METAMODEL CRITICS APPROACH FOR DESIGNING METAMODELS IN MODEL-DRIVEN SOFTWARE ENGINEERING

By

MOHAMMAD ALIF BIN MOHAMMAD ALLAUDIN

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfilment of the Requirements for the Degree of Master of Science

July 2021

FSKTM 2021 14

Abstract of thesis presented to the Senate of Universiti Putra Malaysia in fulfilment of the requirement for the degree of Master of Science

METAMODEL CRITICS APPROACH FOR DESIGNING METAMODELS IN MODEL- DRIVEN SOFTWARE ENGINEERING

By

MOHAMMAD ALIF BIN MOHAMMAD ALLAUDIN

July 2021

Chair : Norhayati Mohd Ali, PhD
Faculty : Computer Science and Information Technology

The advances of Model-Driven Engineering (MDE) or Model-Driven Software Engineering (MDSE) have motivated the use of metamodeling approach in the development of software systems. Metamodeling is the key concept in MDSE approach that describes how domain models are organized: their ontology, syntax and semantic. The model's abstraction levels, and relations of levels are also defined via metamodeling approach. Many software developers applied the MDSE via metamodel-based approach for software development in various domains. However, one of the challenges in MDSE is the quality of the metamodels. A study reported that metamodels possibly have quality defects because metamodel design is related to the cognitive ability of metamodel designers. Hence, the research problems in metamodel design includes syntax errors, poor-constructs, over complex-constructs, semantic defects, and poor quality of the design. Some of the problems in metamodel design includes syntax errors, poor-constructs, over complex-constructs and semantic defects. Many researchers have proposed several guidelines and rules to assist the software developers on how to design a quality metamodel. However, an automated approach to check and detect the errors in metamodel design is still lacking. Thus, in this research, a critic-based approach is proposed to detect metamodel design errors in an automated way and provide suggestions to improve the metamodel design. The research scope is focused on designing a metamodel design using Unified Modeling Language (UML) Class Diagram notation. The aim of this research is to integrate a critic-based approach within modelling tool to assist the software developers in designing a quality metamodel design in MDSE.

The research consists of several phases to achieve the research aim. The early phase of this research was initiated by identifying the requirements of a quality metamodel design via literature review analysis. Several rules and guidelines in designing quality metamodel as proposed by previous researchers are obtained

in the initial phase. In the intermediate phase, metamodel critics are created based on the rules and guidelines of a quality metamodel. A prototype to demonstrate the proof of concept for metamodel critics was developed. Several exemplars of metamodels are used to demonstrate the application of metamodel critics. The final phase of this research is to perform a user evaluation to assess the usability and effectiveness of metamodel critic approach in designing a quality metamodel.

The main contribution of this research is the mechanism to detect metamodel design errors and provide suggestions for improvement of metamodel design in MDSE via critic-based approach. Another important contribution is the development of a metamodel critic tool that can be applied to assist software developers in designing a quality metamodel using the Unified Modelling Language (UML) notations.

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia sebagai memenuhi keperluan untuk Ijazah Sarjana Sains

PENDEKATAN KRITIS METAMODEL UNTUK MEREKA BENTUK METAMODEL DALAM KEJURUTERAAN PERISIAN BERPANDUKAN MODEL

Oleh

MOHAMMAD ALIF BIN MOHAMMAD ALLAUDIN

Julai 2021

Pengerusi : Norhayati Mohd Ali, PhD
Fakulti : Sains Komputer dan Teknologi Maklumat

Kemajuan dalam Kejuruteraan Berpandukan Model (KBM) atau Kejuruteraan Perisian Berpandukan Model (KPBM) telah memberi motivasi dalam penggunaan pendekatan metamodel di dalam pembangunan sistem perisian. Metamodel ialah konsep utama dalam pendekatan KPBM yang menerangkan bagaimana model domain disusun: ontologi, sintaks, dan semantik. Tahap pengabstrakan model dan tahap hubungan model juga telah ditakrifkan melalui pendekatan metamodel. Kebanyakan pembangun perisian menggunakan KPBM melalui pendekatan berasaskan metamodel untuk pembangunan perisian dalam pelbagai domain. Walau bagaimanapun, salah satu cabaran dalam KPBM ialah kualiti bagi metamodel. Oleh itu, masalah kajian dalam reka bentuk metamodel termasuk ralat sintaks, binaan yang lemah, binaan yang lebih kompleks, kecacatan semantik dan kualiti reka bentuk yang rendah. Satu kajian melaporkan bahawa metamodel mungkin mempunyai kecacatan kualiti kerana reka bentuk metamodel berkait dengan keupayaan kognitif pereka bentuk metamodel. Beberapa masalah reka bentuk metamodel termasuk kesalahan sintaks, binaan tidak memuaskan, binaan terlampau kompleks dan kesalahan semantik. Ramai penyelidik telah mencadangkan beberapa garis panduan dan peraturan untuk membantu pembangun perisian tentang cara untuk mereka bentuk satu metamodel yang berkualiti. Walau bagaimanapun, pendekatan automatik yang menyemak dan mengesan kesalahan pada reka bentuk metamodel masih lagi kekurangan. Oleh itu, dalam kajian ini, pendekatan berasaskan kritik telah dicadangkan bagi mengesan kesalahan reka bentuk metamodel secara automatik dan menyediakan cadangan untuk memperbaiki reka bentuk metamodel. Skop penyelidikan tertumpu kepada mereka bentuk reka bentuk metamodel menggunakan tatatanda Rajah Kelas Bahasa Permodelan Bersepadu (UML). Matlamat penyelidikan ini ialah untuk mengintegrasikan pendekatan berasaskan kritik dalam alat pemodelan untuk membantu pembangun perisian mereka bentuk metamodel yang berkualiti dalam KPBM.

Penyelidikan ini merangkumi beberapa fasa bagi mencapai matlamat penyelidikan. Fasa awal penyelidikan ini dimulakan dengan mengenalpasti keperluan bagi kualiti reka bentuk metamodel melalui analisis kajian literatur. Beberapa peraturan dan garis panduan dalam mereka bentuk metamodel berkualiti yang telah dicadangkan oleh penyelidik yang lalu telah diperolehi dalam fasa awal. Di fasa pertengahan, kritik metamodel telah dibina berdasarkan peraturan dan garis panduan kualiti bagi satu metamodel. Satu prototaip untuk membuktikan konsep kritik metamodel telah dibangunkan. Beberapa contoh metamodel telah digunakan untuk menunjukkan penggunaan kritik metamodel. Fasa terakhir penyelidikan ini ialah melakukan penilaian pengguna untuk menilai kebolehgunaan dan keberkesanan pendekatan kritik metamodel dalam mereka bentuk satu metamodel yang berkualiti.

Sumbangan utama penyelidikan ini ialah pembangunan mekanisma untuk mengesan kesalahan reka bentuk metamodel dan memberikan cadangan bagi memperbaiki reka bentuk metamodel dalam KPBM melalui pendekatan berasaskan kritik. Sumbangan penting yang lain ialah pembangunan alat kritik metamodel yang boleh diaplikasikan untuk membantu pembangun perisian mereka bentuk metamodel yang berkualiti menggunakan notasi *Unified Modelling Language (UML)*.

ACKNOWLEDGEMENTS

In the Name of Allah, the Most Gracious and the Most Merciful
All the praises and thanks be to Allah,
and His blessings are for the righteous deeds.
Humblest gratitude to the Prophet Muhammad ﷺ, his family and companions,
for their way of life has been a continuous guidance for me.

First and foremost, I would like to sincerely thank my supervisor, Associate Professor Dr. Norhayati Mohd Ali for her continuous guidance, understanding, patience and most of all, encouragement for me throughout the years to complete this thesis. It has been such an experience to have her as my supervisor. I would also like to extend that gratitude to my supervisory committee, Associate Professor Ts. Dr. Novia Indriaty Admodisastro and Associate Professor Ts. Dr. Rodziah Atan for their advice and supports for the completion of this thesis.

I'd like to thank my parents, Naterah and Allaudin, without whom I would have never given a thought to continue my study up to this level. I wish to give thanks to the rest of my family members who has been supportive as well.

Lastly, I like to give special thanks to my friends and colleagues who never given up on me and encourage me with their wisdoms and advice. Thank you all for helping me to finish this thesis.

Mohammad Alif Bin Mohammad Allaudin
March 2023

This thesis was submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfilment of the requirement for the degree of Master of Science. The members of the Supervisory Committee were as follows:

Norhayati Mohd Ali, PhD
Associate Professor
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Chairman)

Novia Indriaty Admodisastro, PhD
Associate Professor Ts.
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

Rodziah Atan, PhD
Associate Professor Ts.
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia
(Member)

_____
ZALILAH MOHD SHARIFF, PhD
Professor and Dean
School of Graduate Studies
Universiti Putra Malaysia

Date: 09 March 2023

Declaration by the Graduate Student

I hereby confirm that:

- this thesis is my original work;
- quotations, illustrations and citations have been duly referenced;
- this thesis has not been submitted previously or concurrently for any other degree at any institutions;
- intellectual property from the thesis and the copyright of the thesis are fully-owned by Universiti Putra Malaysia, as stipulated in the Universiti Putra Malaysia (Research) Rules 2012;
- written permission must be obtained from the supervisor and the office of the Deputy Vice-Chancellor (Research and innovation) before the thesis is published in any written, printed or electronic form (including books, journals, modules, proceedings, popular writings, seminar papers, manuscripts, posters, reports, lecture notes, learning modules or any other materials) as stated in the Universiti Putra Malaysia (Research) Rules 2012;
- there is no plagiarism or data falsification/fabrication in the thesis, and scholarly integrity is upheld in accordance with the Universiti Putra Malaysia (Graduate Studies) Rules 2003 (Revision 2015-2016) and the Universiti Putra Malaysia (Research) Rules 2012. The thesis has undergone plagiarism detection software

Signature: _____ Date: _____

Name and Matric No.: MOHAMMAD ALIF BIN MOHAMMAD ALLAUDIN

Declaration by Members of the Supervisory Committee

This is to confirm that:

- the research and the writing of this thesis were done under our supervision;
- supervisory responsibilities as stated in the Universiti Putra Malaysia (Graduate Studies) Rules 2003 (Revision 2015-2016) are adhered to.

Signature: _____
Name of Chairman of Supervisory Committee: Norhayati Mohd Ali, PhD

Signature: _____
Name of Member of Supervisory Committee: Novia Indriaty Admodisastro, PhD

Signature: _____
Name of Member of Supervisory Committee: Rodziah Atan, PhD

TABLE OF CONTENTS

		Page
ABSTRACT		i
ABSTRAK		iii
ACKNOWLEDGEMENTS		v
APPROVAL		vi
DECLARATION		viii
LIST OF TABLES		xii
LIST OF FIGURES		xiii
LIST OF ABBREVIATIONS		xv
CHAPTER		
1	**INTRODUCTION**	1
	1.1 Research Background and Motivation	1
	1.2 Problem Statements	2
	1.3 Research Objectives	3
	1.4 Research Questions	3
	1.5 Research Scope	3
	1.6 Research Contributions	4
	1.7 Thesis Organization	5
2	**LITERATURE REVIEW**	6
	2.1 Introduction	6
	2.2 Model-Driven Software Engineering	6
	2.3 Metamodeling in Model-Driven Software Engineering	8
	2.4 Critiquing in Software Engineering	14
	2.5 Related Works in Metamodeling	17
	2.6 Summary	21
3	**RESEARCH METHODOLOGY**	22
	3.1 Introduction	22
	3.2 Research Phases	23
	3.2.1 Literature Review of Model-Driven Software Engineering, Metamodeling Approach, and Critic-Based Approach in Software Engineering	23
	3.2.2 Identify the Requirements for Metamodel Critic Approach	24
	3.2.3 Formulate Metamodel Critic Specification and Develop Metamodel Critics for Metamodel Design	28
	3.2.4 Proof of Concept for Metamodel Critics via Metamodel Design Exemplars	28
	3.2.5 Evaluate the Metamodel Critic Approach for Metamodel Design via User Evaluation	29
	3.3 Summary	29

4	**METAMODEL CRITICS FOR METAMODEL DESIGN**		30
	4.1	Introduction	30
	4.2	Quality Attributes of Metamodel Design	30
	4.3	Overview of Metamodel Critic Approach	31
		4.3.1 Critic Rules and Guidelines for Metamodel Design	32
		4.3.2 Metamodel Critics	35
		4.3.3 Parsing Process of Metamodel Design	41
		4.3.4 Critics and Feedback of Metamodel Design	42
	4.4	Summary	42
5	**RESULT AND DISCUSSION**		44
	5.1	Introduction	44
	5.2	Procedures for the Evaluation	44
	5.3	Evaluation of Metamodel Critic Using Exemplars of Metamodel	45
	5.4	Questionnaire Design	48
	5.5	Evaluation Criteria	48
		5.5.1 Usability Evaluation	49
		5.5.2 Effectiveness Evaluation	49
	5.6	User Evaluation Results	50
		5.6.1 Evaluation Results from Software Developers	50
		5.6.2 Evaluation Results from Postgraduate Students	53
	5.7	Summary	59
6	**CONCLUSION**		60
	6.1	Introduction	60
	6.2	Significance of the Research	60
	6.3	Limitation of the Research	60
	6.4	Future Works	61
	6.5	Summary	62
REFERENCES			63
APPENDICES			68
BIODATA OF STUDENT			83
LIST OF PUBLICATIONS			84

LIST OF TABLES

Table		Page
2.1	Library of Metamodel Quality Properties	12
2.2	Quality Attributes for Metamodels	13
2.3	Related Work on Critic-Based Approach in Software Engineering	16
2.4	List of Related Work in Metamodel in Different Domain	21
3.1	Requirements of Metamodel Critic Approach	25
4.1	The Mapping of Metamodel Critics with Quality Attributes	33
4.2	Metamodel Critique	34
5.1	Demographic Result from Software Developers	50
5.2	Demographic Result Percentage for Software Developers	51
5.3	SUS Result from Software Developers	51
5.4	General Questions Result Part 1	52
5.5	General Questions Result Part 2	52
5.6	Level of Importance Result from Software Developers	53
5.7	Demographic Result for Postgraduate Students	54
5.8	Demographic Result Percentage for Postgraduate Students	55
5.9	SUS Result from Postgraduate Students	56
5.10	General Questions Result Part 1 (Postgraduate Students)	57
5.11	General Questions Result Part 2 (Postgraduate Students)	57
5.12	Level of Importance Result from Postgraduate Students	58

LIST OF FIGURES

Figure		Page
2.1	Metamodeling Layers in MOF adapted from Metamodeling and Model Transformations in Modeling and Simulation	8
2.2	Base Metamodel adapted from Design Patterns for Metamodel	9
2.3	The GuestBook Model in UML adapted from Template Programming for Model-Driven Code Generation	10
2.4	Very Simple Metamodel for the Java Language adapted from Template Programming for Model-Driven Code Generation	10
2.5	Language Use Solution to the Linguistic / Ontological Paradox adapted from Ontological and Linguistic Metamodelling Revisited: A Language Use Approach	11
2.6	Assessing the Quality of Metamodels adapted from Assessing the Quality of Metamodels	14
2.7	Hypertension Management Workflow (Overall) adapted from A Flexible Metamodeling Approach for Healthcare System	18
2.8	Multiple Metamodeling Hierarchy adapted from A Flexible Metamodeling Approach for Healthcare System	19
2.9	HL7 Metamodel Element adapted from Working with the HL7 Metamodel in a Model-Driven Engineering Context	20
3.1	Research Methodology	23
3.2	The Mapping of Metamodel Critic Approach to the Critic Taxonomy adapted from A Taxonomy and Mapping of Computer-Based Critiquing Tools	27
4.1	Overview of Metamodel Critic Approach	32
4.2	Critique for MC1	36
4.3	Critique for MC2	36
4.4	Critique for MC3	37
4.5	Critique for MC4	37

4.6	Critique for MC5	38
4.7	Critique for MC6	38
4.8	Critique for MC7	39
4.9	Critique for MC8	39
4.10	Critique for MC9	40
4.11	Critique for MC10	40
4.12	Critiquing Component of Metamodel Critic Approach	42
5.1	Library Element Metamodel	45
5.2	Library Element Metamodel Critic Output	46
5.3	ALMA-C Metamodel	46
5.4	ALMA-C Metamodel Critic Output	47
5.5	SmartCity DSL Metamodel	47
5.6	SmartCity DSL Metamodel Critic Output	48

LIST OF ABBREVIATIONS

ALMA-C	Agent-based Land Market for Coast
AUTOSAR	AUTomotive Opens Systems ARchitecture
CASE	Computer Aided Software Engineering
DECS	Diagram Editor Constraints System
DSL	Domain Specific Language
DSML	Domain Specific Modelling Language
GDPR	General Data Protection Regulation
GPL	General Purpose Language
IDE	Integrated Development Environment
MBSD	Model-Based Software Development
MCID	Model and Code Inconsistency Detection
MDE	Model-Driven Engineering
MDSE	Model-Driven Software Engineering
MOF	Meta-Object Facility
OCA	Orthogonal Classification Architecture
OMG	Object Management Group
QM4MM	Quality Model for Metamodel
SecTro	Secure Tropos
SUS	System Usability Scale
UML	Unified Modelling Language
ViDI	Visual Design Inspection
XMI	XML Metadata Interchange
XML	eXtensible Markup Language

CHAPTER 1

INTRODUCTION

1.1 Research Background and Motivation

Model-Driven Software Engineering (MDSE) includes different model-driven approaches to software development that includes domain-specific modelling, model-driven architecture, and model-integrated computing. The MDSE approach has been seen to manage the increasing of software complexity according to (Hutchinson, Rouncefield, and Whittle 2011; Gascueña, Navarro, and Fernández-Caballero 2012; Hinkel and Strittmatter 2017; Madni and Sievers 2018; Bettini et al. 2019). The key concept from MDSE is the metamodels which are essential to define the modeling primitives used in modeling activities (Basciani et al. 2019).

Most of the metamodeling tools employ a constraint specification approach for governing the syntax and semantics of model elements and the values of their attributes. Thus, the process of specifying constraints for metamodeling tools is more complex as it uses formal approaches and requires deep cognitive load. Hence, the need to have a systematic construction and use of models as main artifacts in MDSE process is essential. Therefore, the focus of this research is to propose a mechanism to detect potential problems at metamodel elements using a critic-based approach between model and metamodel.

The critic-based approach concept is mainly to identify potential problems; provide suggestion and possibly offer automated or semi-automated artefacts improvements to metamodel designers (Ali, Hosking, and Grundy 2013). Previous studies show that critiquing-based approach supports the human-computer collaborative problem solving. Hence, the objective of this research is to formulate a critic specification mechanism for metamodel elements in a modelling tool to help assist metamodel designers in creating a good quality metamodel. The critic specification mechanism for the metamodel will be validated against metamodel quality attributes based on the metamodel design guidelines from previous studies.

The expected result of this project is a new mechanism for specifying critics at a metamodel level that will assist metamodel designers in designing quality metamodel and checking consistency of model and metamodel in MDSE.

1.2 Problem Statements

MDSE includes various model-driven approaches to software development, including domain-specific modelling, model-driven architecture, and model-integrated computation. The MDSE approach has been seen to manage the increasing of software complexity according to (Hutchinson, Rouncefield, and Whittle 2011; Gascueña, Navarro, and Fernández-Caballero 2012; Hinkel and Strittmatter 2017; Madni and Sievers 2018; Bettini et al. 2019). There are some proposed design patterns by (Cho and Gray 2011) and by (van Emde Boas 2004) that designers can follow in designing a metamodel. The construction of metamodeling tools is complex as it requires solid skills in software modelling and programming. (Williams et al. 2013) claimed that there is little support to assist metamodel developers on the required properties of metamodels. A study by (López-Fernández, Guerra, and de Lara 2014) has provided a set of rules, constraints, and specifications for designers to use them to design a quality metamodel. (Bettini et al. 2019) emphasized that metamodel design must be accurate and need to consider essential quality factors, such as maintainability, reusability, and understandability.

In addition, most of the metamodeling tools employ a constraint definition / specification approach (e.g., DECS by (Qattous 2009), Marama by (Grundy et al. 2013)) for governing the syntax and semantics of model elements and the values of their attributes. Thus, the process of specifying constraints for metamodeling tools is more complex as it uses formal approach, and it involves deep cognitive load. (Ma, He, & Liu, 2013) also reported that “it is unavoidable that metamodels have quality defects because their design is related to the cognitive ability of designers”. Similarly, (Cho and Gray 2011) also stated that the quality of a metamodel design may also vary according to the designer’s domain knowledge and modelling language expertise. Some of the problems in metamodeling design include syntax errors, poor constructs, over-complex constructs, and semantic defects (Ma, He, and Liu 2013). (Ali, Hosking, & Grundy, 2013) stated that the use of critic concept has not to date been applied within metamodeling tools. Thus, the need to have a systematic construction and use of metamodels as main artifacts in MDSE process is essential.

Several studies emphasized on the quality attributes of metamodel design. Thus, (Bertoa and Vallecillo 2010; Ma, He, and Liu 2013; López-Fernández, Guerra, and de Lara 2014; Bettini et al. 2019; Basciani et al. 2019) have suggested several quality attributes to be applied in designing a quality metamodel design. In this research, three quality attributes from the literatures have been selected to assess the metamodel design. The three quality attributes are well-structuredness, correctness, and completeness. These three quality attributes are selected based on the metamodel design guidelines and rules from (López-Fernández, Guerra, and de Lara 2014; van Emde Boas 2004) that we adopted for this research.

Realizing the existing problems of metamodeling design and critics to detect potential problems / errors for metamodel elements do not exist yet, thus we are proposing a new mechanism / method to specify critics for metamodel elements. However, as designer may complete the task of designing the metamodel, there are concerns regarding its evolution which discusses several stages to improve the design to be better in its quality as changes to the software may affect its documents. Such documents must have the design in which consists of the model and metamodel for future designer and developer's references.

1.3 Research Objectives

The main objective of this research is to utilise critique-based approach to enhance a metamodel design by utilising the critique rules and design guidelines that has been proposed by previous researchers. To achieve this, the research objectives are as follow:

1. To propose a metamodel critic approach in designing a quality metamodel.
2. To formulate a critic specification mechanism for metamodel elements in a modelling tool. (i.e., Metamodel Critics)
3. To embed the critic specification mechanism with the metamodel specifications.
4. To validate the critic specification mechanism (i.e., Metamodel Critics) against the metamodel quality attributes; namely well-structuredness, completeness, and correctness attributes using the critiquing approach for metamodel critic approach usability.

1.4 Research Questions

Based on the mentioned research objectives of this study, the research questions that was formulated are as follow:

1. What is the mechanism to specify critics for metamodel elements?
2. How to specify critics for a metamodel using the identified mechanism?
3. How can the metamodeling critics (i.e., Metamodel Critics) enhance the quality of metamodel design against the specified metamodel quality attributes from previous research?

1.5 Research Scope

This research work focuses on the implementation of metamodel critique in the same environment as the application that can be used to design a metamodel. In this implementation, the critique is use as an output to gives the metamodel designer on how to improve the metamodel design by showing the critique type, critique description, designing guideline and recommendation on how to improve the design of metamodel that designer has created. Based on the critique,

designer follow the recommendation of the critique to mend the design and re-check the metamodel design again until it produces either no or minimum number of the critique for the metamodel that has been created.

The scope of this research work is to implement the critiquing mechanism in the same environment as the metamodel design application, namely Eclipse MARS2 IDE. The metamodel design environment is an extension of Eclipse MARS2 IDE, Papyrus, which will save the metamodel design file in a Unified Modeling Language (.uml) format using class diagram notations. The reason of limiting the file to be save in a (.uml) allows for the metamodel design to be parse for the process of checking the design that can produce the critique output that will provide reason and recommendation to fix the metamodel design fault that is detected as design error against the rules and guidelines that has been implemented in the metamodel critic tool.

1.6 Research Contributions

The research work in this thesis contributes to the field of Model-Driven Software Engineering, particularly on metamodel designing part with the use of critic-based approach in software engineering. The main contribution from this research includes:

1. The use of critic-based approach that helps metamodel designers to improve the design of metamodel by providing feedback to them by giving suggestions to amend the errors that was made during the checking of metamodel design.
2. The research has produced a plugin tool that works in the same environment of designing a metamodel to help metamodel designers do the checking for the metamodel design. Designers may do the checking as many times as they require until they are satisfied with the metamodel design. This saves up the time since metamodel designers doesn't have to switch between windows and can do the corrections straight away based on the given suggestions.
3. This metamodel critic tool included several components that made the checking for metamodel properties possible. The components include XMI parser and the critique that made up of several parts. The tool main purpose is to prove the concept of using critic-based approach can be helpful for designers to produce a quality metamodel design.
4. This research work contributed to the body of knowledge of MDSE specifically in the modelling and designing activities. Thus, the development of metamodel critic tool would support for assessing the quality of modelling artifacts.

1.7 Thesis Organization

This thesis includes five chapter in total. Chapter 1 explains a brief discussion background, problem statement, research objectives, research questions, research scopes and the significance of the study.

Chapter 2 explains a literature study to review the main principles of MDSE, metamodeling in MDSE, critique-based approach in software engineering and related subjects to the focus of the research work.

Chapter 3 explains the general research methodology used to accomplish the objective of the research. It presents the framework of the research work and discuss the stages in detail.

Chapter 4 discusses in detail the metamodel critic approach that has been developed to aid in evaluation of the research. The chapter covers the design of the critique mechanism in the prototype, implementation, and survey parameters.

Chapter 5 presents the data analysis and the results of the survey in a controlled group that is conducted.

Finally, the research work is presented with a conclusion and discussion for future work in Chapter 6.

REFERENCES

Abdulkareem, Soran Mahmood; Ali, Norhayati Mohd; Admodisastro, Novia; Sultan, Abu Bakar Md. 2017. “Class Diagram Critic: A Design Critic Tool for UML Class Diagram.” *Advanced Science Letters* 23 (11): 11567–11571. doi: https://doi.org/10.1166/asl.2017.10330.

Ali, Norhayati Mohd, John Hosking, and John Grundy. 2013. “A Taxonomy and Mapping of Computer-Based Critiquing Tools.” *IEEE Transactions on Software Engineering* 39 (11): 1494–1520. doi: 10.1109/TSE.2013.32.

Ali, Norhayati Mohd, John Hosking, John Grundy, and Jun Huh. 2010. “End-User Oriented Critic Specification for Domain-Specific Visual Language Tools,” 297–300.

Alroobaea, Roobaea, and Pam J Mayhew. 2014. “How Many Participants Are Really Enough for Usability Studies?” In *Proceedings of 2014 Science and Information Conference, SAI 2014*, 48–56. doi: 10.1109/SAI.2014.6918171.

Amjath Jamal, Nur Amirah, and Norhayati Mohd Ali. 2017. “Comparative Critiquing and Example-Based Approach for Learning Client-Server Design.” *2017 IEEE Conference on E-Learning, e-Management and e-Services (IC3e)*. IEEE, 30–35.

Basciani, Francesco, Juri di Rocco, Davide di Ruscio, Ludovico Iovino, and Alfonso Pierantonio. 2019. “A Tool-Supported Approach for Assessing the Quality of Modeling Artifacts.” *Journal of Computer Languages* 51 (April). Elsevier Ltd: 173–192. doi: 10.1016/j.cola.2019.02.003.

Basha, N Md Jubair, Salman Abdul Moiz, and Mohammed Rizwanullah. 2012. “Model Based Software Development: Issues & Challenges.” *International Journal of Computer Science and Informatics (IJCSI)* 2 (1,2): 226–230.

Berry, Daniel M. 1992. “Academic Legitimacy of the Software Engineering Discipline,” no. November.

Bertoa, Manuel F, and Antonio Vallecillo. 2010. “Quality Attributes for Software Metamodels.” *13th TOOLS Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2010)*. doi: 10.1.1.309.8687.

Bettini, Lorenzo, Davide di Ruscio, Ludovico Iovino, and Alfonso Pierantonio. 2019. “Quality-Driven Detection and Resolution of Metamodel Smells.”

IEEE Access 7. IEEE: 16364–16376. doi: 10.1109/ACCESS.2019.2891357.

Cetinkaya, Deniz, and Alexander Verbraeck. 2011. “Metamodeling And Model Transformations In Modeling And Simulation.” *Proceedings of the 2011 Winter Simulation Conference*, 3043–3053.

Cho, Hyun, and Jeff Gray. 2011. “Design Patterns for Metamodels.” *ACM International Conference on Systems, Programming, Languages, and Applications: Software for Humanity, SPLASH’11 and the Co-Located Workshops: DSM’11, TMC’11, AGERE’11, AOOPES’11, NEAT’11, and VMIL’11*, 25–31. doi: 10.1145/2095050.2095056.

Dashofy, Eric M, André van der Hoek, and Richard N Taylor. 2002. “An Infrastructure for the Rapid Development of XML-Based Architecture Description Languages.”

Durisic, Darko, Miroslaw Staron, Matthias Tichy, and Jörgen Hansson. 2019. “Assessing the Impact of Meta-Model Evolution: A Measure and Its Automotive Application.” *Software and Systems Modeling* 18 (2). Springer Berlin Heidelberg: 1419–1445. doi: 10.1007/s10270-017-0601-1.

Eriksson, Owen, Brian Henderson-Sellers, and Pär J. Agerfalk. 2013. “Ontological and Linguistic Metamodelling Revisited: A Language Use Approach.” *Information and Software Technology* 55 (12). Elsevier B.V.: 2099–2124. doi: 10.1016/j.infsof.2013.07.008.

Fatehah, Murni, Vitaliy Mezhuyev, and Mostafa Al-Emran. 2021. “A Systematic Review of Metamodelling in Software Engineering.” *Recent Advances in Intelligent Systems and Smart Applications* 295: 3–27.

Filatova, Tatiana, Anne Van Der Veen, and Alexey Voinov. 2008. “An Agent-Based Model for Exploring Land Market Mechanisms for Coastal Zone Management,” no. January.

Fischer, Gerhard. 2012. “Context-Aware Systems: The ‘Right’ Information, at the ‘Right’ Time, in the ‘Right’ Place, in the ‘Right’ Way, to the ‘Right’ Person,” 287–294.

Floch, Antoine, Tomofumi Yuki, Clement Guy, Steven Derrien, Benoit Combemale, Sanjay Rajopadhye, and Robert France. 2011. “Model-Driven Engineering and Optimizing Compilers: A Bridge Too Far?” 6981: 608–622. doi: 10.1007/978-3-642-24485-8_45.

France, Robert, and Bernhard Rumpe. 2007. “Model-Driven Development of Complex Software: A Research Roadmap Model-Driven Development of Complex Software: A Research Roadmap,” no. June. doi: 10.1109/FOSE.2007.14.

Gascueña, José M., Elena Navarro, and Antonio Fernández-Caballero. 2012. “Model-Driven Engineering Techniques for the Development of Multi-Agent Systems.” *Engineering Applications of Artificial Intelligence* 25 (1): 159–173. doi: 10.1016/j.engappai.2011.08.008.

Goeken, Matthias, and Stefanie Alter. 2009. “Towards Conceptual Metamodeling of IT Governance Frameworks Approach - Use -– Benefits,” no. Cmmi: 1–10.

Grundy, John C., John Hosking, Karen Na Li, Norhayati Mohd Ali, Jun Huh, and Richard Lei Li. 2013. “Generating Domain-Specific Visual Language Tools from Abstract Visual Specifications.” *IEEE Transactions on Software Engineering* 39 (4): 487–515. doi: 10.1109/TSE.2012.33.

Hegedus, Abel, Dénes Harmath, and Zoltán Ujhelyi. 2017. “VIATRA/Query/UserDocumentation/API/BaseIndexer.” https://wiki.eclipse.org/VIATRA/Query/UserDocumentation/API/BaseIndexer.

Hinkel, Georg, and Misha Strittmatter. 2017. *On Using Sarkar Metrics to Evaluate the Modularity of Metamodels*. doi: 10.5220/0006105502530260.

Hutchinson, John, Mark Rouncefield, and Jon Whittle. 2011. “Model-Driven Engineering Practices in Industry.” *33rd International Conference on Software Engineering (ICSE 2011)*, 633–642. doi: 10.1145/1985793.1985882.

Ii, Leo C Ureel, and Charles Wallace. 2015. “WebTA: Automated Iterative Critique of Student Programming Assignments.”

Kessentini, Wael, and Manuel Wimmer. 2018. “Automated Metamodel / Model Co-Evolution: A Search-Based Approach Automated Metamodel / Model Co-Evolution: A Search-Based Approach,” no. September. doi: 10.1016/j.infsof.2018.09.003.

Knauss, Eric, Daniel Lübke, and Sebastian Meyer. 2009. “Feedback-Driven Requirements Engineering: The Heuristic Requirements Assistant.” *Proceedings - International Conference on Software Engineering*, 587–590. doi: 10.1109/ICSE.2009.5070562.

Lewis, James R., and Jeff Sauro. 2009. “The Factor Structure of the System Usability Scale.” In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5619 LNCS:94–103. doi: 10.1007/978-3-642-02806-9_12.

López-Fernández, Jesús J., Esther Guerra, and Juan de Lara. 2014. “Assessing the Quality of Meta-Models.”

Ma, Zhiyi, Xiao He, and Chao Liu. 2013. “Assessing the Quality of Metamodels.” *Bmj* 329 (4): 621–621. doi: 10.1136/bmj.329.7466.621-b.

Madni, Azad M., and Michael Sievers. 2018. “Model-Based Systems Engineering: Motivation, Current Status, and Research Opportunities.” *Systems Engineering* 21 (3). John Wiley and Sons Inc.: 172–190. doi: 10.1002/sys.21438.

Martínez-García, A., J. A. García-García, M J Escalona, and C. L. Parra-Calderón. 2015. “Working with the HL7 Metamodel in a Model Driven Engineering Context.” *Journal of Biomedical Informatics* 57: 415–424. doi: 10.1016/j.jbi.2015.09.001.

Mohammed, Osman, Norhayati Mohd Ali, Novia Admodisastro, and Jamilah Din. 2017. “Inconsistency Detection of Model and Code via Critic-Based Approach,” 1–5. doi: 10.1166/asl.2011.1261.

Nicola, Matthias, and Jasmi John. 2014. “XML Parsing: A Threat to Database Performance XML Parsing: A Threat to Database Performance,” no. November: 1–5. doi: 10.1145/956863.956898.

Paasivaara, Maria, and Casper Lassenius. 2004. “Using Iterative and Incremental Processes in Global Software Development.”

Qattous, Hazem Kathem. 2009. “Constraint Specification by Example in a Meta-CASE Tool.” *Proceedings of the Doctoral Symposium for ESEC/FSE on Doctoral Symposium*, 13–16. doi: 10.1145/1595782.1595787.

Rabbi, Fazle, Yngve Lamo, and Wendy Maccaull. 2014. “A Flexible Metamodelling Approach for Healthcare Systems.”

Robbins, Jason E, and David F Redmiles. 2000. “Cognitive Support, UML Adherence, and XMI Interchange in Argo / UML.”

Rosique, Francisca, and Fernando Losilla. 2018. “A Domain Specific Language for Smart Cities †,” no. November 2017. doi: 10.3390/ecsa-4-04926.

Salleh, Masrina A, Mahadi Bahari, and Nor Hidayati Zakaria. 2018. “ScienceDirect An Overview of Software Functionality Service: A Systematic Literature Review.” *Procedia Computer Science* 124. Elsevier B.V.: 337–344. doi: 10.1016/j.procs.2017.12.163.

Sendall, Shane, and Wojtek Kozaczynski. 2003. “Model Transformation: The Heart and Soul of Model-Driven Software Development.” *IEEE Software* 20 (5): 42–45. doi: 10.1109/MS.2003.1231150.

Trochim, William. 2006. “Qualitative Validity.” *Research Methods Knowledge Base*, January.

Tymchuk, Yuriy, Mohammad Ghafari, and Oscar Nierstrasz. 2016. “When QualityAssistant Meets Pharo Enforced Code Critiques Motivate More Valuable Rules.” *International Workshop on Smalltalk Technologies*.

Tymchuk, Yuriy, Andrea Mocci, and Michele Lanza. 2015. “Code Review: Veni, ViDI, Vici.”

van Emde Boas, Ghica. 2004. “Template Programming for Model-Driven Code Generation Model-Driven Software Development,” 1–17.

Wachsmuth, Guido. 2007. “Metamodel Adaptation and Model Co-Adaptation.” *ECOOP 2007–Object-Oriented Programming* 4609: 600–624. doi: 10.1007/978-3-540-73589-2_28.

Williams, James R, Athanasios Zolotas, Nicholas Matragkas, Louis M Rose, Dimitios S Kolovos, Richard F Paige, and Fiona A C Polack. 2013. “What Do Metamodels Really Look Like?” *EESSMOD @ MoDELS*, no. 1078: 55–60.

Wurster, Michael, Uwe Breitenbücher, Michael Falkenthal, Christoph Krieger, Frank Leymann, Karoline Saatkamp, and Jacopo Soldani. 2020. “The Essential Deployment Metamodel: A Systematic Review of Deployment Automation Technologies.” *SICS Software-Intensive Cyber-Physical Systems* 35 (1). Springer Berlin Heidelberg: 63–75. doi: 10.1007/s00450-019-00412-x.