, we apply two evolution operations in parallel. A base model is transformed into two alternative evolved models by different evolutions. We consider the operation pairs refine-refine, extend-extend, refine-reduce, and extend-reduce.

Provoking Inconsistencies We require a part of the data points to be potentially inconsistent. For the two-model scenarios, we have no general prior intuition about which evolutions, i.e., before-after combinations, may be regarded as inconsistent. Only in the case of *transfer*, we purposefully include contradictions by transferring between unrelated domains. In the three-model scenarios, we treat overlapping edits (syntactical merge conflicts) across both lines of evolution as potential inconsistencies. For each three-model scenario, domain, and operation combination, we create one data point without a potential inconsistency and one with.

Number of Data Points We created 62 data points. Note that the data point IDs are not sequential due to organizational constraints during the dataset creation. Out of these 62 data points, 26 are two-model scenarios (1-13, 51-63). Half of the two-model scenarios are explicitly modeled and part of the CoModE dataset (1-13). The other half is derived by swapping the order of the base model and evolved model (51-63). 32 out of the 62 data points are three-model scenarios (14-45). Four data points present linked models (71-74). Of the 58 two- and three-model data points, 14 use the abstract domain. The other data points use concrete domains.

4. Study Design

4.1. Objective

This study aims to answer the initially stated questions Q1-Q7 (see Section 1.3). Using these answers, this study subsequently aims to answer the central question of whether a shared mental model of consistency exists within a group of software engineering novices (RQA). We furthermore want to gain a broad initial understanding of the perception of consistency, including tendencies and patterns (RQB). The objective of this study is not to find a specific consistency definition.

4.2. Participants

The "students group" comprises undergraduate computer science students at the Dresden University of Technology. We question students in the last two weeks of the introductory software engineering course. Based on the curriculum, the students are familiar with different diagram types of the UML and the concept of MDSD. We also question three modeling experts. The "experts group" comprises researchers with expertise on model evolution, metamodeling, and MDSE for robotics. The experts are affiliated with the authors of this paper but have no involvement in this research beyond participating in the study. They are not part of the author list and have no personal interest in the outcome of this study.

4.3. Process

We use different questioning processes for the student and expert groups, which we present below.

Students Group Process

1. We prepare question packages containing seven randomly selected consistency problems (data points) from the CoModE dataset each. We balance the distribution of two- and three-model scenarios across packages. We attach a questionnaire (Section 4.4) to each problem. Each problem (visualized models) is printed on a page of paper, and the questionnaire is attached to the page. Each question package includes a cover page with the basic instructions.
2. We join the students' seminar sessions of the software engineering course and distribute the question packages to the students. We visit several sessions over the course of one week. Each student receives one question package and is asked to answer it. The students get 15 minutes to complete the question package.
3. We transcribe the answered questionnaires to a digital spreadsheet. This data is the input for the subsequent analyses.

Table 1 Data Points in the CoModE v.1 Dataset Overview of the 62 authored data points. The second column indicates whether it is a two-model or a three-model scenario. In two-model cases, the reverse operations are denoted in brackets. *Conflicting* samples are purposefully designed to contain potential inconsistency, e.g., by overlapping edits that could cause contradictions. Note that the data point IDs are not sequential due to organizational constraints during dataset creation.

IDs (Reverse)	Scenario Type	Operations	Domain	Conflicting	Samples (Reverse)
1 (51)	2 Models	Refine (Simplify)	abstract		1 (+1) = 2
2 (52), 3 (53)	2 Models	Refine (Simplify)	concrete		2 (+2) = 4
4 (54)	2 Models	Extend (Reduce)	abstract		1 (+1) = 2
5 (55), 6 (56)	2 Models	Extend (Reduce)	concrete		2 (+2) = 4
7 (57), 8 (58)	2 Models	Rename (Rename)	concrete		2 (+2) = 4
9 (59)	2 Models	Transfer (Transfer)	abstract		1 (+1) = 2
10 (60), 11 (61)	2 Models	Transfer (Transfer)	concrete		2 (+2) = 4
12 (62), 13 (63)	2 Models	Transfer (Transfer)	concrete	yes	2 (+2) = 4
14	3 Models	Refine vs. Refine	abstract	no	1
15, 16, 17	3 Models	Refine vs. Refine	concrete	no	3
18	3 Models	Refine vs. Refine	abstract	yes	1
19, 20, 21	3 Models	Refine vs. Refine	concrete	yes	3
22	3 Models	Extend vs. Extend	abstract	no	1
23, 24, 25	3 Models	Extend vs. Extend	concrete	no	3
26	3 Models	Extend vs. Extend	abstract	yes	1
27, 28, 29	3 Models	Extend vs. Extend	concrete	yes	3
30	3 Models	Refine vs. Reduce	abstract	no	1
31, 32, 33	3 Models	Refine vs. Reduce	concrete	no	3
34	3 Models	Refine vs. Reduce	abstract	yes	1
35, 36, 37	3 Models	Refine vs. Reduce	concrete	yes	3
38	3 Models	Extend vs. Reduce	abstract	no	1
39, 40, 41	3 Models	Extend vs. Reduce	concrete	no	3
42	3 Models	Extend vs. Reduce	abstract	yes	1
43, 44, 45	3 Models	Extend vs. Reduce	concrete	yes	3
71	2 Models (Multi. Hierarchies)	Extend (one sided)	concrete		1
72	2 Models (Multi. Hierarchies)	Extend (two sided)	concrete		1
73	2 Models (Multi. Hierarchies)	Extend (two sided)	concrete		1
74	2 Models (Multi. Hierarchies)	Reduce (two sided)	concrete		1

We inform the students that their participation is voluntary, anonymous, and they can stop at any time. We do not collect any personal data. Each rated problem is regarded as an individual result. If a student only rated $n < 7$ out of the seven handed-out problems, we consider the n problems as results. If the questionnaire remains blank, we exclude it.

Experts Group Process

1. We invite the three experts into a moderated meeting.
2. The moderator introduces the rating task to the experts.
3. The moderator presents each consistency problem from the CoModE dataset one by one on a projector. The problems are presented in a random order. The presentation has the same format as the problems printed out for the students. The experts receive the same rating questions as the students, except the confidence rating (see Section 4.4).
4. The experts have about 1 minute to discuss each problem and reach a consensus on the assessment. The moderator supports the consensus-finding process, e.g., by encouraging voting or by hinting at the time spent. The moderator records the joint ratings. The moderator also notes comments raised by the experts during the discussion.

The experts' participation is voluntary and anonymous to outsiders. The experts were informed that they can abort the rating process at any time.

4.4. Questionnaire

We designed questionnaires to conduct the consistency assessments. The questionnaires exist in three variations: *student*, *student-linked*, and *expert*. The questionnaires, as printed, are part of the supplementary material.

The *student* variation is used for data points 1-63 in the students group. It comprises: Marking the problem as not understandable or unable to rate; Boolean rating of the consistency (Yes/No); Gradual rating of the consistency (9P Likert: Low consistency ... high consistency); Confidence ratings for both assessments (2x 5P Likert: Low...high); Field for comments. The *expert* variation consists of the same questions, except for the confidence ratings. The *student-linked* questionnaire is used for the linked models (71-74). It comprises: A Boolean consistency rating before and after the evolution; A 9P Likert rating after the evolution; A question about how the evolution is perceived (consistency... improving/decreasing/neutral).

4.5. Analysis Criteria

We now present the criteria under which we answer questions Q1-Q7. For the analysis of Q1 to Q6, we exclude the data points 71-74. They are treated in Q7 exclusively.

Happens the consistency assessment among groups in unison, or is it scattered? (Q1) We answer this question based on the students' ratings. We formulate the criteria *C1:* The boolean

consistency assessment is in unison if for at least 90% of the problems a supermajority of at least 66.7% agrees on the rating. If a weaker version of the criteria (C1') is met for at least 66.7% of the problems, we consider the consistency assessment in unison but with insecurity.

For the rating on a 9-point Likert scale, we consider the consistency assessment to be in unison when we observe a compact distribution of ratings. *C2:* The gradual consistency assessment is in unison if the range $AVG \pm 1SD$ of the ratings lies within three points on the scale for at least 90% of the consistency problems, i.e., if $2SD \leq 3$. If a weaker version of the criteria (C2') is met for at least 66.7% of the consistency problems, we consider the consistency assessment to be in unison but with insecurity.

Is the consistency assessment rather a Boolean decision or a gradual decision? (Q2) We answer the question using the students' ratings. If the consistency assessment is a Boolean decision, the Likert ratings should be close to the extremes. *C3:* The consistency assessment is a Boolean decision if the AVG of the ratings lies within the first two values of the range $(1, 2)$ or $(8, 9)$ corresponding to the Boolean rating for at least 90% of the problems. If a weaker version of the criteria (C3') is met for at least 66.7% of the consistency problems, we consider the consistency assessment to be still Boolean but with insecurity.

Do modeling experts and modeling novices assess consistency equally or differently? (Q3) We consider the students' ratings as being similar to the experts' ratings if: *C4:* The students' Boolean averages are equal to the experts' Boolean ratings for a supermajority of more than 66.7% of the cases; and *C5:* The experts' ratings lie within the range $AVG \pm 1SD$ of the students' ratings for at least 90% of the cases. If both criteria are met, we consider the consistency assessment of the students and experts to be equal. If only C4 is met, we consider the Boolean consistency assessment to be equal. If only C5 is met, we consider the gradual consistency assessment to be equal. Otherwise, we consider the assessments as different.

Is the consistency assessment identical or different for reversed operations in two-model scenarios, i.e., is it symmetrical? (Q4) We formulate the criteria: *C6:* The consistency assessment for reversed operations in two-model scenarios is equal if the Boolean ratings are equal for at least 90% of the cases. We consider the assessment to be not equal but similar if the Boolean ratings are equal for at least 66.7% of the cases (C6').

What is the influence of the types of model evolution operations on the consistency assessment? (Q5) We evaluate this question without specific criteria, using a general quantitative and qualitative analysis of the results.

Is there an observable influence of subjective interpretations on the consistency assessment? (M6) We evaluate this question without specific criteria, based on students' confidence ratings and comments that explain their reasoning.

Differs the assessment in the case of more complex models, e.g., linked compositional models? (M7) To answer this question, we quantitatively analyze the results of the linked cases and conclude potential differences relative to the other cases. The investigation of Q7 is not within the main focus of this work, but rather for outlining future research directions.

Table 2 The collected rating counts from the students group

Scenario Type	Two-model	Three-Model	Linked
Ratings	191	215	46
Unproductive	9	8	5
Productive	182	207	41
No. Data Points	26	32	4
Avg. Ratings/DP	7.0	6.5	10.25

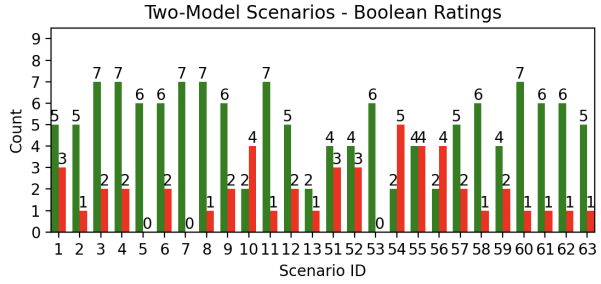
5. Results

This section starts with the presentation of general findings and conclusions. We proceed with answering the questions Q1-Q7 subsequently. We present the key findings and conclusions in outlined boxes. Findings are results from quantitative evaluation, while conclusions are results from qualitative evaluation and data interpretation.

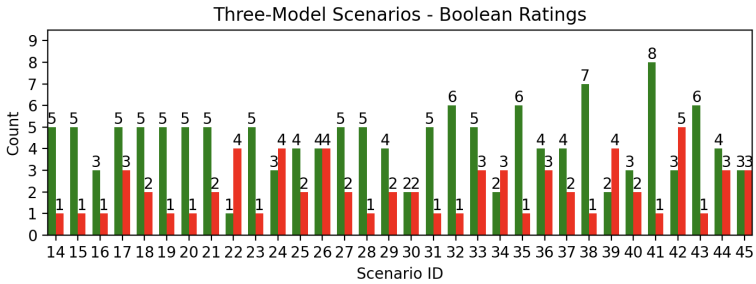
5.1. General Findings

Participation and Data Collection We received 452 answered questionnaires (rated scenarios) from 100+ students. Each of these students answered the questionnaires for one to seven consistency problems. The exact number of participants is not known due to the interruptible and anonymous questioning process. Table 2 shows the counts of the ratings collected. The number of productive ratings is equal to or exceeds the targeted minimum of 5 per data point, except for two data points (13, 16), for which we received 3 and 4 productive ratings, respectively. We keep both scenarios in the analysis, as they do not pose outliers rating-wise. From the expert group, we collected one consensual productive rating per data point.

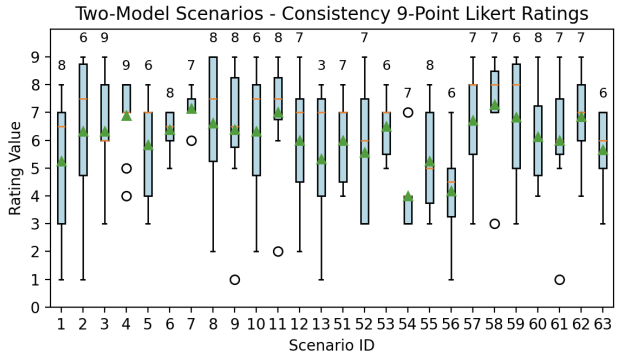
Students' Ratings Figure 3 shows the results for the two-model and three-model ratings of the student group. Figure 3a shows the Boolean results for the two-model scenarios. Except for the data points 10, 54, 55, and 56, the majority of problems are considered Boolean consistent by a majority of ratings. The three data points 10, 54, and 56, which were majority-wise rated as not consistent, present reduction operations (54, 56) and a transfer operation (10). Figure 3c shows the Likert ratings of the two-model scenarios. The rating average (per case) is 6.11; the average standard deviation is 1.95. This shows that the data points were rated as rather consistent, but with large insecurities. The plots show that the participants considered the full range of the rating scale. This indicates that the Boolean ratings alone do not provide a sufficiently precise picture of the consistency assessment. During the study design, we expected that all two-model scenarios, except transfer, would be considered clearly consistent because they show only simple "from-A-to-B" model evolutions. However, the results show that participants rated the scenarios with notable variation and differentiated among them.



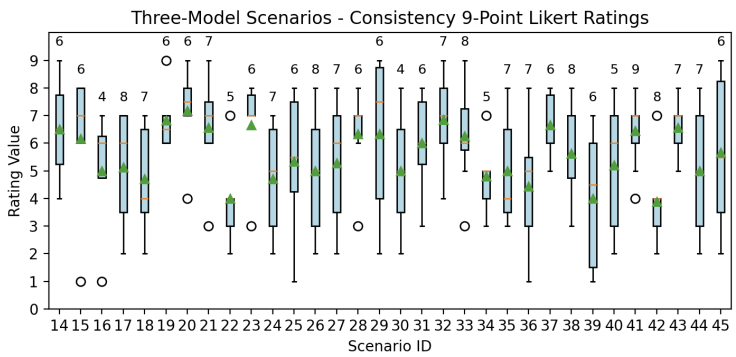
(a) The boolean rating distribution for two-model scenarios.



(b) The boolean rating distribution for three-model scenarios.



(c) The gradual rating distribution for two-model scenarios.



(d) The gradual rating distribution for three-model scenarios.

Figure 3 Rating distributions for the different scenarios and answer types. The Boolean ratings (top) show "YES" (consistent) ratings in green and "NO" (inconsistent) ratings in red. The gradual ratings (bottom) show the assessments on a 9/point Likert scale ranging from "9: High Consistency" to "1: Low Consistency".

Finding: Model evolutions/transformations $m \rightarrow m'$ are not clearly considered consistent. There is a non-negligible amount of rating variation on the observed scales.

The bar chart in Figure 3b shows the boolean ratings for the three-model scenarios. We see a majority of "YES" (high consistency) ratings. The only cases with a majority of "NO" ratings are data points 22, 24, 39, and 42. In contrast to the two-model scenarios, we see no specific reason, such as operation type, why these scenarios were rated as not consistent. The inconsistencies provoked by introducing syntactic contradictions into some scenarios have no clear impact on the rating.

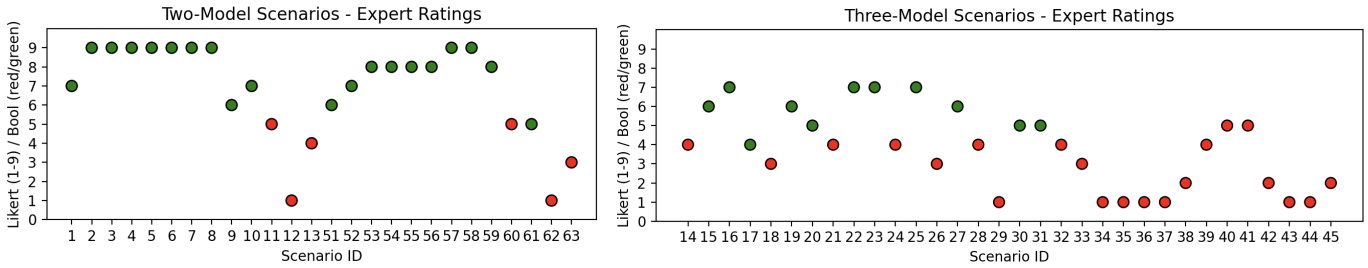
Finding: Syntactical contradictions in parallel model evolution scenarios do not visibly impact the consistency assessment of the case. The student group does not primarily base consistency on syntactic compatibility.

The box plot in Figure 3d shows the ratings of the three-model scenarios on a 9-point Likert scale. The rating average (per case) is 5.6; the average standard deviation is 1.87. Although the average rating tends towards high consistency, it is lower than for the two-model scenarios.

Participants were asked to rate their confidence in each rating on a 5-point Likert scale from "5: High Confidence" to "1: Low Confidence". For the boolean ratings, the average confidence value is 3.70 (SD 0.89) for the two-model and 3.52 (SD 0.96) for the three-model scenarios. The gradual ratings show slightly lower average confidence: 3.41 (SD 1.05) for the two-model and 3.25 (SD 1.08) for the three-model scenarios. The confidence ratings show that the participants were rather confident in their ratings, but not overly confident.

Looking at the two cases 27 and 32 presented as examples in Fig. 2, we received the following results: Case 27 has an average consistency rating of 5.33 on a 9-point Likert scale with a large standard deviation of 2.43. The average Boolean rating is 0.5, meaning that equally many students voted for "consistent" as voted for "inconsistent". Case 27 shows two changes that would lead to a syntactic conflict if merged. However, one could also say that both evolutions essentially convey the same information at a semantic level. Overall, the participants are far from a shared opinion. Case 32, which shows no syntactic conflicts at all, has an average consistency rating of 6.86 on a 9-P Likert scale (SD 1.64) and a Boolean average of 0.86. With that, case 32 is one of the few data points that are clearly rated by a clear majority to show consistent models after evolution. Thus, for this case, one could assume a shared opinion of the participants. Still, the standard deviation remains large. We also cannot explain why the ratings are this way for case 32, as the similar neighboring data point 33 again shows an indecisive Boolean rating and a lower Likert rating. We conclude that although the participants share an opinion on a few cases, this alignment is not constant. We see no evidence that the rating results are safely predictable on a per-case basis.

All of the aforementioned values are directly taken from Table 4 and Fig. 3.



(a) The experts' combined ratings for the two-model scenarios. **(b) The experts' combined ratings for the three-model scenarios.**

Figure 4 The combined experts' ratings for the two-model and three-model scenarios. The positions of the points show the rating of the experts on a 9-point Likert scale from "9: High Consistency" to "1: Low Consistency". The color of the points indicates the boolean rating, with green for "YES" and red for "NO".

Students' Comments We received 16 comments from the student group for the two-model scenarios, ranging from single words to a few sentences or bullet points. Some comments consider details highly specific to the assessed data point. However, most comments indicate that "consistency" is considered a soft property assessed on the "soundness" of the evolution scenario. Examples are: "[...] Reasonable renaming does not decrease consistency", "Only one component and some attribute[s] change, but the model is consistent". One comment shows that consistency is being thought of as a form of equality. We received 22 comments for the three-model scenarios. Many comments consider consistency as a form of semantic soundness. A few comments state that the participants did not base the consistency assessment on comparisons of the model variants, but rather on their individual differences from the base model. None of the comments provides evidence that syntactical mergeability is considered an important factor.

Conclusion: The students' consistency assessment is mostly based on soft factors, such as the soundness of the scenario. If an evolution scenario "makes sense" from a participant's perspective, it is likely considered consistent.

Experts' Ratings The experts' ratings are presented in Fig. 4. In the two-model scenarios, we observe a strong tendency toward positive ratings, with 14 data points rated 8 or 9 (high consistency) on the 9P Likert scale. Only six two-model data points were rated as Boolean inconsistent with the matching Likert ratings. These data points (11-13, 60, 62, 63) present evolutions with *transfer* operations. The three-model scenarios were rated as more diverse on both the Boolean and Likert scales. Notably, 14 of 16 points presenting cases in which one line of evolution performs a reduction operation were rated as inconsistent by the experts. Apart from that, no effects of the evolution operation are visible. The intentional inclusion of syntactical contradictions also has no visible effect.

Experts' Feedback The expert group provided no explicit comments, but the moderator collected verbal feedback and remarks during the rating process. The compiled summary of the notes was approved by the expert group afterward. We present the summary in the following: *"The three experts stated their understanding of all scenarios and considered them meaningful model evolution cases. During the scenario rating, the discussion was lively but always led to a consensus that everyone accepted. No expert vetoed a rating. To assess consistency, the experts first sought to build a shared understanding of the scenario. As they had no issues with understanding the syntax, they focused on the semantic aspects. The experts tried to understand why the models exist and what the evolution scenario aims to achieve. During this process, they tried to keep their interpretations free from non-modeled aspects. In the two-model scenarios, the experts focused on the evolution that was performed. They considered evolutions consistent that preserve the models' semantics or extend their worlds. Model pairs resulting from evolutions that removed information were considered to be rather inconsistent. In the three-model scenarios, the experts assessed the compatibility of the evolved models. Model variants that were considered semantically contradicting or generally not sound were considered not consistent. The aspect of syntactic contradictions, e.g., mergeability, was sometimes part of the discussion but had only a limited impact on the final rating. Potential merge conflicts were often detected, but not considered an opposing factor for a consistent pair of models. The nature of the experts' assessments, based on the semantic interpretation of the models, leads to issues when assessing abstract scenarios with no evident semantics. The reasoning had to be based mainly on syntax. These cases lead to the heaviest discussions."*

Conclusion: The experts' consistency assessment is mostly based on the semantic interpretation of the models and their compatibility. Syntactical aspects have a low impact.

5.2. Criteria Evaluation

This section evaluates the criteria C1-C6 based on the rating results. We present the detailed results of the students' group in the Tables 3 and 4. A visual overview is given in Figure 3. The expert's ratings are shown in Figures 4a and 4b.

C1 We count 35 cases in which a supermajority of votes agree on the same Boolean rating, and 23 cases in which they do not. This leads to agreement in 60.34% of cases. Neither C1 nor C1' is met.

Table 3 Rating results of the two-model scenarios Overview of the numerical results. "9-P" refers to the rating on the 9-point Likert scale. "BOOL" refers to the Boolean rating (Yes: 1; No: 0).

Data Point	1	2	3	4	5	6	7	8	9	10	11	12	13
9-P AVG	5.25	6.33	6.33	6.89	5.83	6.38	7.14	6.62	6.38	6.33	7.00	6.00	5.33
9-P SD	2.38	2.92	1.83	1.37	2.03	0.70	0.64	2.60	2.45	2.49	2.12	2.39	3.09
BOOL AVG	0.62	0.83	0.78	0.78	1.00	0.75	1.00	0.88	0.75	0.33	0.88	0.71	0.67
Data Point	51	52	53	54	55	56	67	58	59	60	61	62	63
9-P AVG	6.00	5.57	6.50	4.00	5.25	4.17	6.71	7.29	6.83	6.12	6.00	6.86	5.67
9-P SD	1.51	2.38	1.12	1.31	1.85	1.86	2.12	1.91	2.41	1.76	2.39	1.64	1.49
BOOL AVG	0.57	0.57	1.00	0.29	0.50	0.33	0.71	0.86	0.67	0.88	0.86	0.86	0.83

Table 4 Rating results of the three-model scenarios Overview of the numerical results. "9-P" refers to the rating on the 9-point Likert scale. "BOOL" refers to the Boolean rating (Yes: 1; No: 0).

Data Point	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
9-P AVG	6.50	6.17	5.00	5.12	4.71	6.83	7.17	6.57	4.00	6.67	4.71	5.33	5.00	5.29	6.33	6.33
9-P SD	1.71	2.48	2.35	2.03	1.83	1.07	1.57	1.76	1.67	1.70	1.91	2.43	2.18	2.19	1.60	2.81
BOOL AVG	0.83	0.83	0.75	0.62	0.71	0.83	0.83	0.71	0.20	0.83	0.43	0.67	0.50	0.71	0.83	0.67
Data Point	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45
9-P AVG	5.00	6.00	6.86	6.25	4.80	5.00	4.43	6.67	5.62	4.00	5.20	6.44	3.88	6.57	5.00	5.67
9-P SD	2.24	1.73	1.64	1.71	1.33	1.85	2.13	1.11	1.58	2.45	2.32	1.26	1.36	1.05	2.27	2.69
BOOL AVG	0.50	0.83	0.86	0.62	0.40	0.86	0.57	0.67	0.88	0.33	0.60	0.89	0.38	0.86	0.57	0.50

C2 This criterion is met for cases with $SD \leq 1.5$. We count 12 out of 58 such cases. This leads us to regard only 20.69% of the cases as being rated in unison. Neither C2 nor C2' is met.

Finding: The consistency assessments by the student group do not happen in unison. There is no strong agreement in the majority of cases for both Boolean and gradual ratings.

C3 We count no cases for which the AVG is ≤ 2 or ≥ 8. Neither C3 nor its weaker version C3' is met.

Finding: The consistency assessments on the gradual scale make use of the full rating range and are not concentrated on the extreme values. Consistency does not appear to be assessed Boolean.

C4 In cases where the number of "Yes/No" votes in the student group is equal, we count it as "Yes". We count 31 ratings made in agreement and 27 ratings in disagreement. Only 53.45% of the ratings are in agreement. Criteria C4 is not met.

C5 We count 27 of 58 cases in which the experts' ratings lie within that range. This leads to 46.55% of the cases agreeing. Criteria C5 is not met.

Finding: The students and experts rate consistency differently. There is no strong agreement in the majority of cases for both Boolean and gradual ratings.

C6 For the expert group, we count 2 out of 13 reversed operation pairs with non-equal Boolean ratings. This results in 84.62% of cases being rated symmetrically. Criteria C6 is not met for the expert group. For the student group, we count 3 out of 13 reversed operation pairs with non-equal Boolean ratings. This results in 76.92% of cases being rated symmetrically. Criteria C6 is not met for the student group. Notably, the student group's ratings differ strongly in multiple cases while still agreeing on the same majority.

Finding: There is weak evidence that the direction of the model evolution operation does not impact the consistency assessment in a two-model scenario. In some cases, a model evolution $m \rightarrow m'$ is considered consistent, while its reverse $m' \rightarrow m$ is not.

5.3. Analytical Evaluation

Q5 In general, we observe no strong influence of specific operations on the consistency assessment. Only for reduction and transfer (renaming) operations, we see a weak tendency towards low consistency ratings in the student group. This aligns with the expert's feedback that information loss is considered problematic in model evolution.

Q6 We observed no strong personal bias in the expert's interpretation of the scenarios, but extensive discussions occurred during the consistency assessment. Oftentimes, experts agreed that a particular data point showed inconsistency but disagreed about its severity. For the student group, several comments reflect varying subjective interpretations. We observe that: (1) Some participants base their assessment on syntactical details; (2) Some participants consider large-scale semantic and structural aspects and disregard smaller details; (3) The consistency assessment does not always happen within the closed world given by the models, but is influenced by the participant's ideas about the modeled domain, e.g., how things "should" be modeled. We are not confident in drawing general conclusions from the comments. At best, we can conclude that contextual information is important for guiding the consistency assessment.

Q7 In the previous analysis, we excluded the four data points that present linked compositional models. The participants were asked to rate the consistency of these cases before and after the evolution, as well as how the evolution impacts consistency (improving, decreasing, neutral). Data points 71 and 72 present scenarios in which one model has a "requires" relationship to another. This relationship is satisfied before the evolution, but

(from the authors' point of view) violated after the evolution. Data point 73 presents a scenario where two models show different views on the same entity with a "correspondence"-like relationship between them. The models show clear matches for each element before the evolution, which are (from the authors' point of view) violated after the evolution. Data point 74 presents the reverse scenario of data point 73.

Table 5 presents the results for the linked cases. The models in data points 71 and 72 are rated consistent by a majority before the evolution. After the evolution, the cases were still rated as *consistent*, but by a smaller majority. The evolution itself was rated as consistency-decreasing by most participants, but not by a majority. For case 73, the rating is less pronounced. The consistency is assessed as rather high but not as changing. In case 74, the consistency is assessed as low before the evolution and higher after the evolution. Notably, the evolution rating does not match the decrease in the Boolean rating; e.g., for data point 71, four participants moved from consistent to inconsistent, but seven participants rated the evolution as consistency-decreasing. The remaining data points show similar, non-matching ratings of the evolution operation. This supports the previously made conclusions about the gradual nature of consistency assessments.

Conclusion: Although a model evolution is considered consistency-decreasing or -increasing, the assessment of the models before and after the evolution may remain the same. Introducing a few contradictions in a consistent set of models does not necessarily make the set of models appear inconsistent and vice versa.

6. Threats to Validity

In the following, we discuss the most relevant factors that limit the validity of our results.

Number of Data Points & Number of Ratings The dataset comprises 62 data points. We received 452 ratings. This leads to an average of 7.3 ratings per data point. We acknowledge that this is a low number of ratings per data point. However, we also need to acknowledge that the number of ratings we could collect with the available resources (participants, time) is bounded by external factors. We deliberately chose a rather high number of data points. By doing so, although we limit the statistical expressiveness per data point, we reduce bias caused by too few data points. Looking at the collected results, we do not expect more ratings to reduce the already large standard deviations; instead, they will likely get even bigger. We concluded that no shared mental model of consistency exists among the group of participants. We do not expect that more ratings would change this conclusion.

Limitations of the Data Points The CoModE dataset v.1 contains only small consistency problems. The used modeling language and visualization technique may impact the understanding of the problems. The sample size and generalization problems are well-known issues in empirical research. However, for our research objective, we argue that these issues have a low impact on the validity of our findings and conclusions. From the beginning, we restrict our investigation to the abstract, yet very restricted, case of purely compositional models. The evaluation criteria (Cx) are formulated positively in terms of a shared mental model of consistency – which, if it exists, would require generalization to other modeling scenarios. However, even in our very specific and restricted case, we do not find evidence of a shared mental model of consistency among the participants. Thus, the limitations of the data set (size and language) do not impact the validity of our conclusions.

Validity of the Students' Ratings It is possible that participants from the student group rated cases randomly or misunderstood the models due to limited knowledge on the topic. This threat cannot generally be fully mitigated in an anonymous, unsupervised questionnaire study. However, we took several countermeasures to reduce this threat. First, by conducting the study within the university course on software engineering towards the end of the semester, we ensured that the participants had at least heard of the relevant concepts. As the course has no presence requirement, we expect the participating students to be motivated. Second, each questionnaire started with two questions to verify that the participant understood the presented models. Third, no participant was forced to participate, nor was anyone required to answer all the questions in the handed-out package. We expect participants with less motivation or insufficient understanding to either not participate or drop out early, without producing many misleading ratings.

Biased Expert Ratings We use the ratings of a small group of experts to compare them to the students' ratings. The experts' ratings may be biased by their experience and background. We are confident to say that the selected experts are well-qualified researchers in model-driven software engineering. To reduce the individual bias of single opinions, we conducted the experts' rating as a group discussion. The experts had to reach a consensus for each rating. Naturally, collecting ratings from a larger group of experts would further reduce the bias. However, for our study, we are confident that the method used is sufficient to draw valid conclusions.

Bias in the Authors' Evaluation We, as the research team, might be biased in our discussion of the results. Although most conclusions were drawn from quantitative data, some were based on a qualitative assessment of the results. We, as the authors, can not be fully objective in this assessment. We are transparent about all steps taken in the analysis and provide all data, tools, and scripts for reproducibility. Thus, we enable the reader to draw their own conclusions using the provided supplementary material (7).

7. Summary & Outlook

Based on the findings and conclusions, we can summarize that the terms "consistent" and "inconsistent" are subject to wide interpretation. We conclude that no shared mental model of consistency exists within and among the investigated groups of software engineers (RQA). However, we observe and conclude interesting aspects of the consistency assessments (RQB). A commonality we can state with confidence is the gradual, i.e.,

Table 5 Rating results of the linked scenarios from the students group.

Data Point	71	72	73	74
Boolean Rating Before Evolution (Consistent/Inconsistent)	11/2	13/4	3/2	1/5
Operation (Consistency... Improving/Decreasing/Neutral)	2/7/4	5/7/5	2/2/1	3/3/0
Boolean Rating After Evolution (Consistent/Inconsistent)	7/6	8/9	3/2	3/3
9P Likert AVG (SD) After Evolution	5.31 (1.86)	5.12 (2.61)	7.00 (1.26)	6.00 (2.00)

non-Boolean nature of consistency. Given the opportunity, consistency was rated on a 9-point scale using the full spectrum. Furthermore, we observed that introducing or removing a few contradictions in a model or a set of models does not necessarily change the overall assessment from consistent to inconsistent, or vice versa. Consequently, if consistency is considered a property of a set of models, or even of a system in general, it must be handled as a gradual property. In case a method requires a Boolean consistency assessment, it must be well-communicated where the threshold between "consistent" and "inconsistent" is located. Although we gained insights about "How" consistency is assessed, we cannot determine a shared mental model of "What" consistency is among the participants in our study. Such a shared mental model might just not exist. Therefore, we recommend that tool builders, researchers, and methodologists not assume a shared understanding of consistency among modelers. Any constraint or requirement on consistency must be clearly defined and communicated to enable a common understanding. Our study is only an initial step towards understanding what software engineers perceive as consistent or inconsistent. Future work must extend this initial investigation to experienced groups of engineers in their respective domains. Thus, it may be possible to identify domain-specific shared mental models on consistency.

Supplementary Material

To support the reproducibility of our analysis results and enable the reader to draw their own conclusions, we provide two distinct repositories. First, a GitHub repository containing the maintained version of the CoModE dataset (Kegel & Contributors 2025). Second, a longer-term data archive on ZENODO (Kegel 2026).

For getting insights into the semi-automatic data creation process and the up-to-date dataset in raw format (XML), we refer to the GitHub repository. To access the full supplementary data for this work, we refer to the ZENODO package.

Acknowledgments

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1608 – 501798263.

References

Albers, A., Koziolek, A., Voelk, T., Klippert, M., Pfaff, F., Stolpmann, R., & Schwarz, S. (2024, 06). Identification of inconsistencies in agile cps engineering with formula student..

Bertossi, L. (2006, June). Consistent query answering in databases. *SIGMOD Rec.*, *35*(2), 68–76.

Cannon-Bowers, J. A., Salas, E., & Converse, S. (1993). Shared mental models in expert team decision making. *Individual and group decision making: Current issues*, *221*, 221–46.

Carroll, J. M., & Olson, J. R. (1988). Mental models in human-computer interaction. *Handbook of human-computer interaction*, 45–65.

Ellis, C. A. (1977). Consistency and correctness of duplicate database systems. In *Proceedings of the sixth acm symposium on operating systems principles* (p. 67–84). New York, NY, USA: Association for Computing Machinery.

Färber, H., Pascual, R., Stübinger, T., & Ulbrich, M. (2026). Observable Semantics for Characterising Consistency Between Heterogeneous Models. In D. Bianculli & E. Gómez-Martínez (Eds.), *Software Engineering and Formal Methods.*

Feichtinger, K., Kegel, K., Pascual, R., Aßmann, U., Beckert, B., & Reussner, R. (2024). Towards formalizing and relating different notions of consistency in cyber-physical systems engineering. In *Proceedings of the acm/ieee 27th international conference on model driven engineering languages and systems* (p. 915–919). New York, NY, USA: Association for Computing Machinery.

Feichtinger, K., Meixner, K., Rinker, F., Koren, I., Eichelberger, H., Heinemann, T., ... Schmid, K. (2022). Industry voices on software engineering challenges in cyber-physical production systems engineering. In *2022 ieee 27th international conference on emerging technologies and factory automation.*

Finkelstein, A. (2000). A foolish consistency: Technical challenges in consistency management. In M. Ibrahim, J. Küng, & N. Revell (Eds.), *Database and expert systems applications* (pp. 1–5). Berlin, Heidelberg: Springer Berlin Heidelberg.

Finkelstein, A., Kramer, J., Nuseibeh, B., Finkelstein, L., & Goedicke, M. (1992). Viewpoints: A framework for integrating multiple perspectives in system development. *International Journal of Software Engineering and Knowledge Engineering*, *2*(01), 31–57.

Johnson-Laird, P. N. (2004). The history of mental models. In *Psychology of reasoning* (pp. 189–222). Psychology Press.

Kegel, K. (2026). *Supplementary material: Investigating novice modelers' intuitive consistency notions for the case of compositional models / comode v.1 dataset.* Zenodo. Retrieved from https://doi.org/10.5281/zenodo.19040498 doi: 10.5281/zenodo.19040498

Kegel, K., & Contributors. (2025). *Comode repository on github.* Retrieved from https://github.com/convidev-tud/comode-dataset

Kegel, K., Götz, S., & Aßmann, U. (2025). Branch drift: A visually explainable metric for consistency monitoring in collaborative software development. *IEEE Access*, *13*, 116972-116993.

Klare, H., Kramer, M. E., Langhammer, M., Werle, D., Burger, E., & Reussner, R. (2021). Enabling consistency in view-based system development — the vitruvius approach. *Journal of Systems and Software*, *171*, 110815.

Knapp, A., & Mossakowski, T. (2018). Multi-view consistency in uml: A survey. In *Graph transformation, specifications, and nets: In memory of hartmut ehrig* (pp. 37–60). Springer.

Kosiol, J., Strüber, D., Taentzer, G., & Zschaler, S. (2020). Graph consistency as a graduated property. In F. Gadducci & T. Kehrer (Eds.), *Graph transformation* (pp. 239–256). Cham: Springer International Publishing.

LaToza, T. D., Venolia, G., & DeLine, R. (2006). Maintaining mental models: a study of developer work habits. In *Proceedings of the 28th international conference on software engineering* (p. 492–501). New York, NY, USA: Association for Computing Machinery.

Littman, D. C., Pinto, J., Letovsky, S., & Soloway, E. (1987). Mental models and software maintenance. *Journal of Systems and Software*, *7*(4), 341-355.

Marchezan, L., Kretschmer, R., Assunção, W. K. G., Reder, A., & Egyed, A. (2023). Generating repairs for inconsistent models. *Software and Systems Modeling*, *22*(1), 297–329.

Mens, T., & Van Gorp, P. (2006). A taxonomy of model transformation. *Electronic Notes in Theoretical Computer Science*, *152*, 125-142. (Proceedings of the International Workshop on Graph and Model Transformation (GraMoT 2005))

Nuseibeh, B. (1996). To be and not to be: on managing inconsistency in software development. In *Proceedings of the 8th international workshop on software specification and design* (p. 164-169).

Pascual, R., Beckert, B., Ulbrich, M., Kirsten, M., & Pfeifer, W. (2025). Formal Foundations of Consistency in Model-Driven Development. In T. Margaria & B. Steffen (Eds.), *Isola* (pp. 178–200).

Stevens, P. (2018). Towards sound, optimal, and flexible building from megamodels. In *Proceedings of the 21th acm/ieee international conference on model driven engineering languages and systems* (p. 301–311). New York, NY, USA: Association for Computing Machinery.

Traiger, I. L., Gray, J., Galtieri, C. A., & Lindsay, B. G. (1982, September). Transactions and consistency in distributed database systems. *ACM Trans. Database Syst.*, *7*(3), 323–342.

Usman, M., Nadeem, A., Kim, T.-h., & Cho, E.-s. (2008). A survey of consistency checking techniques for uml models. In *2008 advanced software engineering and its applications* (p. 57-62).

Voelk, T. A., Gesmann, L., Götze, S., Feichtinger, K., Schwarz, S. E., Düser, T., ... Albers, A. (2025). Misunderstand me correctly - comprehensibility in interdisciplinary collaboration. *Proceedings of the Design Society*, *5*, 2441–2450.

Wickens, C. D., Gordon, S. E., Liu, Y., & Lee, J. (2004). *An introduction to human factors engineering* (Vol. 2). Pearson Prentice Hall Upper Saddle River, NJ.

Yu, X., & Petter, S. (2014). Understanding agile software development practices using shared mental models theory. *Information and Software Technology*, *56*(8), 911-921.

About the authors

Karl Kegel is a Research Associate and Doctoral Candidate at Dresden University of Technology (Germany), Chair for Software Technology. His research interests include software evolution and software quality assurance, particularly in model-driven software engineering. You can contact the author at karl.kegel@tu-dresden.de.

Kevin Feichtinger is a Postdoctoral Researcher at the Dependability of Software-intensive Systems Group at Karlsruhe Institute of Technology (KIT), Germany. His current research interests include software product lines, configurable systems, variability modeling, model transformations, software evolution, software consistency, and software development for Cyber-Physical Systems. You can contact the author at kevin.feichtinger@kit.edu.

Terru Stübinger is a PhD Candidate at the Application-oriented Formal Verification Group at Karlsruhe Institute of Technology (KIT). You can contact the author at stuebinger@kit.edu.

Romain Pascual is an Assistant Professor at CentraleSupèlec Université Paris-Saclay, France. He is a member of the ARCADE team of the MICS laboratory. His research focuses on formal methods and software engineering, with particular interests in model consistency, model-driven engineering, and graph transformations You can contact the author at romain.pascual@centralesupelec.fr.

Bernhard Beckert is full professor for Application-oriented Formal Verification and Head of Division II at Karlsruhe Institute of Technology (KIT), as well as director at FZI Research Center for Information Technology. His research focuses on the practical application of formal, logic-based methods for the specification, verification, and analysis of software to increase the reliability and security of critical systems. You can contact the author at beckert@kit.edu.

Ralf Reussner is full professor for the Dependability of Software-intensive Systems at Karlsruhe Institute of Technology (KIT) and director at FZI Research Center for Information Technology. He has developed the Palladio Software Architecture Simulator and currently serves as the spokesperson of the Collaborative Research Center 1608 „Convide – Consistency in the View-based Development of Cyber-Physical Systems“. You can contact the author at ralf.reussner@kit.edu.

Uwe Aßmann has been a Professor of software engineering with the Faculty of Computer Science, TU Dresden, since 2004, and was the Dean of the Faculty, from 2016 to 2021. He is well known for his work in metamodeling, adaptive petri nets, and context-role-oriented modeling. Furthermore, his research interests include model-driven software engineering, in particular for embedded systems and collaborative robotics.You can contact the author at uwe.assmann@tu-dresden.de.