Journal of Universal Computer Science, vol. 27, no. 1 (2024), 2987-2989
submitted: 1/1/2024, accepted: 1/1/2024, appeared: 1/1/2024 CC BY-ND 4.0

An Empirical Study on the Correctness and Effort to Integrate Feature Models

Vinicius Bischoff
(PPGCA, University of Vale do Rio dos Sinos, São Leopoldo, RS, Brasil
viniciusbischof@unisinos.br)

Kleinner Farias
(PPGCA, University of Vale do Rio dos Sinos, São Leopoldo, RS, Brazil
https://orcid.org/0000-0003-1891-3580, kleinnerfarias@unisinos.br)

Abstract: Feature model integration is pivotal in software development, particularly in evolving software product lines through new feature accommodations. Despite its significance, the influence of developers' experience on integration efforts, and correctness remains inadequately understood. This study conducted a controlled experiment with 25 participants (15 students, 7 professionals) adhering to established empirical study guidelines. Each participant addressed 10 experimental tasks, encompassing 250 integration scenarios, aimed at exploring two core research inquiries. Effort and correctness rate in integrating feature models were quantified, revealing that students exerted higher effort (29.23%) and achieved a greater number of correct integrations (39.53%) compared to professionals. Notably, this superiority lacked statistical significance. Additionally, the study underscores practical implications and noteworthy challenges for the scientific community. The findings lay a foundation for subsequent studies, delving into software development tasks where students and professionals may achieve comparable results. Ultimately, this study marks an initial stride towards an ambitious agenda, empirically advancing feature model integration methodologies.

Keywords: Feature model, model integration, empirical study, controlled experiment, software modeling, integration effort
Categories: H.3.1, H.3.2, H.3.3, H.3.7, H.5.1
DOI: 10.3897/jucs.<SubmissionNumber>

1 Introduction

The integration of software models plays an essential role in various tasks performed by software developers throughout the software product development cycle [Bischoff et al. 2019]. For example, developers use model integration strategies to reconcile software models developed concurrently, mainly to accommodate new features into evolving software systems [Abouzahra et al. 2020]. In the context of collaborative software development, for example, geographically distributed teams can work simultaneously on specific parts of a feature model relevant to them [Abouzahra et al. 2020]. This allows developers to focus more precisely on parts of the model that need adaptations and improvements, usually required by customers to adapt the system in the face of changes in business rules. However, at some point, the changes made in parallel need to be integrated to generate a consolidated view of the model.

Faced with the difficulty of integrating software design models, the academy has proposed some model integration strategies in the last decade [Bischoff et al. 2018, Bécan et al. 2015, Acher et al. 2013, Andersen et al. 2012, Acher et al. 2009]. Both commercial and academic model integration tools suffer from the composition conflict problem [Bischoff et al. 2019]. That is, the to-be-integrated models may conflict with each other and these conflicts must be resolved manually. In practice, detecting and resolving conflicts in feature models is still a highly-intensive manual task [Sharbaf et al. 2022, Abouzahra et al. 2020, Farias et al. 2015]. This means that model integration remains a manual and error-prone task [Bischoff et al. 2018].

The *integration of feature models* can be defined, broadly speaking, as a set of steps that must be performed over two input models, the base model (M_A) and delta model (M_B), to produce an output intended model, M_{AB}. Figure 1 exhibits an illustrative schema of generic model integration. The base model receives the changes in the delta model to transform into an intended model. The changes to be accommodated in the base model can be expressed for example through change requests. The problem is that the base model often becomes a composed model with problems. In this sense, an extra effort must be spent to solve the problems, generating the intended model. Developers can integrate the input models in different ways manually using different model integration strategies. Unfortunately, the output composed model (M_{CM}) and the output intended model (M_{AB}) are often different ($M_{CM} \neq M_{AB}$). The input models tend to conflict with each other in some way due to changes done in parallel [Sharbaf et al. 2022]. Thus, developers need to invest some effort to detect and resolve these conflicts.

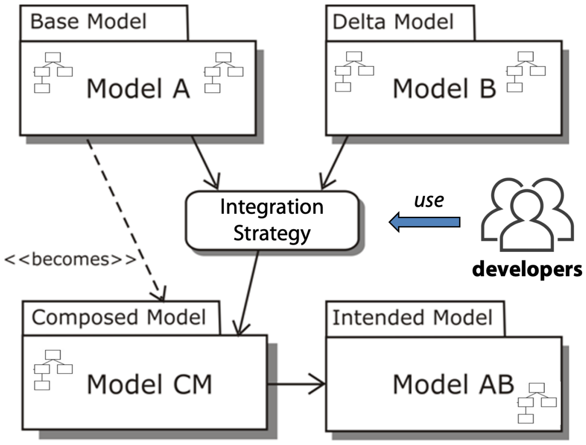


Figure 1: A schematic illustration of a generic model integration.

When several developers work in parallel on the same model, conflicts end up being inevitable [Sharbaf et al. 2022]. The research problem of integrating feature models is about combining multiple feature models into a single, unified model, ensuring that they fit together seamlessly and avoiding any conflicts or inconsistencies. To understand this, suppose we have different feature models that represent different aspects or variations of a product. For instance, think of a smartphone with separate feature models for the camera, operating system, and storage options. Each feature model describes the available choices and relationships for that particular aspect. The integration challenge arises when we create a comprehensive feature model that incorporates all the options from the individual models. This merging process involves bringing together the various feature model elements and their choices to form a cohesive and complete representation.

Although feature models have been used in the literature [Bischoff et al. 2019], little has been done to empirically analyze the effort that developers need to invest to perform integrations. Moreover, the literature does not explore the influence of the experience of developers [Filippo et al. 2010] in the effort and correctness of the integrations carried out. Previous studies [Filippo et al. 2010] have investigated the influence of experience on comprehension tasks supported by UML stereotypes. Studies [Sharbaf and Zamani 2020, Mahmood et al. 2020, Abouzahra et al. 2020] indicate that model merging is still an open problem. In addition, evidence from the industry suggests that model integration and conflict resolution end up being full-time job [Farias et al. 2015].

The feature model consists of a high-level model that has been widely used to represent the characteristics and possible configurations of software products extracted from software product lines. A feature of a software system can be seen as a software functionality or expected behavior of a software system. The software industry faces restrictions on the applicability of models in development teams [Bischoff et al. 2019, Asadi et al. 2016], which raises the following question: To what extent were the produced feature models integrated correctly? Determining the efficiency and effectiveness in the quality of the integration process, since the existing conflicts affect the comprehensibility of the models, increasing the risk of delays in software projects due to rework, as well as increasing production costs. These challenges become decisive for improving best practices in software process management.

This study, therefore, performs a controlled experiment with 25 participants (15 students and 7 professionals) following well-known guidelines to run empirical studies [Wohlin et al. 2012]. Each participant answered 10 experimental tasks, representing 250 integration scenarios to explore two research questions. For this, we quantified the effort and correctness rate of integration of feature models realized by our participants. In particular, we investigate the effects of the integration of feature models, concerning the effort and correctness from the perspective of students and professionals in the context of the evolution of feature models. The main research contribution of this article is the empirical knowledge about the impact of the experience of students and professionals on the effort and correctness to integrate feature models. The collected data reveals that experience yielded a reduction in integration effort while showing no significant impact on the correctness of integration. Specifically, students exhibited a higher integration effort (2.21) compared to professionals (1.71), indicating an increase of 29.23%. Additionally, students demonstrated a higher correctness rate (0.60) in contrast to professionals (0.43), signifying a 39.53% increase in correctness.

These findings shed light on the impact of developers' experience on the integration of feature models, emphasizing the varying levels of effort and correctness in different groups. Notably, academic students, despite their lower experience, demonstrated efficiency in integration tasks, although not always statistically significant. Professionals, on the other hand, often invested less effort but faced challenges in producing correct integrations. This research brings forth intriguing implications, pointing towards the need for further exploration in software development tasks where students and professionals can achieve similar results. Moreover, our study provides a foundational step towards advancing feature model integration methodologies, highlighting the significance of considering developers' experience in the process. It uncovers the complexities in model integration, and the variations in effort and correctness across different developer groups, contributing to the ongoing discourse in software engineering research.

This article significantly extends our previous work [Bischoff et al. 2018] in several aspects. First, the entire research protocol was revised to make each step of the empirical study more crystal clear, including the description of each step of the experimental process, scientific intuitions for formulating hypotheses, explanation of study variables, review of analysis procedures, detailing of the questionnaire used in the experimental study, testing of hypotheses formulated using statistical methods and detailing implications and research opportunities. In particular, hypothesis testing procedures have been improved by tabulating the data differently and by applying new statistical tests to allow for more careful analysis. Third, we detail the procedures followed to mitigate threats to the validity of the results obtained. In addition, this article draws some intriguing implications and worth-investigating challenges for the next few years by the scientific community (Section 5.3).

The remainder of the paper is organized as follows. Section 2 outlines the main concepts discussed throughout the paper. Section 3 compares this study with others, highlighting their main differences and commonalities. Section 4 describes the adopted study methodology. Section 5 presents the study results and introduces some additional discussions. Section 6 discusses some strategies followed to mitigate threats to the validity of our study. Finally, Section 7 presents some conclusions and future directions.

2 Background

This section introduces the main concepts necessary for understanding the reported study. Section 2.1 describes the concepts of feature modeling through examples and their main purposes. Section 2.2 defines the concept of integration of feature models. Section 2.3 presents a model integration process.

2.1 Feature models

The feature model is considered a high-level model used to express the products of a software product line representing the characteristics of a specific domain, its variability, and similarities, as well as its relationships [Kang et al. 1990]. The main objective of the feature model is to model the common properties and possible product variables of a production line, including their interdependence [Bischoff et al. 2019, Czarnecki et al. 2002]. The features represent the attributes of the application of a given domain, being directly related and visible to the final customer [Kang et al. 1990]. A feature model can be seen as a compact representation of characteristics of a software product, and can also represent a functionality or behavior that a software should have. The features of a software system are organized through a diagram, named *feature diagram*.

Moreover, it is a tree, in which the root represents a concept and its leaves are features connected by edges that represent its state. Its status is displayed using intuitive notations to represent the points of variation. Figure 2 presents a feature model and its notations. The example illustrates the notations typically used to represent the relationships between features, mandatory, optional, exclusive, and inclusive alternative, and transverse relationships, exclusion, and dependency. Note that the dependency and exclusion relationship are cross-tree constraints, different from the other relationship types. The hierarchical relationship is defined between an ancestral feature and its descendant features. A descendant feature can only be part of a product in which its ancestral feature appears:

- **Mandatory**: A child feature, that has a mandatory relationship, is included in all products where the parent feature appears. In the example, given that *A* is root, this makes *B* always present in any product configuration.
- **Optional**: The child feature can have a relationship defined as optional. Thus, it can be included optionally in all products where its main functionality is included. In Figure 2, feature *G* may or may not be included in the software product derived from this feature model.
- **Alternative-Exclusive**: A set of child features is defined as an alternative, when only one feature can be selected, the others being excluded. The parent feature is part of the product. In Figure 2, only feature *E* or *F* can be selected.
- **Alternative-Inclusive**: A set of child features can be added additionally to the products in which the parent feature appears. In the example, features *C* or *D* (or both) can be selected.
- **Dependency**: Selecting a feature also implies selecting another feature.
- **Exclusion**: Selecting a feature prevents you from selecting another feature.

According to the example, the root feature *A* represents a concept or functionality. The features defined below it represent the possibilities of variation existing in this domain. As can be seen, feature *B* is mandatory, i.e., implying that it is necessary to define a feature *B*. There are features *C* and *D* below *B*. As they are inclusive alternative features, the selection of both features may occur. However, the exclusive alternative features *E* and *F*, when selected, imply the exclusion of another. As an example of an optional feature, we have feature *G*, in addition, the feature-oriented domain analysis notation [Kang et al. 1990], allows the use of dependency and exclusion between features.

2.2 Integration of feature models

The integration of feature models can be briefly defined as the set of activities that need to be performed to produce a properly integrated model [Farias 2012]. These activities are carried out on two input models (FM_A and FM_B), aiming to produce an intended or desired model (FM_{AB}) that includes requests for evolution or changes. However, the desired model is not always produced, generating an integrated model with a problem (FM_I). Efforts must be made to identify and resolve such problems to produce the desired model. Typically, the production of the intended model is not so obvious due to the presence of conflicting elements of FM_A and FM_B. We use the terms integrated model (FM_I) and intended model (FM_{AB}) to differentiate between the output model produced with problems and the model desired. As previously mentioned, usually FM_I and FM_{AB} do not match because the input models conflict with each other in some way [Sharbaf et al. 2022]. The higher the number of inconsistencies in FM_I, the more distant it is from FM_{AB}. This may mean, for example, a high effort to be spent to derive FM_{AB} from FM_I (or not).

Figure 3 presents an example of the integration of two feature models. The first model (FM_A) is the base feature model. The second model (FM_B) is the delta model that represents the changes that should be inserted into the base model to transform it into an output intended model (FM_{AB}). The FM_{AB} has all the desired features of a particular software system. If we want to derive a software product from FM_A, this

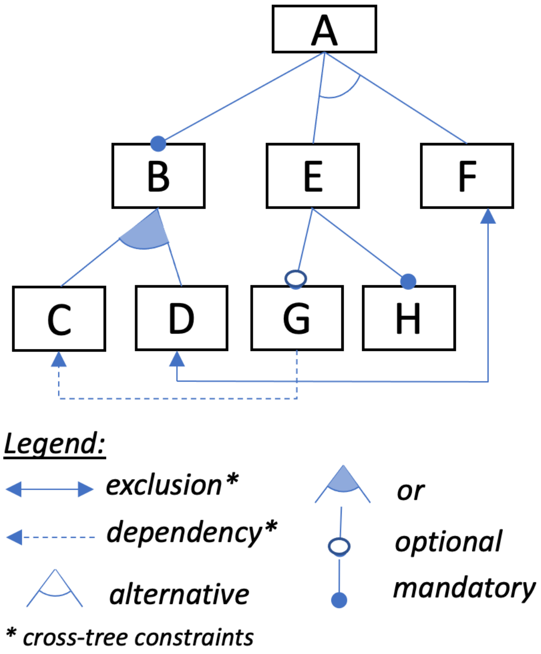


Figure 2: Example of a feature model.

product should have the feature A and optionally the feature B. However, in the FM_B all software products have features A and C. FM_A and FM_B are integrated using the following algorithm called "merge strategy": For all corresponding elements in FM_A and FM_B, the elements should be combined. The combination depends on the element type. Elements in FM_A and FM_B that are not equivalent remain unchanged and are inserted into the output model directly. But, the desired model FM_{AB} was not produced.

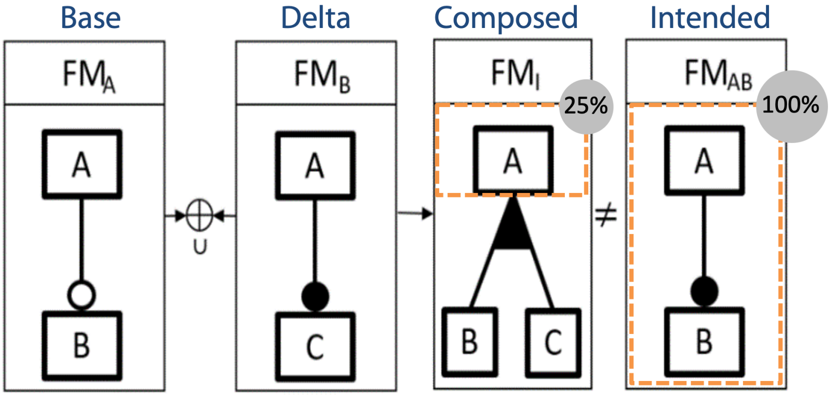


Figure 3: Example of integration of feature models.

Empirical studies [Farias et al. 2015, Farias et al. 2014] have revealed that the integration of design models remains a highly intensive manual task because the model elements to be composed usually conflict with each other in some way and such conflicts should be properly solved. Otherwise, inconsistencies are inserted into the output integrated model. Figure 3 illustrates two conflicts. Conflict is a contradicting value assigned to the properties of the FM_A and FM_B. The first conflict is that we have one feature named *B* in FM_A, while we have the other feature named *C* in FM_B. The second conflict is that the relationship between features *A* and *B* in FM_A is optional, while the relationship between the features *A* and *C* in the FM_B is mandatory.

With these conflicts at hand, software developers need to invest the effort to detect and resolve these conflicts. However, usually, these conflicts are not correctly understood and properly solved in real-world settings. In part, this difficulty in resolving could be explained by the lack of information about project decisions made at the time conflicts arise. Therefore, resolving conflicts becomes a challenging task [Abouzahra et al. 2020].

Consequently, instead of producing an output-intended model, as would be expected, the integrations end up producing an output-composed model with inconsistencies. Thus, developers need to invest some extra effort to detect and resolve the created inconsistencies. In this case, the composed and intended models are inconsistent. Inconsistencies are contradicting values between the output-composed model and the output-intended model. In this case, we have two inconsistencies: the first one would be that features *B* and *C* were inserted into the output-composed model FM_I, rather than just the feature B as would be expected in the intended model FM_{AB}. The second inconsistency is that an alternative relationship between features *A*, *B* and *C* was created, rather than a mandatory relationship between the features *A* and *B* (as expected in FM_{AB}). Therefore, the output-composed model has just 25% of the output-intended model.

2.3 Model integration process

To facilitate comprehension of feature model integration, Figure 4 presents a four-step generic process for model integration. This process has been documented in a prior work [Farias et al. 2018]. The integration steps are outlined as follows:

- **Phase 1: Analysis of the Input Models.** During the analysis step, techniques are employed to examine and verify the feature input models. One common aspect checked is whether the feature models are represented using the same notation. For example, if both feature models are expressed using the Feature-Oriented Domain Analysis (FODA) notation. This verification ensures consistency in the representation of feature models, facilitating the subsequent integration process. This phase is necessary because the tools can integrate models represented in different notations. If the models are represented using the same notation, then the next step is to compare them

- **Phase 2: Model Comparison.** In the comparison step, the integration technique identifies and establishes equivalences between the elements of the input feature models. These elements can include features, constraints, relationships, and dependencies present in the models. By identifying corresponding elements in both models (FM_A and (FM_B)), the integration process can align and integrate them effectively.

- **Phase 3: Integration.** The integration step involves the actual merging and combination of similar elements identified in the previous step. Elements from the input feature models that have been deemed equivalent are integrated into a single composed feature model (FM_{CM}). However, this integration process may introduce conflicts or inconsistencies due to differences in the original models. These conflicts need to be addressed to ensure that the resulting composed feature model aligns with the desired feature model (FM_{AB}).

- **Phase 4: Evaluation.** In this step, the integrated feature model (FM_{CM}) is assessed to determine its compliance with well-formed rules and constraints. This evaluation helps ensure that the composed feature model is structurally sound and satisfies the predefined requirements and constraints. If the integration operators used in the previous steps offer guarantees by construction, this evaluation step may not be necessary. However, in cases where such guarantees are not provided, the evaluation process becomes crucial to identify any deviations from the desired feature model (FM_{AB}).

Overall, these four steps — analysis, comparison, integration, and evaluation — constitute a systematic approach to integrating feature models. Each step plays a vital role in ensuring the compatibility, consistency, and correctness of the integrated feature model, ultimately leading to the generation of the desired feature model with minimal inconsistencies and effort.

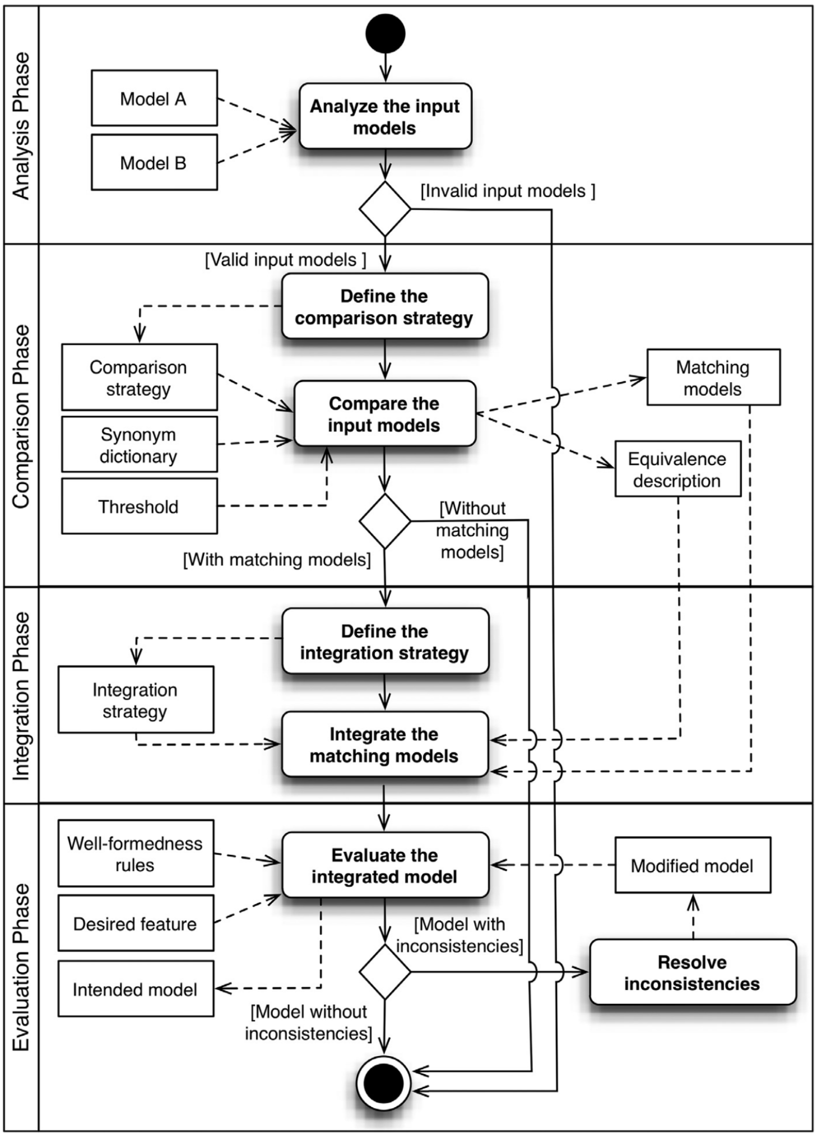


Figure 4: A general purpose model integration process (from [Farias et al. 2018])

3 Related work

This section compares the reported empirical study with the literature. For this, Section 3.1 analyses all the studies that we were aware of during the selection process. Section 3.2 introduces a comparative analysis of the related works, highlighting their commonalities and differences.

3.1 Analysis of related works

We explored the literature to find works close to ours, considering the empirical nature and configuration of our study. In total, five articles were surveyed for convenience, using the ACM Digital Library[1], IEEE Xplore[2] and Google Scholar[3]. We considered three selection criteria when using the ACM, IEEE, and Google Scholar databases to search for our related studies. First, they have been widely used in systematic reviews of the literature, demonstrating a rigorous basis for studies in this line of research [Bischoff et al. 2019, Menzen et al. 2021]. Another important criterion was the ease of use. That is, the authors are largely familiar with such search engines. Third, both databases are continuously updated containing relevant articles in this field of research involving the integration of software models. By considering these selection criteria, we can enhance the relevance, quality, and timeliness of the articles we can find in the ACM, IEEE, and Google Scholar databases, helping us to retrieve valuable scientific literature for our research.

Farias *et al.* (2015) [Farias et al. 2015]. This study reports that one of the main limitations for the adoption of model composition techniques based on both specifications (e.g., Epsilon) and heuristics (e.g., IBM RSA) is the lack of knowledge about their effects on developers' effort. To mitigate this lack of knowledge, the article presents a controlled experiment to investigate the effort applied in different model composition techniques and detect and resolve inconsistencies in the output-composed models. The techniques were evaluated with 144 evolution scenarios, producing 2,304 compositions of UML models. Participants had professional experience. The findings suggest that techniques based on heuristics require less effort than techniques based on a specification to produce the intended models, in addition, there is no significant difference in the correction of composite models and the use of manual heuristics outperforms automated counterparts in the composition of models.

Farias *et al.* (2014) [Farias et al. 2014]. This study analyzes the lack of information on indicators that help developers identify models resulting from composition heuristics, with a high probability of presenting inconsistencies and understanding which composed models (UML models) need more effort to be investigated through an exploratory study. This study evaluates stability as an indicator of inconsistency rate and resolution effort in model composition activities, through 180 compositions carried out to develop design models for three product lines. The results of this exploratory study indicate that stable models are a good indicator of composition inconsistency and resolution effort. Although the study brings interesting results, the article does not report a comparative analysis between professionals and students considering the feature model integration effort. Considering the empirical strategy used, the study applied a quasi-experiment and not a controlled experiment

Asadi (2016) [Asadi et al. 2016]. The present article introduces a series of visualization and interaction interventions designed to represent and configure feature models. These interventions are empirically evaluated through a controlled experiment to assess their impact on the configuration process for application engineers. The research aims to examine the extent to which visualization and interaction interventions enhance the configuration process. To achieve this, a controlled empirical study was conducted at Simon Fraser University in Canada, involving 20 participants who were students. It

[1] https://dl.acm.org/
[2] https://ieeexplore.ieee.org/Xplore/home.jsp
[3] https://scholar.google.com/

should be noted that the study did not account for the effort expended by professionals and students in integrating feature models. Moreover, the study did not control for the participants' profiles, such as distinguishing between professionals and students. The empirical study revealed significant improvements in completion time for both comprehending and modifying the feature model configuration due to the employed visualization and interaction interventions. Furthermore, the participants reported that the proposed interventions were user-friendly and easy to learn. The primary contribution of the study lies in identifying a set of visualization and interaction interventions that positively affect comprehension and modification tasks associated with the configuration of feature models. This set of interventions serves as a foundation for further exploration of visualization and interaction techniques in the configuration process of software product lines. Although the study presents intriguing findings, it is important to note the limitations. The study does not incorporate a controlled investigation that considers the integration effort invested by professionals and students. Additionally, the study fails to address the distinction between professional and student participants, which could have potential implications for the generalizability of the results.

Perez *et al.* (2020) [Perez et al. 2020]. This study highlights the importance of maintenance activities embedded in Feature Location information in software artifacts. The study activities are carried out manually or automated, seeking to facilitate the maintenance tasks and evolution of typical engineering software related to features. For example, when modifying and removing features in a product line, the work does not specify the integration between Feature Location. This process consumes large amounts of time and effort, without guaranteeing good results from the development teams. The article proposes to compare manual and automated FL in a group of 18 people (5 specialists and 13 non-specialists), i.e., professionals and graduate students in an industrial domain. In addition, they seek to evaluate the productivity, performance, and ease of use of both treatments in a controlled environment. The study does not evaluate the correctness rate among graduate students and professionals, as well as partially traces the implications of both treatments. Finally, the authors provide some research opportunities to improve the results of manual and automated Feature Location techniques. Although interesting findings are reported, it fails to perform a comparative analysis between professionals and students considering the effort to integrate feature models and the correctness of the integrations.

Bürdek *et al.* (2016) [Burdek et al. 2016]. This work describes the evolution of the Continuous Feature Models (FM) to meet the software requirements of the product line. The evolution of the product line leads to changes in FM. As a result, product line engineers often face problems. For example, a high rate of cohesion between the features and their semantic representation (optional, mandatory, alternative, Or), which requires great effort on the part of the team. In this context, the work presents a formal approach to compare two incoming feature models, through a case study in conjunction with experimental data. However, the work does not portray how the empirical study was conducted. That is, it is understood that this is not an experiment carried out in a controlled environment. Furthermore, the authors do not present an assessment of the proposed approach between professionals (with experience) and students (without experience). Just as they do not measure inconsistencies and the effort required to use the approach. Finally, the study presents implications and future research opportunities that aim to assist the scientific community and the industry in conducting the integration and comparison of FMs.

3.2 Comparative analysis of the works

This section contrasts the surveyed works with our work. This comparison, based on comparison criteria (C), serves to identify some similarities and differences. The comparison criteria are presented below:

1. **Main contribution (C1)**: Studies that report their obtained contributions or findings from experimental studies performed using feature models.
2. **Experimental study (C2)**: Studies that are a controlled experiment.
3. **Context (C3)**: Studies that were performed in a controlled environment in academia.
4. **Participant profile (C4)**: Studies that consider students and industry professionals.
5. **Study variables (C5)**: Studies that consider the effort spent by professionals and students, as well as the correctness of the output composed models as study variables.
6. **Implications and Research Opportunities (C6)**: Studies that outline implications and highlight research opportunities.
7. **Use of feature model (C7)**: Studies that carry out investigations on using feature models.
8. **Comparative analysis between professionals and students (C8)**: The study performs a comparative analysis between professionals and students considering the effort to integrate feature models and the correctness of the integrations.

Table 1 presents the comparison considering these criteria. The Related Work column indicates the source of the research being analyzed. It lists various research studies, including Farias et al. (2015), Farias et al. (2014), Asadi (2016), Perez et al. (2020), and Bürdek et al. (2016). The Comparison Criteria column outlines the criteria. It highlights variations in how well these studies meet the specified criteria. That is, it reveals that some studies perform well in certain criteria while falling short in others, indicating areas of strength and weakness in each work. This analysis can guide researchers and practitioners in understanding the limitations and strengths of these studies. For instance, some studies fail to fully meet fundamental criteria, such as ”Main contribution” or ”Experimental study,” suggesting a need for more comprehensive and rigorously designed research in these areas. The absence of comparative analysis between professionals and students in several studies highlights a potential research opportunity to investigate the impact of experience on feature model integration more comprehensively.

We emphasize that the proposed empirical study was the one that most met the criteria (C1-8), highlighting its contributions and limitations. Therefore, we see a research opportunity to carry out an empirical study that contemplates all the comparison criteria. We perform a controlled experiment to generate empirical knowledge and insights, as well as report some intriguing implications and challenges that can be explored by the scientific community in the coming years.

Related Work	Comparison Criteria							
	C1	C2	C3	C4	C5	C6	C7	C8
Proposed Empirical Study	●	●	●	●	●	●	●	●
Farias *et al*. (2015) [Farias et al. 2015]	●	●	●	●	◐	●	○	○
Farias *et al*. (2014) [Farias et al. 2014]	●	○	○	○	●	●	○	○
Asadi (2016) [Asadi et al. 2016]	●	●	●	○	○	●	●	○
Perez (2020) *et al*. [Perez et al. 2020]	◐	●	●	●	○	◐	◐	○
Bürdek *et al*. (2016) [Burdek et al. 2016]	●	○	○	○	○	●	●	○

Legend:

● Meets Fully, ○ Does not meet

◐ Meets partially, ⊘ Not Applicable

Table 1: A comparative analysis of the related works.

4 Study Methodology

This section describes the methodology used in our study. Section 4.1 describes our objective and research questions. Section 4.2 introduces the formulated hypotheses. Section 4.3 explains the study variables. Section 4.4 deals with the context and subject selection. Section 4.5 presents the adopted experimental process, experiment design, and the material used. Section 4.6 describes the analysis procedures. Section 4.7 explains the questionnaire used in our study. All these methodological steps were followed based on well-established practical guidelines about empirical studies presented in [Wohlin et al. 2012].

4.1 Objective and research questions

Our study seeks to evaluate the effects of experience on two variables: effort and correctness. These effects were investigated through evolution scenarios of feature models. In this sense, we use the GQM template [Wohlin et al. 2012] to state the objective of this evaluation, as follows:

Analyze *the experience of students and professionals*
for the purpose of *investigating their effects*
with respect to *effort and correctness*
from the perspective of *the execution of feature model integration tasks*
in the context of *evolving feature models.*

Little is currently known if professionals tend to invest less effort to integrate feature models, at least while generating correctly integrated models, when compared to students, for example. Suppose a student has a success rate in an integration activity close to that of a professional. So, it makes no sense to allocate a professional if a student can achieve a similar result, requiring a lower cost. Our investigation follows this line of reasoning, seeking to understand if there is a significant difference in the results obtained by students and professionals, in terms of the effort of integration and correctness. Thus, we focus on exploring two Research Questions (RQ), as follows:

- **RQ1:** What is the effect of the experience on the integration effort?

- **RQ2:** What is the effect of the experience on the correctness of the integration?

4.2 Hypothesis formulation

To answer RQ1, we analyze one research hypothesis that investigates the impact of the experience on the effort invested to integrate feature models.

First hypothesis (H1). The integration of feature models requires the manipulation of conflicting models, which requires certain skills (e.g., knowledge of the integration process, change request comprehension, analytical thinking, and decision-making) to properly circumvent the situation. If conflicting changes are inadequately resolved, then models with inconsistencies are typically generated, affecting the syntactic and semantic properties in the model. A consequence of this would be the production of an integrated model that does not match the desired or intended model. Perhaps experience can be a decisive factor in circumventing conflict resolution, while it will allow you to find a coherent solution based on previous experiences. However, this is not yet evident in the context of integrating feature models. If the effort invested by more experienced people is high, then the allocation of experienced people to perform the integration of feature models becomes questionable. Perhaps the simplicity of the feature models favors people with little experience, investing an effort similar to the more experienced ones. Based on this claim, we formulate our first hypothesis.

Null Hypothesis 1, ($H1_{null}$): There is no difference in the means of the integration effort ($\mathbb{IE}$) invested by students and professionals to combine feature models.
$H1_{null}$: $\mathbb{IE}(FM_A, FM_B\ )_{Prof} = \mathbb{IE}(FM_A, FM_B\ )_{Stud}$
Alternative Hypothesis 1, ($H1_{alt}$): There is a statistically significant difference in the means of the integration effort ($\mathbb{IE}$) invested by students and professionals to combine feature models.
$H1_{alt}$: $\mathbb{IE}(FM_A, FM_B\ )_{Prof} \neq \mathbb{IE}(FM_A, FM_B\ )_{Stud}$

Second hypothesis (H2). The second hypothesis seeks to investigate whether experience influences the integration of feature models by providing a greater number of correct integrations. In this sense, we conjecture that more experienced professionals can produce models correctly integrated in a larger number when compared to students. This conjecture may also not be confirmed. Perhaps students perform the integration with more caution, favoring the integration of models, especially with small models. If professionals generate integrated models with a correctness rate similar to that generated by students, then it makes more sense to allocate them to more complex activities, where experience is a prerequisite.

Null Hypothesis 2, ($H2_{null}$): There is no difference in the means of the correctness rate ($\mathbb{CR}$) produced by students and professionals when integrating feature models.
$H2_{null}$: $\mathbb{CR}(FM_A, FM_B\ )_{Prof} = \mathbb{CR}(FM_A, FM_B\ )_{Stud}$
Alternative Hypothesis 2, ($H2_{alt}$): There is a statistically significant difference in the means of the correctness rate ($\mathbb{CR}$) produced by students and professionals when integrating feature models.
$H2_{alt}$: $\mathbb{CR}(FM_A, FM_B\ )_{Prof} \neq \mathbb{CR}(FM_A, FM_B\ )_{Stud}$

4.3 Study variables

Table 2 presents the study variables. The independent variable of the study is the experience of the participants, which is classified as nominal, assuming two possible values: *Student* (Stud) or *Professional* (Prof). The participants were classified into two groups. Those studying in technical courses or university were considered as students. Those who exercised a professional activity were considered professionals, highlighting that all professionals had graduated.

The dependent variables were two: integration effort ($\mathbb{IE}$) and correctness rate ($\mathbb{CR}$). The first refers to the time invested by the participants to perform integrations, assuming values from 0 to 60. Each participant had to answer 10 questions. Thus, if a participant had an effort of 15 minutes, then he/she needs, on average, 1.5 minutes to answer each question. The second variable quantifies the rate of the correct answer, representing the choice of a correct integration. If the participant chooses the alternative that has the integrated model correctly, then the answer is correct. The variable calculates the rate of correct answers per question. For example, if 3 out of 10 answers are correct, then the correctness rate for the question was 0.3.

Variable	**Name**	**Scale**
Independent	Main	Nominal: {Student, Professional}
Dependent	Integration Effort ($\mathbb{IE}$)	Interval [0..60]
Dependent	Correctness Rate ($\mathbb{CR}$)	Interval [0..1]

Table 2: Study variables.

4.4 Context and subject selection

The students were invited to participate in the experiment so that we could have subjects with different backgrounds and levels of expertise. The professionals held a Master'degree and Bachelor'degree (or equivalent) and knew software modeling and programming. The professionals were from companies located in southern Brazil, while the students were from a postgraduate program in Applied Computing at the University of Vale do Rio do Sinos in Brazil. The graduate students attended one course with the following theme: Software Engineering. The experiment was part of the postgraduate course (at Unisinos) and was performed as a laboratory exercise. The authors trained participants so that everyone had a minimum level of knowledge about feature models and integration tasks.

The participants performed 10 experimental tasks related to the integration of feature models and were familiar neither with these tasks nor with the design models. The tasks represented cases where the participants were not the initial designers of the feature models. The models used in our study were based on different application domains, including financial and health care. Each experimental task contains an Evaluation Scenario (ES) in which two feature models (FM_A and FM_B) should be integrated. The experiment questionnaire presented 5 answer options, with the participant choosing only one option, which would represent the desired integration of the feature models.

4.5 Experimental process, design and material

Figure 5 shows the experimental process followed to perform the empirical study, which is organized into 3 phases. Each phase is described as follows:

- **Phase 1: Training and Exercises.** All participants were trained to ensure that they acquired the necessary familiarity with integration strategies, i.e., possible ways to perform integrations. Note that no particular integration techniques were used in our study. In this sense, the training focused on demonstrating integrations similar to those present in the experimental tasks. As we are investigating the influence of experience on feature model integrations, then it would not make sense to force participants to use a particular technique. Rather, we allowed everyone to develop their integration strategy based on their own experience.

  The participants had no previous experience with feature modeling. It was necessary to carry out careful training so that everyone could acquire knowledge concerning feature modeling. This learning curve was overcome and verified through exercises. That is, participants performed feature modeling exercises to ensure and demonstrate their knowledge about variability modeling. In this phase, we explain the entire experimental process, the integration techniques, the notations used for feature models (their annotations and relationship configuration), the step-by-step process to perform the integrations, the procedures and materials used in the experiment, including the questionnaire and how to record the time.
- **Phase 2: Integration of Feature Models.** The second step is to analyze the FM_A and FM_B input models of each scenario based on the descriptions of change requests in each question, which define how the elements of FM_A were changed. Participants must identify the conflicts and mentally try to resolve them. The instructions on how the integrations must be carried out are at the top of the questionnaire. Each participant must consider these instructions to integrate the feature models, which consists of choosing an alternative in the questionnaire used in the study (Figure 6). That is, based on the instructions and the input models (FM_A and FM_B), each participant should produce a new model as output, FM_{AB}. The measurement of the integration effort (time in minutes) is collected during this activity. Each participant records the integrated feature model according to the questionnaire shown in Figure 5.
- **Phase 3: Questionnaire.** This phase aimed to conduct a questionnaire about the profile of the participants. Each participant fills out a questionnaire, which allows them to collect information about their professional experience, academic background, modeling and development experience, gender, and age.

The experimental design of this study is characterized as a no repeated measure between-subjects design [Wohlin et al. 2012]. The models used in this experiment were feature diagrams with about 10 features, 7 relationships, 3 depth levels, and 3 conflicts, on average, by feature model. We chose to use small models for two reasons: (1) large models would require the need to control the size variable, something outside the scope of this study; and (2) controlled experiments should not use artifacts that make the activities tiring and extensive by bringing unnecessary content to the study in question.

4.6 Analysis procedures

Quantitative analysis. After collecting the data, the first step was to make a descriptive analysis to understand the distribution of the collected data. In this sense, descriptive

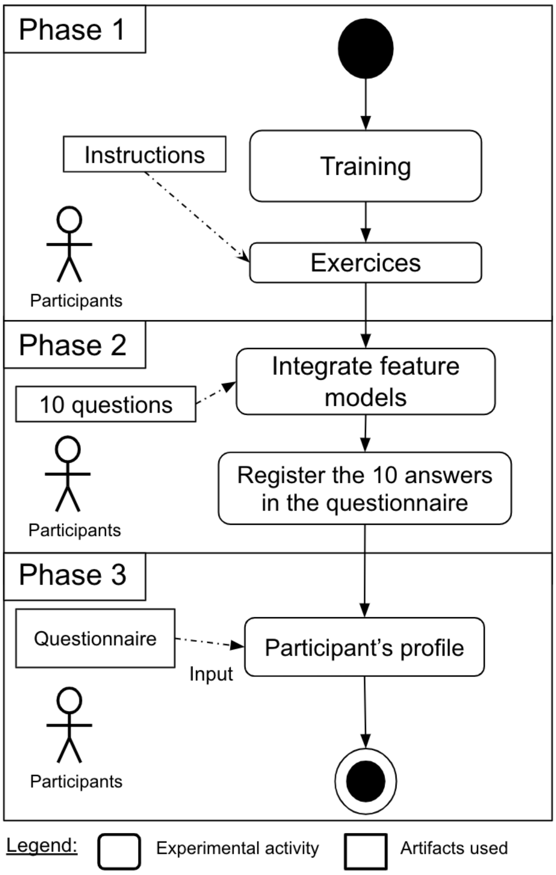


Figure 5: Experimental process followed in our study.

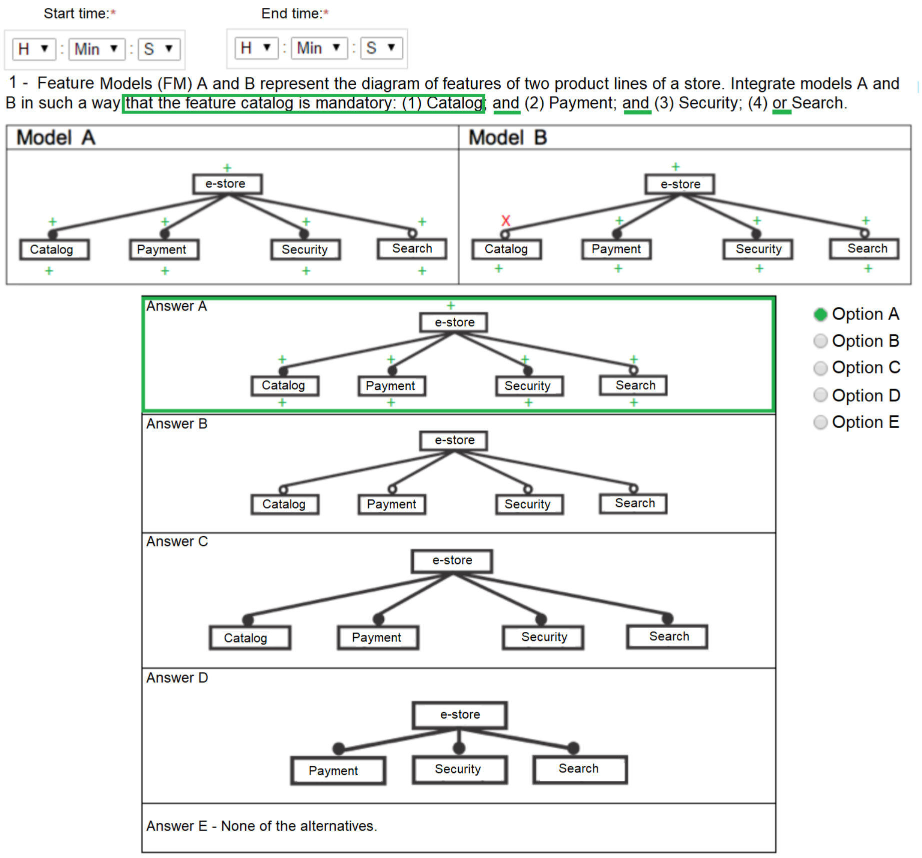
statistics were produced to analyze the normal distribution [Wohlin et al. 2012]. The analysis of the normal distribution is essential when defining which statistical methods are adequate to test the formulated hypotheses. The Kolmogorov-Smirnov test was applied to pinpoint deviations from normality. We use statistical inference methods to test the formulated hypotheses. The level of significance of the hypothesis tests was $\propto$ = 0.05. To test the first and second hypotheses, we applied the independent group t-test for the ten tasks. This test is similar to the Mann-Whitney but requires two separate sets of independent and normally distributed samples. Note that we have a no repeated measure between-subjects design.

4.7 Questionnaire

The questionnaire has ten questions applied to the integration of feature models. Questions 01-06 and Question 10 present two feature models as input (FM_A and FM_B), which after completing their composition return a new model, FM_I.

Questions 07, 08, and 09 present only one feature model that supports changes in its relationship (exclusions and dependencies) between features. They seek to know how these dependencies or exclusions affect the feature model and what the perception of analysts and developers will be. The models proposed for features occur from the derivation of a product line, which is derived from the literature (car, cell phone, and the sales portal of a store). These are the models from which the participants idealized their integration, according to the requirements established in each of the questions. Each question has five alternatives, and the participant can choose a single alternative, filling in the starting and ending times of each question, thus obtaining the time to perform each task.

Figure 6 presents one of the questions. We emphasize that feature model A concerning feature model B, presents a semantic non-conformity, which refers to the relationship (mandatory/optional). In every question in our empirical study, the participants analyze two input feature models (FM_A and FM_B) and then choose an answer. After the execution of the tasks by the participants, the data undergoes an analysis, to calculate the integration effort and quantify the correctness rate.



Start time:* H ▾ : Min ▾ : S ▾

End time:* H ▾ : Min ▾ : S ▾

1 - Feature Models (FM) A and B represent the diagram of features of two product lines of a store. Integrate models A and B in such a way that the feature catalog is mandatory: (1) Catalog; and (2) Payment; and (3) Security; (4) or Search.

Answer E - None of the alternatives.

● Option A
○ Option B
○ Option C
○ Option D
○ Option E

Figure 6: First question used in our empirical evaluation.

5 Study Results

This section analyzes the data set obtained from the empirical study. We test the formulated hypotheses by applying statistical tests using the RStudio tool[4]. Section 5.1 discusses the obtained results related to the first hypothesis. Section 5.2 presents the collected data related to the second hypothesis (H2).

5.1 Integration effort and experience

Descriptive statistics. This section discusses the data collected regarding the impact of the experience of the participants on the integration effort. To do this, we compute descriptive statistics to understand the distribution of the obtained data, including its main trends and dispersion. In this sense, descriptive statistics are carefully computed as grasping the data distribution and the main trends is essential. Not only the main trend was calculated using the two most used statistics to discover trends (mean and median), but the dispersion of the data around them was also computed through the standard deviation. Table 3 exhibits the collected data related to the integration effort. Figure 7 shows the boxplot of the integration effort. Note that these statistics are calculated based on a number (N) of 10 questions, 70 questions realized by professionals and 180 by students. The normality test of Kolmogorov-Smirnov indicates that the data are normally distributed. After analyzing these statistics, we realized that the experience had little impact on the integration effort. The main finding is that the integration effort invested by students and professionals was similar. This result is supported by some observations.

Variable	**Group**	**N**	**Min**	**25th**	**Mean**	**Med**	**75th**	**Max**	**SD**	**%**
$\mathbb{IE}$	Student	10	1.37	1.72	2.19	2.21	2.60	2.95	0.47	+29.23%
$\mathbb{IE}$	Professional	10	1	1.45	1.91	1.71	2.41	3.4	0.67	

Legend: minimum (Min), median (Med), maximum (Max), standard deviation (SD), #questions (N), first quartile (25th), third quartile (75th), integration effort ($\mathbb{IE}$)
%: difference between median

Table 3: The descriptive statistics of the integration effort.

First, the mean of the integration effort invested by the students is slightly higher than the effort invested by the professionals. That is, the integration effort for students (2.19) was higher than that for professionals (1.91), representing an increase of 14.65%. The median presented similar results. The integration effort for students (2.21) was higher than that for professionals (1.71), representing an increase of 29.23%. This means that students and professionals have invested a similar effort to answer the formulated questions. One implication would be that if a feature model integration activity needs to be done, then it would be indifferent in terms of effort whether it will be carried out by students or professionals. Another finding is that the effort in both cases tends to be close to the central tendency, with a standard deviation equal to 0.47 and 0.67, instead of spreading out over a large range of values. Moreover, analyzing and possibly removing outliers from the data is essential to draw valid conclusions from the collected

[4] https://www.rstudio.com

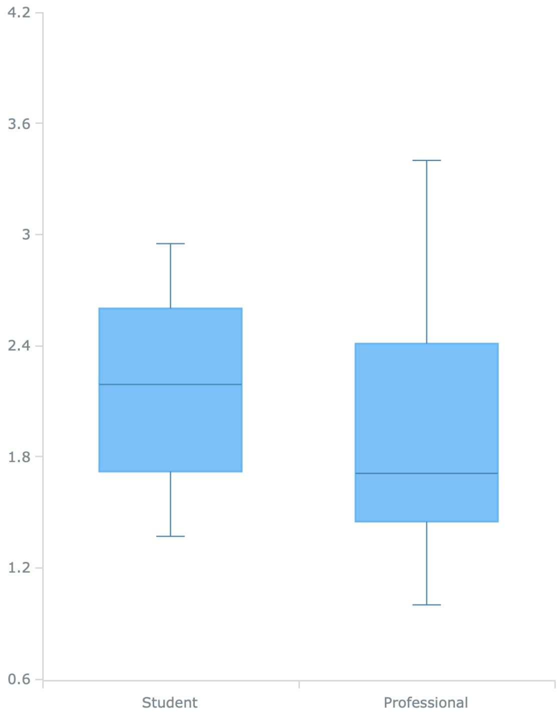


Figure 7: Box-plot of the integration effort

data. Outliers are extreme values that may influence the conclusions of our conclusions. Outliers were not found in our study.

Hypothesis test (H1). We performed a statistical test to evaluate whether the difference between the integration effort produced by students and professionals (although small) is statistically significant. As we hypothesize that the integration effort tends to be different, the mean difference test will be performed as a two-tailed test, considering a significance level at 0.05 (p $\leq$ 0.05) to indicate a true significance. As the collected data did not violate the assumption of normality, the parametric, independent group t-test was used to test the first hypothesis. Table 4 presents the obtained results. We can see that the group means are not statistically significantly different because the value in the p-value row is higher than 0.05. Therefore, the null hypothesis ($H1_{null}$) that advocates that the effort invested by students and professionals is equal cannot be rejected. That is, there is no statistical significance to affirm that students invested more or less effort than professionals when integrating feature models. While the findings of our study are intriguing, it is important to note that they cannot be readily generalized to other contexts, domains, and samples.

Variable	Mean Difference	S.E. Difference	t	DF	p-value
Effort	-0.275	0.275	-1	18	0.331
Correctness	-0.132	0.098	-1.35	18	0.195

Legend:
SE: Standard Error, DF: Degree of Freedom

Table 4: The results of the hypothesis tests.

Summary for the integration effort: *The collected data indicate that the experience did not favor the reduction of the integration effort. An independent group t-test indicated that t= -1 with 18 degrees of freedom, and p-value = 0.331. This implies that professionals did not invest statistically significantly lower effort (1.91 ± 0.71 min) to executing integration tasks compared to students (2.19 ± 0.49 min). The means of the two groups were not significantly different. Therefore, failing to reject the first null hypothesis.*

5.2 Correctness and integration techniques

Descriptive statistics. This section analyzes the obtained data regarding the impact of experience on the correctness rate. Again, we calculate descriptive statistics to reveal the data distribution, including its main trends and dispersion, as previously done. Table 5 shows the correctness of the integrations produced by students and professionals. Figure 8 shows the box plot of the correctness rate. As previously mentioned, these results are computed considering the number (N) of 10 questions, 70 questions realized by professionals and 180 by students. We performed the Kolmogorov-Smirnov test to examine the normality of the collected data. The normality test suggests that the data are normally distributed.

The main finding is that although the students had less experience, they obtained better results. This can be explained for some reasons. First, on average, the correctness rate produced by the students is slightly higher than the rate generated by the professionals. That is, the correctness rate for students (0.56) was higher than that for professionals (0.43), representing an increase of 30%. The median also favored the students, presenting an even better value than the average. That is, the median of the correctness rate for students (0.6) was higher than that for professionals (0.43), representing an increase of 39.53%. The standard deviation also showed a behavior similar to that presented in the effort variable, showing a tendency towards centralization instead of dispersion. The standard deviation of the correctness rate is close to zero for students (0.2) and professionals (0.21). No outlier was identified in our study. Perhaps due to the simplicity of the feature model and the greater attention invested in the experiment, students were able to obtain better results. On the other hand, it may have happened that professionals, seeing the simplicity of the models, neglected the execution, not investing due attention. This result brings an interesting aspect by not following the popular wisdom that more experienced people tend to always get better results. Although the students presented a more favorable result, it is still not possible to say whether this gain had statistical significance or not. Thus, the next step was to investigate whether this result was statistically significant, thus testing our second hypothesis.

Variable	Group	N	Min	25th	Mean	Med	75th	Max	SD	%
$\mathbb{CR}$	Student	10	0.23	0.6	0.56	0.60	0.70	0.9	0.2	+39.53%
$\mathbb{CR}$	Professional	10	0.14	0.26	0.43	0.43	0.6	0.86	0.21	

Legend: minimum (Min), median (Med), maximum (Max), standard deviation (SD), #questions (N), first quartile (25th), third quartile (75th), correctness rate ($\mathbb{CR}$) %: difference between median

Table 5: The descriptive statistics of the correctness rate.

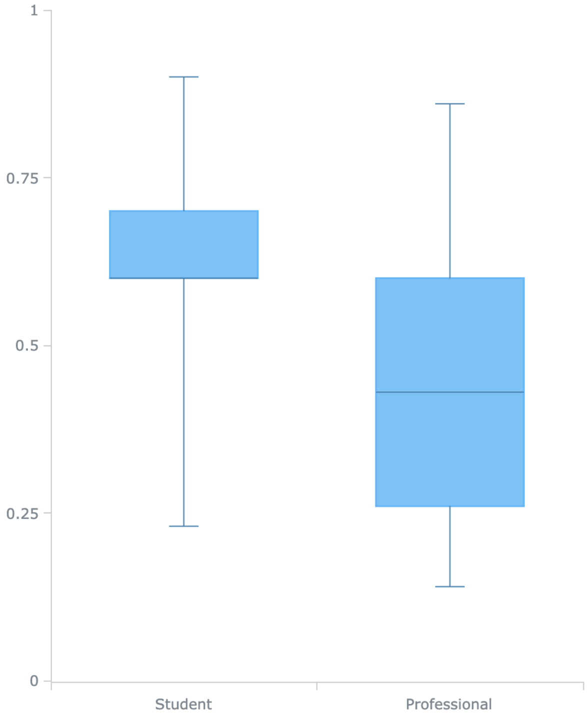


Figure 8: Box-plot of the correctness.

Hypothesis test (H2). As our data is normally distributed, the independent group t-test was applied to check whether the perceived difference between the correctness rate values produced by students and professionals is statistically significant. The t-test was performed as two-tailed and with a significance level of 0.05 (p $\leq$ 0.05) to indicate a true significance. Table 4 exhibits the results considering the hypothesis testing. As the p-value row is higher than 0.05, the correctness rate obtained by students and professionals is not statistically significantly different. That is, there is no significant difference between means. Therefore, our second null hypothesis ($H2_{null}$), claiming the correctness of integrations would be equal, cannot be rejected. Our data suggests that there is no statistical significance to conclude that when integrating feature models, professionals will produce a significantly larger number of correctly integrated models. Although our results are interesting they cannot be generalized to other contexts, domains, and samples.

Summary for the correctness rate: *The obtained data suggest that the level of experience did not favor the production of correctly integrated feature models. An independent group t-test indicated that t= -1 with 18 degrees of freedom, and p-value = 0.195. This means that professionals did not produce a statistically significantly higher correctness rate (0.435 ± 0.225) compared to students (0.567 ± 0.212). The means of the two groups were not significantly different. Therefore, failing to reject the second null hypothesis.*

5.3 Additional discussion

After producing and explaining the produced empirical knowledge, the next step is to draw some additional discussions. For this, we also consider our experience acquired in previous experimental studies on the integration of software design models [Graeff et al. 2023, Carbonera et al. 2023, Farias et al. 2019, Manica et al. 2018, Farias et al. 2013, Farias et al. 2018, Farias et al. 2012].

5.3.1 Empirical study, replications and human factors

A quality model for integration of feature models. Some quality models for design modeling have been proposed in the last decades [Lange 2007]. However, these quality models aim at software modeling in general rather than the integration of feature models itself. Further studies might extend these quality models for attending to quality issues in feature modeling. This extension might be based on practical knowledge derived from developers' experience in integrating UML diagrams in practice and from researchers' knowledge in conducting empirical studies, including controlled experiments, industrial case studies, quasi-experiments, interviews, and observational studies. This evidence-based quality model might guide developers and researchers about how to plan and run empirical studies addressing integration issues. The coming guidance might have a unifying terminology for activities and artifacts related to integration tasks and a systematic relation between quality notions and metrics for qualitative and quantitative assessment. These terminologies and relations might help to identify and empirically evaluate possible factors or indicators of effort, accuracy, granularity, and scalability of integration of feature models. For instance, a quality model might help developers select metrics and procedures to evaluate how integration-confusing factors — i.e., the level of abstraction, domain issues, and type of techniques — would affect the precision and accuracy of the current approaches. Moreover, a quality model might also serve as a reference frame to structure empirical studies performed by other researchers in the future. Without this reference frame, the replication and contrast of empirical studies as well as the generalization of their results get impaired.

Insights and practical knowledge on the model comparison effort. Based on the quality model, researchers might investigate the side-effects of integration confusing factors on the developers' effort. Some influential factors might be considered in this investigation, such as the type of integration techniques and the decomposition mechanism used to structure feature models. Moreover, researchers might also explore to what extent the rationale behind design decisions during the modeling process of feature models could influence the matching relations between the elements of feature models.

Empirical findings might also enhance the knowledge about the impact of such factors on the developers' effort to apply integration techniques and detect and mitigate improper or counter-intuitive similarities between elements of feature models. A counter-intuitive similarity would be an equivalence between elements of diagrams by a comparison technique that would be contrary to the developer's intuition or conventional wisdom. Additionally, we might bring together insights about how to evaluate the developers' effort, decrease error proneness by realizing integration, and tame the side effects of the influential factors in practice. The current body of knowledge on model comparison cloud be ameliorated by: (1) testing out recurring claims formulated by experts that were never evaluated; and (2) identifying correlations between comparison-influencing factors and variables involved throughout the model integrations. For example, most studies to date fail to analyze which types of differences between feature models make the integration techniques more error-prone by producing counter-intuitive equivalences more frequently; (3) elaborating a deep knowledge to support the formulation of theories on the integration of feature models; (4) providing a solid background to inspire the creation of the next-generation integration techniques and tools; and pinpointing when the model integration techniques work and when they do not work.

More empirical knowledge about integrating feature models. Although we have presented an empirical study, further studies need to be carried out to create a plethora of evidence-based knowledge. For example, it would be interesting to conduct a controlled experiment to assess the impact of integration techniques on the effort invested by developers to produce correct models. Currently, developers can use techniques based on specification and heuristics to perform the integrations. However, little is known about the benefits of these different strategies. Furthermore, no experimental study has yet been carried out to assess the influence of previous experience with feature modeling on the effort to integrate feature models. For example, the scientific community could run a controlled experiment with participants with experience with feature models and participants with no experience with feature models to evaluate how much this experience can influence the results of the integrations.

Impacts of human factors on feature model integration. An intriguing research challenge that the scientific community needs to investigate is the impact of human factors on feature model integration. Specifically, exploring how human cognitive processes, decision-making, and collaboration influence the effectiveness and efficiency of integrating feature models. We have noted that integrating feature models involves complex tasks such as identifying commonalities, resolving conflicts, and managing dependencies. These tasks require human involvement, and understanding the human factors at play can provide valuable insights into improving the integration process. For example, research could explore how different cognitive biases or heuristics affect decision-making during feature model integration. Investigating how cognitive biases, such as confirmation bias or anchoring bias, may influence the identification and resolution of conflicts can shed light on potential pitfalls and suggest strategies to mitigate their impact.

Collaborative and integration. Furthermore, examining the collaborative aspects of feature model integration is essential. Collaborative integration involves multiple stakeholders with diverse perspectives, expertise, and decision-making styles. Understanding how communication, coordination, and negotiation among stakeholders impact the integration process can lead to the development of effective collaborative techniques and tools. In addition, human factors such as individual expertise, experience, and training could also be investigated to determine their influence on the efficiency and accuracy of feature model integration. Exploring how different levels of expertise or training impact integration outcomes can inform the development of training programs or guidelines to enhance integration performance. Addressing the research challenge of human factors in feature model integration requires interdisciplinary collaboration between software engineering, cognitive psychology, and human-computer interaction fields. By investigating these human factors, the scientific community can advance our understanding of how to optimize feature model integration processes and develop more efficient and user-centered integration techniques and tools.

5.3.2 Flexible technique, AI tools, and user experience issues

Flexible technique to identify similarity. Little has been done in academia to develop flexible techniques to support different strategies for identifying similarities between feature models. In addition, it is essential to develop a technique for identifying similarity between features and their relationships, which allows the application of multiple strategies of comparison, identification, and validation of similarity based on syntactic and semantic characteristics, aided by the operation of relational logic to detect and solve inconsistencies. With the application of these techniques, we will be able to seek to improve the efficiency of the algorithm, as well as perform the comparison and identification of similarity between the features and their relationships, validating the applied feature model.

Technique for integrating feature models. Recent literature reviews indicate that FODA notation [Bischoff et al. 2019] is the most used notation to represent feature models. However, the current literature has not given due attention to the production of effective techniques for the integration of models represented in this notation. The academic community would benefit from the development of a technique for the integration of feature models. It would be very useful to have techniques that support different operations or integration methods, including joining, intersection, and difference, in a semi-automatic or automatic way.

Machine learning, gamification, and AI-assisted integration approach. Future research might investigate the application of machine learning models to predict integration outcomes and, in turn, suggest conflict resolution strategies, thus minimizing the effort that software developers should invest to produce a desired output model. We emphasize that the key motivation for using a particular model integration technique is to reduce the developers' effort to produce the desired output model. If an integration technique reduces the effort to produce an output model — however increases the effort to detect and resolve the inconsistencies (or vice versa) — then it is quite arguable whether it can be applied in mainstream software projects. This is particularly due to its detrimental impact on the overall efforts expended by developers. This was, that future AI-assisted integration approaches might explore how artificial intelligence can aid in automating integration tasks and suggest (optimal) integration strategies based on historical data. Software merging remains a difficult and highly error-prone task [Seibt et al. 2021, Vale et al. 2021] and this reality is no different in the context of software model integration [Sharbaf et al. 2022]. For this, we also suggest (1) investigating strategies for leveraging the expertise of developers, including students and professionals, to enhance integration outcomes, and (2) discussing the concept of gamification to make integration tasks more engaging and improve the quality of integration.

Easy-to-use integration tool. Recent research [Farias et al. 2014, Farias et al. 2015] has pointed out that the current model integration tools tend to require a lot of effort from users. This turns out to be counterproductive, making the integration task costly and error-prone. Easy-to-use integration tools could reduce the effort by making the model integration process more intuitive, for example, by clearly showing similarity relationships, the well-formed rules being challenged, the overlap between the model elements, the impact of integration on quality attributes of the models, the indication of conflicts between parts of the models, and strategic information for resolving conflicting changes.

5.3.3 Implications for software product line engineering

Resource allocation and expertise management. Our study suggests that leveraging a mix of developers, including students, in integration processes could be a strategic approach. Students showcased a significant level of proficiency, implying that resource allocation should not be solely based on experience but should consider individual capabilities. Software product line engineering could benefit by strategically distributing integration tasks based on expertise and skill sets. Thus, software product line managers can tailor their resource allocation strategies, ensuring that tasks are assigned based on competency rather than experience alone. Moreover, software development teams could also benefit from a balanced mix of developers, optimizing efficiency and outcome quality in integration efforts.

Efficiency and cost-effectiveness. The produced empirical knowledge indicates that students, despite investing more effort, achieved a greater number of correct integrations. This finding implies that integrating feature models efficiently does not necessarily entail increased costs. It suggests a potential avenue for cost-effectively achieving integration goals, making integration processes more efficient without significantly escalating expenses. In this way, organizations, mainly distributed teams, can design their integration strategies, optimizing for cost-effectiveness by incorporating methodologies that allow for higher effort but yield a larger number of correct integrations. This approach can lead to streamlined integration processes within software product lines.

Knowledge transfer and training strategies. Our results highlight that students, despite their limited professional experience, demonstrated notable competency. This can indicate the value of effective knowledge transfer and training strategies. Tailored training programs focusing on integration methodologies can enhance the proficiency of developers, ensuring that they contribute effectively to integration tasks. We believe that software development teams can design and implement targeted training programs, enhancing the integration skills of their members. By investing in training initiatives that focus on integration best practices, companies (and their teams) can also elevate the competency of their developers and improve overall integration outcomes.

5.3.4 Take-home messages regarding feature model integration

Experience and integration outcome. The lack of statistical significance in the superiority of students over professionals highlights that experience alone does not guarantee better integration outcomes. Integration success is influenced by multiple factors beyond experience, such as problem-solving skills, adaptability, and approach to integration. Therefore, software development teams should recognize that while experience is valuable, other traits and skills also play a significant role in achieving successful integration. It necessitates a holistic evaluation of a developer's capabilities beyond mere experience.

Need for in-depth exploration. Our study emphasizes the need for a deeper exploration of software development tasks, aiming to identify scenarios where both students and professionals can achieve comparable results. Understanding the specific tasks where students excel can provide insights into optimizing task assignments. For example, organizations can optimize the recommendation of integration task — typically characterized as a difficult, time-consuming, and often error-prone task [Gustavo et al. 2021]. This way, further research is warranted to pinpoint the specific integration tasks where students demonstrate prowess. This knowledge can guide software engineering teams in task allocation and project planning to maximize efficiency and productivity across diverse skill levels.

Commencement of an ambitious research agenda. We can see our initial results as a starting point for a more ambitious research agenda aimed at empirically advancing feature model integration methodologies. These results, together with the knowledge acquired from previous studies, may serve as valuable insights for developers to save time and avoid conflict problems or even as guidelines for tool builders to better support practitioners. Moreover, our results generate opportunities for researchers to improve the state of the art of integration of feature models in the context of software product line engineering. This way, this study establishes a foundational empirical framework for subsequent, more extensive research into integration dynamics. Additionally, it encourages researchers to investigate the intricate aspects of feature model integration and prompts the development of novel methodologies and approaches in this domain.

6 Threats to validity

This study may have some threats to validity concerning statistical conclusion validity, construct, internal, and external threats. In this sense, we discuss some strategies used to mitigate these threats.

Statistical conclusion validity. We checked whether the independent and dependent variables were properly submitted to statistical methods. We analyzed whether the presumed cause and effect covary and how strongly they covary [Wohlin et al. 2012]. We studied the normal distribution of the collected sample, seeking to minimize the threats to the causal relation between the research variables. Thus, we verified which parametric or non-parametric statistical methods might be used. We applied the Kolmogorov–Smirnov test to check the normal distribution of the data. Since the assumptions of the statistical test were not violated, we are confident that the test statistics were chosen properly. Moreover, concerning statistical significance we tested all hypotheses considering the significance level at 0.05 level ($p \leqslant 0.05$).

Construct validity. Our main concern was checking if we were measuring what we thought we were measuring. By doing so, we had a certain concern about checking whether (or not) the quantification methods of the dependent variables were carefully defined, and the measures were accurately registered. The form of quantifying the dependent study variables is widely accepted in the literature, being its quantification method reused from previous work [Farias et al. 2015, Farias et al. 2019]. In addition, the experimental design used is well-documented in the literature and the experimental process is close to the previous empirical study already published [Farias et al. 2012]. Therefore, we believe that the construction of our study is reliable.

Internal validity. A causal relation involving the independent and dependent variables needs to be valid. In this sense, we sought to check that the questionnaire response preceded the effort invested and the assertiveness of the answers, thus assuring the temporal precedence criterion. Additionally, we also observed the co-variation of the measures of the variables, i.e., the level of experience led to varying the integration effort and correctness. Still, we did not observe a clear cause for the detected co-variation among study participants. Our previous experience running empirical studies [Farias 2012, Farias et al. 2013, Farias et al. 2014, Farias et al. 2015] helped us to minimize the chances of the dependent variables being affected by other existing variables, other than the level of experience. Although this is a difficult activity to guarantee, we try to do our best. In this sense, we believe that the internal validity has been carefully managed.

External validity. To what extent are the findings of this study applicable in other contexts? In this sense, the findings reported here may be considered more widely, if the context of their use is close to the configuration of the study presented in Section 4. For example, the participants need to realize integrations through questionnaires. This reality shows a not-very practical perspective of our study. Despite this, our evaluated hypotheses can show that for certain activities, manipulating simple artifacts, professionals and students can obtain similar results.

7 Conclusions and Future Work

The integration of feature models plays a key role in many software engineering activities, for example, by developing SPL design models to add new features and reconcile conflicting models developed in parallel. Many integration techniques have been proposed to support the integration of these FMs. However, we identified a lack in the literature on empirical studies on the integration of feature models. This article, therefore, reported on a controlled experiment that evaluated the effects of experience on the integration effort and the correctness of the integrations.

Our initial hypotheses were that the professionals perform the tasks with less effort and produce a higher rate of correctness than their counterparts. In total, 25 participants quantified 250 integrations to test two formulated hypotheses. Our findings indicate that the experience of students and professionals provided a difference in the effort invested and in the correct responses. Despite this, this difference was not statistically significant. Thus, we concluded that students and professionals end up having similar results when integrating simple feature models.

In future work, we intend to replicate this study with a larger number of participants and control the level of complexity of the experimental tasks. Finally, the issues outlined throughout the study can encourage other researchers to replicate our study in the future under different circumstances. We see our study as a first step in a more ambitious agenda on better supporting the integration tasks of feature models.

References

[Abouzahra et al. 2020] Abouzahra, A., Sabraoui, A., Afdel, K.: “Model Composition in Model Driven Engineering: A systematic literature review”; Information and Software Technology, 2020.

[Andersen et al. 2012] Andersen, N., Czarnecki, K., She, S., and Wasowski, A.: “Efficient synthesis of feature models”; 16th International Software Product Line Conference, 2012, Volume 1, pp. 106–115.

[Acher et al. 2009] Acher, M., Collet, P., Lahire, P., France, R.: “Composing feature models”; International Conference on Software Language Engineering, pp. 62–81, 2009

[Acher et al. 2013] Acher, M., Combemale, B., Collet, P., Barais, O., Lahire, P., France, R. B.: “Composing your compositions of variability models”; International Conference on Model Driven Engineering Languages and Systems, pp. 352–369, 2013.

[Asadi et al. 2016] Asadi, M., Soltani, S., Gasevic, D., Hatala, M.: “The effects of visualization and interaction techniques on feature model configuration”; Empirical Software Engineering, 2016, Vol 21, pp. 1706–1743.

[Bécan et al. 2015] Bécan, G., Behjati, R., Gotlieb, A., Acher, M.: "Synthesis of attributed feature models from product descriptions"; 19th International Conference on Software Product Line, pp.1–10, 2015.

[Bischoff et al. 2018] Bischoff, V., Farias, K., Gonçales, L.: "Evaluating the Effort of Integrating Feature Models: A Controlled Experiment"; 30th International Conference on Software Engineering and Knowledge Engineering, California, USA, July, 2018, pp. 326–325.

[Bischoff et al. 2019] Bischoff, V., Farias, K., Goncales, L., Barbosa, J.: "Integration of feature models: A systematic mapping study"; Information and Software Technology, 2019, Vol 105, pp. 209-225.

[Burdek et al. 2016] Bürdek, J. et al.: "Reasoning about product-line evolution using complex feature model differences"; Automated Software Engineering, 2016, Vol. 23, pp. 687-733.

[Carbonera et al. 2023] Carbonera et al.: "Software Merge: A Two-Decade Systematic Mapping Study"; XXXVII Brazilian Symposium on Software Engineering, 2023, pp. 99-108.

[Czarnecki et al. 2002] Czarnecki, K., Østerbye, K., Völter, M.: "Generative programming"; European Conference on Object-Oriented Programming, 2002, June Springer, Berlin, Heidelberg, pp. 15-29.

[Farias et al. 2015] Farias, K. et al.: "Evaluating the Effort of Composing Design Models: A Controlled Experiment"; Software and Systems Modeling, 2015, Vol. 14, No. 4, pp. 1349–1365.

[Farias et al. 2013] Farias, K. et al.: "Analyzing the effort of composing design models of large-scale software in industrial case studies"; Proc. International Conference on Model Driven Engineering Languages and Systems, Miami, USA, September 2013, pp. 639-655.

[Farias et al. 2012] Farias, K.: "Empirical evaluation of effort on composing design models"; Proc. ACM/IEEE 32nd International Conference on Software Engineering, Cape Town, South Africa, May 2010, Vol. 2, pp. 405-408.

[Farias et al. 2019] Farias, K. et. al.: "UML2Merge: a UML extension for model merging"; IET Software, 2019, 13(6), pp. 575-586.

[Farias et al. 2018] Farias, K. et al.: "Toward an Architecture for Model Composition Techniques."; Proc. 27th International Conference on Software Engineering and Knowledge Engineering, Pittsburgh, USA, July 2018, pp. 656-659.

[Farias et al. 2014] Farias, K., Garcia, A., Lucena, C.: "Effects of stability on model composition effort: an exploratory study"; Software and Systems Modeling, 2014, Vol 13, No. 4, pp. 1473-1494.

[Farias 2012] Farias, K.: "Empirical Evaluation of Effort on Composing Design Models"; Ph.D. Thesis, 2012, Department of Informatics, PUC-Rio, Rio de Janeiro, RJ, Brazil.

[Farias et al. 2012] Farias, K., Garcia, A., Lucena, C.: "Evaluating the impact of aspects on inconsistency detection effort: a controlled experiment"; Proc. International Conference on Model Driven Engineering Languages and Systems, Innsbruck, Austria, September 2012, pp. 219-234.

[Filippo et al. 2010] Filippo et al.: "How developers' experience and ability influence web application comprehension tasks supported by UML stereotypes: A series of four experiments"; IEEE Transactions on Software Engineering, 2010, vol. 36, no. 1, January/February.

[Gustavo et al. 2021] Gustavo et al.: "Challenges of resolving merge conflicts: A mining and survey study"; IEEE Transactions on Software Engineering, 2021, vol. 48, no. 12, pp. 4964-4985.

[Graeff et al. 2023] Graeff et al.: "On the Prediction of Software Merge Conflicts: A Systematic Review and Meta-analysis"; XIX Brazilian Symposium on Information Systems, 2023, pp. 404-411.

[Menzen et al. 2021] Menzen, J., Farias, K., Bischoff, B.: "Using biometric data in software engineering: a systematic mapping study"; Behaviour and Information Technology, Vol. 40, No. 9, pp. 880-902, 2021.

[Kang et al. 1990] Kang, K. et al.: "Feature-oriented domain analysis (FODA) feasibility study"; No. CMU/SEI-90-TR-21. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 1990.

[Lange 2007] Lange, C.: "Assessing and Improving the Quality of Modeling A Series of Empirical Studies about the UML"; Ph.D. Thesis, Technische Universiteit Eindhoven, Eindhoven, 2007.

[Manica et al. 2018] Manica, M. et al.: "Effects of Model Composition Techniques on Effort and Affective States: A Controlled Experiment (S)"; Proc. 30th International Conference on Software Engineering and Knowledge Engineering, Redwood City, USA, January 2018, pp. 304-303.

[Mahmood et al. 2020] Mahmood, W. et al.: "Causes of merge conflicts: a case study of Elastic-Search"; 14th International Working Conference on Variability Modelling of Software-Intensive Systems, February, pp. 1-9, 2020.

[Perez et al. 2020] Perez, F., Echeverria, J., Lapena, R., Cetina, C.: "Comparing manual and automated feature location in conceptual models: A Controlled experiment"; Information and Software Technology, 2020, Vol. 125, pp. 106-337.

[Seibt et al. 2021] Seibt, G., Heck, F., Cavalcanti, G., Borba, P., Apel, S.: "Leveraging structure in software merge: An empirical study"; 2021, IEEE Transactions on Software Engineering, 48(11), pp. 4590-4610.

[Sharbaf and Zamani 2020] Sharbaf, M., Zamani, B.: "Configurable three–way model merging"; 2020, Software: Practice and Experience.

[Sharbaf et al. 2022] Sharbaf, M., Zamani, B., and Sunyé, G.: "Conflict management techniques for model merging: a systematic mapping review"; 2022, Software and Systems Modeling, pp. 1-49.

[Vale et al. 2021] Vale, G., Hunsen, C., Figueiredo, E., Apel, S.: "Challenges of Resolving Merge Conflicts: A Mining and Survey Study"; 2021, IEEE Transactions on Software Engineering, 48(12), pp. 4964-4985.

[Weber et al. 2016] Weber, V. et al.: "Detecting inconsistencies in multi-view UML models"; International Journal of Computer Science and Software Engineering, 2016, Vol. 5, No. 12, pp. 260-264.

[Wohlin et al. 2012] Wohlin, C. et al.: "Experimentation in Software Engineering"; Springer, Heidelberg, Germany, 2012.