The MERLIN Method to create Modeling Assistants for Model-Driven Development Tools

David Mosquera [1,2]

[1] Universitat Politècnica de València, Camí de Vera, Valencia 46022, Spain
[2] Zürich University of Applied Sciences, Gertrudstrasse 15, Winterthur 8400, Switzerland

Abstract

Several authors have proposed novel model-driven development (MDD) tools for years, promising to increase software development productivity and decrease software time-to-market. Although their effort to achieve such a promise, MDD tools have not shown a significant difference in benefits compared to classical code-centric development. This issue has led some authors to identify challenges that model-driven engineers—i.e., who create MDD tools—should address to improve current MDD tools. Specifically, some of these challenges arise from the lack of well-designed assistance during modeling in MDD tools. Due to that, some authors have proposed modeling assistants to address such challenges. However, some modeling assistants lack modeling context-awareness, hindering user experience in MDD tools. On the other hand, some authors have proposed context-aware modeling assistants. Nevertheless, such authors use domain- and modeling-task-dependent methods for proposing such context-aware modeling assistants, lacking generality. Therefore—in this Ph.D. thesis—we propose MERLIN: a MEthod for cReating modeLIng assistaNts in the context of MDD tools. MERLIN allows model-driven engineers for implementing context-aware modeling assistants by using a domain- and modeling-task-independent method. We frame our research using the Design Science method, proposing a set of goals and research questions. We expect that modeling assistants implemented by using MERLIN increase the technology acceptance of MDD tools. Finally, we discuss the progress achieved so far and the research plan.

Keywords

Modeling assistants, Method, Model-driven development, Context-aware assistance

1. Introduction

Model-driven development (MDD) aims to increase development team productivity and decrease software time-to-market [1]. MDD tools use text-based and graphical-based models to automatically transform them into functional software to achieve such a goal. Some authors have performed experiments in search of evidence of MDD tools' benefits in terms of quality, effort, productivity, among others [2], [3]. However, they observe no significant difference between the MDD and the classical code-centric approaches regarding such benefits. These results show that MDD tools are still maturing, having challenges to address [4–7]. Specifically, Mussbacher et al. [7] identify that improving modeling assistants is an urgent challenge to achieve the MDD benefits.

> ***Modeling assistant:*** we refer to "modeling assistant" in this paper as any software artifact that aims to assist users in performing a modeling task such as creating, refining, and tracing models in the context of an MDD tool.

Some authors have proposed modeling assistants, addressing some of the Mussbacher et al. [7] challenges [8–19]. However, some of them lack modeling context-awareness [8, 10, 12–14, 17, 18]: one of the main features identified by Mussbacher et al. [7] to improve user experience in MDD tools. On the other hand, some authors have proposed context-aware modeling assistants [9, 11, 15, 16].

Proceedings of the Doctoral Consortium Papers Presented at the 34th International Conference on Advanced Information Systems Engineering (CAiSE 2022), June 06–10, 2022, Leuven, Belgium
EMAIL: mosq@zhaw.ch (D. Mosquera)
ORCID: 0000-0002-0552-7878 (D. Mosquera)


However, they have followed domain- and modeling-task-dependent methods—i.e., they have flowed methods that set the proposed modeling assistants to a particular modeling language and a specific modeling task—lacking generality. Therefore, in this Ph.D. Thesis we propose MERLIN: a MEthod for cReating modeLIng assistaNts in the context of MDD tools. MERLIN allows model-driven engineers—i.e., who develop MDD tools—to create context-aware modeling assistants. MERLIN is domain- and modeling-task-independent, allowing model-driven engineers to create modeling assistants to help users during modeling tasks in MDD tools. We frame our research in the Design Science method [20], establishing a set of goals and research questions to be addressed.

In this paper, we show the progress achieved so far in our research. We discuss some progress related to the *problem investigation* task, including a focus group and a systematic mapping. Finally, we show a preliminary MERLIN design and a proof of concept. As further steps, we will improve MERLIN by applying Method Engineering efforts. Moreover, we will implement modeling assistants with industry partners using MERLIN, acquiring new requirements to improve our work and data to validate our hypotheses.

This paper is structured as follows: In Section 2, we review related works on modeling assistants in MDD tools, motivating this Ph.D. Thesis; in Section 3, we show the research method, goals, and research questions around MERLIN; in Section 4, we explore the progress achieved so far in this Ph.D. Thesis; and, finally, in Section 5, we discuss some conclusions and further steps.

2. Related works and motivation

Modeling assistants have gained attention in the last years by several researchers [8–19]. After reviewing the proposed modeling assistants, we classify them into two different types depending on what they aim to assist on: i) modeling assistants for creating models [13, 14, 16, 17]; and ii) modeling assistants for refining existing models [8–12, 15, 18, 19]. Such modeling assistants assist users in one or more modeling tasks related to each classification. We deeply describe these approaches as follows:

- **Creating models.** These approaches focus on assisting users to create models in MDD tools, decreasing the modeling complexity. Fraj et al. [17] assist users in creating models by using abstract and interactive templates, specifically in the cloud service business context. Such templates allow users to create models automatically without interacting directly with the modeling language. On the other hand, Savary-Leblanc [13], Agt-Rickauer et al. [16], and Steimann and Ulke [14] propose modeling assistants that recommend possible elements during modeling. Such recommendations help users devise relationships, attributes, and terms, among other model elements, based on external knowledge repositories.
- **Refining models.** These approaches focus on easing model refinement in MDD tools, improving the models' maintainability. Pourali and Atlee [11] propose a Focus+Context approach to reduce the cognitive challenges of model debugging. They improve users' ability to maintain more error-free models, reducing time invested on model debugging tasks. Wang and Cavarra [10], Paz et al. [8], and Chavez et al. [19] propose modeling assistants for ensuring models' consistency with software artifacts, such as other models, code, and documents. Such model checking approaches allow users to uncover inconsistencies between models, improving models' quality. Finally, Cabral and Sampaio [12], Shen et al. [18], Kehrer et al. [9], and Ohrndof et al. [15] devise approaches for helping users on model repairment tasks by recommending model changes, improving model versioning.

The reviewed modeling assistants [8–19] exemplify research efforts that researchers have done to assist users in MDD tools. However, we observe some approaches [8, 10, 12–14, 17, 18] lack what Mussbacher et al. [7] have identified as a challenge in modeling assistance: context-awareness during modeling. Context-awareness during modeling allows modeling assistants to understand the users' characteristics such as behaviors, skills, and needs, improving the user experience with the MDD tool. On the other hand, some authors have proposed context-aware modeling assistants [9, 11, 15, 16]. However, they use methods that limit the assistance to specific domains—such as limiting the modeling language to UML [11]—and specific modeling tasks—such as limiting the assistance to model repairment [9]—lacking generality. Therefore, the following main technical research problem arises:

(TRP) How to **design a method** that satisfies **domain- and modeling-task-independence requirements** for **creating context-aware modeling assistants** in the context of **MDD tools**?

3. MERLIN: Research method, goals, and research questions

To address the main TRP, we propose to design MERLIN: a MEthod for cReating modeLIng assistaNts in the context of MDD tools. We frame this Ph.D. Thesis in the Design Science method [20]. The object of study in the Design Science method is an artifact in context. In this Ph.D. Thesis, MERLIN is the artifact we will design and investigate in the context of modeling assistance in MDD tools. Moreover, we specify a set of Knowledge, Instrument design, Prediction, and Artifact design goals to frame our research project. Knowledge goals are to describe phenomena and to explain them [20]. Instrument design goals are the lowest-level design goals [20]. Artifact design goals aim to solve, mitigate, or improve some problem in context [20]. Finally, Prediction goals are beliefs about what will happen in the future [20]. We present the research goal hierarchy of the Design Science for MERLIN in Figure 1.

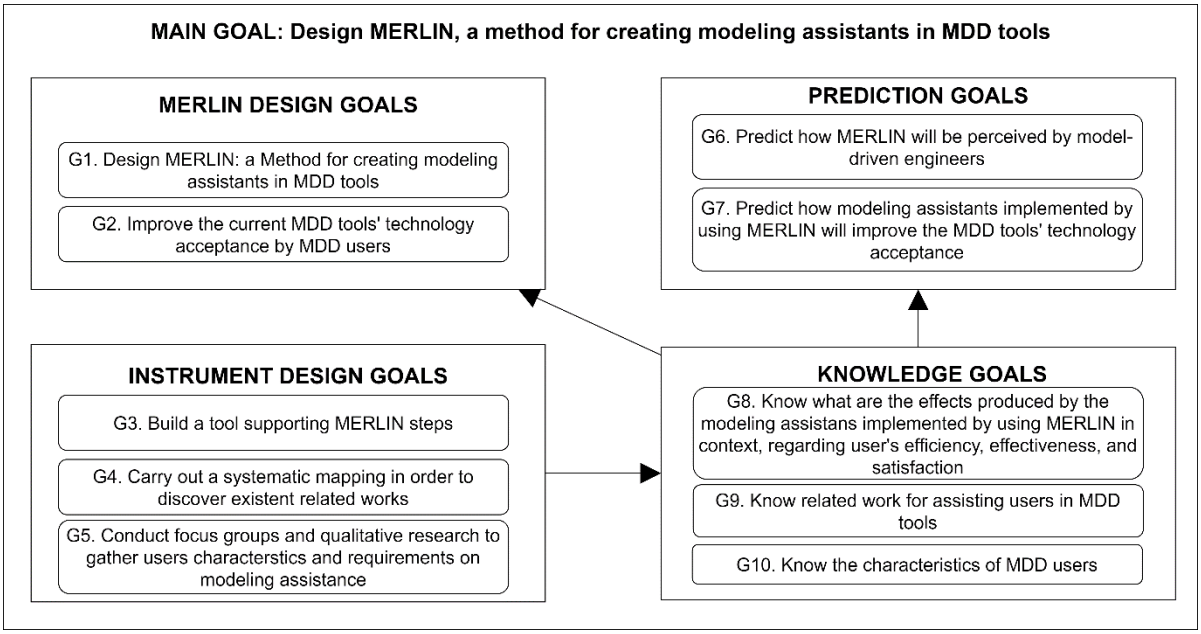

Figure 1: Goal hierarchy of the Design Science for MERLIN.

Regarding Knowledge goals, we aim to explore related works on modeling assistants in MDD tools (G9), gather characteristics of the MDD tools' users (G10), and know the effects of modeling assistants implemented by using MERLIN in context (G8). To address G9, we plan to carry out a systematic mapping to discover existent related works (G4). Moreover, we plan to conduct focus groups and qualitative research, gathering user characteristics and requirements on modeling assistance to address G10. Having G4 and G5 addressed, we will design MERLIN (G1), aiming for improving the current MDD tools' technology acceptance by MDD users (G2) based on what we gathered on the systematic mapping and qualitative research. To answer G8, we build a tool for supporting MERLIN steps (G3) to test modeling assistants implemented using MERLIN in terms of user efficiency, effectiveness, and satisfaction.

We propose G6 and G7 prediction goals to conduct empirical research with MERLIN and generalize the results into any MDD tool and modeling assistant that MERLIN can be applied for. These goals will require several empirical exercises that are maybe out-of-scope in this Ph.D. Thesis. However, we include them since we plan to perform empirical research, first steps to address G6 and G7.

The proposed goals arise some challenges that we must meet in the context of this Ph.D. Thesis. Hence, we devise technical research problems (TRP)s and knowledge questions (KQ)s based on such challenges. Technical research problems—a.k.a design problems—aim to (re)design an artifact, contributing to the achievement of some goal [20]. On the other hand, knowledge questions ask for knowledge about the world without calling for an improvement [20]. We present the list of research questions (RQ)s derived from the TRPs and KQs as follows:

• **RQ1.** (KQ) *What are the characteristics of MDD users?* This research question is motivated by G10. To answer RQ1, we will conduct focus groups and qualitative research with the users as we established in G5.
• **RQ2.** (KQ) *What are the existing modeling assistants to assist users in MDD tools?* This research question is motivated by G9. To answer RQ2, we will perform a systematic mapping looking for related works on modeling assistants in MDD tools as we established in G4.
• **RQ3.** (TRP) *How to design the MERLIN method that satisfies domain- and modeling-task-independence requirements for creating context-aware modeling assistants that increase MDD tools' technology acceptance?* This research question contains the main TRP we motivated in Section 2. Moreover, RQ3 relies on G1 and G2 since both goals aim for designing MERLIN. To answer RQ1, we will make a Method Engineering effort to design MERLIN successfully.
• **RQ4.** (TRP) *How to develop a tool for supporting the MERLIN method?* This research question is motivated by G3. To answer RQ4, we will explore software development technologies that allow model-driven engineers to integrate the modeling assistants developed by using MERLIN with their under- and yet-developed MDD tools.
• **RQ5.** (KQ) *What effects produce modeling assistants implemented using MERLIN in context in terms of the TAM?* This research question is founded on G5. To answer RQ5, we plan to perform empirical research in academic and industry contexts. The results of answering RQ5 will be the first steps to address G6 and G7. Based on RQ5, we propose the following three hypotheses (H):

H1. MDD tool users' efficiency improves when they use modeling assistants implemented by using MERLIN.
H2. MDD tool users' effectiveness improves when they use modeling assistants implemented by using MERLIN.
H3. MDD tool users' satisfaction—i.e., perceived ease of use, perceived usefulness, and intention to use—improves when they use modeling assistants implemented by using MERLIN.

We formulate these hypotheses to test MERLIN based on the technology acceptance model (TAM) proposed by Moody [23].

Since we frame this Ph.D. Thesis in the Design Science method [10], we will perform the activities around three tasks (T): i) (T1) *problem investigation*, ii) (T2) *treatment design*, and iii) (T3) *treatment validation*. We show our proposed design cycle for designing MERLIN based on the goals and research questions in Figure 2.

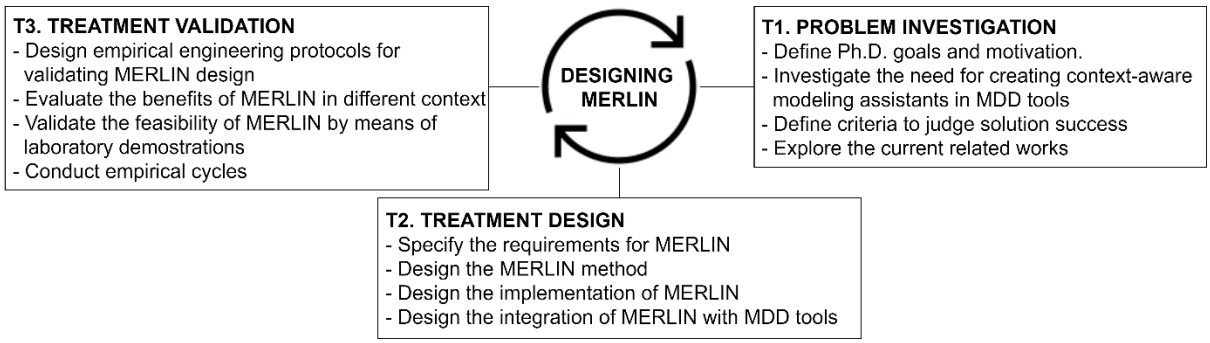

Figure 2: Proposed design cycle for designing MERLIN, based on the Design Science method [10].

4. MERLIN: The progress achieved so far

Since we started this Ph.D. Thesis, we have progressed in answering the proposed RQs and addressing the research goals. First, we conducted a focus group with 14 subjects, having both expert and novice users of MDD tools. We answered questions such as: i) what challenges perceive MDD users during modeling? ii) what are the features of current modeling assistants that users like/dislike? and iii) what are the users' needs that are not yet satisfied by the current modeling assistants? As a result, we gather a set of prioritized requirements that help us characterize users of MDD tools, i.e., that help us address RQ1. In future cycles, we will replicate this focus group having more users of MDD

tools, improving the set of prioritized requirements. We have already reported these results, expecting to publish them in the following months.

At the same time, we have been conducting a systematic mapping to gather existing modeling assistants. We reviewed more than 1,800 papers based on a database search strategy, and, currently, we are performing a *snowballing* backward/forward search strategy based on the first selected primary studies. We extract data around their goals, limitations, and evaluations. As a result, we will have the data to address RQ2. We are working on finishing the *snowballing* search and on reporting the results.

Up to this point, we described progress related to the *problem investigation* (T1) task. We have also made progress in the *treatment design* (T2) task, aiming to answer RQ3. We conceive MERLIN as a method to implement context-aware modeling assistants. To this end, we plan to adapt the formal framework for context-aware systems proposed by [22] to the context of modeling assistance in MDD tools. So, we expect the MERLIN method will be composed of the following steps:

1. *Framing the modeling assistance*: Model-driven engineers propose modeling assistants to ease modeling tasks. So, as the first step, model-driven engineers should frame the modeling assistance in the MDD tools, defining the modeling tasks the modeling assistant will assist on. That includes answering *why, when,* and *how* will the modeling assistant be used? Answering *why*, model-driven engineers will identify the problems and goals to be addressed by using the modeling assistant. Answering *when,* model-driven engineers will briefly describe the modeling assistant's contexts of use. Finally, answering *how,* model-driven engineers frame the means to assist the user.
2. *Designing the modeling assistant sensors*: To achieve the identified goals and problems, modeling assistants need to acquire data. Thus, we propose that model-driven engineers design *sensors*[2], acquiring such data and transforming it into meaningful information. Designing a sensor comprises defining a set of inputs, outputs, and how inputs will be transformed into outputs.
3. *Identifying the modeling contexts*: Based on the sensors' outputs, modeling assistants need to detect the different modeling contexts. Thus, model-driven engineers should identify the modeling contexts, considering their answer of *when* the modeling assistant will be used. Identifying the modeling contexts comprise describing the context, defining the context states, and establishing how sensor outputs will trigger each context state.
4. *Defining the interaction between the user and the modeling assistant*: Having established the possible triggerable context states, modeling assistants need to interact with the users. Therefore, model-driven engineers should define the interaction between the user and the modeling assistant in the MDD tool, considering their answer of *how* the modeling assistant will be used. Defining such interaction comprises selecting the dialog between user and modeling assistant, which information will be displayed, and which information the modeling assistant will request to the user.
5. *Implementing the modeling assistant functionalities*: Finally, model-driven engineers should implement the modeling assistant functionalities, including selecting the implementation technology and deciding how to integrate it with an MDD tool.

4.1. MERLIN: Proof of concept

We have implemented a modeling assistant by applying the current status of MERLIN. The main goal of such a modeling assistant is to help users during model refinement. Primarily, we focus on assisting users in creating and maintaining traces between models in MDD tools, decreasing traceability effort. For the sake of simplicity, we briefly summarize the design and implementation of this modeling assistant in Figure 3. Moreover, we deeply reported this modeling assistant design and implementation on a separate paper, currently under review.

As final comments regarding MERLIN progress achieved so far, we expect to improve MERLIN with more formal Method Engineering efforts in the future. Such an improvement will require more iterations in the design cycle. Moreover, we will work hand-by-hand with industry partners to develop

[2] Model-driven engineers can propose logical, virtual, or physical sensors [22]. Such classification depends on how the sensor acquire the data.

and test novel context-aware modeling assistants to their MDD tools by using MERLIN, addressing RQ4 and RQ5.

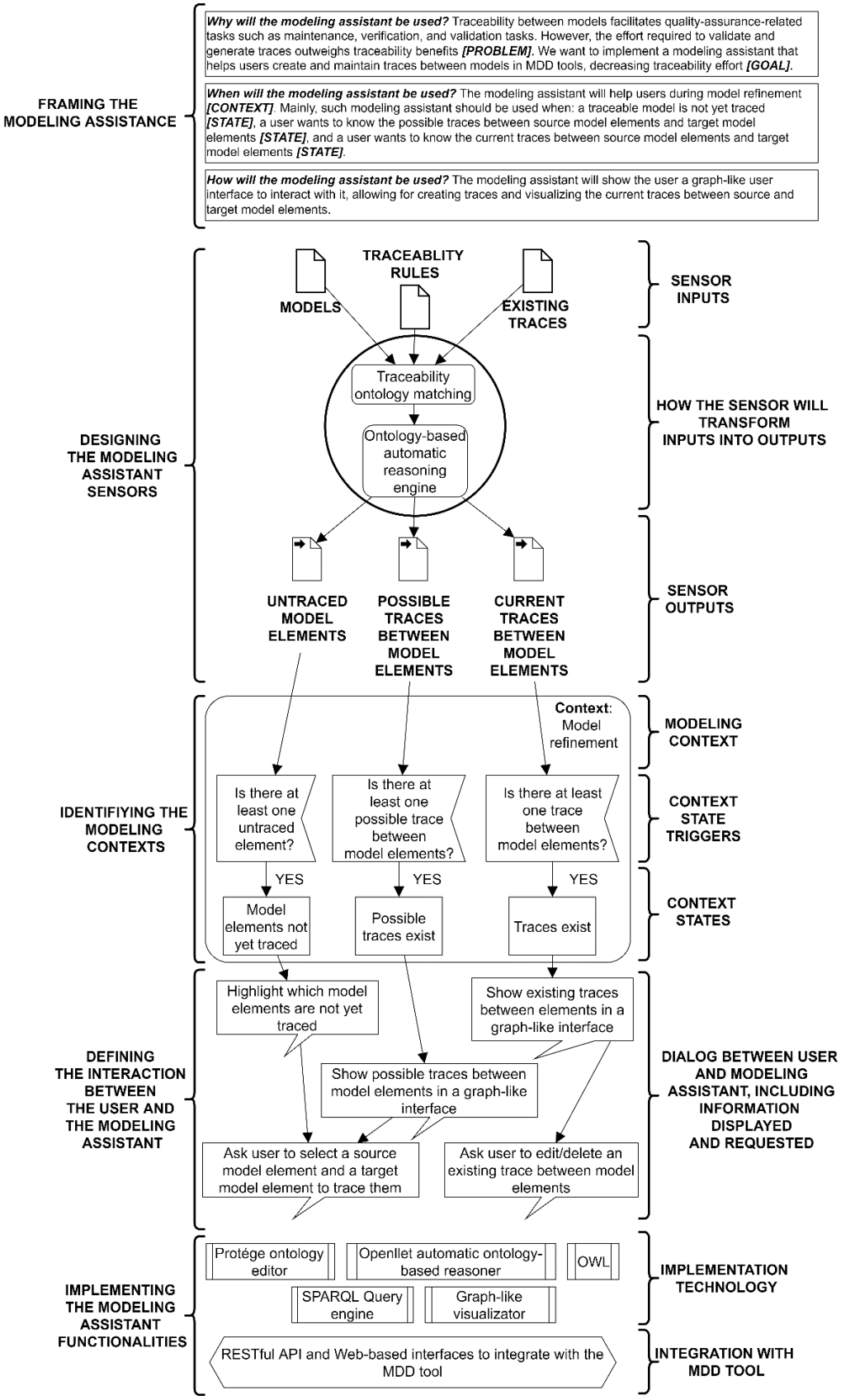

Figure 3: Modeling assistant overview implemented by using MERLIN.

5. Conclusions and further steps

Modeling assistants are a cornerstone to achieve what MDD tools promise as benefits compared to other development approaches. Because of that, several researchers have proposed modeling assistants [8]–[19] to help users create and refine models in MDD tools. However, some proposed modeling assistants lack awareness of modeling context, hindering the user experience with the MDD tool [7]. On the other hand, some authors propose context-aware modeling assistants using domain- and modeling-task-dependent methods, lacking generality. Therefore, in this Ph.D. Thesis, we propose MERLIN: a MEthod for cReating modeLIng assistaNts in the context of MDD tools. MERLIN is a domain- and modeling-task-independent method that allows model-driven engineers—i.e., who develop MDD tools—to implement context-aware modeling assistants.

In this paper, we summarize the research method, goals, and research questions around the development of this Ph.D. Thesis. We framed our research by using the Design Science method [20]. As a result, we will perform activities related to three tasks: *problem investigation, treatment design, and treatment validation*. Such activities aim to address the proposed goals and research questions. As hypotheses, we expect that modeling assistants implemented by using MERLIN increase the technology acceptance of MDD tools based on the technology acceptance model (TAM) proposed by Moody [23].

We also discussed the progress achieved so far on our research. We have done some research effort related to the *problem investigation* task, conducting a focus group and a systematic mapping. Moreover, we have progressed in designing the MERLIN method during the *treatment design* task. We showed a preliminary design of MERLIN, including a proof of concept with a modeling assistant to help users with model traceability in MDD tools. All this progress has been reported, and we are expecting to be published soon at research conferences and journals. As further steps, we will continue improving MERLIN by applying formal Method Engineering efforts. This will require several iterations of the design cycle. In addition, we will create modeling assistants using MERLIN with industrial partners. Such interaction with industry partners will allow us to acquire new requirements for MERLIN and its tool support and provide us with data to validate our hypotheses.

6. Acknowledgements

This Ph.D. Thesis is supervised by Prof. Oscar Pastor at PROS-VRAIN: Valencian Research Institute for Artificial Intelligence, Universitat Politècnica de València (UPV), Valencia, Spain; and Dr. Marcela Ruiz at Software Systems Research Group, Zürich University of Applied Sciences (ZHAW), Winterthur, Switzerland. This research is fully funded by the ZHAW Institute for Applied Information Technology (InIT), the Innosuisse Flagship SHIFT project, and the ZHAW School of Engineering.

7. References

[1] S. Sendall and W. Kozaczynski, Model transformation: the heart and soul of model-driven software development, IEEE Software, 20.5 (2003): 42–45.

[2] J. I. Panach, S. España, Ó. Dieste, Ó. Pastor, and N. Juristo, In search of evidence for model-driven development claims: An experiment on quality, effort, productivity and satisfaction, Information and Software Technology, 62.1 (2015): 164–186.

[3] Á. Domingo, J. Echeverría, Ó. Pastor, and C. Cetina, Evaluating the Benefits of Model-Driven Development, in: Proceedings of the International Conference on Advanced Information Systems Engineering, CAiSE'20, 2020, pp. 353–367.

[4] P. K. Aggarwal, S. Sharma, Riya, P. Jain, and Anupam, Gaps identification for user experience for model driven engineering, in: Proceedings of the 11th International Conference on Cloud Computing, Data Science and Engineering, 2021, pp. 196–199.

[5] S. Abrahao, F. Bourdeleau, B. Cheng, S. Kokaly, R. Paige, H. Stoerrle, J. Whittle, User Experience for Model-Driven Engineering: Challenges and Future Directions, in: Proceedings of the 20th International Conference on Model Driven Engineering Languages and Systems, MODELS 2017, 2017, pp. 229–236.

[6] A. Bucchiarone, F. Ciccozzi, L. Lambers, A. Perantonio, M. Tichy, M. Tisi, A. Wortmann, V. Zaytsev, What Is the Future of Modeling?, IEEE Software, 38.2 (2021): 119–127.
[7] G. Mussbacher, B. Combemale, J. Kienzle, S. Abrahao, H. Ali, N. Bencomo, M. Burgueño, G. Engels, P. Keankean, J. Jézéquel, T. Kühn, S. Mosser, H. Saharaoui, E. Syriani, D. Varró, M. Weyssow, Opportunities in intelligent modeling assistance, Software and Systems Modeling, 19.5 (2020): 1045–1053.
[8] A. Paz, G. el Boussaidi, and H. Mili, checsdm : A Method for Ensuring Consistency in Heterogeneous Safety-Critical System Design, IEEE Transactions on Software Engineering, 47.12 (2021): 2713–2739.
[9] T. Kehrer, U. Kelter, and G. Taentzer, A rule-based approach to the semantic lifting of model differences in the context of model versioning, in: Proceedings of the 26th IEEE/ACM International Conference on Automated Software Engineering, ASE2011, 2011, pp. 163–172.
[10] C. Wang and A. Cavarra, Checking Model Consistency Using Data-Flow Testing, in: Proceedings of the 16th Asia-Pacific Software Engineering Conference, ASPEC2009, 2009, pp. 414–421.
[11] P. Pourali and J. M. Atlee, A Focus+Context Approach to Alleviate Cognitive Challenges of Editing and Debugging UML Models, in: Proceedings of the 22nd International Conference on Model Driven Engineering Languages and Systems, MODELS, 2019, pp. 183–193.
[12] G. Cabral and A. Sampaio, Automated formal specification generation and refinement from requirement documents, Journal of the Brazilian Computer Society, 14.1 (2008): 87–106.
[13] M. Savary-Leblanc, Improving MBSE Tools UX with AI-Empowered Software Assistants, in: Proceedings of the 22nd International Conference on Model Driven Engineering Languages and Systems Companion, MODELS-C, 2019, pp. 648–652.
[14] F. Steimann and B. Ulke, Generic Model Assist, in: LNCS, vol. 8107, 2013, pp. 18–34.
[15] M. Ohrndorf, C. Pietsch, U. Kelter, L. Grunske, and T. Kehrer, History-based Model Repair Recommendations, ACM Transactions on Software Engineering and Methodology, 30.2 (2021): 1–46.
[16] H. Agt-Rickauer, R.-D. Kutsche, and H. Sack, Automated Recommendation of Related Model Elements for Domain Models, in: Communications in Computer and Information Science, 2019, pp. 134–158.
[17] I. ben Fraj, Y. BenDaly Hlaoui, and L. BenAyed, A reactive system for specifying and running flexible cloud service business processes based on machine learning, in Proceedings of the 45th Annual Computers, Software, and Applications Conference, COMPSAC, 2021, pp. 1483–1489.
[18] W. Shen, K. Wang, and A. Egyed, An Efficient and Scalable Approach to Correct Class Model Refinement, IEEE Transactions on Software Engineering, 35.4 (2009): 515–533.
[19] H. M. Chavez, W. Shen, R. B. France, B. A. Mechling, and G. Li, An Approach to Checking Consistency between UML Class Model and Its Java Implementation, IEEE Transactions on Software Engineering, 42.4 (2016): 322–344.
[20] R. J. Wieringa, Design Science Methodology for Information Systems and Software Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
[21] D. L. Moody, The Method Evaluation Model: A Theoretical Model for Validating Information Systems Design Methods, in: ECIS 2003 Proceedings, 2003, pp. 79–96.
[22] B. Djoudi, C. Bouanaka, and N. Zeghib, A formal framework for context-aware systems specification and verification, Journal of Systems and Software, 122.1 (2016): 445–462.