Collaborative Model-Driven Software Engineering: a Classification Framework and a Research Map

Mirco Franzago, Davide Di Ruscio, Ivano Malavolta and Henry Muccini

Abstract—*Context:* Collaborative Model-Driven Software Engineering (MDSE) consists of methods and techniques where multiple stakeholders manage, collaborate, and are aware of each others' work on shared models.
Objective: Collaborative MDSE is attracting research efforts from different areas, resulting in a variegated scientific body of knowledge. This study aims at identifying, classifying, and understanding existing collaborative MDSE approaches.
Method: We designed and conducted a systematic mapping study. Starting from over 3,000 potentially relevant studies, we applied a rigorous selection procedure resulting in 106 selected papers, further clustered into 48 primary studies along a time span of 19 years. We rigorously defined and applied a classification framework and extracted key information from each selected study for subsequent analysis.
Results: Our analysis revealed the following main findings: (i) there is a growing scientific interest on collaborative MDSE in the last years; (ii) multi-view modeling, validation support, reuse, and branching are more rarely covered with respect to other aspects about collaborative MDSE; (iii) different primary studies focus differently on individual dimensions of collaborative MDSE (i.e., model management, collaboration, and communication); (iv) most approaches are language-specific, with a prominence of UML-based approaches; (v) few approaches support the interplay between synchronous and asynchronous collaboration.
Conclusion: This study gives a solid foundation for classifying existing and future approaches for collaborative MDSE. Researchers and practitioners can use our results for identifying existing research/technical gaps to attack, better scoping their own contributions, or understanding existing ones.

Index Terms—Model-Driven Engineering, Collaborative Software Engineering, Systematic Mapping study.

1 Introduction

Collaborative software engineering (CoSE) deals with methods, processes and tools for enhancing *collaboration, communication, and co-ordination (3C)* among team members [1]. The importance of CoSE is evident [2]–[4] and recently empowered by the prominence of agile methods, open-source software projects, and global software development [1]. CoSE is not only about software development team members, but it also embraces external and non-technical stakeholders, like customers and final users, as advised by current research on participatory design methods [5], [6].

CoSE can be applied to different artifacts towards the engineering of software systems. When focusing on software design, considered to be one of the key aspects of software engineering [7], multiple stakeholders with different technical knowledge and background collaborate on the system design [8]. In this context, shared *models* are used as *"a reduced representation of some system that highlights the properties of interest from a given viewpoint"* [9]. Models allow each stakeholder to focus on domain-specific concepts, to abstract upon the aspects of the system in which she is more expert, and to assess specific properties of the system early in the life cycle. A model is a specific design artifact that can be either graphical, XML-based, or textual, and can have an unambiguously defined semantics, which allows precise information exchange and many additional usages. By systematically using models as first-class entities for describing specific aspects of a software system, Model-Driven Software Engineering (MDSE) provides suitable engines for defining, analyzing, and manipulating those models throughout the system development life cycle, including syntactical validation, model analysis, model simulation, model transformations, model execution, and model debugging [4].

This work focuses on specializing CoSE to MDSE. In the context of this work, our definition of **collaborative MDSE approach** is a *method or technique in which multiple stakeholders manage, collaborate, and are aware of each others' work on a set of shared models*. A collaborative MDSE approach is composed of three main complementary dimensions (the way those dimensions have been elicited is carefully discussed in Section 3): a *model management* infrastructure for managing the life cycle of the models, a set of *collaboration* means for allowing involved stakeholders to work on the modelling artifacts collaboratively, and a set of *communication* means for allowing involved stakeholders to be aware of the activities of the other stakeholders, exchange messages and information within the team, sharing (design) decisions, reducing potential ambiguities, and more.

Collaborative MDSE is gaining a growing interest in both academia and practice [3], [4]. A number of *research projects* are being run to enable large teams of modelers to construct and refine large models in a collaborative manner [3]. For instance, the Dawn Eclipse project[1] is investigating collabo-

- *M. Franzago, D. Di Ruscio and H. Muccini are with the Department of Information Engineering, Computer Science and Mathematics (DISIM), University of L'Aquila, Italy.*
  E-mail: {mirco.franzago, davide.diruscio, henry.muccini}@univaq.it
- *I. Malavolta is with the Department of Computer Science, Vrije Universiteit Amsterdam, The Netherlands.*
  E-mail: i.malavolta@vu.nl

Manuscript received Month XX, 20XX; revised Month XX, 20XX.

1. http://wiki.eclipse.org/Dawn

rative UIs to provide collaborative access for GMF diagrams; EMFCollab[2] is a light-weight solution to let multiple users edit a single EMF model concurrently. The Modeling Team Framework (MTF)[3] open source project under the Eclipse Modeling Framework Technology Project (EMFT) provides a mechanism like a meta repository for software configuration management of Eclipse projects. GenMyModel[4] supports the design and sharing of design diagrams in real-time: the metamodel-compliance of the diagrams is always ensured, there are no conflict resolution for the end-users and no lock-unlock barriers. UNICASE[5] is an open-source project that, based on the Eclipse platform, offers a unified repository for software engineering projects and specific views on this model for project participants like architects, project manager or developers. Collaboro[6] is an open-source project allowing both developers and users to participate together to the creation and evolution of domain-specific modeling languages [10]. MDEForge [11] is a web-based modeling platform realizing an extensible and community-based repository of modeling artifacts; MDEForge enables the development, analysis and reuse of modeling artifacts by adopting a software-as-a-service paradigm[7].

A body of knowledge in the *scientific literature* about collaborative model-driven software engineering (MDSE) exists as well (e.g., [12]–[19]). Such studies are scattered across different independent research areas, such as software engineering, model-driven engineering, languages and systems, and model integrated computing. However, *a holistic view on what Collaborative MDSE is, its components, and challenges in collaborative MDSE* is still missing.

Goal of this study is to identify, classify, and understand approaches that support collaborative MDSE. We focus on those approaches in which several distributed technical and/or non-technical stakeholders collaborate to produce and manage models of a software system, working in a shared environment, either synchronously or asynchronously. Stakeholders can include, but are not limited to, technical actors (modelers, designers, developers), domain experts, non-technical managers, customers, and users of the software system. We are interested in identifying and analyzing the different approaches to support multi-user modeling tasks where the models can be either domain-specific or domain-independent. In any case, studied approaches must consider the models as first-class elements within the whole software process. Also, studied approaches must provide synchronization mechanisms, e.g. conflicts management/resolution, conflicts avoidance, versioning and roll-back support.

In order to tackle our goal we apply a well-established methodology from the medical and software engineering research communities called **systematic mapping study** (SMS) [20], [21]. Following known SMS protocols, we have scrutinized more than 3,000 research articles. 48 of them have been selected after a careful analysis based on rigorously-defined inclusion and exclusion criteria, and deeply analyzed.

The **main contributions** of this study are:

- the definition of the *complementary dimensions* of collaborative MDSE: model management, collaboration, and communication;
- the elicitation of a *taxonomy* of collaborative MDSE, used to create a reusable *classification framework* for understanding, classifying, and comparing present and future work on collaborative MDSE. The framework has been realized by following an incremental process that, starting from the main definition provided above, elaborated on each individual parameter to identify all possible characteristics of collaborative MDSE;
- the identification of current characteristics, challenges and shortcomings, and publication trends with respect to collaborative MDSE approaches.

To the best of our knowledge, this paper presents the first systematic investigation into the state of the art of research on collaborative MDSE. The results of this study provide a complete, comprehensive and replicable picture of the state of the art of research on collaborative MDSE, helping researchers and practitioners in finding characteristics, limitations, and gaps of current research on the topic.

Article outline. This article is structured as follows. Section 2 provides the needed background on Model-Driven Software Engineering, Collaborative Software Engineering, and Collaborative MDSE, and motivates the need for this study. Section 3 presents collaborative MDSE and its dimensions: model management, collaboration, and communication, whereas Section 4 presents the design of our study. Sections 5, 6, 7, 8, 9, and 10 elaborate on the obtained results. Section 10 presents results orthogonal to the research questions, whereas an overall discussion is presented in Section 11. Threats to validity are analyzed in Section 12. Section 13 closes the paper and discusses future work.

2 Background

In this section we introduce main concepts that found the basis of Collaborative MDSE. More specifically, we present model-driven software engineering and collaborative software engineering in Section 2.1 and Section 2.2, respectively. Systematic studies with different scopes but related to collaborative MDSE are discussed in Section 2.3. The need for our systematic mapping study on collaborative MDSE is motivated in Section 2.4.

2.1 Model-Driven Software Engineering

Model-Driven Software Engineering (MDSE) refers to the systematic use of models as first-class entities throughout the software engineering life cycle. Model-driven approaches shift development focus from third-generation programming language codes to models expressed in proper domain-specific modeling languages [22]. The objective is to increase productivity and reduce time to market by enabling the development of complex systems by means of models defined with concepts that are much less bound to the underlying implementation technology

2. http://qgears.com/products/emfcollab/
3. http://www.eclipse.org/proposals/mtf/
4. http://www.genmymodel.com/
5. http://marketplace.eclipse.org/content/unicase
6. http://som-research.github.io/collaboro
7. http://www.mdeforge.org/

and are much closer to the problem domain. This makes the models easier to specify, understand, and maintain [9] helping the understanding of complex problems and their potential solutions through abstractions. In MDSE, models are expressed in terms of concepts formalized in *metamodels*, and *model transformations* are employed either to produce target models or to generate textual artifacts. Model transformations can be employed for different reasons e.g., to enable interoperability of different tools, to take advantage of analysis tools provided by the target language, and to generate various artifacts related to the modeled systems (such as source code, configuration files, and documentation). Models are specified by means of editors providing modelers with textual or graphical constructs, and facilities for checking the validity of the specified models with respect to the corresponding metamodels.

The relevance of MDSE is evident by the increasing interest in scientific challenges in the area, active technology projects (e.g., the Eclipse Modeling Project[8]) and numerous industrial projects ranging from direct applications of MDSE concepts and tools, to those developing its foundations. There are many success stories of MDSE, ranging from the development of large-scale enterprise Web applications, to clinical data management, and to public authority data interchange [23]–[31].

Even though current modeling tools provide features that can simplify and automate many steps of model-based development, obstacles to the wider adoption of MDSE technologies still exist [32]. One limiting factor is related to the limited support for collaborative MDSE [11], which focuses on the need for consistent reuse of modeling artifacts that are collaboratively produced and consumed during the different development phases.

2.2 Collaborative Software Engineering

As quoted from [2, p. 3], *"[...] any software project with more than one person is created through a process of collaborative software engineering"*. The importance of CoSE is evidenced in a number of sources [2]–[4]. Its growth is strongly associated to the prominence of outsourcing, open source software projects, global software engineering processes, and distributed agile methods [2, ch.1 & ch.19]. If we add that 86% of the (architectural) design decisions are group based [33], where each group comprises from five to ten members [34], the role of collaborative software engineering becomes easily clear.

Orthogonally to the *collaboration, communication, and coordination (3C)* dimensions [1], the three key insights for CoSE are remarked in [2, ch. 1]: software engineering collaboration is model-based, software project management creates the organizational structure under which collaboration is fostered, and global software engineering challenges increase collaboration complexity. Very importantly for this study is the first key insight highlighting the model-based nature of CoSE. Much collaboration in software engineering is related to software-related artifacts (such as, UML diagrams, source code, and bug reports), therefore, most of the collaboration in software engineering is over a set of formal or semi-formal artifacts co-workers collaborate upon.

8. https://projects.eclipse.org/projects/modeling

Stakeholders, with different roles and concerns (ranging from technical to external), collaborate on the creation of different models each one representing the system from a different perspective [35]. Stakeholders collaborate over diverse organizational structures [36], [37] enforced by modern organizational practices, such as open-source, outsource, multi-site and agile software. Software developers work is organized around networks, communities, groups and they are exposed to one or more organizational barriers, i.e., impediments, social, organizational or otherwise that hinder the harmonious operation across the organizational social structure [36]. Knowledge must be produced and exchanged among and across teams, exacerbating the need of knowledge definition (through agreed ontologies), knowledge exchange (based on agreed protocols) and awareness, and coordination.

The *global* dimension of software engineering adds further complexities, such as geographical, temporal, cultural and linguistic distance. Model-based collaboration tools, process support tools, awareness tools, and collaboration infrastructures have been developed for improving the collaboration, communication, and co-ordination between and among teams [38].

Building a theoretical understanding of CoSE is still considered a challenge today [1]. This is a further motivation for this work.

2.3 Existing Systematic Studies related to Collaborative MDSE

In this section we discuss other existing systematic studies (literature review (SLR) and mapping study (SMS)) related to this work. Based on our knowledge and after a manual search, we did not find any existing systematic study on the topic. In any case, in the following we report those studies that, even if they have different scopes and objectives, can be related to our research.

Table 1 shows the existing systematic studies, their specific focus, and quality assessment. Based on the criteria explained in [39], we calculate the total score of each study by summing up the answer to each of the following questions (Yes(Y)=1, Partly(P)=0.5, No(N)=0)[9]

- Q1 Are the systematic study inclusion and exclusion criteria described and appropriate?
- Q2 Is the literature search likely to have covered all relevant studies?
- Q3 Did the reviewers assess the quality/validity of the included studies?
- Q4 Were the basic data/studies adequately described?

A systematic literature review on the models of collaboration in the domain of distributed software development (DSD) is presented in [40]. This study focuses on the models and tools for DSD *based on life cycle of traditional software development (and its variations), where each phase of the cycle is performed*. Differently, in our study we focus on the various aspects of the collaboration in the model-driven software engineering domain, where the models are placed as first-class artifacts.

9. Note that the score reflects how well the empirical study has been conducted, rather than the tightness of the related study to this paper.

TABLE 1: Existing systematic studies on collaborative software engineering and collaborative modeling

Study	Year	Q1	Q2	Q3	Q4	Total	Focus
[40]	2011	Y	Y	Y	Y	4.0	Ways of collaboration used in DSD
[41]	2012	Y	P	P	Y	3.0	Tools supporting distributed teams in GSE
[42]	2008	P	Y	P	P	2.5	Collaborative modeling support systems

In [41] a systematic mapping study is proposed with the objective to discover all the tools available in the literature supporting global software engineering (GSE) activities. Our work is more comprehensive because it identifies (in addition to tool support) also methods, techniques and approaches to support modeling activities in collaborative settings with the focus on model-based activities. Moreover, in the literature there are other systematic studies in the DSD and GSE scope [43], but all of them focus on tools and/or approaches to address issues like collaboration process management, team members awareness or collaboration tools support; there is no existing study specifically focusing on collaborative MDSE.

Finally, in [42] the authors identify challenges and best practices in collaborative modeling activity, where modelers, end-users and experts are all involved in the model-based design of the system, and collaborate to create a shared understanding of the system under development (or part of it). Among others, the main difference between this study and ours is that in [42] collaborative modeling is considered as *the joint creation of a shared graphical representation of a system*, i.e., a sketching activity where the created models are used as communication means between team members. This kind of models (possibly conforming to some syntactical rules) are very different from our definition of models, where modeling is a complex activity based on precise models whose semantics is rigorously defined according to a specific modeling language. These models allow precise information exchange but also many additional usages, including: syntactical validation, model checking, model transformation, code generation [4, § 2.1].

2.4 The need for a Systematic Mapping Study on Collaborative MDSE

This research complements the existing studies described in Section 2.3 to investigate the state-of-research about collaborative MDSE. So far, a large body of knowledge has been proposed in both modeling software systems (e.g., model-driven engineering techniques, domain-specific modeling languages, model transformations, etc.), and collaboration for software production (e.g., global software engineering methods, methods for participatory design of software systems, version control systems, etc.).

Even if the progress of research on the above mentioned areas has started more than a decade ago and the various research communities are still very active, we did not find any evidence that could help us in assessing the impact of existing research on collaborative MDSE. Thus, in this study we aim to identify, classify, and understand existing research on collaborative MDSE. Those activities will help researchers and practitioners in identifying trends, limitations, and gaps of current research on collaborative MDSE and its future potential.

3 Collaborative Model-Driven Software Engineering

In Sections 2.1 and 2.2 we outlined the main concepts of MDSE and collaborative software development, respectively. In this section we go a step further and, in order to avoid a potential threat to the external validity of our study, we define a reference definition for characterizing collaborative approaches in the context of MDSE. In order to provide an objective definition of collaborative MDSE, we rigorously applied the following process:

1) we analyzed a set of studies about MDSE approaches with a strong focus on collaboration;
2) we performed an investigation on existing literature about collaborative approaches for software engineering in general;
3) based on the analyzed papers, we produced a tentative definition of collaborative MDSE;
4) we asked to a pool of MDSE and global software engineering experts to objectively assess the soundness of the obtained definition and we refined the definition according to the feedback provided by the experts[10].

Table 2 presents the set of pilot studies on MDSE approaches with a strong focus on collaboration; we identified them by performing a preliminary screening on the available literature about collaborative MDSE. For what concerns the second point of our semi-systematic process, in order to avoid a potential bias with respect to the construct validity of our study, we decided not to restrict our focus on MDSE-based approaches only. Indeed, in this step we considered relevant papers on collaborative and global software engineering which contribute with definitions of types of tools, requirements, building blocks, and various categories for characterizing approaches for collaborative and global software engineering. Table 3 presents the set of studies we identified in this step.

As a result of the above mentioned process, in the context of this study a **collaborative MDSE approach** can be defined as a *method or technique in which multiple stakeholders manage, collaborate, and are aware of each others' work on a set of shared models*. As shown in Figure 1 [46], a collaborative MDSE approach considers models as first-class elements that drive both the software development activities and the other model-based tasks in the context of a software engineering process [4], and is composed of three main complementary dimensions:

- a *model management* infrastructure for managing the life cycle of the models; such an infrastructure may

10. We performed this step (i) by directly discussing collaborative MDSE with personal contacts and (ii) during the 2016 edition of a workshop dedicated to collaborative modelling in MDE, co-located with the MODELS conference.

TABLE 2: Selected studies about MDSE approaches with a strong focus on collaboration

Study	Title	Year
Maróti et al. [12]	Next Generation (Meta) Modeling: Web-and Cloud-based Collaborative Tool Infrastructure	2014
Syriani et al. [13]	AToMPM: A Web-based Modeling Environment	2013
Farwick et al. [14]	A web-based collaborative metamodeling environment with secure remote model access	2010
Thum et al. [15]	SLIM - A Lightweight Environment for Synchronous Collaborative Modeling	2009
Cataldo et al. [16]	CAMEL: a tool for collaborative distributed software design	2009
Bruegge et al. [17]	Unicase- an Ecosystem for Unified Software Engineering Research Tools	2008
De Lucia et al. [18]	Enhancing collaborative synchronous UML modelling with fine-grained versioning of software artifacts	2007
Kelly et al. [19]	Metaedit+: a fully configurable multi-user and multi-tool case and came environment	1996

TABLE 3: Selected studies about collaborative and global software engineering

Study	Title	Year
Dullemond et al. [44]	Collaboration Spaces for Virtual Software Teams	2014
Beecham et al. [41]	Tools used in Global Software Engineering: A systematic mapping review	2012
Lanubile et al. [45]	Collaboration Tools for Global Software Engineering	2010
Whitehead [38]	Collaboration in Software Engineering: A Roadmap	2007

contain a (possibly distributed) repository for managing the persistence of the models and their related metadata, a modelling tool [4] for creating, editing, or deleting models, interchange formats for sharing models across projects and organizations (see Section 5);

- a set of *collaboration* means for allowing involved stakeholders to work on the modelling artifacts collaboratively and to coordinate themselves as a team (e.g., model versioning systems, model merging mechanisms, conflict management and visualization systems, model comparison engines) as discussed in Section 6;
- a set of *communication* means for allowing involved stakeholders to be aware of the activities of the other stakeholders and exchange messages and information within the team. Those activities can be carried out by sharing (design) decisions, tracking discussions among the stakeholders with regard to modeling artifacts, collecting relevant information about various aspects of the project (e.g., activities dashboard, general notifications of activities, issue trackers, asynchronous messaging systems, chat, social networks, wiki) as discussed in Section 7.

An orthogonal aspect related to the previous dimensions is *automation* i.e., *the capability of partially or fully automating software engineering activities with the aim of increasing both quality and productivity* [47]. Automation is provided by tools and classifying them in the context of collaborative MDSE represents an important contribution, which deserves to be investigated in a separated work already planned as future work.

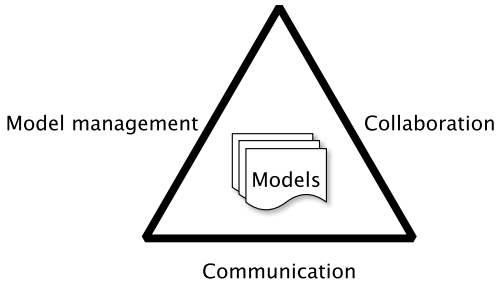

Fig. 1: Complementary dimensions of collaborative MDSE

It is important to note that a common trait across the above mentioned dimensions is that in all of them (i) stakeholders work as a team (so our definition does not focus on those approaches involving a single stakeholder) and (ii) the focus is on the software-intensive aspect of the system being modeled, meaning that the activities related to model management, collaboration, and communication are performed in the context of more large-grained software engineering activities like software design, requirements elaboration, system validation, testing, evolution and maintenance.

We consider such a definition of collaborative MDSE approach throughout our investigation, especially during the studies selection, data extraction, and data analysis activities.

Example. In the following we describe WebGME, the first study of Table 2, in order to give a concrete idea about the typical traits and features of a collaborative MDSE approach. WebGME is a web-based infrastructure for supporting the collaborative modelling, analysis, and synthesis of complex, large-scale information systems [12], [48].

For what concerns *model management*, WebGME is modelling language independent and allows stakeholders to create their own domain-specific modelling language by means of a web-based editor, which permits to specify the metamodel of the new language and the concrete representations of its metaclasses; starting from the created metamodel, WebGME automatically configures itself to support the newly specified language at the modelling level. Models are edited using web-based clients running in the stakeholders' browsers so that no installation or configuration of specific clients is needed, and the editors are platform independent; those clients support a number of different visualization techniques for the models. A set of multiple APIs are provided to interface WebGME with external tools, custom domain-specific visualization components, and external code generation engines. Also, the infrastructure provides extension points to customize it with plugins, language decorators, visualizers, or new elements in the user interface.

Collaboration is performed at the modelling level in real-time (i.e., all changes are immediately propagated to all the involved stakeholders, similar to how Google Docs works).

WebGME supports model versioning and conflict resolution via a lightweight branching scheme, which is transparently supported by the infrastructure: through WebGME model branching system each stakeholder can monitor in real-time information about the activity of other stakeholders (i.e. the history of actions, commits, timestamps, etc. performed by other users). WebGME also provides a mechanism for managing cross-cutting concerns that allows stakeholders to collaboratively visualize and modify associations among elements belonging to different models.

From the *communication* perspective, WebGME does not provide built-in nor external communication means. The users can not directly communicate, and the workspace awareness is allowed through the branch history actions' log; also, the WebGME web editor shows in real-time (i) if the currently edited model is in sync with respect to the models edited by the team, (ii) if the client editor is currently connected to the network, and (iii) the specific branch in which the stakeholder is currently working on.

4 Research Method

This study has been carried out by following the guidelines for performing systematic mapping studies [20], [21], [49]. The whole study can be divided into three main phases: *planning*, *conducting*, and *documenting*. Each phase has been performed by one or more members of the research team of this study (see Appendix A). In order to mitigate potential threats to validity and possible biases, the artifacts produced during each phase of the study have been circulated to external experts for independent review. More specifically, we identified two classes of external experts: SLR experts and domain experts. SLR experts are contacted for getting feedback about our review protocol, possible unidentified threats to validity, possible problems in the overall construction of our study; whereas, domain experts are contacted for getting feedback about whether our review protocol and final reports can be effective with respect to the object of our mapping study (i.e., collaborative MDSE approaches). All the external reviewers provided their feedback and their suggestions have been implemented in the study.

To allow replication and verification of our study, a complete replication package[11] is publicly available to interested researchers. It includes a description of the review protocol, the list of all considered and selected studies, the description of the parameters for the data extraction activity (data extraction form), raw data, and the R scripts for data analysis. In the following we go through each phase of the process

Planning. It aims at (i) establishing the need for performing a mapping study on collaborative MDSE (see Sections 2.3 and 2.4), (ii) identifying the main research questions (see Section 4.1), and (iii) defining the protocol to be followed by the involved researchers while carrying on each step of the whole review process. The output of our planning phase is a well-defined review protocol [50], that underwent an external evaluation by the previously mentioned external reviewers.

11. http://www.di.univaq.it/mirco.franzago/collaborativeMDSE/

Conducting. In this phase we set the previously defined protocol into practice. More specifically, we performed the following activities:

- *Search and selection*: we (i) considered the search strings identified in Section 4.2.1 and we applied them to electronic data sources, and (ii) apply backward and forward snowballing techniques for expanding the set of considered studies [51]. Then, the potentially relevant studies have been filtered in order to obtain the final list of primary studies to be analyzed (for the complete list of primary studies see after the Section References). Sections 4.2.1 and 4.2.2 describe in details the search and selection strategy of this research.
- *Classification framework definition*: we defined the set of parameters to compare the primary studies and for collecting the information needed for analyzing the primary studies. The design of the classification framework is based on the research questions [49].
- *Data extraction*: in this activity we went into the details of each primary study, and we filled a corresponding data extraction form, as defined in the previous activity. Filled forms have been collected and aggregated in order to be ready to be analyzed during the next activity. More details about this activity are presented in Section 4.3.
- *Data synthesis*: this activity focused on a comprehensive summary and analysis of the data extracted in the previous activity. The main goal of this activity is to elaborate on the extracted data in order to address each research question of our study (see Section 4.1). The details about this activity are in Section 4.4.

Documenting. According to the guidelines provided in [20], the main activities performed in this phase have been: (i) a thorough elaboration on the data extracted in the previous phase with the main aim of setting the obtained results in their context, (ii) a thorough analysis and discussion of possible threats to validity (see Section 12), and (iii) the writing of a report describing the performed mapping study; the produced report has been evaluated by the previously mentioned external reviewers and is the basis of this paper.

4.1 Research Questions

We formulate the goal of this research by using the Goal-Question-Metric perspectives (i.e., purpose, issue, object, viewpoint [52]). Table 4 shows the result of the above mentioned formulation.

TABLE 4: Goal of this research

Purpose	Identify, classify, and understand
Issue	the publication trends, characteristics, and challenges
Object	of existing collaborative MDSE approaches
Viewpoint	from a researcher's viewpoint.

In the following we present the research questions we translated from the above mentioned overall goal. For each research question we also provide its primary objective of investigation.

- **RQ1**: *What are the characteristics of collaborative MDSE approaches?* This research question has been decomposed into more detailed sub-questions in order for it to be addressed, sub-questions coming from the three dimensions of collaborative MDSE.
  - **RQ1.1**: *What are the characteristics of the model management infrastructure of existing collaborative MDSE approaches?*
  - **RQ1.2**: *What are the characteristics of the collaboration means of existing collaborative MDSE approaches?*
  - **RQ1.3**: *What are the characteristics of the communication means of existing collaborative MDSE approaches?*

  Objective: to *identify* and *classify* existing collaborative MDSE approaches according to the three dimensions of collaborative MDSE.
  Outcome: a *map* that classifies a set of collaborative MDSE approaches based on different categories (e.g., characteristics of collaborative model editing environments, model versioning mechanisms, model repositories, support for communication and decision making, etc.).
- ***RQ2*** *- What are the challenges and shortcomings of existing collaborative MDSE approaches?*
  Objective: to *identify* current limitations and challenges with respect to the state of the art in collaborative MDSE.
  Outcome: a *map* that classifies collaborative MDSE approaches with respect to their limitations, faced challenges, and future work.
- **RQ3**: *What are the publication trends that can be deduced from the scientific publications about collaborative MDSE approaches over time?*
  Objective: to *identify* and *classify* the interest of researchers in collaborative MDSE approaches and their various characteristics over time.
  Outcome: a *map* that classifies the collected primary studies according to publication year, venue, applied research strategies, etc. Also, the map classifies collected primary studies according to their focus on the various characteristics of collaborative MDSE approaches over time.

The classification resulting from our investigation on RQ1, RQ2, and RQ3 provides a solid foundation for a thorough identification and comparison of existing and future approaches for collaborative MDSE. This contribution is useful for both researchers willing to further contribute with new collaborative MDSE approaches, or willing to better understand or refine existing ones.

The above listed research questions drove the whole study, with a special influence on the studies search and selection, data extraction, and data analysis activities.

4.2 Search and Selection Strategy

When searching among potentially relevant studies it is fundamental to achieve a good trade-off between the coverage of existing research on the considered topic and to have a manageable number of studies to be analyzed. As shown in Figure 2, we designed our search and selection process as a multi-stage process in order to have full control on the number and characteristics of the studies being considered.

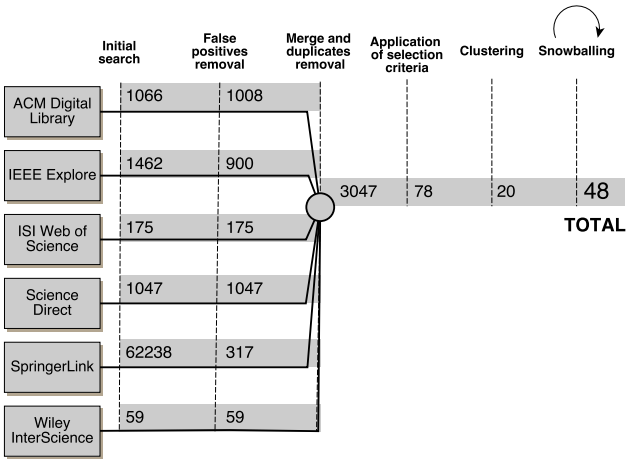

Fig. 2: Search and selection process

4.2.1 Search Strategy

As discussed in [53], an optimum search strategy is expected to provide effective solutions to the following questions: *which*, *where*, *what*, and *when*. In the following we discuss how our search strategy answers each of them.

Which approaches? Our search strategy consists of two main steps: (i) an automatic search on Electronic Data Sources (EDS) and (ii) a snowballing procedure. During the first step, following the guidelines in [20], we composed the search string (Listing 1) based on identified keywords from research questions and area of study (i.e., collaborative MDSE).

As recommended in guidelines for systematic studies, we complemented our automatic search in order to extend the coverage on the topic [21]. For this purpose, we applied a snowballing procedure on the results of the automatic search[12]. The *start set* for the snowballing procedure was composed by the selected papers retrieved by the automatic search.

Our selection criteria (see Section 4.2.2) have been applied to each potentially relevant study (either coming from the automatic search or from the snowballing) and, if a paper was included, snowballing was applied iteratively. The procedure ended when no new papers were found.

Where to search? Figure 2 shows the electronic databases we used for the automatic search. These are the main sources of literature for potentially relevant studies on software engineering [54]. Also, these EDSs have been selected from the recommendations made by experts in the area of software engineering. We do not use Google Scholar since it may generate many irrelevant results and have considerable overlap with ACM and IEEE on software engineering literature [54]; nevertheless, we used Google Scholar in the forward snowballing procedure [51].

12. Snowballing refers to using the reference list of a paper (backward snowballing) or the citations to the paper (forward snowballing) to identify additional papers [51].

What to search? A suitable search string is the input to the EDSs identified in the previous section. According to the guidelines provided in [20], we used the following systematic strategy for constructing our search string:

1) derive major aspects relevant to the study, according to the research questions and to a set of relevant *pilot* studies (Table 2 shows the considered pilots). Each aspect is represented by a "cluster" that groups a set of terms; identified clusters are *collaborative* and *MDSE*;
2) add keywords (main terms) to each cluster obtained from known primary studies and research questions;
3) identify and include in a cluster synonyms and related terms of the main terms;
4) incorporate alternative spellings and synonyms using Boolean *OR*;
5) link the cluster keywords using Boolean *AND*;

Following this strategy, after a series of test executions and refinements the resulting search string is shown in the Listing 1. Each electronic data source has a specific syntax for search strings, so we adapted our generic search string to the specific syntax and criteria of each electronic data source.

Listing 1: Composed search string

```
(collaborat* OR coordinat* OR cooperat* OR
concur* OR global)
AND
(MDE OR MDD OR MDA OR MDS* OR EMF OR DSL OR
DSML OR "model driven" OR "eclipse modeling
framework" OR "domain specific language" OR
"domain specific modeling language")
```

Due to the nature of the involved electronic databases and indexing systems, search results could include also elements that were clearly not research papers, such as conference and workshop proceedings, standards specifications, textbooks, editorials, etc. After the initial search, we manually removed all those *false positive* results in order to have a coherent set of potentially relevant research studies [13]. By referring to Figure 2, after merging all the studies and removing duplicates we obtained 3,047 potentially relevant studies.

When and what time span to search? We included in our search *all* the studies coming from the selection step, avoiding publication year constraints, so we did not consider publication year as criterion for the search and selection steps.

4.2.2 Selection Strategy

As suggested in [20], we decided the selection criteria of this study during its protocol definition, to reduce the likelihood of bias. In the following we provide inclusion (I) and exclusion (E) criteria of our study:

- I1 - Studies proposing an MDSE method or technique for supporting the collaborative work of multiple stakeholders on models. The study, to be included, has to cover the three complementary dimensions of model management/collaboration means/communication means.
- I2 - Studies in which models are the primary artifacts within the collaboration process.
- I3 - Studies providing some kind of validation or evaluation of the proposed method or technique (e.g., via a case study, a survey, experiment, exploitation in industry, formal analysis, example usage).
- I4 - Studies subject to peer review [49] (e.g., journal papers and papers published as part of conference proceedings will be considered, whereas white papers will be discarded).
- I5 - Studies written in English language and available in full-text.
- E1 - Studies discussing *only* business processes and collaboration practices, without proposing a specific method or technique.
- E2 - Secondary studies (e.g., systematic literature reviews, surveys, etc.).
- E3 - Studies in the form of tutorial papers, long abstract papers, poster papers, editorials, because they do not provide enough information.

Inclusion/exclusion criteria should be aligned with the research questions: indeed, all the criteria together aim to guide the answering process to the research questions [55]. As in any systematic literature study, the definition of inclusion/exclusion criteria has been guided by 2 main drivers: (i) keeping the focus of the selected papers on the scope of the study (i.e., "semantic" criteria), and (ii) avoiding grey or not scientific works containing many relevant keywords, but with no relevance for the body of knowledge advancement of the topic (i.e., "syntactic" criteria). Specifically, in this work, I1, I2 and E1 are the semantic criteria: they are strongly related to RQ1 (RQ1.1, RQ1.2, RQ 1.3); the other criteria are syntactic: they are not related to a specific RQ, instead they are important to keep the selected papers scientifically relevant for our study.

In this context, each study has been included as primary study if it satisfied *all* inclusion criteria, and it has been discarded if it met *any* exclusion criterion. The definition of the above mentioned criteria has been tested by considering the pilot studies (Table 2); the criteria have been incrementally refined until they were covering all the pilot studies. On the 3,047 potentially relevant papers, we performed a first manual step applying the selection criteria on title and abstract of the papers, obtaining 160 papers (it took 120 man-hours). On these 160 we performed a comprehensive second manual step reading the whole papers full-text (title, abstract, keywords, sections and appendices, if any (this activity took 100 man-hours). After the application of the above mentioned selection criteria we obtained a set of 78 potentially relevant studies. It is important to note that from the 3,047 potentially relevant studies we obtained only 78

13. The false positive removal process was completely manual, except for SpringerLink due to its internal querying algorithm. SpringerLink, at the time of our search, did not allow to exclude paper references from the search metadata, bringing to thousands of unmanageable false positive results (62,238 in our case). We overcame this problem by implementing a script that was able to locally apply the search string (see Listing 1) on the previously downloaded 62,238 results. Thus, we automatically selected 389 works that we subsequently refined manually by obtaining 317 results for the following steps.

primary studies because on the one hand (i) our search string is quite inclusive (e.g., papers with the terms concurrency and MDE in the abstract actually matched the search string) in order to do not loose any potentially relevant paper but, on the other hand, (ii) we carefully applied all selection criteria, where I1, I2, and E1 allowed us to consider only those studies that were really falling within the scope of our research.

Most of the potentially relevant studies (≈90%) were excluded by applying the selection criterion I1. This fact is due to two main reasons. Firstly, due to time constraints, as soon as one inclusion criteria was not met, then we directly excluded the currently considered paper; secondly, since I1 is the criterion with the strongest relation to the concept of collaborative MDSE, it cut out the majority of the out-of-scope papers (e.g., approaches unsuitable for the software engineering domain, non-collaborative approaches, specific approaches that do not cover all the three dimensions of C-MDSE, etc.). Also I2 and E1 can be considered "semantic" criteria, they excluded ≈9% of the remaining set of excluded papers. Only the last ≈1% was excluded applying the other criteria. As suggested by [49], two researchers assessed a random sample of the studies. For assessing the objectivity of this phase the inter-researcher agreement has been measured, achieving a promising 0.89 value for the Cohen Kappa statistic[14].

Moreover, we performed a clustering activity on the 78 studies in order to group papers presenting different aspects of the of the same approach. More specifically, if an approach was published in more than one paper (for example, if a conference paper was extended to a journal version), only one instance has been counted as a primary study. In those cases the journal version of the study has been preferred, as it is supposed to be the most complete; nevertheless, both versions have been used in the data extraction phase [49] and in the analysis of the publication trends (RQ3, see Section 9). After this clustering activity we obtained 20 "clusters" of independent primary studies.

Finally, we complemented the results of the automatic search with a snowballing activity starting from the 20 primary studies resulting from the automatic search. The snowballing activity involved a manual screening of 2,345 potentially relevant studies (it took 160 man-hours), leading to 28 additional primary studies, which lead to the final set of 48 primary studies (see Figure 2). Interestingly, we obtained more primary studies from the snowballing activity (i.e., 28 studies) with respect to those obtained from the automatic search (i.e., 20 studies). This phenomenon can be explained by the fact that collaborative MDSE is a relatively new research topic; we noticed that researchers used a very heterogeneous terminology when writing the title, abstract, and keywords of their studies, leading to the fact that our automatic search may have missed some potentially relevant studies. We included the snowballing activity in order to (i) mitigate this potential threat to the validity of our study and (ii) to keep the automatic search manageable, given the available time and resources. As extensively discussed in [51], snowballing is particularly useful for extending automatic searches since new studies almost certainly must cite at least one paper among the previously identified relevant studies. Also, as empirically assessed in [56], regardless of the differences in the actual numbers and figures obtained when applying automatic search and snowballing, similar patterns and conclusions are identified when using those techniques, especially when they are used in combination (like we do in this study).

14. To be successful, the result of the Cohen Kappa statistic must be above or equal to 0.80 [21].

4.3 Data Extraction

The goal of this step is to identify and collect from the selected primary studies (complete list after Section References) the appropriate and relevant information to answer our research questions (see section 4.1). To achieve this goal, we defined a rigorous classification framework to organize the extracted data in a structured manner; the classification framework is composed of a set of concepts, references among concepts and attributes representing the set of data items extracted from each primary study.

The creation of an effective classification framework demands a detailed analysis of the contents of each primary study. In light of this, we followed a systematic process called *keywording* [57] for defining our classification framework so that it took all the primary studies into account [57].

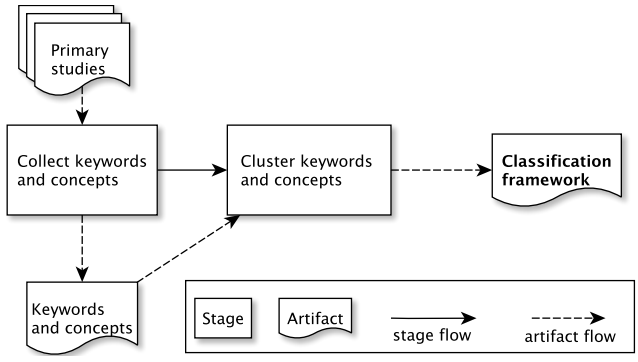

Fig. 3: Overview of the keywording process

As shown in Figure 3, our keywording process is composed of two main steps:

1) *Collect keywords and concepts*: we collected keywords and concepts by reading each primary study. When all primary studies have been analyzed, all keywords and concepts have been combined together to clearly identify the context, nature, and contribution of the research. The output of this stage is the set of keywords extracted from the primary studies.
2) *Cluster keywords and concepts*: when keywords and concepts have been finalized, then we performed a clustering operation on them in order to have a set of representative clusters of keywords. We identified the clusters by applying the open card sorting technique [58] to categorize collaborative MDSE related keywords into relevant groups: each cluster represents one of the aspects under a specific dimension of our classification. After the clusterization step, keywords and concepts within a cluster have been structured in terms of composition, attributes, cardinalities, etc. In order to minimize bias,

this step has been performed by two researchers and the results have been double-checked by the other two researchers. The output of this stage is the finalized classification framework containing all the identified concepts, references, and attributes, each of them representing a specific aspect regarding collaborative MDSE.

In order to make our data extraction strategy more robust, we preliminarily performed a sensitivity analysis to assess whether the results will be consistent independently from the researcher performing the analysis [49]. More specifically, we considered a random sample of 10 primary studies and two authors of this study classified them independently, and each disagreement have been discussed and resolved. Once the extraction strategy has been assessed and set up, the first author of this study read the full text of each primary study and extracted its data according to the classification framework with the support of the other members of the research team (see Appendix A). A detailed data extraction form document as the complete extracted data spreadsheet are available within the replication package at www.di.univaq.it/mirco.franzago/collaborativeMDSE/.

4.4 Data Synthesis

The data synthesis activity involves collating and summarizing the data extracted from the primary studies [21, § 6.5] with the main goal of understanding, analyzing, and classifying current research on collaborative MDSE. We designed and executed our data synthesis activity by following lessons learned and findings presented by Cruzes et al. in [59]. Specifically, our data synthesis is divided into two main phases: vertical analysis, and horizontal analysis.

Vertical analysis. We analyzed the extracted data to find trends and collect information about each research question of our study. In this case, we applied the *line of argument* synthesis [49]: firstly, we individually analyzed each primary study in order to document it and to tabulate its main features with respect to each specific concept of the classification framework; secondly, we analyzed the set of primary studies as a whole in order to reason about potential patterns and trends. We present the results of our vertical analysis by grouping them according to our research questions across Sections 5, 6, and 7, 8, and 9.

Horizontal analysis. In this phase we analyzed the extracted data to explore possible relations across different dimensions and facets of our research. We cross-tabulated and grouped the data and made comparisons between two or more concepts of our classification framework. In this context, we used contingency tables analysis as the strategy for evaluating the actual existence of those relations. In the following we describe the steps we followed for performing the horizontal analysis:

1) Contingency tables construction: we computed a contingency table for every possible pair of concepts within our classification framework[15].
2) Relevant insights formulation: we collaboratively created and discussed a set of 67 potentially relevant insights to be investigated, where each relevant insight is composed of the following elements:
   a) `concepts`: a pair of concepts of the classification framework,
   b) `proposedBy`: the team member proposing the insight,
   c) `hypotheses`: a set of hypotheses to be confirmed by the contingency table,
   d) `table`: a reference to the contingency table between the two paired concepts.
3) Data analysis: we iteratively analyzed each potentially relevant insight created in the previous step in order to check if its contingency table actually confirms or disproves its related hypotheses.
4) Reporting: we reported each result obtained in the previous step as a short paragraph similar to those in Section 10.
5) Filtering: we filtered out all the results which were either (i) not supported by a sufficient number of data points, or (ii) chaotic, not revealing any evident pattern. This filtering step was performed manually and collaboratively by the four co-authors until a full agreement about the inclusion of each pair was reached. More specifically, because of the semantic nature of the pairs composing the selected contingency tables, no fixed rules could have been defined to drive the filtering step (i.e., a specific number of data points or criteria to detect chaotic behavior). Consequently, we analyzed collaboratively table by table searching for interesting findings through a selection based on an inter-researchers agreement.

To clarify how data have been cross-tabulated, it is possible to consider as an explanatory example the case involving the following pair of concepts: $< CollaborationType, VersioningType >$; their corresponding extracted data is reported in Table 14 and Table 16. This case involves only one hypothesis, i.e., that "*approaches without versioning support are equally distributed across approaches supporting synchronous and asynchronous collaboration*". Thanks to the vertical analysis (see Table 14), we already know that approaches without support for versioning are 28; by matching those 28 approaches with those in Table 14, we also know that 22 studies over those 28 support synchronous collaboration, whereas only 6 studies over 28 support asynchronous collaboration, thus falsifying the formulated hypothesis. A complete discussion of the results of the horizontal analysis is presented in Section 10.

5 Model Management (RQ1.1)

In this section we describe the characteristics of collaborative MDSE approaches with respect to the model management facilities provided by the adopted tools. By analyzing the primary studies under such a dimension, a set of representative concepts have been identified as shown in Fig. 4. The first layer of the taxonomy is composed of five elements: supported artifact, modeling language, editor, application domain, and multi-view support. In the following subsections the obtained data for each element are discussed.

15. this step is automatically performed once via a dedicated R script available in our replication package.

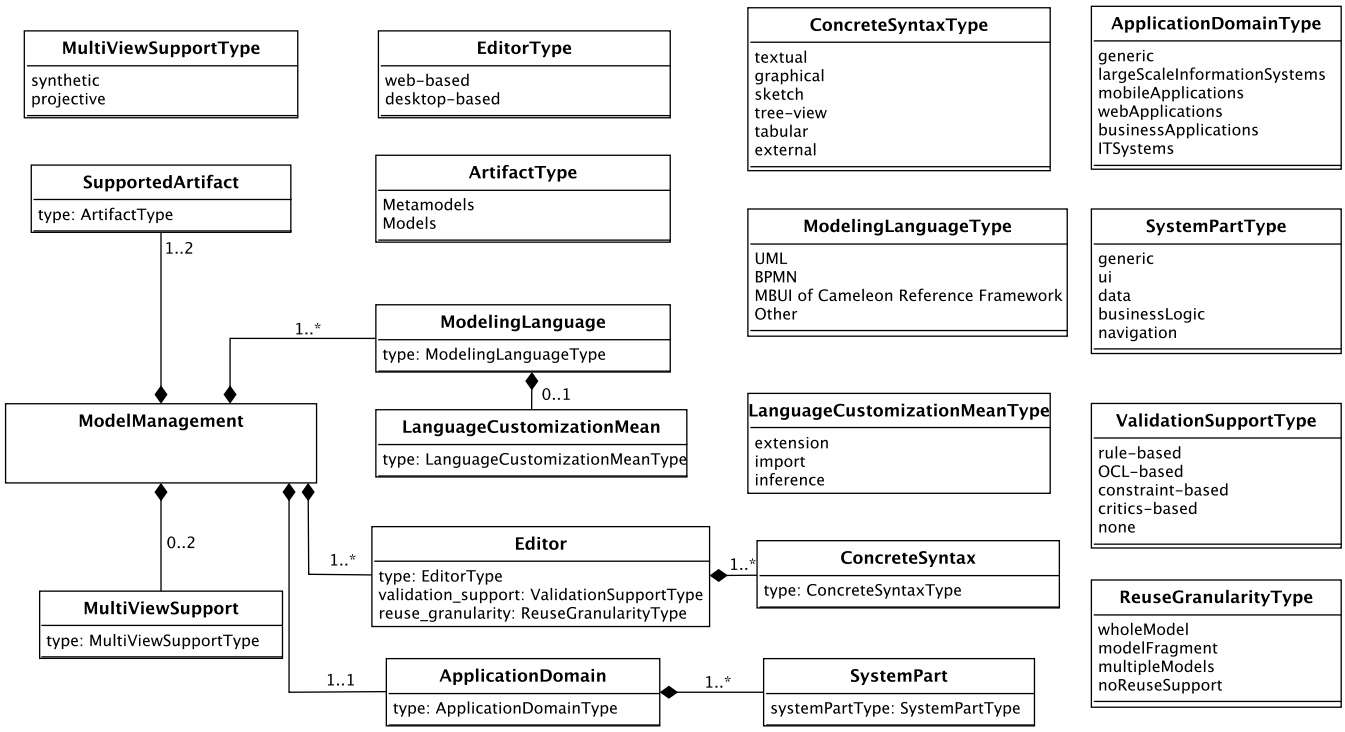

Fig. 4: Taxonomy for the management support of collaborative MDSE approaches

5.1 Supported Artifact

Stakeholders may need to collaborate on different kinds of artifacts. They can collaborate on the specification of models, or they can even define new modeling languages to cover a particular application domain. Most of the analyzed approaches (44 out of 48 studies) support the collaborative definition of models (as shown in Table 5). Only few approaches (4 out of 48 studies) provide modelers with the means to develop in a collaborative manners both models and metamodels. Indeed, further artifacts are typically employed when applying model driven techniques e.g., model transformations and code generators. Even though none of them are considered by the analysed studies, we expect that in the near future they will be also supported by collaborative tools. Such a confidence is supported by some recent works [60]–[63] that in our opinion can go towards that direction.

TABLE 5: Distribution of supported modeling artifacts

Artifact Type	#Studies	Studies
Model	44	P1, P3, P4, P5, P6, P7, P8, P9, P10, P12, P14, P15, P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P28, P29, P30, P31, P32, P34, P35, P36, P37, P38, P39, P40, P41, P42, P43, P44, P45, P46, P47, P48
Model and Metamodel	4	P2, P11, P13, P33

5.2 Modeling Language

During the analysis of the obtained data we noticed a prevalence of collaborative approaches, which are designed to support specific modeling languages (`ModelingLanguageType` concept in Fig. 4) like UML, BPMN, and MBUI of the Cameleon Reference Framework [16] (27 out of 48 studies, see Table 6). According to the extracted data, UML is the most considered modeling language since 20 primary studies are about it.

TABLE 6: Distribution of language specificity

Language specificity	#Studies	Studies
Language specific	27	P3, P6, P7, P9, P10, P15, P18, P19, P20, P21, P24, P25, P26, P27, P29, P30, P31, P32, P34, P35, P36, P37, P38, P42, P43, P44, P45
Language independent	21	P1, P2, P4, P5, P8, P11, P12, P13, P14, P16, P17, P22, P23, P28, P33, P39, P40, P41, P46, P47, P48

More specifically, the means provided by the analyzed approaches to add or refine new languages are essentially three: *import*, *extension*, and type *inference* as identified in the `LanguageCustomizationMeanType` concept in Fig. 4 and shown in Table 7. Even though language customization is not widely supported yet (26 approaches out of the 48 do not provide any language customization means), the *import* technique is the most recurrent one: language definitions are packaged in modules, which can be installed or removed in the provided modeling tools, depending on the stakeholder needs. In the case of the *extension* technique, modelers are provided with extensibility constructs, which permit to extend already existing modeling languages to better

16. http://giove.isti.cnr.it/projects/cameleon.html

cover the concepts of the considered application domain. The approaches implementing the *inference* mechanism (i.e., P2, P5, P22, and P48) permit to define custom languages by sketching new modeling constructs, which then will be used to (semi-) automatically infer their types.

TABLE 7: Distribution of language customization mean type

Language Customization Mean Type	#Studies	Studies
None	26	P3, P6, P7, P9, P10, P15, P18, P19, P20, P24, P25, P26, P27, P29, P30, P31, P32, P34, P35, P36, P37, P38, P42, P43, P44, P45
Import	14	P1, P4, P8, P12, P14, P16, P17, P23, P28, P39, P40, P41, P46, P47
Extension	6	P8, P11, P13, P14, P21, P33
Inference	4	P2, P5, P22, P48

5.3 Editor

In any modeling approach the features provided by the supporting editor can make the difference. According to the extracted data, the analyzed approaches are endowed with editors, which can be distinguished under several characteristics as presented by the concept `Editor` and related elements in Fig. 4. One of the main distinguishing elements is represented by the way editors can be used, i.e., whether they need to be installed on the client machines, or whether they are ready to use since Web-based. As shown in Table 8 most of the editors are desktop-based only, and 15 approaches provide modelers with Web-based editing tools only. Papers P23 and P24 do not provide any client since they are mainly about conceptual aspects of real-time collaborations.

TABLE 8: Distribution of editor type

Editor Type	#Studies	Studies
Desktop-based	28	P1, P2, P5, P9, P12, P15, P17, P21, P22, P25, P26, P27, P28, P29, P30, P31, P33, P36, P37, P38, P39, P40, P42, P43, P44, P46, P47, P48
Web-Based	15	P3, P4, P6, P7, P10, P11, P13, P14, P16, P18, P19, P32, P34, P35, P45
Both	3	P8, P20, P41
Not information	2	P23, P24

Another characteristic aspect of modeling editors is represented by the concrete syntaxes that modelers can use to specify models. According to the concept `ConcreteSyntaxType` shown in Fig. 4, models can be specified by means of several notations, which can be textual, graphical, tree-based, and tabular. It is not possible to identify the notation which best fits any situation. For instance, developers might prefer textual notations, whereas business analysts might like more graphical or tabular-based syntaxes. As shown in Tab. 9, most of the analyzed approaches provide modelers mainly with graphical and tree-based notations. Some approaches permit to specify models in a flexible way by enabling the definition of sketches that might be manipulated later to introduce typing information. The approaches labeled as *external* are not restricted to particular concrete syntaxes. They provide collaborative engines, able to manage models developed by means of external tools.

TABLE 9: Distribution of concrete syntax type

Concrete Syntax Type	#Studies	Studies
Graphical	38	P2, P3, P4, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P18, P19, P20, P21, P22, P24, P25, P26, P27, P29, P30, P32, P34, P35, P38, P39, P40, P41, P42, P43, P44, P45, P47, P48
Tree-based	14	P7, P8, P13, P15, P19, P21, P22, P33, P34, P39, P41, P42, P47, P48
Sketch-based	7	P2, P8, P9, P27, P31, P41, P43
External	6	P1, P5, P17, P23, P28, P46
Textual	5	P13, P16, P21, P24, P45
Tabular	2	P12, P21

When developing modeling artifacts, having the availability of techniques and tools enabling early checks of the specified models is of paramount importance. Validation activities can be performed e.g., to check static semantics aspects, or functional and extra-functional properties of the system being modeled. As shown in Fig. 4 by means of the `ValidationSupportType` concept, several techniques can be adopted to validate models. All of them share the idea of querying the analyzed models in order to find occurrences of some (anti-)patterns of interest. The languages and tools, which are provided to specify such patterns (e.g., OCL expressions, rule-based predicates, etc.) represent the distinguishing elements of the considered validation mechanisms. As shown in Table 10 only 15 approaches out of the 48 primary studies implement some model validation support.

TABLE 10: Distribution of validation support type

Validation Support Type	#Studies	Studies
OCL-based	6	P17, P33, P35, P41, P46, P48
Constraint-based	5	P21, P32, P41, P45, P48
Rule-based	3	P12, P16, P37
Critics-based	1	P41

With the concept `ReuseGranularityType` shown in Fig. 4 we refer to the mechanisms provided by the considered model management infrastructure to search and reuse already specified models (or part of them), by resembling the established reuse and integration practices in software development. In the case of MDSE, similar modeling artifacts often need to be developed from scratch, thus raising the upfront investment and compromising the productivity benefits of model-based processes. For instance, when modelers identify the need for a domain-specific modeling language, it is quite common to implement it from scratch instead of reusing already developed languages that might satisfy their requirements [64]. By analyzing the selected

primary studies, we noticed that reuse can occur at different levels of granularity (see Table 11): P11 and P14 provide the means to reuse already specified models in new projects, whereas P13 and P35 rely on the notion of *package* consisting of a set of interrelated models, which are made reusable as whole. Other approaches permit to link and reuse specific *fragments* of existing modeling artifacts.

TABLE 11: Distribution of model reuse granularity

Model Reuse Granularity	#Studies	Studies
Model Fragment	7	P2, P6, P12, P30, P34, P38, P41
Model	2	P11, P14
Package	2	P13, P35

5.4 Application Domain

According to the analyzed data, most of the considered primary studies support the specification of models for any application domain as shown in Table 12. Only few approaches are defined for dealing with specific kinds of systems i.e., mobile applications, Web application, business applications, and IT systems. An additional level of specificity we have identified is represented by the supported system parts, i.e., user interface, business logic, navigational structure, and data layer. For instance, P19 provides an approach for the collaborative modeling of user interfaces for Web applications.

TABLE 12: Distribution of application domain and system parts

Application Domain Type	#Studies	Studies	Focus
Generic	44	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P15, P16, P17, P18, P20, P21, P22, P23, P24, P25, P26, P27, P28, P29, P30, P31, P32, P33, P34, P36, P37, P38, P39, P40, P41, P42, P43, P44, P45, P47, P48	*Any*. Only P4 focuses on *UI*
Mobile Applications	1	P14	*UI, Data, Business Logic, Navigation*
Web Applications	1	P19	*UI*
Business Applications	1	P35	*Any*
IT Systems	1	P46	*Any*

5.5 Multi-View Support

To tame the complexity of software systems, some approaches permit to decompose models into different views, each focusing on specific aspects of the whole system. According to the `MultiViewSupportType` concept shown in Fig. 4, multi-view modeling approaches are distinguished between *synthetic*, and *projective* [65]. In the former, a number of views are specified by means of dedicate languages, and the model of the whole system is automatically synthesized from them. In the latter, each view is obtained by projecting the system model by hiding details not relevant for the particular view. According to the analyzed data, only 13 out of the 48 primary studies provide the support for multi-view specifications (see Table 13).

TABLE 13: Distribution of approaches supporting multi-views

Multi-Views Support	#Studies	Studies
Projective	7	P12, P13, P20, P21, P35, P36, P46
Synthetic	5	P4, P8, P14, P24, P41
Both	1	P16

Characteristics of the model management infrastructure (RQ1.1)

Most approaches support the collaborative definition of models, rather than other MDSE artifacts, with a prevalence of collaborative approaches supporting the collaborative work on UML models. Editors are mostly desktop-based, mainly with graphical and tree-based notations. Collaborative model validation is only partially supported. Most of the approaches support the specification of models for any application domain, while multi-view support is quite limited.

6 Collaboration (RQ1.2)

In this section we describe the characteristics of collaborative MDSE approaches with respect to the stakeholders' collaboration aspect. By analyzing the primary studies under such a dimension, specific concepts have been identified as shown in Fig. 5. In particular, according to the elicited `CollaborationType` concept, stakeholders can work on the same modeling artifacts in different manners as shown in Table 14 and described below:

▷ *Synchronous:* in this case collaboration occurs in real-time and stakeholders remotely work together on the same artifacts by resembling face-to-face interactions. Such a kind of collaboration can be the most effective since it permits to solve issues in (near) real-time. As discussed later in the paper, the infrastructure supporting such a kind of collaboration should provide stakeholders with the means to share workspaces and to enable the synchronous editing of the artifacts there in;

▷ *Asynchronous:* stakeholders work on the same artifacts but not at the same time. Since different changes can be performed on the same artifacts from different people and at different time, the employed collaborative infrastructure has to provide stakeholders with the means to detect and resolve conflicting changes that might occur. According to the performed analysis, different strategies can be implemented to propagate changes and to manage conflicting ones. Most of the available asynchronous approaches implement typical mechanisms provided by version control

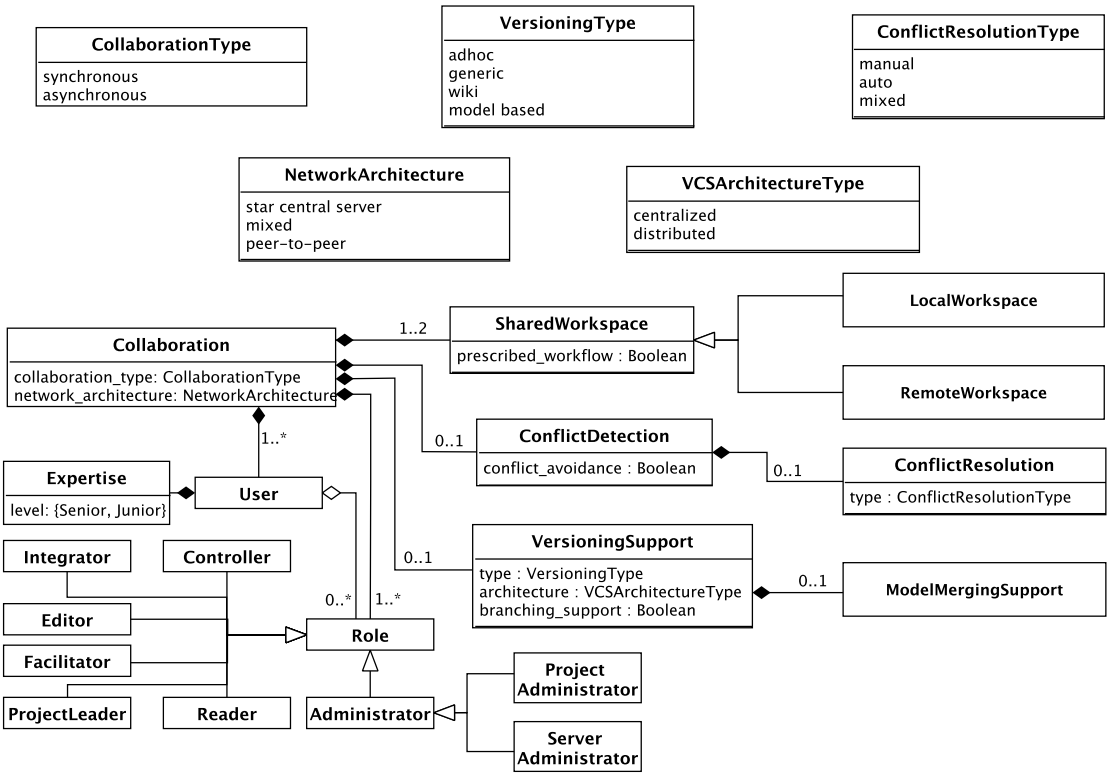

Fig. 5: Taxonomy for the collaboration support of collaborative MDSE approaches

TABLE 14: Distribution of collaboration type

Collaboration Type	#Studies	Studies
Synchronous	24	P2, P4, P7, P9, P10, P11, P13, P14, P15, P16, P17, P18, P19, P24, P25, P26, P27, P29, P31, P32, P39, P40, P43, P44
Asynchronous	20	P1, P3, P5, P6, P8, P12, P20, P21, P30, P33, P34, P35, P36, P37, P38, P42, P45, P46, P47, P48
Both	4	P22, P23, P28, P41

systems like SVN or Git. In particular, modelers perform local changes and then commit them to a central repository. In case of conflicts, modelers have to manually resolve them by possibly interacting with the authors of the conflicting changes. Importantly, before starting any editing session, modelers have to update their local version of the artifacts being modified. In the approaches presented in P36 and P37, modelers do not have to explicitly download the remote version of the edited model, which instead is automatically updated immediately after any commit performed to the central repository. In case of conflicting changes detected during the automatic updates, a general resolution rule is applied i.e., the last operated changes are applied.

According to the `NetworkArchitecture` concept shown in Fig. 5, the adopted collaborative infrastructure can make use of a central server, can be peer-to-peer, or mixed. In case a central server is available (which is the most recurrent configuration as shown in Table 15), collaborative tasks are performed by relying on data remotely stored in a common resource. Alternatively, as presented in P36 and P37 it is possible to work in a collaborative manner by means of peer-to-peer configurations. In particular, all the artifacts, messages, and editing operations are exchanged among the parties involved in the collaborative sessions. In P28, P31, and P41 authors propose a mixed solution relying on a central server providing storage facilities for managing models, whereas messages and editing operations are directly exchanged among the involved stakeholders.

TABLE 15: Distribution of network architecture

Network Architecture	#Studies	Studies
Central Server	43	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P29, P30, P32, P33, P34, P35, P38, P39, P40, P42, P43, P44, P45, P46, P47, P48
Peer to peer	2	P36, P37
Mixed	3	P28, P31, P41

Further than the types of collaboration and network architectures, the taxonomy we extracted and shown in Fig. 5 consists of additional concepts as presented in the remaining of the section. In particular, Section 6.1 describes the concepts related to the versioning support (when available), whereas Section 6.3 presents the concepts related to the editing facilities provided by the adopted modeling tools.

6.1 Versioning Support

Similarly to source code, modeling artifacts can evolve during their life-cycle e.g., due to refinements operated on previously represented abstractions or to the specification of unforeseen requirements. In such cases, having the availability of versioning systems able to keep track of the performed changes is of paramount importance e.g., to perform change impact analysis or to simply retrieve the subsequent versions of a given modeling artifact. Even though the importance of model versioning has been largely recognized by modeling experts, there is still a deficiency of such a facility in currently available collaborative tools (28 out of 48 primary approaches do not provide any model versioning support as shown in Table 16).

TABLE 16: Distribution of versioning type

Versioning type	#Studies	Studies
None	28	P2, P3, P4, P7, P9, P10, P11, P12, P14, P15, P16, P17, P19, P24, P25, P26, P27, P29, P30, P31, P32, P36, P37, P39, P40, P43, P44, P47
Adhoc	9	P1, P8, P13, P18, P20, P23, P33, P42, P48
Models	5	P21, P22, P28, P34, P38
Generic	4	P5, P35, P41, P46
Wiki	2	P6, P45

According to the extracted taxonomy, two types of architecture can be employed for supporting the versioning of the modeling artifacts, i.e., *centralized* and *distributed* (`VCSArchitectureType` concept in Fig. 5). As shown in Table 17, the majority of primary studies supporting a versioning system adopt a centralized architecture (19 studies over 20). Interestingly, only 1 primary study adopts a distributed versioning architecture (P33), where the DICoMEF approach for semi-automatic conflict detection, reconciliation, and merging for models and metamodels has been presented. The DICoMEF approach is fully distributed, but the process of conflict detection and merging is centralized; more specifically, groups of stakeholders can be dynamically formed in DICoMEF and a designated controller has the responsibility to store, receive and forward model change requests among the members of each group.

TABLE 17: Distribution of VCS architecture type

VCS Architecture Type	#Studies	Studies
Centralized	19	P1, P5, P6, P8, P13, P18, P20, P21, P22, P23, P28, P34, P35, P38, P41, P42, P45, P46, P48
Distributed	1	P33

When developing modeling artifacts by exploiting versioning systems, it might be helpful to create *branches* of the project being developed in order to allow model parts to be specified in parallel. This is particularly required for large models, which demand the participation of several stakeholders with different knowledge. Once branches have been properly developed, they require to be merged back onto the parent branch. According to the extracted data, few approaches, 11 out of 48, provide modelers with branching mechanisms as shown in Table 18.

TABLE 18: Distribution of branching support

Branching Support	#Studies	Studies
Not supported	37	P1, P2, P3, P4, P5, P6, P7, P9, P10, P11, P12, P14, P15, P16, P17, P19, P22, P23, P24, P25, P26, P27, P29, P30, P31, P32, P36, P37, P39, P40, P42, P43, P44, P45, P46, P47, P48
Supported	11	P8, P13, P18, P20, P21, P28, P33, P34, P35, P38, P41

6.2 Conflict Management

When several modelers work in parallel on the same artifacts, it is also necessary to adopt *conflict detection* techniques able to discover conflicts, i.e., discordant changes performed on the same elements by different stakeholders. Most of the analyzed approaches provide modelers with some conflict detection techniques (27 out of the 48 analyzed papers as shown in Table 19). Some of the considered papers (12 out of 27) try to prevent conflicts by adding further constraints on the editing phases, e.g., before changing models, modelers have to lock the elements they want to modify. This way, they have exclusive accesses to model elements and consequently conflicts are simply avoided.

TABLE 19: Distribution of conflict detection support

Conflict detection support	#Studies	Studies
Yes	27	P1, P5, P6, P8, P13, P16, P17, P19, P20, P21, P22, P23, P24, P28, P30, P33, P34, P35, P36, P37, P38, P41, P42, P44, P46, P47, P48
Avoided	12	P2, P3, P9, P10, P11, P12, P14, P18, P32, P39, P40, P45
None	9	P4, P7, P15, P25, P26, P27, P29, P31, P43

In order to reconcile conflicts, which might occur because of discordant changes, resolution means should be provided by the adopted collaborative modeling framework. As shown in Table 20 most of the analyzed approaches providing conflict detection features (17 of 27) prefer to completely delegate the resolution phase to the stakeholders that caused the conflicts. Other approaches (9 of 27) implement a mixed approach, i.e., try to automate the resolution phase as much as possible, and involve humans only for deal with discordant changes, which cannot be automatically solved since further information is required. The approach presented in P36 automates the conflict resolution phase by applying a general rule that when two changes are conflicting, those performed late overwrite those occurred first.

6.3 Shared Workspace

By analyzing the 48 primary studies we identified two recurrent aspects related to how stakeholders can actually

TABLE 20: Distribution of conflict resolution type

Conflict resolution type	#Studies	Studies
Manual	17	P1, P8, P13, P16, P17, P19, P20, P21, P28, P30, P34, P35, P41, P42, P44, P46, P47
Mixed	9	P5, P6, P22, P23, P24, P33, P37, P38, P48
Auto	1	P36

collaborate during the editing phases. Few approaches (only 4 as shown in Table 21) expect a collaboration occurring among several people working face-to-face in the same room. Such a kind of collaboration can turn out to be too restrictive and this justifies why most of the analyzed work provide modelers with a remote shared workspace. The works P28 and P30 provide the support for both kinds of interaction.

TABLE 21: Distribution of shared workspace location type

Workspace location type	#Studies	Studies
Remote	42	P1, P3, P4, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P19, P20, P21, P22, P23, P24, P25, P26, P27, P29, P32, P33, P34, P35, P36, P37, P38, P39, P40, P41, P42, P43, P44, P45, P46, P47
Local	4	P2, P5, P31, P48
Both	2	P28, P30

The process underpinning the collaboration of several stakeholders can be different depending on the roles and expertise of the involved modelers and on how they are geographically distributed. To this end, we analyzed if the primary studies faced the problem to give to the users a workflow to follow during the modeling activities. As shown in Table 22, 27 works prescribed a workflow in terms of steps and roles that have to be followed by a given collaborative editing process. Very few papers (only 5 out of 48) consider this aspect irrelevant and do not describe at all the sequence of actions to be followed during the collaborative modeling process.

TABLE 22: Distribution of prescribed workflow

Prescribed workflow	#Studies	Studies
Yes	27	P1, P5, P6, P8, P9, P10, P11, P13, P14, P16, P17, P22, P24, P28, P30, P32, P33, P34, P36, P37, P38, P41, P42, P44, P45, P47, P48
Partially described	16	P2, P12, P15, P18, P19, P20, P21, P23, P25, P26, P29, P31, P35, P39, P43, P46
No	5	P3, P4, P7, P27, P40

6.4 Roles

While analyzing the considered 48 primary studies concerning the *collaboration aspect*, we have identified several user roles, each involved in collaboration activities with different tasks and rights. More specifically, users collaboratively working on the same modelling artifacts can play one or more of the following main roles:

- *Facilitator:* this group consists of users that can play a key role in any collaborative modelling session. They are senior users that make decisions when needed;
- *Project Administrator:* users in this group have full access to the artifacts being developed including their deletions;
- *Server Administrator:* this group refers to users that can administer also users further than the projects being developed;
- *Reader:* users belonging to such a group can only see developed modelling artifacts without having the possibility to operate changes on them;
- *Integrator:* some approaches recognize such a role representing users, which have the responsibility of performing integration activities i.e., merging changes operated by different developers and ask them to review and agree the integrated results;
- *Controller:* the controller propagates accepted changes to all members of a given group of users, and changes propagated from the controller are applied on the artifacts stored in the main branch of the considered repository;
- *Editor:* users of such group can read and write their local copy of the (meta)models, and can only read those that are in the main branch of the considered repository;
- *Project Leader:* in case of conflicting changes, some approaches consider a particular kind of users that can start discussions among modelers aiming at mitigating conflicting changes and resolve them. Project leaders have such a responsibility.

TABLE 23: Distribution of roles

Roles	#Studies	Studies
Facilitator	8	P3, P9, P21, P23, P30, P32, P37, P45
Administrator (Project/Server)	4	P11, P14, P16, P21
Reader	4	P11, P16, P21, P33
Integrator	1	P1
Controller	1	P33
Editor	1	P33
Project Leader	1	P45

The distribution of such roles in the analyzed approaches is shown in Table 23. The primary studies that are not included in those shown in Table 23 consider all the modelers involved in the collaboration activities as pairs.

Characteristics of collaboration means (RQ1.2)

The collaborative infrastructure is mostly based on a star-like topology with data stored in a central server. Versioning is mostly supported through a centralized versioning system architecture. Branches creation is only sporadically supported while conflict detection is employed by over half of the approaches. Collaboration is mostly based on a remote shared workspace. The

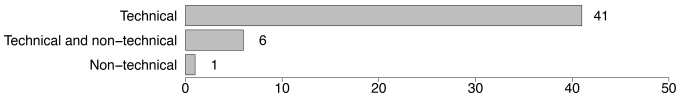

Fig. 7: Results - stakeholder types

workflow specification of how different stakeholders shall interact is described (or partially described) by most of the approaches. MDSE collaboration involves a number of different stakeholders with different roles.

7 COMMUNICATION (RQ1.3)

In this section we describe the characteristics of collaborative MDSE approaches in relation to how they support stakeholders' communication. Figure 6 shows the taxonomy we extracted from the primary studies when focusing on this dimension of collaboration. The first layer of the taxonomy is composed of three elements: target stakeholders, workspace awareness, and communication support. We describe the obtained data for each element in the following subsections.

7.1 Target Stakeholders

By taking inspiration from the conceptual foundations of the ISO/IEC/IEEE 42010 standard [35], a stakeholder can be defined as an individual, team, or organization having an interest in one or more models of the system. Since collaboration intrinsically implies involvement of a potentially heterogeneous set of stakeholders, the first aspect that we want to investigate is whether involved stakeholders are *technical* (e.g., software architects, developers, system engineers) or *non-technical* (e.g., business analysts, UI designers, or even product owners) (see the `StakeholderType` concept in Figure 6). As shown in Figure 7, the vast majority of the studies assume that only technical stakeholders collaborate on models (41/48), followed by only 6 approaches designed for both technical and non-technical stakeholders (P4, P6, P14, P20, P44, P47). Interestingly, there is a unique approach designed for non-technical stakeholders only (P2), published in 2015; the approach allows participants to sketch and use lightweight metamodeling techniques to collaboratively define notations and step-wise guides them towards the formalization of the drawings into precisely defined models.

During the analysis of obtained data we also noted that there are 9 approaches that are dedicated to *specific types of stakeholders*. Those approaches provide some specific instrument or modeling language tailored to a specific application domain (e.g., data-intensive mobile apps in P14) or stakeholders' role (e.g., software architects in P47). In Table 24 we present the full list of types of stakeholders as they have been reported in those studies.

7.2 Workspace Awareness

Workspace awareness is defined as the up-to-the-moment understanding of other stakeholders' interaction with a shared workspace [66]. In our context, the workspace comprises the modelling environment provided by the collaborative MDSE approach (e.g., the online web editor in which stakeholders create and modify models in real-time in P13) and all its additional facilities like a shared dashboard (e.g., in P21), integration to a wiki-based engine (e.g., in P7), or special notification areas in the tool (e.g., in P24). Intuitively, in the context of our study workspace awareness focuses on the level of understanding that each stakeholder has on the operations performed by other stakeholders on the modelling artifacts. It is important to note that workspace awareness is limited to events happening in the workspace [66], it does not encompass verbal communication or informal awareness mechanisms like chats, emails, or documentation exchanged outside the workspace provided by the MDSE approach.

TABLE 24: Approaches dedicated to specific types of stakeholders

Study	Specific target stakeholders	Year
P4	Developer, functional analyst, usability expert, domain expert, designer	2014
P6	Business process expert, modeling expert	2013
P14	Designer, end user, customer, information architect, UI designer, app developer, back-end developer, content producer, project manager	2014
P17	Software architect	2010
P19	Designer, developer, software architect, software engineer	2013
P31	Analyst, business process stakeholder	2011
P44	Domain expert, model engineer	2012
P45	Business analyst	2011
P47	Software architect	2015

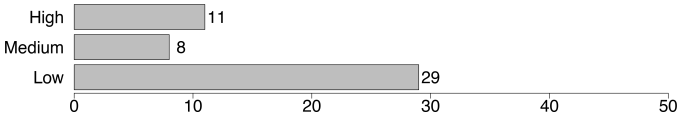

Fig. 8: Workspace awareness levels

Table 25 details the *tools* provided by each approach for supporting the workspace awareness among stakeholders. From the extracted data we can observe that there is a certain fragmentation with respect to the types of tools for workspace awareness, ranging (among many) from real-time model updates (28/48 primary studies), to the highlighting of other stakeholders' cursors or pointers (5/48), to model update notifications via automatic emails (2/48). It should be noted that many collaborative MDSE approaches (specifically, 36/48) provide multiple workspace awareness tools in combination; this means that stakeholders can benefit from the characteristics of each available tool in terms of interaction with other stakeholders, understandability of the models being produced, teamwork, and situational awareness of their current project.

In order to better investigate how our primary studies perform with respect to workspace awareness, we assess their *awareness level* according to the elements of workspace awareness presented in [66, Table I]. More specifically, here we consider the *who*, *what* and *where* elements relating to the present state of a project; the final awareness level of a primary study is *Low* if it supports only one element (according to the I1 inclusion criterion, there are no primary

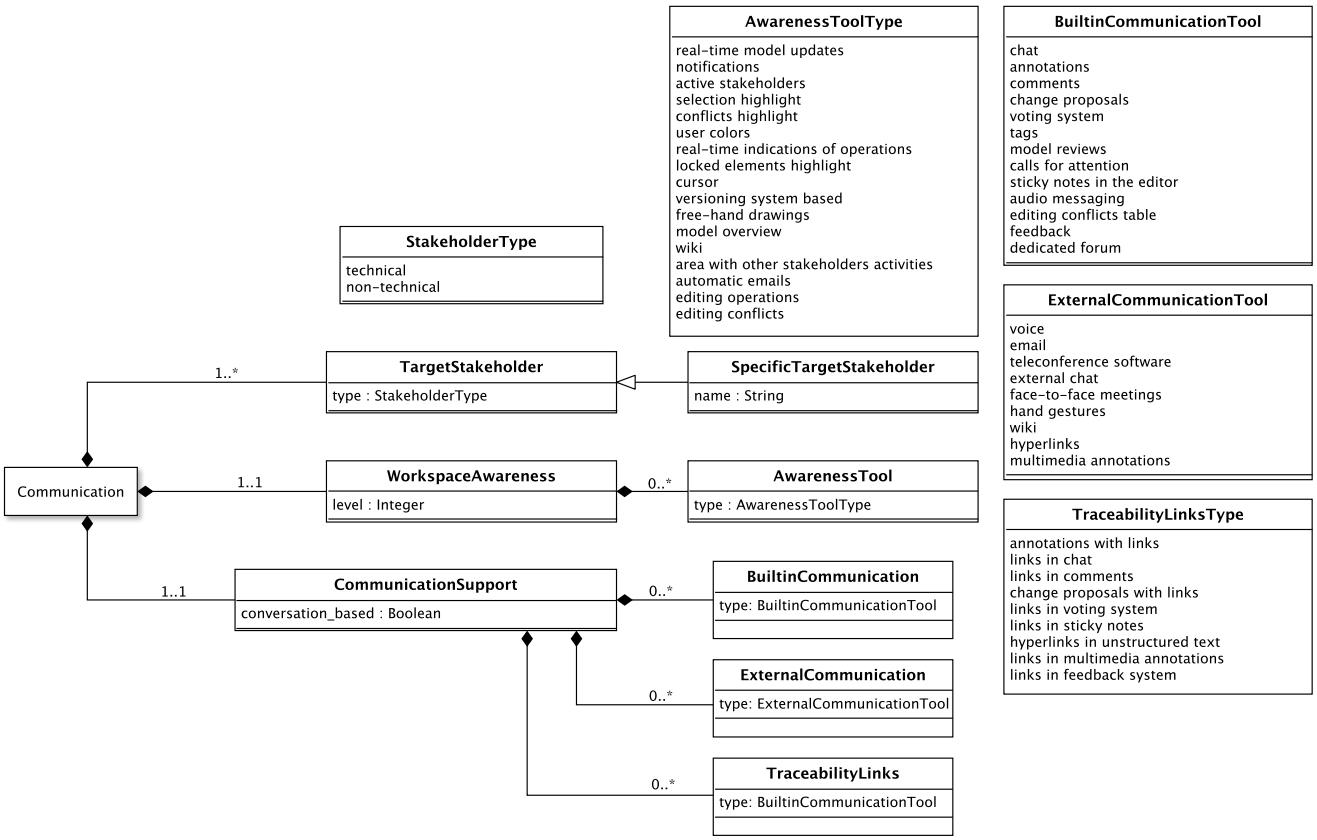

Fig. 6: Taxonomy for the communication support of collaborative MDSE approaches

TABLE 25: Distribution of workspace awareness tools

Awareness tools	#Studies	Studies
Real-time model updates visualized directly in the modeling editor	28	P2, P4, P7, P9, P10, P11, P13, P14, P15, P16, P17, P18, P19, P22, P24, P25, P26, P27, P29, P31, P32, P36, P37, P39, P40, P41, P43, P44
Generic *notification* system (e.g., popup messages, etc.)	20	P6, P7, P10, P13, P14, P15, P16, P17, P18, P21, P23, P24, P28, P33, P34, P38, P42, P45, P46, P47
List of currently *active stakeholders*	15	P6, P7, P9, P10, P18, P27, P30, P31, P32, P36, P39, P40, P41, P43, P44
Highlight the selection of model elements by other stakeholders	10	P2, P6, P7, P10, P11, P25, P27, P41, P43, P44
Highlight conflicts on model elements after concurrent editing	8	P6, P22, P24, P33, P34, P44, P47, P48
Each user as an associated *color* for highlighting her actions or messages	8	P7, P9, P27, P31, P39, P40, P41, P44
Real-time indication about the *operations performed by other stakeholders*	8	P22, P34, P38, P39, P40, P41, P44, P48
Highlight locked model elements by other stakeholders	7	P2, P3, P8, P12, P28, P29, P34
Each stakeholder can see other stakeholders' *cursor or pointer*	6	P18, P25, P29, P39, P40, P43
Based on the *versioning system* (e.g., commit messages, update notifications, etc.)	5	P1, P5, P35, P42, P46
Free-hand drawing in a special area of the client tool for sketching ideas	3	P9, P39, P40
Model overview where stakeholders can visually see where other stakeholders are working	3	P9, P39, P43
Based on a *wiki* engine (e.g., updates log, active users, etc.)	2	P6, P45
Dedicated area with general information about *other stakeholders' activities* (e.g., log)	2	P6, P10
Notifications via *automatic emails*	2	P20, P34
Dedicated area with *editing operations* details (e.g., operations' timestamps)	2	P8, P21
Dedicated area with editing *conflicts* to manage	1	P38

studies with zero workspace awareness elements in our study), $Medium$ if it supports two elements, or $High$ if it supports all elements. As shown in Figure 8, the majority of the primary studies have a Low workspace awareness level (29/48), whereas $Medium$ and $High$ levels are quite balanced with 8 and 11 primary studies, respectively. This

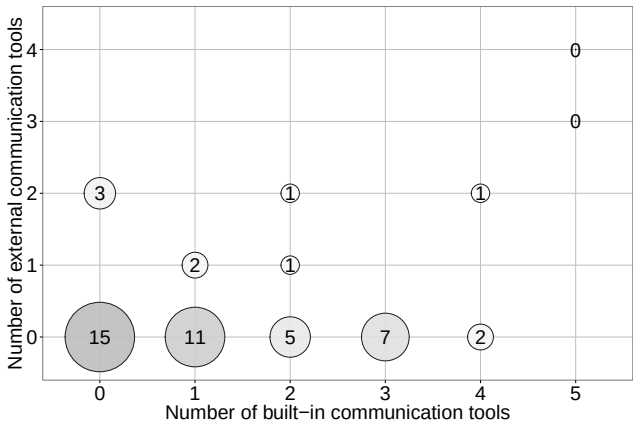

Fig. 9: Results - Number of built-in and external communication tools

result is quite interesting because it can be seen as an indication of room for improvement in terms of workspace awareness for collaborative MDSE approaches. For instance, workspace awareness support for the WebGME (P13) users is allowed through real-time updates in the modeling editor and through an implicit lock that prevent to modify/delete a model element already selected by another user: the locked element gives to the user an indication of where another user is working on (i.e. *where* dimension support). Unfortunately the modeling updates just "appear" in the editor without any indication about who is updating nor the details of the current actions that other users are performing; this is a clear gap in the *who* and *what* dimensions of workspace awareness on which the authors of WebGME may work.

7.3 Communication Support

As already discussed in Section 3, in MDSE communication support allows stakeholders to discuss and present what they and other stakeholders are doing on specific modelling artifacts. Historically, software engineers have adopted a wide range of *communication tools and technologies*, such as telephone, teleconferences, email, voice mail, discussion lists, the Web, instant messaging, voice over IP, etc. [67]. Our primary studies present a wide set of communication tools and technologies; we classified them into built-in and external communication tools. Built-in communication tools are provided by a collaborative MDSE approach and are integrated into it (e.g., models annotations in the editor, internal chat), whereas external communication tools are only prescribed by the collaborative MDSE approach and do not live inside it (e.g., voice calls, emails, face-to-face meeting).

Table 26 presents the distribution of all *built-in and external communication tools* emerging from our analysis. It is easy to note that there is a great variability here, where we have chat (18/48) and model annotations (13/48) as clear winners for built-in communication means; voice (3/48), email (2/48), and teleconference software (2/48) are the most prescribed external communication means.

TABLE 26: Communication tools

Built-in comm. tools	**#Studies**	**Studies**	**Conversation-based (subset)**
Chat	18	P7, P9, P10, P11, P16, P18, P22, P24, P25, P26, P27, P29, P32, P34, P39, P40, P43, P44	P7, P9, P10, P11, P16, P18, P22, P24, P25, P26, P27, P29, P32, P34, P39, P40, P43, P44
Annotations	13	P3, P6, P7, P8, P12, P18, P20, P21, P26, P27, P30, P31, P40	-
Comments	8	P6, P8, P12, P20, P21, P27, P30, P45	P6, P8, P12, P20, P21, P45
Change proposals	7	P21, P22, P30, P33, P39, P40, P48	-
Voting system	6	P22, P30, P39, P40, P45, P48	-
Tags	1	P6	-
Model reviews	1	P8	-
Calls for attention	1	P9	-
Sticky notes in the editor	1	P9	P9
Audio messaging	1	P25	P25
Editing conflicts table	1	P27	P27
Feedback	1	P38	-
Dedicated forum	1	P45	P45
External comm. tool	**#Studies**	**Studies**	**Conversation-based (subset)**
Voice	3	P2, P28, P30	P2, P28, P30
Email	2	P20, P47	P47
Teleconference software	2	P28, P30	P28, P30
External chat	1	P47	P47
Face-to-face meetings	1	P48	-
Hand gestures	1	P2	P2
Wiki	1	P20	-
Hyperlinks	1	P3	-
Multimedia annotations	1	P33	-

By looking at the obtained numbers, it is evident that there is a certain unbalance with respect to the number of built-in and external communication tools; indeed, as shown in Figure 9, there is a strong predominance of built-in communication tools with respect to external ones. Interestingly, we found out also that 15 out of 48 primary studies do not provide any clear indication about how communication is supported by the proposed approach, unveiling a potential gap in how state-of-the-art approaches support the three dimensions of collaborative MDSE. Moreover, 11 studies provide only one (built-in) communication tool, followed by 14 studies providing more than one built-in communication tool (e.g., the approach presented in P39 supports an internal chat system, an engine for making change proposals in the models, and a voting system for those proposals). Finally, there are few studies in which external communication tools are prescribed; in this context we can observe that 3 studies propose a combined use of two external communication tools (P2, P28, P47), 2 studies support the use of one built-in and one external communication tool (P3 and P33), and other 3 studies support and prescribe other different combinations of communication tools (P20,

P30, and P48). None of the considered studies propose the use of more than 4 built-in and more than 2 external communication-tools.

We also analyzed our primary studies in terms of support for conversation-based communication. In this context, the idea is that stakeholders may need to communicate with each other with a question-and-answer interaction style, where communication contents about some modelling artifact can be organized hierarchically and linked to other communication contents (e.g., a comment in a model can be marked as a reply to another comment). In the last column of Table 26 we report the studies providing conversation-based communication.

TABLE 27: Types of supported tracing links

Traceability link type	#Studies	Studies
Annotations on model elements	14	P3, P6, P7, P8, P12, P18, P20, P21, P26, P27, P30, P31, P33, P40
Links to model elements in *chat* text	6	P9, P11, P26, P39, P40, P44
Comments on model elements	4	P8, P12, P20, P21
Change proposals linked to model elements	3	P21, P22, P48
Voting system for change proposals linked to model elements	2	P22, P48
Links to model elements in the *Wiki* text	1	P20
Sticky notes on model elements	1	P9
Hyperlinks to model elements in unstructured textual documents	1	P3
Links to model elements in *multimedia annotations*	1	P33
Feedback system linked to model elements	1	P38

What distinguishes collaboration in MDSE within software engineering is that collaboration in MDSE is artifact-based; indeed, the focus of MDSE activities is on the production of new models, the creation of shared meaning around the models, and elimination of error and ambiguity within the models [67]. Under this perspective, it is fundamental to have *traceability links to the models* in order to keep a structured link between the design decisions discussed in communication-oriented contents (e.g., the text of a chat or the page of a Wiki) and modeling artifacts (e.g., a model or a specific model element). In Table 27 we present the primary studies providing those tracing links.

Characteristics of communication means (RQ1.3)

The vast majority of the primary studies are intended to support the collaboration among technical stakeholders, with a unique approach designed for non-technical ones. Many are the tools for workspace awareness (being real-time updates and notification systems the most frequent). Still, the awareness level in most of the studies is low. Communication tools and technologies, classified into built-in and external, are diverse, with chat and annotations being the most used ones (in the *built-in* category). Conversation-based communication is quite frequently used, especially through chat and comments. Traceability links are frequently kept through annotations or links to model elements in chat text.

8 Challenges and Shortcomings (RQ2)

In this section we focus on the challenges that researchers are facing and have identified either as actual limitations of their approaches or future enhancements that will be considered in the future. In this context, the main objective is to answer research question RQ2, that is, to identify current limitations and challenges with respect to the state of the art in collaborative MDSE. In order to achieve this goal, for each primary study we collected all the information provided by its authors regarding (i) identified limitations of the proposed approach, (ii) identified challenges that have not been solved in the current form of the proposed approach, and (iii) discussed directions for future work. Such information has been extracted by (i) making a thorough reading of the full text of each primary study, and (ii) applying the open card sorting technique [58], similarly to what has been done for building the classification framework in the context of RQ1. After the application of the card sorting technique we noticed that the identified clusters of limitations and shortcomings could have been further grouped into two main groups:

- Limitations and shortcomings related to aspects of the C-MDSE taxonomy[17] (see Table 28);
- Limitations and shortcomings focussing on qualitative aspects of the presented approaches, such as their usability, performance and scalability (see Table 29).

Table 28 provides an overview of the limitations and shortcomings which have been mentioned more than once among our primary studies. By looking at the table it is evident that the most relevant challenge with respect to collaborative MDSE is *conflicts management*; indeed, even if in the recent years some approaches for managing this aspect of collaborative modeling have been proposed (e.g., P46, P47), 15 studies over 48 still mention conflict management either as a limitation or as an aspect to improve in the future. Model synchronization and propagation of changes across models edited within the collaborating team are also mentioned as relevant challenges (9 studies over 48).

As shown in the table, many other limitations and challenges have been identified in the primary studies, ranging from improving the support for the collaboration workflow of the team (4 studies), interoperability with external modeling, analysis, or simulation tools (2 studies), and so on. Finally, in 11 cases the authors mentioned approach-specific limitations and challenges, e.g., *the definition and application*

17. Concepts of the classification framework are referred in the third column by means of a dot notation. For instance, with *Collaboration.ConflictDetection.ConflictResolution* we refer to the concept *ConflictResolution* contained in the *ConflictDetection* entity belonging to the *Collaboration* dimension as shown in Fig. 5

of transformations between models across different abstraction levels (P19), the fact that *the technology for deploying software tools over the web is still immature* (P3), or the need to *manage the logs in a systematic manner* (P26).

As shown in Table 29, from a qualitative perspective we identified three recurrent areas of improvement for C-MDSE approaches. Specifically, *usability* has been mentioned in 7 primary studies over 48, followed by *performance* (3/48) and *scalability* (3/48) improvement. In conclusion, from the extracted data we can observe that identified limitations and shortcomings are quite fragmented, where each research group is focusing on specific sub-problems related to collaboration in MDSE (this is also evident from the paper fragments reported in Table 29). Nevertheless, the points discussed above can be seen as an indication of the specific areas within collaborative MDSE that will likely receive scientific interest in the future; so, future researchers on collaborative MDSE can use them as a compass towards making an impact in this specific research domain.

Challenges and Shortcomings (RQ2)

Limitations and shortcomings are varied. Conflicts management, while addressed in some papers, is still frequently considered to be a limitation or to require some improvement. Model synchronization with change propagation is also mentioned as relevant challenge. From a qualitative perspective, usability, performance and scalability are the most mentioned areas of improvement for future C-MDSE approaches.

9 Publication Trends (RQ3)

In this section we present the publication trends on collaborative MDSE approaches. In order to provide a complete picture about the number and types of publications on the topic, in this section we consider *all* the selected publications, independently of the clustering step we performed during the search and selection phase (see Section 4.2.2). More specifically, for answering RQ3 we considered a total of 106 publications, including both the 78 publications that we selected before the clustering step and the 28 publications resulting from the snowballing activity. For each primary study we extracted publication year, publication venue, and publication type. In the following we discuss the obtained results.

Figure 10 presents the distribution of the publications on collaborative MDSE approaches[18]. From the collected data, we can observe that relatively few studies were published until 2003, whereas we can notice a growth of the number of publications starting from 2004. More precisely, the average number of publications between 1996 and 2003 is less than 1 study per year, whereas it reached a value of 8.33 publications per year in the period between 2004 and 2015 . This result confirms the scientific interest and need of research on collaborative MDSE approaches in the last years.

The first publication on collaborative MDSE was published in 1996 (P12), where the authors presented MetaEdit+. The key features of MetaEdit+ as collaborative MDSE approach are: (i) the support of high-level specification languages (i.e., what we could call a domain-specific language today), (ii) an open architecture in which the models repository is agnostic of the used tools and provides dedicated APIs to tool providers, (iii) a set of mechanisms for concurrent access of repository data via different tools and by different types of users, (iv) the support for different alternative views of the same models, such as matrices, tables, etc. As a comparison to a modern approach for collaborative MDSE we refer the reader to our discussion of WebGME in Section 3.

We classified analyzed research studies in order to assess their distribution by (i) type of publication (i.e., journal, conference, or workshop paper) and (ii) targeted publication venues.

Figure 11 shows the publication types of the analyzed primary studies over the years [19]. The most common publication type is conference with 62 (65.72%) studies over 106, followed by workshop papers with 27 (28.62%) studies, and finally journal papers with only 17 (18.02%) studies. Such a high number of conference and workshop papers may indicate that this topic is still a young research theme, despite some studies have been already published in the nineties.

TABLE 30: Publication venues with more than one publication on collaborative MDSE

Publication venue	Type	#Publications
ACM/IEEE international conference on Model Driven Engineering Languages and Systems (MODELS)	Conference	5
International Conference on Global Software Engineering (ICGSE)	Conference	5
Comparison and Versioning of Software Models (CVSM)	Workshop	5
International Conference on Software Engineering (ICSE)	Conference	4
ACM Symposium on Applied Computing (SAC)	Conference	4
ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW)	Conference	4
International Conference on Computational Science (ICSS)	Conference	4
Hawaii International Conference on System Sciences (HICSS)	Conference	3
Others	-	72
TOTAL	-	106

Table 30 shows the publication venues that hosted more than two publications (the last row of the table is an aggregate of all the publication venues with two or less publications). From these data we can notice that research on collaborative MDSE is spread across a large number of venues (76 venues for 106 publications) spanning different research areas like (global) software engineering, MDSE, system engineering, business informatics, programming languages. We can elaborate this finding as an indication that collaborative MDSE is perceived today as orthogonal with respect to many other research areas, rather than a specific research topic. Nevertheless, the organization of

18. Our search process covers the research studies published until January 2016, thus potentially partial data for 2015.

19. See previous footnote.

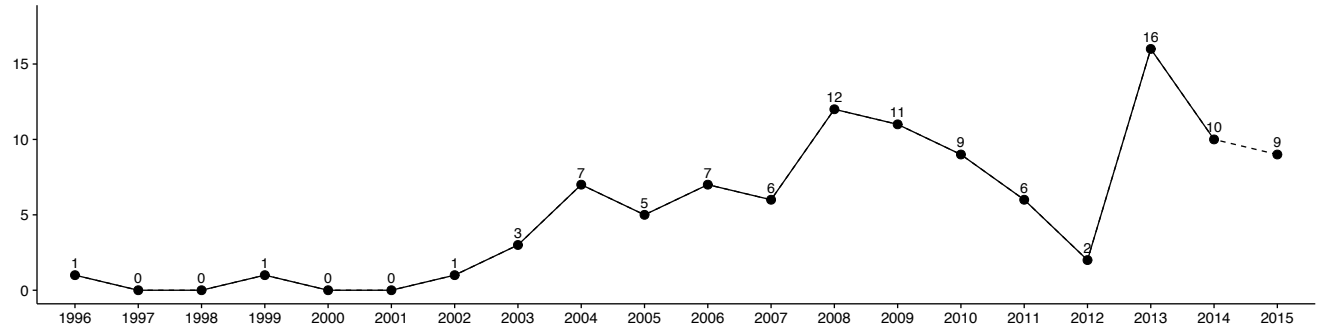

Fig. 10: Distribution of primary studies by year

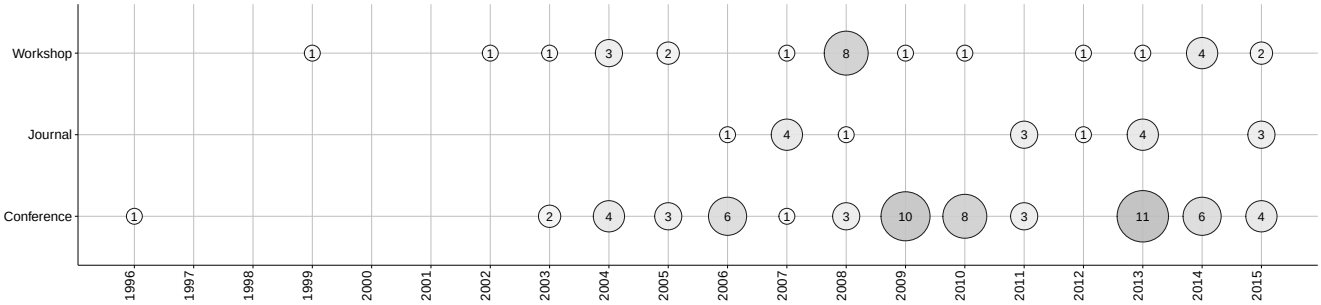

Fig. 11: Distribution of primary studies by type of publication

scientific events (e.g., conferences or workshops) fully dedicated to collaborative MDSE may help in giving a clearly defined identity of the research community working on this topic. The first step towards this change is the international workshop on Collaborative Modelling in MDE (COMMitMDE 2016)[20], at its second edition and co-located with the ACM/IEEE international conference on Model Driven Engineering Languages and Systems (MODELS), the most known forum in the MDSE community.

Publication trends (RQ3)

The first primary paper on collaborative MDSE dates back to 1996. While the number of studies published in the 1996-2003 time frame has been quite limited, a growth can be noticed from 2004 onwards. Most of the papers have been published in conferences and workshops, across a large number of different venues.

10 Orthogonal Findings

This section reports the results orthogonal to the vertical analysis presented in the previous sections. For the purpose of this section, we cross-tabulated and grouped the data, we made comparisons between pairs of concepts of our classification framework, and identified perspectives of interest.

Collaboration types VS versioning. As previously explained, the majority (22/28) of approaches supporting *synchronous collaboration do not provide any means for model versioning*. This result is surprising since users of those approaches collaborate in real-time on models, but they do not have any information about how the models evolved throughout their life span, they cannot perform rollback operations to past versions of the models, they do not have a vision about who worked on the models in the past, etc. Such operations play a key role when collaboratively working on source code development by means of versioning systems like SVN and Git. In other settings, real-time collaboration is endowed with versioning facilities. For instance, Google Docs permits different users to collaboratively work on the same document, and gives the possibility to revert textual changes, to keep track of the modifications done on the text, and each user is aware of the editing operations done by the other contributors. Thus, we can reasonably expect that synchronous collaboration and model versioning will play together a similar role in collaborative MDSE approaches as well.

The situation about approaches with asynchronous collaboration is more stable, with the majority (18/24) of approaches supporting some kind of model versioning, though it is quite widespread with 7 approaches providing an ad-hoc versioning system, 5 providing model-level versioning, 4 reusing a generic text-based one, and 2 building on a wiki-based versioning system. Nevertheless, 6 approaches do not provide any versioning support; in those cases the approach internally manages and stores past versions of the models (mainly for merging and conflict detection), but they are not explicitly exposed to the modeler.

Collaboration types VS conflict management. It emerged that *conflicts are managed in a very variegated manner in approaches with synchronous collaboration*. More specifically,

20. Web: http://cs.gssi.it/commitmde2016/; Proceedings of the first edition: http://ceur-ws.org/Vol-1717/

9 approaches do not directly manage conflicts (e.g., the order of edit operations over time is used as driver for conflict management), 9 approaches avoid conflicts on models by design (e.g., locking mechanisms on the fragments of models being edited), 9 approaches provide explicit conflict detection mechanisms (e.g., via 3-way merging). In any case, complementing synchronous collaboration with real-time communication mechanisms will surely help in terms of conflict avoidance, mitigation, and resolution.

For what concerns approaches with asynchronous collaboration, the majority (21/24) manages conflicts with dedicated engines, whereas the remaining approaches (3/24) have mechanisms for conflicts avoidance.

Collaboration types VS communication. Chat is the clear winner in approaches with synchronous collaboration (14/27), where in some cases it has been complemented with other communication means such as model element annotations (6/27), change proposals (2/27), comments on models (1/27), etc. Approaches with asynchronous collaboration are far more widespread in terms of communication means, no clear trend can be identified here.

Collaboration types VS shared workspace. Collaborative MDSE surely helps for geographically distributed teams, but also teams working in the same place. In this case we assess which collaboration type has been more investigated by researchers, depending on whether the targeted stakeholders are geographically distributed or not. We found a striking balance when cross-checking those two aspects of collaboration, with asynchronous and synchronous collaboration similarly tailored for geographically-distributed (22/27 vs 18/24, respectively), localized teams (2/27 vs 2/24 cases, respectively), and mixed teams (1 case each).

Multi-view VS modeling editors. We cross-checked the support for multi-view modeling with the types of modeling editors. It emerged that *multi-view modeling is not correlated with specific types of editors* (e.g., graphical, textual, etc.). However, we noticed that *the majority of approaches supporting external third-party editors (5/6) supports only single-view modeling*. This finding can be seen as an indication of future research in which multi-view modeling and third-party editors can be integrated in order to let users benefit from (re-)using familiar modeling editors when doing multi-view modeling. As of today, the only approach supporting multi-view modeling with third-party editors is P46, where MagicDraw[21] and Eclipse are used as external editors via dedicated model adapters.

Multi-view VS UML. *The majority of UML-based approaches (17/22) do not provide any means for supporting multi-view collaborative modeling*; i.e., those approaches support only to edit one UML diagram at a time. This result is quite interesting since UML is intrinsically a multi-view modeling language, where multiple diagrams can be instantiated from a single UML model. It does not come as a surprise that the remaining 5 approaches (namely, P20, P21, P35, P36, P46) are based on projectional views.

Multi-view VS workspace awareness. In our vertical analysis we highlighted that only 11 primary studies over 48 score high in terms of workspace awareness. Interestingly, the majority of them (10/11) are single-view approaches. We can interpret this result as an indication that so far researchers have focused their efforts on the simplest case, where stakeholders collaborate on the same models and concepts. The only exception to this trend is P41, where real-time model updates, user selection highlighting, active users, and their actions with personalized colors are all integrated within a meta-tool for (synthetically linked) domain-specific modeling languages.

Modeling editors VS stakeholder types. The underlying rationale behind cross-checking the proposed modeling editors and stakeholder types is to identify which modeling editors researchers perceive as more suitable for technical or non-technical stakeholders. In the following we categorize our observations with respect to the types of involved stakeholders:

- *Technical and non-technical stakeholders*. Interestingly, in almost all approaches in which these two types of stakeholders collaborate (5/6), the graphical editor is the only means for manipulating models (the only exception is P45, where the graphical editor is complemented by a textual one). This decision may be rooted into usability and understandability aspects of the editors for non-technical users. This reflection unveils an interesting gap in current research on collaborative MDSE: since it is well known that graphical editors are not really suitable for large or complex models [68], as of today collaborative MDSE approaches allow technical and non-technical stakeholders to collaborate only on small-scale simple models; given that the trend of having larger and larger models is evident [68], future collaborative MDSE approaches shall provide efficient and elegant solutions to this limitation.
- *Technical stakeholders only*. Across the 41 approaches for technical stakeholders only we can see a wide spreading of many combinations of modeling editors, with a clear prevalence of graphical, tree-based, and textual editors. An interesting perspective is given by the fact that external third-party editors are used only by technical stakeholders. This fact can be due to the need of technical stakeholders to actually perform operations on the models for which specific tools are strictly needed (e.g., analysis, code generation, etc.).
- *Non-technical stakeholders only*. The only approach dedicated to non-technical stakeholders only is FlexiSketch (P2), which is the only approach providing a sketch-based modeling editor (complemented with a graphical one). Proposed in 2015, FlexiSketch can be seen as one of the first attempts in achieving fast and flexible editing with non-technical stakeholders, together with a sketch recognition algorithm and collaborative features (they use the whiteboard metaphor). This line of research may play a relevant role in the future of collaborative MDSE in terms of a better involvement of non-technical stakeholders into MDSE and model-based development in general.

21. http://www.nomagic.com/products/magicdraw.html

11 Discussion

The body of knowledge of this work relies on a total of 106 selected papers, clustered into 48 primary studies (as discussed in Section 4.2.2). Each paper has been selected according to our definition of collaborative MDSE; more specifically, each selected paper is a scientific peer-reviewed article where multiple stakeholders manage, collaborate, and are aware of each others' work on a set of shared models, and covers the three dimensions of model management, collaboration, and communication.

Our study reveals that some of the different taxonomy elements are more rarely covered with respect to others. Specifically, *multi-views*, *validation support*, *reuse support*, and *branching* are individually covered by no more than one third of the papers. Let us clarify that this result by no means implies that there is limited literature or support on collaborative multi-views management, validation support, etc. For example, multi-view modeling and management is becoming prominent in software engineering, both in the domain of software architecture description (e.g., [69], [70]) and in model-driven engineering (e.g., [71]). Instead, this means that among all the selected papers (covering at the same time model management, collaboration, and communication elements), those four elements of the taxonomy have a more limited application.

We also noticed that different primary studies focus differently on individual dimensions of collaborative MDSE. The *Model Management* dimension is strongly covered by papers P35 (supporting 11 over 12 aspects of management), and P12, P13, P14, P21, P41, and P46 (covering 10 aspects). On the opposite, P23, P24, P31, and P44 support only six aspects of model management. *Collaboration* is totally covered by P8, P21, and P33 (with 12 over 12 aspects covered), while minimally covered by P4, P7, and P27 (covering only aspects of collaboration). *Communication* is extensively supported by P6, P20, P44, and P48 (with 8/10 aspects covered), while a limited support is provided by P5, P13, P15, P23, P35, P36, P37, P41, P42, and P46 (with 3/10 aspects covered). This reflects the fact that, while all our primary studies cover (by definition) the three dimensions, their primary focus is still on one specific dimension, rather than on all of them. Our interpretation of this phenomenon is that scientific publications, in order to make a contribution, preferably focus on a specific aspect or dimension, rather than presenting a broad and complete approach with respect to all the three dimensions. This trend may be different in practice, where collaborative MDSE tools might instead cover many dimensions at once. We will analyze, as part of our future work, how commercial tools match with our taxonomy and dimensions.

What our study reveals is also that many of the analyzed approaches (20 primary studies) are built specifically for the UML, and support the collaborative work on more than one UML model (11 studies). Metamodel-level collaborative work is supported by four studies (P2, P11, P13, and P33 in Tab. 5). UML is by far the most known modeling language in MDSE, so, it comes with no surprise that most of the approaches focus on collaborative aspects of UML. Still, seven approaches focus on other languages (such as BPMN). This can be interpreted as, a more limited in scope, but still relevant interest into collaborative MDSE outside the UML. As reported before, while this figure is representative of the state of the art in collaborative MDSE research, it does not represent in any way the state of the practice (e.g., industrial modeling tools) that will be investigated in future work.

Another result, to be further evaluated through a state of the practice analysis, is that only four studies support the interplay between synchronous and asynchronous collaboration mechanisms (see Table 14): P22 supports an asynchronous editing with a synchronous conflict resolution, while P23, P28, and P41 support both asynchronous and synchronous mechanisms with partial locking of the whole model. Again, this does not signify that there is limited interest on the combination of synchronous and asynchronous mechanisms. Still, academic papers covering the three collaborative MDSE dimensions have little focus on this aspect.

12 Threats to Validity

In 2015 Petersen et al. proposed an up-to-date set of guidelines for conducting systematic mapping studies in software engineering [20]. In that paper the authors also proposed a check-list for objectively assessing the quality rating for systematic mapping studies. According to the metrics defined in Petersen's quality checklist, we achieved an outstanding score of 61.5%, defined as the ratio of the number of actions taken in comparison to the total number of actions reported in the quality checklist. The quality score of our study is far beyond the scores obtained by existing systematic mapping studies in the literature, which have a distribution with a median of 33% and 48% as absolute maximum value [20]. We achieved such a high level of quality by (i) carefully designing our study in advance, (ii) formalizing such a study design into a research protocol which was subject to external reviews by independent researchers, (iii) by conducting our study by carefully following well-accepted and updated guidelines of systematic mapping studies [20], [21], and (iv) by assessing, validating, and discussing the results and potential threats in each phase of the study. In the following we detail the main threats to validity of our study and how we mitigated them.

External validity. It refers to the generalizability of obtained results and findings [49]. In our study, the most severe threat related to external validity may consist of having a set of primary studies that is not representative of the whole research on collaborative MDSE. We mitigated this potential threat by following a search strategy including both automatic search and backward-forward snowballing of selected studies. Moreover, defining, iteratively refining and piloting, and validating a set of well-defined inclusion and exclusion criteria contributed to reinforce the external validity of our study. Another potential threat could have been the consideration of studies published in the English language only. However, the English language is the most widely used language for scientific papers, so this threat can be reasonably considered as minimal. Along the same lines, gray literature (e.g., white papers, non-reviewed publications or books, etc.) is not included in our research; this potential threat is intrinsic to our study design since we want to focus exclusively on the state of the art presented

in high-quality scientific studies. A further threat to external validity can be associated by the fact that, while including academic papers, we did not include tools and industrial research con collaborative MDSE. This potential threat is mitigated by avoiding to provide any conclusion that may be biased by the scientific nature of this study, and by planning a future work devoted to collaborative MDSE tools analysis.

Internal validity. It refers to the level of influence that extraneous variables may have on the design of the study. We mitigated this potential threat to validity by (i) rigorously defining and validating the protocol of our study, and (ii) defining our classification framework by carefully following the keywording process (see Section 4.3), which has been also validated using the pilot studies. Regarding the validity of the synthesis of collected data, when doing both the vertical and horizontal analysis we employed well-assessed descriptive statistics, so the threats were minimal. Furthermore, during the horizontal analysis we also cross-analyzed the values of different concepts and attributes of the classification framework in order to make a sanity test of the extracted data. This analysis helped us in identifying and fixing some minimal issues about the consistency of the extracted data.

Construct validity. It concerns the validity of extracted data with respect to the research questions. In the context of mapping studies it mainly deals with the selection of the primary studies with respect to how they really represent the population in light of the research questions. We mitigated this potential source of threats in different ways. More specifically, the automatic search has been performed on multiple electronic databases to avoid potential biases due to publishers' policies and business concerns. Also, we are reasonably confident about the construction of the search string used in our automatic search since the terms used were extracted from the research questions and refined by analyzing the set of pilot studies. Moreover, the automatic search is complemented by the snowballing activity, thus making us reasonably confident about our search strategy. After having collected all relevant studies from the automatic search, we rigorously screened them according to well-documented inclusion and exclusion criteria (see Section 4.2.2). Also, in order to assess the quality of the selection process, both principle and secondary researchers assessed a random sample of studies, and inter-researcher agreement has been statistically measured, obtained a promising Cohen-Kappa coefficient of 0.89.

Conclusion validity. It concerns the relationship between the extracted data and the obtained results. We mitigated potential threats to conclusion validity by applying well-accepted systematic methods and processes throughout our study and we documented all of them in our research protocol, so that this study can be replicated by other researchers interested in collaborative MDSE. Moreover, we are aware that other researchers may identify concepts and attributes different from the ones in our classification framework. We mitigated this potential threat by (i) letting the concepts and attributes emerge from the pilot studies and refining them throughout the data extraction activity, (ii) performing an external evaluation by independent researchers who were not involved in our research, and (iii) having the data extraction process conducted by two researchers. We also avoided to discuss results that may not be directly related to the extracted data. We therefore avoided to include any finding that, since solely extracted from research papers, may be not representative of the collaborative MDSE community.

13 Conclusions

Collaborative MDSE consists of *methods or techniques in which multiple stakeholders manage, collaborate, and are aware of each others' work on a set of shared models*. A collaborative MDSE approach is composed of 3+1 main complementary dimensions. Models are the central pillar, representing the artifact to be managed, communicated, and used for collaboration purposes. The *model management* dimension includes the editing, multi-view, and tool support components. The *collaboration* dimension brings with it versioning, branching, merging and conflict management facilities. The *communication* dimension takes into account stakeholders, awareness, and communication support.

In this study we present a systematic mapping study with the goal of identifying, classifying, and understanding existing collaborative MDSE approaches. Starting from over 3,000 potentially relevant studies, we applied a rigorous selection procedure resulting in 48 primary studies along a time span of nineteen years. Moreover, we rigorously defined a classification framework with the target of extracting from each primary study information pertaining to publication trends, characteristics, and challenges faced by researchers over the years.

After analyzing and thoroughly discussing the extracted data we obtained the following results: (i) there is a growing scientific interest on collaborative MDSE in the last years, with the majority of studies published in a (very heterogenous set of) conferences and workshops; (ii) while they are becoming prominent in model-driven software engineering, multi-view modeling, models validation, reuse, and branching are more rarely covered with respect to other aspects about collaborative MDSE; (iii) different primary studies focus differently on individual dimensions of collaborative MDSE (i.e., model management, collaboration, and communication); (iv) most approaches are metamodel-specific (i.e., they do not support ad-hoc user-defined DSMLs), with a prominence of UML-based approaches; (v) while a number of approaches focus on either synchronous or asynchronous communication means, only 4 support the interplay between them. As already mentioned in Section 11, those results while thoroughly representing the state of the research in collaborative MDSE (compatibly with our definition and protocol), do not reflect industrial practices or tools.

In addition to the previously described results we also obtained a number of interesting insights for each research question of our study, they are reported in dedicated summary boxes in Sections 5, 6, 7, 8, 9. Also, we identified perspectives of interest crossing different dimensions and aspects of collaborative MDSE, they are reported in Section 10.

What is next? If on the one hand our analysis of collaborative MDSE approaches has revealed a growing scientific interest in the topic over the years, on the other hand the

time is ripe for investigating on the real *needs* that MDSE practitioners experience while collaborating on modeling artifacts. In this study we reported on the lack or limited presence of certain collaboration features, still, we could not report on the impact due to those limitations. Such analysis will help to better shape the modeling tools of tomorrow, and their core features.

While introducing this paper, we briefly reported on existing (industrial) tools supporting collaborative MDSE. A matching of existing tools to our classification framework would reveal how existing MDSE tools support the identified collaboration dimensions as well as their maturity and adoption. Existing tools can then be mapped to the practitioners needs, so to provide a clear view on collaboration via modeling technologies.

What we consider to be in our personal wish list is the realization of a collaborative modeling framework that, while being able to host modeling artifacts such as models, metamodels, transformations, and megamodels, may provide a layer of services for the collaborative management of those artifacts. The first step towards this direction consists in defining a set of collaboration links among modeling artifacts supporting the propagation of changes among them.

To conclude: software production is more and more subject to globalization, with teams required to work distributed and with fast pace, and with stakeholders coming with different potentially conflicting concerns. Those needs impose higher degree of automation, collaboration awareness, and distributed and fast decision making. We expect that whatever a model will look like in the next few years, being them specifying a cyber-physical space, an IoT architecture, or the collaboration of smart objects in a smart city, potentially they will be managed collaboratively [72]. Each stakeholder will come with its own model kind, specialized to certain analysis or code generation. She will be able to search for a model or a transformation, combine models, evolve models and metamodels. Collaboration will be there, and automation will be required at different levels.

ACKNOWLEDGMENTS

We would like to thank the following external reviewers of this study: Dimitrios S. Kolovos (University of York, UK), Muhammad Ali Babar (University of Adelaide, Australia), and Patricia Lago (Vrije Universiteit Amsterdam, The Netherlands). They provided invaluable feedback about our research protocol. We also thank the anonymous reviewers whose suggestions substantially improved the quality of this study.

REFERENCES

[1] I. Mistrík, J. Grundy, A. van der Hoek, and J. Whitehead, "Collaborative software engineering: Challenges and prospects," in *Collaborative Software Engineering*, I. Mistrík, J. Grundy, A. Hoek, and J. Whitehead, Eds. Springer Berlin Heidelberg, 2010, pp. 389–403.

[2] I. Mistrík, J. Grundy, A. Hoek, and J. Whitehead, Eds., *Collaborative Software Engineering*. Springer Berlin Heidelberg, 2010.

[3] D. S. Kolovos, L. M. Rose, N. Matragkas, R. F. Paige, E. Guerra, J. S. Cuadrado, J. De Lara, I. Ráth, D. Varró, M. Tisi, and J. Cabot, "A research roadmap towards achieving scalability in model driven engineering," in *Proceedings of the Workshop on Scalability in Model Driven Engineering*, ser. BigMDE '13. New York, NY, USA: ACM, 2013, pp. 2:1–2:10, http://doi.acm.org/10.1145/2487766.2487768.

[4] M. Brambilla, J. Cabot, and M. Wimmer, *Model-driven software engineering in practice*. Morgan & Claypool Publishers, 2012, vol. 1, no. 1.

[5] D. Schuler and A. Namioka, Eds., *Participatory Design: Principles and Practices*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1993.

[6] K. Vredenburg, J.-Y. Mao, P. W. Smith, and T. Carey, "A survey of user-centered design practice," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '02. New York, NY, USA: ACM, 2002, pp. 471–478, http://doi.acm.org/10.1145/503376.503460.

[7] B. Bruegge and A. H. Dutoit, *Object-Oriented Software Engineering Using UML, Patterns and Java-(Required)*. Prentice Hall, 2004.

[8] P. J. Denning, "Design thinking," *Commun. ACM*, vol. 56, no. 12, pp. 29–31, Dec. 2013, http://doi.acm.org/10.1145/2535915.

[9] B. Selic, "The pragmatics of model-driven development," *IEEE Softw.*, vol. 20, no. 5, pp. 19–25, Sep. 2003, http://dx.doi.org/10.1109/MS.2003.1231146.

[10] J. L. C. Izquierdo and J. Cabot, "Community-driven language development," in *2012 4th International Workshop on Modeling in Software Engineering (MISE)*. IEEE, 2012, pp. 29–35.

[11] J. Di Rocco, D. Di Ruscio, L. Iovino, and A. Pierantonio, "Collaborative repositories in model-driven engineering," *IEEE Software*, vol. 32, no. 3, pp. 28–34, May 2015.

[12] M. Maróti, T. Kecskés, R. Kereskényi, B. Broll, P. Völgyesi, L. Jurácz, T. Levendoszky, and Á. Lédeczi, "Next generation (meta) modeling: Web-and cloud-based collaborative tool infrastructure," *Proceedings of MPM*, p. 41, 2014.

[13] E. Syriani, H. Vangheluwe, R. Mannadiar, C. Hansen, S. Van Mierlo, and H. Ergin, "Atompm: A web-based modeling environment." in *Demos/Posters/StudentResearch@ MoDELS*. Citeseer, 2013, pp. 21–25.

[14] M. Farwick, B. Agreiter, J. White, S. Forster, N. Lanzanasto, and R. Breu, *A web-based collaborative metamodeling environment with secure remote model access*. Springer, 2010.

[15] C. Thum, M. Schwind, and M. Schader, "Slim - a lightweight environment for synchronous collaborative modeling," in *Model Driven Engineering Languages and Systems*. Springer, 2009, pp. 137–151.

[16] M. Cataldo, C. Shelton, Y. Choi, Y.-Y. Huang, V. Ramesh, D. Saini, and L.-Y. Wang, "Camel: a tool for collaborative distributed software design," in *ICGSE 2009. Fourth IEEE International Conference on Global Software Engineering, 2009.* IEEE, 2009, pp. 83–92.

[17] B. Bruegge, O. Creighton, J. Helming, and M. Kögel, "Unicase–an ecosystem for unified software engineering research tools," in *Third IEEE International Conference on Global Software Engineering, ICGSE*, vol. 2008. Citeseer, 2007.

[18] A. De Lucia, F. Fasano, G. Scanniello, and G. Tortora, "Enhancing collaborative synchronous uml modelling with fine-grained versioning of software artefacts," *Journal of Visual Languages & Computing*, vol. 18, no. 5, pp. 492–503, 2007.

[19] S. Kelly, K. Lyytinen, and M. Rossi, "Metaedit+ a fully configurable multi-user and multi-tool case and came environment," in *Advanced Information Systems Engineering*. Springer, 1996, pp. 1–21.

[20] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and Software Technology*, vol. 64, pp. 1–18, 2015.

[21] B. A. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," 2007.

[22] J. Bézivin, "On the unification power of models," *Softw Syst Model*, vol. 4, no. 2, pp. 171–188, may 2005, http://dx.doi.org/10.1007/s10270-005-0079-0.

[23] D. Di Ruscio, R. F. Paige, and A. Pierantonio, "Guest Editorial to the Special Issue on Success Stories in Model Driven Engineering," *Sci. Comput. Program.*, vol. 89, no. PB, pp. 69–70, 2014, http://dx.doi.org/10.1016/j.scico.2013.12.006.

[24] M. Brambilla and P. Fraternali, "Large-scale model-driven engineering of web user interaction: The webml and webratio experience," *Science of Computer Programming*, vol. 89, pp. 71–87, 2014.

[25] J. Davies, J. Gibbons, J. Welch, and E. Crichton, "Model-driven engineering of information systems: 10 years and 1000 versions," *Science of Computer Programming*, vol. 89, pp. 88–104, 2014.

[26] A. Nadas, T. Levendovszky, E. K. Jackson, I. Madari, and J. Sztipanovits, "A model-integrated authoring environment for privacy

policies," *Science of Computer Programming*, vol. 89, pp. 105–125, 2014.

[27] J. Davies, J. Gibbons, S. Harris, and C. Crichton, "The cancergrid experience: metadata-based model-driven engineering for clinical trials," *Science of Computer Programming*, vol. 89, pp. 126–143, 2014.

[28] J. Hutchinson, J. Whittle, and M. Rouncefield, "Model-driven engineering practices in industry: Social, organizational and managerial factors that lead to success or failure," *Science of Computer Programming*, vol. 89, pp. 144–161, 2014.

[29] F. Büttner, U. Bartels, L. Hamann, O. Hofrichter, M. Kuhlmann, M. Gogolla, L. Rabe, F. Steimke, Y. Rabenstein, and A. Stosiek, "Model-driven standardization of public authority data interchange," *Science of Computer Programming*, vol. 89, pp. 162–175, 2014.

[30] J. S. Cuadrado, J. L. C. Izquierdo, and J. G. Molina, "Applying model-driven engineering in small software enterprises," *Science of Computer Programming*, vol. 89, pp. 176–198, 2014.

[31] E. Dubois, C. Bortolaso, D. Appert, and G. Gauffre, "An mde-based framework to support the development of mixed interactive systems," *Science of Computer Programming*, vol. 89, pp. 199–221, 2014.

[32] J. Whittle, J. Hutchinson, M. Rouncefield, H. Burden, and R. Heldal, "Industrial adoption of model-driven engineering: Are the tools really the problem?" in *Lecture Notes in Computer Science*. Springer Science Business Media, 2013, pp. 1–17.

[33] D. Tofan, M. Galster, and P. Avgeriou, "Difficulty of architectural decisions a survey with professional architects," in *Software Architecture*, ser. Lecture Notes in Computer Science, K. Drira, Ed. Springer Berlin Heidelberg, 2013, vol. 7957, pp. 192–199.

[34] S. Rekha and H. Muccini, "Suitability of software architecture decision making methods for group decisions," in *Software Architecture*. Springer, 2014, pp. 17–32.

[35] *ISO/IEC/IEEE 42010, Systems and software engineering — Architecture description*, ISO, December 2011.

[36] D. A. Tamburri, P. Lago, and H. van Vliet, "Organizational social structures for software engineering," *ACM Computing Surveys*, pp. 1–35, 2012.

[37] N. Nagappan, B. Murphy, and V. Basili, "The influence of organizational structure on software quality: an empirical case study," in *International conference on Software engineering*. Leipzig, Germany.: IEEE, May 2008, pp. 521–530.

[38] J. Whitehead, "Collaboration in software engineering: A roadmap," in *2007 Future of Software Engineering*. IEEE Computer Society, 2007, pp. 214–225.

[39] B. Kitchenham, R. Pretorius, D. Budgen, O. P. Brereton, M. Turner, M. Niazi, and S. Linkman, "Systematic literature reviews in software engineering–a tertiary study," *Information and Software Technology*, vol. 52, no. 8, pp. 792–805, 2010.

[40] R. G. C. Rocha, C. Costa, C. M. de Oliveira Rodrigues, R. R. de Azevedo, I. H. de Farias Junior, S. R. de Lemos Meira, and R. Prikladnicki, "Collaboration models in distributed software development: a systematic review." *CLEI Electron. J.*, vol. 14, no. 2, 2011.

[41] J. Portillo-Rodríguez, A. Vizcaíno, M. Piattini, and S. Beecham, "Tools used in global software engineering: A systematic mapping review," *Information and Software Technology*, vol. 54, no. 7, pp. 663–685, 2012.

[42] M. Renger, G. L. Kolfschoten, and G.-J. De Vreede, "Challenges in collaborative modelling: a literature review and research agenda," *International Journal of Simulation and Process Modelling*, vol. 4, no. 3, pp. 248–263, 2008, http://dx.doi.org/10.1504/IJSPM.2008.023686.

[43] A. Marques, R. Rodrigues, and T. Conte, "Systematic literature reviews in distributed software development: A tertiary study," in *Global Software Engineering (ICGSE), 2012 IEEE Seventh International Conference on*, Aug 2012, pp. 134–143.

[44] K. Dullemond, B. van Gameren, and R. Van Solingen, "Collaboration spaces for virtual software teams," *Software, IEEE*, vol. 31, no. 6, pp. 47–53, 2014.

[45] F. Lanubile, C. Ebert, R. Prikladnicki, and A. Vizcaíno, "Collaboration tools for global software engineering," *IEEE software*, vol. 27, no. 2, p. 52, 2010.

[46] D. Di Ruscio, M. Franzago, H. Muccini, and I. Malavolta, "Envisioning the future of collaborative model-driven software engineering," in *Proceedings of the 39th International Conference on Software Engineering Companion*, ser. ICSE-C '17. IEEE Press, 2017, pp. 219–221.

[47] P. Grnbacher and Y. Ledru, "Automated software engineering: introduction," *ERCIM News*, vol. 58, July 2004.

[48] M. Maróti, R. Kereskényi, T. Kecskés, P. Völgyesi, and A. Lédeczi, "Online collaborative environment for designing complex computational systems," *Procedia Computer Science*, vol. 29, pp. 2432–2441, 2014.

[49] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*, ser. Computer Science. Springer, 2012.

[50] M. Franzago, D. Di Ruscio, I. Malavolta, and H. Muccini, "Protocol for a Systematic Mapping Study on Collaborative Model-Driven Software Engineering," DISIM - University of L'Aquila, Tech. Rep. TR-001-2016, https://arxiv.org/pdf/1611.02619.pdf.

[51] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, London, England, United Kingdom, May 13-14, 2014*, 2014, p. 38, http://doi.acm.org/10.1145/2601248.2601268.

[52] V. R. Basili, G. Caldiera, and H. D. Rombach, "The goal question metric approach," in *Encyclopedia of Software Engineering*. Wiley, 1994.

[53] H. Zhang, M. A. Babar, and P. Tell, "Identifying relevant studies in software engineering," *Inf. Softw. Technol.*, vol. 53, no. 6, pp. 625–637, Jun. 2011, http://dx.doi.org/10.1016/j.infsof.2010.12.010.

[54] L. Chen, M. A. Babar, and H. Zhang, "Towards an evidence-based understanding of electronic data sources," in *Proceedings of the 14th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE'10. Swinton, UK, UK: British Computer Society, 2010, pp. 135–138, http://dl.acm.org/citation.cfm?id=2227057.2227074.

[55] M. Kuhrmann, D. M. Fernández, and M. Daneva, "On the pragmatic design of literature studies in software engineering: an experience-based guideline," *Empirical Software Engineering*, pp. 1–40, 2016.

[56] S. Jalali and C. Wohlin, "Systematic literature studies: database searches vs. backward snowballing," in *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*. ACM, 2012, pp. 29–38.

[57] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE'08. Swinton, UK, UK: British Computer Society, 2008, pp. 68–77, http://dl.acm.org/citation.cfm?id=2227115.2227123.

[58] D. Spencer, *Card sorting: Designing usable categories*. Rosenfeld Media, 2009.

[59] D. S. Cruzes and T. Dybå, "Research synthesis in software engineering: A tertiary study," *Information and Software Technology*, vol. 53, no. 5, pp. 440–455, 2011.

[60] J. D. Rocco, D. D. Ruscio, A. Pierantonio, J. S. Cuadrado, J. de Lara, and E. Guerra, "Using ATL transformation services in the mdeforge collaborative modeling platform," in *Theory and Practice of Model Transformations - 9th International Conference, ICMT 2016, Held as Part of STAF 2016, Vienna, Austria, July 4-5, 2016, Proceedings*, 2016, pp. 70–78.

[61] A. Lajmi, J. Martinez, and T. Ziadi, "DSLFORGE: textual modeling on the web," in *Proceedings of the Demonstrations Track of the ACM/IEEE 17th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2014), Valencia, Spain, October 1st and 2nd, 2014.*, 2014.

[62] C. C. Manzanares, J. S. Cuadrado, and J. de Lara, "Building MDE cloud services with Distil," in *Proceedings of the 3rd International Workshop on Model-Driven Engineering on and for the Cloud 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2015), Ottawa, Canada, September 29, 2015.*, 2015, pp. 19–24.

[63] H. Bruneliere, J. Cabot, and F. Jouault, "Combining model-driven engineering and cloud computing," in *Modeling, Design, and Analysis for the Service Cloud-MDA4ServiceCloud'10: Workshop's 4th edition (co-located with the 6th European Conference on Modelling Foundations and Applications-ECMFA 2010)*, 2010.

[64] F. Basciani, J. Di Rocco, D. Di Ruscio, A. Di Salle, L. Iovino, and A. Pierantonio, "Mdeforge: an extensible web-based modeling platform," in *Proceedings of the 2nd International Workshop on Model-Driven Engineering on and for the Cloud co-located with the 17th International Conference on Model Driven Engineering Languages and*

Systems, CloudMDE@MoDELS 2014, Valencia, Spain, September 30, 2014., 2014, pp. 66–75.

[65] A. Cicchetti, F. Ciccozzi, and T. Leveque, "A hybrid approach for multi-view modeling," *Electronic Communications of the EASST*, vol. 50, 2012.

[66] C. Gutwin and S. Greenberg, "A descriptive framework of workspace awareness for real-time groupware," *Computer Supported Cooperative Work (CSCW)*, vol. 11, no. 3-4, pp. 411–446, 2002.

[67] J. Whitehead, I. Mistrík, J. Grundy, and A. van der Hoek, "Collaborative software engineering: Concepts and techniques," in *Collaborative Software Engineering*. Springer, 2010, pp. 1–30.

[68] D. S. Kolovos, L. M. Rose, N. Matragkas, R. F. Paige, E. Guerra, J. S. Cuadrado, J. De Lara, I. Ráth, D. Varró, M. Tisi *et al.*, "A research roadmap towards achieving scalability in model driven engineering," in *Proceedings of the Workshop on Scalability in Model Driven Engineering*. ACM, 2013, p. 2.

[69] I. Malavolta, P. Lago, H. Muccini, P. Pelliccione, and A. Tang, "What industry needs from architectural languages: A survey," *IEEE Trans. Software Eng.*, vol. 39, no. 6, pp. 869–891, 2013, http://dx.doi.org/10.1109/TSE.2012.74.

[70] P. Lago, I. Malavolta, H. Muccini, P. Pelliccione, and A. Tang, "The Road Ahead for Architectural Languages," *Software, IEEE*, vol. 32, no. 1, pp. 98–104, 2015.

[71] R. F. Paige, P. J. Brooke, and J. S. Ostroff, "Metamodel-based model conformance and multiview consistency checking," *ACM Trans. Softw. Eng. Methodol.*, vol. 16, no. 3, Jul. 2007, http://doi.acm.org/10.1145/1243987.1243989.

[72] F. Ciccozzi, I. Crnkovic, D. Di Ruscio, I. Malavolta, P. Pelliccione, and R. Spalazzese, " Model-Driven Engineering for Mission-Critical IoT Systems ," *IEEE Software*, vol. 34, no. 1, pp. 46–53, Jan 2017.

[73] H. Zhang and M. A. Babar, "Systematic reviews in software engineering: An empirical investigation," *Information and Software Technology*, vol. 55, no. 7, p. 1341 1354, 2013.

SELECTED PRIMARY STUDIES

[P1] C. Barlelt, G. Molter, and T. Schumann, "A model repository for collaborative modeling with the jazz development platform," in *HICSS'09. 42nd Hawaii International Conference on System Sciences, 2009.* IEEE, 2009, pp. 1–10.

[P2] D. Wüest, N. Seyff, and M. Glinz, "Flexisketch team: collaborative sketching and notation creation on the fly," in *Proceedings of the 37th International Conference on Software Engineering-Volume 2*. IEEE Press, 2015, pp. 685–688.

[P3] T. N. Graham, H. D. Stewart, A. R. Kopaee, A. G. Ryman, and R. Rasouli, "A world-wide-web architecture for collaborative software design," in *Software Technology and Engineering Practice, 1999. STEP'99. Proceedings*. IEEE, 1999, pp. 22–29.

[P4] A. García Frey, J.-S. Sottet, and A. Vagner, "Ame: an adaptive modelling environment as a collaborative modelling tool," in *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems*. ACM, 2014, pp. 189–192.

[P5] P. Sriplakich, X. Blanc, and M.-P. Gervais, "Collaborative software engineering on large-scale models: requirements and experience in modelbus," in *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 2008, pp. 674–681.

[P6] S. Erol and G. Neumann, "A case-study of wiki-supported collaborative drafting of business processes models," in *2013 IEEE 15th Conference on Business Informatics*. IEEE, 2013, pp. 382–390.

[P7] M. Dirix, A. Muller, and V. Aranega, "Genmymodel: an online uml case tool," in *ECOOP - European Conferences on Object-Oriented Programming*, 2013.

[P8] M. Elaasar and J. Conallen, "Design management: a collaborative design solution," in *European Conference on Modelling Foundations and Applications*. Springer, 2013, pp. 165–178.

[P9] M. Cataldo, C. Shelton, Y. Choi, Y.-Y. Huang, V. Ramesh, D. Saini, and L.-Y. Wang, "Camel: a tool for collaborative distributed software design," in *ICGSE 2009. Fourth IEEE International Conference on Global Software Engineering, 2009.* IEEE, 2009, pp. 83–92.

[P10] C. Thum, M. Schwind, and M. Schader, "Slim - a lightweight environment for synchronous collaborative modeling," in *Model Driven Engineering Languages and Systems*. Springer, 2009, pp. 137–151.

[P11] M. Farwick, B. Agreiter, J. White, S. Forster, N. Lanzanasto, and R. Breu, *A web-based collaborative metamodeling environment with secure remote model access*. Springer, 2010.

[P12] S. Kelly, K. Lyytinen, and M. Rossi, "Metaedit+ a fully configurable multi-user and multi-tool case and came environment," in *Advanced Information Systems Engineering*. Springer, 1996, pp. 1–21.

[P13] M. Maróti, T. Kecskés, R. Kereskényi, B. Broll, P. Völgyesi, L. Jurácz, T. Levendovszky, and Á. Lédeczi, "Next generation (meta) modeling: Web-and cloud-based collaborative tool infrastructure." in *MPM@ MoDELS*, 2014, pp. 41–60.

[P14] M. Franzago, H. Muccini, and I. Malavolta, "Towards a collaborative framework for the design and development of data-intensive mobile applications," in *Proceedings of the 1st International Conference on Mobile Software Engineering and Systems*. ACM, 2014, pp. 58–61.

[P15] S. Lili and S. R. Sutarsa, "Mue: Multi user uml editor," in *Information and Communication Technology Seminar*, 2005, p. 41.

[P16] E. Syriani, H. Vangheluwe, R. Mannadiar, C. Hansen, S. Van Mierlo, and H. Ergin, "Atompm: A web-based modeling environment." in *Demos/Posters/StudentResearch@ MoDELS*. Citeseer, 2013, pp. 21–25.

[P17] J. young Bang, D. Popescu, G. Edwards, N. Medvidovic, N. Kulkarni, G. M. Rama, and S. Padmanabhuni, "Codesign: a highly extensible collaborative software modeling framework," in *2010 ACM/IEEE 32nd International Conference on Software Engineering*, vol. 2. IEEE, 2010, pp. 243–246.

[P18] V. M. Penichet, J. A. Gallud, R. Tesoriero, and M. Lozano, "Design and evaluation of a service oriented architecture-based application to support the collaborative edition of uml class diagrams," in *International Conference on Computational Science*. Springer, 2008, pp. 389–398.

[P19] V. Genaro Motti, D. Raggett, S. Van Cauwelaert, and J. Vanderdonckt, "Simplifying the development of cross-platform web user interfaces by collaborative model-based design," in *Proceedings of the 31st ACM international conference on Design of communication*. ACM, 2013, pp. 55–64.

[P20] B. Bruegge, A. H. Dutoit, and T. Wolf, "Sysiphus: Enabling informal collaboration in global software development," in *2006 IEEE International Conference on Global Software Engineering (ICGSE'06)*. IEEE, 2006, pp. 139–148.

[P21] B. Bruegge, O. Creighton, J. Helming, and M. Kögel, "Unicase–an ecosystem for unified software engineering research tools," in *Third IEEE International Conference on Global Software Engineering, ICGSE*, vol. 2008. Citeseer, 2007.

[P22] P. Brosch, M. Seidl, K. Wieland, M. Wimmer, and P. Langer, "We can work it out: Collaborative conflict resolution in model versioning," in *ECSCW 2009*. Springer, 2009, pp. 207–214.

[P23] A. Cicchetti, H. Muccini, P. Pelliccione, and A. Pierantonio, "Towards a framework for distributed and collaborative modeling," in *WETICE'09. 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises.* IEEE, 2009, pp. 149–154.

[P24] H.-m. Cai, X.-f. Ji, and F.-l. Bu, "Research of consistency maintenance mechanism in real-time collaborative multi-view business modeling," *Journal of Shanghai Jiaotong University (Science)*, vol. 20, pp. 86–92, 2015.

[P25] C. Cook and N. Churcher, "Constructing real-time collaborative software engineering tools using caise, an architecture for supporting tool development," in *Proceedings of the 29th Australasian Computer Science Conference-Volume 48*. Australian Computer Society, Inc., 2006, pp. 267–276.

[P26] D. Xu, J. Kurogi, Y. Ohgame, and A. Hazeyama, "Distributed collaborative modeling support system associating uml diagrams with chat messages," in *33rd Annual IEEE International Computer Software and Applications Conference*, vol. 1. IEEE, 2009, pp. 367–372.

[P27] N. Boulila, "Group support for distributed collaborative concurrent software modeling," in *19th International Conference on Automated Software Engineering, 2004. Proceedings.* IEEE, 2004, pp. 422–425.

[P28] S. Krusche and B. Bruegge, "Model-based real-time synchronization," in *International Workshop on Comparison and Versioning of Software Models (CVSM14)*, 2014.

[P29] M. C. Pichiliani and C. M. Hirata, "A guide to map application components to support multi-user real-time collaboration," in *International Conference on Collaborative Computing: Networking,*

Applications and Worksharing. IEEE, 2006, pp. 1–5.
[P30] P. Rittgen, "Collaborative modeling: A design science approach," in *Hawaii International Conference on System Sciences (HICSS)*, 2009, pp. 1–10.
[P31] N. Baloian, G. Zurita, F. M. Santoro, R. M. Araujo, S. Wolfgan, D. Machado, and J. A. Pino, "A collaborative mobile approach for business process elicitation," in *Computer Supported Cooperative Work in Design (CSCWD), 15th International Conference on*. IEEE, 2011, pp. 473–480.
[P32] N. Baghaei, A. Mitrovic, and W. Irwin, "Supporting collaborative learning and problem-solving in a constraint-based cscl environment for uml class diagrams," *International Journal of Computer-Supported Collaborative Learning*, vol. 2, no. 2-3, pp. 159–190, 2007.
[P33] A. A. Koshima and V. Englebert, "Collaborative editing of emf/ecore meta-models and models: Conflict detection, reconciliation, and merging in dicomef," *Science of Computer Programming*, vol. 113, pp. 3–28, 2015.
[P34] A. De Lucia, F. Fasano, G. Scanniello, and G. Tortora, "Enhancing collaborative synchronous uml modelling with fine-grained versioning of software artefacts," *Journal of Visual Languages & Computing*, vol. 18, no. 5, pp. 492–503, 2007.
[P35] V. Kulkarni, S. Reddy, and A. Rajbhoj, "Scaling up model driven engineering–experience and lessons learnt," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2010, pp. 331–345.
[P36] A. Mougenot, X. Blanc, and M.-P. Gervais, "D-praxis: A peer-to-peer collaborative model editing framework," in *Distributed Applications and Interoperable Systems*. Springer, 2009, pp. 16–29.
[P37] J. Michaux, X. Blanc, M. Shapiro, and P. Sutra, "A semantically rich approach for collaborative model edition," in *Proceedings of the ACM Symposium on Applied Computing*. ACM, 2011, pp. 1470–1475.
[P38] A. De Lucia, F. Fasano, G. Scanniello, and G. Tortora, "Concurrent fine-grained versioning of uml models," in *Software Maintenance and Reengineering, 2009. CSMR'09. 13th European Conference on*. IEEE, 2009, pp. 89–98.
[P39] J. Gallardo, C. Bravo, and M. A. Redondo, "A model-driven development method for collaborative modeling tools," *Journal of Network and Computer Applications*, vol. 35, no. 3, pp. 1086–1105, 2012.
[P40] R. Duque, J. Gallardo, C. Bravo, and A. J. Mendes, "Defining tasks, domains and conversational acts in cscw systems: the space-design case study." *J. UCS*, vol. 14, no. 9, pp. 1463–1479, 2008.
[P41] N. Zhu, J. Grundy, J. Hosking, N. Liu, S. Cao, and A. Mehra, "Pounamu: A meta-tool for exploratory domain-specific visual language tool development," *Journal of Systems and Software*, vol. 80, no. 8, pp. 1390–1407, 2007.
[P42] L. Murta, H. Oliveira, C. Dantas, L. G. Lopes, and C. Werner, "Odyssey-scm: An integrated software configuration management infrastructure for uml models," *Science of Computer Programming*, vol. 65, no. 3, pp. 249–274, 2007.
[P43] K. M. Hansen and C. H. Damm, "Building flexible, distributed collaboration tools using type-based publish/subscribe-the distributed knight case." in *IASTED Conf. on Software Engineering*, 2004, pp. 595–600.
[P44] S. Forster, J. Pinggera, and B. Weber, "Collaborative business process modeling." in *EMISA*, vol. 206. Citeseer, 2012, pp. 81–94.
[P45] C. D. Francescomarino, C. Ghidini, M. Rospocher, L. Serafini, and P. Tonella, "A framework for the collaborative specification of semantically annotated business processes," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 23, no. 4, pp. 261–295, 2011.
[P46] M. Breu, R. Breu, and S. Löw, "Moveing forward: Towards an architecture and processes for a living models infrastructure," *Int. J. Adv. Life Sci*, vol. 3, 2011.
[P47] J. Y. Bang and N. Medvidovic, "Proactive detection of higher-order software design conflicts," in *Software Architecture (WICSA), 2015 12th Working IEEE/IFIP Conference on*. IEEE, 2015, pp. 155–164.
[P48] K. Wieland, P. Langer, M. Seidl, M. Wimmer, and G. Kappel, "Turning conflicts into collaboration," *Computer Supported Cooperative Work (CSCW)*, vol. 22, no. 2-3, pp. 181–240, 2013.

Mirco Franzago is a PhD student in Information and Communication Technologies (Software Engineering and Intelligent Systems curriculum) at the Department of Information Engineering, Computer Science and Mathematics (DISIM) - University of L'Aquila, Italy. His research focuses on collaborative software engineering, model-driven engineering (MDE) and mobile enabled systems, in particular on how MDE techniques can be exploited to support stakeholders' collaboration during the design and development of complex and mobile-enabled software systems. He is program committee member of MOBILESoft International Conference, COMMitMDE International Workshop, and reviewer for the major conferences and workshops in his fields of interest (ICSE, ASE, MoDELS, and others).

Davide Di Ruscio is Assistant Professor at the Department of Information Engineering Computer Science and Mathematics of the University of L'Aquila. His main research interests are related to several aspects of Model Driven Engineering (MDE) including domain specific modelling languages, model transformation, model differencing, model evolution, and coupled evolution. He has published more than 100 papers in various journals, conferences and workshops on such topics. He has been co-guest editor of a number of journal special issues. He has been in the PC and involved in the organization of several workshops and conferences, and reviewer of many journals like IEEE Transactions on Software Engineering, Science of Computer Programming, Software and Systems Modeling, and Journal of Systems and Software. He is member of the steering committee of the International Conference on Model Transformation (ICMT), of the Software Language Engineering (SLE) conference, of the Seminar Series on Advanced Techniques & Tools for Software Evolution (SATTOSE), and of the Workshop on Modelling in Software Engineering at ICSE (MiSE). More information is available at http://www.di.univaq.it/diruscio.

Ivano Malavolta is Assistant Professor at the Vrije Universiteit Amsterdam, The Netherlands, Department of Computer Science, Faculty of Sciences. He co-organized all the editions of the international workshop on collaborative modelling in MDE (COMMitMDE), co-located with the MODELS 2016 conference. His research focuses on software architecture, model-driven engineering (MDE), and mobile-enabled systems, especially how MDE techniques can be exploited for architecting complex and mobile-enabled software systems at the right level of abstraction. Recently, he is applying empirical methods to assess practices and trends in the field of software engineering. He authored more than 60 papers in international journals and peer-reviewed international conferences proceedings; they include articles published in the IEEE Transactions on Software Engineering (TSE) and the International Conference on Software Engineering (ICSE), which are considered the leading journal and conference in the field of software engineering, respectively. He received a PhD in computer science from the University of L'Aquila in 2012. He is a member of ACM and IEEE. More information is available at http://www.ivanomalavolta.com.

Henry Muccini is an Associate Professor in Software Engineering from the University of L'Aquila, Italy. He received is PhD degree from the University of Rome La Sapienza in 2002, and he has been visiting professor at the University of California, Irvine. My research interests are in the Software Engineering field, and more specifically, on software architecture descriptions and analysis, model driven engineering, and engineering Cyber-Physical Systems. Henry is General Chair of MOBILESoft 2017, and has been co-chair of the Program Committee of WICSA 2016 (the 13th Working IEEE/IFIP Conference on Software Architecture), as well as of EUROMICRO SEAA and QSIC in 2012. He is the theme issue editorial board member of IEEE Software, a member of the IFIP WG 2.10 on Software Architecture, and the responsible of the CINI laboratory on Smart Cities and Communities, as well as of the Living Lab, at the University of L'Aquila. He is the person in charge of two international double degree master programmes in Computer Science and Software Engineering. More detailed information may be found at http://www.HenryMuccini.com and https://it.linkedin.com/in/henrymuccini.

TABLE 28: Identified limitations and shortcomings according to the C-MDSE taxonomy

Limitations and shortcomings	#Studies	Studies	Related concepts of the classification framework
Conflicts management	15	P1, P5, P6, P11, P13, P16, P17, P22, P23, P24, P28, P29, P42, P43, P48	Collaboration.ConflictDetection.ConflictResolution
Model *synchronization* and change propagation	9	P4, P8, P11, P19, P21, P29, P38, P43, P46	Collaboration.VersioningSupport.ModelMerging
Constraints specification, enforcing, and conformance checks	5	P11, P13, P33, P42, P46	ModelManagement.Editor.validationSupport
Support for *collaboration workflows* and integration with development *process*	4	P19, P21, P33, P39	Collaboration.SharedWorkspace.prescribed_workflow
Tool improvement	4	P9, P12, P28, P40	Communication.WorkspaceAwareness.AwarenessTool, ModelManagement.Editor
Support for *versioning* and models editing history	3	P10, P11, P48	Collaboration.VersioningSupport
Support for combined asynchronous and synchronous collaboration	2	P16, P37	Collaboration.collaboration_type
Independence from any specific modeling language	2	P6, P42	ModelManagement.ModelingLanguage
Interoperability with *external tools*	2	P9, P12	ModelManagement.Editor.ConcreteSyntaxType
Better *coverage* of modeling languages concepts	2	P10, P36	ModelManagement.ModelingLanguage.LanguageCustomizationMean
Support for *audio communication*	2	P31, P44	Communication.CommunicationSupport

TABLE 29: Identified limitations and shortcomings (qualitative aspects)

Qualitative aspects of C-MDSE	#Studies	Studies	Examples (paper fragments)
Usability	7	P4, P24, P31, P39, P44, P45, P47	▲ *. . . evaluation techniques of the so called groupware usability to test the suitability of AMEs with real stakeholders . . . Remote collaboration and inclusion of visualization techniques for better interaction. (P4)* ▲ *Operation undo has not been taken into account for the moment. It requires further study and efforts. (P24)* ▲ *Students reported that the tool usability could be improved, so as to allow analysts to better follow the dynamics of interviews. Some students, for instance, decided to use different tools for documenting the interview, due to their difficulties in using NetSketcher accordingly . . . and the use of "drag and drop" for choosing and placing process elements into the drawing space. (P31)* ▲ *. . . the development of more complex editors is not one of our priorities, although we consider it to be one of our objectives for the improvement of the method. (P39)* ▲ *. . . we plan to extend our prototype with additional features to increase the usability of the tool. An example would be the integration of speech. This would complement the chat window with a convenient way of communicating with other participants and approaches a face-to-face interaction. (P44)* ▲ *We will also improve the usability of the tool, thanks to the execution of further case studies and experiments. (P45)* ▲ *. . . exploring different ways of delivering feedback to architects, the effect of variations in the immediacy with which feedback is delivered. (P47)*
Performance improvement	3	P31, P42, P43	▲ *Students suggested ideas for improving NetSketcher, such as: improvements in gesture recognition for process elements - it seems that it needs to be faster in order to allow them to follow the interview speed (P31)* ▲ *limitations of the Odyssey-SCM current release regards configuration constraints, partial check-outs, and workspace caching (P42)* ▲ *If larger amounts of data were to be transmitted, or a large number of clients were to be connected it would be necessary to reconsider . . . A major reason for choosing the current strategy is, however, the simplicity and flexibility of decentralization (P43)*
Scalability improvement	3	P12, P42, P43	▲ *First, it does not address the need for multiple distributed repositories which is typical for large scale software development. Second, its concurrency management strategies can be too demanding for large scale software repositories. (P12)* ▲ *Currently, the server layer is centralized. The adoption of a distributed server layer, is left for future work. (P42)* ▲ *A major reason for choosing the current strategy is, however, the simplicity and flexibility of decentralization (P43)*

APPENDIX A
RESEARCH TEAM

Four researchers carried on this study so that potential biases have been controlled [73]. Each researcher had a specific role within the team; these are identified roles:

1) *Principle researcher*: PhD student with knowledge about model-driven engineering and development; he performed the majority of the planning and conducting activities of the study, and provided support in the reporting activity;
2) *Secondary researcher*: post-doctoral researcher with expertise in both SLR methodologies and model-driven engineering; he has been mainly involved in (i) the planning phase of the study, and (ii) supporting the principle researcher during the whole study, e.g., by reviewing the data extraction form, selected primary studies, extracted data, produced reports, etc.; also, he has been in charge of driving the reporting phase of the study;
3) *MDE expert*: senior researcher with a several-years expertise on model-driven engineering methods and techniques; he has been mainly involved in (i) supporting the principle and secondary researchers with respect to any potential issue or discussion related to the MDE methodology, (ii) participating to the reporting phase of the study;
4) *Advisor*: senior researcher with many-years expertise in software engineering and model-based design and development. He made final decisions on conflicts and options to 'avoid endless discussions' [73], and supported the other researchers during the data synthesis, findings synthesis, and report writing activities.