This version is a pre-print. Please refear to the publisher (Springer) to access the post-edition paper version here: https://doi.org/10.1007/978-3-031-05760-1_27

How to cite: Mosquera, D., Ruiz, M., Pastor, O., Spielberger, J. (2022). Assisted-Modeling Requirements for Model-Driven Development Tools. In: Guizzardi, R., Ralyté, J., Franch, X. (eds) Research Challenges in Information Science. RCIS 2022. Lecture Notes in Business Information Processing, vol 446. Springer, Cham. https://doi.org/10.1007/978-3-031-05760-1_27

Assisted-Modeling Requirements for Model-Driven Development Tools

David Mosquera[1[0000-0002-0552-7878]], Marcela Ruiz[1[0000-0002-0592-1779]], Oscar Pastor[2[0000-0002-1320-8471]], Jürgen Spielberger[1[0000-0003-2617-3535]]

[1] Zürich University of Applied Sciences, Gertrudstrasse 15, Winterthur 8400, Switzerland
{mosq, ruiz, spij}@zhaw.ch

[2] PROS-VRAIN: Valencian Research Institute for Artificial Intelligence - Universitat Politècnica de València, València, Spain
opastor@dsic.upv.es

Abstract. Model-driven development (MDD) tools allow software development teams to increase productivity and decrease software time-to-market. Although several MDD tools have been proposed, they are not commonly adopted by software development practitioners. Some authors have noted MDD tools are poorly adopted due to a lack of user assistance during modeling-related tasks. This has led model-driven engineers—i.e., engineers who create MDD tools—to equip MDD tools with intelligent assistants, wizards for creating models, consistency checkers, and other modeling assistants to address such assist-modeling-related issues. However, is this the way MDD users expect to be assisted during modeling in MDD tools? Therefore, we plan and conduct two focus groups with MDD users. We extract data around three main research questions: i) what are the challenges perceived by MDD users during modeling for later code generation? ii) what are the features of the current modeling assistants that users like/dislike? and iii) what are the user's needs that are not yet satisfied by the current modeling assistants? As a result, we gather requirements from the MDD users' perspective on how they would like to be assisted while using MDD tools. We propose an emerging framework for assisting MDD users during modeling based on such requirements. In addition, we outline future challenges and research efforts for next-generation MDD tools.

Keywords: Model-driven development, Focus group method, Framework, Assisted-modeling, Modeling assistants.

1 Introduction

Model-driven development tools (MDD) allow software development teams to increase productivity and decrease software time-to-market [1]. MDD tools use models for automatically generating software source code. However, MDD tools are rarely adopted by software development practitioners. Some authors show that MDD tools have not yet surpassed the benefits of classical approaches such as the code-centric approach [2]. This has motivated researchers to investigate and establish unaddressed challenges to improve the adoption of MDD tools [3–8]. Significantly, some authors have identified challenges about assisting MDD users during modeling, such as increasing user-centric approaches instead of technology-centric [6], understanding the modeling context [8], improving model management [5], etc.

MDD User: We refer to "MDD user" in this paper to anyone who has had experience with an MDD tool to generate a software artifact.

Identified challenges as [3–8] have motivated model-driven engineers—i.e., engineers who create MDD tools—to equip MDD tools with modeling assistants. However, is this the way MDD users expect to be assisted during modeling in MDD tools? Having this question in mind, we plan and conduct two focus groups based on [9] by following the World Café [10] discussion method. We show our research overview in Fig. 1.

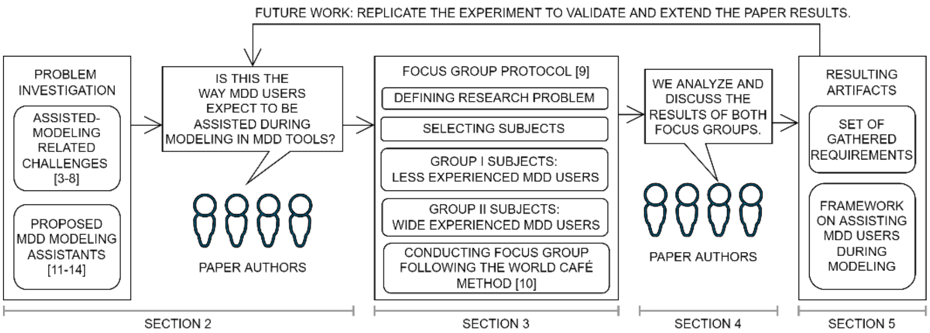

Fig. 1. Research overview.

We segmentate the focus groups as follows: the first group comprises less experienced MDD users, while the second group includes wide experienced MDD users. We aim to answer the following research questions by conducting such focus groups: i) what are the challenges perceived by MDD users during modeling for later code generation? ii) what are the features of the current modeling assistants that users like/dislike? and iii) what are the user's needs that are not yet satisfied by the current modeling assistants? We contrast the identified challenges with what related research previously observed. As a result, we gather a set of requirements of how MDD users expect to be assisted during modeling based on the challenges, features that users like/dislike, and their unsatisfied needs. Moreover, we propose an emerging framework for assisting MDD users during modeling based on such requirements. We expect researchers to propose novel modeling assistants to fulfill MDD users' requirements based on the proposed emerging framework.

The paper is structured as follows: in Section 2, we review identified challenges in assisting MDD users during modeling; in Section 3, we show the proposed focus group protocol; in Section 4, we present and discuss the focus group results; in Section 5, we gather the requirements based on the focus group results, and we propose the emerging framework for assisting MDD users during modeling; in Section 6, we discuss some threats to validity and limitations of our research; and, finally, in Section 7 we draw some conclusions and future work.

2 Background and motivation

Some authors have been interested in the challenges surrounding MDD tools. This section explores the challenges identified by such authors, emphasizing challenges in assisting MDD users during modeling.

Abrahao et al. [6] describe User eXperience (UX) challenges in MDD tools. They represent challenges such as integrating models with user needs, identifying UX features in MDD tools, and increasing MDD tools interoperability. Regarding assisting MDD users during modeling, the work emphasizes how to transform the current MDD focus on technology to focus on the users themselves. That implies understanding the needs and contexts of the MDD users. Likewise, Aggarwal et al. [5] discuss similar challenges to those identified by Abrahao et al. [6]. However, they highlight the MDD tool customization and specific domain support as relevant challenges for assisting MDD users during modeling.

Gottardi et al. [4] perform a systematic mapping looking for general-purpose challenges in model-driven software engineering. They identify two types of challenges after reviewing 4859 studies: maintenance and methodology challenges. Additionally, they identify maintenance challenges related to assisting MDD users during modeling, such as improving debuggers, model comparators, and model version managers.

Bucchiarone et al. [3] identify several challenges in model-driven software engineering. They classify such challenges into social, foundation, domain, community, and tool challenges. Mainly, they discuss assist-modeling-related challenges into the “tool challenges” classification, such as: including human-readable requirements, integrating heterogeneous models into views, improving visualization, allowing tool scalability, and including model traceability.

Mussbacher et al. [8] explicitly identify challenges in assisted modeling in model-driven software engineering. They focus on identifying challenges regarding MDD users and their needs during modeling. Such challenges include understanding the modeling context, understanding the modeler's skills and behavior, and transferring knowledge to different domains.

Bucchiarone et al. [7] discuss the importance of modeling adoption in organizations and the progress achieved so far. They emphasize challenges, such as using artificial intelligence, including multi-paradigm modeling, and improving model management to assist users during modeling.

Up to this point, we have explored some papers comprising challenges in MDD related explicitly to assisting MDD users during modeling. Such papers have inspired novel approaches focused on developing modeling assistants for MDD users. We refer to "modeling assistants" as any software artifact intended to assist MDD users during modeling. Some examples of such novel approaches are the following: intelligent modeling assistants [11], wizards for generating models [12], and model consistency checkers [13, 14]. All these data show the perspective of researchers on how to assist software modeling. However, the researchers’ perspective could differ from the way that MDD users expect to be assisted. Thus, the following main research question (MRQ) arises:

MRQ: *Is this the way MDD users expect to be assisted during modeling in MDD tools?*

3 Focus groups protocol

We propose to conduct two focus groups [9]: a cost-efficient way of obtaining practitioner and user experience [15] to gather data around the proposed MRQ. Moreover, we specialize the MRQ into the following three research questions:

- ***(RQ1) What are the challenges perceived by MDD users during modeling for later code generation?*** We expect MDD users to describe the challenges they perceive during modeling to contrast them with the challenges devised by related works (see Section 2). Moreover, we expect MDD users to classify each challenge depending on the impact (high or low) if such challenge is addressed and the urgency to be addressed (urgent or not urgent).
- ***(RQ2) What are the features of the current modeling assistants that users like/dislike?*** We expect MDD users to identify the features of modeling assistants with which they have interacted and classify what they like/dislike about them. Then, we match such features with the challenges identified in RQ1. Thus, we observe which challenges have been addressed by using modeling assistants, which should remain—i.e., which MDD users like—and which should be improved—i.e., which MDD users dislike.
- ***(RQ3) What are the user's needs that are unsatisfied by the current modeling assistants?*** We expect MDD users to analyze the modeling assistants they interacted with and specify which needs are currently unsatisfied. Then, we match such unsatisfied needs with features and challenges identified in RQ2 and RQ1. Furthermore, we ask MDD users to prioritize their unsatisfied needs by using MoSCoW: a low-effort and high-consistent requirement prioritization method [16, 17]. Therefore, we observe modeling-assistance-related needs that MDD tools must/should/could/will not satisfy in the future based on the MDD users' contributions.

3.1 Selecting subjects

We want to gather information from MDD users, so we select MDD users as the focus group subjects. However—to the best of our knowledge—a formal definition of "MDD user" has not been proposed in the literature. Indeed, some authors [6] point out that defining the MDD users is a current research challenge due to the many potential users of MDD tools. To overcome such a lack of the "MDD user" definition, we review some MDD tool-related papers to extract the authors' terms to refer to their users. As a result, we propose the following types of MDD users to select our focus group subjects[1]:

- ***(Type A)*** Non-IT related business-level users, referred to in the literature as business-level users [18], business analysts [19], end-users [19, 20], and domain users that are not necessarily computer scientists [21].

[1] Hereafter, we use the term "subject/subjects," referring to the focus group subject/subjects that fits/fit in one of the established MDD user types.

- ***(Type B)*** IT level users not strictly related to modeling, referred to in the literature as professional engineers with design experience [22], software developers [23–25], and IT personnel [26].
- ***(Type C)*** IT level users strictly related to modeling, referred in the literature as modelers [27], designers [28, 29], requirements engineers [30], model-driven developers [31], and application architects [32].

3.2 Segmentation

We conduct two focus groups composed of similar subjects—i.e., subjects with similar backgrounds and contexts—, facilitating the discussion among them [9, 33]. We show in detail the group segmentation in the following subsections.

Group I (GI)
GI comprises 11 subjects, all of whom are students of the bachelor course Rapid Software Prototyping (RASOP) at the Zürich University of Applied Sciences (ZHAW). The RASOP course is an elective 4 ECTS course offered to all engineering programs. We ask subjects to fulfill a demographic survey comprising data about their majors, their MDD tools experience, industry experience, and the type of MDD user they identify the most according to Section 3.1. We observe subjects are between 3rd and 4th year of their undergraduate studies (32.8 months avg. 6.0 months std. dev.), pursuing majors such as Computer Sciences (36.4%), Mechanical Engineering (27.3%), Industrial Engineering (18.2%), Electrical Engineering (9.1%), and Environmental Engineering (9.1%). Regarding their experience with MDD tools, most of them (90.9%) have three months of theoretical and practice training about MDD tools taught in the RASOP course. Only one subject (9.1%) has two months of experience using MDD tools before taking the RASOP course, having five months of experience in total. Regarding industry experience, most of the subjects (63.6%) do not have any industry experience yet. On the other hand, some subjects (36.4%) have industry experience from 0.13 to 24 months in process automation, web development, technical illustration, and software development. Finally, subjects identify themselves as Type A (45.5%) and Type B (54.5%) MDD users. The last result is consistent with what subjects answered about their majors. Most of the subjects with non-IT-related majors—e.g., Industrial Engineering—identify themselves with a non-IT-related business level MDD user—i.e., Type A MDD user. Likewise, most subjects with IT-related majors—e.g., Computer Sciences—identify themselves with an IT level MDD user not strictly related to modeling—i.e., Type B MDD user.

Group II (GII)
GII comprises three software engineering practitioners with wide experience in MDD tools working at Posity AG. Posity AG is a Swiss software development enterprise whose primary software development tool is an MDD tool named “Posity Design

Studio" (PDS) [2]. Software engineering practitioners at Posity AG mainly develop data-centric cloud applications by using Posity models in PDS. As we did with GI, we asked GII subjects to fulfill a demographic survey comprising data about their background, their years of industry experience, their expertise with MDD tools, and the type of MDD user with which they identify the most according to Section 3.1. We observe all subjects have a Computer Science background with 10 to 35 years of industry experience. Moreover, all subjects have extensive experience using MDD tools, having from 5 to 30 years of experience. Finally, subjects identify themselves as Type B (33.3%) and Type C (66.7%) MDD users.

3.3 Conducting the focus group sessions

We conduct two focus group sessions from 2 to 3 hours long with two moderators (the first two authors of this paper). We select the World Café method: "a simple yet powerful conversational process that helps groups of all sizes to engage in constructive dialogue" [10] to guide the discussion and interaction during the focus group session. Moreover, focus group subjects face each RQ following a brainstorming strategy [34]. First, subjects generate contributions to answer the RQ; then, subjects evaluate each contribution by discussing, improving, and refining its content. For the sake of simplicity, in this paper, we do not describe in full the setup of the World Café method, nor do we show the "raw" data obtained in the focus group. The data discussed in Section 4 have been prepared and refined by the authors of this paper to facilitate data presentation and analysis. However, we designed a public repository the focus group data can be consulted, including the protocol and the "raw" data[3].

4 Results and Discussion

4.1 RQ1: What are the challenges perceived by MDD users during modeling for later code generation?

We identify a set of 12 challenges: six contributed by GI, three contributed by GII, and three contributed by both (see Fig. 2). According to subjects' classification, six challenges are high priority and urgent, five challenges are high priority and not urgent, no challenge is low priority and urgent, and one challenge is low priority and not urgent. We discuss the identified challenges in the following subsections.

High priority and urgent challenges

GI subjects identify "decrease model and tool complexity" (C1) as a challenge since they state MDD tools are complex to use, hindering their usability. C1 agrees with Abrahao et al. [6] about the complexity of MDD tools, which negatively affects the UX

[2] https://posity.ch

[3] https://github.com/DavidMosquera/RCIS2022-Focus-Group-Data

during modeling. They affirm that MDD tools are much more complex than they need to be.

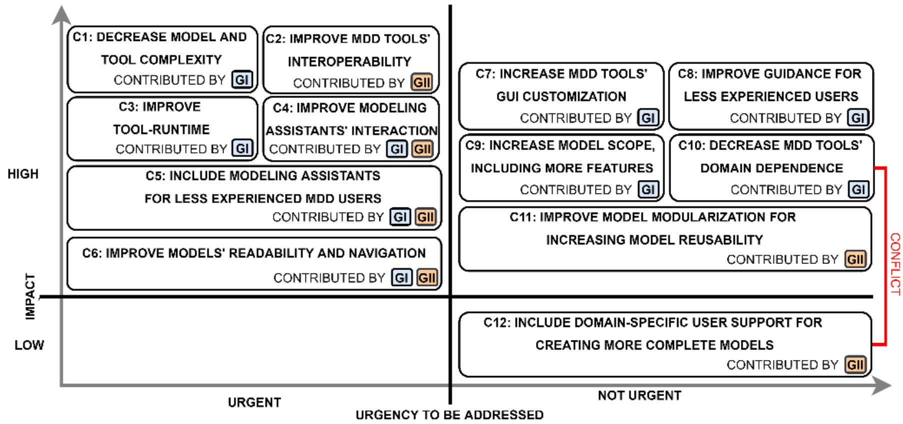

Fig. 2. Modeling challenges in MDD tools perceived by GI and GII subjects.

GII subjects identify “improve MDD tools’ interoperability” (C2) as a challenge. They state MDD tools are poorly integrated with other tools, hindering import data for creating models from different sources such as existing databases, CASE (Computer-Aided Software Engineering) tools, and other MDD tools. Abrahao et al. [6] also identify the interoperability of MDD tools as a challenge. They refer to this as “DSL-babel” (Domain Specific Language), since the more MDD tools are proposed with their DSLs, the more overall interoperability decrease.

GI subjects identify “improve tool-runtime” (C3) as a challenge since they state sometimes MDD tools runtime is bad, hindering a good experience during modeling. Bucchiarone et al. [3] refer to improving runtime as a challenge but mainly focus on model transformation languages and engines. C3 complements the vision of Bucchiarone et al. [3] to cover the runtime during transformations between models and the MDD tools in general.

Both groups identify two challenges regarding modeling assistants: i) “improve modeling assistants’ interaction” (C4) and ii) “include modeling assistants for less experienced MDD users” (C5). In C4, subjects state modeling assistants for creating models lack usability-related features, hindering their usability, such as undo-redo commands. In C5, subjects state that facing a "blank sheet of paper" to start modeling is sometimes frustrating—especially for less experienced MDD users. So, they consider including new modeling assistants—e.g., templates—based on expert knowledge for easing model creation by less experienced MDD users as a challenge. Mussbacher et al. [8] also discuss challenges related to modeling assistants—named in their paper as intelligent modeling assistants. They coincide with C4 and C5 since modeling assistants should adapt to MDD users’ context and skills to improve user interaction and ease modeling.

Both groups identify “improve models’ readability and navigation” (C6) as a challenge. Subjects state that graphic models contain a lot of information, getting complex very soon and hindering their readability. They point out that such a lack of readability is due to the nature of the modeling language rather than the complexity of what they are trying to model. Moreover, they experience problems navigating between models, hindering data visualization in and between models. Bucchiarone et al. [3] also refer to readability as a challenge but mainly focus on making modeling languages "human-readable." C6 complements Bucchiarone et al. [3] since it requires improving modeling language readability and navigation through big complex models and between models.

High priority and not urgent challenges

GI subjects identify "increase MDD tools' GUI (Graphical User Interface) customization" (C7) as a challenge. GI subjects state that MDD tools' GUI does not allow them to customize shortcuts and interface element locations, negatively affecting the usability. Abrahao et al. [6] also identify MDD tools' customization as a challenge, allowing for adapting menus, pallets, and workflows for improving UX during modeling. On the other hand, Mussbacher et al. [8] highlight that modeling assistants should allow customization to increase transparency.

GI subjects identify "improve guidance for less experienced MDD users" (C8) as a challenge. They state that when they try to use a new MDD tool, there is an entry barrier, hindering the MDD tool guidance. Some authors also identify user training and guidance as a challenge since improving training and guidance to less experienced MDD users decreases the learning curve of MDD tools [4–7].

GI subjects identify "increase the model scope and include more features" (C9) as a challenge. They state that MDD tools do not yet have all the functionalities developed using programming frameworks, limiting the software development scope. Bucchiarone et al. [7] mention that several initiatives started to address C9, extending the set of features offered in MDD tools reducing the gap between programming-based tools and MDD tools. But it remains an unaddressed challenge.

GI subjects identify "decrease MDD tools domain dependence" (C10) as a challenge since MDD tool's domain dependence limits the modeling scope—mainly when the model domain differs from the MDD tool domain. C10 is opposite to challenges identified by some authors regarding MDD tools' domain dependence. Some authors specify that domain dependence is required to improve MDD users' productivity since the general-purpose tool is never really fit for purpose; one size does not provide all [3, 6, 7]. However, Mussbacher et al. [8] point out that defining appropriate, generic, domain-independent modeling interfaces and protocols is challenging for designing high integrable modeling assistants and MDD tools. Therefore, C10 and Mussbacher et al. [8] are complementary points of view.

GII subjects identify "improve model modularization for increasing model reusability" (C11) as a challenge. They state that models are poorly modularized, hindering reusing some model elements into other models. The lack of model reusing causes them to repeat information on several models, decreasing maintainability. Bucchiarone et al. [7] also highlight model reusability as a challenge that can be addressed by using AI tools. Mussbacher et al. [8] go further and include reusability in modeling assistants to reuse them in different domains, tools, and contexts.

Low priority and not urgent challenges

GII subjects identify "include domain-specific user support for creating more complete models" (C12) as a challenge. They state domain-independent modeling assistants—e.g., wizards for creating models—do not allow them to make more specific models, limiting the modeling assistants' scope. As discussed in C10, some authors agree with C12 since domain dependence is required to improve MDD users' productivity [3, 6, 7]. However, C12 is conflictive with C10 since C10 aims to decrease MDD tools domain dependence. The conflict between C12 and C10 shows that there should be a trade-off between domain dependence and domain independence in MDD tools.

4.2 RQ2: What are the features of the current modeling assistants that users like/dislike?

After discussing the challenges perceived by MDD users in Section 4.1, subjects identify modeling assistants' features that they like/dislike. Subjects identify a set of 11 modeling assistants' features: seven features that they like and four features that they dislike (see Fig. 3). We match them with the identified challenges and discuss the identified features in the following subsections.

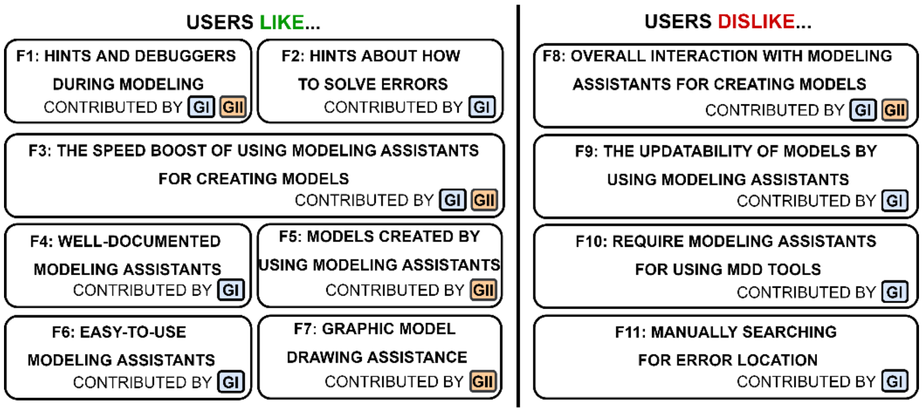

Fig. 3. Features of the current modeling assistants identified by GI and GII subjects.

Features subjects like

Both groups like hints and debuggers during modeling (F1). We match F1 with C1 (decrease model and tool complexity) since such debuggers and hints decrease model and tool complexity, easing error finding in models. Secondly, we observe that subjects like the following features related to C4 (improve modeling assistants' interaction): i) the speed boost during model creation by using modeling assistants (F3), ii) easy-to-use (F6) and well-documented (F4) modeling assistants, and iii) models created by using modeling assistants for creating models (F5). We match C4 with F3, F4, F5, and F6, since they aim to improve modeling assistants' interaction by bringing easy-to-use well-documented modeling assistants that boost modeling speed and create high-quality models. On the other hand, GII subjects like graphic model drawing assistance (F7) such as visual guides and entities connection. We observe such drawing assistance improves model readability, so we match F7 with C6 (improve models' readability and navigation). Finally, GI subjects like hints about how to solve errors (F2). We match F2 with C8 (improve guidance for less experienced users) since hints about how to solve errors increase the level of guidance for less experienced MDD users.

Feature subjects dislike

GI subjects dislike requiring modeling assistants to successfully use MDD tools (F10). They state MDD tools should be intuitive and easy to use without requiring modeling assistants. We match F10 with C1 (decrease model and tool complexity) since modeling assistants intend to decrease model tool complexity, but F10 shows subjects dislike the excessive use of modeling assistants. Secondly, both groups dislike the interaction with modeling assistants for creating models (F8). They dislike modeling assistants with many configurations, irreversible actions, required technical information, and poor dialog with the user to gather the data. We match F8 with C4 (improve modeling assistants' interaction) since F8 shows subjects dislike current modeling assistants' interaction. On the other hand, GI subjects dislike the updatability of models created by using modeling assistants (F9) since they experienced difficulties editing and completing them. We match F9 with C6 (improve models' readability and navigation) since GI subjects experienced such issues due to the lack of readability of the resulting models. Finally, GI subjects dislike manually searching for error locations (F11). We match F11 with C8 (improve guidance for less experienced users) since avoiding manually searching for error locations will improve the guidance for less experienced users.

4.3 RQ3: What are the user's needs that are not yet satisfied by the current modeling assistants?

After identifying challenges and features of current modeling assistants, group subjects describe a set of 10 needs that are not yet satisfied by modeling assistants. We ask them to use the MoSCoW requirements prioritization method [16, 17]. As a result, group subjects classify six needs as "must have" priority, three needs as "should have" priority, one need as "could have" priority, and no need as "will not have" priority (see Fig. 4). We deeply discuss and match such needs with what groups have answered in Section 4.1 and Section 4.2 in the following subsections.

"Must have" priority needs

"Must have" needs are non-negotiable and mandatory to be satisfied [16, 17]. Both groups agree that MDD tools *must* improve user dialog with modeling assistants (N3) and include undo/redo commands (N1). Moreover, GII subjects state modeling assistants' interaction *must* be improved by decreasing the technical knowledge required for using them (N6).

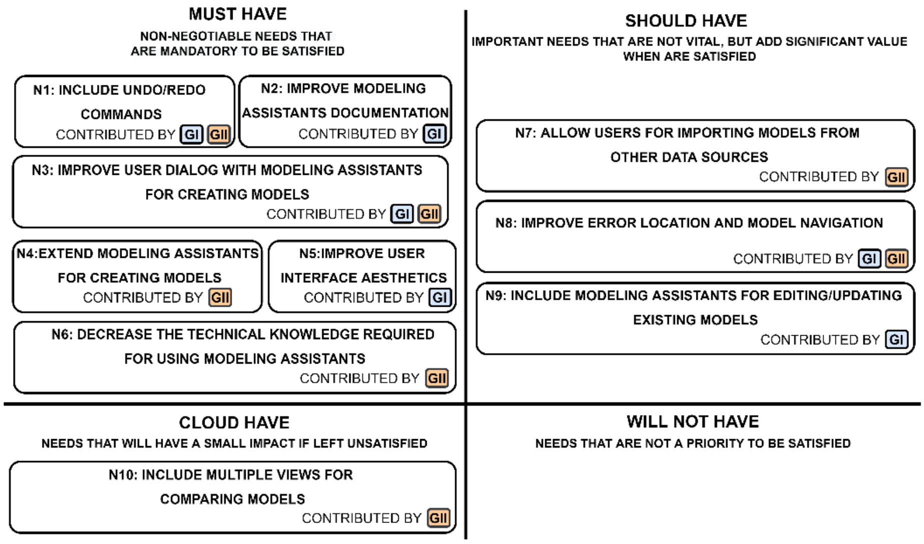

Fig. 4. Needs that are not yet satisfied by current modeling assistants; identified and prioritized by GI and GII subjects.

Furthermore, GI subjects state that MDD tools *must* improve their modeling assistants' documentation (N2). We match N3, N1, N6, and N2 with C4 (improve modeling assistants' interaction), F8 (subjects dislike overall interaction with modeling assistants for creating models), and F4 (subjects like well-documented modeling assistants) since addressing such set of needs will improve the general user interaction with modeling assistants (C4 and F8) and will bring well-documented modeling assistants also improving their interaction (C4 and F4). Secondly, GI subjects state user interface aesthetics *must* be improved (N5). We match N5 with C7 (increase MDD tools' GUI customization) since increasing GUI customization implies improving the GUI aesthetics. Finally, GII subjects state that modeling assistants must be extended to create models (N4). We match N4 with C12 (include domain-specific user support for creating more complete models) since GII subjects state some extension of modeling assistants *must* be included, such as domain-specific modeling assistants for creating models.

"Should have" priority needs

"Should have" priority needs are important needs that are not vital but add significant value when they are satisfied [16, 17]. Both groups agree MDD tools *should* improve error location and model navigation (N8). We match N8 with F11 (subjects dislike manually searching for error location) and C8 (improve guidance for less experienced users) since avoiding manually searching for errors will improve error location, also improving guidance for less experienced users. Secondly, GI subjects state MDD tools *should* include modeling assistants for editing/updating existing models (N9). We match N9 with C5 (include modeling assistants for less experienced MDD users) since both state MDD tools should include more modeling assistants—especially for less experienced users. Finally, GII subjects identify that MDD tools *should* allow users to import models from other data sources (N7). We match N7 with C2 (improve MDD tools' interoperability) since addressing N7 will increase MDD tools' interoperability.

"Could have" priority needs

"Could have" priority needs will have a small impact if left unsatisfied [16, 17]. GII subjects state MDD tools *could have* multiple views for comparing models (N10). We match N10 with C6 (improve models' readability and navigation) and F7 (subjects like graphic model drawing assistance) since including such views for comparing models will complement visual model drawing assistance, improving models' readability as a result.

5 Gathered requirements and an emerging framework

We gather a set of 12 requirements based on the focus group data and matchups between modeling challenges, current modeling assistants' features, and unsatisfied needs (see Table 1). To do so, we use identified challenges and their priority as the foundation for each proposed requirement. Then, if possible, we add which modeling assistants' features should remain—i.e., those that subjects like—and those that should be improved—i.e., those that subjects dislike. Finally, if possible, we include the unsatisfied needs for identifying what MDD tools must/could/should/will not have to address their associate challenge. After analyzing the gathered requirements in Table 1, we propose an emerging framework for assisting MDD users during modeling in MDD tools (see Fig. 5).

Table 1. Proposed MDD users' requirements based on focus group data.

MDD Users' Requirement
R1: Improving modeling assistants' interaction is a **high** priority and **urgent** challenge (C4). From current modeling assistants for addressing C4, MDD users **like** that they are easy-to-use (F6) and well-documented (F4), boosting modeling speed (F3) and producing high-quality models (F5). But MDD users **dislike** modeling assistants' overall interaction with the user (F8). In the future, MDD tools **must** improve user dialog with modeling assistants for creating models (N3), include undo/redo commands (N1), decrease required technical knowledge to use them (N6), and improve their documentation (N2) to address C4.
R2: Increasing MDD tools' GUI customization is a **high** priority and **not an urgent** challenge (C7). In the future, MDD tools **must** improve their GUI aesthetics (N5) to address C6.
R3: Include modeling assistants for less experienced MDD users is a **high** priority and **urgent** challenge (C5). In the future, MDD tools **should** include modeling assistants for editing/updating existing models (N9) to address C5.
R4: Improving the interoperability of MDD tools is a **high** priority and **urgent** challenge (C2). In the future, MDD tools **should** allow users to import models from other data sources (N7) to address C2.
R5: Improving guidance for less experienced users is a **high** priority and **not an urgent** challenge (C8). From current modeling assistants for addressing C8, MDD users **like** hints about how to solve errors, but MDD users **dislike** manually searching for error locations. In the future, MDD tools **should** improve error location and model navigation (N8) to address C8.

MDD Users' Requirement
R6: Improving models' readability and navigation is a **high** priority and **not an urgent** challenge (C6). From current modeling assistants for addressing C6, we **like** graphic model drawing assistance (F7), but we **dislike** the updatability of models created by modeling assistants (F9). In the future, MDD tools **could** have multiple views for comparing models (N10) to address C6.
R7: Including domain-specific user support for creating more complete models is a **low** priority and **not an urgent** challenge (C12). In the future, MDD tools **must** extend their modeling assistants' for creating models in specific domains (N4) to address C12.
R8: Improving tools runtime is a **high** priority and **urgent** challenge (C3).
R9: Decreasing model and tool complexity is a **high** priority and **urgent** challenge (C1). From current modeling assistants for addressing C1, MDD users **like** hints and debuggers during modeling (F1), but MDD users **dislike** requiring modeling assistants for using MDD tools (F10).
R10: Increasing model scope, including more features, is a **high** priority and **not an urgent** challenge (C9).
R11: Improving model modularization for increasing model reusability is a **high** priority and **not an urgent** challenge (C11).
R12: Decreasing MDD tools' domain dependence is a **high** priority and **not an urgent** challenge (C10).

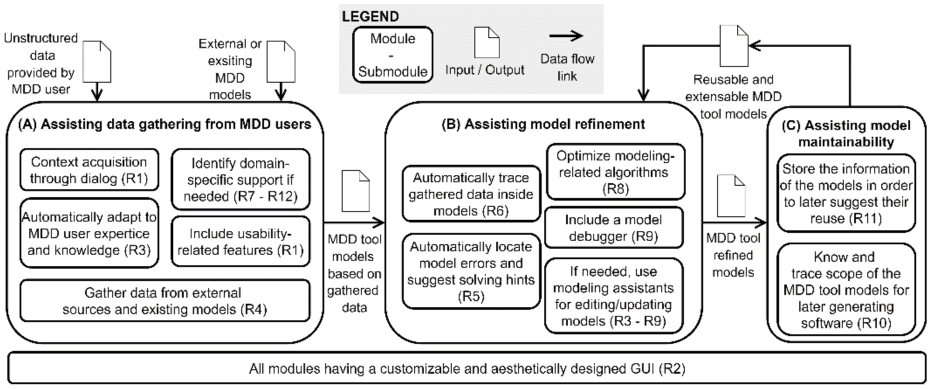

Fig. 5. Proposed emerging framework for modeling assistance in MDD tools.

The proposed emerging framework aims for allowing model-driven engineers and researchers to improve their modeling assistants based on MDD users' requirements. So, we divide such framework into three modules: A) assisting data gathering from MDD users, B) assisting model refinement, and C) assisting model maintainability. Module A aims to assist data gathering for creating models by using unstructured data or existing MDD models from external sources provided by the MDD user. Thus, Module A adapts to the user improving the user interaction and boosting modeling speed. On the other hand, Module B assists MDD users in refining the models created by Module A by tracing the data and easing error solving. Moreover, Module B includes optimized algorithms for improving tool runtime, and modeling assistants to improve model updatability. Finally, Module C allows MDD users to maintain the models through time by suggesting reusing existing models and tracing model scope into the generated software. All proposed modules must have a customizable and aesthetically designed GUI.

Some authors have already proposed frameworks for assisting modeling in MDD tools. For instance, Mussbacher et al. [8] propose a framework on intelligent modeling assistants, mainly focusing on the user interaction with the modeling assistants—such as module A of our framework. We note that our results reinforce such a research line by adding the "assisting model refinement (B)" and "assisting model maintainability (C)" modules. Thus, the frameworks can complement each other and generate modeling assistants closer to what the MDD users expect during modeling.

6 Threats to validity and limitations

We have identified some threats to validity and limitations during the execution of our focus groups. Regarding *conclusion validity,* we recognize that our research has a low statistical power since the population sample is small—i.e., having a sample of 14 subjects threatens our conclusion validity. Despite this, we consider our results useful and a first step to continue increasing the population sample by replicating the experiment, especially the Type C subjects, since they are a minority in our focus group segmentation (2 out of 14 subjects). Moreover, we decided to select subjects with similar backgrounds, making the focus groups homogeneous. This decision allows us to increase the *conclusion validity* since we avoid variations on the results due to individual differences among the focus group subjects—a.k.a. *random heterogeneity of subjects'* threat. However, having homogeneous groups also reduces the *external validity,* limiting our ability to generalize the focus group results. To avoid this threat arising in further replications of our focus group, we consider having more heterogeneous groups mixing them by subject types—e.g., having groups with the same number of Type A, B, and C subjects. Furthermore, the results from software engineering practitioners—i.e., GII subjects—are limited to employees from one enterprise that uses an MDD tool to develop software—i.e., Posity AG. This segmentation reduces the generality of our results since other software development enterprises do not use only MDD tools to build software. Therefore, we plan to include more software engineer practitioners from different enterprises and backgrounds in future replications, avoiding these external validity threats. Regarding *construct validity,* we decided that the focus group subjects will face each RQ "from scratch." This may have caused already identified challenges in the literature not to be discussed during the focus group sessions. However, we also avoid the subjects being biased from previously conceived challenges since we aim to gather requirements directly from MDD users and compare them with such challenges—i.e., we avoid the *interaction of testing and treatment*—increasing the *construct validity*. To overcome both limitations, we propose to use an intermediate model—e.g., the proposed framework in this paper, existing usability heuristics, among others—that allows us to discuss both subjects and literature challenges. Thus, subjects will not be biased with existing challenges, and we can identify which identified challenges are not addressed allowing us to observe unidentified challenges.

7 Conclusions and Further Work

In this paper, we have executed two focus groups based on [9] and following the World Café method [10] to answer three research questions: i) what are the challenges perceived by MDD users during modeling for later code generation? ii) what are the features of the current modeling assistants that users like/dislike? and iii) what are the user's needs that are not yet satisfied by the current modeling assistants? Such research questions aimed to collect data to gather the perspective of MDD users on how they expect to be assisted during modeling in MDD tools. After conducting the focus groups, we observed all identified challenges match or complement at least one challenge previously identified by researchers in the literature [3–8]. Moreover, we matched features that MDD users like/dislike, their unsatisfied needs, and their perceived modeling challenges to gather requirements to assist MDD users during modeling. So, we identified which features of the modeling assistants should remain and which should be improved based on what MDD users like/dislike. Furthermore, we identified which features modeling assistants must/should/could/will not have to satisfy MDD users' needs based on MoSCoW [16, 17] prioritization method. As a result, we gathered 12 requirements based on such data. Then, we proposed an emerging framework composed of three modules: A) assisting data gathering from MDD users, B) assisting model refinement, and C) assisting model maintainability. This emerging framework is a starting point for model-driven engineers and researchers to improve their modeling assistants and increase MDD tools adoption in practice. As future work, we expect to build modeling assistants following the proposed emerging framework and validate them in experiments with MDD users. Moreover, we will continue replicating our focus group, collecting more requirements to increase, improve, and validate the gathered requirements and the proposed framework. Our objective in the future is to generalize the results to a global definition of MDD users.

Acknowledgments

This work has been supported by the Zürich University of Applied Sciences (ZHAW) – School of Engineering: Institute for Applied Information Technology (InIT). Moreover, we would like to thank all RASOP course students and Posity AG practitioners for actively participating during the focus group sessions, allowing us to gather all the data we used to build our research.

References

1. Sendall, S., Kozaczynski, W.: Model transformation: the heart and soul of model-driven software development. IEEE Software. 20, 42–45 (2003).
2. Panach, J.I., España, S., Dieste, Ó., Pastor, Ó., Juristo, N.: In search of evidence for model-driven development claims: An experiment on quality, effort, productivity, and satisfaction. Information and Software Technology. 62, 164–186 (2015).

3. Bucchiarone, A., Cabot, J., Paige, R.F., Pierantonio, A.: Grand challenges in model-driven engineering: an analysis of the state of the research. Software and Systems Modeling. 19, 5–13 (2020).
4. Gottardi, T., Vaccare Braga, R.T.: Understanding the Successes and Challenges of Model-Driven Software Engineering - A Comprehensive Systematic Mapping. In: 2018 XLIV Latin American Computer Conference (CLEI). pp. 129–138. IEEE (2018).
5. Aggarwal, P.K., Sharma, S., Riya, Jain, P., Anupam: Gaps identification for user experience for model driven engineering. In: 11th International Conference on Cloud Computing, Data Science and Engineering. pp. 196–199. IEEE (2021).
6. Abrahao, S., Bourdeleau, F., Cheng, B., Kokaly, S., Paige, R., Stoerrle, H., Whittle, J.: User Experience for Model-Driven Engineering: Challenges and Future Directions. In: 20th International Conference on Model Driven Engineering Languages and Systems. pp. 229–236. IEE (2017).
7. Bucchiarone, A., Ciccozzi, F., Lambers, L., Pierantonio, A., Tichy, M., Tisi, M., Wortmann, A., Zaytsev, V.: What Is the Future of Modeling? IEEE Software. 38, 119–127 (2021).
8. Mussbacher, G., Combemale, B., Kienzle, J., Abrahão, S., Ali, H., Bencomo, N., Búr, M., Burgueño, L., Engels, G., Jeanjean, P., Jézéquel, J.-M., Kühn, T., Mosser, S., Sahraoui, H., Syriani, E., Varró, D., Weyssow, M.: Opportunities in intelligent modeling assistance. Software and Systems Modeling. 19, 1045–1053. (2020).
9. Kontio, J., Bragge, J., Lehtola, L.: The Focus Group Method as an Empirical Tool in Software Engineering. In: Guide to Advanced Empirical Software Engineering. pp. 93–116. (2008).
10. Tan, S., Brown, J.: The World Café in Singapore. The Journal of Applied Behavioral Science. 41, 83–90 (2005).
11. Savary-Leblanc, M.: Improving MBSE tools UX with AI-Empowered software assistants. In: 22nd International Conference on Model Driven Engineering Languages and Systems Companion. pp. 648–652. IEEE (2019).
12. ben Fraj, I., BenDaly Hlaoui, Y., BenAyed, L.: A reactive system for specifying and running flexible cloud service business processes based on machine learning. In: 45th Annual Computers, Software, and Applications Conference (COMPSAC). pp. 1483–1489. IEEE (2021).
13. Chavez, H.M., Shen, W., France, R.B., Mechling, B.A., Li, G.: An Approach to Checking Consistency between UML Class Model and Its Java Implementation. IEEE Transactions on Software Engineering. 42, 322–344 (2016).
14. Wang, C., Cavarra, A.: Checking Model Consistency Using Data-Flow Testing. In: 16th Asia-Pacific Software Engineering Conference. pp. 414–421. IEEE (2009).
15. Kontio, J., Lehtola, L., Bragge, J.: Using the focus group method in software engineering: obtaining practitioner and user experiences. In: International Symposium on Empirical Software Engineering, ISESE ’04. pp. 271–280. IEEE (2004).
16. Ali Khan, J., Ur Rehman, I., Hayat Khan, Y., Javed Khan, I., Rashid, S.: Comparison of Requirement Prioritization Techniques to Find Best Prioritization Technique. International Journal of Modern Education and Computer Science. 7, 53–59 (2015).
17. Hatton, S.: Early prioritisation of goals. In: Advances in Conceptual Modeling – Foundations and Applications, ER. pp. 235–244. Springer (2007).
18. Sinha, S., Astigarraga, T., Hull, R.B., Jean-Louis, N., Sreedhar, V., Chen, H., Hu, L.X., Carpi, F.E., Cannata, J.A.B., Loach, W.: Auto-Generation of Domain-Specific Systems: Cloud-Hosted DevOps for Business Users. In: 13th International Conference on Cloud Computing (CLOUD). pp. 219–228. IEEE (2020).
19. Sousa, K., Mendonça, H., Lievyns, A., Vanderdonckt, J.: Getting users involved in aligning their needs with business processes models and systems. Business Process Management Journal. 17, 748–786 (2011).

20. Pérez, F., Valderas, P., Fons, J.: Towards the Involvement of End-Users within Model-Driven Development. In: International Symposium on End-User Development. pp. 258–263. Springer (2011).
21. Fuhrmann, H., von Hanxleden, R.: Taming Graphical Modeling. In: Model-Driven Engineering Languages and Systems. pp. 196–210. Springer (2010).
22. Paz, A., el Boussaidi, G., Hafedh, M.: checsdm: A Method for Ensuring Consistency in Heterogeneous Safety-Critical System Design. IEEE Transactions on Software Engineering. 47, 2713–2739 (2020).
23. Schottle, M., Kienzle, J.: Concern-oriented interfaces for model-based reuse of APIs. In: 18th International Conference on Model Driven Engineering Languages and Systems. pp. 286–291. IEEE (2015).
24. Ohrndorf, M., Pietsch, C., Kelter, U., Grunske, L., Kehrer, T.: History-based Model Repair Recommendations. ACM Transactions on Software Engineering and Methodology. 30, 1–46 (2021).
25. Getir, S., Grunske, L., Bernasko, C.K., Käfer, V., Sanwald, T., Tichy, M.: CoWolf – A Generic Framework for Multi-view Co-evolution and Evaluation of Models. In: International Conference on Theory and Practice of Model Transformations. pp. 34–40. Springer (2015).
26. Akiki, P.A., Bandara, A.K., Yu, Y.: Cedar studio: an IDE supporting adaptive model-driven user interfaces for enterprise applications. In: 5th ACM SIGCHI symposium on Engineering interactive computing systems. pp. 139. ACM Press (2013).
27. Oberweis, A., Reussner, R.: Model Validation and Verification Options in a Contemporary UML and OCL Analysis Tool. In: Modellierung. pp. 203–218 (2016).
28. Danenas, P., Skersys, T., Butleris, R.: Extending Drag-and-Drop Actions-Based Model-to-Model Transformations with Natural Language Processing. Applied Sciences. 10, 1–37 (2020).
29. Ameedeen, M.A., Bordbar, B., Anane, R.: Model interoperability via Model Driven Development. Journal of Computer and System Sciences. 77, 332–347 (2011).
30. Pires, P.F., Delicato, F.C., Cóbe, R., Batista, T., Davis, J.G., Song, J.H.: Integrating ontologies, model driven, and CNL in a multi-viewed approach for requirements engineering. Requirements Engineering. 16, 133–160 (2011).
31. Araùjo De Oliveira, R., Dingel, J., Oliveira, R.: Supporting Model Refinement with Equivalence Checking in the Context of Model-Driven Engineering with UML-RT. In: Model-Driven Engineering, Verification and Validation Workshop at the MODELS conference. (2017).
32. Ricci, L.A., Schwabe, D.: An authoring environment for model-driven web applications. In: 12th Brazilian symposium on Multimedia and the web. pp. 11–19. ACM Press (2006).
33. Morgan, D.: Focus Groups as Qualitative Research. SAGE Publications, Inc., 2455 Teller Road, Thousand Oaks California 91320 United States of America (1997).
34. Paetsch, F., Eberlein, A., Maurer, F.: Requirements engineering and agile software development. In: 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises. pp. 308–313. IEEE (2003).