Model Driven Development Success Cases for Domain-Specific and General Purpose Approaches

A Systematic Mapping

Thiago Gottardi[1] and Rosana T. V. Braga[1]

Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo
{gottardi,rtvb}@icmc.usp.br

Abstract. In the context of a Model Driven Software Engineering research effort, it was identified the need of surveying the success cases categorized by specific domains and by general purpose development processes. However, no systematic mapping related to this specific context was found. Therefore, we have conducted a systematic mapping with two objectives. The first objective was to identify specific domains in which MDSE is successful, while the second objective was to identify what are the challenges to apply this methodology to general purpose development processes. As results, we have identified that MDSE success cases are clustered into four domains: business information systems, network system design, web software applications and embedded systems. We could only identify five studies related to general purpose approaches and their challenges. The analysis of the results indicate that MDSE application is consolidated in specific domains. A common feature identified among studies related to general purpose processes is that their authors have reported a lack of methodologies that support MDSE in software projects since the inception phase. This secondary study was also the first to be conducted using a collaborative systematic mapping tool.

1 Introduction

Model Driven Software Engineering (MDSE) is a specific case of Model Driven Development (MDD) applied to software engineering. In these development methods, models are active artifacts during the software engineering process. Therefore, these models are not only used to describe designs and concepts, they can also be employed to drive the software development [5].

In order to develop software using MDSE, it is necessary to use modeling languages that allow the creation of machine readable models, which, in turn, can be executed or transformed into the final software. In this manner, it is possible to replace the source-code by models that may represent the software in higher abstraction levels, tightening the semantic gap between problem and its software solution [9].

Besides adequate modeling languages, development teams also require specific tools to edit, validate and transform models, which may be specific to the project or domain. Then, it is possible to categorize development teams that use MDSE into two groups. The first group is composed by developers working on specific domains that have suitable tools to use MDSE since the inception of their development projects. However, the second group works on domains with no previous tool definition, so they must either adopt general purpose or develop their own MDSE tools by adopting methodologies, for instance, MDA (Model-Driven Architecture) [13].

We have conducted a secondary study with the objective of identifying success cases in both domain specific and general purpose MDSE approaches. Therefore, the objective was to answer questions that relate to both of these groups.

Despite knowing that MDSE has been used successfully in several domains, we believe that the state-of-the art should be made much clearer. The identified related secondary studies are not recent and are not based on systematic literature review approaches. Therefore, the main aim of this systematic mapping is to capture the distribution of domains and discuss challenges related to MDSE.

This work contributes to the study of MDSE approaches by identifying studies in a range of twenty-nine years of publications, categorizing success cases and summarizing a set of identified challenges.

This paper is structured as follows: in Section 2 there is a summary of related works; Section 3 is used to present the adopted review process; the study results are presented in Section 4; Section 6 contains our prospections for future work and our conclusions.

2 Related Works

Model Driven Development is not a new topic in software engineering. We have found 11 secondary studies among the search results. It is worth mentioning that the work by Asadi and Ramsin [2] is a secondary study closely related to this one. However, their work is not a systematic mapping but a literature survey. Also, their work is specific to MDA while we intended to identify every MDSE related approach, including non MDA-based.

In summary, there were no systematic mappings among the secondary studies. The identified systematic reviews were not related to the application of MDSE in general purpose use cases.

Whittle *et al.* [17] have put forward a survey applied on professional software developers in order to identify their problems when using MDSE in practice. For example, some authors have pointed that that the MDSE tools would evolve significantly and solve most of challenges [14], however, Whittle *et al.* have gathered evidences that indicate that this conception is not accurate [17]. In their study, they have identified that despite improvement of tools, there are still difficulties faced by developers, related to lack of methods, lack of training and misconceptions about the usage of models. The main similarity of this work is that it also investigates challenges related to MDSE application and we also argument that these difficulties could be all related to the lack of adequate methodology.

3 Methods

A systematic mapping is a specific method of literature review. It allows identifying and quantifying primary studies relevant to research questions in a specific knowledge area. In order to establish a systematic and repeatable literature review process, guidelines by Kitchenham and Charters [12] were followed.

An excerpt of the protocol definition is visible on Table 1. This table contains two columns. These columns are arranged into field name and value pairs that include the objective, questions, intervention, control, results, source selection criteria, and study selection criteria. Among these fields, the questions and study selection criteria are frequently referenced during the results of the conduction phase presented in Section 4.

The most important item of the protocol is its objective. By intending to identify MDSE success/failure cases and challenges, we have devised two questions. The first question aims to identify the success cases and the failures in specific domains. The results of this question are important because we are also looking for challenges in MDSE outside these domains. If the success cases became too ubiquitous, then, one could argue that our secondary question is irrelevant. This is because our secondary question is related to challenges encountered when applying MDSE into domains which do not have visible success cases. Without visible success cases, there would be a lack of tools or methodology, thus making the challenges more apparent.

Another important item of the protocol is the set of inclusion and exclusion criteria. The inclusion criteria are marked with a leading "I" whereas the exclusion criteria are marked with an "E". The "I1" and "I2" were created when intending to respond the primary question, while the "I3" is related to the secondary. The exclusion criteria are employed to remove the unrelated studies and other results that are not primary studies.

The keywords shown on Table 2 were used as basis to create the search string. Among every row, the conjunction operation (represented by "$\wedge$") is applied while the disjunction operation is applied among the synonyms of each keyword. Therefore, the final search string is $(A) \wedge (B) \wedge (C) \wedge (D)$.

The search was carried out through the following search engines: ACM Digital Library, Engineering Village Compendex, IEEE Xplore and Elsevier Scopus. The string was also tested on Springer Link and Google Scholar, however, the conducting phase using these search engines has been canceled because it was not possible to calibrate the search string providing the same semantics, causing these engines to return enormous numbers of unrelated results.

After ensuring that there were no duplicated studies, the review was conducted by analyzing studies returned by each search engine. The search engine priority was set by reviewing first the engine that returned the higher number of previously known studies.

The data extraction form contains fields that must be filled during the extraction phase. The planned form contains five fields:

1. Identified Success Case Domain;

Table 1. Protocol Definition

Protocol Item	Item Description
Objective	The objective of this literature review is to identify success cases and challenges in domain specific and general purpose MDSE approaches.
Primary Question	What are the specific domains in which developers have achieved success by employing MDSE?
Secondary Question	What are the the general purpose MDSE approaches and what are the challenges to create such approaches? Does MDA solve these challenges?
Intervention	Studies related to MDSE approaches and their challenges must be identified and categorized.
Control	The search results must involve a list of studies related to the questions that are known by the researchers. This list includes articles and books by Pastor, Whittle and Czarnecki.
Results	Quantitative data on approach frequency distribution within domain categories. Qualitative data on reported challenges.
Source selection criteria:	Source must index studies on Computer Science, Mathematics or Engineering. Source must allow Boolean operators. Source must be accessible by the researchers.
Selection Criteria:	**Inclusion:** – I1 - Primary studies that present a success case of MDD, MDSE, DSL or MDA in a specific domain; – I2 - Primary studies that present a non success case of MDD, MDSE, DSL or MDA in a specific domain; – I3 - Primary studies that present challenges of applying MDD, MDSE, DSL or MDA in general purpose projects; **Exclusion:** – E1 - Unrelated to MDD, MDSE, DSL or MDA. – E2 - Not a primary study.

2. Identified Failure Case Domain;
3. Identified MDSE problem or challenge;
4. Identified Validation type;
5. Presents Solution for an MDSE challenge.

The valid values for each of the enumerated fields are presented on Table 3. These valid values are defined as nominal sets, i.e., groupings of enumerated and named items that represent categories important to our study.

The first nominal set is named as Domain Set, which contains 57 nominals, including domains related to Web, Embedded Systems, Business Information

Table 2. Search String Definition

Identifier	Keyword	Synonyms and Related

A	Software Development	– software development;
		– software engineering;

B	Approach	– approach;
		– process;

C	Support	– tool;
		– support;

D	MDSE and MDD	– mdd;
		– development;
		– mde;
		– engineering;
		– software;
		– mda;
		– model-driven architecture;
		– model driven architecture;
		– Model-Driven;
		– model;
		– driven;
		– model-driven.

Table 3. Valid Values for Data Extraction Fields

Field Number	Field Name	Field Type	Cardinality	Nominal Set
1	Identified Success Case Domain	Subset of Nominals	zero to many	Domain Set
2	Identified Failure Case Domain	Subset of Nominals	zero to many	Domain Set
3	Identified MDSE Problem or Challenge	Subset of Nominals	zero to many	Problem Set
4	Identified Validation Type	Subset of Nominals	zero to many	Validation Set
5	Presents Solution for an MDSE challenge	One Nominal item	one	Boolean Set

Systems, Telecommunications and Networking, Industrial Control Systems, Military, Parallelism, Simulation, Computer Aided Design, Education and Computer Games. A complete list of these nominals is shown within the study packing[1].

The quality criteria were divided into two groups. The first group is used to select the studies related to domain successes and failures. In this case, we have defined that these studies must not present the domain simply as a case study, that is, to avoid papers that use a domain as a validation without providing a practical success or failure report.

[1] Available at: `https://webmail.correio.usp.br/home/gottardi@icmc.usp.br/msmdomain/` and `http://rosana.labes.icmc.usp.br/msmdomain/`

Considering the second group, it is related to the studies that present challenges or problems related to MDSE. We have planned strategies for defining if these studies are relevant for our secondary study. Therefore, we have defined that only the studies that either have a solution for the challenges or contain a validation should be carried into the discussion activity.

It is also important to point that a custom developed set of tools were employed during conduction phase. Their requirements involved supporting the hierarchical nominals and allowing the researchers to cooperate on the same review and to provide real-time reports about the review evolution and preliminary summarization via a web-page that features graphics and descriptive statistics. More details of these tools are also available within the packing documents.

4 Study Results

This section contains the summarization of results, which was carried out after conducting the extraction phase. We provide qualitative and quantitative data that were used to respond the research questions.

4.1 Search Results

The aim of this subsection is to provide information regarding the results returned by the search engines. Figure 1 contains a plot that was created to allow a general visualization of the distribution of studies collected. The graph was designed as a stacked bar plot, which allows the viewer to compare the portion of duplicated and unique results returned from each search engine.

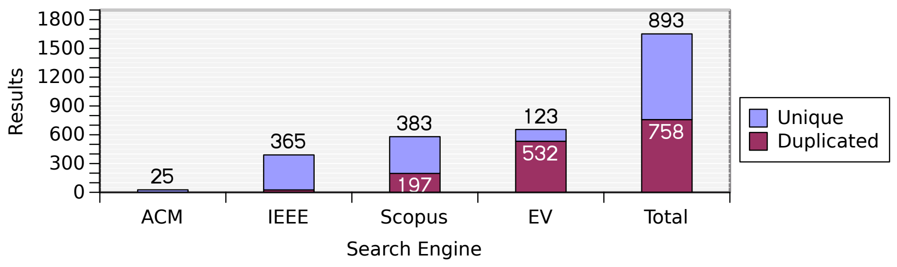

Fig. 1. Source Distribution

The columns of Figure 1 are named as “ACM”, “IEEE”, “Scopus” and “EV” since they represent, respectively, the search engines ACM Digital Library, IEEE Xplore, Elsevier Scopus and Elsevier/Engineering Village Compendex.

In this review, studies were not filtered by their publication year. The oldest study that passed the selection phase was published in 1985 and was returned by IEEE Xplore and written by Hoffnagle and Beregi [10]. Their study is related to automated software generation, however, since there is no explicit model as input, it was not categorized as MDSE.

In order to improve the visualization of the plots here shown, it was established that they would start from one year before the year that got the oldest results. In every bar plot, the vertical axis contains the number of studies, whereas the horizontal axis may represent categories, process phases or years.

Alkadi and Carver wrote the oldest study that was considered as related to MDSE by this review process. It was published in 1998. They created an approach which employs models for test case generation [1]. Bislimovska *et al.* wrote a paper which was used as example of a recent study that employs MDSE in the Web domain [4]. During the review process, we have also categorized the studies which have abstracts that present or discuss MDA-based approaches.

4.2 Domain Categorization: Results

The aim of categorizing the studies was to identify software solution domains in which MDSE is successful. In Figure 2 there is a plot which allows easy visualization of a large set of data gathered during this categorization process. It is possible to identify the evolution of the most common domains. As seen on the legend, there are four domains, namely "Embedded", "Web", "Network" and "Business", which represent the domains of Embedded Systems, Web Systems, Network Systems and Business Information Systems, respectively.

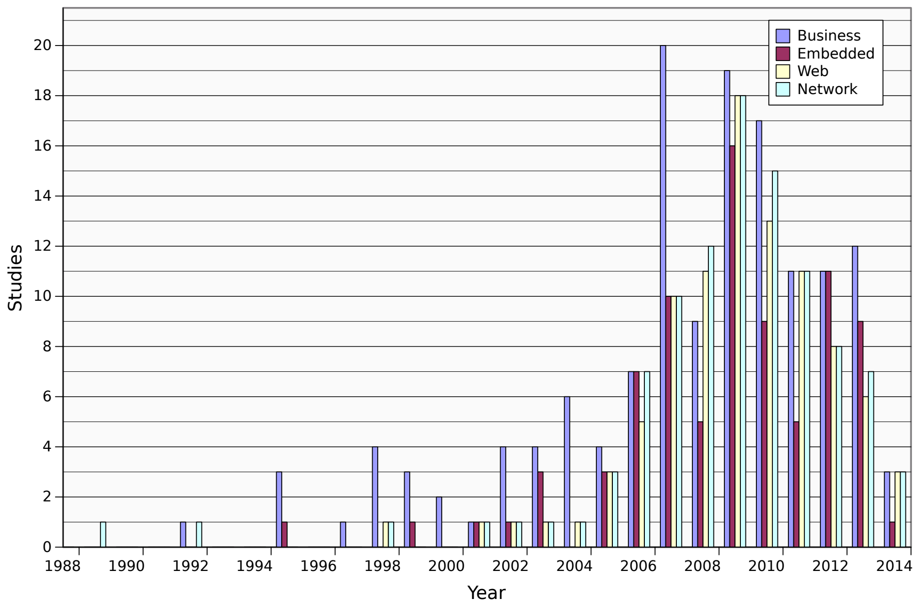

Fig. 2. Evolution of Most Common Domains

Besides not planned prior to execution, it has been identified that the use of MDSE in Embedded System Domain is very scattered among different subdomains. To avoid concerns related to how broad is our categorization of this domain, we have identified subdomains of use of MDSE Embedded Systems. These subdomains are presented in Figure 3 which contains a plot showing the evolution of number of studies that are related to the use of MDSE and these domains. The subdomains “Avionics and Aviation”, “Robotics”, “Agriculture”, “Cruise Control”, “Home Automation”, “Sensors and Actuators” and “Road Vehicle” are distributed per year. It is important to point that the “Total” bar is also the sum of the studies that do not have a specific subdomain among Embedded Systems.

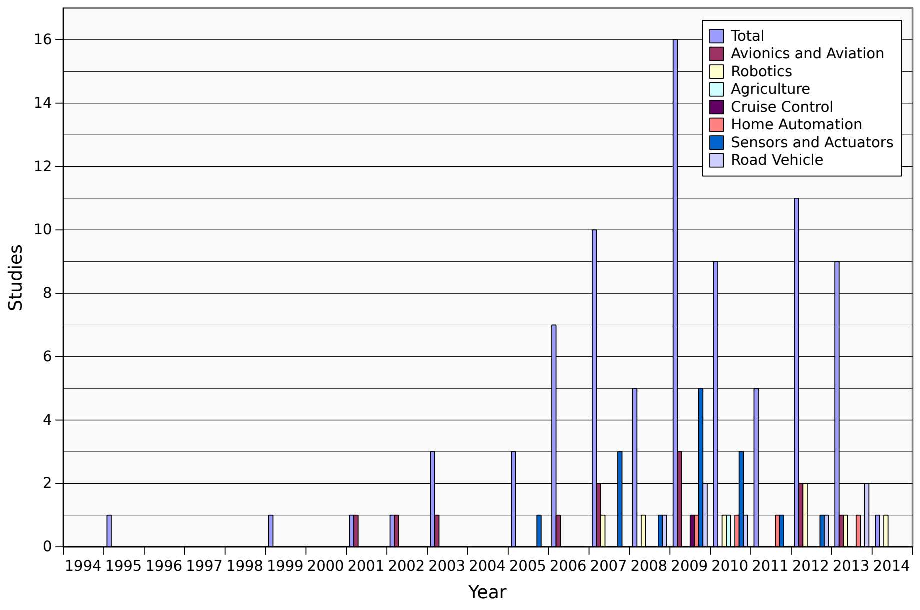

Fig. 3. Evolution of Embedded Systems Subdomains

4.3 MDSE Challenges: Results

The goal of this subsection is to provide quantitative results regarding the number of studies that present challenges on applying MDSE into general purpose projects.

We have identified 4 studies that contain references to maintenance challenges. We have also classified 4 studies that discuss methodology challenges. Considering these studies, there is the total number of 7 unique studies, since there is a single study which is common in both listed categories. Further discussion regarding this data is presented in Section 5.

5 Discussion

5.1 Domain Distribution

The results helped to confirm our previous expectations about MDSE tools intended for Business Information Systems, Web and Embedded Systems. By reading the related studies, we believe that a constant concern among these approaches is to accelerate the implementation of data entities and their manipulation.

However, we did not expect that the network domain would achieve high frequency in the identified distribution. Most of the MDSE studies in the network domain are related to the development of parallel distribution of software execution.

5.2 General Purpose Challenges

The General Purpose Challenge is related to the secondary question of this study. This question was planned because our research team is investigating the application of MDSE in non-conventional domains and in general purpose projects.

Following the results of this secondary study, the number of primary and secondary studies related to challenges in unconventional domains and in general purpose methodologies of MDSE application is far below the expected. Besides that, since the search results were carried out exhaustively, the data extraction was conducted as planned.

During the conduction we have identified 4 works that point methodology problems in MDSE besides the proposal of MDA [6,3,7,11]. We have also found 4 studies that contain information on maintenance problems that occur in projects which employ MDSE [11,15,16,8]. It is important to note that there is a single study which is common to both categories. Consequently, there are 7 studies presented in this subsection.

The summarization of these studies is presented on Table 4. This table contains seven columns. In these columns, it is possible to visualize the reference number of each study, authors, title, problem type and study type. The problem types are split into two columns: methodology and maintenance, which represent, respectively, methodology problems and maintenance problems. Then, the study lines that deal with specific problem types have their respective cells shaded in order to indicate that they relate to the problem type.

Chitforoush *et al.* are among the researchers that identified that MDSE lacks methodology, processes and guidelines to instruct when developers should use each model in a MDA-based project. In their attempt to provide a solution to this problem, they have defined a general methodology framework based on MDA

Table 4. MDSE Challenges Summarization

Reference	Author	Title	Methodology Problem	Maintenance Problem	Study Type
[7]	Chitforoush *et al.*	Methodology support for the model driven architecture	▨		Method Proposal
[3]	Asadi *et al.*	An MDA-based system development lifecycle	▨		Method Proposal
[6]	Cernickins *et al.*	An outline of conceptual framework for certification of MDA tools	▨		Method Proposal
[16]	Teppola *et al.*	Challenges in Deployment of Model Driven Development	▨	▨	Experience Report
[15]	Seifert *et al.*	Engineering long-lived applications using MDA		▨	Experience Report
[11]	Hovsepyan *et al.*	From aspect-oriented models to aspect-oriented code? The maintenance perspective		▨	Experimental Assessment
[8]	Er and Tekinerdogan	MoDSEL: Model-driven software evolution language		▨	Language Proposal

[7]. These authors claim that their framework is flexible enough to be adapted to various processes and needs. They also compared their framework to similar approaches. However, they do not provide validation on the efficiency of their approach. They have suggested this validation as future work but we could not find it published.

Asadi *et al.* are from the same research department and created another solution to the problem identified by Chitforoush *et al.*. They have defined a MDSE development life-cycle [3], which has as main advantage to propose more specific process definitions which aim to guide the developers using the MDA methodology. However, the stricter definition could also affect flexibility. The authors also compared their approach to related approaches and we could not find a validation study.

Both of these studies have identified problems in the methodology of previous works, but there is no validation whether the problem was completely solved.

Besides these papers, Cernickins *et al.* also have described a methodological problem, however, their focus is on tool certification. Besides, their certification framework contains guidelines that can be employed to identify if a tool set or a project is lacking an important activity or feature which they claim to be necessary [6].

The study written by Teppola *et al.* contains an experience report created by applying surveys on software development companies. These authors have described that software developers using MDSE perceive both maintenance and lack of methodology issues. They claim that despite the advantages that have been experienced by the developers while using MDSE, there are still several issues to be treated. The authors conclude that the developers are optimistic hoping that these issues are solved because they approve the use of MDSE regardless of its current limitations [16].

Seifert *et al.* have also written an experience report. In their work, they point that the use of MDSE increases dependency of the tool chain. They argue that this problem is not only limited to custom made tool chains because tools may be updated and become incompatible to the older model instances. This is caused by changes on the language definition. For instance, new UML definition versions are made available and tool developers may follow the new definitions and break backward compatibility [15].

Hovsepyan *et al.* have studied the impact of code-generation on software maintenance. The most important contribution of this study when compared to others presented herein is the in depth statistical analysis of the maintenance impact using metrics to compare the results in a quantitative manner. However, it focuses on specific models for a programming paradigm [11].

The last and most recent identified work in the challenge category is the study written by Er and Tekinerdogan. These authors describe a language named "MoDSEL" (Model-Driven Software Evolution Language). They claim that this language can be used to compare models, track their changes and identify maintenance impacts. Therefore, this is a solution that deals with the maintenance problems that may be found in software projects employing MDSE [8].

5.3 Limitations of this Study

The aim of this subsection is to provide details on the identified limitations and what we have done to mitigate them. The limitations were categorized by their origin, which include "Search Strategy", "Study Selection", "Data Extraction", and "Researcher Bias".

Search Strategy There are a few limitations related to our search strategy. The first of them is regarding to the search string. After calibrating the string in order to achieve most relevant results, it is possible that the string lost part of the original intended semantics and may fail to return some of the intended studies.

To mitigate this problem, we have identified what search engines indexed the preliminary known primary study defined as "control" in the protocol. Then we confirmed that the search string was enough to cause the search engine to return the indexed known studies.

The number of search engines is less than the initially planned. It was originally intended to include the databases by ISI Web of Knowledge, Science Direct, Wiley InterScience and Google Scholar. The decision to cancel conduction of results carried by Springer Link was late in our process. This was decided because this search engine provided a small relative number of relevant studies and the high number of results was affecting the review time. After these issues, we believe that it is important to plan the execution phases by reviewing the studies from each database sequentially in a "shortest job first" strategy instead of our adopted sequence.

Study Selection The guidelines by Kitchenham and Charters [12] have been defined considering that the selection activity should only be used to define which studies should proceed to the extraction activity. However, the authors preferred to categorize every study since the selection phase.

The impact of this approach is unknown. In our strategy, the selection phase became longer, however, we believe that it was positive to avoid another limitation of this phase: it was not possible to reject studies during the selection phase without providing general categories to the study, which is an evidence that no studies could have been rejected without proper reading of its abstract.

Data Extraction The main item analyzed during data extraction was identifying the set of categories in which each study should be linked to. This also includes the application domains.

A constant concern during the execution was to provide an exhaustive categorization of every returned study. Several categories were planned before the conduction. After discovering more specific studies, the authors then decided to create an hierarchical category definition interactively in order to provide an in depth report.

Since it was not possible to define if the interactively defined categories should be applied to studies reviewed prior to their definition, only a few categories were selected, and the review was restarted considering the new category set.

Since only the studies that provided information related to problems to apply MDSE were selected for discussion, the rate of discussed studies has became much inferior than the expected numbers. Besides that, since the primary question dealt in this study was to provide a systematic mapping related to the most common domains, this issue should not affect its credibility.

Researcher Bias Since this work was carried out by two researchers, the risk of researcher bias affecting the results is considerably high.

In order to mitigate this threat, we have defined keywords for each category and established systematic approaches to carry the review as impartial as possible.

Every study that presents information regarding MDSE challenges were reviewed extensively by every researcher. However, the studies unrelated to the challenges topic were not fully read during the process.

6 Conclusions

A secondary study has been presented in this paper. After reviewing a total of 893 studies, the most common domains have been identified as well as discussions related to challenges developers face while attempting to apply MDSE to projects dealing with uncommon or too specific domains.

As part of results summarization, we have identified that the MDSE success domains are clustered into four domains: Business Information Systems, Embedded Systems, Web Systems and Networking. This data was presented quantitatively considering the success cases that were not only used as case studies.

During our searches, we could not find a report on a failure case, still, we identified challenges and presented these qualitatively and a discussion section.

There are 7 identified studies which are related to MDSE. This discussion involved challenges related to software maintenance and methodology issues. The studies have also been categorized and summarized.

This secondary study presented the review of studies from 1985 to 2014, however, the most recent study related to MDSE challenges was published in 2012. Therefore, it is difficult to point whether a challenge has been dealt with in the past two years.

Besides that, we could find no evidences that the proposed solutions were in fact used. Considering the maintenance problems, we argue that these issues could rise in any project and they should be mitigated since its beginning.

Proposing guidelines as how to mitigate these issues are planned among our future works. Therefore, we intend to validate our systematic approach and the developed support tool. We are also extending our literary review in order to explain precisely the low number of studies that discuss the methodology problems for general purpose development. We also argue that it is important to publish failure cases studies. We believe that these studies would also help to understand the methodology problems.

This work is part of a research project about investigating processes in MDSE and their impact in domains in which MDSE support tools are not available. We are also investigating the impacts of software project tool support using experimental studies. Another investigation involves surveying students and practitioners.

The authors would also like to invite any reader to check the complete description of the methods, support tool and a more in depth discussion regarding the study subject. This content is available within the study packing[2].

Acknowledgements. Thiago Gottardi would like to thank CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) for grant received during the development of this work.

References

1. Alkadi, I., Carver, D.: A testing assistant for object-oriented programs. In: Aerospace Conference, 1998 IEEE. vol. 4, pp. 149–158 vol.4 (Mar 1998)
2. Asadi, M., Ramsin, R.: Mda-based methodologies: An analytical survey. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 5095 LNCS, 419–431 (2008)

[2] Available at:
`https://webmail.correio.usp.br/home/gottardi@icmc.usp.br/msmdomain/` and `http://rosana.labes.icmc.usp.br/msmdomain/`

3. Asadi, M., Ravakhah, M., Ramsin, R.: An mda-based system development lifecycle. In: Proceedings - 2nd Asia International Conference on Modelling and Simulation, AMS 2008. pp. 836–842. Kuala Lumpur (2008)
4. Bislimovska, B., Bozzon, A., Brambilla, M., Fraternali, P.: Textual and content-based search in repositories of web application models. ACM Trans. Web 8(2), 11:1–11:47 (Mar 2014)
5. Brambilla, M., Cabot, J., Wimmer, M.: Model-driven Software Engineering in Practice. G - Reference, Information and Interdisciplinary Subjects Series, Morgan & Claypool (2012)
6. Cernickins, A., Nikiforova, O., Ozols, K., Sejans, J.: An outline of conceptual framework for certification of mda tools. In: Proceedings of the 2nd International Workshop on Model-Driven Architecture and Modelling Theory-Driven Development, MDA and MTDD 2010, in Conjunction with ENASE 2010. pp. 60–69. Athens (2010)
7. Chitforoush, F., Yazdandoost, M., Ramsin, R.: Methodology support for the model driven architecture. In: Proceedings - Asia-Pacific Software Engineering Conference, APSEC. pp. 454–461. Nagoya (2007)
8. Er, E., Tekinerdogan, B.: Modsel: Model-driven software evolution language. In: Mernik, M. (ed.) Formal and Practical Aspects of Domain-Specific Languages: Recent Developments, pp. 572–594 (2012)
9. France, R., Rumpe, B.: Model-driven development of complex software: A research roadmap. In: 2007 Future of Software Engineering. pp. 37–54. FOSE '07, IEEE Computer Society, Washington, DC, USA (2007)
10. Hoffnagle, G.F., Beregi, W.E.: Automating the software development process. IBM Systems Journal 24(2), 102–120 (1985)
11. Hovsepyan, A., Scandariato, R., Van Baelen, S., Berbers, Y., Joosen, W.: From aspect-oriented models to aspect-oriented code? the maintenance perspective. In: AOSD.10 - 9th International Conference on Aspect-Oriented Software Development. pp. 85 – 96. Rennes and Saint-Malo, France (2010)
12. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. Tech. Rep. EBSE 2007-001, Keele University and Durham University Joint Report, UK (2007), `http://www.dur.ac.uk/ebse/resources/guidelines/Systematic-reviews-5-8.pdf`
13. OMG: Overview and guide to OMG's model driven architecture. http://www.omg.org/cgi-bin/doc?omg/03-06-01 (May 2010), `http://www.omg.org/cgi-bin/doc?omg/03-06-01`
14. Pastor, O., Molina, J.C.: Model-Driven Architecture in Practice: A Software Production Environment Based on Conceptual Modeling. Springer-Verlag New York, Secaucus, NJ, USA (2007)
15. Seifert, T., Beneken, G., Baehr, N.: Engineering long-lived applications using mda. In: Proceedings of the Eighth IASTED International Conference on Software Engineering and Applications. pp. 241–246 (2004)
16. Teppola, S., Parviainen, P., Takalo, J.: Challenges in deployment of model driven development. In: Software Engineering Advances, 2009. ICSEA '09. Fourth International Conference on. pp. 15–20 (Sept 2009)
17. Whittle, J., Hutchinson, J., Rouncefield, M., Burden, H., Heldal, R.: Industrial adoption of model-driven engineering: Are the tools really the problem? In: Moreira, A., Schätz, B., Gray, J., Vallecillo, A., Clarke, P. (eds.) Model-Driven Engineering Languages and Systems, Lecture Notes in Computer Science, vol. 8107, pp. 1–17. Springer Berlin Heidelberg (2013)