



Run Run Shaw Library

香港城市大學
City University of Hong Kong

Copyright Warning

Use of this thesis/dissertation/project is for the purpose of private study or scholarly research only. ***Users must comply with the Copyright Ordinance.***

Anyone who consults this thesis/dissertation/project is understood to recognise that its copyright rests with its author and that no part of it may be reproduced without the author's prior written consent.

CITY UNIVERSITY OF HONG KONG
香港城市大學

A Study of Traditional and AI-driven
Approaches for Agile Requirements
Engineering
關於傳統以及人工智能驅動方法
在敏捷需求工程中的研究

Submitted to
Department of Computer Science
電腦科學系
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
哲學博士學位

by

LI Yishu
李一舒

AUGUST 2024
二零二四年八月

Declaration

I declare that the thesis entitled “A Study of Traditional and AI-driven Approaches for Agile Requirements Engineering” represents my own work, except where due acknowledgments are made, and that it has not been previously included in a thesis, dissertation or report submitted to this university or any other institution for a degree, diploma, or other qualifications.

LI Yishu
AUGUST 2024

Abstract

In recent decades, the agile software development method has seen extensive adoption across companies of various sizes, serving as a cornerstone in the software development process. Requirements Engineering (RE) stands pivotal in this process, providing the foundation for effort management, architectural design, development framework construction, and testing strategy definition. Understanding the challenges inherent in agile RE within real-world contexts is crucial for developing advanced RE techniques tailored to the complexities of agile development.

This thesis presents two case studies conducted on industry-academia projects to explore the challenges encountered in agile RE within real-world contexts. Through action research methodology, traditional RE techniques were applied to address identified challenges. Furthermore, the thesis proposes leveraging Large Language Models (LLMs) to automate certain agile RE tasks, such as requirements elicitation and analysis, to reduce human effort in agile development.

Unprecedented global challenges, exemplified by the COVID-19 pandemic, have necessitated widespread adoption of Work-From-Home (WFH) arrangements, presenting formidable obstacles in conveying requirements within agile RE frameworks. Given the expected continuation of WFH practices post-COVID, identifying and addressing these challenges is paramount. Amid the pandemic, we engaged in an industry-academia collaboration, employing action research methodology to comprehensively analyze agile RE practices within the WFH context. Through our study, key challenges were identified, and collaborative RE techniques were proposed and iteratively refined through three intervention cycles, complemented by in-depth interviews for thorough analysis. Additionally, insights gleaned from this thesis provide valuable guidance on effective collaborative RE techniques and lessons learned.

Machine Learning (ML) has emerged as a core technology across various domains, garnering substantial attention in Software Engineering (SE). However, the complexity inherent in developing ML applications introduces additional challenges

to RE activities. Addressing concerns raised by RE researchers, this thesis aims to observe RE activities for ML-enabled systems in real-world contexts, focusing on an ML-enabled FinTech project. By engaging data scientists to clarify ML-related requirements, our study explores RE difficulties from both the data scientist and requirement engineer perspectives. Additionally, a proposed RE framework iteratively adapts to data and model relevance in ML-enabled FinTech application development, facilitating the completion of RE activities tailored to ML-specific requirements.

Within agile RE frameworks, Generating Acceptance Criteria (GAC) to elaborate user stories assumes critical importance during sprint planning. However, the lack of labeled datasets tailored specifically to User Stories associated with Acceptance Criteria (US-AC) poses significant challenges for supervised learning techniques that attempt to automate this process. Recent advancements in LLM technology offer potential solutions for automating the process, motivating the proposal of an automated framework named SimAC. SimAC leverages LLMs to simulate collaborative practices within an agile environment, facilitating improved GAC generation. Owing to the unavailability of ground truths, we invited practitioners to build a gold standard serving as a benchmark to evaluate the completeness and validity of auto-generated US-AC against human-crafted ones. Additionally, we invited eight experienced agile practitioners to evaluate the quality of US-AC using the INVEST framework. The results demonstrate consistent improvements across all tested LLMs, including the LLaMA and GPT-3.5 series. Furthermore, our study also provides comprehensive case studies to illustrate SimAC's effectiveness and limitations, shedding light on the potential of LLMs in automated agile RE.

Finally, for further analyzing the requirements, automating the derivation of conceptual models, such as class diagrams, from user stories holds promise in streamlining agile projects. Our study explores the potential of LLMs and a Chain-of-Thought (CoT) prompting technique, namely CoTUML, tailored to automate this task. Through a comprehensive preliminary study, we compare LLM-based approaches with guided human extraction, demonstrating the superiority of LLM-based approaches, particularly when combined with well-crafted few-shot prompts. CoTUML further enhances the comprehensiveness of relationship identification. Additionally, qualitative analyses shed light on areas of suboptimal performance, offering avenues for improvement and guiding future research in requirement analysis automation.

Keywords: Agile Requirements Engineering, Action Research, Large Language Models, Prompt Engineering.

CITY UNIVERSITY OF HONG KONG
Qualifying Panel and Examination Panel

Surname: LI
First Name: Yishu
Degree: Doctor of Philosophy
College/Department: Department of Computer Science

The Qualifying Panel of the above student is composed of:

Supervisor(s)

Prof. KEUNG Wai Jacky Department of Computer Science
City University of Hong Kong

Qualifying Panel Member(s)

Prof. WANG Shiqi Department of Computer Science
City University of Hong Kong

Prof. SONG Linqi Department of Computer Science
City University of Hong Kong

This thesis has been examined and approved by the following examiners:

Prof. KEUNG Wai Jacky Department of Computer Science
City University of Hong Kong

Prof. WONG Hau San Department of Computer Science
City University of Hong Kong

Prof. LEUNG Wing Ho Howard Department of Computer Science
City University of Hong Kong

Dr. LAI Richard Department of Computer Science and
Information Technology
La Trobe University

Acknowledgements

First of all, I would like to express my sincere gratitude to my supervisor, Prof. Jacky Keung. Throughout my entire Ph.D. journey, he has provided me with immense support and guidance. He encouraged me to collaborate with industry partners and explore the tasks in real-world cases. Additionally, he taught me how to conduct research in the field of software engineering and how to collaborate with the broader community. Beyond research, he imparted essential skills in collaboration and teaching, which have greatly supported my career development. These experiences have been instrumental in my continuous growth and improvement.

I would also like to thank my Ph.D. qualifying panel members, Prof. Shiqi Wang and Prof. Linqi Song, for their valuable time and rigorous evaluation. Their constructive comments and probing questions have significantly enhanced the quality of my work. Furthermore, I am also very grateful to my dissertation examiners, Prof. Hau San Wong, Prof. Wing Ho Howard Leung, and Prof. Richard Lai, for their invaluable feedback and suggestions on my thesis. Their constructive insights have inspired me and helped me improve the quality of this thesis.

Special appreciation is extended to my collaborators, Dr. Zhen Yang and Dr. Kwabena Ebo Bennie, for their support and guidance throughout this journey. Furthermore, I am grateful to our industrial partners for their invaluable assistance. Additionally, I would like to express gratitude to all the invited practitioners who participated in the human evaluation process, enriching our studies with their insights and expertise.

I would like to express my heartfelt gratitude to my family for their constant support and encouragement, particularly during the challenging time of the pandemic. Spending nearly one and a half years away from home has limited my time with them and reduced opportunities to care for my parents. I deeply appreciate their understanding and patience. I am especially thankful to my little brother, Dr. Li Yuen Kit Howard, for his company, support, and care over these past four years.

Finally, my sincere thanks go to my dear besties Yuki and PigGroup, as well as my lovely lab mates Xiaoxue Ma, Jingyu Zhang, and Yihan Liao, for their companionship and support throughout my Ph.D. journey.

List of Publications

Published and Under Reviewed Journal Papers

- (1) **Yishu Li**, Jacky Keung, Zhen Yang, Xiaoxue Ma, Jingyu Zhang, and Shuo Liu. “SimAC: Simulating Agile Collaboration to Generate Acceptance Criteria in User Story Elaboration.” *Automated Software Engineering Journal (ASE)*, 2024, doi:10.1007/s10515-024-00448-7
- (2) Jingyu Zhang, Jacky Keung, Yan Xiao, Yihan Liao, **Yishu Li**, Xiaoxue Ma. “UniAda: Universal Adaptive Multi-objective Adversarial Attack for End-to-End Autonomous Driving Systems.” *IEEE Transactions on Reliability (TR)*, 2024, doi:10.1109/TR.2024.3394894
- (3) Shuo Liu, Jacky Wai Keung, Zhen Yang, Yihan Liao, **Yishu Li**. “TerGEC: A Graph Enhanced Contrastive Approach for Program Termination Analysis.” *Science of Computer Programming (SCP)*, 2024, doi:10.1016/j.scico.2024.103141
- (4) Zhen Yang, Jacky Keung, Zeyu Sun, Yunfei Zhao, Ge Li, Zhi Jin, Shuo Liu, and **Yishu Li**. “Improving domain-specific neural code generation with few-shot meta”. *Information and Software Technology (IST)*, 2023, doi: 10.1016/j.infsof.2023.107365.
- (5) Xiaoxue Ma, Jacky Keung, Pinjia He, Xiao Yu, Yan Xiao, and **Yishu Li**. “A Semi-supervised Approach for Industrial Anomaly Detection via Self-adaptive Clustering.” *IEEE Transactions on Industrial Informatics (TII)* 2023, doi: 10.1109/TII.2023.3280246.
- (6) Xiaoxue Ma, Jacky Keung, Xiao Yu, Huiqi Zou, Jingyu Zhang, and **Yishu Li**. “ATTSUM: A Deep Attention-Based Log Summarization Model for Bug Report Title Generation.” *IEEE Transactions on Reliability (TR)*, 2022, doi:10.1109/TR.2023.3236404.

- (7) Xiaoxue Ma, Jacky Keung, Zhen Yang, Xiao Yu, **Yishu Li**, and Hao Zhang. “CASMS: Combining clustering with attention semantic model for identifying security bug reports”. *Information and Software Technology (IST)*, 2022, 147:106906.
- (8) Xiaoxue Ma, Huiqi Zou, Jacky Keung, Federica Sarro, Xiao Yu, **Yishu Li**, Zhen Yang, Jingyu Zhang. “On the Influence of Data Resampling for Deep Learning-Based Anomaly Detection: Insights and Recommendations.” *Transactions on Software Engineering (TSE)*. [Under Review]

Published and Under Reviewed Conference Papers

- (1) **Yishu Li**, Jacky Keung, Kwabena Ebo Bennin, Xiaoxue Ma, Jingyu Zhang, and Zhen Yang. “Agile Requirements Engineering in a Distributed Environment: Experiences from the Software Industry during Unprecedented Global Challenges.” *The 48th IEEE International Conference on Computers, Software, and Applications, IEEE (COMPSAC 2024)*. (Accepted in April 2024)
- (2) **Yishu Li**, Jacky Keung, Xiaoxue Ma, Jingyu Zhang. “LLM-based Class Diagram Derivation from User Stories with Chain-of-Thought Prompting.” *The 48th IEEE International Conference on Computers, Software, and Applications, IEEE (COMPSAC 2024)*. (Accepted in April 2024)
- (3) **Yishu Li**, Jacky Keung, Xiaoxue Ma. “Integrating Generative AI in Software Engineering Education: Practical Strategies” *The 10th International Symposium on Educational Technology, IEEE (ISET 2024)*. (Accepted in March 2024)
- (4) **Yishu Li**, Jacky Keung, Kwabena Ebo Bennin, Xiaoxue Ma, Yangyang Huang, and Jingyu Zhang. “Towards Requirements Engineering Activities for Machine Learning-enabled FinTech Applications” *The 30th Asia-Pacific Software Engineering Conference: APSEC 2023. IEEE p. 121-130.*
- (5) **Yishu Li**, Jacky Keung, Xiaoxue Ma, Jingyu Zhang, Zhen Yang, and Shuo, Liu. “Learning Gaps in Project-based Requirements Engineering Education - A Case Study of Student Projects.” *The 9th International Symposium on Educational Technology, pp.239-243, IEEE (ISET 2023)*.
- (6) Jingyu Zhang, Jacky Keung, Xiaoxue Ma, Xiangyu Li, Yan Xiao, **Yishu Li**, and Wing Kwong Chan. “Enhancing Valid Test Input Generation with Distribution

Awareness for Deep Neural Networks.” *The 48th IEEE International Conference on Computers, Software, and Applications, IEEE (COMPSAC 2024)*.

Table of Contents

Declaration	ii
Abstract	iii
Qualifying Panel and Examination Panel	v
Acknowledgements	vi
List of Publications	viii
List of Figures	xvi
List of Tables	xviii
1 Introduction	1
1.1 Agile RE in WFH Environment	2
1.2 Agile RE in ML-enabled Development	3
1.3 SimAC: LLMs for Requirement Elicitation	4
1.4 CoTUML: LLMs for Requirement Analysis	7
1.5 Contributions	9
1.6 Organizations	10
2 Literature Review	11
2.1 Empirical Methodologies	11
2.1.1 Action Research	11
2.1.2 Practitioner Engagement	12
2.2 RE in the Real-World Context	13
2.2.1 RE in Startup	13
2.2.2 RE in WFH	14
2.2.3 RE for FinTech Development	15

Table of Contents

2.3	RE in Agile Development	16
2.3.1	Communication in Agile RE	16
2.3.2	Elicitation in Agile RE	17
2.3.3	Analysis in Agile RE	19
2.4	RE and AI	20
2.4.1	RE for AI	21
2.4.2	AI for RE	22
2.5	LLMs and Prompts	24
2.5.1	Overview of LLMs	24
2.5.2	Prompt Engineering	25
2.6	Summary	26
3	Agile RE in WFH Environment	27
3.1	Introduction	27
3.2	Methods	29
3.2.1	Study Context	29
3.2.2	WFH and Face-to-Face Agile Practices	29
3.2.3	Agile RE Evaluation Framework	30
3.2.4	Action Research	32
3.2.5	Data Collection	34
3.2.6	Data Analysis	35
3.3	Intervention	36
3.3.1	Cycle I	36
3.3.2	Cycle II	38
3.3.3	Cycle III	41
3.4	Evaluation	42
3.4.1	Constant Collaboration	43
3.4.2	Online Design Thinking	43
3.4.3	Mixed Language Presentation	44
3.4.4	Roster Customer	46
3.5	Threats	47
3.5.1	Transferability	47
3.5.2	Reflexivity	48
3.5.3	Resonance	48
3.6	Summary	48

4	Agile RE in ML-enabled Development	50
4.1	Introduction	50
4.2	Methods	51
4.2.1	Case Selection	51
4.2.2	Study Context	52
4.2.3	Procedure of Study	52
4.2.4	Data Collection	53
4.2.5	Data Analysis	54
4.3	RE Activities	54
4.3.1	Requirement Elicitation	56
4.3.2	Requirement, Data and Model Analysis	57
4.3.3	Requirement Negotiation	58
4.3.4	Requirement Modeling	59
4.3.5	Requirement Validation and Verification	59
4.3.6	Requirement Specification	60
4.4	ML-related Requirement Characteristics	60
4.4.1	Data Requirements	61
4.4.2	Model Requirements	63
4.5	Discussion	65
4.5.1	Difficulties in RE Activities	65
4.5.2	ML-related Requirements	65
4.5.3	Generalization	66
4.6	Threats	67
4.6.1	Multivocality	67
4.6.2	Credibility	68
4.6.3	Transferability	68
4.7	Summary	68
5	SimAC: Requirement Elicitation in Agile RE	70
5.1	Introdcution	70
5.2	Preliminary Analysis	72
5.2.1	Role Division	72
5.2.2	Procedures	73
5.3	SimAC	75
5.3.1	Problem Formulation	75
5.3.2	Overview of SimAC	75

Table of Contents

5.3.3	Phase I: Collaboration	75
5.3.4	Phase II: Deduplication	80
5.4	Experimental Setup	81
5.4.1	Research Questions	81
5.4.2	Dataset	83
5.4.3	Baselines	85
5.4.4	Implementation	86
5.4.5	Evaluation	87
5.5	Experimental Results	93
5.5.1	RQ1: Generality of SimAC	93
5.5.2	RQ2: Ablation Study for Role Groups	97
5.5.3	RQ3: Analysis of K-shot Prompting	99
5.5.4	RQ4: Analysis of Similarity Thresholds	103
5.6	Case Studies	106
5.6.1	Case 1: SimAC on GPT-3.5	106
5.6.2	Case 2: SimAC on OpenLLaMA	109
5.6.3	Case 3: Failures of SimAC	110
5.7	Threats	111
5.7.1	Internal Validity	111
5.7.2	External Validity	112
5.7.3	Conclusion Validity	113
5.7.4	Construct Validity	114
5.8	Summary	114
Appendix 5.A	Guidance of Questionnaire	116
5.A.1	Background Introduction	116
5.A.2	Actions	117
Appendix 5.B	Sample of Evaluation on US-AC	117
6	CoTUML: Requirement Analysis in Agile RE	119
6.1	Introduction	119
6.2	CoTUML	121
6.2.1	Steps of Class Diagram Derivation	121
6.2.2	CoT Prompting	122
6.2.3	Implementation	125
6.3	Experiment	126
6.3.1	Dataset	126

6.3.2	Comparison Baselines	127
6.3.3	Settings	127
6.3.4	Evaluation	128
6.4	Results	129
6.4.1	RQ1: Performance of LLM-based Approaches	129
6.4.2	RQ2: Effective Settings in CoTUML	130
6.5	Case Studies	132
6.5.1	Case Study on Class Identification	132
6.5.2	Case Study on Attribute Identification	134
6.5.3	Case Study on Relationship Identification	135
6.6	Threats	137
6.6.1	Construct Validity	137
6.6.2	Conclusion Validity	137
6.6.3	Internal Validity	138
6.6.4	External Validity	138
6.7	Summary	138
7	Conclusion and Future Work	140
7.1	Studies on Traditional RE Techniques	140
7.1.1	Summary	140
7.1.2	Contributions	141
7.1.3	Limitations and Future Work	142
7.2	Studies on LLM-based RE Techniques	142
7.2.1	Summary	142
7.2.2	Contributions	143
7.2.3	Limitations and Future Work	144
	References	145

List of Figures

1.1	Brief workflow of SimAC.	6
2.1	An example of elaborating user story associated with acceptance criteria.	18
2.2	Agile practice in sprint planning: collaborative acceptance criteria generation from agile team members.	19
3.1	Evaluating challenges and risks exposed from RE activities. Adapted from [1] and integrated observations under WFH context.	32
4.1	Flow of RE activities in the development process of the ML-enabled system in Project ISIP.	56
4.2	Guidance framework intersected in RE activities for developing ML-enabled FinTech systems, consisting of data and model requirement characteristics.	61
5.1	Acceptance criteria generated by LLM acting in different roles.	74
5.2	Two phases of GAC activities: I. collaboration (Stage I: role-based prompt initialization; Stage II: acceptance criteria generation); II deduplication.	76
5.3	Three components in the role-based prompt (P_{ro}): context, role instructions, and K-shot prompting.	77
5.4	Collaborative roles-playing LLM in three-steps of US-AC elaboration (exemplified in <i>gpt-3.5-turbo</i>).	80
5.5	Two components in the regular prompt (P_{re}): context and K-shot prompting.	82
5.6	Case 1: Generate acceptance criteria with the regular and role-based prompt (G28 US3).	108
5.7	Case 1:Generate acceptance criteria from different roles (G28 US3).	109

List of Figures

5.8	Case 2: Generate acceptance criteria with the regular and role-based prompt (G24 US8).	110
5.9	Case 3: Generate acceptance criteria with the regular and role-based prompt in different parameter sizes (G11 US37).	112
6.1	Phases in class diagram derivation.	122
6.2	Structure of CoTUML: three-step CoT promptings in class diagram derivation.	122
6.3	An example of CoT prompting in three steps of class diagram derivation.	126
6.4	Interactive input in the CoT prompting, exemplified in <i>gpt-3.5-turbo</i>	128
6.5	Case study on class identification (PP).	133
6.6	Case study on attribute identification (PP).	135
6.7	Case study on relationship identification (DH).	136

List of Tables

3.1	Team structure and background of developers for ORG	30
3.2	Key similarities and differences in WFH and Face-to-Face environments	31
3.3	Intervention cycles in project development	33
3.4	Sample questions of semi-structured interview	35
3.5	Challenges, RE practices, causes of requirements changes and risks revealed in WFH-based agile RE practices	36
4.1	Summary of ML-related requirements	53
4.2	Difficulties of RE activities in ML-enabled FinTech system, from perspectives of DS and BA	55
5.1	Description of examples in K-shot prompting generated under different scenarios	84
5.2	Details of selected projects in testing dataset D_t	85
5.3	Description of the practitioners included in this study	90
5.4	Average satisfaction on INVEST score, comparative results on US-AC generated by different LLMs	94
5.5	Significance analysis of quality evaluation on different LLMs	95
5.6	Kappa score for assessing inter-rater agreements on the baseline evaluations	95
5.7	Completeness and validity of US-AC generated by different LLMs	96
5.8	Significance analysis of relevance evaluation for different LLMs	97
5.9	Average satisfaction on INVEST score, evaluating by combinations of different role groups	98
5.10	Significance analysis of INVEST score for different settings on combinations of role groups	98
5.11	Kappa score and differences for assessing inter-rater agreements on role group combination evaluations	99

List of Tables

5.12	Completeness and validity in combinations of different role groups . . .	99
5.13	Significance analysis of completeness and validity for different settings on combinations of role groups	100
5.14	Average satisfaction on INVEST score, evaluating by K-shot prompting	100
5.15	Significance analysis of INVEST score for different settings on K-shot prompting	101
5.16	Kappa score and differences for assessing inter-rater agreements on k-shot settings	101
5.17	Completeness and validity in different settings on K-shot prompting .	102
5.18	Significance analysis of completeness and validity for different settings on K-shot prompting	102
5.19	Average satisfaction on INVEST score, evaluating by semantic similarity thresholds	103
5.20	Significance analysis of INVEST score for different settings on similarity threshold	104
5.21	Kappa score and differences for assessing inter-rater agreements on similarity threshold	104
5.22	Completeness and validity in different settings on similarity threshold	105
5.23	Significance analysis of completeness and validity for different settings on similarity threshold	105
6.1	Completeness and validity on PP	130
6.2	Completeness and validity on DH	131
6.3	Effectiveness of prompting interaction	131
6.4	Effectiveness of number of examples	132
6.5	Partial identified and unidentified classes	133
6.6	Unidentified and erroneously identified attributes	134
6.7	Unidentified and erroneously identified relationships	136

Chapter 1

Introduction

Requirements Engineering (RE) serves as the fundamental cornerstone of the software development process, providing the basis for effort management, architectural design, development framework construction, and definition of testing strategy. The quality of requirement pertains to the specific attributes of requirements that contribute to the successful and cost-effective realization of a system, addressing the user's needs [2, 3].

Over the past decades, the agile software development method has experienced widespread adoption and usage by both small and large established companies [4, 5]. Wang et al. [6] highlighted the significance of RE within the context of agile practice. Their findings indicated that over 90% of the interviewed agile practitioners regarded RE as an important aspect of their work.

This thesis first studies the requirement activities of an industry-academia software development project that adopted agile practices in a real-world context. Based on the challenges exposed in the case, we approached the action research methodology for the project team, delivering the traditional RE techniques in the project to assist in alleviating the observed challenges. Recently, Large Language Models (LLMs), pre-trained by a huge amount of human-written text/code, have drawn immense interest in the Software Engineering (SE) community owing to their powerful generality even without supervised fine-tuning [7–9]. Motivated by the powerful generality of LLMs, we proposed some LLM-based frameworks to automate the agile RE tasks.

This chapter first introduces the context of the approached action research in the industry-academia project, including agile RE from the perspectives of Work-From-Home (WFH) in Section 1.1 and Machine Learning (ML) development in Section 1.2. Subsequently, Section 1.3 introduces the LLM-based technique developed for generating acceptance criteria during requirements elicitation, while Section 1.4

demonstrates different prompting techniques incorporated within LLMs aimed at expediting the derivation of class diagrams for requirement analysis. The contributions are presented in Section 1.5. Lastly, Section 1.6 presents the outline of this thesis.

1.1 Agile RE in WFH Environment

The industry-academia project was collaborated by the City University of Hong Kong and a startup company in Hong Kong during the COVID-19 pandemic. The significance of learning agile RE in the WFH context has been underscored by the outbreak of the COVID-19 pandemic and subsequent lockdown measures, which necessitated a shift to remote work for software engineers in many companies [10, 11]. Agile teams heavily rely on collaboration, making the WFH conditions particularly challenging for requirements engineers [12]. Notably, the number of professionals working from home on a permanent basis is expected to continue increasing [13]. In fact, a CFO survey by Gartner reported that 74% of companies plan to transition some employees to permanent remote work arrangements after COVID-19 [14].

The WFH experience during the COVID-19 restrictions differs from the typical WFH process and may prove stressful and frustrating for software developers [15]. Particularly, individuals in software teams who simultaneously faced home teaching responsibilities found themselves in atypical circumstances that posed challenges to productive software development [15].

Therefore, gaining insights into the RE activities within this extreme WFH context can serve as a valuable contingency strategy for unforeseen WFH scenarios in the post-COVID era. By adopting the methodology of action research, we comprehensively delineate the primary challenges experienced in the WFH environment. In addressing these challenges, a sequence of collaborative RE techniques, such as constant collaboration, online design thinking, mixed language presentation, and roster customer, were implemented across three intervention phases. Subsequent to these interventions, interviews were conducted to scrutinize the outcomes, facilitating insights into collaborative RE methodologies and accumulating invaluable lessons.

1.2 Agile RE in ML-enabled Development

In the collaborative project, the deliverable is a FinTech product that integrates ML techniques. ML is a core technology that has been applied in different application domains in recent years and has gained significant attention in SE. ML was successfully and widely applied in image recognition, Natural Language Processing (NLP), and outperforming humans in complex games [16, 17]. More and more ML components are developed and integrated to improve products and processes [17] with one of the success factors enabling ongoing ML being the large amounts of data [17].

However, developing ML-enabled systems requires additional expertise in data science [18], which introduces challenges in the SE development process [19]. For instance, data and models should be validated as thoroughly as code for ML components, which lacks best practices for doing so [19, 20]. In the SE development process, the RE artifact is the core foundation for building a development framework and defining testing strategy [21]. Serving in the key SE development process, RE practitioners inevitably face critical challenges in handling ML-related requirements. In the survey conducted in [22], the respondents reported that RE is the most challenging activity when developing ML systems.

Scharinger et al. [23] examined the experiences of industrial ML pioneers, and they argued that they expect an increase in the discussions of RE for ML projects in the SE community. They note that ML-enabled software development projects rarely followed the structured RE processes used by non-ML-related projects [23]. Results of the interview conducted in [17] indicate that RE must evolve to specify the ML systems. Other studies [24, 25] have pointed out the problems and challenges related to RE for ML, such as the unclear requirements specification definitions [25]. However, these studies discussed the general issues but did not combine in-depth review with real-world context.

Simultaneously, the Financial and Banking sector is one of the most popular domains to be adopting and integrating ML in their software [26]. The advantages of integrating ML technologies in the domain of Financial Technology (FinTech) include providing more competitive and efficient services. Applying ML technology in financial projects has increased the interest of stakeholders, especially in Hong Kong, which is one of the most prominent international financial centers in the world¹. In a survey conducted in financial services [27], the results revealed that 55% of the

¹<https://www.hkma.gov.hk/eng/key-functions/international-financial-centre/hong-kong-as-an-international-financial-centre/>

respondents were implementing Artificial Intelligence (AI) enabled products while 77% of them expected AI as an essential factor to their business. Several global surveys suggest that ML is poised to become the backbone of the financial industry in the near future [28].

Therefore, understanding RE activities in ML-enabled systems from the FinTech domain can assist in developing more competitive FinTech products in the financial center. Inspired by expectations from the RE community and the interests of stakeholders from the financial industry, we conducted an in-depth study on the RE activities in the industry-academia project. Our researchers participated in the collaboration to explore the difficulties of RE activities in a real-world development context. Based on the observation, we employed the intervention that guided and altered the RE activities in the collaborative project, which aimed to explore what requirement characteristics should be considered when handling ML-related requirements.

Given the significance of data and model considerations in the development of ML-enabled FinTech applications, a RE guidance framework is proposed. This framework, incorporating specific ML-related requirement characteristics essential for the completion of ML-enabled application development tasks, is iteratively employed to make changes to RE activities. Feedback from practitioners suggests that these practices effectively mitigate challenges associated with enhancing data quality and validating model requirements within RE activities. Furthermore, insights gained from both researchers and practitioners are studied, offering practical recommendations to SE and RE communities grappling with similar challenges in analogous contexts.

1.3 SimAC: LLMs for Requirement Elicitation

Elaborating user stories emerged as the prevailing method for requirements elicitation within agile projects [6]. A user story is a short-loose text describing the needs of customers, while acceptance criteria usually appear as an extension of the user story. Acceptance criteria define what is “done” for the given user story [29], enhancing the comprehension of practitioners on the requirements and facilitating the development process. The BABOK Guide², one of the most popular industry standards of presenting RE techniques [30], emphasizes the critical nature of user story elaboration, as it constitutes a fundamental component of the agile requirement analysis process. During

²<https://www.iiba.org/career-resources/a-business-analysis-professionals-foundation-for-success/agile-extension/>

1.3 SimAC: LLMs for Requirement Elicitation

elaboration, the detailed design and acceptance criteria for the user story are defined in order to deliver a working solution. However, this process involves iterative collaboration among agile team members to refine and finalize the details in each sprint. Consequently, Generating Acceptance Criteria (GAC) during the process of user story elaboration is labor-intensive but essential in agile RE.

Agile methodologies aim to enhance product development efficiency [31]. Automating labor-intensive tasks in agile development provides a practical means of streamlining processes, thereby enhancing efficiency. GAC demands extensive human involvement to ensure a comprehensive understanding of project requirements. Although automating this task may inadvertently result in team members lacking a full grasp of requirements, the generation of high-quality acceptance criteria at the initial stage can establish a solid foundation for further refinement, thereby facilitating more efficient collaboration. Building upon this understanding, our study aims to explore an automation paradigm for GAC to reduce the initial human effort involved in the process.

Few studies have dedicated themselves to automating GAC due to the absence of labeled datasets in User Stories associated with Acceptance Criteria (US-AC). That is to say, task-specific learning-based techniques have difficulty in automating this process. Recently, LLMs, pre-trained by a huge amount of human-written text/code, have drawn immense interest in the SE community owing to their powerful generality even without supervised fine-tuning [7–9]. For example, Xia et al. [7] revealed that directly applying state-of-the-art LLMs can substantially outperform all existing automated program-repairing techniques since traditional learning-based techniques are limited to the reliance on bug-fixing datasets to craft fix templates. Li et al. [8] proposed to approach role-playing and inception prompting to guide the communicative LLMs toward the programming task autonomously. In summary, LLM-based approaches circumventing supervised fine-tuning bring a chance to transition the labor-intensive SE tasks toward an automated paradigm.

Motivated by the powerful generality of LLMs, we propose SimAC, a framework that **Simulates** agile collaboration to generate **Acceptance Criteria** in user story elaboration by employing LLMs. Figure 1.1 presents the brief of SimAC, which encompasses two phases: collaboration and deduplication. In the collaboration phase, an LLM is initiated with prompts that provide instructions and illustrative examples to demonstrate how collaborative tasks interact. Following a prescribed procedure of create-update-update, the LLM acts collaboratively in three groups of roles: the requirement analyst, quality analyst, and other team members. The requirement analyst

Chapter 1. Introduction

first creates an initial suite of US-AC based on the responsibilities of comprehending and transferring requirements. Subsequently, the quality analyst updates the US-AC to supplement some testable details. After that, the US-AC is further updated by other team members based on their expertise. In the deduplication phase, we utilize the Spacy [32], an open-source Python library, to compute the similarity score of the generated acceptance criteria, which aims to eliminate the redundancy and keep the most informative items in the US-AC instance. The finalized US-AC will serve as a working solution reference for the practitioners.

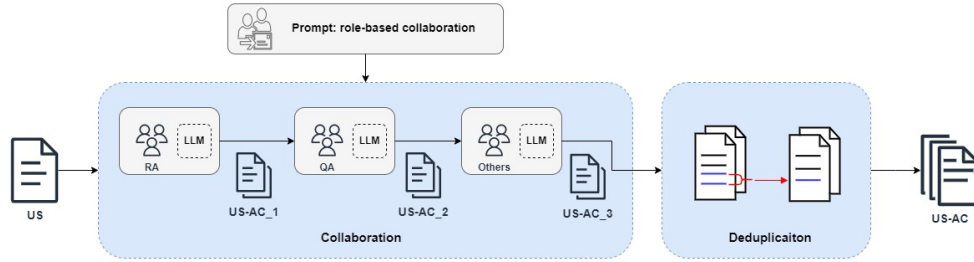


Fig. 1.1 Brief workflow of SimAC.

We conducted an extensive empirical study using a dataset sourced from [33] to examine the effectiveness of SimAC. Considering the incorporation of industry perspective and the lack of ground truths in the US-AC dataset, we adopt human evaluation to evaluate both the quality and the relevance of the auto-generated US-AC. The quality assessment aims to determine whether the US-AC follows INVEST (Independent-Negotiable-Valuable-Estimable-Small-Testable) standards as proposed in [34]. Meanwhile, the relevance assessment serves to measure the extent to which the auto-generated US-AC provides relevant information compared to human-crafted acceptance criteria, with indicators of completeness and validity.

Owing to the huge overhead on time during the human evaluation, we randomly selected 30 samples of US-AC for analysis. During the quality evaluation, we engaged experienced agile practitioners to score US-AC from 0 to 4 in each individual metric of INVEST. Furthermore, to evaluate the relevance, practitioners were invited to familiarize themselves with the project context and then crafted gold standards for the aforementioned 30 examples. The gold standard serves as a benchmark for evaluating completeness and validity.

The quality assessment reveals a significant improvement in satisfaction across all INVEST metrics for the generated US-AC through SimAC. Besides, SimAC also boosts all its underlying LLMs' performance (including LLaMA and GPT-3.5 series), demonstrating its generality among different backbones. Particularly, *gpt-3.5-turbo*

model obtains the biggest improvement from 2.89 to 3.21 in terms of the INVEST score on average. Furthermore, the relevance assessment indicates that US-ACs generated by SimAC with *gpt-3.5-turbo* model achieve 60.60% completeness and 51.29% validity, representing improvements of 29.48% and 15.56% respectively. The introduction of role-based prompts in SimAC effectively enhanced the completeness of the output across all baselines. Our results indicate that the US-AC generated by our proposed framework can receive satisfactory quality and relevance. Such high-quality output can serve as a reliable reference for further refinement in agile sprint planning, thereby augmenting collaboration efficiency.

1.4 CoTUMML: LLMs for Requirement Analysis

In the analysis phase of agile RE, the refined user stories can be transformed into more detailed and specific specifications. One effective approach for this refinement process is the derivation of conceptual models, such as Unified Modeling Language (UML) diagrams [35].

Bragilovski et al. [36] proposed guidelines aimed at enhancing flexibility and reducing cognitive effort in the task of translating user stories into UML class diagrams. However, despite these advancements, human involvement continues to be necessary. LLMs have demonstrated exceptional proficiency in NLP tasks, making them highly relevant in the field of RE, where effective handling of natural language is critical. In particular, ChatGPT has gained significant traction in RE [37], being employed for tasks such as identifying inconsistencies in natural language requirements [38].

Based on the aforementioned considerations, the main objective of our study is to assess the effectiveness of LLMs in the process of deriving class diagrams within the agile RE context. The fundamental motivation behind this research is to establish an automated approach that utilizes LLMs to generate class diagrams directly from user stories. This transition towards automation holds promising potential for reducing human effort while providing a valuable reference for requirement analysis.

The design of appropriate prompts plays a crucial role in maximizing the effectiveness of LLMs [39]. One state-of-the-art prompting technique that has shown promising results is Chain-of-Thought (CoT) prompting, as introduced in [40]. CoT prompting involves instructing LLMs to generate a series of intermediate natural language reasoning steps, known as a CoT, before producing the final output. These reasoning steps provide a step-by-step description of the problem-solving process.

Chapter 1. Introduction

By utilizing CoT prompting, LLMs are guided to think through the problem in a structured manner, enabling them to generate more accurate and coherent responses. This approach leverages the inherent capabilities of LLMs to perform complex natural language reasoning and provides a structured framework for organizing their outputs. The generated CoT serves as a transparent representation of the model’s decision-making process, allowing for a better understanding and evaluation of its outputs.

We aim to enhance the reliability and comprehensibility of the LLM-based approach for deriving class diagrams from user stories. To reduce the need for human intervention in the process, we have developed a three-phase CoT prompting incorporated with the LLM approach, namely **CoTUML**, for automating the derivation of class diagrams. This proposed method involves three key phases in its reasoning steps: class identification, attribute identification, and relationship identification.

In CoTUML, the prompting introduces step-by-step reasoning. During the class identification phase, CoTUML begins by identifying the primary entities derived from user stories. Subsequently, the focus shifts to recognizing attribute-related indicators in the second phase. Finally, it examines operation-related indicators to determine the relationships among the identified classes.

This approach is applied to user stories obtained from two different projects, as sourced from [36]. These user stories are utilized as input for the derivation process, and the resultant outputs are formatted to demonstrate identified classes, attributes, and relationships. These generated outputs serve as a reference for the subsequent construction of the class diagram.

We conducted a comprehensive analysis of the LLM-generated output. This analysis covered both quantitative and qualitative aspects to evaluate the effectiveness and accuracy of the LLM-based approaches, especially CoTUML. The quantitative analysis involved measuring completeness and validity to assess the performance of the LLM-based method in identifying classes, attributes, and relationships. Furthermore, the qualitative analysis involved a thorough examination of the generated class diagrams to evaluate their overall quality and adherence to the user stories. We aimed to gain insights into the capabilities and limitations of the LLM-based approach with different promoting techniques in deriving class diagrams from user stories.

The result shows that the LLM-based approach, when effectively utilized with carefully designed few-shot prompting, surpasses guided human extraction in the identification of classes from user stories. The integration of CoT prompting also improves the comprehensiveness of relationship identification between the identified

classes. However, in terms of attribute identification, the LLM-based approach, despite incorporating both few-shot prompting and CoT prompting, demonstrates inferior performance compared to human extraction. Our case studies provide a systematic summary and analysis of instances wherein the CoT prompting enhances the performance and instances where the LLM-based approach does not exhibit improvement. These insights are valuable for further exploration and refinement in this research direction. By identifying the limitations and areas where the LLM-based approach falls short, we can focus on addressing these challenges and improving the performance of automated class diagram derivation from user stories.

1.5 Contributions

This thesis undertakes an investigation into the challenges encountered in RE within real-world scenarios, particularly focusing on the context of agile software development. It proposes novel frameworks leveraging LLMs to automate RE tasks within agile development processes.

The first contribution is to identify the challenges of agile software development in the real-world context.

In the WFH environment, especially in unexpected challenges such as the COVID-19 pandemic. To alleviate the challenges, some traditional human-centered techniques are proposed, and some of the lessons have been learned. It provides a comprehensive account of the challenges and practices of agile RE within the WFH context of unprecedented global challenges. By conducting an in-depth analysis of a single case with shared WFH experiences, the findings can be generalized to other similar contexts of the uncertain distributed working environment.

Furthermore, the study delineates the challenges specific to agile software development in the realm of ML-enabled products. Recognizing the pivotal role of data analysis in ML-related software development, a guidance framework encompassing factors from both model and data domains is proposed to aid project development. The implementation of this framework is critically evaluated.

The second contribution involves the formulation of two LLM-based frameworks, SimAC and CoTUML, aimed at automating agile RE tasks.

SimAC incorporates the role-based prompt to initiate LLMs to collaborate and generate high-quality acceptance criteria for a given user story. The output provides meaningful references for practitioners, reducing their effort in drafting and refining acceptance criteria. To the best of our knowledge, it is the first automated framework

Chapter 1. Introduction

for generating acceptance criteria. We conducted extensive studies to evaluate SimAC implemented on four LLMs covering diverse series and sizes. To reflect the real-world feedback and overcome the ground truth unavailability problem, we engaged industrial practitioners to evaluate the quality of US-AC, demonstrating the effectiveness of SimAC and proving the potential of LLMs in the GAC task. We released the gold standard for the dataset crafted by industry practitioners. This dataset establishes a valuable foundation for further research on agile RE.

CoTUML is a novel prompting technique tailored explicitly for the derivation of class diagrams. Overall, our findings support the potential of the LLM-based approach to automate certain aspects of class diagram derivation, particularly in class and relationship identification. However, more attention and investigation are needed to overcome the limitations in attribute identification and further enhance the accuracy and reliability of the LLM-based approach. These research insights contribute to the ongoing exploration of leveraging LLMs in agile RE and provide a foundation for future advancements in this field.

1.6 Organizations

The rest of this thesis is organized as follows.

In Chapter 2, we review related work to this thesis, including empirical methods, RE in the real-world context, RE in Agile Development, RE and AI, and LLMs for RE.

Chapter 3 introduces the challenges associated with agile RE in the WFH context. We propose collaborative RE techniques to mitigate these challenges and share the lessons learned.

In Chapter 4, we discuss the challenges encountered in RE activities during the development of ML-enabled systems. Based on an understanding of data and model relevance in real-world scenarios, we propose a guidance framework that provides checklist guidance for conducting RE activities tailored to ML-enabled system development.

Chapter 5 describes SimAC, which incorporates role-based prompting into LLMs to simulate agile practices, automating the task of GAC during agile sprint planning.

Chapter 6 presents CoTUML, which introduces CoT prompting in LLMs to derive class diagrams from user stories.

We conclude the thesis in Chapter 7, which includes summarizing the research findings, highlighting the main contributions, and offering suggestions for future work.

Chapter 2

Literature Review

This chapter reviews the prior work relevant to this thesis. Firstly, we introduce the empirical methodologies employed in this thesis in Section 2.1. Following this, we review related work, including RE in a real-world context (Section 2.2), RE in agile development (Section 2.3), RE and AI (Section 2.4), and overview of LLMs and prompts (Section 2.5).

2.1 Empirical Methodologies

In this section, we discuss the rationale for employing action research in our studies and the inclusion of practitioners in case studies for empirical analysis to evaluate our proposed frameworks.

2.1.1 Action Research

Action research has gained significant popularity in recent years as a research methodology [41, 42]. This approach emphasizes organizational learning as an integral part of the research process [42]. SE, being an applied engineering science, largely relies on empirical observations and, increasingly, ethnography for theory development [42]. Additionally, new methods are frequently applied in industrial projects, allowing for learning and refinement during their implementation. Consequently, action research is often favored as one of the popular research methodologies in SE.

In RE, which serves as the foundational process of software development, conducting empirical research on requirement quality is essential [3]. The systematic mapping study conducted in [3] reveals that empirical research on requirements quality primarily employs experiments and case studies. The authors indicated that many

Chapter 2. Literature Review

empirical studies lack industrial participants, underscoring the importance of industrial relevance in these analyses.

The action research methodology emphasizes active changes initiated by researchers. Unlike other research methodologies, which may primarily focus on observation and learning or evaluation, action research places greater emphasis on intervention (similar to experiments), context (akin to case studies and observations), and iterative learning [42]. Wohlin and Rainer [43] argued that a study is regarded as action research if it shares similarities with the case study research method but specifically focuses on active participation to effect change.

An action research cycle typically encompasses five key stages: diagnosis, action planning, action taking, evaluation, and learning [42]. Several studies have adopted these stages to investigate issues identified in practical scenarios [44, 45]. For instance, Kalenda et al. [44] performed an action research study within a large enterprise to scale agile software development processes, identifying challenges, success factors, and specific scaling practices. Similarly, De Sousa et al. [45] utilized action research methodologies to examine the difficulties encountered when implementing Scrum in outsourced government projects. Similar to the “industry as a lab” approach [42], which integrates academic research with industry product performance observations, these studies offer valuable insights for both industry practitioners and the academic community.

Acknowledging the benefits accrued through the identification of authentic industry requirements, we employ action research within a collaborative venture encompassing both academia and industry. Such studies are dedicated to exploring RE techniques within the WFH environment and RE guidance for the development of ML-enabled systems.

2.1.2 Practitioner Engagement

The domain of NLP for RE is witnessing significant growth and advancement, with the objective of leveraging NLP methodologies, tools, and resources to augment the RE process. Nonetheless, only a limited number of studies [46, 47] have been conducted in industrial settings. The systematic mapping study [48] underscores the issue by revealing that a mere 7.18% of NLP-based RE studies evaluate within an industrial context, highlighting a general lack of industrial evaluation of NLP for RE research results. One of the primary causes is getting practitioners involved in research, which is widely acknowledged as a challenging task due to time and other restrictions [49].

The significant disparity between the state-of-the-art and the state of the practice in current NLP for RE research is evident. It is indicated by insufficient industrial evaluation, scant evidence of industrial acceptance of proposed tools, the absence of shared RE-specific linguistic resources, and the deficiency in NLP for RE research expertise to provide guidance on the selection of NLP technologies [48].

Agile RE, in particular, heavily relies on domain expertise, necessitating increased practitioner engagement for comprehensive evaluation. Therefore, in our proposed LLM-based techniques, we actively involve practitioners in the evaluation process, specifically for drafting the gold standard to evaluate the LLM-generated solution against human craft ones.

2.2 RE in the Real-World Context

In this section, we present an overview of RE activities within various real-world contexts derived from our action research. These contexts include RE practices in startup companies, RE in the WFH environment, especially during the COVID-19 pandemic, and RE methodologies tailored for FinTech development.

2.2.1 RE in Startup

Startup companies, often operating with limited resources, strive to deliver innovative products and software-intensive services [4]. Consequently, ensuring effective control over the quality of RE artifacts holds significant importance in the context of startup project development. A comprehensive survey conducted by [4] revealed that startups operating in uncertain and rapidly changing environments often opt for agile methodologies.

In the systematic mapping study of [50], 43 primary studies were identified and mapped to investigate software development in startup companies. The studies reported establishing an engineering process for collecting, defining, and managing requirements in a startup context is challenging. In [50], the authors conducted a systematic mapping study on software development in startup companies. They found that startups often adopt market-driven rather than customer-specific requirements as they produce software for a growing target market.

Paternoster et al. [50] found one of the most frequently reported contextual features of startups concerns the general lack of resources. Klotins et al. [51] examined 88 startup experience reports and indicated the root cause of startup failures

is the three types of anti-patterns: 1) not getting the first product release, 2) not attracting customers to the product, and 3) challenges of scaling the product for new markets. Zorzetti et al. [52] conducted a case study with two industry software development teams to characterize a development approach combination of agile software development, user-centered design, and lean startup, exposing how these approaches can be intertwined in a single development process and how they affect development. The results reveal that the combined approach pushes the team to think critically throughout development.

Rafiq et al. [53] identified a notable gap in the studies regarding the investigation of RE practices in software startups. Similarly, the systematic mapping study [54] corroborated this finding by indicating the scarcity of studies contextualized within startup environments. Berg et al. [54] emphasized the need for research to identify RE practices applicable to different lifecycle stages, which would assist startups in selecting appropriate RE techniques for their specific situational contexts.

To address the lack of real-world case studies in existing research, our action research, conducted within an industry-academia collaboration involving a startup, aims to provide valuable insights into the practical application of RE techniques in a startup context.

2.2.2 RE in WFH

Research on Global Software Development (GSD) has increased in the last few years [55, 56]. Herbsleb [57] described a globally-distributed project for large companies. This study [57] reveals dealing with unstable requirements is a big challenge, not only for end-user requirements but anything that specifies what a team should deliver.

Ebert et al. [58] summarized the experiences and trends from the international conference on GSD. They indicated collaborative development environments and tools enable effective development. The authors investigated two RE tools (i.e., Doors¹ and IRqA²) and found such document-oriented tool can help to manage use cases and requirements. They insisted effective tool support for collaboration is a strategic initiative for any company with distributed resources. In the systematic literature review of [59], the authors explored the tools used in GSD. The authors found that the issue tracking system is the most frequently provided feature for requirements tools, which allows users to be aware of important issues or changes. Jalali and Wohlin

¹ www.ibm.com/software/awdtools/doors

² www.visuresolutions.com

[60] conducted a systematic review to investigate the GSD and agile practices. From the cases they reported, there were some success factors, such as distributed business relationships. The idea of GSD has evolved into the remote work (i.e., WFH) we know today.

WFH under normal circumstances differs from WFH due to COVID-19 restrictions [61, 62, 15, 13]. The restricted COVID-19 WFH makes the engineers work under stress and with not fully provisioned home office [15]. Miller et al. [11] surveyed WFH during the COVID-19 pandemic and investigated how managers and leaders can support and sustain team productivity during times of crisis and beyond. They identified that communication, camaraderie, and team cohesion affect software development team productivity. One of the critical issues reported by the respondent is WFH, which leads to difficulties in communicating with teammates [11].

Russo et al. [10] analyzed specific working activities for WFH and evaluated how WFH influences software engineers' perceived productivity and well-being. In this study, the participants reported spending less time in meetings and less time for breaks when working remotely. They argued these were less common, possibly because of the developers' adoption of working remotely. The authors suggested WFH should not be presented as a challenge for either organizations or developers [10].

Smite et al. [63] illustrated the new trends in the demands for flexibility among company employees and modifications to work policies that are emerging in response to these demands. The authors found the demand for flexibility concerning the privilege to work from home apparent within the study. However, these studies did not discuss agile RE challenges and their consequent risks in-depth, especially in the WFH context of unprecedented global challenges.

2.2.3 RE for FinTech Development

FinTech systems are rapidly becoming a new class of software systems with associated high risks. Several papers have been published related to FinTech system development. A systematic literature review of [64] explored the challenges and trends of FinTech development. The meta-analysis results show that determining the business models and models appropriate to the culture of each country is challenging. A case study on FinTech development in China [65] revealed the success of financial innovation regarding how well finance serves business and real-life needs.

However, studies specifically investigating RE for FinTech systems are limited. [66] conducted a case study on agile software processes in FinTech companies,

providing recommendations for agile adaptation but without integrating the specific context of FinTech. In contrast, Sharifi et al. [67] proposed user requirements notation, a requirement-based approach, which is about modeling and analyzing the requirements of a FinTech system and capturing its design rationale. The result indicates such a requirement-based approach can potentially facilitate the certification process of FinTech systems.

2.3 RE in Agile Development

This section demonstrates the importance of communication in agile RE and presents an overview of tasks in agile RE activities, including requirement elicitation and requirement analysis.

2.3.1 Communication in Agile RE

Cao and Ramesh [68] conducted an empirical study on agile RE practice. Data from 16 organizations were collected to compare the differences between agile RE and traditional RE. The participants reported that intensive communication is the most vital RE practice between developers and customers [68]. Schön et al. [69] argued that software development had moved away from plan-driven to value-driven process models and believed human values mainly drive product development with agile methodologies.

From the systematic literature review conducted by [69], the authors identified that agile software development integrated with four human-related methods (i.e., human-centered design, design thinking, contextual inquiry, and participatory design) did increase knowledge about user needs in development. Through experience reports and case studies extracted from 52 articles in the systematic literature review of industrial large-scale agile transformation, Dikert et al. [70] found 38% of teams met four main RE challenges: high-level requirements management largely missing in agile, requirement refinement challenging, creating and estimating user stories hard, and the gap between long and short term planning. One of the key success factors of requirement management in agile transformation is recognizing the importance of the product owner role. A well-implemented product owner role understands the use of agile methods to collaborate with businesses and technology to improve work productivity. Another critical success factor is investing in learning to refine the requirements. They indicated a dedicated “requirement architect” can help to bridge

the gap between development and requirement management. These studies show that human-centric communication plays one of the most critical roles in agile RE.

2.3.2 Elicitation in Agile RE

User Story and Acceptance Criteria

The widespread adoption of agile software development methods increases the usage of user stories as requirements elicitation methods in industry [71]. User stories and acceptance criteria are two key concepts in agile requirements specification [72], where the former is a brief description of some aspects of a software system [29]. The latter encompass functional behavior, business rules, and quality aspects to be tested and are attached to a user story [72].

The popular and widely adopted Connextra template [71–73] of user story is: “As a ⟨type of user⟩, I want to ⟨goal⟩, [so that ⟨reason⟩].” In this template, the user story captures the essential elements of a requirement: ⟨type of user⟩ defines *who* it is for, ⟨goal⟩ denotes *what* is expected from the system, and optionally, ⟨reason⟩ expresses *why* it is important [74].

Within the project environment, user stories exhibit a spectrum of granularity levels, which contains epic, story, feature, and theme [75, 76, 72]. [76] introduced a macro-level structure of user story. Initially, epics serve as user stories with a high level of abstraction in the early stages of agile project development, requiring subsequent refinement into smaller stories for more precise requirement articulation. Subsequently, when introducing features into the system, smaller stories can be derived from these epics [75]. At the story level, the user story should contain a feature or part of the features that express the delimited characteristics of a system. Stories sharing common features may be consolidated into the same theme, representing a collection of lower-level user stories.

At the story level, a user story is usually attached to acceptance criteria constructed for the development process [76, 72]. Acceptance criteria define the functionalities required by a user story [34], containing preconditions, triggering events, and post conditions [77]. According to the interviews conducted in [78], the two most popular formats of acceptance criteria are scenario-oriented (Given/When/Then template) and rule-oriented (checklist template)³. This thesis adopts the checklist template because some cases are not applicable to scenario-oriented structures. For instance, some user

³ <https://www.altexsoft.com/blog/acceptance-criteria-purposes-formats-and-best-practices/>

Chapter 2. Literature Review

stories describe system-level functionality that acceptance criteria are hard to represent as the Given/When/Then template.

In this thesis, we discuss the user story at the story level, which can be elaborated with the acceptance criteria in agile project development. Figure 2.1 illustrates an example of US-AC. Following the Connextra template, the user story defines the user type as a standard user. Then, it specifies the goal, which is to receive an email notification so that the user can reply to the comments through email. The user story provides a brief introduction to the functionality that needs to be developed. In the sprint planning phase, the user story is further elaborated with detailed acceptance criteria. For example, it extends the elaboration of the user story to specify how the email should be notified and what contents should be included in the email. These details, such as email content, define what should be included to achieve the goal of receiving an email notification. Generated acceptance criteria serve as a valuable reference for delivering the working solution. In accordance with the guidelines delineated in the US-AC, developers can determine an appropriate data format for email content, while testers can formulate a comprehensive series of test cases designed to validate the functionalities.

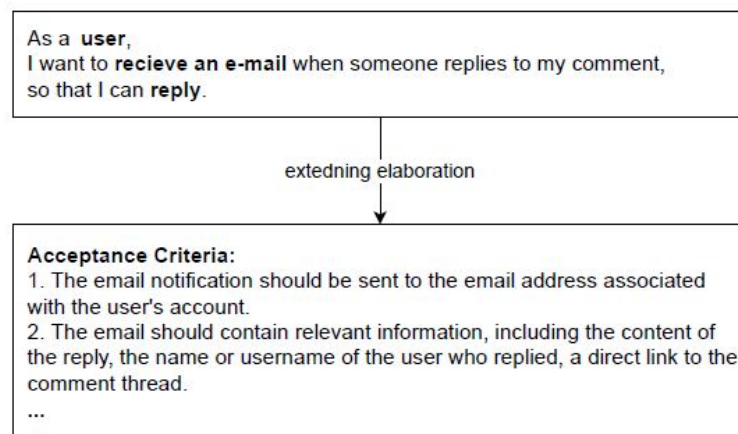


Fig. 2.1 An example of elaborating user story associated with acceptance criteria.

Collaboration in GAC

The most adopted practice in agile development is the hybrid method of Scrum and eXtreme Programming (XP) [31]. Sverrisdottir et al. [31] presented the key activities in

2.3 RE in Agile Development

Scrum practice. Building upon these established practices, we elaborate on the specific procedures involved in GAC from the sprint planning phase onwards, as depicted in Figure 2.2.

Within the lifecycle of agile project development, the delineation of the product backlog, which encompasses user stories, defects, tasks, and other work elements, is an endeavor undertaken collaboratively by the requirement analyst and stakeholders [29]. In the recurrent sprint planning sessions, the requirement analyst selects the user story from the product backlog and integrates it into the sprint backlog after agile team discussion.

Subsequently, GAC requires collaboration from the agile team members. The workflow of GAC is illustrated in the dashed box of Figure 2.2, where the requirement analyst is responsible for creating an initial version of acceptance criteria [31, 79], and then other agile team members update or revise the previously generated acceptance criteria if necessary [29]. Finally, the refined US-AC is delivered to the team as the reference for subsequent sprint development activities.

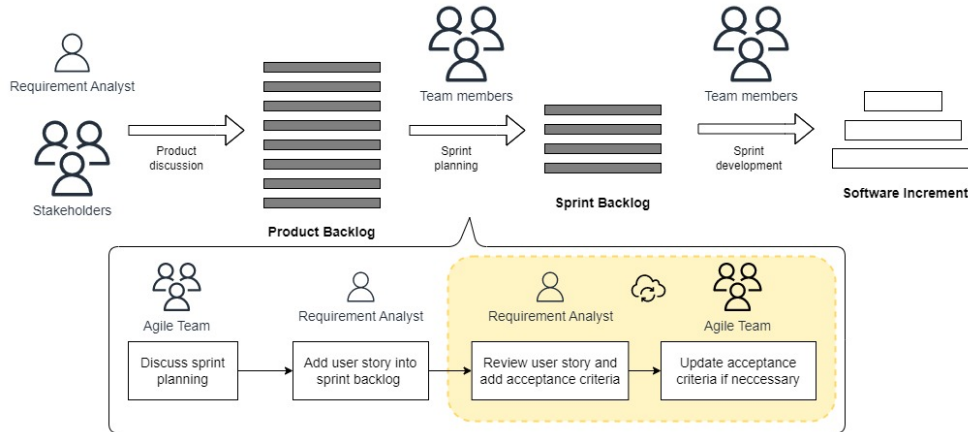


Fig. 2.2 Agile practice in sprint planning: collaborative acceptance criteria generation from agile team members.

2.3.3 Analysis in Agile RE

Conceptual Models

During requirement analysis, deriving conceptual models from user requirements is a crucial process that underpins subsequent stages of the development lifecycle. Conceptual models play a vital role in refining high-level requirements into more detailed specifications by delineating the primary entities and relationships inherent in

Chapter 2. Literature Review

these requirements [80–82]. According to [83], conceptual models are predominantly graphical representations that capture both static and dynamic phenomena within a domain. Furthermore, Mylopoulos [84] claimed that conceptual models provide formal descriptions of various aspects of the physical and social world to facilitate understanding and communication. This thesis focuses on graphical conceptual models that depict a domain in structural terms, encompassing entities, attributes, and relationships, to enhance stakeholders’ comprehension of the domain in RE.

Derivation of Class Diagram

Several studies [80, 85–87] have explored the derivation of conceptual models from use cases. However, there is a scarcity of methods for deriving conceptual models from user stories [36], where most of the work relies on human effort [48, 36]. For instance, Berry et al. [88] introduced four types of NLP tools for handling requirements, one of which pertains to the extraction of conceptual models from natural language description requirements. Lucassen et al. [89] proposed an automated approach, leveraging the Visual Narrator tool, for extracting structural conceptual models, such as class diagrams, from a set of user stories. This method incorporates NLP heuristics from existing literature, achieving commendable precision and recall. However, its implementation poses challenges due to the extensive array of linguistic patterns inherent in natural language. Based on the previous work, Bragilovski et al. [36] designed example-based guidelines to derive conceptual models (i.e., use case diagrams and class diagrams) from user stories. The results indicate that these guidelines enhance the completeness and validity of conceptual models in cases of medium complexity, although subjects expressed neutrality regarding the usefulness of the guidelines. Notably, these approaches rely on human effort. The emergency of LLMs brings a chance to transition from manual efforts to an automated workflow through well-designed prompt input.

2.4 RE and AI

This section provides an overview of the literature concerning RE activities for AI product development and AI techniques developed for RE tasks.

2.4.1 RE for AI

Requirement Activities

In the survey conducted in [22], the authors reported RE was the most difficult activity for developing ML systems, which involves requirements analysis and specification in the initial phase and an acceptance inspection in the final phase. They emphasized that prior estimation and assurance of achievable accuracy are particularly hard to manage. Additionally, defining Non-Functional Requirements (NFRs) for ML systems is a big challenge because there is no unified collection or consideration of many NFRs for ML [90]. Similarly, in the systematic literature review of [91], the authors highlighted the need for new methods and solutions that address NFRs.

Data Requirement

Requirements must ensure the appropriate quality of data, determine data availability, establish methods for testing, and select suitable data for training purposes [92]. Shin et al. [93] identified that one of the primary issues with data requirements is the high cost associated with data generation. Additionally, significant challenges include data availability [94] and data quality [95]. According to Ries et al. [96], Deep Learning (DL) models heavily depend on datasets, an aspect often overlooked in RE. The study aims to identify the necessary requirements for dataset structures, including the datatypes, attributes, and properties that need to be elicited during the RE process.

In the systematic study on RE for AI [92], the authors suggested that practitioners should maintain a checkpoint to note all required elements to create an AI software system. Additionally, this study highlighted that the predominant challenges discussed in RE for AI research pertain to issues surrounding data requirements and the deliberation of trade-offs during AI software development.

Ethical Requirement

Ahmad et al. [92] observed a notable scarcity of empirical studies addressing the ethical and trust-related aspects of RE for AI. Similarly, Hall et al. [97] highlighted that explainability requirements could conflict with risk factors inherent in AI systems, particularly in safety-critical applications. Additionally, Bruno et al. [98] noted that explainability requirements might clash with other requirements such as security, cost, and precision. These findings underscore the importance of integrating ethical considerations into RE activities for AI. To address these conflicts, Schwammberger

[99] proposed using the Softgoal Independency Graph (SIG) to model explainability alongside other NFRs, aiming to minimize potential conflicts.

Direction of Research

Pei et al. [100] performed a literature review on RE for ML, and step-by-step went through the process of collaborative requirements analysis. From their case study on industrial data-driven intelligence applications, they observed three topics that require further study: designing adaptable requirements models to dynamic changing scenarios and connecting with sustainable active ML pipelines; producing a reliable overall cost estimation of a given project; and implementing simulation-based prototyping for the development of ML applications [100]. Furthermore, Ahmad et al. [92] examined 43 studies, none of which referenced agile settings.

2.4.2 AI for RE

Over the years, many AI techniques have been employed to represent and analyze requirements, ranging from knowledge, representation, and NLP [101].

NLP Techniques for RE

Many RE activities (e.g., requirements elicitation and specifications) require text analysis and human analyst engagement and have benefited from the NLP techniques in the past decades. According to the systematic mapping study conducted in [48], the top-ranked NLP tools developed for RE are those designed for requirement modeling [89] and detection [102], followed by those aimed at extraction [103] and classifications [104]. Additionally, [48] reports the commonly used NLP techniques in RE research are based on lexical or syntactic analysis, such as Parts Of Speech (POS) tagging [105] and tokenization [106]. These NLP-based tools contribute a lot to RE automation.

For example, in requirements elicitation activities, NLP-based tools are developed to reduce the human effort needed to understand the project requirements. Xiao et al. [107] proposed a general framework incorporating syntactic and semantic analysis to extract model instances of natural language software documents and produce formal specifications. By leveraging universal sentence encoder [108] and BERT [109], Nema et al. [110] proposed classifying the privacy information and employing summarized topic clusters by extracting representative reviews from the cluster. In the requirements

modeling activities, the development team will derive the domain model for the requirements, where the NLP-based tools can assist the human analyst. For instance, Thakur and Gupta [111] developed AnModeler, a tool for generating analysis models from software requirements specified as use cases. It extracts the Type Dependencies (TD) and POS tags of sentences and then identifies sentence structures using a set of framed rules. Finally, the TDs, POS-tags, and sentence structures are utilized to generate the class diagrams. Several NLP-based tools have been developed to assist with requirements management and classify functional requirements (FRs) and non-functional requirements (NFRs). Hey et al. [112] proposed NoRBERT, which fine-tunes BERT on the task of FRs and NFRs classification.

Moreover, Jin [113] presented how the environment can be elaborated with the smart space, and her finding suggests quick-witted intelligence when responding to its inhabitants. [114] investigated how to develop ML and DL techniques for RE tasks, such as applying ML on RE prioritization [115] and using word embeddings to identify domain-specific ambiguities in requirements natural languages [116]. It has gained increasing interest in applying AI, especially ML models, to RE tasks to help improve RE quality during the software development process.

LLMs for RE

In light of the impressive capabilities demonstrated by recently released LLMs in NLP-related tasks, RE researchers have started exploring the breakthrough of integrating with LLMs. Zhang et al. [37] conducted a preliminary evaluation of ChatGPT utilized in two types of requirements information retrieval tasks: requirements classification and information extraction. The results highlight the impressive ability of ChatGPT to retrieve requirements information from different types of artifacts involving multiple languages under a zero-shot setting. They argued it is worthwhile for the research and industry communities to study generative LLMs for RE tasks and develop corresponding tools. Ferrari et al. [117] examined the efficacy of ChatGPT in generating UML diagrams and sequence diagrams based on natural language requirements. Their findings suggest that augmenting ChatGPT with additional contextual cues and domain-specific knowledge holds promise for enhancing its capacity in model generation. Spoletini and Ferrari [118] proposed a vision to employ fine-tuned LLMs to support the formalization and explanation of requirements. The study outlines a roadmap towards realizing this vision, which includes leveraging LLMs to automatically produce formalized properties, using LLMs

to explain and generalize counterexamples to rectify the resulting code, and employing LLMs to generate explanations and recommendations.

2.5 LLMs and Prompts

This section further offers a review of LLMs and the prompting techniques.

2.5.1 Overview of LLMs

LLMs have showcased their remarkable capabilities in a variety of tasks, including machine translation [119], summarization [120], and question-answering [121]. LLMs undergo unsupervised training using billions of open-source text tokens to achieve comprehensive language modeling. The utilization of diverse data sources during LLM training enriches their cross-domain knowledge, thereby facilitating effective generalization for corresponding downstream tasks [122].

There are three categories of LLMs: encoder-only model, encoder-decoder model, and decoder-only model [123]. Encoder-only LLMs accept an input sequence and transform it into a vectorized representation, which are text understanding models without text generation capability [124]. These encoder models, exemplified by the widely used BERT [109], can be fine-tuned for various downstream tasks like sentiment analysis [125] and named entity recognition [126]. In contrast, encoder-decoder LLMs incorporate a decoder network, which generates an output sequence through iterative produced tokens or symbols based on the context vector and previously generated tokens. Encoder-decoder models, such as BART [127] and T5 [128], are particularly good at tasks necessitating intricate mappings between the input and output sequence, particularly where capturing the relationships between the elements in both sequences is crucial. Some common use cases for encoder-decoder models include machine translation [129, 130] and summarization [131, 132]. Decoder-only models typically adopt architectures such as autoregressive models, wherein the output is generated token-by-token, with each token conditioned on preceding tokens and context. Unlike encoder-only and encoder-decoder LLMs, decoder-only LLMs do not have an encoder component to process the input data but only a decoder component that directly generates an output sequence based on a given context or input. The autoregressive nature inherent in decoder-only models ensures the token-by-token generation of output, utilizing previously generated tokens as context for subsequent token generation [123]. Decoder-only models have proven to

be highly effective in various NLP tasks that can be framed as autoregressive problems, such as text generation [133].

Recent studies [134–136] mostly focus on training LLMs with decoder-only architecture because of its effectiveness and flexibility [137]. The decoder-only LLMs demonstrate superior zero-shot and few-shot generalization capacities compared to other architectures [137, 138]. Besides, the parameter sizes of encoder-decoder models are relatively small, such as CodeT5 [139] and UniXcoder [140] less than 1B, resulting in a deficiency in emergent abilities [137], rendering them unsuitable for tasks without fine-tuning.

Given the inherent nature of GAC and the prevalent utilization of decoder-only LLMs in practice, this thesis exclusively focuses on decoder-only LLMs. Among the decoder-only LLMs, the GPT (Generative Pre-trained Transformer) families and LLaMA series stand out for their popularity and prowess due to the large corpora on which they are trained. For example, GPT-3 is trained on 45TB of text data and has 175 billion parameters. LLaMA is trained on 1.4 trillion tokens with a variety of model sizes ranging from 7 billion to 65 billion parameters [123].

While the content generated by LLMs is often grammatically correct, they may not always be semantically correct [141]. Despite their robust performance, LLMs sometimes struggle to produce accurate answers when faced with complex tasks. In response, researchers have proposed various prompting techniques to enhance the reasoning capability of LLMs in NLP tasks [122].

2.5.2 Prompt Engineering

Prompts are instructions given to an LLM to enforce rules, automate processes, and ensure specific qualities of generated outputs [142]. Recent studies have shown that well-designed prompting methods can significantly improve the performance of LLMs.

Collaborative Prompting

Recent research indicates an increasing interest in utilizing multi-agent collaboration techniques to unleash the power of LLM further. For instance, VirsualGPT [143] and HuggingGPT [144] explored the collaboration of LLMs with other models, specifically the LLMs are used as decision centers to control and invoke other models to handle vision, speech, and signals. Dong et al. [145] developed a self-collaboration prompt framework to guide ChatGPT to understand collaboration in the task of code generation. These studies primarily employ qualitative analysis of case studies to

demonstrate the efficiency of the proposed framework. In [145], the authors also conducted a quantitative analysis to evaluate the self-collaboration code generation. Our work aligns with the concept of collaboration of LLMs, but it is uniquely positioned with the field of agile RE, which provides practical guidance to automation on GAC in agile RE.

Chain-of-Thought Prompting

The CoT prompting technique, as introduced in [40], is a relatively recent advancement in prompting methods. CoT prompting involves instructing LLMs to generate intermediate natural language reasoning steps, or a CoT, before producing the final output. This technique has demonstrated state-of-the-art performance in natural language generation tasks and has prompted further research in related areas, such as self-consistency prompting [146] and least-to-most prompting [147]. It is important to note that these prompting techniques, including CoT prompting, were primarily developed and evaluated in the context of natural language generation tasks. They have not been specifically tailored or extensively explored for the task of deriving class diagrams from user stories.

While CoT prompting has shown promise in natural language generation, our study investigates its effectiveness and applicability in the domain of agile RE and the derivation of class diagrams. By focusing on this specific task, we contribute to the understanding of how prompting techniques, specifically CoT prompting, can be adapted and utilized in different problem domains, providing insights and advancements specific to the task of deriving class diagrams from user stories.

2.6 Summary

In this chapter, we first introduce the empirical methodologies employed in this thesis, including action research and practitioner engagement. After that, we review RE within different real-world contexts derived from our action research, including RE in startup, RE in GSD, RE during COVID-19, and RE for FinTech development. Subsequently, we discuss the background of RE activities in agile development as related to our action research and simulations with LLMs. We then delve into the intersection of RE and AI. Finally, we provide a detailed introduction to LLM techniques and prompt engineering. This chapter serves as a foundational overview for the thesis.

Chapter 3

Agile RE in WFH Environment

3.1 Introduction

The emergence of unprecedented global challenges, exemplified by the COVID-19 pandemic, mandated a widespread transition to WFH arrangements for project teams. This shift posed significant challenges in conveying requirements within the framework of agile RE. While numerous studies have scrutinized the impact of transitioning work routines during the pandemic, there remains a dearth of research focused on the specific challenges encountered by agile RE within the WFH context. Given the pervasive shift in the worldwide software development ecosystem, where WFH is projected to persist even in the post-COVID era, it is imperative to elucidate the challenges associated with WFH-based agile RE practices.

In response to the exigencies of the pandemic, we engaged in a collaborative industry-academia project with one startup. Utilizing action research methodology, this study conducted a comprehensive analysis of agile RE practices, delineating the key challenges encountered within the WFH environment. To address these challenges, several collaborative RE techniques were implemented across three intervention cycles. Subsequent interviews were conducted to scrutinize the outcomes meticulously.

The primary objective of this study is to gain a comprehensive understanding of the challenges faced by agile RE teams in startups operating within the WFH context, particularly when sudden constraints arise. To facilitate a deeper comprehension and offer practical recommendations for agile RE in various WFH work environments, it is imperative to examine real-world cases within the WFH conditions. The challenges faced by requirements engineers in Chinese startup software development companies

Chapter 3. Agile RE in WFH Environment

were particularly pronounced due to the implementation of strict lockdown policies in China. We hypothesize that these challenges are highly susceptible to disruption caused by the abrupt transition to WFH. Drawing from these observations, our aim is to explore the ways in which collaborative RE techniques can effectively alleviate these challenges. To guide this study, three Research Questions (RQs) are formulated as follows:

- RQ1. What are the agile RE challenges a startup team faces under the WFH context of unprecedented global challenges?
- RQ2. What collaborative RE techniques can help alleviate the observed challenges under the WFH context?
- RQ3. To what extent can these RE techniques help to alleviate observed challenges?

We report the experience of a startup named ORG¹, which embraced agile as their software development approach and engaged in an industry-academia collaborative project throughout the COVID-19 pandemic. To address RQ1, we employed action research interventions and followed the agile RE challenges observation framework [1] integrating with the WFH context imposed by the pandemic, which aims to explore the RE challenges and risks in each intervention cycle. In the context of restricted WFH conditions, we introduced several collaborative RE techniques to address RQ2. Specifically, we investigated various collaborative RE techniques across multiple RE activities, including requirement elicitation, modeling, transfer, prioritizing, and demonstration. Through interviews, feedback collection from different stakeholders, and analysis of relevant documents, we observed the impact of the applied techniques outlined in RQ3.

This study [148] was accepted by the 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC 2024). The remainder of this chapter is structured as follows: Section 3.2 details our action research approach. Section 3.3 outlines the challenges identified and addressed throughout three intervention cycles, while Section 3.4 presents the key lessons learned from the study. In Section 3.5, we examine potential threats to validity, focusing on transferability and reflexivity in the evaluation of action research. Finally, Section 3.6 concludes our study.

¹ We named the company as ORG due to their wish to remain anonymous.

3.2 Methods

Following the definition in [149, 43], we choose an AI-advisor platform development project (a cooperation project hosted by ORG and City University of Hong Kong) as our case for studying the agile RE challenges faced by startups during COVID-19 pandemic. ORG was a suitable partner in this study since the participants were mainly located in three regions of China's Greater Bay Area (i.e., Hong Kong, Macao, and Mainland China) under stringent lockdown control. One of our researchers participated in the project to make changes to RE activities. Wohlin and Rainer [43] argued that a study is regarded as action research if the study shares similarities with the case study research method while focusing on active participation for change. Therefore, our study is labeled as action research, and this chapter reports the lessons learned.

3.2.1 Study Context

The industrial company ORG participated in an academia-industry research project for the development of an AI-advisor software. The project's main deliverable was to build an investment advisory platform integrated with an artificial intelligence engine. Agile was adopted as the mainstream development approach in this cooperative project. Aligning with the productive structure of the agile team discussed in [150], the project team had 8 members involved as presented in Table 3.1. All participants have at least two years of relative working experience. Due to the limited resources of the startup company, no particular testers were engaged. The role of a tester was performed by the Project Manager (PM), Business Analyst (BA), and stakeholders. The team members were mainly located in three regions in China's Greater Bay Area (i.e., Hong Kong, Macao, and Mainland China). Because of the strict border control and lockdown policy, the team members used the WFH shift during the project period. The last column in Table 3.1 summarized each team member's WFH duration.

3.2.2 WFH and Face-to-Face Agile Practices

According to the Agile Manifesto [151], both WFH and face-to-face environments adhere to core Agile principles, including iterative development, customer collaboration, and responsiveness to change. However, the dynamics of collaboration differ significantly between these two settings. Specifically, Mancl and Fraser [152] highlighted the challenges faced by teams during the sudden shift to WFH necessitated by the COVID-19 pandemic, particularly regarding the integration of

Table 3.1 Team structure and background of developers for ORG

Role	Qt.	Background	Location	WFH Duration
Stakeholders	2	Over 10-year relationship manager experience on global bank	Hong Kong, Macao	Project Period
Project Manager (PM)	1	Over 10-year experience of financial IT project management	Shenzhen	Project Period
Business Analyst (BA)	1	3-year business requirement analysis experience on global bank	Hong Kong	5 months
Data Scientist (DS)	1	2-year experience of data analysis on bank	Hong Kong	4 months
Developer	2	3 to 5 years' development experience on IT products	Hong Kong	4 to 7 months
UI Designer	1	Freelancer with 3-year UI design experience	Tianjin	Project Period
Tester	0	Holding concurrent post by PM, BA and stakeholders	Mixed	Mixed

Project Period: 15 months, lasting from September 2020 to November 2021

virtual collaboration practices. The study emphasized the need to learn from these experiences to optimize remote Agile practices. Additionally, Nolan et al. [15] provided a comparative analysis of the similarities and differences between WFH and face-to-face Agile practices. Building on these insights and integrating our project experiences, we identified several key similarities and differences as presented in Table 3.2, which serve as primary activities in action research for observation.

3.2.3 Agile RE Evaluation Framework

In [1], the authors conducted an empirical study into agile RE practices. They applied grounded theory to identify six agile RE practices, which are listed and described as follows:

- **Face-to-face Communication (P1):** In Agile RE, face-to-face communication is preferred over written specifications.
- **Iterative RE (P2):** Requirements emerge during the development process instead of being pre-defined.
- **Constant Planning (P3):** In agile software development, implement the features with the highest priority.

Table 3.2 Key similarities and differences in WFH and Face-to-Face environments

		WFH	Face-to-Face
Similarities	Collaboration	Both methods emphasize teamwork and communication, whether it's through digital tools or in-person meetings.	
	Daily Stand-up	Daily stand-ups or scrums are a key part of Agile in both settings. Teams discuss progress, obstacles, and plans for the day.	
	Sprint	Sprint planning and retrospectives remain crucial components, focusing on planning the work to be done and reflecting on what was accomplished.	
Differences	Collaboration Dynamics	Requires more structured communication, with scheduled meetings and reliance on collaborative tools for documentation and knowledge sharing.	Collaboration is often more dynamic, with easier access to team members, leading to more fluid idea exchanges.
	Tools and Technology	Heavily dependent on technology for all aspects of communication, collaboration, and project management.	Primarily relies on in-person meetings, whiteboards, and direct interaction, though digital tools are still used.
	Problem Resolution	May require more formalized processes to resolve issues due to the lack of physical proximity.	Problems can often be addressed immediately with quick discussions or impromptu meetings.

- **Extreme Prioritization (P4):** Adapt and react quickly to changes the environment demands.
- **Prototyping (P5):** Use prototypes developed for communication and demonstration.
- **Reviews and Tests (P6):** Use frequent review meetings for requirements validation.

Ramesh et al. [1] proposed a framework for evaluating the impact and appropriateness of agile RE practices by relating them to RE risks. Additionally, Ramesh et al. [1] consolidated nine distinct risks (with minimal overlap) and mapped them with agile practices and challenges. In the systematic literature review of [153], the authors summarized the four most common RE risks in GSD as follows: 1) knowledge sharing; 2) client and vendor relationship; 3) problems with process definition; 4) communication problems.

In our study, we followed the same framework proposed by [1] but integrated with and adapted to a WFH context to explore the challenges and risks exposed by

the six agile RE practices. Based on observation and interview results collected, we mapped the challenges and most recurrent RE risks proposed in [153] by analyzing the relationships between practices, challenges, and risks to answer RQ1. The overall mapping is presented in Figure 3.1. In Section 3.3, we summarize and discuss each challenge with detail and analyze its consequent risks for different action research intervention cycles.

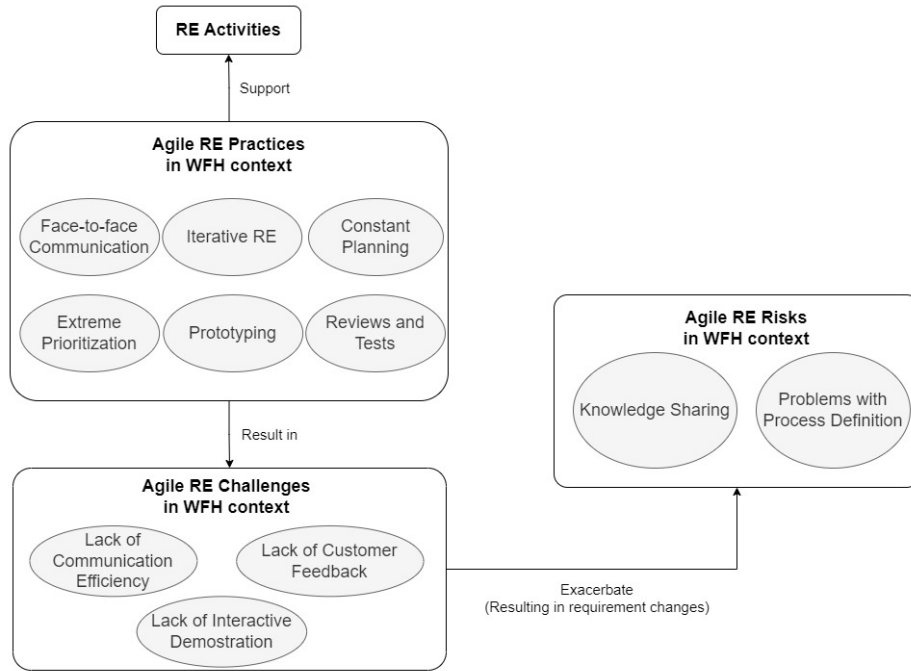


Fig. 3.1 Evaluating challenges and risks exposed from RE activities. Adapted from [1] and integrated observations under WFH context.

3.2.4 Action Research

Research Activities

Kalenda et al. [44] adopted a 5-step process to carry out action research, which includes diagnosis, action planning, action taking, evaluation, and specifying learning. Staron [42] recommended to follow a 5-step cycle to guide the action research in SE. De Sousa et al. [45] conducted an action research following the major steps of situation diagnosis, action planning, and action operation. Following the action research process presented in [45, 44, 42], we adopted the 5-step action research cycle in each development phase,

conducted the evaluation, and specified the learning steps concurrently in the next phase or at the end of the project period.

To observe the challenges and provide improvement on the WFH-based RE practices, the researchers made interventions that are tailored to the local practice. The project schedule was divided into three phases of development. Following the project schedule and agile sprint practice, our research activities were split into three problem-solving cycles, aligning with the project development phase. Table 3.3 presents the detailed intervention period. The evaluation and reflection steps of Phase I intersect with the diagnosis and planning step in Phase II. The setup ensures the previous changes are further improved in the next iterative changes.

Table 3.3 Intervention cycles in project development

Intervention	Step	Period
Cycle I	Diagnosis (PP I)	09/2020 - 10/2020
	Planning (PP I)	09/2020 - 10/2020
	Changes (PP I)	11/2020 - 02/2021
	Evaluation and Reflection (PP II)	01/2021 - 03/2021
Cycle II	Diagnosis (PP II)	01/2021 - 03/2021
	Planning (PP II)	01/2021 - 03/2021
	Changes (PP II)	04/2021 - 07/2021
	Evaluation and Reflection (PP III)	06/2021 - 08/2021
Cycle III	Diagnosis (PP III)	06/2021 - 08/2021
	Planning (PP III)	06/2021 - 08/2021
	Changes (PP III)	10/2021 - 11/2021
	Evaluation and Reflection (PP III)	12/2021 - 06/2022

PP - Project Phase

Industrial-Academia Collaboration

To gain an in-depth understanding of the problem, authors in action research conducted participant observations [154]. Following the practice of [155], one of our authors worked as the BA in the development team for the whole project period. Following [42], SE practices (i.e., organization and product performance improvement) and SE theories (i.e., concepts of “industry as a lab”) are taken as input for the diagnosing steps. The development activities were conducted by the BA and the development team. The activities of collecting evidence, such as observations, were conducted by the rest of the research team. Working in a cross-functional role, the BA took part in the research activities for solution discussion.

3.2.5 Data Collection

[156] presents multiple data collection techniques in a case study research. In our study, the main techniques applied are *observation*, *reflection*, *interview*, and *relevant documents*. The practical experiences we report in this article are based on the data collected to answer the three formulated RQs. Below, the different data collection methods used for our study are presented.

Observation

In the diagnosis period, the observation was planned, where the research team participated in the sprint discussion and review meetings and conducted informal interviews with the development team. The BA took responsibility for requirement analysis in the project development. The rest of the researchers participated in observing the requirements-related activities, such as requirement elicitation, daily meetings for requirement delivery, and product demonstrations. These activities were held in sprint meetings.

Reflection

Apart from the planned observation, we interviewed the other team members to reflect on requirements-related challenges reported by the BA. This was to mitigate the bias raised by a single participant. Such reflections are presented combined with the analysis to answer RQ1.

Interview

Other essential feedback and information were mainly collected through unstructured interviews and semi-structured interviews with different parties, i.e., stakeholders, developers, PMs, and potential customers. The unstructured interviews were held during the sprint review meetings of reflection. No prearranged questions were prepared for the unstructured interviews, which allowed for in-depth discussion of a topic to capture more informative feedback. The semi-structured interviews employed a set of feedback questions as presented in Table 3.4, allowing the order of questions to vary to fit the conversation and interviewee's roles. When evaluating the adopted RE techniques to answer RQ3, we analyzed the feedback collected from the respondents. The main questions used for the interviews are presented in Table 3.4. IQs 1, 2, and 8 were delivered to the interviewers iteratively in each intervention cycle, which is

supportive evidence of analysis from the research team to answer RQ1. IQs 3 to 7 were delivered to the interviewers during the evaluation and reflection period, which targets to answer RQ3.

Table 3.4 Sample questions of semi-structured interview

Q#	Interview Questions	Interviewer(s)	Period
IQ1	What type of requirements changes are mainly raised in this project?	stakeholders, PM	Cycle I, II, III
IQ2	What are the causes for requirements changes?	stakeholders, PM, developers	Cycle I, II, III
IQ3	How do you think constant collaboration helps with understanding requirements?	stakeholders, PM, developers	Cycle I
IQ4	How do you initially capture the requirements for the platform?	developers	Cycle II
IQ5	How do design thinking activities help with your understanding of requirements?	stakeholders, developers	Cycle II
IQ6	How does mixed language help with your understanding of requirements?	developers	Cycle II
IQ7	How do you think the roster customer testing work in your project?	stakeholders, PM, customers	Cycle III
IQ8	To what extent does the product meet your expectations?	stakeholders, customers	Cycle I, II, III

Relevant Documents

With permission from ORG, we collected all requirement documents, project performance reports, architecture design specifications, and others. Our analysis and discussion in Sections 3.3 and 3.4 are based on these documents. During the 3 phase development period, 25 major requirement change requests were raised during development. In the acceptance test for each sprint, 120 defects were identified and reported in total.

3.2.6 Data Analysis

Data analysis was conducted parallel to the data collection approach of [157]. In each of the intervention cycles, we conducted a diagnosis based on our observations and reflections. Montgomery et al. [3] conducted a systematic mapping study of empirical research on requirements quality. From the 105 relevant primary studies, the authors found that empirical requirements quality research has primarily focused on “requirements” with little attention given to other artifacts such as use cases, user stories, or issues in issue trackers. They suggest that future research can consider

insights as this field grows. Following the suggestion, requirements changes, user stories, and defect lists are considered for data analysis in the diagnosis step in each intervention cycle.

Based on the diagnosis, the research team developed some RE techniques to answer RQ2. In the evaluation and reflection phases in each intervention cycle, we compared the defect lists linked to each technique and interviewed the project teams to collect further feedback to answer RQ3. In the discussion in Section 3.4, the lessons learned are analyzed using defect tracking and respondent feedback.

3.3 Intervention

In this section, we report the challenges observed from each intervention cycle, which answers RQ1. Then, we discuss the planning and changes in the cycle to answer RQ2, respectively. Table 3.5 presents the challenges and risks diagnosed from each intervention cycle.

Table 3.5 Challenges, RE practices, causes of requirements changes and risks revealed in WFH-based agile RE practices

Interventions	Challenges	Practices	Causes	RE Risks
Cycle I	C1	P1, P3, P6	understanding gap	RS1: knowledge sharing
Cycle II	C2	P5, P6		RS2: problems with process definition
Cycle III	C3	P2, P3, P6	customer inability; rapid market change	

3.3.1 Cycle I

Diagnosis

Lack of communication efficiency (C1). Face-to-face communication is one of the key agile manifesto principles because such communication is the most effective and efficient method to convey information within a development team [158]. Another important agile manifesto principle is “Business people and developers must work together daily throughout the project” [158]. Therefore, in intervention Cycle I, the research team mainly observes agile practice P1.

This project was conducted during the pandemic, with team members in three regions under locked-down restrictions. The project team could only communicate

online during the WFH period, precluding any possibility of face-to-face interactions. With agile practice, the team was to have regular meetings for each sprint. For requirement clarification, it wasn't easy to explain with details without face-to-face communication. The BA pointed out the lack of efficiency when communicating with WFH team members.

***BA:** During the WFH period, we had to adopt pure online communication. Compared with practicing working in the office, I must spend more time on online discussions to convey the requirements. I found it was less efficient to communicate with stakeholders because they were WFH for the whole project period. For clarification on each requirement, we spent more time discussing whether it was expected. Instead, our communication efficiency improved when I worked with other developers in the same working location.*

Intersected in agile practices P3 and P6, the research team conducted informal interviews with the development team to observe their feelings on the WFH working procedure. During the sprint reflection interview, the developers identified significant gaps in understanding when relying solely on online communication methods. Both front-end and back-end developers reported that the effectiveness of integration development was diminished under these conditions, as it necessitated the coordination of multiple engaged parties, which proved challenging in a purely virtual environment.

For answering IQ1 and IQ2 in Cycle I, the respondents reported the requirement changes mainly caused by the understanding gap. Evidently, communication remains a significant challenge in agile RE during the pandemic.

The challenges C1 reported above reveal that the understanding gap was enlarged under a strict WFH context with a lack of communication efficiency during the pandemic. The limited understanding of the requirements makes obtaining a shared understanding in the team difficult. Furthermore, observations indicate that C1 has heightened the frequency of adjustments to requirement planning, thereby posing a substantial risk to requirements management.

Planning

Constant communication. Compared to traditional development methods, agile methods rely on extensive collaboration, i.e., face-to-face communication [12]. Inayat et al. [12] discussed how agile RE could resolve five challenges compared to traditional

Chapter 3. Agile RE in WFH Environment

RE in the systematic review. Agile methods depend on tacit knowledge and frequent face-to-face communication to support knowledge sharing [12]. To emphasize the importance of face-to-face communication in agile RE and concerning challenge C1, we propose to define some rules on WFH communication.

Changes

Quick review meetings. Since the project faced instant communication and demonstration difficulties during the pandemic, more regular online requirement discussion meetings were held between BA and stakeholders. At the beginning of the day, the BA held half an hour meeting for requirements delivery to ensure developers clarified the detailed requirements. At the end of the day, another quick review meeting lasting for 15 minutes allows BA to review the daily progress and verify whether the development fits the requirements.

Regular response for instant group messages. During WFH, the team could not discuss the issues face-to-face, and the communication was shifted to online discussion. The team members were located in different areas, as presented in Table 3.1, and various daily chat tools were used in these areas. The PM requested team members to report queries to the group chat for instant discussion. The BA regularly checked the group message in working time so that reported issues could be discussed in time.

Ad-hoc discussions. Apart from these periodic updates, the ad-hoc discussion was used to assist with communication and to mitigate the communication gap and delay. If the team found any issues requiring multi-members for discussions, the ad-hoc meeting would be held for a quick and short discussion engaging all related members.

3.3.2 Cycle II

Diagnosis

Lack of interactive demonstration (C2). The platform was developed for both web and mobile usage, incorporating interactive functions that necessitate touch interactions on the mobile client. From observation of P5, prototyping is a key activity in the early phase of the project development, which offers a better understanding for project team members. This activity mainly relies on the engagement of the BA and the UI designer. The UI designer is located in Tianjian, which must WFH all the project time. The BA reflected one of the typical cases as follows:

BA: When discussing with the UI designer, I found it difficult to express the underlying interactive requirements without face-to-face communication (such as informal conversations), which made the design modified several times to get the final version of the template.

The development team reported difficulties demonstrating interactive requirements in RE practices of P5 and P6. The front-end developer specifically mentioned that interactive behavior implementation was challenging to clarify in online discussion mode in a WFH environment because such requirements cannot be demonstrated through virtual communication.

Through answering IQ1 and IQ2 in Intervention Cycle II, the development team reported that the major requirement changes were mainly caused by the understanding gap. Analyzing these changes reveals two significant requirements that were requested to update the interactive behavior in the platform. The absence of interactive demonstrations emerged as a critical challenge, contributing to the understanding gap among various stakeholders and leading to changes in the initial requirements.

The challenges C2 reported in Cycle II reveal that the understanding gap was enlarged with a lack of interactive demonstration. The constrained demonstration of requirements significantly impedes the establishment of knowledge sharing within the team. Similar to the risk delineated in C1, C2 has augmented the frequency of adjustments to requirement planning, which is a massive risk to requirements management.

Planning

Online design thinking. Design thinking (DT) is a process that emerged from design areas. Carell et al. [159] studied two cases to explore how software companies adopt DT in development. A key finding is that it is helpful for requirement engineers to adopt the mindset of designers to deal with the challenges of digital transformation projects [159]. DT approach with development context is based on four principles as follows: users' needs take center stage; deep understanding rather than large numbers of cases; interdisciplinary team; and following a clear process ("understand, empathize, define, ideate, prototype, and test") [159]. Penzenstadler [160] argued that focusing on the technical perspective requires a deeper look at the social implications of the technology. She suggested stepping back to explore "why" we are undertaking the project or program [160]. Inspired by these two prior studies, we propose to adopt design thinking in WFH prototyping to alleviate C2. To evaluate the effectiveness of

the changes, we recorded some informal interviews during the changes. Additionally, for the communication between the UI designer and the BA, the changed versions are recorded and compared.

Mixed language presentation. In agile software development, the most used technique is user story [74]. Some researchers argued user stories must be written following business language [34]. However, Bjarnason et al. [161] found software requirement specifications with more technical language may reduce the effort for coding, testing, and maintenance. The study in [12] indicates non-functional requirements are neglected if user stories satisfy system or product features only. We adopted the mixed language technique to present online to alleviate C2 and its consequent risks.

Changes

Online brainstorming. The project was to build an AI-advisor platform that includes major investment-related features. In the understanding phase, the main focus was understanding the problem and its elements. Following the vital principle (i.e., user's needs take center stage) of DT presented in [159], we stepped into the users and held some online activities based on a human-centered design approach. We invited 8 people from various professions (i.e., two relationship managers in the bank, three experienced investor users, one software architect, one web developer, and one UI designer) for online brainstorming, which gave the project team a deeper understanding of the user behavior and day-to-day business use of the platform.

Online demonstration. We studied four competitive products to help empathize with users and determine expected functionalities. Based on ideas extracted from these activities, the BA built and demonstrated the prototype to potential users. During the online demonstration, all participants run the same product to ensure all are on the same line. These prototypes offered the team possibilities for clarifying and testing interaction.

Write user stories with technical acceptance criteria. To reduce the risk of enlarging the knowledge gap, the BA in our collaborative project adopted the user-story model proposed in [161] and [12], using both “acceptance criteria” and “technical aspects” when drafting user stories. Some requirements, like web client and back-end server communication, contain functional and technical aspects. When writing user stories for such cases, user behavior acceptance criteria and API definition were considered to assist with coding and testing.

Demonstrate with technical data examples. A mixed-language technique was adopted when demonstrating the requirements to the developers. The BA presented technical examples (e.g., API tokens) to developers to convey the expected data requirements.

3.3.3 Cycle III

Diagnosis

Lack of customer feedback (C3). Customer involvement is one of the principles of agile practice [158]. However, the project was performed during the pandemic, and the customer could only engage online at the beginning of the brainstorming and testing stage. This limitation of constrained involvement in sprint meetings (P2, P3, and P6) became evident that some feature implementations failed to align with user expectations, particularly those featuring interactive design elements. From the review of requirements changes, two major causes were the customer's inability to participate in sprint User Acceptance Testing (UAT) and rapid market change. As a startup team, ORG did not have a large customer group for preliminary testing to investigate the most popular user behavior. The BA reported insufficient customer involvement, which made the user experience less than expected.

***BA:** In the potential customer testing stage, some users proposed a set of requirements enhancements. Such enhancement is minor, but it is pretty user-friendly. However, it was neglected. From the reflection interview, the user experience could be improved during the early stages if more customers are engaged in the agile practice.*

Apart from the user experience problem, the lack of customer involvement made the platform less market-sensitive. The platform was built for an investment portfolio recommendation. The contagion effect exists in the US and China stock markets during the pandemic [162], making the data-related requirement frequently change to fit the market trend. Moreover, the project's objective entailed the development of AI-driven financial products, necessitating the integration of rapidly evolving ML models tailored for financial systems, such as [163]. The WFH policy makes delivery lack instant market-related feedback from customers during development, leading to a proportion of later requirements changes.

Chapter 3. Agile RE in WFH Environment

C3 results in the lack of knowledge sharing between the customer and the agile team. It has the potential to precipitate changes in requirements, thereby constituting an additional risk to requirement management and the subsequent product release.

Planning

Roster customer. Customer involvement is one of the principles of agile practice [158]. However, this project was not entirely market-driven. The customer could only engage at the beginning of the brainstorming and testing stage. The insufficiency of customer involvement led to some implementations failing to meet user expectations, particularly regarding interactive features. As a startup team, the project team did not have a large customer group for preliminary testing to investigate the most popular user behavior. To alleviate the challenges in C3, team members were invited to consider themselves as customers to test the deliverables.

Changes

Testing using our staff as end-user. The BA, PM, and stakeholders conducted the UAT in each iterative review and testing activity. Additionally, the stakeholders occasionally demonstrated the deliverables to potential customers to collect instant feedback from potential customers.

3.4 Evaluation

To answer RQ3, we applied the comparative analysis for each technique proposed. The diagnosis and planning period is construed as the pre-treatment period, while the duration subsequent to the implementation of the change is designated as the post-treatment period. For each of the intervention cycle, we conducted some informal interviews with the participants. Additionally, semi-structured interview discussions targeting IQ3 to IQ8 were conducted to assess the efficacy of each technique. Furthermore, the defect list related to each challenge was analyzed to determine the effectiveness of the applied collaborative techniques. In this section, we provide the lessons learned (“+” indicates positive practice and “-” refers to the negative practice) from the analysis of the responses to questions in Table 3.4.

3.4.1 Constant Collaboration

The constant WFH collaboration allows rapid requirement validation and changes to fit market changes. During the project, the sprint increment was demonstrated to all, allowing stakeholders to figure out unsatisfied deliverables and adjust the subsequent sprint plan to include those requirement change requests. Stakeholders, in their response to IQ3, noted that ongoing collaboration empowers them to promptly respond to market fluctuations and modify requirements accordingly. A typical instance highlighting this dynamic was observed in Intervention Cycle II, constituting the post-treatment phase subsequent to Intervention Cycle I.

***Stakeholder A:** In the developed platform, a typical business flow is to generate the AI-based investment portfolio recommendation. The original requirement was designed to contain technology, healthcare, and financial services as equity sectors. In early August 2021, we requested to add education as one of the major sectors in portfolio construction since we sensed the impact of the draconian education reform policies suddenly released in late July 2021. To fit the market change in China, the project team adjusted the requirements in a timely manner.*

When answering IQ8, the stakeholders assign a satisfaction rating of 4 out of 5 to development, signifying that the majority of the implementation aligns with the acceptance criteria. Participants perceived that the compensatory constant collaborative activities facilitate the management of requirement changes, enabling the team to respond promptly, even within a fully distributed environment. Stakeholders posit that the rapid adjustment of requirements accommodates market changes, thereby maximizing business value.

Lessons learned 1 (+) : Increasing constant collaboration and defining communication rules allows the agile team to keep the alignment in requirement understanding under the WFH context, which can narrow the understanding gap on requirements.

3.4.2 Online Design Thinking

In addressing IQ4 and IQ5, developers expressed a challenge in acquiring a thorough understanding of the business flow during Project Phase I, which hindered their

Chapter 3. Agile RE in WFH Environment

grasp of the project requirements. They noted that the application of design thinking methodologies proved instrumental in gaining valuable insights into the conceptualization of the product's essence and attributes during Project Phase II, particularly within the post-treatment phase of Intervention Cycle II. Specifically, the back-end developer emphasized that participating in brainstorming discussions facilitated a clearer comprehension of the project's intricacies.

***Developer A:** During Phase I development, I faced some difficulties in understanding the business flow, such as what was the usage of the client dashboard and staff dashboard. Through the brainstorming discussion held ahead of Phase II development, I gained a clearer understanding of the business flow for the newly added module.*

Moreover, stakeholders responding to IQ5 affirmed that despite alterations to certain requirements during the project's duration, the deliverable ultimately adhered to the predetermined expectations. They emphasized that the management of these changes remained well-controlled, effectively addressing market fluctuations.

In contrast to the defects documented in Project Phase I, a reduced number of issues emerged concerning the functionalities that had been discussed during the design thinking stage during Project Phase II development. During the reflection discussion, the stakeholders and potential users reported that most of the deliverables met their expectations. The developers stated the online brainstorming discussion and prototype demonstration cultivated a cognitive framework, facilitating their comprehension of the business flow. The feedback reveals that design thinking contributes to requirement transfer and narrows the understanding gap in the subsequent development stage.

Lessons learned 2 (+) : The integration of design thinking into agile RE within the WFH context serves as a supportive mechanism for team members, facilitating an understanding of the business flow and contributing to the enhancement of associated business value.

3.4.3 Mixed Language Presentation

From the feedback of the developers, user stories with technical descriptions narrowed part of the understanding gap and accelerated their coding process. Developers recommended this practice as a positive approach to mitigate re-engineering resulting from knowledge gaps. In responding to IQ6, both front-end and back-end developers

argued that the adoption of a mixed language technique could prove beneficial in bridging the comprehension divide. The front-end developer further reflected that employing mixed language techniques aided in capturing a more comprehensive understanding for design purposes.

***Developer B:** In the integration development part, it was unable to discuss with teammates to receive instant feedback. It was difficult for me to understand the non-functional requirements fully. I captured what and how the design should be done using mixed language techniques containing technical language.*

Combining the analysis of the defect lists, it was determined that, during the pre-treatment period, 14 out of 49 defects (28.6%) were raised concerning integration functionalities, whereas in the post-treatment period, 8 out of 71 defects (11.3%) were identified. This observation implies that including technical descriptions in user stories could help alleviate the understanding gap among the development team and contribute to coding and testing.

Lessons learned 3 (+) : Utilizing a combination of different concerns in online requirements communication can assist with transferring from requirements to coding and testing.

In most software projects, providing the user stories with business acceptance criteria and technical description, as proposed in the study [161], does not imply successful implementation and delivery. Most often, a small proportion of features will have to be modified during the latter stages of software development because of the non-functional requirements being neglected. The project team faced a similar challenge during the WFH pandemic. Developers highlighted a specific instance wherein the API design was prematurely finalized in an early sprint, only to require updates in subsequent sprints due to limited discussion opportunities during WFH conditions.

***Developer A:** We discussed the function of clicking the “Save” button on the page - updating a user profile. The original requirement only defines acceptance criteria as “When clicking Save button, the data of client information will be saved to database table `tbl_user`, via invoking API `upd_user_profile`”. The testing in the first ten sprints was smooth, with*

Chapter 3. Agile RE in WFH Environment

good loading performance. However, when it came to the latter sprint testing, a performance issue was exposed due to the inappropriate design of the API. The insertion of user information was designed to transfer data fields one by one. However, database operation performance could be improved if these fields could be structured and combined as JSON objects for data transfer. WFH made us less discuss and consider the performance issues.

Non-functional requirements, such as those pertaining to system performance, often face inadvertent oversight during the design phase, particularly within a startup team with a lack of instant communication among different parties. The PM reported that oversight of non-functional requirements necessitates adjustments or modifications to planning in subsequent sprints, thereby diminishing the overall effectiveness of agile management.

Lessons learned 4 (-) : The WFH context, characterized by reduced communication, has a tendency to result in the inadvertent neglect of non-functional requirements.

3.4.4 Roster Customer

As a startup team, the development team faced constraints in conducting extensive preliminary testing due to the limited size of the customer group available. A retrospective analysis of the defect list revealed instances where certain defects were identified by customers only in the later stages of the development process. For example, one requirement necessitated modifications approximately six months into the project when a potential customer participated in the integration testing for the initial milestone deliverable. Although the requested enhancement was minor, rendering the system more user-friendly, it was unfortunately overlooked at the early phase of the development.

During demonstrations to potential customers, the majority of functionalities were perceived to align with market needs and business objectives. However, discrepancies between user expectations and observed behaviors were noted. In response to IQ7 and IQ8, one customer highlighted that requisite changes were raised in later sprints, attributing this delay to the absence of early engagement in UAT.

Customer: *For some requested features, the user experience was not good, which did not fit our user preference in practice. We proposed some changes in a later sprint. This could be avoided if we could be engaged during the UAT phase.*

In Intervention Cycle III, the proposed actions of roster customers yielded limited enhancements. Consistent feedback from customers remained similar during both the pre-treatment and post-treatment periods. Upon reflection, it is evident that increased customer engagement within agile practices has the potential to enhance user experiences during the initial stages of development.

Lessons learned 5 (-) : The omission of customer engagement poses a challenge for WFH-based agile RE, resulting in a user experience that deviates from the anticipated expectations.

3.5 Threats

Action research should be assessed with the following criteria: *reflexivity*, *credibility*, *resonance*, *usefulness* and *transferability* [164]. This section discusses the threats to the validity of the results presented.

3.5.1 Transferability

Concerning its transferability, which implies the possibility of plausibly applying it to other circumstances, our study is limited in two aspects. The first issue is related to the limitation of generalization. The lessons learned in Section 3.4 should be considered in light of the limit of our study. Like most qualitative research, our analysis does not support statistical generalization from a sample to a population. We study a single case in depth, with a common shared WFH background. The results can be transferable to other circumstances with similar contexts.

The second limitation to transferability pertains to the application of agile methodologies. Similar to many startups, the project team exhibits deficiencies in adhering to certain processes integral to agile practices. In the reported project, some role configurations deviate from established agile practices. Despite these deviations, the project team endeavored to align with predominant agile practices concerning requirements activities, thereby mitigating potential repercussions on the agile RE process.

3.5.2 Reflexivity

With regard to reflexivity, which is related to reflection on potential biases, the main limitation is the analysis, which mainly relies on the reflection of the project team members. In this study, we designed interview questions (Table 3.4) to obtain information from the perspective of different interviewees to alleviate the threat. In addition, when diagnosing the challenges in Section 3.3, we integrate the analysis on requirement changes to reduce the bias from the observation of the research team. When discussing the lessons learned in Section 3.4, the defect lists are combined to reduce bias from the views of respondents.

3.5.3 Resonance

Considering the resonance, one limitation relates to the project period. The project was performed during the COVID-19 pandemic, with a strict and uncertain WFH context. The majority of the countries have canceled the border restriction post-pandemic. We will not be able to apply the action research on more startups. However, WFH is increasing in software development companies. The insights gleaned from this experience report provide a practical reference for the RE community, facilitating an exploration of solutions towards mitigating challenges inherent in agile RE within the context of WFH, particularly in instances of uncertainty.

3.6 Summary

This study presents an industrial experience report from a startup company that implemented the agile development method during the COVID-19 pandemic. By analyzing the agile RE practices and related documents, we identified three key challenges specific to agile RE within the WFH context: lack of communication efficiency, lack of interactive demonstration, and lack of customer feedback. These challenges further exacerbate two RE risks: knowledge-sharing difficulties and issues with process definition. To mitigate these challenges, the company implemented various collaborative practices. The effectiveness of these practices was evaluated based on feedback obtained from interviews, and valuable lessons were derived from the action research.

One positive practice that emerged from the study was the increased emphasis on constant collaboration and the establishment of clear communication rules, which helped bridge the understanding gaps in requirements. Another positive practice

3.6 Summary

observed in our study was the integration of design thinking principles into WFH-based agile RE, allowing the team to better comprehend the business flow. Additionally, the utilization of mixed language presentations proved to be beneficial in conveying requirements for coding and testing.

Conversely, the study also revealed two negative lessons. Firstly, the WFH context made non-functional requirements more prone to being neglected, thus exacerbating the risk of knowledge-sharing difficulties. Secondly, the inability to engage customers effectively in user acceptance testing resulted in a misalignment between the product's user experience and user expectations, representing a significant risk.

The ongoing trend of professionals transitioning to permanent WFH arrangements is anticipated to witness a continuous rise [13]. This shift poses significant challenges for startups engaging in agile RE practices within the WFH paradigm. While limited research studies exist on WFH-based agile RE, this study aims to contribute practical experience to the RE community.

Chapter 4

Agile RE in ML-enabled Development

4.1 Introduction

The intricate nature inherent in the software development of ML applications poses additional challenges to RE endeavors [22]. RE researchers have voiced concerns regarding the necessity for expanded discourse on RE for ML, advocating for further real-world case studies to assess RE activities within practical ML-enabled applications [23]. Several global surveys suggest that ML is poised to become the backbone of the financial industry in the near future [28], particularly in financial centers like Hong Kong. Consequently, comprehending RE activities in ML-enabled systems within the FinTech domain can contribute to the development of more competitive FinTech products in financial centers.

Motivated by expectations from the RE community and stakeholders within the financial industry, we conduct an in-depth investigation of RE activities in ML-enabled FinTech systems through collaboration with a financial industry partner in Hong Kong. During this industry-academia collaboration, a DL-based portfolio recommendation system was designed and developed. Our researchers actively participated in the case study to explore the challenges of RE activities within a real-world development context. Based on our observations, interventions were employed to guide and modify RE activities within the selected project, aimed at discerning which requirement characteristics should be prioritized when addressing ML-related requirements. Overall, two RQs were formulated to guide the investigation and provide insightful answers.

RQ1. What are the difficulties in RE activities for developing ML-enabled FinTech systems in a real-world context?

RQ2. What characteristics should be considered when handling ML-related requirements in RE activities?

To answer RQ1, we recorded the complete RE process of activities in a real case. Then, we analyzed our observations together with the collected documents to summarize the difficulties faced during the execution of the RE activities. Based on the observations, we proposed a set of characteristics of data and model requirements that intersected in guiding RE activities. We employed the framework and iteratively made active changes. Finally, we discussed the framework in detail and explored the effectiveness as described in RQ2.

This study was published at the 2023 30th Asia-Pacific Software Engineering Conference (APSEC 2023) [165]. The rest of the chapter is organized as follows. Section 4.2 emphasizes the methodology employed in our study. Based on the details of RE activities and proposed practices described in Section 4.3 and 4.4, the lessons learned are discussed in Section 4.5. In Section 4.6, we provided the analysis of limitations. Finally, Section 4.7 concludes the study.

4.2 Methods

Various research approaches exist for evaluating and comparing conceptual modeling methods, each with advantages and limitations [166]. To answer the RQs, we employed the action research method. One of our researchers participated in the project to make changes to RE activities. The authors of [43] argued that such a study is regarded as action research if the study shares similarities with the case study research method while focusing on active participation for change. It is well-known that the case study method is suitable for testing theories and artifacts in complex settings where there is little control over the variables [167]. The method enables the collection of data and evidence about the expressiveness and usefulness of the framework. The changes and observations were integrated into our case and intervened in a real-world context. Hence, such an action research method helps to address and contribute to the RQs.

4.2.1 Case Selection

The main objective of the study is to comprehensively explore the RE activities involved in the development of ML-enabled FinTech systems. The unit of analysis in our study is defined as a project with real-life context that ideally meets a particular set of criteria.

Chapter 4. Agile RE in ML-enabled Development

Firstly, considering the relevancy of the framework objectives, the project should be executed as a development project that includes ML techniques. Additionally, the project should consist of RE activities for traditional and other data requirements except for the ML-enabled part such that we can compare the RE activities for both traditional and ML-related requirements to answer RQ1. Lastly, the outcome of the project should be a digital financial product. Thus, we evaluate the case study of ML-enabled software systems from the domain of FinTech. These criteria ensure the richness and availability of data for validating the proposed practices and finding evidence for their usefulness.

Several industry-academia FinTech projects were performed in our research group. Based on the above selection criteria, the industry-academia project Integrated Smart Investment Platform (ISIP) was selected as our case for conducting this action research.

4.2.2 Study Context

ISIP is a platform that contains digital services such as robo-advisor, user behavior analysis, financial modeling customization, and sales management. The target users of this platform are financial institutions as well as individual investors. It aims to provide affordable financial advisory services to anyone anywhere and at any time. The project was collaboratively run by ORG and the City University of Hong Kong.

4.2.3 Procedure of Study

The study was conducted in four steps, aligning with the five-step research process described in Section 3.2, by integrating observation and diagnosis for discussion, as well as evaluation and reflection for discussion. The first step of the study was *active participation*. Two of our authors participated in the whole process of project development, which lasted for one and a half years. One worked as the BA to make active changes to RE activities. Another one assisted in the back-end development and model integration. The second step was *observation*, which ran concurrently with the active participation step. We recorded the participant observations for further analysis. Through the observations and diagnosis, the difficulties of RE activities were summarized. The third step was *action*. Based on the previous studies and observed challenges, we planned and proposed a guidance framework to consider characteristics from data and model requirements when performing RE activities on ML-related requirements. The last step, *reflection*, was performed at the end of each intervention. This step involves evaluating interventions by reviewing relevant documents and gathering participant reflections to distill lessons learned.

4.2.4 Data Collection

[156] presents multiple data collection techniques for case studies and action research. In our study, the main techniques applied are *observation* and *relevant documents*.

Participant Observation

One of our authors worked as the BA and was responsible for the requirements-related activities. As a result, our authors fully recorded the requirements and activities. The project adopted the agile development mode and held sprint review meetings. In every sprint review meeting, the project team validated the product increments and the model development. Our research team observed and recorded all the reflection parts from the sprint review meetings.

Relevant Documents

With permission from ORG, we collected all necessary materials, including requirement documents, data specifications, model specifications, and others. Our observation and analysis in Sections 4.3, 4.4, and 4.5 are based on these documents. Table 4.1 shows the summary of documents and ML-related highlights. A hybrid ML model with Long-Short-Term Memory (LSTM) [168] and Convolutional Neural Network (CNN) [169] was developed in this project. LSTM was utilized to transfer vital time series financial data in the network, and CNN was embedded to extract and learn from the features.

Table 4.1 Summary of ML-related requirements

Document	Data and ML related Details
Business Requirements Document	201 data descriptions 19 ML-related data descriptions
User Stories	147 user stories 15 ML-related user stories
Data Specifications	507 data fields 44 ML-related data
Model Specifications	12 trained features 27 features analysis

4.2.5 Data Analysis

The analysis was conducted using qualitative methods. The procedure followed a reviewed protocol that used a clear chain of evidence [170]. To answer RQ1, we participated in the whole project development period to observe real-life practices. The DS and BA summarized the difficulties in each RE activity in sprint review meetings based on their observations and collected documents. Other authors reviewed and recorded the findings. Regarding RQ2, we intervened and approached the proposed guidance in RE activities. In the sprint review meetings, feedback was collected from the project team to review the applied practices that contain the proposed consideration factors. Combined with the development progress in each RE activity, the effectiveness was discussed by the collaborative project team members, and the results were recorded and reviewed.

4.3 RE Activities

The ML-enabled system in the FinTech domain contains a massive amount of data, which is the core requirement. Altarturi et al. [171] revealed that traditional RE methods focus on user-centric requirements but not data requirements for developing a big data system. Bosch et al. [172] emphasized the need to adapt and complement old practices and techniques with new ones rather than replace the old practices entirely in data-driven development. The development of the ML-enabled FinTech system faces similar problems because of the nature of the ML models, which rely on large amounts of training data. In this study, we approached RE activities by considering the data and model requirement analysis and negotiation on ISIP. Then, we conducted an in-depth review of our practices to answer RQ1. Based on the observations, we explored the difficulties in RE activities for the ML-enabled system from the FinTech domain.

The main participants in RE activities are one Data Scientist (DS), one Business Analyst (BA), and two stakeholders (SH). The RE process in the case followed traditional RE activities (i.e., elicitation, analysis, modeling, validation, and specification). Combined with the context of ML-enabled system development, one new RE activity was introduced: data and model analysis conducted by the DS. The negotiation process was adjusted to engage multiple parties (DS, SH, and BA) to tackle the conflicts for ML-related requirements. The validation and verification (V&V) process was adjusted to engage DS to assist with validating and verifying the ML-related requirements.

Figure 4.1 presents the typical processes of RE activities applied in the case. The DS, SH, and BA were engaged in the initial elicitation process. With elicited requirements, the BA analyzed all requirements while the DS performed the analysis on data and model requirements parallelly. After that, the DS and BA posed conflicting requirements for negotiation. If the conflicts required re-analysis, the DS and BA would perform another round of analysis. In the negotiation stage, the SH helped emphasize the needs of the business and make the decision. Once the conflicts were resolved in the negotiation, the BA modeled the requirements for further V&V. In the process of V&V, the DS helped validate the ML-related requirements. Modeling and V&V were bidirectional. If the V&V failed, the requirements would be placed back into modeling for the next validation iteration. During validation, the newfound conflicts would lead to the next round of negotiation and modeling. Once all the validation was finalized, the BA would draft the specification to guide further development and testing.

In the following discussion, we summarized the difficulties observed from each RE activity. The observations mainly relied on collecting feedback from the DS and BA. Table 4.2 presents the problems in summary.

Table 4.2 Difficulties of RE activities in ML-enabled FinTech system, from perspectives of DS and BA

RE Activities	Difficulties
Requirement Elicitation	<ul style="list-style-type: none"> • (DS) ML-related data requirements are rare and ambiguous in business requirements. • (BA) ML-related data requirements lack explainability from the user's point of view.
Requirement, Data and Model Analysis	<ul style="list-style-type: none"> • (DS) Data quality is hard to guarantee in a real-world context. • (BA) Data quantity increases the workload of requirement analysis.
Requirement Negotiation	<ul style="list-style-type: none"> • (DS and BA) Balancing business needs and model requirements is challenging.
Requirement Modeling	<ul style="list-style-type: none"> • (BA) ML-related features are vague when presented in user stories.
Requirement Validation and Verification	<ul style="list-style-type: none"> • (BA) Defining the acceptance criteria for validating ML-related requirements is difficult. • (DS and BA) The verification rules of ML-related requirements should be adjusted regularly.
Requirement Specification	<ul style="list-style-type: none"> • (BA) The specifications in the ML-enabled system should be defined as auto-adaptive.

Chapter 4. Agile RE in ML-enabled Development

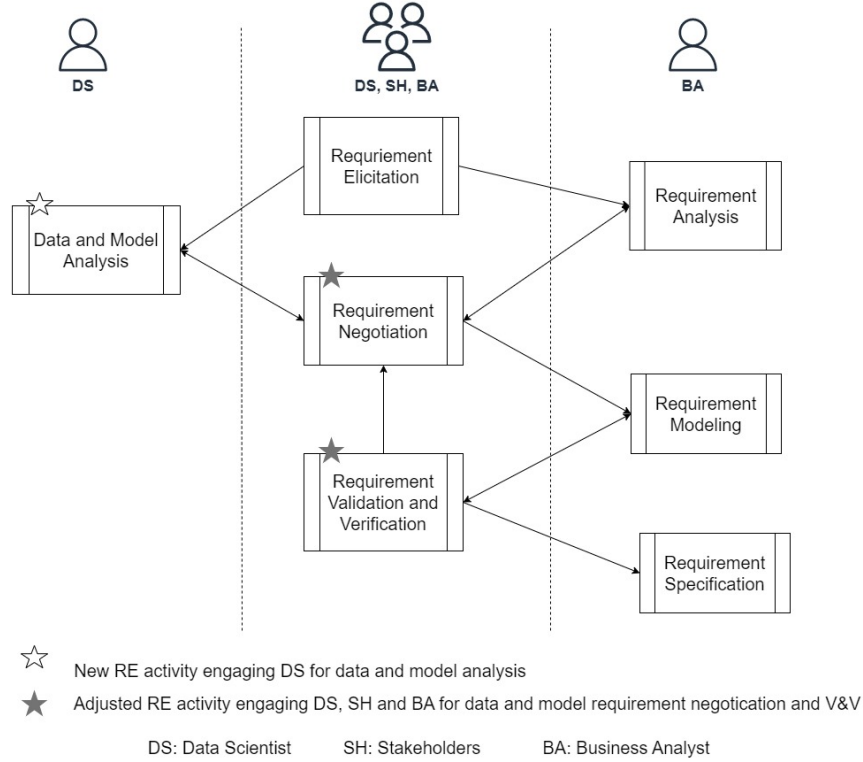


Fig. 4.1 Flow of RE activities in the development process of the ML-enabled system in Project ISIP.

4.3.1 Requirement Elicitation

The traditional requirement elicitation process is an activity that requirement engineers carry out with customers and software end-users to define the application domain, services provided in the system, software performance, hardware requirements, and so on [173]. ML systems usually profit from additional data that increase the quantity and quality of the core data [17]. Data is an integral part of the requirement for the ML-enabled system, especially in the FinTech domain, since data contributes to the implementation of both financial functionalities and model training.

In our practice, the DS was injected into the communication to define data requirements for modeling. In the stage of requirement elicitation, the DS and BA were both engaged in identifying data for requirements that can be used to generate business values and subsequently enhance user requirements. The SH also participated in this stage and fed the business requirements for elicitation. Combining analysis with collected documents, we summarized two challenges in this activity.

DS: ML-related data requirements are rare and ambiguous in business requirements. The DS stressed the importance of data quality targets for ML models, and indicated requirements must be precise for a successful project. ML-related data requirements are rarely described in the business requirement document (BRD) we collected. From Table 4.1, we can see that only 19 ML-related data (with 9.45% among 201 data descriptions) are demonstrated in BRD. Due to the lack of expertise in the ML domain from the view of business, only ambiguous descriptions were presented. For example, “*It requires to backtest the model with 15 years of historical data until the most significant attributes found.*” describes the requirements for the training process of the predictive model. However, how to define valuable “*historical data*” was neglected. The DS stated the ambiguous data requirements increased the difficulty of defining precise data requirements to feed into ML models.

BA: ML-related data requirements lack explainability from the user’s point of view. The DS is responsible for eliciting the required data for feeding ML development in the requirement elicitation stage. From our observation, some data requirements proposed by the DS focus more on fitting the ML algorithms but lack explainability from the user’s point of view. The BA reported such requirements made data requirements challenging to elicit and explain to customers. In the proposed ML model in the project, one of the training features is the “*convolution of the market price*”. The BA reported such data contains the nature of the ML model, which spent much time explaining its business value to the customers.

4.3.2 Requirement, Data and Model Analysis

The requirement analysis process aims to transform stakeholders’ requirement-driven view of desired services into a technical view of a required product that could deliver [174]. Paralleling the requirement analysis process, the DS performed data and model analysis. In this stage, the DS should analyze and combine with the context to decide what model should be appropriately adopted in the project and what data should be used for ML modeling. In the requirement analysis process, we observed and summarized two difficulties.

DS: Data quality is hard to guarantee in a real-world context. The DS concerns data quality on completeness, consistency, and correctness [17]. A common approach to meet the insatiable need for data in ML is to feed public datasets to model training. However, public datasets may conflict with the correctness dimension of an industrial project. Correctness refers to the degree to which engineers can rely on true data [17].

In the ISIP, some open-source data were collected and fed into the model training process. The DS reported it is challenging to control the quality of data collected from real-world contexts.

BA: Data quantity increases the workload of requirement analysis. A general argument in ML modeling is “*The model’s performance relies on more data*” [17]. However, for the increased data requirements, the BA should identify the additional data sources as part of the analysis with stakeholders. As reported by the BA, such practice led to an increased workload of identifying the “protected” characteristics of customer data. The DS proposed feeding some customer data into model training to predict the investment preferences of customers. In the financial system, some customer data are sensitive for protection. The BA must locate them and pose them into negotiation with multiple parties.

4.3.3 Requirement Negotiation

The requirement negotiation happens at the beginning stage in the traditional RE process [175]. Altarturi et al. [171] proposed the negotiation stage conducted by DS and the requirements engineer for developing a big-data system. Similarly, requirement negotiation is an essential RE process for developing the ML-enabled system. In the practice of ISIP, the negotiation process was adjusted to tackle conflicts of ML-related requirements. After performing analysis from BA and DS, some data requirements were posed for negotiation to decide on data taken in or taken out. The SH was also engaged in making the business decision. The major challenge in this stage is balancing needs from the business and model perspectives.

DS and BA: Balancing business needs and model requirements is challenging. Observing from the practices, some data requirements conflict after running through the data analysis and requirement analysis process. For example, the data related to *transaction details* would help improve the performance of the model if transaction recordings could be used for investment preference prediction in the training process. However, from the view of user needs, the data contains sensitive information, and it is difficult to negotiate with businesses to get approval. Therefore, the DS and BA should pose conflicting requirements and negotiate with the SH to consider the in-or-out decision at this stage.

4.3.4 Requirement Modeling

Following the agile practice, the user story model was adopted to present requirements in ISIP. This process finalizes the consolidated model to yield the user stories before validation. Unlike traditional requirements, presenting ML-related requirements in user stories is a challenge.

BA: ML-related features are vague when presented in user stories. ML-centric software goes through frequent revisions initiated by model changes, parameter tuning, and data updates [176]. In the practice of drafting user stories, ML-related features are vague or hard to define. For example, when defining the user story “*automatically calibrate input data in ML training process*”, the rules of calibration are vague in the draft user stories. One of the causes is the rapid market change, which means that the data needs to be updated continuously to fit the changes.

4.3.5 Requirement Validation and Verification

Requirement validation is the process of checking whether requirements indeed define what the customer wants from the system [173]. Requirement verification is a process of proving that each requirement has been satisfied [177]. Terry Bahill and Henderson [177] presents multiple approaches that can help achieve the verification targets, such as logical argument, inspection, expert review, test, or demonstration. Defining the acceptance criteria is a challenging task concerning ML-related requirements. In the V&V process, if the V&V fails, the requirements will be placed back for the next round of modeling or negotiation until it fits the acceptance criteria. Two difficulties were reported in the activities of V&V.

BA: Defining the acceptance criteria for validating ML-related requirements is difficult. Evaluating the model performance is one of the most critical stages of ML system development. It is difficult to define appropriate rules for assessing the model performance in a real-life context. The metrics utilized for evaluating time series data in ISIP are R^2 (R square score), MSE (mean squared error), and $MAPE$ (mean absolute percentage error). However, it is challenging to combine them with financial explanations for briefing customers to verify the standard criteria.

DS and BA: The verification rules of ML-related requirements should be adjusted regularly. Many ML systems need to be retrained over time to adjust to the recent data [17]. As a result, verification needs to be performed continuously. The DS and BA must cooperate to adjust the rules of verification regularly.

4.3.6 Requirement Specification

Once the requirements have been validated and verified, the user and software requirements can be written in a requirements document. Unlike conventional requirements, both the data and model specification should be documented in the ML-enabled system requirement specification. In [178], the authors proposed four components for defining requirement specification in machine-learned components: dataset specification, model specifications, development process specification, and traceable path. Following the suggestion in [178], we included the above four components in our practice and summarized the difficulties in specification activity.

BA: The specifications in the ML-enabled system should be defined as auto-adaptive. Calibration of data and models is a regular activity in ML model development. For the ML-enabled system, the dataset should be specified in the source and format. Based on the algorithm adopted in the system, model specifications typically define how the properties should be held during implementation. The training process steps should be specified in the development process specification. For verification, it is pivotal to construct a traceable path to demonstrate the compliance of the source code. To fit the calibrating changes, the BA must define an auto-adaptive specifications format to fit ML process auto-calibration.

4.4 ML-related Requirement Characteristics

Amershi et al. [176] discovered the typical nine stages of ML workflow widely adopted in multiple Microsoft teams. In the workflow, some stages are data-oriented (i.e., data collection, learning, and labeling), while others are model-oriented (i.e., model requirements, feature engineering, model training, evaluation, deployment, and monitoring). In summary, the RE activities in ML workflow mainly consider two types of requirements: data and model requirements.

Motivated by the observations discussed in Section 4.3 and inspired by the ML nine-stage development framework presented in [176], we proposed a framework consisting of a set of ML-related requirements characteristics that should be considered to guide RE activities when developing ML-enabled systems. This framework was applied in RE practices in later sprint development for intervention practice. Figure 4.2 presents the characteristics from perspectives of data and model requirements, which includes six key requirement components (each is labeled as an ellipse) intersected in RE activities. For each requirement component, the characteristics are labeled

4.4 ML-related Requirement Characteristics

as dashed rectangles. In this section, we discussed these characteristics in detail to answer RQ2.

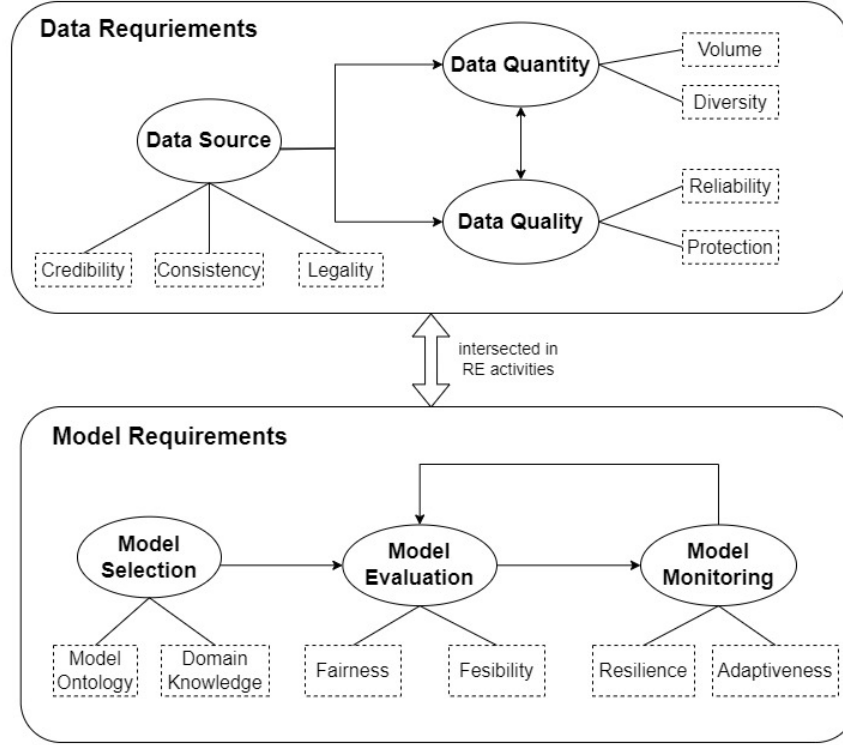


Fig. 4.2 Guidance framework intersected in RE activities for developing ML-enabled FinTech systems, consisting of data and model requirement characteristics.

4.4.1 Data Requirements

Regarding a FinTech system, data is the most crucial requirement. Extending to developing the ML-enabled system in the FinTech domain, the data can be classified into two parts. One is financial-related data formulating the core functionalities of financial services, and the other is model-related data that supports the development of the ML model.

Considering the importance of data in ML-enabled FinTech systems, the data source, quantity, and quality are key requirement components that intersect in RE activities. The requirement engineers should ensure the quality of the data source at the early stages of RE activities such as elicitation and analysis. Data quality and quantity should be intersected to guide the complete process of RE activities.

Data Source

Hybrid datasets were utilized to model the prediction of the equity price in developing ISIP. Short-term technical price-related data (e.g., open and close price) and long-term fundamental statistical data (e.g., price-earning ratio and cash flow) were collected for the model development. Technical data were mainly gathered from open source, while the fundamental data were primarily collected from financial services or extracted from the annual financial reports of the assets. Additionally, some customer data was fed for training as user preference features. Seeing the diversity of financial data sources, *reliability*, *consistency*, and *legality* are three characteristics proposed to be taken into account when performing requirement elicitation and analysis.

The ***credibility*** of the open-source data is paramount. The BA should assist in verifying whether the data is trustworthy from the perspective of business requirements at the early stage, avoiding the cost of rework.

Consistency is the concerned characteristic regarding the paid data collected from financial services. From the view of requirement, the BA and DS should figure out whether the paid data is consistent with open-source data when evaluating different dimensions (i.e., technical and fundamental) of the equity. In that case, the data fed into model training could keep alignment to ensure the correctness of the model prediction.

For the customer data utilized in the process of ML model training, identifying the ***legality*** of the source plays a vital role. Reviewing the RE practices, we found that the DS was concerned more about how to use the data to improve the performance of ML models. However, the DS neglected the availability of the data, which requires the engagement of the BA to verify the legality from the view of business requirements.

Data Quantity

When developing the ML-enabled system, numerous data should be fed into the training process. As discussed in the requirement analysis activity in Section 4.3, the workload of analysis for data requirement will be increased in this stage. Considering the data quantity, one of the characteristics that need to be considered is the ***volume*** (i.e., what should be the definition of “enough” for feeding data into model training). Reviewing the project practices, the data quantity should be defined by considering the nature of the data as well as the preliminary statistical experiment.

Another characteristic of the data quantity that should be concerned is the ***diversity*** of datasets. The data quantity should not be identical for all. Take long-term fundamental statistical data trained in the project as an example. These data are

4.4 ML-related Requirement Characteristics

not daily time series data, which should not share the same dimension as short-term technical data. The diversity of data quantity also impacted drafting data specifications in the last step of requirement activities. This consideration factor requires the BA to produce different formats of the specifications.

Data Quality

Vogelsang and Borg [17] argued more data does not mean better. The data quality must be the key ML-related requirement component to be controlled in the whole RE activities when developing ML-enabled systems. In ML workflow, data cleaning involves removing inaccurate or noisy data [176]. Reviewed by the DS, the data quality is hard to guarantee in a real-world context. The open-source dataset is unreliable because it is free for usage, and no one is paid for maintenance. Therefore, the BA must assist in controlling the *reliability* of the data from collection data sources to finalizing data usage in model development.

According to the reflection by the BA, another characteristic that needs to be considered for data quality relies mainly on defining the “protected” attribution from the financial data of customers. It helps to prevent the leakage of sensitive customer data. Therefore, controlling the *protection* attribute is crucial when performing RE activities to protect the business values.

4.4.2 Model Requirements

The ML models can be described in diverse ways depending on the techniques applied [179]. Regarding the model requirements, the model selection, validation, and monitoring process were engaged with the BA for defining specifications. The BA should participate in explaining business requirements for model selection. When the model is developed and deployed, defining evaluation benchmarks and rules for monitoring calibration in requirements is crucial. The components of evaluation and monitoring are cycle interactive. Therefore, we discussed these three model requirement components in the following subsections.

Model Selection

In the stage of model requirements following the process of [176], the DS should decide on appropriate model adoption for the project context. ML model selections

Chapter 4. Agile RE in ML-enabled Development

are based on the consideration of attributes of the models and business requirements. This process was integrated into requirement elicitation activity by the DS and BA.

Based on the comprehension of the *model ontology*, the project team applied a hybrid DL model as equity price prediction in the project. For the BA, it is vital to learn the underlying model ontology of the selected model. Thus, the BA was able to transfer the knowledge in coming activities like negotiation, specification, and presenting the system to different parties.

The DS was requested to learn the business values when selecting the model. In this stage, the BA should transfer the *domain knowledge* (i.e., explaining each financial item) to elicit the model requirements. Therefore, domain knowledge is another characteristic that needs to be taken into account in the model selection.

Model Evaluation

Combined with the real-world context of FinTech, the ML model evaluation is a challenging problem. One considered characteristic in the model evaluation component is *feasibility*. The evaluation rules must be executable realistically so practitioners can verify the requirements.

In the developed system, the recommended portfolio generated from the ML model consists of equities from different regions. The simple benchmark of the index is insufficient. When defining the evaluation rules for requirements verification, *fairness* should be another key characteristic for consideration. Combining the recommendation from domain experts and analysis of the model development, we defined the hybrid benchmark consisting of different weights of different indexes for evaluating the model performance. The fairness of model evaluation defined in the specification can help to better define the acceptance criteria for the system.

Model Monitoring

The FinTech system contains a large amount of market data so that market change can easily lead to the calibration of the training dataset and rules. Therefore, in model monitoring, we proposed including *resilience* as a consideration characteristic, that is, adding changeable rules to monitor market changes.

The aim of applying ML technologies in the FinTech domain is to make the service more convenient. One of the designs in the ML-enabled system is to allow auto-adaptive rules. Following the rules of the design, the monitoring mechanism must

be adaptive as well. Thus, *adaptiveness* is the next characteristic concerning model monitoring in requirement validation activity.

4.5 Discussion

In this section, we discuss the lessons learned combined with the observations from Section 4.3 and feedback on practices approached in Section 4.4.

4.5.1 Difficulties in RE Activities

By reviewing the summarized difficulties presented in Table 4.2, it was observed that the BA and DS faced diverse difficulties when performing their ML-related RE activities. The difficulties reported in the requirement elicitation and analysis activities were mainly caused by a lack of domain knowledge from the perspectives of DS and BA. For modeling, V&V, and specification activities, the difficulties were mainly caused by the insatiable nature of the ML models and the rapid changes in financial data.

Similar to the importance of negotiation mentioned in big data system development [171], our observations reveal that requirements negotiation is a key activity, and thus, it is vital to engage the services of a requirement engineer and data scientist. The data and model requirements can be specified through negotiation, alleviating several potential conflicts.

Lessons learned 1: The lack of domain knowledge, the insatiable nature of ML models, and the rapid changes in financial data pose difficulties in performing RE activities for ML-enabled Fintech system development. In contrast to traditional RE activities, requirements negotiation should be adjusted to include and engage the services of a data scientist and requirement engineer to alleviate conflicts of ML-related requirements.

4.5.2 ML-related Requirements

In our reflection, data requirements mainly concern source, quality, and quantity components. Table 4.1 shows the data count majority for FinTech functionalities and the model training. However, more is not better [17]. Balancing the data quality and quantity from the source is essential.

Chapter 4. Agile RE in ML-enabled Development

Considering the context in the FinTech domain, customer financial data is sensitive. Protection is one essential characteristic to be concerned with when feeding customer data into ML model training. The DS and BA should pay more effort to verify the legality of the data source and the reliability of the data quality when handling open source market data. The feedback from the project team reveals such practices in later sprints tackle the difficulties reported in analysis and negotiation activities.

Lessons learned 2: When developing the ML-enabled FinTech system, the inclusion of customer financial data and open-source market data in the ML training easily causes legality and reliability problems. Therefore, it becomes imperative to protect and validate training data associated with customers and financial data throughout the requirement elicitation and analysis phases.

The ML-enabled system development workflow is non-linear, containing loops for feedback calibration [176]. Integrated with the FinTech context, the financial data changes rapidly, which requires auto-feeding into model training. Therefore, resilience and adaptiveness should be included in defining the rules in model monitoring. The project team reports the V&V process benefits from these practices.

Lessons learned 3: In ML-enabled Fintech system development, the rapid change of market data and model training ontology make auto-adaptive model monitoring rules essential in activities of ML-related requirements verification and validation.

4.5.3 Generalization

This study aims to investigate RE activities for ML-enabled applications within the FinTech domain while also exploring how these activities may share similarities and differences with those in other domains.

Much like the challenges faced in FinTech, ML models in other domains can demand vast amounts of data and undergo rapid evolution. Across all domains, effective requirements negotiation should involve not only traditional stakeholders but also data scientists and domain experts. This inclusive approach is essential for bridging the gap between technical capabilities and domain-specific needs, thereby preventing conflicts that could arise from misunderstandings regarding ML-related requirements. Furthermore, requirements negotiation must account for the evolving

nature of data and model performance, adapting to the unique dynamics of each domain.

When handling data requirements, the inclusion of sensitive data, whether financial, medical, or personal, raises similar legality and reliability concerns across various ML applications. Protecting data privacy, ensuring data is legally sourced, and validating its reliability are crucial steps in the development of any ML system. The sensitivity of data may vary across domains. For instance, patient health records may require different protection measures compared to social media data. Customizing data protection strategies to fit the specific requirements of each domain is necessary.

Similarly, when dealing with model requirements, rapid changes in data are not unique to the financial sector; they are common in domains like e-commerce, social media, and autonomous systems. Auto-adaptive model monitoring becomes essential when data patterns are subject to frequent change, requiring ongoing validation and updates to the model. Different domains will have different rates and types of data change. For instance, seasonal trends in retail differ from the real-time requirements of autonomous vehicles. Monitoring rules should be tailored to the specific characteristics and dynamics of the data in each domain.

Lessons learned 4: Although the lessons learned from the FinTech domain are applicable to other ML applications, extending them requires careful consideration of domain-specific factors, including data type, regulatory requirements, and the dynamics of data change. By tailoring these insights to the unique context of each domain, ML developers can effectively address challenges across diverse applications.

4.6 Threats

We identified the limitations associated with this research and the measures we took to mitigate these issues in our research design. We approached the ACM SIGSOFT empirical standard [164] in our research. This section discusses the limitations reviewed from the criteria proposed in [164].

4.6.1 Multivocality

Multivocality refers to the property of being based on people with different opinions and backgrounds [164]. The main limitation of our study is that our study relied on

observation and reflections from the perspectives of BA and DS, who were major participants in ML-related RE activities. The activities engaged different parties (e.g., stakeholders, developers, and customers). The reflections from BA and DS included feedback from these parties, who provided suggestions from different perspectives.

4.6.2 Credibility

Credibility refers to the extent to which conclusions are supported by rich, multivocal evidence [164]. Concerning credibility, the limitation of our study is the analysis focuses mainly on participant observation. To alleviate the limitation, our analysis combines with the ML-related documents, which supports a qualitative analysis.

4.6.3 Transferability

Transferability refers to the extent to which the results of a study could plausibly apply to other sites, people, or circumstances [164]. Even though we only studied one case in our study, the context and research design are presented with details, which share the lessons learned for other cases with similar contexts and settings. The ML technologies applied in this case are DL models (i.e., LSTM and CNN), which follow a similar SE development process with other supervised learning techniques [176]. The adjusted RE activities and proposed framework can be transferred to other project teams adopting similar ML techniques.

4.7 Summary

In this study, we explored the RE activities in the real-world context of developing the ML-enabled FinTech software system. From the observations and analysis, the difficulties (presented in Table 4.2) were reported from the view of the data scientist and requirement engineer, which answers RQ1. The lack of domain knowledge, the insatiable nature of ML models, and the rapid changes in financial data are the primary causes that pose the difficulties. Among the RE activities, the data scientist plays a crucial role in negotiating, validating, and verifying ML-related requirements.

To explore what characteristics should be considered to handle ML-related requirements in RE activities (RQ2), we proposed a guidance framework intersected in RE activities from perspectives of data requirements (source, quantity, and quality) and model requirements (selection, evaluation, and monitoring). Finally, we

reviewed the action study and shared the lessons learned. The feedback from the participants of RE activities reveals these practices, such as considering data protection, adaptive evaluation rules, and others, should be well adopted in ML-enabled system development.

To broaden our results, we plan to extend our study to include interviews with more requirement engineers in the ML system development teams, sharing a similar context with our study. We also aim to explore a broader range of cases across different domains to generalize the proposed framework, which can guide RE activities for developing ML-enabled applications in various domains. In addition, we plan to conduct a deep review from the perspective of other ML-related requirements, such as incremental learning, data streaming, model storage, and so on. Furthermore, we will investigate suitable statistical measures for data requirements, including methods for quantifying noise and identifying outliers. These approaches will enable us to gain deeper insights into RE activities in real-world practice.

Chapter 5

SimAC: Requirement Elicitation in Agile RE

5.1 Introduction

In the realm of agile RE, the creation of Generating Acceptance Criteria (GAC) to expound upon user stories assumes a crucial role during the sprint planning phase, serving as a guideline for delivering functional solutions. The process of generating GAC entails extensive collaboration and human intervention. However, the absence of specifically tailored datasets designed for User Story paired with Acceptance Criteria (US-AC) poses notable challenges for supervised learning methodologies aiming to automate this process. Recent advancements in LLMs have demonstrated their noteworthy text-generation capabilities, obviating the necessity for supervised fine-tuning. Consequently, LLMs present the potential to automate this process.

In light of this, we introduce SimAC, a framework harnessing LLMs to simulate agile collaboration, delineated into three distinct role groups: requirement analyst, quality analyst, and others. Guided by role-specific prompts, LLMs assume these roles sequentially, adhering to a *create-update-update* paradigm in GAC formulation. Given the absence of definitive benchmarks, we engaged practitioners to establish a gold standard to assess the comprehensiveness and validity of auto-generated US-AC against manually crafted ones. Additionally, we enlisted the evaluation of eight seasoned agile practitioners utilizing the INVEST framework to gauge the quality of US-AC.

The primary objective of this work is to propose an automated acceptance criteria generation framework, namely SimAC, which aims to reduce the initial human effort

involved in GAC during user story elaboration in agile software development. To gain an in-depth understanding of the effectiveness and practicality of SimAC, we proposed the following RQs to guide our study.

RQ1. How is the generality of SimAC employed in different LLMs? This RQ investigates whether SimAC carries universal effectiveness on diverse LLMs. To answer it, LLMs with different families and parameter sizes are included in experiments.

RQ2. What is the contribution of each role group in SimAC? SimAC incorporates the cooperative endeavors of three distinct role groups in GAC, adhering to agile practice. In this RQ, we aim to discern the individual contributions of these role groups.

RQ3. How do K-shot promptings impact the performance of SimAC? Within SimAC, a diverse set of examples are integrated into prompts to guide the LLM in GAC. Hence, this RQ aims to examine the impact of the number of interactive examples.

RQ4. How do different settings of the similarity threshold impact the performance of SimAC? In the deduplication phase in SimAC, the similarity score is utilized to eliminate the redundancy. Different settings of the similarity score threshold produce distinct outcomes. Thus, this RQ aims to explore the impact of these settings on the SimAC's performance.

The findings indicate consistent enhancements across all tested LLMs, including the LLaMA and GPT-3.5 series. Notably, SimAC notably amplifies the efficacy of *gpt-3.5-turbo* in GAC, yielding enhancements of 29.48% in completeness and 15.56% in validity, alongside the highest INVEST satisfaction rating of 3.21/4. Furthermore, this research offers extensive case studies to elucidate SimAC's efficacy and constraints, thereby illuminating the potential of LLMs in automating agile RE.

This study [180] was accepted by the Automated Software Engineering Journal (ASEJ 2024). The rest of the chapter is organized as follows. Section 5.2 presents the preliminary analysis. Section 5.3 introduces the framework proposed in our study, and Section 5.4 demonstrates the experiment design settings. Section 5.5 showcases the detailed experiment setups and results for answering RQs. Furthermore, we provide comprehensive case studies on Section 5.6. In Section 5.7, the threats of our study are reviewed. Finally, Section 5.8 concludes our study.

5.2 Preliminary Analysis

In this section, we perform a preliminary analysis of the GAC process to investigate what roles should be integrated into our proposed framework and how the procedures should be implemented.

5.2.1 Role Division

In an agile team, there are several key roles that collaborate to deliver software increment, as presented in Figure 2.2. These roles may vary slightly depending on the specific agile framework being used but commonly include Product Owner (PO), Business Analyst (BA), Quality Analyst (QA), Developers (Dev), Project Manager (PM), and other relevant team members.

PO and BA are two roles that mainly participated in agile requirement activities [6]. The PO acts as a liaison between the development team and stakeholders, ensuring that the product aligns with stakeholder expectations and requirements. On the other hand, the BA focuses on analyzing the business needs of the organization and translating them into functional requirements for the development team to implement. It is noted that in some agile teams, the PO is frequently staffed by a BA [181] owing to their shared expertise and knowledge. Due to their joint responsibility for documenting acceptance criteria as the initial step in the GAC process, we have combined these two roles for introduction to the LLM. Following the convention observed in [6], we use the umbrella name Requirement Analyst (RA) to represent these two roles. Both of the roles share the essential responsibilities of comprehending and transferring requirements, making them integral contributors to initiating the US-AC.

Within an agile team, the QA is crucial for ensuring that the software product meets the desired quality standards. Their primary responsibilities involve test design and execution. Given that acceptance criteria serve as the reference for acceptance testing [182, 183] in agile development, the QA assumes a pivotal role in refining acceptance criteria by incorporating testable specifics. Therefore, the QA constitutes the second group introduced to LLM for GAC.

During sprint planning sessions, the agile team actively participates in discussions, as depicted in Figure 2.2. Other participants, such as developers and project managers, are responsible for updating acceptance criteria based on their respective expertise, if necessary. However, empirical observations suggest that these roles exert less effort on GAC and provide support only when essential. Given this observation, the remaining

roles are grouped as Others and introduced to the LLM to provide supplementary details.

Based on the above observations and analysis, we categorize the roles into three primary groups in our proposed framework for simulating the collaboration process in GAC: RA, QA, and Others.

5.2.2 Procedures

Following the agile practice presented in Figure 2.2, the acceptance criteria are firstly crafted by the RA, then it is collectively revised by the agile team members.

To establish an appropriate procedural design introduced to the LLM, we constructed a preliminary analysis dataset D_p . Excluding projects selected for evaluation, D_p was constructed by randomly selecting two samples from each of the remaining group projects. During the preliminary analysis, we designed a basic command (referred to as *cmd_t*) to direct the LLM to act in a specific group of roles separately. Three groups of roles in *cmd_t* are specified in angle brackets (*role*).

cmd_t: *Write acceptance criteria for a given user story as a $\langle role \rangle$.*

Our findings demonstrate a high level of semantic similarity among the acceptance criteria generated by different roles for the overall US-AC. Specifically, the similarity values are 0.9748 for RA vs. QA, 0.9751 for RA vs. others, and 0.9762 for QA vs. Others. Despite the pronounced similarity observed, it is noteworthy that each role group contributes distinct nuances based on its respective expertise. To evaluate the significance of such nuances, our researchers conducted a meticulous examination of the newly introduced contents. The investigation indicates that 72.22% of the additions from QA and 61.11% of those from Others were deemed worthy of further discussion.

Figure 5.1 presents an example. The user story delineates the researcher's desire to insert Greek symbols into the logbook entries. The acceptance criteria, denoted as ACs_1, ACs_2, and ACs_3, are generated by the LLM acting in roles as RA, QA, and Others, respectively. The bolded terms accentuate key phrases within each item. The three sets of outputs share substantial information, indicating a high similarity. The acceptance criteria produced by RA predominantly emphasize functional descriptions, including the basic operational flow on the insertion and specific consideration on performance and compatibility, as highlighted in blue. Furthermore, the outputs generated by LLMs acting as QA and Others introduce new criteria, distinguished by underlining and color-coding, with green indicating topics worthy of discussion

Chapter 5. SimAC: Requirement Elicitation in Agile RE

in agile practice and red denoting less pertinent aspects. In this case, Points 6 and 7 within ACs_2 offer valuable QA insights into test scenarios, encompassing the edit options and the manual insertion methods. Additionally, Point 3 in ACs_3 offers analogous information to Point 6 in ACs_1 while supplementing nuanced details, such as examples of more commonly used symbols. These details are more pertinent from the developers' standpoint, relating to implementation specifics. Such role-specific information contributes to meaningful collaboration within the GAC.

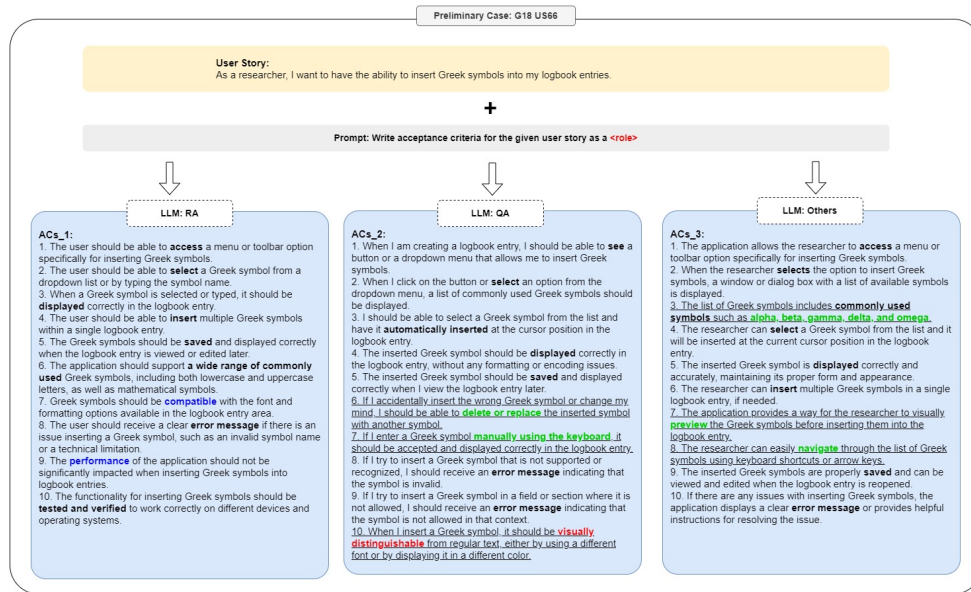


Fig. 5.1 Acceptance criteria generated by LLM acting in different roles.

The observed high similarity suggests that there is no necessity to create separate sets of US-AC for each role group. Instead, it is advisable to modify the acceptance criteria when specific updates are required, aligning with actual agile collaboration. The significance of the newly introduced nuances underscores the necessity for collaboration among diverse role groups because such additions yield meaningful discussion points. Consequently, we introduce a collaborative procedure as *create-update-update* to our proposed framework.

The observed high similarity suggests that there is no necessity to create separate sets of US-AC for each role group. Instead, it is advisable to modify the acceptance criteria when specific updates are required, aligning with actual agile collaboration. The significance of the newly introduced nuances underscores the necessity for collaboration among diverse role groups because such additions yield

meaningful discussion points. Consequently, we introduce a collaborative procedure as *create-update-update* to our proposed framework.

5.3 SimAC

Inspired by the insights derived from agile practice, we propose SimAC, which is a collaboration-simulation framework to empower LLMs in the task of GAC. This section presents the problem formulation and detailed design of our proposed framework.

5.3.1 Problem Formulation

Our study aims to leverage LLMs to generate acceptance criteria, thereby extending and defining the “done” of the story. More formally, given the user story x and the designated procedures f leveraging the LLMs, our objective is to formulate a series of prompts p introduced to the LLM that yields an estimation of US-AC, namely $\hat{y}$, which is expected to approximate the human-satisfied US-AC y . Subsequently, the generated $\hat{y}$ will be delivered for agile sprint planning discussion. The problem can be formulated as follows.

$$\arg \min_p ||y - \hat{y}||, \text{ where } \hat{y} = f(x, p) \quad (5.1)$$

5.3.2 Overview of SimAC

As shown in Figure 5.2, to simulate the inherent collaborative process of GAC in agile practice, we design SimAC, a two-phase framework, including the collaboration phase and the deduplication phase. SimAC instructs the LLM to act as different role groups, each tasked with distinct responsibilities aligned with the practices in the dashed frame of Figure 2.2. During the collaboration phase, the LLM sequentially performs different role groups to create or update the acceptance criteria for the given user story. These acceptance criteria are then processed in the deduplication phase, which aims at eliminating redundant information.

5.3.3 Phase I: Collaboration

The collaboration phase comprises two stages. Firstly, the LLM is initiated by role-based prompts with demonstrated examples. Following this, the LLM acts as the designated role group for GAC.

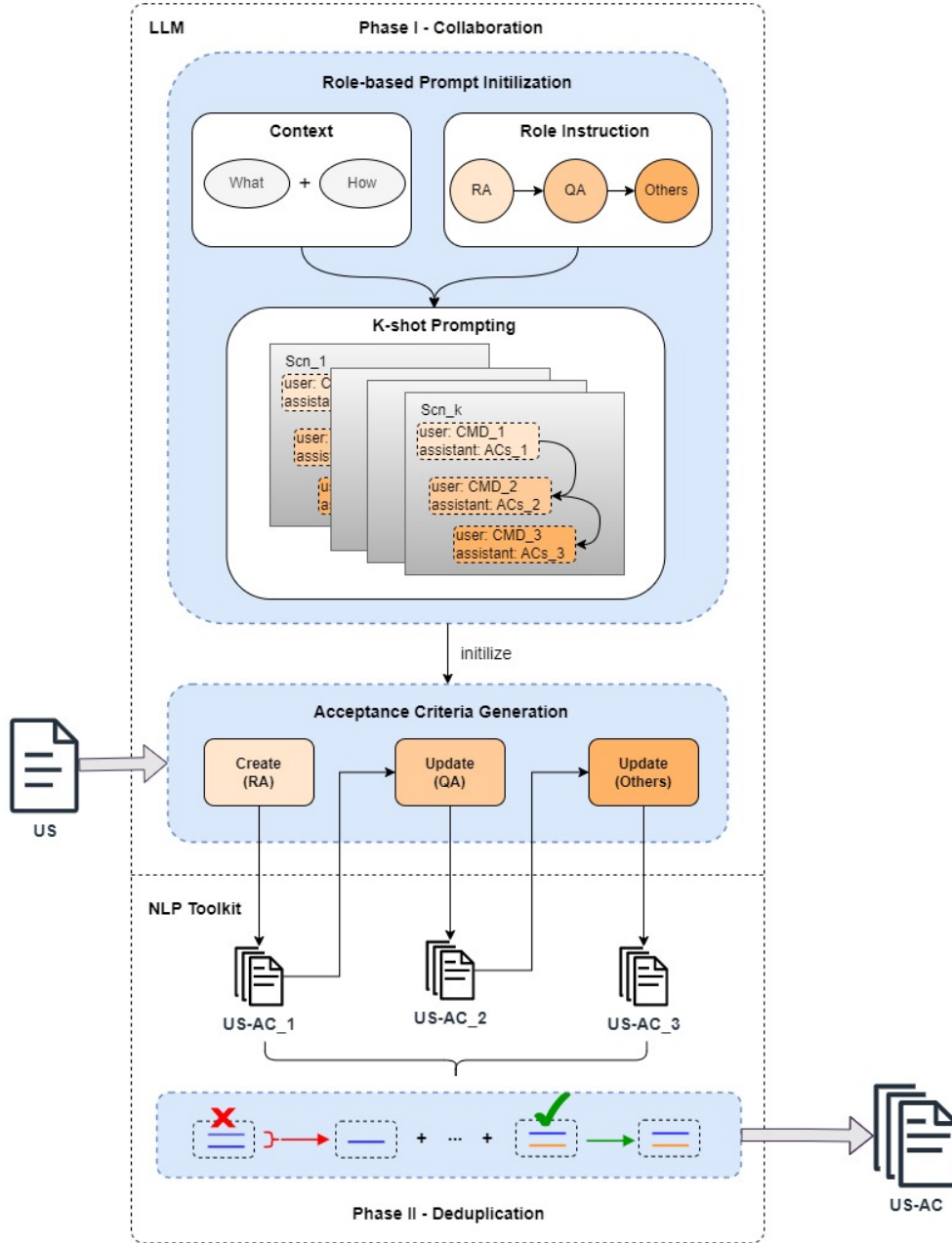


Fig. 5.2 Two phases of GAC activities: I. collaboration (Stage I: role-based prompt initialization; Stage II: acceptance criteria generation); II deduplication.

Role-based Prompt Initialization

Figure 5.3 presents the detailed role-based prompt (P_{ro}) tailored for SimAC. This prompt encompasses three essential components: context, role instruction, and K-shot prompting.

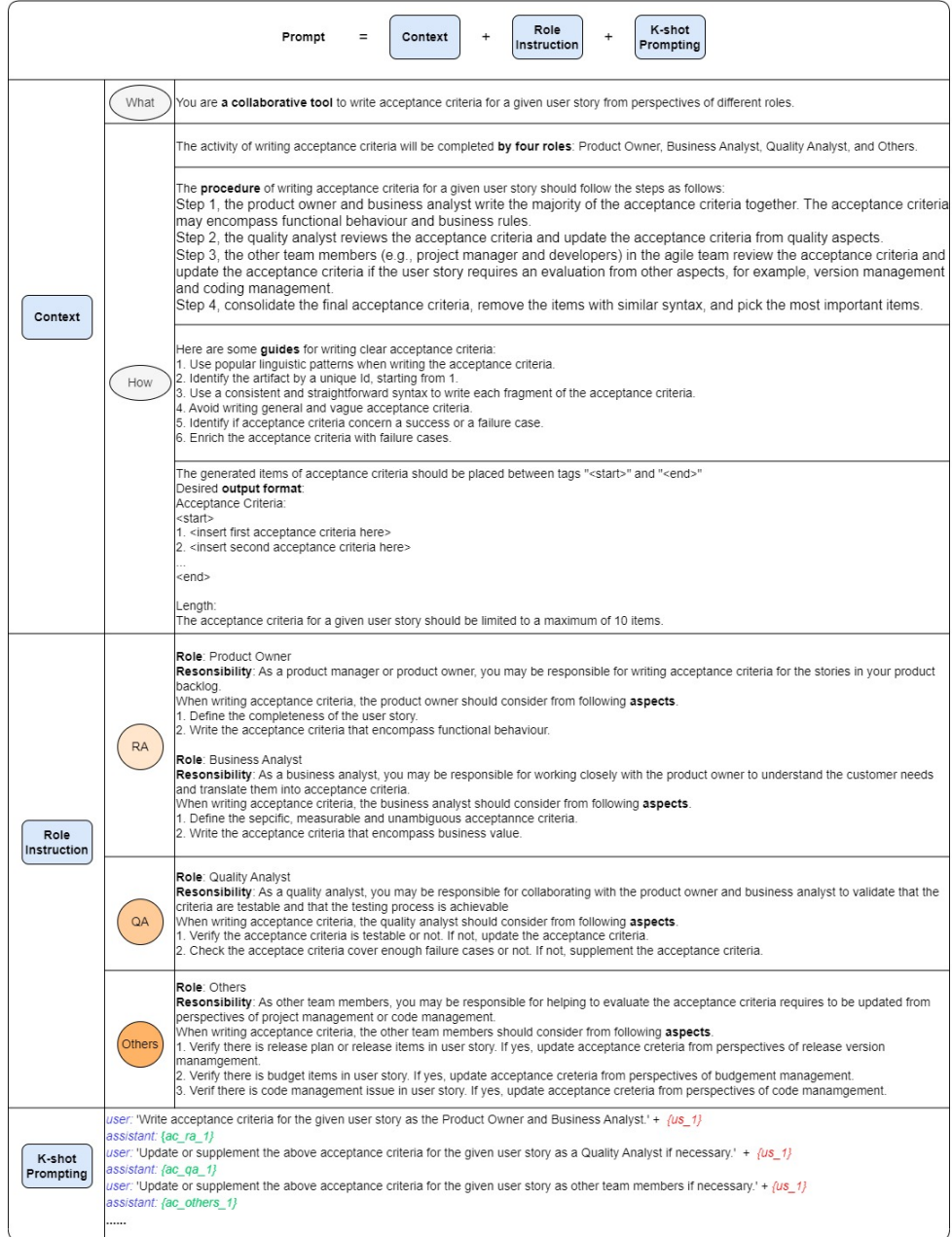


Fig. 5.3 Three components in the role-based prompt (P_{ro}): context, role instructions, and K-shot prompting.

Context In order to facilitate and enhance collaborative efficiency, the definition of GAC is specified in the context of the prompt. As dictated in the prompt, the LLM is designated as a collaborative tool to generate acceptance criteria for the given

user story. Following the good practices of prompt guided by OpenAI¹, the context encompasses four facets aimed at instructing the LLM on its behavior in GAC: roles engaged in the activity, procedures of GAC, guidelines of writing clear acceptance criteria, and output formats.

As previously discussed, to simulate the agile practice as highlighted with the dashed frame in Figure 2.2, the LLM is initially instructed to act as three distinct role groups: RA, QA, and others. Subsequently, a procedural workflow is introduced, which aligns with a sequence of activities: creation by RA, update by QA, and second-round update by Others. In accordance with the recommendation proposed in [72], the guidelines for writing clear acceptance criteria are introduced in the context. Finally, the designed output format is specified, which is a mechanism aimed at facilitating subsequent data extraction and processing endeavors.

Role Instruction The prompt employed to initiate the LLMs primarily consists of comprehensive role instructions, which share the knowledge of role-based collaborations in practice. For each participating role, the responsibilities and aspects that need to be considered in the activity are introduced to the LLM.

The expertise of each role engaged in the activity is emphasized so that the LLM can receive a better understanding and act as different experts. As discussed in role division, three groups of roles are introduced. Following the responsibility of different role groups in the agile team discussed in [181], we deliver the detailed instructions to different role groups as follows.

- **Requirement Analyst (RA)**
 - **Product Owner (PO)** should ensure the completeness of the user story with product functionalities.
 - **Business Analyst (BA)** is responsible for presenting the business value in the user stories.
- **Quality Analyst (QA)** should verify whether the acceptance criteria are testable and ensure the acceptance criteria should cover as many different case scenarios as possible.
- **Other Team Members (Others)**

¹<https://help.openai.com/en/articles/6654000-best-practices-for-prompt-engineering-with-openai-api>

- **Developer** (Dev) should assist with considering the implementation details to be measurable or not.
- **Project Manager** (PM) takes care of the release schedule and budget management-related stuff.

K-shot Prompting As illustrated in [184], GPT-3 exhibits promising results in few-shot settings in some NLP tasks. Illustrative examples are integrated into prompts to enhance the comprehensive understanding of collaborative role activities. Following the collaborative work process, we incorporate K-shot promptings. In each shot, the LLM acts in accordance with the prescribed procedure. The prompt examples replicate different scenarios involving different roles in the activity. Further details regarding the experimental settings of the examples will be discussed in Section 5.4.

Acceptance Criteria Generation

Following initialization with the role-based prompt, the LLM adheres to predefined procedures, namely *create-update-update*, to execute the task of GAC using the designated command for each step. Given that the RA serves as the umbrella name within SimAC, the LLM may lack a comprehensive understanding of its meaning. To mitigate this potential ambiguity, we incorporate the roles of PO and BA in the command introduced to the LLM. The commands for each step are presented as follows.

cmd_1 (command for creation): *Write acceptance criteria for the given user story as the Product Owner and Business Analyst.*

cmd_2 (command for 1st update): *Update or supplement the above acceptance criteria for the given user story as a Quality Analyst if necessary.*

cmd_3 (command for 2nd update): *Update or supplement the above acceptance criteria for the given user story as other team members if necessary.*

Figure 5.4 demonstrates the step-by-step prompt with commands (exemplified in *gpt-3.5-turbo*). Initially, the LLM acts as the role of RA to generate a set of acceptance criteria for the given user story. Subsequently, the LLM transitions to the role of QA to update the acceptance criteria. Finally, the LLM assumes the role of Others,

contributing to supplement the US-AC with additional details from their own expertise. The abbreviations used in Figure 5.4 are elucidated as follows.

- **<context>** - the context component in the prompt (as presented in Figure 5.3).
- **<role>** - the role instruction component demonstrated in prompt (as presented in Figure 5.3).
- **cmd** - the commands for each step (as listed in the previous discussion).
- **{us}** and **{ac}** - the user stories and the exemplared acceptance criteria generated by different roles as demonstrated in examples (as presented in Figure 5.3)
- **{US}** and **{AC}** refer to the given user story and the acceptance criteria generated from each step.

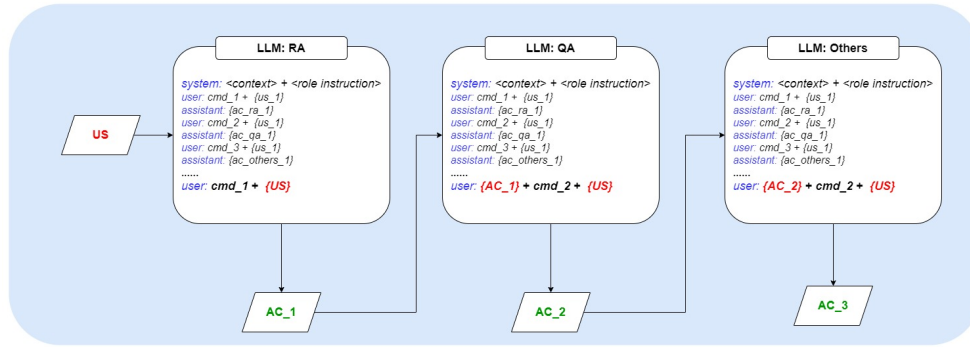


Fig. 5.4 Collaborative roles-playing LLM in three-steps of US-AC elaboration (exemplified in *gpt-3.5-turbo*).

5.3.4 Phase II: Deduplication

Throughout the collaboration process, each procedural step yields its own set of acceptance criteria. Nevertheless, different roles may introduce modifications to the US-AC that convey similar meanings. To alleviate redundancy, SimAC employs a semantic similarity processing mechanism to scrutinize and refine the US-AC prior to its finalization.

In the deduplication phase, the acceptance criteria extracted from the three sets of US-AC are arranged in pairs to calculate similarity scores. Once the scores surpass a specified threshold, one of the acceptance criteria in this pair will be excluded to eliminate redundancy.

Cosin similarity (c_{sim}) is a widely implemented metric for measuring similarity between sentences [185]. The sentences are converted into vectors based on TF-IDF. The c_{sim} between two sentence vectors (S_A and S_B) is computed by taking the dot product of the vectors and dividing it by the product of their magnitudes. Mathematically, it is defined as:

$$c_{sim} = \frac{S_A \cdot S_B}{\|S_A\| \|S_B\|} \quad (5.2)$$

5.4 Experimental Setup

In this section, we outline the experimental setup of this study, including RQs with corresponding experimental design, dataset, baselines, implementation, and evaluation employed.

5.4.1 Research Questions

Here, we introduce four RQs guiding this study and outline the respective experimental designs to address them.

RQ1: How is the generality of SimAC employed in different LLMs?

This RQ aims to assess the effectiveness of SimAC and its generality on various LLMs. In this study, different LLMs of diverse sizes are selected for investigation, including *gpt-3.5-turbo*, *text-davinci-003*, *OpenLLaMA-7B*, and *OpenLLaMA-13B* as further introduced in Section 5.4.3. Through this examination, we are able to quantitatively assess the generalization capability of SimAC within the domain of GAC and explore the feasibility of automating this task.

In SimAC, the LLM is prompted by P_{ro} (as presented in Figure 5.3) to simulate collaborative interactions. To facilitate the comparative analysis, we meticulously designed a regular prompt (P_{re}), which requests the LLM to generate acceptance criteria directly. The P_{re} excludes the role instruction, which only introduces the context and good practices. The structure and details of the regular prompt template are outlined in Figure 5.5. The command introduced to the LLM initiated by the regular prompt is shown below:

cmd: *Write acceptance criteria for the given user story.*

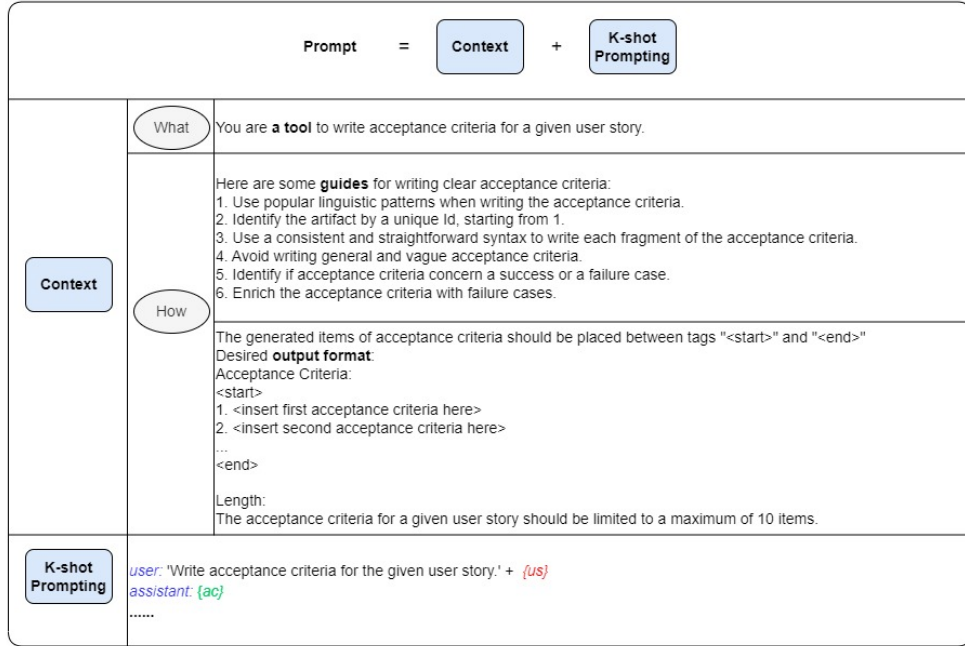


Fig. 5.5 Two components in the regular prompt (P_{re}): context and K-shot prompting.

RQ2: What is the contribution of each role group in SimAC?

SimAC incorporates the collaborative efforts of three distinct role groups in GAC, following agile practices. Each role group brings specific expertise to the GAC process. In RQ2, we aim to discern the individual contributions of these role groups.

To achieve this, we deconstruct the role groups as three combinations (denoted as R): **1)** RA (i.e., $R = 1$), **2)** RA and QA (i.e., $R = 2$), and **3)** RA, QA, and Others (i.e., $R = 3$). The assessment of each combination enables a comprehensive analysis to evaluate the impact of each role group contribution within the SimAC.

RQ3: How do K-shot promptings impact the performance of SimAC?

Within SimAC, a variety of examples are integrated into prompts to guide the LLM in GAC. While adding more examples offers increased reference, it also consumes valid prompt capacity and may lead to redundancy. Thus, RQ3 targets to explore the impact of K-shot prompting in SimAC.

Due to the maximum token constraint of LLMs' input, we designate the number of examples in K-shot prompting as **0**, **1**, **3**, and **4**. Here, K denotes the number of sets of interactive examples covered. Each set of examples adheres to the *create-update-update* process, comprising three distinct examples. The examples were

crafted by the invited practitioners based on their professional expertise. In order to bolster the resilience of LLMs, we deliberately selected some common examples demonstrated in online resources, avoiding reliance on D .

Table 5.1 provides an overview of the examples utilized, demonstrating varying engagement levels from different roles. Drawing from observations in daily practice, some roles tend to make minor modifications to US-AC due to a diminished necessity required from them. Following the *create-update-update* process in GAC, the predominant scenario (sc_1) entails all role groups participating equally in creating or updating the US-AC, as evident from the modification of ac_1 detailed in Table 5.1. The second scenario (sc_2) involves updates contributed by both QA and Others, albeit with minor alterations to the US-AC. As sc_2 delineated in Table 5.1, the RA creates ac_2 , while QA and Others slightly update on ac_2 . Furthermore, the third scenario (sc_3) exclusively involves participation from RA and QA, without contributions from Others. The sc_3 outlined Table 5.1 illustrates ac_3 is created by RA and slightly updated by QA, with no involvement from Others.

In **zero-shot prompting** ($K = 0$), no collaborative examples are provided to the LLM. Only the context and role instructions are presented. For **one-shot prompting** ($K = 1$), a single set of interactive examples is introduced to the LLM, reflecting the most common scenario (sc_1). In **three-shot prompting** ($K = 3$), three sets of examples covering sc_1 , sc_2 , and sc_3 are provided. However, due to token constraints, **four-shot prompting** ($K = 4$) cannot include six sets of examples covering all three scenarios. Therefore, our focus is exclusively on sc_1 and sc_2 , as they are the most commonly occurring scenarios observed in daily practice.

RQ4: How do different settings of the similarity threshold impact the performance of SimAC?

In this RQ, we aim to identify the optimal threshold (t) for reducing the redundancy of generated US-AC. We perform a pairwise comparison on US-AC generated in D_p . The results reveal the sentence similarity scores fall within a range spanning from 0.85 to 0.95. Within this range, we designate the lower, middle, and upper bounds as potential threshold values, where t is set as **0.85**, **0.9**, and **0.95** in this experiment.

5.4.2 Dataset

To assess the effectiveness of SimAC, a publicly available dataset (D) as outlined in [33] is utilized in our study. Three projects are selected to construct our test

Table 5.1 Description of examples in K-shot prompting generated under different scenarios

K	Scenarios (#)	Examples (#)	Type	Collaboration in GAC
0	0	0	-	-
1	1	3	sc_1	E1.1: RA creates ac_1 for us_1 . E1.2: QA updates on ac_1 for us_1 . E1.3: Others update on ac_1 for us_1 .
3	3	9	sc_1	E1.1: RA creates ac_1 for us_1 . E1.2: QA updates on ac_1 for us_1 . E1.3: Others update on ac_1 for us_1 .
			sc_2	E2.1: RA creates ac_2 for us_2 . E2.2: QA slightly updates on ac_2 for us_2 . E2.3: Others slightly update on ac_2 for us_2 .
			sc_3	E3.1: RA creates ac_3 for us_3 . E3.2: QA slightly updates on ac_3 for us_3 . E3.3: Others do not update on ac_3 for us_3 .
4	2	12	sc_1	E1.1: RA creates ac_1 for us_1 . E1.2: QA updates on ac_1 for us_1 . E1.3: Others update on ac_1 for us_1 .
			sc_2	E2.1: RA creates ac_2 for us_2 . E2.2: QA slightly updates on ac_2 for us_2 . E2.3: Others slightly update on ac_2 for us_2 .
			sc_1	E4.1: RA creates ac_4 for us_4 . E4.2: QA updates on ac_4 for us_4 . E4.3: Others update on ac_4 for us_4 .
			sc_2	E5.1: RA creates ac_5 for us_5 . E5.2: QA slightly updates on ac_5 for us_5 . E5.3: Others slightly update on ac_5 for us_5 .

sc_1 refers to 1st type of scenario

us_1 refers to User Story 1

ac_1 refers to acceptance criteria attached for us_1

E1.1 refers to the 1st version of ac_1 as illustrated in the example

E1.2 refers to the 2nd version of ac_1 as illustrated in the example

5.4 Experimental Setup

dataset (D_t) containing different sizes of actors ($\langle \text{type of user} \rangle$), thereby encompassing anticipated requirements from diverse stakeholders. Project NSF (G11) incorporates multiple actors within its user stories, delineating user needs from both business flow and development progression perspectives. Project UniBath (G24) entails user stories with medium-sized actors predominantly focusing on feature-based descriptions. Conversely, Project Zooniverse (G28) comprises user stories with only a few actors, primarily emphasizing the user flow of both administrative and regular users. These three projects offer representative examples encompassing diverse levels of user story complexity for the purposes of our experimental evaluations. Table 5.2 presents the details of the selected projects. The random samples refer to the user stories being randomly picked for final evaluation.

Table 5.2 Details of selected projects in testing dataset D_t

ID	Acronym	Project Description	US (#)	Actors (#)	Random Samples
G11	NSF	New version of the NSF website: redesign and content discover	73	22	[30, 63, 53, 37, 26, 57, 65, 2, 40, 58]
G24	UniBath	Institutional data repository for the University of Bath	53	11	[50, 27, 7, 25, 11, 26, 16, 21, 8, 1]
G28	Zooniverse	Citizen science platform that allows anyone to help in research tasks	60	3	[6, 49, 5, 20, 25, 50, 17, 3, 15, 21]

5.4.3 Baselines

Considering GAC is a text generation task, baselines are selected from casual language models. The selected models encompass various LLM families, including GPT families and LLaMA families, spanning a range of parameter sizes (7B, 13B, and over 175B) and originating from diverse sources (both closed-source and open-source).

GPT-3.5 family The GPT-3.5 series models are trained with over 175B parameters. Among the GPT-3.5 families, *gpt-3.5-turbo* is a chat text completion model, which allows the roles playing in chat interaction. It is the most capable and cost-effective model in the GPT-3.5 family. Furthermore, *gpt-3.5-turbo* was trained in both natural languages as well as codes, which is suitable for comprehending software development processes. Therefore, it has been selected to assess the efficacy of SimAC.

Within the GPT-3.5 family, *text-davinci-003* is recognized for its versatility in handling a wide array of NLP-related tasks with higher quality. As part of our comparative analysis, it is selected as another baseline. This allows us to assess the performance of SimAC with an LLM trained in pure natural language.

Open-LLaMA LLaMA [186] is competitive with the best models developed by Meta AI². OpenLLaMA [187] is a permissively licensed open-source reproduction of LLaMA. It is published under the business-friendly Apache 2.0 license. It achieves a similar performance as LLaMA, and it is free to use for commercial purposes without any restrictions. OpenLLaMA offers a range of models, including 3B, 7B, and 13B, trained on 1T tokens. In our experiment, *OpenLLaMA-7B* and *OpenLLaMA-13B* models are utilized to evaluate the generality of SimAC across LLMs with different parameter sizes.

5.4.4 Implementation

Regarding LLMs, we implement GPT-3.5 by invoking OpenAI's API³. According to the guidelines, the temperature values vary between 0 and 2. Higher values increase the randomness of the output, while lower values will make the results more focused and deterministic. Therefore, the default *temperature* = 1 is adopted to achieve moderate results. For open-source LLMs, such as OpenLLaMA, we instantiate them with their replication packages and load their weights from HuggingFace⁴. Additionally, the parameter *max_new_tokens* is configured to 500 in the OpenLLaMA series, which is deemed an appropriate length for the newly generated acceptance criteria.

Towards SimAC, the settings of its backbone LLMs are kept as default. In the collaboration phase, the role-based prompt (P_{ro}) is constructed as Figure 5.3 to initialize the LLMs. Additionally, Spacy⁵ is employed to compute the cosine similarity in the deduplication phase. Considering the diversity of text background generated, the vector package *en_core_web_lg* is adopted for calculating the similarity score. The package is trained with about 500k word vectors, whereas the calculation of US-AC similarity can benefit from a large vocabulary.

²<https://ai.meta.com/>

³ <https://platform.openai.com/docs/models/gpt-3-5>

⁴ <https://huggingface.co/openlm-research>

⁵<https://spacy.io/usage/linguistic-features>

5.4.5 Evaluation

We employ both quantitative and qualitative methods to evaluate the proposed SimAC framework. Quantitatively, the evaluation covers quality and relevance metrics. Qualitatively, specific cases are selected for an in-depth examination to assess the effectiveness of the SimAC framework.

Metrics

Quality In agile practice, the product backlog items are commonly represented as user stories with the extension of acceptance criteria [29]. Following the principles behind the Agile manifesto [151], the most well-known quality evaluation framework for product backlog items is the six attributes method called **INVEST** [188, 34, 75, 189]. The metrics, introduced by Bill Wake⁶, stands for **I**ndependent-**N**egotiable-**V**aluable-**E**stimable-**S**mall-**T**estable. The survey in [189] reveals that a majority of practitioners leverage the INVEST framework as quality guidance when crafting agile product backlog items. Recognizing the absence of formalized evaluation metrics specifically tailored to US-AC and the significant relevance of user story and acceptance criteria introduced in Section 2.3.2, we follow the widely acknowledged agile product backlog item evaluation standards, namely INVEST, for assessment in this study. Here, we integrate the explanation introduced in INVEST⁶ and the measurement details discussed in [190] to extend every attribute for evaluating the US-AC as follows.

- **Independent:** The US-AC should be as independent as possible. It can be implemented and delivered without being dependent on other stories.
- **Negotiable:** The US-AC is not an explicit contract and should leave space for discussion. It can be rewritten depending on business, market, technical, or any other type of requirement by team members.
- **Valuable:** The US-AC must deliver value to the stakeholders by clearly specifying solutions.
- **Estimable:** The US-AC should be estimable. It is well enough understood by team members to be able to determine its relative size (in development and cost).
- **Small:** The US-AC should not be so big as to become impossible to plan/task/prioritize within a level of accuracy.

⁶<https://xp123.com/articles/invest-in-good-stories-and-smart-tasks/>

- **Testable:** The US-AC must provide the necessary information to make test development possible.

Halme et al. [191] operationalized the INVEST as binary evaluation. That is, the evaluators in [191] consider whether a user story fulfilled the requirements of an INVEST attribute, scoring 0 or 1. Lombriser et al. [192] employed a five-score scale (1 to 5) to evaluate the quality of requirements (i.e., user stories and acceptance test) generated from gamification practices. Compared with binary and Likert-scale evaluation mechanisms, the latter method offers a more comprehensive understanding of the contents. Therefore, we adopted Likert-type questionnaires to explore the INVEST mnemonic for the US-AC. Each attribute of INVEST was evaluated using expert opinions on a 5-point Likert scale, where scores of 0-4 represent the scale of *poorly*, *unsatisfactorily*, *reasonably*, *satisfactorily*, and *very satisfactorily*.

For each individual attribute of INVEST, we calculated the mean (m) and standard deviation (sd). The m provides an assessment of the average score derived from the evaluated samples, indicating the central tendency of the measurements. Conversely, the sd quantifies the extent of dispersion in the observations, elucidating the divergence of individual data points from the mean value. The cumulative means (Avg.) of all the INVEST attributes are subsequently averaged, collectively offering a comprehensive indicator of the overall satisfaction level on the quality of US-AC.

Relevance To support agile RE effectively, the automatically generated US-AC must achieve a level of relevance that is as good as that achieved by a human practitioner. The primary metrics commonly employed to assess the relevance of generated results are **completeness** and **validity**. Given the absence of ground truth in the dataset, employing a gold standard as a benchmark for evaluation is considered appropriate under reasonable conditions [193]. In a study focusing on deriving conceptual UML diagrams from user stories, Bragilovski et al. [36] evaluated the completeness and validity of the student-generated diagrams against the gold standard crafted by professionals. Similarly, for generating candidate testable references for given requirements, Peng et al. [194] assessed the precision and recall by examining the automatic abstraction matching with the manually created assumptions. Following these practices, two practitioners were initially invited to craft a gold standard for a given user story within D_t . The practitioners were encouraged to thoroughly examine the project background and the referenced products in order to produce the gold standard as convincingly as possible. Subsequently, the relevance of US-AC generated

by SimAC was evaluated through manual comparison with the human-crafted gold standard.

Considering the potentially many-to-many mapping relationship between the generated acceptance criteria items in the gold standard and the SimAC-generated US-AC, we divided the true positives into two categories. True positives in the SimAC solution (TP_s) signify the number of instances present in the US-AC generated by the SimAC that also aligns with the instances in the gold standard. Conversely, false positive (FP) denotes the count of instances in the SimAC-generated US-AC that do not correspond to instances within the gold standard. The metric *validity* (val) is defined as the ratio of TP_s to the total instances in the LLM-based solution ($TP_s + FP$). In information retrieval terms, validity equates to precision.

$$val = \frac{|TP_s|}{|TP_s| + |FP|} \quad (5.3)$$

True positives in the gold standard (TP_g) refer to instances present in the gold standard that correspond to instances generated in the SimAC solution. The term false negative (FN) denotes the count of instances that exist in the gold standard but cannot be identified in the automatically generated US-AC. The metric *completeness* (com) is defined as the ratio between TP_g and the number of instances in the gold standard ($TP_g + FN$). In information retrieval terms, completeness equates to recall.

$$com = \frac{|TP_g|}{|TP_g| + |FN|} \quad (5.4)$$

Methodologies

Practitioner Engagement Agile RE, in particular, heavily relies on domain expertise, necessitating increased practitioner engagement for comprehensive evaluation. In our study, we actively involved industrial practitioners in various aspects, including scoring the INVEST, crafting the gold standard, and calibrating true positives mapping. In line with the methodology adopted in [73], which involves industrial experts to label the user story, we extended invitations to eight well-trained agile practitioners from different IT software companies to participate in the evaluation process. The backgrounds of these evaluators are presented in Table 5.3, demonstrating their extensive experience. Each of them has more than four years of experience in agile practices in the development team.

The quality evaluation is divided into five groups of solutions: role combination, similarity threshold, K-shot prompting, GPT-3.5 baselines, and OpenLLaMA baselines. Each evaluation group includes two evaluators from distinct roles to ensure comprehensive evaluations. This approach facilitates the collection of insights from a variety of role perspectives, allowing for an in-depth analysis.

Considering the workload of human evaluation and the time restrictions imposed by practitioner evaluations, the evaluation sample size was intentionally diminished to facilitate a more meticulous and profound analysis. We randomly picked and delivered ten comparative groups of US-AC from each selected project. In total, we delivered 30 sets of US-AC for each evaluation group.

To mitigate bias arising from the distribution of roles within various evaluation groups, it was stipulated that one of the evaluators in each group must be an RA, who is the primary participant in requirement activities. Moreover, two of the authors meticulously reviewed the scores across all evaluation groups. In cases where groups were either underestimated or overestimated, evaluators were convened for discussion to ensure alignment in estimation levels.

Table 5.3 Description of the practitioners included in this study

Evaluator	Working Experience	Agile Experience	Role	Engagement in Evaluation Group
P1	13 years	8 years	QA	Baselines (OpenLLaMA)
P2	15 years	6 years	PM	Role Combination
P3	15 years	8 years	PM	Similarity Threshold
P4	8 years	5 years	BA	Baselines (OpenLLaMA, GPT-3.5)
P5	6 years	4 years	PO	Role Combination
P6	9 years	5 years	Dev	Baselines (GPT-3.5)
P7	10 years	8 years	BA	K-shot Prompting, Similarity Threshold
P8	10 years	6 years	QA	K-shot Prompting

Questionnaire: INVEST Scoring The quality evaluation was conducted using a questionnaire method. Prior to the evaluation, evaluators were provided with instructions within the questionnaire package and participated in an online demonstration. Appendix 5.A presents the instructions delivered for evaluation, while Appendix 5.B furnishes a sample of a group of US-AC that needs to be assessed in the questionnaire. In accordance with the protocol, each set of questionnaires was allocated

a three-week window for evaluators to score the US-AC with a comprehensive and rigorous analysis.

The sequence in which outputs from different models are presented for review can impact the judgments made by evaluators. To minimize potential bias in the evaluation process, evaluators were directed to conduct a comprehensive review of the entire evaluation corpus to ensure accurate scoring.

To mitigate the inherent subjectivity associated with the evaluations conducted by two raters, an in-depth analysis was undertaken to assess the degree of congruence between their assigned scores. For each metric in INVEST, the difference in scores assigned by the two evaluators was calculated for each instance. A significant deviation exceeding a threshold of 2 points was considered indicative of a substantial inconsistency in their assessments, particularly concerning satisfaction levels. Subsequently, to enhance the reliability of the evaluation, a third expert was enlisted to assess the specific instance independently. The third expert was selected from the pool of experts (i.e., P1 to P8), excluding individuals whose roles were already involved in the current evaluation group. Finally, the two scores demonstrating the least disparity among all three assessments were identified and used to calculate the final average score.

The inter-rater agreement between two evaluators is measured by Cohen's Kappa [195]. Following the established categorization scheme proposed by [196], the values of the kappa coefficient are categorized into specific ranges, denoting the degree of agreement. Specifically, kappa scores falling within the intervals of 0.21 to 0.40, 0.41 to 0.60, and 0.61 to 0.80 are interpreted as indicative of *fair*, *moderate*, and *substantial* agreement, respectively. For each evaluation group, the level of inter-rater agreement is reviewed. Instances where the agreement falls below the threshold for fair agreement are subjected to discussion, and scores are adjusted accordingly to achieve agreement.

Manual Matching: Gold Standard The gold standard was collaboratively developed by three invited practitioners from different roles (i.e., P7, P2, P8), drawing upon their comprehensive comprehension of the entire project. Leveraging their comprehensive understanding of the entire project, the practitioners engaged in a multi-step process.

Initially, they familiarized themselves with the background of the selected projects by thoroughly analyzing all user stories to form a conceptual understanding of the intended product. Subsequently, they delved into understanding the referenced product of each requirement. For Project NSF, practitioners utilized an available online

repository package⁷ to establish a local environment and comprehend the business flow. For Project UniBath, the researcher showcased the data repository business flow of our university to enhance practitioners' understanding. In the case of Project Zooniverse, practitioners registered and interacted with the platform⁸ to grasp its functional flows. Drawing from this comprehensive understanding, the practitioners collaborated in drafting acceptance criteria. Finally, they engaged in discussions to distill key acceptance criteria gleaned from their background analysis and experience, culminating in the production of a reconciled version, i.e., the gold standard, which served as a benchmark for the relevance evaluation.

In the relevance evaluation process, two of our authors, who are proficient in requirement analysis, compared the auto-generated US-AC with the gold standard. Instances displaying similar meanings to those outlined in the gold standard were categorized as TP_s and TP_g . In cases where instances exhibited ambiguous connotations differing from those in the gold standard, our authors sought suggestions from practitioners to facilitate evaluation and discussion, thereby ensuring informed decisions regarding their classification as true positives or otherwise.

Statistical Analysis

To investigate whether the improvement of SimAC against each backbone is statistically significant, we perform the t -test [197] with a confidence level of 95% to make pairwise comparisons in all experiments. If the obtained p -value is less than 0.05, there is a statistically significant difference between the two methods. While a p -value value can inform whether an effect exists, it cannot reveal the size of the effect [197]. Subsequently, we further conduct Cohen's d (a standardized mean difference) analysis [198] to measure the effect size, which is measured by the standardized difference between two means. Following the classification in [197], the effect size can be classified as *small* ($0.2 \leq d < 0.5$), *medium* ($0.5 \leq d < 0.8$), and *large* ($0.8 \leq d$). According to Cohen, a medium effect indicates the difference is visible to the naked eye of a careful observer, while a small size presents a limited effect [199]. A large effect size means that the result has meaningful practical significance. The combination of t -test and Cohen's d analysis offers us an in-depth review of the improvement on different evaluation metrics.

⁷<https://github.com/nsf-open/nsf/tree/master>

⁸<https://blog.zooniverse.org/>

5.5 Experimental Results

In this section, we present the experimental results to answer the RQs.

5.5.1 RQ1: Generality of SimAC

Quality Table 5.4 presents the quality evaluation results of SimAC (implemented with P_{ro}) compared to its different underlying LLM baselines (utilized with P_{re}), revealing a notable improvement in satisfaction (s) with the quality of the US-AC generated by SimAC. It is evident that the GPT-3.5 series has already achieved a commendable level of performance, rating as *satisfactorily* ($s > 2$). When collaboratively embedded with the SimAC, these LLMs attain a higher level of satisfaction categorized as *very satisfactorily* ($s > 3$). Particularly, *gpt-3.5-turbo* initialized by P_{ro} consistently performs the best, obtaining the highest average score of 3.21. Notably, such integration leads to a rating of *very satisfactorily* ($s > 3$) across all other INVEST metrics, except for S . In the case of OpenLLaMA family models, the US-AC generated by P_{re} falls into the category of *poorly* ($s < 1$). Consistently, the utilization of P_{ro} does yield some improvement, albeit it remains within the realm of *unsatisfactorily* ($1 < s < 2$).

To examine whether the improvement of SimAC against each backbone LLM is statistically significant, we perform statistical analysis combining $t - test$ and Cohen’s d as discussed in Section 5.4.5. The results presented in Table 5.5 illustrate the improvements are statistically significant and non-trivial across various metrics, specifically in N , V , E , and T . In the case of I , a significant improvement is evident within three out of the evaluated baselines, except for *OpenLLaMA-13B*. Furthermore, the analysis in S reveals a constrained enhancement, as evidenced by p -value and Cohen’s d observed in *gpt-3.5-turbo* and *OpenLLaMA-7B*.

Considering the agreement between two evaluators, Table 5.6 illustrates that all of the assessment outcomes reside within the categories of *fair* and *moderate*. Such agreement adds credibility to the human evaluation results. When comparing the scores from evaluators of different roles, no substantial discrepancy is discerned, with an average difference of less than 0.2 observed for the GPT-3.5 series. The discrepancies are even smaller when assessing the OpenLLaMA series, with an average difference of less than 0.06.

Relevance Table 5.7 further proves the superior performance of SimAC in enhancing LLMs’ GAC capability in terms of completeness and validity. The results underscore

Table 5.4 Average satisfaction on INVEST score, comparative results on US-AC generated by different LLMs

LLM	Prompt	I		N		V		E		S		T		Avg.	
		m	sd	m	sd	m	sd	m	sd	m	sd	m	sd		
GPT-3.5	gpt-3.5-turbo	P_{r^e}	2.87	0.47	2.90	0.44	3.07	0.41	2.97	0.49	2.57	0.50	2.95	0.41	2.89
		P_{r^o}	3.10	0.44	3.13	0.50	3.65	0.48	3.40	0.62	2.67	0.54	3.30	0.62	3.21
	text-davinci-003	P_{r^e}	2.80	0.48	2.63	0.52	2.57	0.56	2.63	0.58	2.77	0.50	2.60	0.53	2.67
		P_{r^o}	3.00	0.52	2.83	0.49	2.98	0.43	3.08	0.53	3.05	0.57	3.08	0.67	3.01
OpenLLaMA	OpenLLaMA-7B	P_{r^e}	0.83	0.67	0.57	0.65	0.62	0.67	0.70	0.70	0.77	0.77	0.65	0.61	0.69
		P_{r^o}	1.12	0.76	0.97	0.78	1.07	0.90	1.05	0.85	0.88	0.76	1.10	0.84	1.03
	OpenLLaMA-13B	P_{r^e}	0.82	0.60	0.60	0.62	0.67	0.66	0.78	0.58	0.70	0.59	0.87	0.60	0.74
		P_{r^o}	1.15	0.68	1.10	0.75	1.13	0.87	1.15	0.76	1.03	0.74	1.15	0.86	1.12

m - mean
 sd - standard deviation
 P_{r^e} - regular prompt
 P_{r^o} - role-based prompt
 bold indicates better performance

5.5 Experimental Results

Table 5.5 Significance analysis of quality evaluation on different LLMs

p -value	P_{re} Versus P_{ro}					
Cohen's d	I	N	V	E	S	T
gpt-3.5-turbo	5.73e-03 +Medium	7.83e-03 +Small	6.82e-11 +Large	3.86e-05 +Medium	2.96e-01 +Small	6.63e-04 +Medium
text-davinci-003	3.07e-02 +Small	3.26e-02 +Small	1.32e-05 +Large	2.11e-05 +Large	4.33e-03 +Medium	2.52e-02 +Large
OpenLLaMA-7B	3.22e-02 +Small	2.78e-03 +Medium	2.32e-03 +Medium	1.52e-02 +Small	4.05e-01 +Small	1.01e-03 +Medium
OpenLLaMA-13B	5.25e-03 +Medium	1.18e-04 +Medium	1.23e-04 +Medium	3.57e-03 +Medium	7.17e-03 +Small	3.81e-02 +Small

p -value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of P_{ro} .

Table 5.6 Kappa score for assessing inter-rater agreements on the baseline evaluations

kappa (difference)	Pt	I	N	V	E	S	T	Avg.	aggr.
gpt-3.5-turbo	P_{re}	0.65 (-0.13)	0.22 (-0.20)	0.34 (-0.20)	0.33 (-0.27)	0.47 (-0.13)	0.37 (-0.23)	0.40 (-0.19)	Moderate
	P_{ro}	0.41 (-0.13)	0.37 (-0.13)	0.35 (-0.10)	0.25 (-0.27)	0.22 (0.20)	0.53 (0.00)	0.35 (-0.07)	Fair
text-davinci-003	P_{re}	0.51 (0.00)	0.38 (0.33)	0.41 (0.27)	0.26 (0.07)	0.40 (-0.07)	0.61 (0.13)	0.43 (0.12)	Moderate
	P_{ro}	0.69 (0.07)	0.36 (0.13)	0.48 (0.10)	0.48 (0.10)	0.47 (0.10)	0.61 (0.10)	0.51 (0.10)	Moderate
OpenLLaMA-7B	P_{re}	0.33 (-0.13)	0.41 (-0.07)	0.49 (0.10)	0.78 (0.00)	0.27 (-0.33)	0.27 (-0.17)	0.42 (-0.04)	Moderate
	P_{ro}	0.50 (0.03)	0.29 (-0.13)	0.53 (0.07)	0.46 (-0.03)	0.43 (-0.17)	0.42 (0.00)	0.44 (-0.04)	Moderate
OpenLLaMA-13B	P_{re}	0.56 (-0.03)	0.29 (0.00)	0.20 (0.07)	0.31 (-0.17)	0.27 (-0.20)	0.36 (0.00)	0.33 (-0.06)	Fair
	P_{ro}	0.29 (0.03)	0.38 (0.13)	0.62 (0.13)	0.44 (-0.10)	0.27 (0.13)	0.27 (0.03)	0.38 (0.02)	Fair

Pt: Prompt

aggr.: Agreement

negative sign (−) indicates higher score received from RA (vs. QA and Dev)

a substantial improvement in relevance attributed to the US-AC generated by SimAC. Particularly, SimAC embedding with *gpt-3.5-turbo* consistently performs the best, receiving 60.29% completeness and 51.29% validity, marking enhancements of 28.83% and 15.56%, respectively. SimAC slightly enhances *text-davinci-003* in GAC, showing an improvement of 13.83% in completeness and 0.53% in validity.

Despite the underperformance of the OpenLLaMA series, receiving less than 20% in completeness, SimAC still contributes to improving their completeness with improvements of 23.16% for *OpenLLaMA-7B* and 31.41% for *OpenLLaMA-13B*. However, embedding SimAC into the OpenLLaMA series leads to a decrease in validity, suggesting that SimAC might introduce some irrelevant information into the output.

Table 5.7 Completeness and validity of US-AC generated by different LLMs

(%)	Pt	NSF		UniBath		Zooniverse		Avg.	
		com	val	com	val	com	val	com	val
gpt-3.5-turbo	P_{re}	49.69	42.64	48.26	39.92	42.44	50.58	46.80	44.38
	P_{ro}	60.53	54.93	64.92	48.64	56.33	50.28	60.60	↑29.4851.29 ↑15.56
text-davinci-003	P_{re}	31.97	39.96	33.78	44.58	32.25	51.86	32.67	45.47
	P_{ro}	38.25	41.15	38.70	45.45	34.60	50.52	37.18	↑13.8345.71 ↑0.53
OpenLLaMA-7B	P_{re}	16.28	48.50	8.97	30.83	9.27	45.83	11.50	41.72
	P_{ro}	21.00	51.31	8.97	30.50	12.54	46.83	14.17	↑23.1642.88 ↑2.78
OpenLLaMA-13B	P_{re}	15.31	40.28	14.19	40.83	9.43	43.19	12.97	41.43
	P_{ro}	20.44	34.70	19.50	28.89	11.19	47.00	17.05	↑31.4136.86 ↓11.03

Pt: Prompt

bold indicates better completeness

The results in Table 5.8 reveal a statistically significant improvement in completeness upon integrating SimAC into *gpt-3.5-turbo*. However, the enhancement of completeness becomes less evident when introducing SimAC into other LLMs. Meanwhile, the enhancement in validity across all LLMs is not readily discernible (i.e., with *small* effect size).

Table 5.8 Significance analysis of relevance evaluation for different LLMs

p -value (Cohen's d)	P_{re} Versus P_{ro}			
	com		val	
gpt-3.5-turbo	$4.49e - 04$	(+Large)	$6.19e - 02$	(+Small)
text-davinci-003	$1.94e - 01$	(+Small)	$9.64e - 01$	(+Small)
OpenLLaMA-7B	$2.65e - 01$	(+Small)	$8.99e - 01$	(+Small)
OpenLLaMA-13B	$2.12e - 01$	(+Small)	$5.86e - 01$	(-Small)

p -value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of P_{ro} .

Answering RQ1: In summary, SimAC delivers substantial improvements for different LLMs of diverse parameter sizes in terms of both quality and relevance, demonstrating a robust generality across various LLMs. Specifically, the collaboration facilitated by role-based prompts yields high satisfactory quality and achieves completeness rates exceeding 60% when applied to LLMs trained with extensive parameter sizes, exemplified by the GPT-3.5 series. Moreover, this approach enhances the output quality in GAC even when embedded with LLMs trained with relatively smaller parameter sizes, such as those within the OpenLLaMA model family.

5.5.2 RQ2: Ablation Study for Role Groups

Quality Table 5.9 demonstrates satisfaction scores obtained from introducing various role groups to SimAC. The results notably indicate that SimAC, incorporating three distinct role groups, attains the highest satisfaction score of 3.15 on average. However, the inclusion of QA and Others results in the generation of more informative acceptance criteria, making the size of US-AC slightly exceed the scope compatible with sprint implementation. Therefore, the highest score of S is observed in scenarios where only RA is introduced. For the other metrics, the introduction of additional roles yields consistent enhancement. Notably, when three distinct role categories are introduced, the resulting outcomes reach a rating of *very satisfactorily* ($s > 3$).

From a statistical perspective, the US-AC generated by all three role groups ($R = 3$) proves significantly enhanced compared with the inclusion of RA only ($R = 1$), as demonstrated in Table 5.10. The involvement of QA and Others significantly enriches the comprehensiveness of requirement details and introduces additional valuable items

Table 5.9 Average satisfaction on INVEST score, evaluating by combinations of different role groups

<i>R</i>	I		N		V		E		S		T		Avg.
	<i>m</i>	<i>sd</i>	<i>m</i>	<i>sd</i>	<i>m</i>	<i>sd</i>	<i>m</i>	<i>sd</i>	<i>m</i>	<i>sd</i>	<i>m</i>	<i>sd</i>	
1	2.78	0.58	2.63	0.55	2.87	0.65	2.87	0.60	2.92	0.42	2.90	0.60	2.83
2	2.97	0.52	2.92	0.42	3.08	0.67	3.07	0.55	2.70	0.50	3.27	0.63	3.00
3	3.03	0.55	3.12	0.52	3.60	0.59	3.08	0.53	2.60	0.49	3.45	0.62	3.15

m - mean*sd* - standard deviation

bold indicates highest score

R - Groups of roles

Avg. - Average score over INVEST attributes

(*V*) to US-AC, indicated by *large* effect size of enhancement observed. It is noteworthy that the augmentation of *N* and *T* attain significant enhancement with three role groups incorporated.

Table 5.10 Significance analysis of INVEST score for different settings on combinations of role groups

<i>p</i> -value <i>Cohen's d</i>	<i>R</i> = 3					
	I	N	V	E	S	T
<i>R</i> = 1	1.75e-02 +Small	2.79e-06 +Large	2.21e-09 +Large	3.74e-02 +Small	2.57e-04 +Medium	2.84e-06 +Large
<i>R</i> = 2	4.97e-01 +Small	2.32e-02 +Small	1.70e-05 +Large	8.66e-01 +Small	2.71e-01 +Small	1.13e-01 +Small

p-value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of *R* = 3.negative sign (−) indicates worse performance in setting of *R* = 3.

Cohen's kappa scores for this set of evaluations are presented in Table 5.11, which are situated within the 0.4 to 0.6, indicative of *moderate* to *substantial* agreement. Reviewing the difference between the two evaluators, it is evident that the RA evaluator consistently assigns slightly higher scores, with the biggest discrepancy of 0.17 compared to the PM evaluator.

Relevance Table 5.12 illustrates the impact of introducing different role groups into the SimAC on relevance. The findings reveal enhancements in both completeness and validity as more roles are integrated. Evidently, the inclusion of three groups of roles results in the highest level of completeness. Conversely, the incorporation of only the

5.5 Experimental Results

Table 5.11 Kappa score and differences for assessing inter-rater agreements on role group combination evaluations

kappa (difference)	I	N	V	E	S	T	Avg.	Agreement
$R = 1$	0.68 (-0.10)	0.74 (-0.07)	0.54 (-0.13)	0.62 (-0.13)	0.28 (0.10)	0.75 (-0.07)	0.60 (-0.07)	Substantial
$R = 2$	0.38 (0.00)	0.26 (-0.03)	0.61 (-0.10)	0.43 (-0.07)	0.71 (0.07)	0.65 (-0.07)	0.51 (-0.04)	Moderate
$R = 3$	0.43 (0.00)	0.19 (0.03)	0.31 (-0.07)	0.48 (-0.17)	0.72 (0.07)	0.58 (-0.17)	0.45 (-0.05)	Moderate

negative sign (–) indicates higher score received from RA (vs. PM)

RA role or the combination of RA and QA leads to completeness and validity falling below 50%. Upon reviewing by project, the improvement remains consistent across all three projects.

Table 5.12 Completeness and validity in combinations of different role groups

(%)	NSF		UniBath		Zooniverse		Avg.	
	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>
$R = 1$	38.42	39.96	40.94	46.24	36.39	44.21	38.58	43.47
$R = 2$	38.64	38.52	45.75	41.19	43.82	44.05	42.74	41.25
$R = 3$	60.53	54.93	64.92	48.64	56.33	50.28	60.60	51.29

bold indicates highest completeness

The statistical analysis presented in Table 5.13 indicates the enhancement for both completeness and validity is significant and non-negligible (i.e., p -value less than 0.05 with a *large* or *medium* effect size) upon introducing more roles.

Answering RQ2: The most effective practice to approach SimAC is to engage the collaboration involving three distinct groups of roles: requirement analyst, quality analyst, and other team members.

5.5.3 RQ3: Analysis of K-shot Prompting

Quality The efficacy of incorporating different examples in the prompt is presented in Table 5.14. The results demonstrate that, with a setting of $K = 3$, the INVEST score reaches its zenith. The human evaluation consistently maintains *very satisfactorily*

Table 5.13 Significance analysis of completeness and validity for different settings on combinations of role groups

p -value (Cohen's d)	$R = 3$			
	com		val	
$R = 1$	$2.16e - 07$	(+Large)	$4.11e - 02$	(+Medium)
$R = 2$	$5.03e - 05$	(+Large)	$3.41e - 03$	(+Medium)

p -value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of $R = 3$.

($s > 3$) across all INVEST metrics, both with $K = 3$ and $K = 4$. Concerning the increment of K , an overall enhanced level of satisfaction is observed when an increased number of scenarios are encompassed within the prompt examples. Specifically, the setting of $K = 3$ accommodates more scenarios involving different engagement levels, where participants acting in diverse roles can thereby contribute outputs that leverage their respective expertise. Consequently, these detailed acceptance criteria encompass a broader spectrum of negotiable (N), valuable (V), and testable (T) components. In terms of I , the setting of $K = 4$ receives a higher satisfaction score since it benefits from a greater number of examples to generate fewer dependency-related elements.

Table 5.14 Average satisfaction on INVEST score, evaluating by K-shot prompting

K	I		N		V		E		S		T		Avg.
	m	sd	m	sd	m	sd	m	sd	m	sd	m	sd	
0	2.72	0.64	2.43	0.72	2.78	0.72	3.15	0.66	2.15	0.73	3.05	0.75	2.71
1	2.95	0.34	3.02	0.60	3.12	0.52	3.27	0.63	2.67	0.57	3.15	0.55	3.03
3	3.57	0.56	3.63	0.49	3.58	0.59	3.37	0.64	3.23	0.59	3.47	0.50	3.48
4	3.60	0.53	3.48	0.50	3.48	0.60	3.25	0.57	3.08	0.67	3.33	0.54	3.37

m - mean

sd - standard deviation

bold indicates highest score

K - Number of few shots

Avg. - Average score over INVEST attributes

Compared with zero-shot ($K = 0$) and one-shot ($K = 1$) prompting, the findings in Table 5.15 suggest that implementing the few-shot prompting strategy yields significant improvement in other metrics except for E . The evaluators perceive that the acceptance criteria generated by *gpt-3.5-turbo* maintain a consistently high level of satisfaction ($s > 3$) in representing estimable items. Thereby, no significant

5.5 Experimental Results

improvement is observed. Transitioning from $K = 3$ to $K = 4$ does not result in statistically significant differences across all INVEST metrics.

Table 5.15 Significance analysis of INVEST score for different settings on K-shot prompting

p -value	$K = 3$					
Cohen's d	I	N	V	E	S	T
$K = 0$	$4.17e-12$ +Large	$4.78e-19$ +Large	$8.27e-10$ +Large	$6.97e-02$ +Small	$7.67e-15$ +Large	$4.89e-04$ +Medium
$K = 1$	$4.61e-11$ +Large	$8.22e-09$ +Large	$1.17e-05$ +Large	$3.91e-01$ +Small	$4.86e-07$ +Large	$1.28e-03$ +Medium
$K = 4$	$7.39e-01$ -Small	$9.96e-02$ +Small	$3.58e-01$ +Small	$2.93e-01$ +Small	$1.97e-01$ +Small	$1.65e-01$ +Small

p -value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of $K = 3$.

negative sign (−) indicates worse performance in setting of $K = 3$.

Cohen's kappa scores for this particular set of evaluations, as depicted in Table 5.16, fall within the range of 0.2 to 0.4, suggesting a *fair* level of agreement. Notably, in this evaluation, the RA consistently assigns tighter scores across various metrics, except for V . Conversely, when evaluating the value of US-AC, the QA evaluator tends to assign lower scores. One potential explanation for this difference is their respective focus on different aspects of the US-AC.

Table 5.16 Kappa score and differences for assessing inter-rater agreements on k-shot settings

kappa (difference)	I	N	V	E	S	T	Avg.	Agreement
$K = 1$	0.25 (0.43)	0.47 (0.33)	0.28 (-0.03)	0.31 (0.43)	0.33 (0.30)	0.34 (0.30)	0.33 (0.29)	Fair
$K = 2$	0.22 (0.10)	0.22 (0.37)	0.33 (-0.17)	0.32 (0.20)	0.25 (0.07)	0.23 (-0.17)	0.26 (0.07)	Fair
$K = 3$	0.29 (0.33)	0.30 (0.13)	0.40 (-0.03)	0.30 (0.00)	0.27 (0.20)	0.46 (0.00)	0.34 (0.11)	Fair
$K = 4$	0.21 (0.27)	0.27 (0.43)	0.32 (-0.03)	0.43 (-0.17)	0.28 (0.10)	0.35 (0.00)	0.31 (0.10)	Fair

negative sign (−) indicates higher score received from RA (vs. QA)

Relevance Reviewing Table 5.17, the highest completeness is observed for $K = 3$, which remains consistent across all three projects. Remarkably, the completeness exceeds 50% for each project individually, implying that the US-AC generated by SimAC, prompted with three sets of examples, possesses the capability to match half of those written by humans. There is a general trend that a higher number of examples leads to better completeness. However, for $K = 4$, its coverage scenario is narrower than that of $K = 3$, resulting in lower completeness.

Regarding validity, there is no consistent trend indicating that an increasing number of examples yields higher validity. However, the setting of $K = 3$ yields the best validity. One assumption is that the introduction of more scenarios covering different proportions of role group engagement introduces more relevant information, thereby reducing the false positives generated.

Table 5.17 Completeness and validity in different settings on K-shot prompting

(%)	NSF		UniBath		Zooniverse		Avg.	
	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>
$K = 1$	34.94	38.24	43.01	28.13	42.69	30.40	40.21	32.25
$K = 2$	35.47	40.76	45.33	37.00	47.57	48.73	42.79	42.16
$K = 3$	60.53	54.93	64.92	48.64	56.33	50.28	60.60	51.29
$K = 4$	43.11	39.27	52.37	43.01	45.03	46.03	46.84	42.77

bold indicates highest completeness

Statistical analysis in Table 5.18 indicates significant improvements in both completeness and validity upon incorporating examples covering three engagement scenarios.

Table 5.18 Significance analysis of completeness and validity for different settings on K-shot prompting

<i>p</i> -value (Cohen's <i>d</i>)	$K = 3$			
	com		val	
$K = 1$	1.45e-05	(+Large)	1.92e-05	(+Large)
$K = 2$	7.61e-06	(+Large)	2.01e-02	(+Medium)
$K = 4$	4.64e-04	(+Large)	9.27e-03	(+Medium)

p-value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of $K = 3$.

Answering RQ3: Leveraging few-shot prompting, which covers three typical collaboration scenarios, SimAC gains higher satisfaction levels on output quality. The resulting US-AC demonstrates greater alignment with the human-crafted gold standard, rendering them more pertinent and applicable.

5.5.4 RQ4: Analysis of Similarity Thresholds

Quality Table 5.19 presents the effectiveness of employing varying similarity thresholds. Obviously, when t is set as 0.95, the highest average INVEST score is achieved. The increment of the threshold demonstrates a consistent improvement in INVEST.

Table 5.19 Average satisfaction on INVEST score, evaluating by semantic similarity thresholds

t	I		N		V		E		S		T		Avg.
	m	sd	m	sd	m	sd	m	sd	m	sd	m	sd	
0.95	2.75	0.65	2.35	0.61	3.53	0.50	3.50	0.50	2.48	0.68	3.47	0.54	3.01
0.90	2.12	0.64	2.12	0.74	2.15	0.86	2.27	0.82	2.37	0.64	2.15	0.73	2.19
0.85	1.63	0.80	1.58	0.98	1.37	0.86	1.22	0.85	1.70	0.98	1.17	0.96	1.44

m - mean

sd - standard deviation

bold indicates highest score

t - Threshold

Avg. - Average score over INVEST attributes

Reviewing from Table 5.20, it is evident the utilization of higher t preserves more significant valuable information within acceptance criteria. The comprehensive elaborations of details enhance the understandability and testability of the user story, consequently yielding the highest score in E and T . However, the intricate elaborations have the consequence of reducing the size of the user story, potentially extending it beyond the scope of a single sprint. Consequently, in terms of S , the influence of varying t from 0.9 to 0.95 exhibits a relatively less pronounced effect.

The Cohen's kappa scores (Table 5.21) for this set of evaluations range between 0.4 and 0.5, reflecting a *moderate* level of agreement. When comparing the scoring for two evaluators, no significant discrepancies are evident, with the largest difference being less than 0.2.

Relevance The results presented in Table 5.22 indicate that decreasing the similarity threshold results in lower completeness. A decrease in the threshold entails fewer

Table 5.20 Significance analysis of INVEST score for different settings on similarity threshold

<i>p</i> -value	<i>t</i> = 0.95					
<i>Cohen's d</i>	I	N	V	E	S	T
<i>t</i> = 0.90	4.20e-7 +Large	4.16e-02 +Large	3.25e-19 +Large	3.13e-17 +Large	3.33e-01 +Small	2.27e-20 +Large
<i>t</i> = 0.85	1.44e-13 +Large	4.82e-07 +Large	4.36e-33 +Large	1.46e-35 +Large	1.32e-06 +Large	8.43e-32 +Large

p-value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of *t* = 0.95.

Table 5.21 Kappa score and differences for assessing inter-rater agreements on similarity threshold

kappa (difference)	I	N	V	E	S	T	Avg.	Agreement
<i>t</i> = 0.95	0.49 (0.17)	0.88 (0.07)	0.21 (-0.13)	0.23 (-0.20)	0.69 (-0.10)	0.25 (-0.13)	0.46 (-0.05)	Moderate
<i>t</i> = 0.90	0.26 (-0.17)	0.64 (-0.03)	0.45 (-0.17)	0.60 (-0.20)	0.31 (0.07)	0.22 (0.03)	0.41 (-0.07)	Moderate
<i>t</i> = 0.85	0.23 (-0.13)	0.49 (0.17)	0.51 (-0.07)	0.65 (-0.03)	0.42 (0.07)	0.62 (0.07)	0.49 (0.01)	Moderate

negative sign (−) indicates higher score received from RA (vs. PM)

5.5 Experimental Results

acceptance criteria being included, thereby reducing the amount of information available to match items in the gold standard.

Conversely, a decrease in the threshold leads to higher validity in some cases. With a lower threshold, SimAC yields a smaller total instance, consequently resulting in higher validity.

Table 5.22 Completeness and validity in different settings on similarity threshold

(%)	NSF		UniBath		Zooniverse		Avg.	
	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>
$t = 0.95$	60.53	54.93	64.92	48.64	55.42	50.28	60.60	51.29
$t = 0.90$	32.00	65.12	32.42	57.67	28.07	56.02	30.83	59.60
$t = 0.85$	17.39	59.17	21.42	73.33	16.53	51.67	18.45	61.39

bold indicates highest completeness

Statistical analysis (Table 5.23) reveals a significant improvement in completeness. However, the improvement in validity is not as pronounced, with only a *small* effect size observed.

Table 5.23 Significance analysis of completeness and validity for different settings on similarity threshold

<i>p</i> -value (<i>Cohen's d</i>)	$t = 0.95$			
	com		val	
$t = 0.90$	$2.03e - 18$	(+Large)	$1.26e - 01$	(+Small)
$t = 0.85$	$1.69e - 11$	(+Large)	$1.16e - 01$	(+Small)

p-value less than 0.05 is highlighted with grey background.

bold indicates large effect size.

positive sign (+) indicates better performance in setting of $t = 0.95$.

Answering RQ4: It is evident that redundant information is generated by different roles in the process of GAC within SimAC. The subsequent deduplication process can streamline the refined US-AC to a size that fits in appropriate sprint planning and discussion. Importantly, the careful selection of an optimal similarity score threshold can significantly impact the relevance and quality of the US-AC.

5.6 Case Studies

This section presents three case studies exemplifying outputs produced by SimAC, which integrate various LLMs for qualitative analysis. Cases 1 and 2 depict the successful scenarios of SimAC employed in GPT-3.5 and OpenLLaMA series, alongside offering a detailed breakdown of role-related output in Case 1. Case 3 discusses a failed scenario that SimAC employed in LLMs with fewer parameters. Such an approach enables a thorough exploration of the strengths and limitations inherent in SimAC. To ensure a comprehensive analysis infused with industrial perspectives, our researchers collaborated closely with practitioners to scrutinize the intricacies presented within these specific cases.

5.6.1 Case 1: SimAC on GPT-3.5

For the LLMs trained with massive parameter sizes, the LLM initiated with regular prompts has already received highly satisfactory results. The incorporation of role-based prompts in SimAC can elaborate the US-AC with role-related details. Case 1 is a typical example, as shown in Figure 5.6. In Case 1, the goal specified in the user story is to search and discover music or ringback tones based on content similarity.

Introduction on Informative Details

In Figure 5.6, the left panel presents the acceptance criteria generated by SimAC incorporating with the regular prompt, while the right panel presents the output generated based on LLM utilizing the role-based prompt. The underlined items correspond to the acceptance criteria aligned with the human-crafted gold standard, while the points highlighted in red denote aspects deemed valuable by practitioners for inclusion in the discourse.

As can be seen, Point 2 on both sides defines the criteria for the system to display the results based on the content similarity. However, the right panel further extends its elaboration to define the facets and considerations for content similarity. Moreover, compared with Point 7 in the left panel, Point 10 in the right panel elucidates the scenario (e.g., a large volume of contents) necessitating consideration during performance handling. SimAC enables the LLM to delineate the consideration factors, which serves as a valuable point of reference. The RA experts endorse the inclusion of such supplemental content, which aids in formulating the comprehensive requirements and resulting in a highly satisfactory degree in terms of *E*.

Furthermore, Point 9 updates the acceptance criteria by introducing considerations on software compatibility in different platforms, which provides an alignment for development. Points 8 and 11 furnish detailed considerations regarding functionalities warranting discussion or clarification with stakeholders, thereby serving as valuable reference points. Therefore, these supplements contribute to the high score received in V .

In terms of relevance, both approaches receive a TP_s value of 7, as underlined in the middle panels presented in Figure 5.6. However, SimAC generates additional acceptance criteria, which slightly reduces the validity of the case. The TP_g associated with the criteria generated from regular LLM and that produced by the SimAC are 7 and 8, respectively, as underlined in the gold standard panel. In the SimA solution, Point 6 integrates the filter and sort functionalities, whereas they are separately addressed in Points 4 and 8 in the regular LLM solution. Furthermore, the SimAC solution features one additional true positive against the gold standard (i.e., Point 7 in the gold standard panel), as highlighted in Point 11. The support for the partial match mechanism is exclusively mentioned on the SimAC solution, contributing to a TP_g of 8 with higher completeness for this case.

Supplements on Role-related Insights

Figure 5.7 illustrates the detailed acceptance criteria generated or updated by different roles in Case 1. The acceptance criteria updated by the QA role place a particular emphasis on the performance issue and error handling within the search function (as presented in the middle panel). As underlined in Points 10 and 11, these criteria align with the human-crafted gold standard, thus augmenting the completeness. Incrementally, the LLM acts in other roles to address the legacy issue. In this context, the additional considerations primarily stem from the perspective of a project manager. As highlighted in red, these additional considerations serve as valuable points of reference for stakeholder deliberations, even their absence in the original gold standard. It is noteworthy that SimAC utilizes role-playing techniques, facilitating the generation of detailed insights from role-specific perspectives, thereby introducing valuable points for reference and further discussion.

Chapter 5. SimAC: Requirement Elicitation in Agile RE

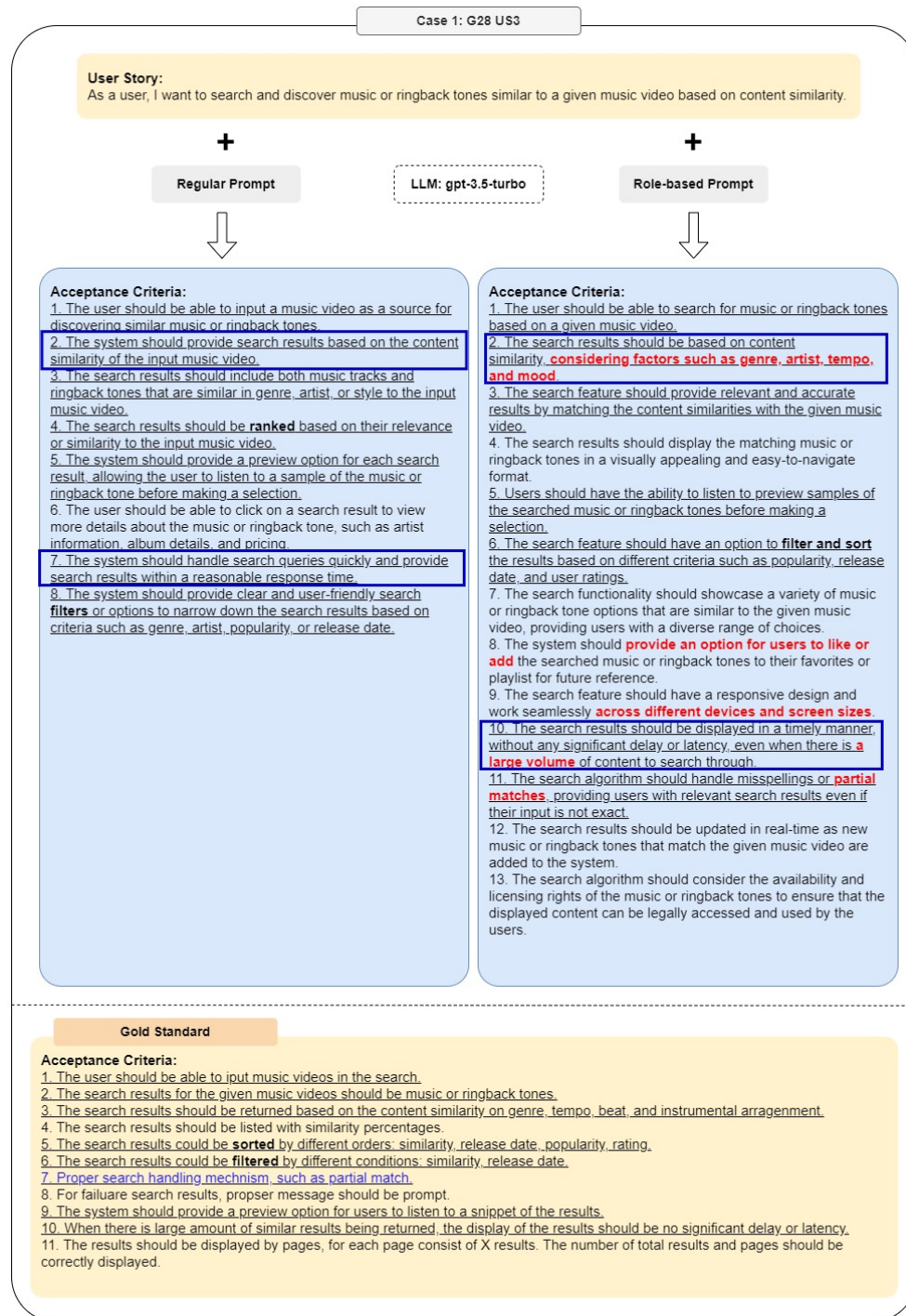


Fig. 5.6 Case 1: Generate acceptance criteria with the regular and role-based prompt (G28 US3).

Finding 1: In contrast to initiating LLMs with regular prompts, SimAC incorporates collaborative procedures into the LLM, thereby introducing more informative acceptance criteria and augmenting insights from role-specific perspectives. These enriched criteria and role-specific enhancements are considered valuable references for practitioners.

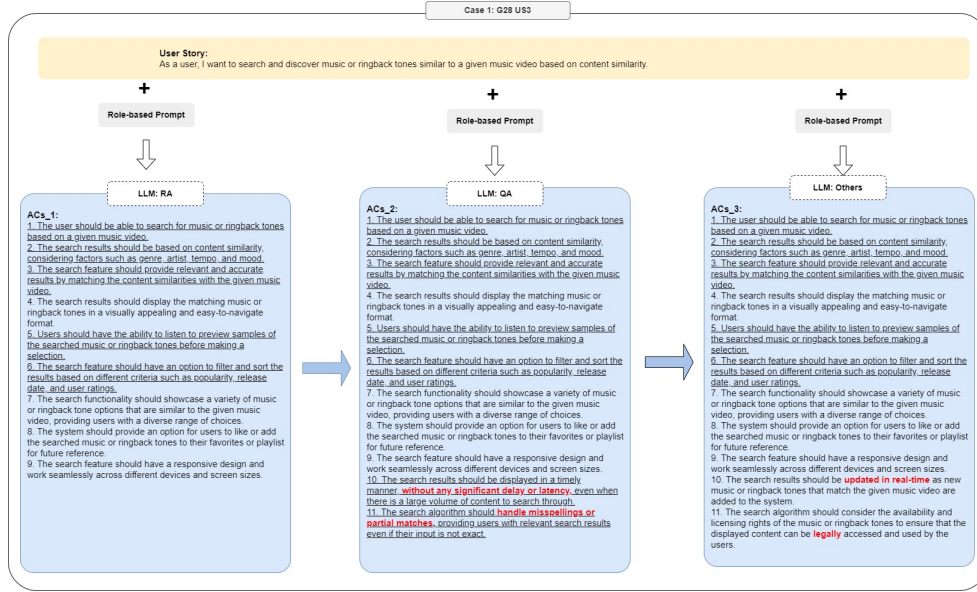


Fig. 5.7 Case 1: Generate acceptance criteria from different roles (G28 US3).

5.6.2 Case 2: SimAC on OpenLLaMA

SimAC boosts the efficacy of GAC, particularly for LLMs with smaller sizes. Figure 5.8 showcases the acceptance criteria generated by *OpenLLaMA-7B* with the regular prompt and the role-based prompt. In Case 2, the goal specified in the user story is to allow the user to deposit arbitrarily large files into the system.

In Figure 5.8, the acceptance criteria generated by the regular prompt, as presented in the left panel, elaborate the user story with limited information. The output provides insights into deposit files of any format, as repeatedly presented in Points 2 and 4. However, it erroneously specifies file limitations (highlighted in blue on the left) despite the user story suggesting the ability to deposit arbitrarily large files. The integration of role-based prompts in SimAC, exemplified in the right panel, rectifies the mistake and extends the depth of elaboration. The output supplements the narrative with specifics regarding file storage location and acceptable compressed file formats. QA experts consider the keywords highlighted in red as more informative criteria, which offer valuable references for designing test strategies.

Considering the relevance, the TP_g remains at 1 (i.e., Point 2 underlined in the bottom panel) for the regular LLM solution, as underlined, where Points 2 and 4 in the middle-left panel convey similar information. In contrast, TP_g stands as 2 in the SimAC solution, reflecting SimAC's capacity to generate more precise and informative criteria, as highlighted in blue at the bottom.

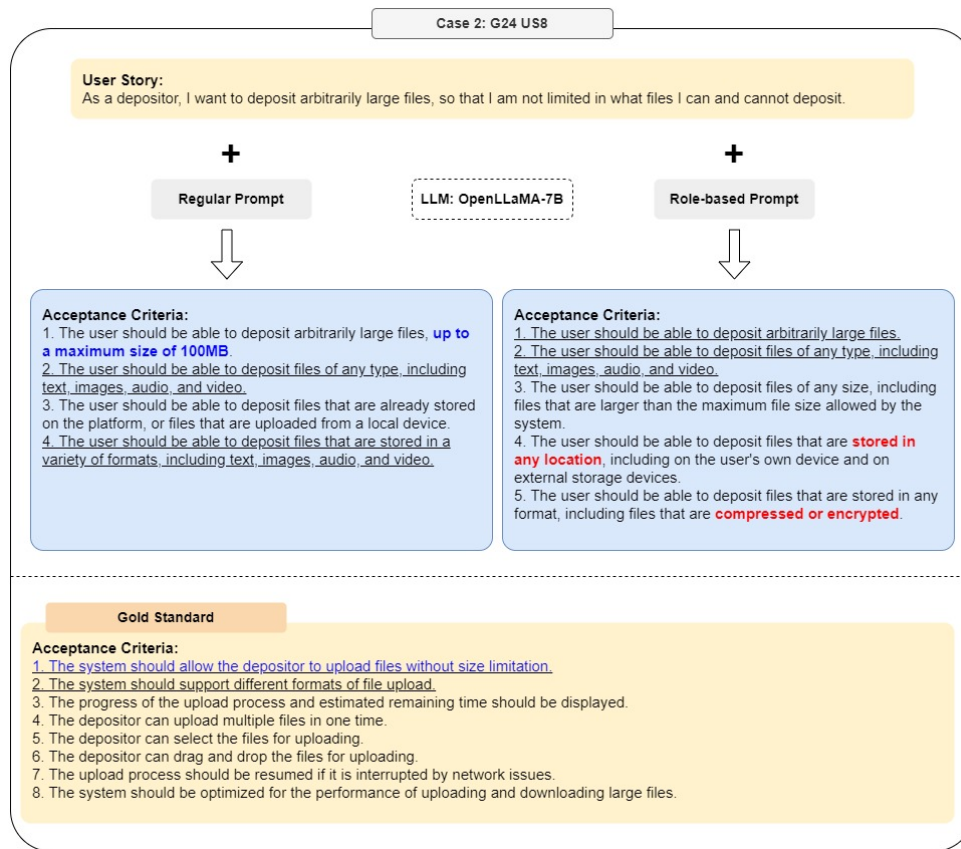


Fig. 5.8 Case 2: Generate acceptance criteria with the regular and role-based prompt (G24 US8).

Finding 2: SimAC enhances the capacity of LLMs trained with constrained parameters to generate more correct and informative details in GAC. Additionally, SimAC slightly improves the completeness of the output. Nonetheless, these improvements fall short of reaching a satisfactory level.

5.6.3 Case 3: Failures of SimAC

For some of the user stories, such as Case 3 presented in Figure 5.9, it is evident that LLMs trained with more limited parameters do not exhibit any improvement even when employing SimAC. The goal specified in Case 3 is to integrate the user-centered design into the recommendation.

In Case 3, no additional information is generated when introducing the role-based prompt on *OpenLLaMA-7B*. One probable explanation is that the LLM lacks

project-specific knowledge, thereby impeding its ability to elaborate on particular scenarios. In this case, the given user story includes vague descriptions, such as “our recommendation”. This kind of description does not furnish sufficient contextual information for LLMs to understand what type of recommendation should be integrated with a user-centered design. Even utilizing SimAC, the LLMs remain unable to extract additional relevant information from their knowledge base.

This limitation persists even when employing LLM trained with larger parameters, such as *OpenLLaMA-13B*. No supplementary acceptance criteria are generated, as presented in the middle-left portion in Figure 5.9. Although the incorporation of SimAC into *OpenLLaMA-13B* yields a more informative outcome, no valuable or pertinent information emerges. Among these solutions generated in this case, only one basic criterion mentions platform access rights, which is in line with the gold standard. This particular scenario underscores the diminished effectiveness of SimAC when it is introduced to LLMs with less proficiency in understanding the context of the user story.

When embedding SimAC with *text-davinci-003*, the resulting US-AC for this case is more reasonable. In the solution generated by the regular prompt, the US-AC places emphasis on the report, a component not explicitly mentioned in the original user story. However, with the introduction of the role-based prompt, the focus of the US-AC shifts towards a demonstration of user-centered design for discussion, a theme more closely aligned with the gold standard. As highlighted in red, experts suggest that these points also offer valuable references for further discussion during sprint planning.

Finding 3: In scenarios where the provided user story offers an indistinct depiction of the project context, the effectiveness of SimAC is diminished when implemented on LLMs trained with fewer parameters.

5.7 Threats

Our results need to be considered in view of threats to validity. We follow [200] to discuss the *internal*, *external*, *conclusion*, and *construct* validity of our study.

5.7.1 Internal Validity

Internal validity concerns the external factors that might affect dependent variables [200], which may be due to individual factors in the proposed method. In the proposed

Chapter 5. SimAC: Requirement Elicitation in Agile RE

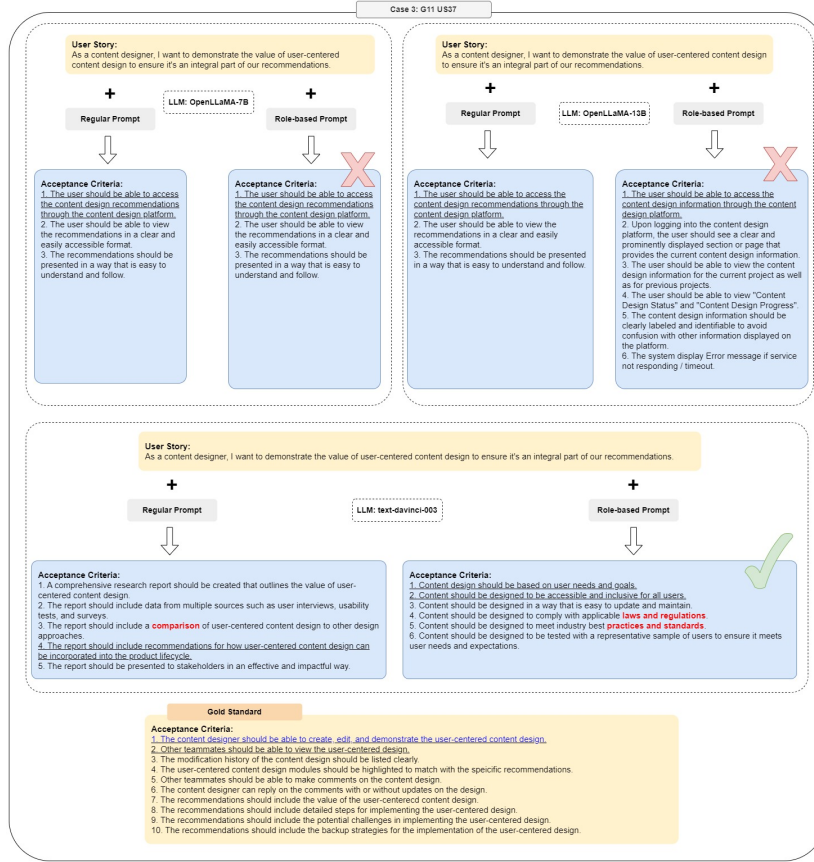


Fig. 5.9 Case 3: Generate acceptance criteria with the regular and role-based prompt in different parameter sizes (G11 US37).

SimAC, a fixed set of examples is utilized, while different sets of examples may yield diverse results. To mitigate the impact of these threats, we purposefully crafted some examples for common projects demonstrated in online resources rather than relying solely on the dataset D itself. This strategy minimizes the overlap of project contexts between prompting examples and the testing dataset, thereby enhancing the resilience of our proposed framework.

5.7.2 External Validity

External validity addresses the generalizability of the results [200]. The absence of first-hand information about these projects leads to an inevitable deviation between the gold standard crafted by our invited practitioners and the ground truths gained in real-world applications. This gap is more evident in agile practices, which necessitates more insights from the project environment [201]. To mitigate this threat, during

the evaluation process and drafting of the gold standard, we involved practitioners to familiarize themselves with the project background. Furthermore, we provided a three-week window for evaluators to review results. To gain more convincing results, one of the future research directions is to collaborate with industry projects and apply our approach in a real-world context to obtain first-hand results.

Another threat arises from the limited experimented models in this study. As mentioned in Section 2.5.1, encoder-only models are only suitable for text understanding without text generation capability, while most encoder-decoder models are in relatively small parameter sizes, such as CodeT5 [139] and UniXcoder [140], leading their poor performance in training-free task adaptation. As such, we focus on prevalent decoder-only models, including a series of GPT-3.5 and OpenLLaMA, where the former is closed-source LLMs while the latter is open-source. Particularly, we select *OpenLLaMA-7B* and *OpenLLaMA-13B* to make comparisons among LLMs of the same family but different sizes. Besides, we select *gpt-3.5-turbo* and *text-davinci-003* to make comparisons among LLMs of similar parameter sizes but diverse pre-training foundations. Hence, diverse LLMs are tested in the work, and we believe such an approach can minimize the identified threat.

5.7.3 Conclusion Validity

Conclusion validity concerns the relationship between the proposed method and the outcome [200]. We followed the various assumptions of the statistical tests when analyzing the results. The t – $test$ combined with Cohen’s d was employed to assess the statistical significance of improvements in each set of evaluation results, including quality INVEST scoring and relevance measurement. In addition, the kappa score was utilized to assess the inter-rater agreement, with only fair or above agreement considered valid, reinforcing our analysis.

Another potential threat arises in the construction of the gold standard, which is intended to serve as a benchmark for evaluation. However, it may involve subjective elements. The possibility of presenting an alternative gold standard may change the results [36]. To mitigate bias, practitioners were invited to collaboratively construct the gold standard and engage in discussions to reach an agreement on its contents. During the evaluation process, two researchers meticulously compared the results with the gold standard. In cases where ambiguity arose, practitioners were involved in discussions to reach a consensus on the final decision. These threats are partially mitigated by practitioner engagement, but they still exist.

5.7.4 Construct Validity

Construct validity primarily concerns the relationships between theory and observations, and these threats are mainly due to the methods employed to evaluate task outcomes [200, 36], which poses three potential threats in our study. In this study, we examined if SimAC improves LLMs’ capability in GAC. The dataset selection in the evaluation may affect the results. To mitigate the threat, we selected three projects that are easy to understand and cover different domains. As presented in Section 5.4.2, these three projects offer representative examples encompassing diverse levels of user story complexity for the purposes of our experimental evaluations.

The property of evaluation metrics presents another significant threat. To the best of our knowledge, there is no widely acknowledged metric designed specifically to assess the quality of US-AC. INVEST mnemonic is widely recognized as a guiding principle for crafting high-quality product backlog items [34, 75, 190, 202]. Considering the significant relevance of the user story and acceptance criteria [29] and following [192] to score INVEST on agile requirement quality, we employ INVEST scoring to evaluate the quality of the auto-generated US-AC. Furthermore, we invited experienced practitioners to make a more accurate assessment of the completeness and validity of auto-generated US-AC. This serves as a remedy for a series of potential problems caused by using the INVEST metric. Therefore, we contend that our evaluation approach is comprehensive.

Another potential threat arises from the evaluation sequence. This sequence may introduce bias into the evaluation process, as perceptions of outputs might be influenced by the order in which they are observed. To mitigate this bias, we offer a three-week evaluation window to the evaluators, enabling them to review their scoring based on a comprehensive understanding of the project background and outputs. Furthermore, to alleviate their workload and improve focus, we divided the evaluations into four groups, ensuring that each evaluator was assigned to fewer than two groups.

5.8 Summary

In this study, we propose the SimAC framework incorporated with the role-based prompt to streamline the iterative discussions in GAC, aiming to reduce human effort in agile RE. To evaluate the effectiveness of SimAC, we applied it to GPT-3.5 and OpenLLaMA models to conduct empirical studies. To gain in-depth insights from SE and RE communities, we invited eight agile practitioners to participate

in the evaluation process. The evaluation leverages INVEST metrics to assess the quality of auto-generated US-AC, particularly focusing on their quality across various attributes. The findings indicate that US-AC generated by SimAC with the *gpt-3.5-turbo* model the *gpt-3.5-turbo* model achieve 60.60% completeness and 51.29% validity, representing significant improvements of 29.48% and 15.56%, respectively. Additionally, through an in-depth examination of specific cases, practitioners emphasized the refined US-AC as a valuable reference for subsequent user story elaboration. This study lays the groundwork for further exploration of automation in agile RE.

Implications for Practitioners An increasing number of practitioners express interest in leveraging LLMs to aid in complex tasks. Although LLMs like the GPT-3.5 series exhibit acceptable outputs, our experimental findings indicate that they tend to yield less relevant results compared to those crafted by humans. SimAC effectively enhances LLM performance, producing higher-quality and more pertinent outcomes. Despite the potential risk of automating this task leading to team members having a limited understanding of requirements, generating high-quality acceptance criteria at the initial stage establishes a robust foundation for subsequent refinement, thereby promoting more effective collaboration.

Implications for Researchers This study represents the first empirical investigation into the capabilities and limitations of recent LLMs in the automated generation of acceptance criteria. Subsequently, insightful case studies were conducted to highlight successes and shortcomings, offering potential avenues for further enhancement. Researchers are encouraged to devise more effective methods to harness the potential of LLMs. While the prompts meticulously crafted at each phase of the SimAC framework have been optimized, the possibility remains that alternative prompt designs may offer superior efficacy. Hence, we call for researchers to delve deeper into prompt design based on SimAC, thereby propelling automation in agile RE forward.

Appendix 5.A Guidance of Questionnaire

5.A.1 Background Introduction

Questionnaire for the effectiveness of different models in the task of generating acceptance criteria to elaborate the given user stories. The User Story associated with Acceptance Criteria is marked as "US-AC".

Please score each US-AC by the five attributes of INVEST evaluation criteria. The score of each attribute is 0 to 4.

- 0 = US-AC fulfilled the X attribute poorly.
- 1 = US-AC fulfilled the X attribute unsatisfactorily.
- 2 = US-AC fulfilled the X attribute to some extent.
- 3 = US-AC fulfilled the X attribute satisfactorily.
- 4 = US-AC fulfilled the X attribute very satisfactorily.

INVEST evaluation attributes:

- Attribute 1 (**Independent**): The US-AC should be as independent as possible. It can be worked on in any order. It allows for true prioritization of each and every story.
- Attribute 2 (**Negotiable**): The US-AC is not an explicit contract and should leave space for discussion. It can be rewritten, depending on business, market, technical, or any other type of requirement by team members.
- Attribute 3 (**Valuable**): The US-AC must deliver value to the stakeholders.
- Attribute 4 (**Estimable**): The US-AC should be estimable. It is well enough understood by team members to be able to determine its relative size (in development and cost).
- Attribute 5 (**Small**): The US-AC should not be so big as to become impossible to plan/task/prioritize within a level of accuracy.
- Attribute 6 (**Testable**): The US-AC must provide the necessary information to make test development possible.

5.B Sample of Evaluation on US-AC

The generations for different models are randomly ordered.

And US-ACs are randomly picked from 3 projects (e.g., g11, g24, g28). Brief description of three projects:

- g11 - New version of the NSF website: redesign and content discovery.
- g24 - Institutional data repository for the University of Bath.
- g28 - Citizen science platform that allows anyone to help in research tasks.

5.A.2 Actions

Please fill in the Tab Scores (Columns D to H) in ScoreSheets.xlsx.

- Column "Group No." - The number of projects.
- Column "US No." - The number of given user stories.
- Column "Model #" - The model number of the generated acceptance criteria.
- Column "Independent" - The score of Independent. (0 to 4)
- Column "Negotiable" - The score of Negotiable. (0 to 4)
- Column "Valuable" - The score of Valuable. (0 to 4)
- Column "Estimable" - The score of Estimable. (0 to 4)
- Column "Small" - The score of Small. (0 to 4)
- Column "Testable" - The score of Testable. (0 to 4)

Please fill in the Tab Information (Column B) in ScoreSheets.xlsx. Column A - Description of your information. Column B - Details of your information.

Appendix 5.B Sample of Evaluation on US-AC

User Story

As an NSF employee, I want to have editorial-planning recommendations so that I can understand how they might impact an eventual CMS / future web presence.

Acceptance Criteria: (generated from Model #1)

- (1) The editorial-planning tool should be available to employees with access to the current CMS or future web presence.
- (2) The tool should provide recommendations for content that is likely to attract the most traffic or engagement.
- (3) The recommendations should be based on data insights and analysis.
- (4) The tool should be user-friendly and intuitive, enabling employees to easily input editorial content and receive recommendations.
- (5) Recommendations should include keywords or search terms that are likely to generate traffic.

Acceptance Criteria: (generated from Model #2)

- (1) The editorial planning recommendations should be based on data-driven analysis, not just assumptions or opinions.
- (2) The CMS should provide a dashboard or interface for displaying the editorial planning recommendations for easy access for NSF employees.
- (3) The editorial planning recommendations should be displayed in an easytounderstand format that presents the relevant information and makes the recommendations actionable.
- (4) The recommendations should address any potential SEO considerations.

Chapter 6

CoTUML: Requirement Analysis in Agile RE

6.1 Introduction

During the modeling phase of agile RE, refined user stories can be translated into more intricate and specific specifications [36]. A valuable strategy for this refinement process involves deriving conceptual models, such as UML diagrams [35]. However, the derivation of conceptual models, particularly class diagrams, from user stories demands substantial manual effort. Ferrari et al. [117] investigated the effectiveness of ChatGPT in generating UML diagrams and sequence diagrams based on natural language requirements. Their findings indicate that enhancing ChatGPT with additional contextual cues and domain-specific knowledge shows potential for improving its capability in model generation. This chapter explores the possibility of automating this task by leveraging LLMs and a tailored prompting technique known as CoT.

The fundamental motivation behind this research is to establish an automated approach that utilizes LLMs to generate class diagrams directly from user stories. This transition towards automation holds promising potential for reducing human effort while providing a valuable reference for requirement analysis. Therefore, we have developed a three-phase CoT prompting incorporated with the LLM approach, namely CoTUML, for automating the derivation of class diagrams.

CoTUML involves three key phases in its reasoning steps: class identification, attribute identification, and relationship identification. These user stories sourced from [36] are utilized as input for the derivation process, and the resultant outputs

are formatted to demonstrate identified classes, attributes, and relationships. These generated outputs serve as a reference for the subsequent construction of the class diagram. This analysis covered both quantitative and qualitative aspects to evaluate the effectiveness and accuracy of the LLM-based approach.

To gain a comprehensive understanding of LLM-based techniques and assess the efficacy of our designed CoT prompting in the process of deriving class diagrams, we have formulated the following RQs for investigation.

RQ1. How are different prompting techniques performed in deriving class diagrams from user stories? This RQ examines the performance of CoT prompting in deriving class diagrams compared to other prompting techniques. To address this, regular prompting methods (e.g., zero-shot and few-shot prompting) are applied to the LLM for evaluation.

RQ2. What effective of different settings in CoT prompting can impact the performance of deriving class diagrams? CoTUML integrates different settings (i.e., a diverse set of examples and different interactive inputs) to initialize the LLM for deriving the class diagram. In this RQ, we aim to identify the specific contributions of these individual settings.

RQ3. What is the performance of CoT prompting in distinct cases? The process of deriving class diagrams involves three distinct steps. This RQ seeks to evaluate the effectiveness of CoTUML in each of these individual steps.

To address RQ1 and RQ2, we conducted experiments on a small dataset for preliminary study. Subsequently, an extensive qualitative analysis of the experimental results was undertaken to address RQ3.

The results demonstrate that LLM-based approaches, particularly when combined with well-crafted few-shot prompts, outperform guided human extraction in identifying classes from user stories. The CoTUML further improves the comprehensiveness of relationship identification. However, we also identified areas of suboptimal performance through qualitative analysis.

This study delves into the reasons behind the identified shortcomings in the LLM-based approach and discusses potential avenues for improvement. The insights gained from this analysis provide valuable guidance for further research in automating the derivation of conceptual models from user stories. The proposed CoT prompting technique offers a promising pathway to automate the derivation of class diagrams in

agile projects, reducing the reliance on manual effort. Our study contributes valuable insights and directions for future research in this field.

This study [203] was accepted by the 2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC 2024). The rest of the chapter is organized as follows. Section 6.2 introduces the CoTUML proposed in our study, and Section 6.3 demonstrates the experiment design settings. Section 6.4 showcases the preliminary quantitative results. Furthermore, we provide a comprehensive discussion on Section 6.5. In Section 6.6, the threats of our study are reviewed. Finally, Section 6.7 concludes our study.

6.2 CoTUML

In this section, we introduce CoTUML, which guides the LLMs in generating intermediate reasoning steps for class diagram construction. CoTUML focuses on class identification, attribute identification, and relationship identification as the key components of this process. By delineating these intermediate steps, CoTUML provides a structured approach to automate the derivation of class diagrams from user stories. The CoTUML framework is implemented through a three-step CoT prompting process, which includes components of *context*, *examples*, and *instructions*.

6.2.1 Steps of Class Diagram Derivation

In accordance with the guidelines presented in [36], we characterize the output of the LLM-based class diagram as triplet format, consisting of three components: *class*, *attribute*, and *relationship*. Each component represents a description that corresponds to the respective element in a finalized class diagram. In earlier work [204, 89, 205, 36], the results show that a systematic deviation process can generate higher-quality models. In a typical human-driven process for class diagram derivation, three identification phases are involved [206, 207, 36]. These phases, depicted in Figure 6.1, entail the initial identification of classes from the given requirements. Subsequently, based on the identified classes, associated attributes are extracted. Lastly, the relationships between classes are defined to culminate in a comprehensive class diagram.

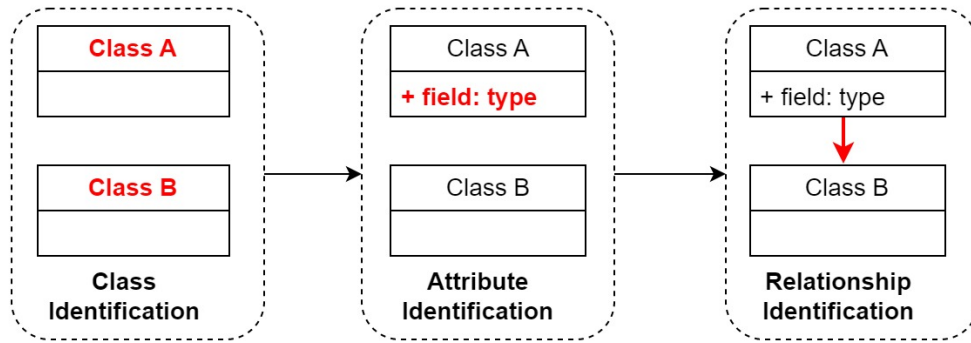


Fig. 6.1 Phases in class diagram derivation.

6.2.2 CoT Prompting

Based on practical observations and aligned with the human extraction guidelines proposed in [36], we introduce a tailored CoT prompting technique for deriving class diagrams from user stories. This prompting technique encompasses three components: context, CoT examples, and instruction message, as illustrated in Figure 6.2.

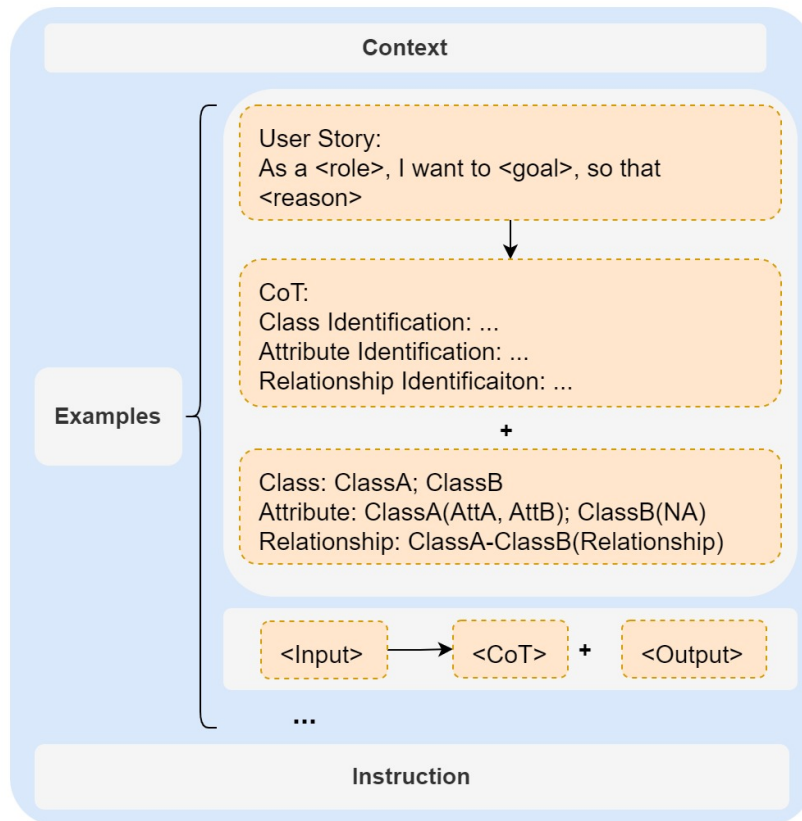


Fig. 6.2 Structure of CoTUML: three-step CoT promptings in class diagram derivation.

Context

The context component of the CoT prompting provides the necessary information and background for the LLMs to understand the task at hand, including the user stories and any additional context relevant to the class diagram derivation. It adheres to the recommended prompting practices outlined by OpenAI¹, incorporating contextual information within the prompt to guide the LLM's behavior and the expected output format.

Noteworthy, the class diagram produced by LLM is at the conceptual level rather than the implementation level. Typically, the conceptual class diagram poses greater challenges in derivation, particularly when the development team lacks familiarity with the problem domain. Hence, the focus of our study is on providing a high-level description of the models rather than delving into low-level intricacies. The resulting output serves as a compass for subsequent stages of code generation or more exhaustive detailed design endeavors.

Additionally, CoTUML stipulates the use of commas to separate attributes associated with the same class. The presence of a relationship between classes is denoted by a hyphen, and the relationship itself is encapsulated within brackets. These conventions help to structure the output and provide a clear representation of the class diagram elements.

By adhering to these guidelines and incorporating them into CoTUML, we aim to guide the LLMs in generating class diagrams that are concise and understandable and serve as a valuable reference for further development and design activities.

CoT Examples

The CoT prompting consists of intermediate reasoning steps that provide examples that demonstrate the thought process involved in deriving class diagrams. These examples serve as guidance for the LLMs in generating coherent CoT and reasoning. Within the prompting framework, the CoT steps are seamlessly integrated into illustrative prompt examples. Each example is composed of three segments: `<input>`, `<CoT>`, and `<output>`.

- `<input>` corresponds to the requirement input formatted as a user story. It provides the initial context for the LLMs to understand the specific task.
- `<CoT>` aligns with the three steps involved in class derivation: class identification, attribute identification, and relationship identification. It

¹<https://help.openai.com/en/articles/6654000-best-practices-for-prompt-engineering-with-openai-api>

encompasses the distinct reasoning steps and instructions for each of these identification steps.

- $\langle \text{output} \rangle$ represents the anticipated output from the LLM. It encapsulates the outcomes of the three identification components and provides an example of the desired class, attribute, and relationship descriptions.

An illustrative instance of the CoT prompting is depicted in Figure 6.3, showcasing how the CoT examples are combined into three steps of the class diagram derivation process to guide the LLM. The description of the user story is “*As an archivist, I want to link electronic versions of researchers’ publications to citations so that I can share them with other researchers*”.

Class Identification The initial phase in deriving a class diagram from natural language requirements involves the identification of classes. In the illustrative CoT examples, special attention is given to the designation of a $\langle \text{role} \rangle$ as the actor in the user story. Following the human extraction guideline [36], the instructional CoT directs the LLM to assess whether the actor engages in operations on the system. If the actor is involved in system operations, it is designated as a class.

The process of class identification primarily entails discerning the primary entity from the user stories. This is typically done by identifying nouns explicitly mentioned within the narratives. These nouns represent the predominant entities within the domain and are often indicative of potential classes.

In the illustrated example delineated in Figure 6.3, the *Archivist* lacks any operations on the system, suggesting that the *Archivist* does not constitute a class but rather functions solely as an actor.

By emphasizing the identification of roles and recognizing nouns as key entities, the CoT example guides the LLM in identifying classes from user stories. This systematic approach helps to ensure that the relevant entities are captured and considered for further analysis and construction of the class diagram.

Attribute Identification The succeeding phase in class diagram derivation involves the identification of attributes associated with the entities. In this phase, the CoT comprises background messages and logical reasoning steps. Initially, it introduces the classes identified in the preceding step. Subsequently, the example guides the LLM in determining the attribute-related indicator associated with the identified class.

In the process of attribute identification, the attribute-related indicator is typically articulated as a noun containing less intricate information. Another prevalent scenario involves a noun signifying a status, denoting that the primary entity possesses such a status. In such instances, the noun indicative of the status should be recognized as the attribute of the primary entity.

As exemplified in Figure 6.3, the term *electric version* imparts information pertaining to the status. Consequently, per this CoT, *isElectric* is identified as an attribute associated with the class *Publication*.

Relationship Identification The final phase in the process of deriving the class diagram involves the identification of relationships among entities, with certain verbs playing a contributory role in defining these relationships. Consistently, we initiate the process by introducing background messages. Building upon the identified classes, the prompting message guides the LLM in discerning messages indicating the connection between the two classes. In the examples of CoT prompting, two frequently utilized relationships, namely *association* and *generalization*, are presented.

In the example presented in Figure 6.3, the verb *link* serves to articulate the association between *Publication* and *Citation*, thereby indicating a relationship between these two classes. Another example provided in the prompt introduces a user story as follows: “As an archivist, I want to manage documentation, including preservation photos, forms, and correspondence so that I can find it easily later.” In this context, the verb *include* signifies a stronger parent-child relationship, implying a generalization relationship.

Instruction

Following the prescribed steps within the CoT framework, the prompt incorporates instructions directing the LLM to generate an output that enumerates three essential components of the class diagram: *class*, *attribute*, and *relationship*.

6.2.3 Implementation

Our proposed CoTUML is a prompting technique designed for the derivation of class diagrams from user stories, and it is not contingent on specific LLM. In this study, we adopt the ChatGPT API, denoted as *gpt-3.5-turbo*, as the default LLM. The provided prompting examples are input into GPT through both the user and assistant roles in the API call.

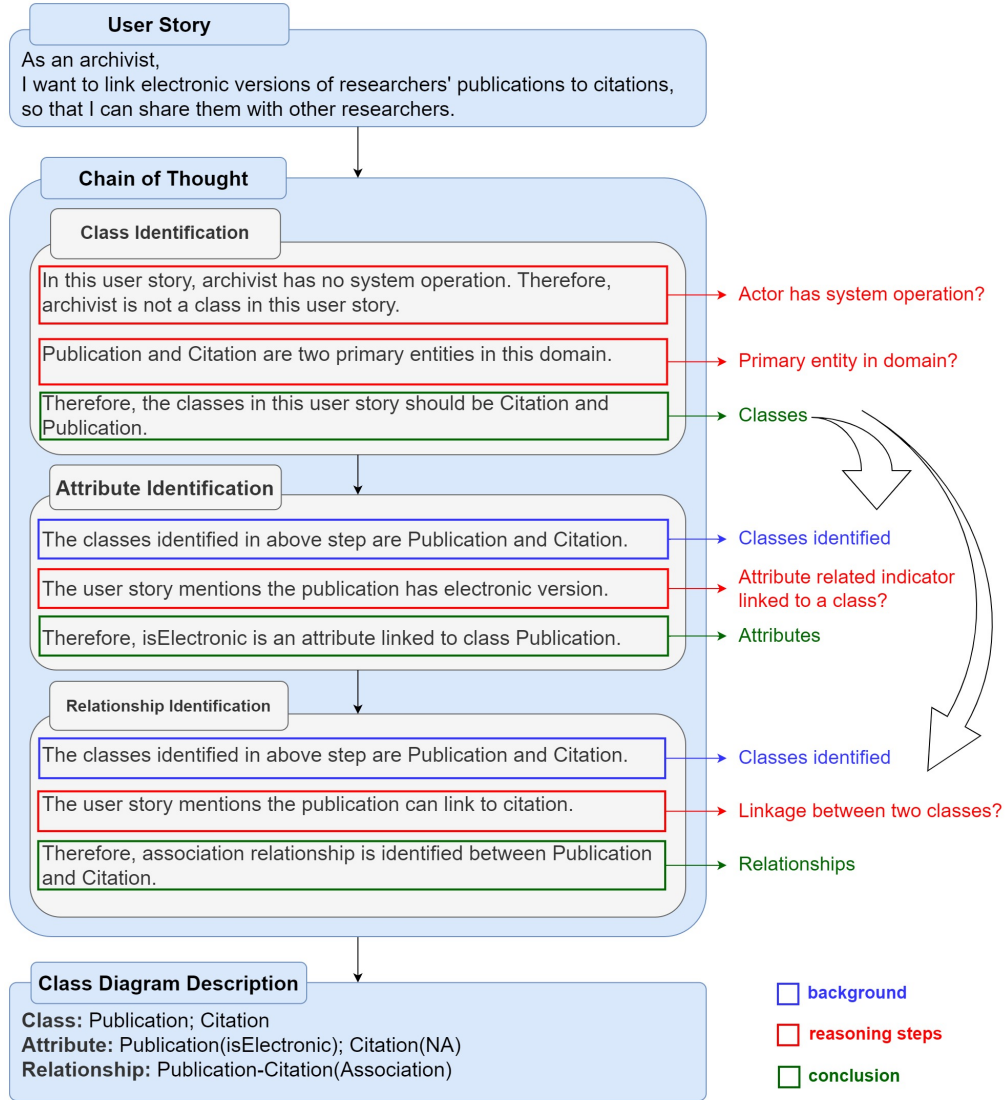


Fig. 6.3 An example of CoT prompting in three steps of class diagram derivation.

6.3 Experiment

6.3.1 Dataset

Following case studies conducted in [205, 36], we select the two projects from user stories collected in [33] to conduct preliminary case studies on evaluating our proposed framework. Project Data Hub (DH) consists of 20 user stories delineating the functionalities of a web interface designed to collect, organize, share, and search

for datasets. Meanwhile, Project Planning Poker (PP) encompasses 22 user stories outlining the requirements for the initial iteration of the planningpoker.com website.

6.3.2 Comparison Baselines

To assess the effectiveness of our approach, we select three mainstream prompting techniques as baselines.

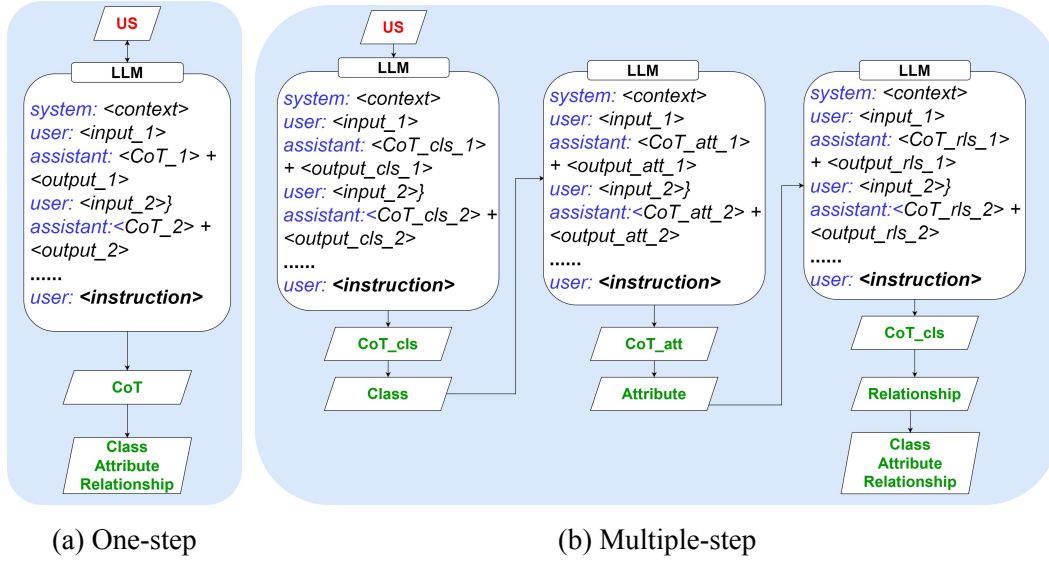
- **Zero-shot Prompting** directly feeds context and instruction to the LLM without examples.
- **Few-shot Prompting** introduce some examples with $\langle \text{input} \rangle$ and $\langle \text{output} \rangle$ pairs fed into the LLM to assist with generating the three components of the class diagram.
- **CoT Prompting** is a variant of few-shot prompting [40]. It exemplified the reasoning steps $\langle \text{CoT} \rangle$ together with the $\langle \text{output} \rangle$ as presented in Figure 6.2.

6.3.3 Settings

To answer RQ2, we designed comprehensive experiments to evaluate the effectiveness of different settings introduced to CoT prompting.

Prompting Interactions In the provided prompting examples, exemplified by one as depicted in Figure 6.3, the CoT incorporates distinct cognitive steps. To investigate the variances arising from presenting the prompt in a single step versus segregating the examples step-by-step, we formulated two modes of prompting interaction as inputs. Figure 6.4 illustrates the two interactive inputs utilized in our experimental design. In the one-step prompting approach, as presented in Figure 6.4a, each example encapsulates all three CoT steps before generating the output. Conversely, in the multiple-step prompting approach as presented in Figure 6.4b, each example encompasses three iterations of the interaction. Within each iteration, there are components for $\langle \text{input} \rangle$, $\langle \text{CoT} \rangle$, and $\langle \text{output} \rangle$. The $\langle \text{CoT} \rangle$ and $\langle \text{output} \rangle$ are encapsulated in separate components, distinctly delineating class (*cls*), attribute (*att*), and relationship (*rls*).

Number of Examples The CoT technique, a variant of few-shot prompting [40], is susceptible to the influence of the number of examples provided [208]. Based

Fig. 6.4 Interactive input in the CoT prompting, exemplified in *gpt-3.5-turbo*.

on such an assumption, we have formulated distinct example templates as inputs, taking into account token constraints. The initial template encompasses a total of 5 examples distributed across 3 scenarios for class identification, 2 scenarios for attribute identification, and 2 scenarios for relationship identification. Subsequently, the second template comprises 6 examples, introducing an additional instance with an extra scenario specifically focused on attribute identification.

6.3.4 Evaluation

Metrics

The two most frequent variables, taken from conceptual modeling research [209, 204], used for measuring conceptual models are *validity* and *completeness*.

Validity True positive (TP) signifies the number of instances present in the solution generated by the LLM that also aligns with the instances in the gold standard. Conversely, false positive (FP) denotes the count of instances in the LLM-generated solution that do not correspond to instances within the gold standard solution. The metric *validity* (val) is defined as the ratio of TP to the total instances in the LLM-based solution ($TP + FP$). In information retrieval terms, validity equates to precision.

$$val = \frac{|TP|}{|TP| + |FP|} \quad (6.1)$$

Completeness The term false negative (FN) denotes the count of instances that exist in the gold standard but cannot be identified in LLM-generated output. The metric *completeness* (com) is defined as the ratio between TP and the number of instances in the gold standard ($TP + FN$). In information retrieval terms, completeness equates to recall.

$$com = \frac{|TP|}{|TP| + |FN|} \quad (6.2)$$

Given that the identification of attributes and relationships is contingent upon the prior identification of connected class entities, we adopted an *adjusted version* of validity and completeness introduced in [205, 36]. The adjusted version calculates the validity and completeness metrics specifically in relation to those attributes and relationships within the gold standard that pertain to identified class entities.

Human Evaluation

We invited some experienced architecture designers and requirement engineers to critically assess the gold standard solutions obtained from [36]. Our authors conducted a comparative analysis, systematically contrasting outputs generated by our proposed LLM-based technique with the established gold standard. Varied depictions of classes, attributes, and their interrelationships were subjected to deliberation among experienced practitioners and our research team to inform the ultimate determination. The evaluation sought to ascertain the validity and comprehensiveness of the class diagrams produced through the LLM-based methodology.

6.4 Results

In this section, we present the quantitative analysis of the experiment results. The results are calculated and compared against a gold standard for evaluation.

6.4.1 RQ1: Performance of LLM-based Approaches

In response to RQ1, we present the *adjusted* completeness and validity of the class diagram components derived from different prompting techniques compared to two forms of human extraction outcomes as documented in [36]: 1) **Guided Derivation**, involving evaluation of class diagrams formulated by students following example-based guidelines; 2) **Unguided Derivation**, where students developed class

diagrams without additional directives. Comparative results for Project PP and Project DH are presented in Table 6.1 and 6.2, respectively.

In terms of class identification, the prompting techniques show higher completeness compared to guided and unguided manual derivation in both two projects. However, for attribute identification, the prompting-based techniques do not perform as well as human extraction in Project PP. In some cases, few-shot and CoT prompting techniques yield a higher count of completed attributes than the output from guided human extraction in Project DH. Similarly, regarding relationship identification, the prompting LLM method does not generate satisfactory results in Project PP, but the few-shot prompting and CoT prompting techniques improve the performance. In Project DH, these techniques achieve better completeness than human extraction.

Upon comparing the three prompting techniques, it is evident that CoT prompting attains the highest level of completeness across all three subtasks (i.e., class, attribute, and relationship identification) in the two projects. In terms of validity, the CoT technique demonstrates superior performance overall in both projects, with the exception of class identification in Project DH.

Table 6.1 Completeness and validity on PP

Approaches	Cls		Att		Rel	
	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>
Zero-shot Prompting	0.75	0.67	0.33	0.50	0.20	0.11
Few-shot Prompting	0.75	0.67	0.40	0.50	0.75	0.38
CoT Prompting (CoTUML)	0.78	0.78	1.00	0.50	1.00	0.40
Guided Derivation	0.78	0.53	0.6	0.49	0.59	0.57
Unguided Derivation	0.74	0.57	0.53	0.51	0.56	0.59

bold - best performance in prompting techniques

grey - better performance than human extraction

6.4.2 RQ2: Effective Settings in CoTUML

To address RQ2, we present the results of different settings applied to CoTUML by project, as demonstrated in Table 6.3 and 6.4.

Concerning Project PP, the outcomes derived from one-step and multi-step interactive inputs exhibit similarity. Notably, the one-step input demonstrates superior performance, especially in the completeness of relationship identification. In the

Table 6.2 Completeness and validity on DH

Approaches	Cls		Att		Rel	
	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>
Zero-shot Prompting	0.75	0.60	0.17	0.17	0.30	0.27
Few-shot Prompting	0.67	0.67	0.50	0.33	0.60	0.40
CoT Prompting (CoTUML)	0.65	0.73	0.50	0.33	0.75	0.50
Guided Derivation	0.67	0.37	0.58	0.31	0.50	0.36
Unguided Derivation	0.62	0.36	0.44	0.29	0.37	0.33

bold - best performance in prompting techniques
grey - better performance than human extraction

context of Project DH, the one-step input yields greater completeness in both class and attribute identification, with no discernible difference in the sub-task of relationship identification. Overall, the one-step input proves to enhance completeness in the overall class diagram derivation process.

Table 6.3 Effectiveness of prompting interaction

Project	Interaction	Cls		Att		Rel	
		<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>
PP	One	0.78	0.78	1.00	0.50	1.00	0.40
	Multiple	0.71	0.78	1.00	0.50	1.00	0.29
DH	One	0.65	0.73	0.50	0.33	0.75	0.50
	Multiple	0.63	0.33	0.33	0.17	0.60	0.50

bold - better completeness in different interactive input mode

Concerning the augmentation of examples to encompass a broader range of scenarios in attribute identification, there is an observed improvement in completeness for class identification in both two projects. However, this augmentation does not yield a comparable improvement in the completeness of attribute and relationship identifications when applied to Project PP. Conversely, in Project DH, the introduction of examples covering more scenarios enhances the capacity to identify attributes and relationships.

Table 6.4 Effectiveness of number of examples

Project	Ex.	Scn.	Cls		Att		Rel	
			<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>	<i>val</i>	<i>com</i>
PP	5	3/2/2	0.83	0.56	1.00	0.50	1.00	0.40
	6	3/3/2	0.78	0.78	1.00	0.50	1.00	0.40
DH	5	3/2/2	0.59	0.67	0.50	0.17	0.83	0.42
	6	3/3/2	0.65	0.73	0.50	0.33	0.75	0.50

bold - better completeness in prompting with different examples

Ex. - Number of examples

Scn. - Number of covered scenarios

6.5 Case Studies

To address RQ3, we conducted case studies on three sub-tasks to provide in-depth qualitative analysis and evaluate the performance of regular prompting and CoT prompting in class diagram derivation.

Given the potential variance in outputs produced by the LLM, each prompting technique was iteratively applied in each project, undergoing five iterations. We summarized the partially identified, erroneously identified, and unidentified instances observed from five iterations. The results for two projects are presented in Tables 6.5, 6.6, and 6.7. The comprehensive review of these instances on a case-by-case basis was undertaken by two of our authors and the invited practitioners, with a subsequent analysis of potential causes. The analysis process was then discussed among all authors to reach an overall agreement on the assumptions made.

6.5.1 Case Study on Class Identification

Table 6.5 presents summarization-related entities that are ascertainable through the CoT prompting yet remain elusive via both few-shot and zero-shot prompting approaches. Figure 6.5 illustrates the intricate details of a user story and its corresponding output. Within this user story, the original requirement specifies two distinct policy types. This particular information is overlooked by the LLM when utilizing few-shot and zero-shot prompting techniques. In CoT prompting, the output elucidates the reasoning steps that encapsulate the summarization of the two distinct policies inherent in the requirement, denoting them as classes *EstimatePolicy1* and *EstimatePolicy2*. This observation aligns with the findings of human extraction reported in [36], where policies were identified to a very limited extent even with

guidelines, primarily due to their absence in explicit terms within user stories and their singular occurrence.

Table 6.5 Partial identified and unidentified classes

Project	Partial Identified	Unidentified
PP	Round; EstimatePolicy1; EstimatePolicy2	EstimationPolicy
DH	Tag ; PublishedDataPackage; Example	SiteDeployment

Red - only identified in solutions generated by CoT prompting

Blue - identified in solutions generated by zero and few-shot prompting, but not CoT prompting

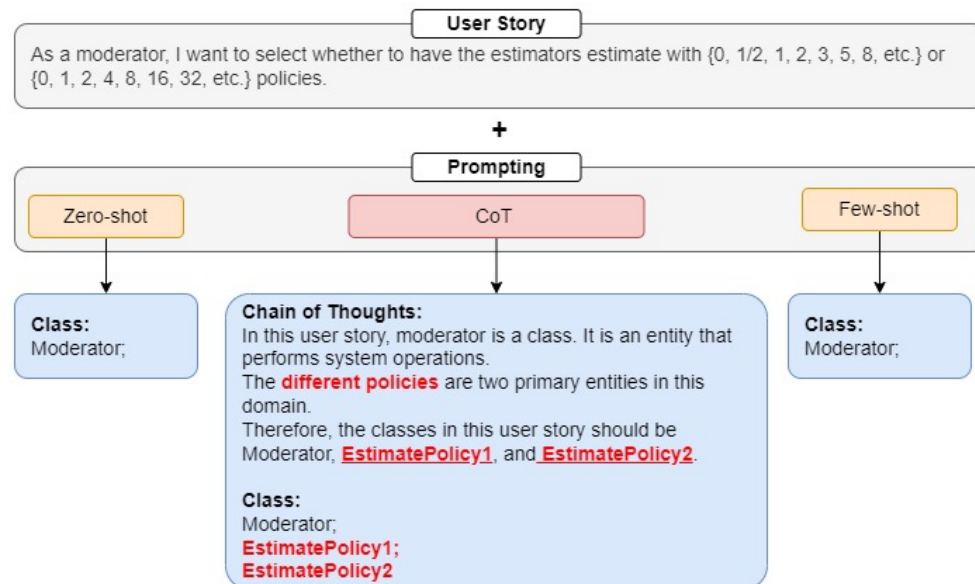


Fig. 6.5 Case study on class identification (PP).

Finding 1: The technique of CoT prompting has been developed to incorporate logical reasoning steps in order to improve proficiency in identifying classes associated with summarization. This aspect is often neglected by the LLM when employing regular prompting techniques.

Reviewing the unidentified classes outlined in Table 6.5 across different prompting techniques, class *EstimationPolicy* serves as an overarching category encompassing classes *EstimatePolicy1* and *EstimatePolicy2*, demanding a more nuanced summarization of knowledge, posing challenges for accurate identification by LLM. Similarly, *SiteDeployment* involves the aggregation of deployment behaviors

derived from user stories, necessitating the introduction of domain knowledge for proper identification. Notably, these classes were also identified to a restricted extent through human extraction, as documented in [36].

Finding 2: The current prompting methodologies utilized in the LLM demonstrate a limitation in their capacity to effectively perform aggregation summarization.

6.5.2 Case Study on Attribute Identification

Table 6.6 presents the erroneously identified attributes on the two projects. In Project DH, the description-related attributes are easily identified as classes. However, such entities contain less information, which is better to be set as attributes to reduce the complexity of the class diagram.

Figure 6.6 illustrates an instance where attributes are incorrectly identified as classes. The CoT generated in this case reveals that the LLM initially misidentified classes in the first step and subsequently, based on this erroneous identification, inaccurately identified the attributes. Despite providing examples specifically tailored to description-related scenarios in our prompting, the LLM appears unable to assimilate the information within the current prompting design.

Table 6.6 Unidentified and erroneously identified attributes

Project	Erroneously Identified	Unidentified
PP	N/A	Item (isEstimated); Estimate (isAgreedUpon)
DH	DataPackage (descriptor, resource, isValidate)	PublishedDataPackage (isSharedPublicly)

Finding 3: LLM-based approaches have a tendency to misidentify information categorized under the description type as classes instead of attributes, leading to erroneous classification.

Reviewing both projects, attributes were identified to a limited extent. As presented in Table 6.6, a predominant portion of the unidentifiable attributes is associated with those related to status (e.g., *isEstimated*, and *isSharedPublicly*) in both PP and DH projects. Despite the explicit mention of a singular type of status in the user stories, the LLM encounters difficulties in comprehending that this status could be identified as an

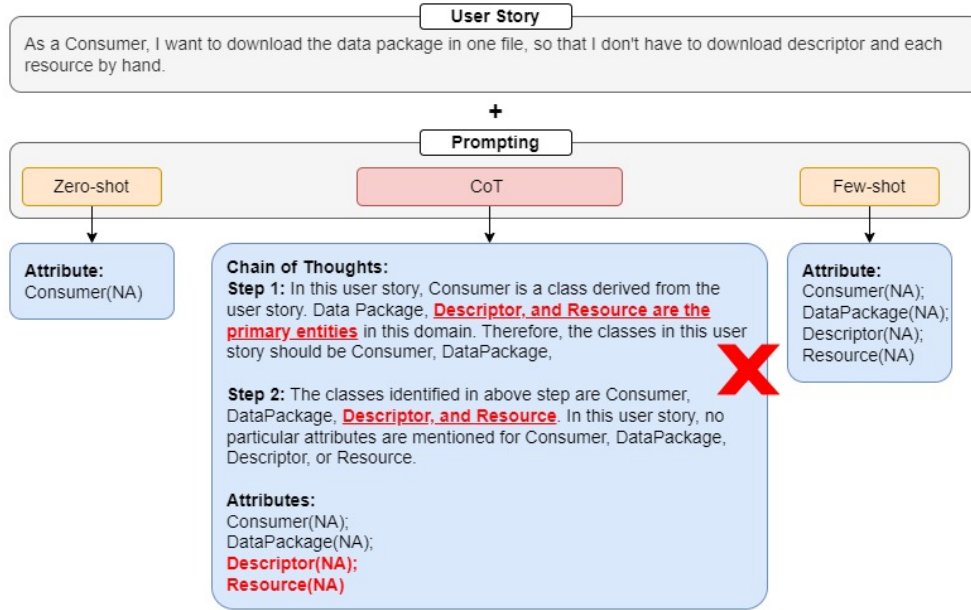


Fig. 6.6 Case study on attribute identification (PP).

attribute signifying various states or conditions. Even in the proposed CoT examples, where specific instances were introduced to guide the LLM in determining whether the entity possesses different statuses, such information is still overlooked.

Finding 4: When employing LLM-based approaches, the description related to the status can often be easily overlooked during the process of attribute identification.

6.5.3 Case Study on Relationship Identification

Table 6.7 provides details on relationships that are either undefined or erroneously identified in both project PP and DH. Examining the relationship identification within project PP, it is noteworthy that the validity is closely approximating 1, indicating that a substantial majority of the identified relationships are accurate.

In Project DH, within user stories related to the identified classes of *Consumer* and *Publisher*, some of them mention that *Consumer* can operate on the profile data of *Publisher*. In the gold standard, the *Consumer* and *Publisher* are intended to operate through the class *Account*. Such an operational association is latent within the messaging and lacks overt noun references in the user story, contributing to an

erroneous identification. In the LLM-based approaches, the association is directly linked between the *Consumer* and the *Publisher*.

Table 6.7 Unidentified and erroneously identified relationships

Project	Erroneously Identified	Unidentified
PP	N/A	Game-EstimatorPolicy (Association)
DH	Publisher-Consumer (Association)	Account-Consumer (Association); DataPackage-PublishedDataPackage (Inheritance)

Finding 5: The interpretation of associations between classes, particularly in situations where explicit noun references for intermediate operations are not provided, as indicated in the user story, can result in misinterpretation when employing LLM-based methods for relationship identification.

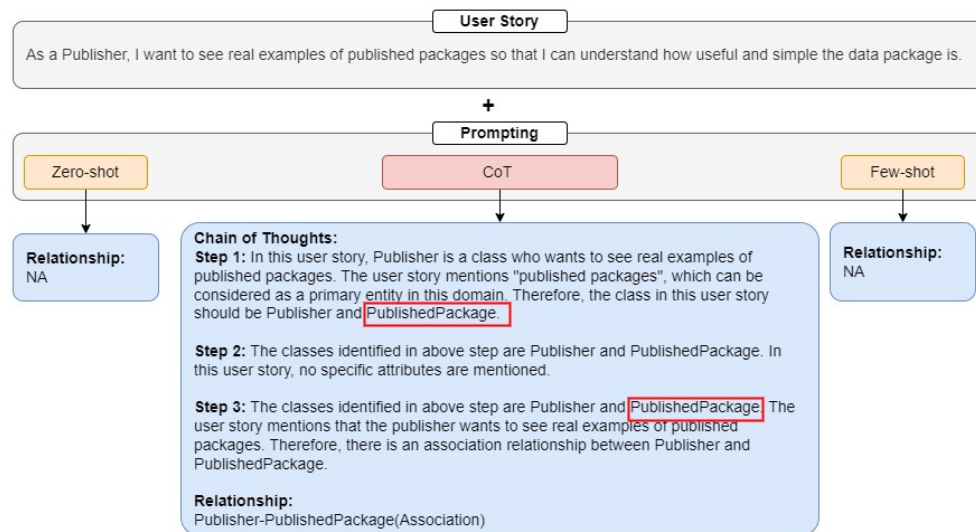


Fig. 6.7 Case study on relationship identification (DH).

Concerning the erroneously identified relationships in Project PP, some user stories elaborate on the actions of the *Estimator* within the context of the *Game*. However, the *Game* is depicted as an operational platform lacking explicit noun references in the user stories. This absence of explicit references results in the non-identification of both the class and the relationship.

Similarly, in Project DH, inheritance is the relationship left unidentified in the LLM-based generation. Figure 6.7 provides a detailed depiction of CoT prompting.

While the *PublishedDataPackage* can be identified, the LLM encounters difficulties in contextual integration to identify the relationship between *DataPackage* and *PublishedDataPackage*, where the latter inherits from the former. Consequently, the inheritance relationship between *DataPackage* and *PublishedDataPackage* remains unidentified. The association relationship between *Publisher* and *DataPackage* is linked for *Publisher* and *PublishedDataPackage* instead in LLM-generated solutions.

Finding 6: During the class identification step, the contextual information often remains elusive, consequently resulting in the subsequent omission of the associated relationship.

6.6 Threats

This section discusses the potential validity threats to our work. Following classifications in [200], these threats concern construct, conclusion, internal, and external validity.

6.6.1 Construct Validity

Construct validity primarily concerns the methods employed to evaluate task outcomes [200], which poses two potential threats in our study. In the proposed CoTUMML, a fixed set of examples is utilized, and different sets of examples may yield diverse results. To mitigate the impact of these threats, we adhere to a guideline proposed in [36], incorporating explanatory reasoning steps into the prompt. Additionally, a threat arises in the construction of the dataset. For the in-depth preliminary study, we purposefully selected a small set of user stories, potentially leading to the existence of representative issues inherent in real-world tasks.

6.6.2 Conclusion Validity

Conclusion validity concerns the relationship between the treatment and the outcome, which is addressed in our study through manual evaluation and case studies involving human assessment, potentially introducing bias. To mitigate this impact, we involve experienced practitioners in reviewing the gold standard and facilitating a comparative assessment between the results generated by the LLM and the gold standard. Additionally, potential findings during the analysis are compared with human

extraction outcomes presented in [36], and discussions within the research team are undertaken to reach a consensus.

6.6.3 Internal Validity

Internal validity concerns the factors that might affect dependent variables [200], which may be due to individual factors in the proposed method, such as the version adopted, the sequence of the prompting input, and others. To mitigate the validity, we designed to examine different input templates as presented in Section 6.3, to evaluate the effectiveness of different types of input.

6.6.4 External Validity

External validity addresses the generalizability of the results [200]. The main threat is the selection of two projects for experiments. The projects and human extraction were collected from undergraduate students in [33, 36]. The potential divergence between projects with industrial settings and those derived from student projects poses a significant threat. To mitigate this disparity, industrial architecture designers and experienced SE course instructors were invited to participate in the analysis process.

6.7 Summary

In this study, we have investigated the efficacy of the CoTUMML in the process of deriving class diagrams from natural language requirements. We designed experiments based on the formulated RQs to assess the effectiveness and limitations of CoT prompting and conducted both quantitative experiments and qualitative case studies to evaluate its performance.

Our experimental results showed that the CoT prompting technique improved the accuracy and completeness of class diagram derivation compared to traditional prompt-based methods. The incorporation of CoT examples in guiding the LLM-based class identification process proved to be effective. However, there were challenges observed in the identification of attributes, where the prompting-based techniques did not perform as well as human extraction.

Through the qualitative case studies, we gained deeper insights into the strengths and limitations of regular prompting and CoT prompting techniques. Analyzing partially identified and unidentified classes allowed us to understand specific scenarios

and challenges encountered during the class diagram derivation process. These case studies provided valuable information for further improving the prompting techniques and addressing potential limitations.

This study contributes to the field by providing empirical evidence and insights into the effectiveness of the CoT prompting technique. Our study can inform the development of automated systems for class diagram derivation and assist software engineers and analysts in efficiently and accurately deriving class diagrams from user stories. In future work, we intend to explore enhancements to the CoT prompting technique, particularly in addressing the challenges observed in attribute identification. Overall, this study advances our understanding of the class diagram derivation process and offers a valuable approach to improving its accuracy and completeness through the application of CoT prompting in the application of LLM for SE.

Implications for Practitioners CoTUMML enhances LLM’s ability to derive conceptual class diagrams for requirement analysis and subsequent design. While there is a potential risk that automating this task could lead to certain oversights, the capability to produce satisfactory initial results provides a strong foundation for further refinement. This approach enhances the effectiveness of requirements analysis and ultimately contributes to a more robust design process.

Implications for Researchers This study represents an initial exploration into automating conceptual class diagram generation using LLMs. Researchers are encouraged to develop more advanced methodologies to fully leverage the capabilities of LLMs. We advocate for further investigation into prompt design based on the limitations found on CoTUMML, with the aim of advancing automation in agile RE.

Chapter 7

Conclusion and Future Work

This thesis conducted action research on agile RE activities to observe the challenges in a real-world context and proposed some traditional RE techniques to alleviate the observed challenges. After that, a series of LLM-based techniques are proposed to automate the agile labor-intensive RE tasks, thereby enhancing the agile software development efficiency. This chapter provides a comprehensive summary of the studies conducted, highlights the contributions made, and suggests potential directions for future research.

7.1 Studies on Traditional RE Techniques

7.1.1 Summary

Incorporating the practical context of real-world settings, we identified challenges pertinent to agile RE in both WFH environments and in the development of ML-enabled products. Through an in-depth analysis of these real-world cases, we proposed several traditional RE techniques to address the identified challenges and reflected on the lessons learned.

Within the WFH paradigm, notable challenges include a lack of communication efficiency, a lack of interactive demonstration, and a lack of customer feedback. These challenges further exacerbate two RE risks: knowledge-sharing difficulties and issues with process definition. Collaborative strategies were implemented to mitigate these observed challenges. Noteworthy positive practices arising from this endeavor included a heightened emphasis on constant collaboration and the establishment of clear communication rules, facilitating the mitigation of comprehension gaps in requirement understanding. Additionally, the integration of design thinking principles

7.1 Studies on Traditional RE Techniques

into remote agile RE demonstrated efficacy in enhancing team comprehension of business processes. Furthermore, the adoption of mixed presentations proved advantageous in conveying coding and testing requirements. Conversely, the study also revealed two negative lessons. The propensity for non-functional requirements to be overlooked in the remote work context exacerbates knowledge dissemination challenges. Moreover, ineffective customer engagement during user acceptance testing emerged as a significant risk, leading to misalignments between product user experience and expectations.

As the integration of ML components into software product development becomes increasingly prevalent, a comprehensive understanding of the RE activities becomes indispensable for establishing a foundation for software development. Upon delving deeper into the intricacies of developing ML-enabled products, challenges were identified from the perspectives of both data scientists and requirement engineers. Predominant among these challenges are the lack of domain knowledge, the insatiable nature of ML models, and the rapid changes in financial data, all of which contribute significantly to the challenges encountered. Within the spectrum of RE activities, the pivotal role of data scientists in negotiating, validating, and verifying ML-related requirements is underscored. Recognizing the paramount importance of data and model development in such systems, a guidance framework is proposed to intersect RE activities, focusing on both data requirements and model requirements. Feedback obtained from participants engaged in RE activities underscores the necessity of embracing practices such as data protection considerations and adaptive evaluation rules in the development of ML-enabled systems.

7.1.2 Contributions

The ongoing trend towards professionals embracing permanent remote work arrangements is anticipated to persistently increase. Concurrently, the integration of ML components into product development is on the rise and aimed at enhancing product efficiency. These transitions pose significant challenges for companies employing agile RE methodologies.

The existing research in these areas remains relatively sparse, highlighting a critical gap in understanding agile RE practices with WFH dynamics and the development of ML-enabled applications. This thesis addresses this gap by offering practical insights and contributing to the broader RE community. By examining the implications of these transitions, the thesis provides valuable guidance for practitioners

Chapter 7. Conclusion and Future Work

navigating the complexities of agile RE environments. Furthermore, it offers a foundation for future research endeavors that will explore how agile methodologies can be refined and adapted to effectively address the challenges posed by WFH and developing ML-enabled applications. Through this exploration, the thesis advances both theoretical and practical understanding of agile RE practices in the landscape of remote work and ML development.

7.1.3 Limitations and Future Work

In both studies, the issue of transferability presents a considerable challenge, thereby limiting the generalizability of the results. To mitigate this limitation, our approach entails a comprehensive examination of a single case, utilizing both qualitative analysis of gathered feedback and quantitative analysis of relevant documents. This mixed-methods approach aims to enhance the transferability of the findings to other contexts that share similar conditions.

Future research efforts will seek to expand upon these findings by incorporating a broader range of companies. This expanded scope will allow for a more thorough examination of how agile RE practices can be leveraged to overcome challenges in remote work environments and in the development of ML-enabled products. By broadening the sample size and context, subsequent studies aim to provide a more robust framework for understanding the effectiveness of agile RE practices across diverse organizational settings. Additionally, to support the development of ML-enabled applications, we will explore appropriate statistical measures for data requirements. These investigations will provide deeper insights into RE activities in real-world scenarios. By integrating our research with practical contexts, we hope to gain a more comprehensive understanding of RE practices and develop effective techniques to address the challenges identified.

7.2 Studies on LLM-based RE Techniques

7.2.1 Summary

Given the advancements in LLM technology in NLP tasks, two of the LLM-based techniques are proposed to automate the labor-intensive agile RE tasks. To further investigate the effectiveness of the proposed techniques, both quantitative and qualitative analyses were conducted.

The SimAC framework incorporated with the role-based prompt to streamline the iterative discussions in GAC aims to reduce human effort in agile RE. To evaluate the effectiveness of SimAC, we applied it to GPT-3.5 and OpenLLaMA models to conduct empirical studies. To assess the efficacy of SimAC, we applied it to the GPT-3.5 and OpenLLaMA models. In order to gather comprehensive insights from the SE and RE communities, we engaged eight agile practitioners in the evaluation process. Evaluation criteria were based on the INVEST metrics, which were employed to evaluate the quality of auto-generated US-AC. Results indicate that US-AC generated by SimAC using the *gpt-3.5-turbo* model demonstrated a notable improvement, achieving 60.60% completeness and 51.29% validity, representing increases of 29.48% and 15.56%, respectively. Furthermore, practitioners highlighted the refined US-AC as valuable references for subsequent user story elaboration, as evidenced by specific case examinations. This study serves as a foundational step toward further exploration of automation within the realm of agile RE.

For CoTUML, we have investigated the efficacy of the CoT prompting technique in the process of deriving class diagrams from natural language requirements. Both quantitative experiments and qualitative case studies are conducted to assess the impact, effectiveness, and limitations of CoTUML. The results showed that the incorporation of CoT prompting notably enhanced the accuracy and comprehensiveness of class diagram derivation compared to regular prompt-based methods. Particularly, CoTUML demonstrated better performance in terms of completeness compared to guided and unguided manual extraction for class identification. However, there were challenges observed in the identification of attributes, where CoTUML did not perform as well as human extraction. Through qualitative case studies, deeper insights were gleaned into the strengths and limitations of CoTUML, which is less efficient in identifying status-related indicators.

7.2.2 Contributions

A growing number of practitioners are increasingly interested in leveraging LLMs to assist with complex tasks across various domains. While LLMs have demonstrated satisfactory performance and versatility, our experimental findings suggest that their outputs often exhibit a tendency to be less relevant and accurate compared to those produced by human experts. This discrepancy highlights the need for refined methodologies to optimize the performance of LLMs in practical applications. In response to these challenges, this thesis introduces innovative techniques such as

Chapter 7. Conclusion and Future Work

SimAC and CoTUML, which bridge the gap between LLM outputs and human expertise.

SimAC and CoTUM not only offer significant improvements in the relevance and utility of generated results but also streamline agile processes by reducing the reliance on manual efforts. These approaches provide valuable references for future research and practical applications, contributing to the advancement of LLM technologies in agile RE tasks. By minimizing human intervention and enhancing the efficiency of LLM-assisted tasks, this thesis seeks to pave the way for more effective and automated solutions in requirements engineering and software development.

7.2.3 Limitations and Future Work

Through comprehensive case studies aimed at exploring the limitations of these proposed techniques, potential avenues for further improvement have been identified. In cases where SimAC fails, LLMs are unable to extract additional relevant information from their knowledge base. One potential improvement is to strengthen the specific project context as part of the role-based prompt. For CoTUML, the description related to status and inheritance relationships is often overlooked during attribute identification. These limitations suggest a future direction for enhancing accuracy by emphasizing status and inheritance relationships in reasoning steps.

While the prompts meticulously crafted within the SimAC and CoTUML frameworks have been optimized, there remains the possibility that alternative prompt designs may offer superior efficacy. Consequently, we advocate for researchers to conduct in-depth investigations into LLM-based approaches, thereby advancing automation in agile RE.

References

- [1] Balasubramaniam Ramesh, Lan Cao, and Richard Baskerville. Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal*, 20(5):449–480, 2010.
- [2] Alan Davis, Scott Overmyer, Kathleen Jordan, Joseph Caruso, Fatma Dandashi, Anhtuan Dinh, Gary Kincaid, Glen Ledebor, Patricia Reynolds, Pradip Sitaram, et al. Identifying and measuring quality in a software requirements specification. In *[1993] Proceedings First International Software Metrics Symposium*, pages 141–152. Ieee, 1993.
- [3] Lloyd Montgomery, Davide Fucci, Abir Bouraffa, Lisa Scholz, and Walid Maalej. Empirical research on requirements quality: a systematic mapping study. *Requirements Engineering*, pages 1–27, 2022.
- [4] Jevgenija Pantiuchina, Marco Mondini, Dron Khanna, Xiaofeng Wang, and Pekka Abrahamsson. Are software startups applying agile practices? the state of the practice from a large survey. In *International Conference on Agile Software Development*, pages 167–183. Springer, Cham, 2017.
- [5] Rashina Hoda, Norsaremah Salleh, and John Grundy. The rise and evolution of agile software development. *IEEE software*, 35(5):58–63, 2018.
- [6] Xinyu Wang, Liping Zhao, Ye Wang, and Jie Sun. The role of requirements engineering practices in agile development: an empirical study. In *Requirements Engineering: First Asia Pacific Requirements Engineering Symposium, APRES 2014, Auckland, New Zealand, April 28-29, 2014. Proceedings*, pages 195–209. Springer, 2014.
- [7] Chunqiu Steven Xia, Yuxiang Wei, and Lingming Zhang. Automated program repair in the era of large pre-trained language models. In *Proceedings of the 45th International Conference on Software Engineering (ICSE 2023). Association for Computing Machinery*, 2023.
- [8] Guohao Li, Hasan Abed Al Kader Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. Camel: Communicative agents for” mind” exploration of large scale language model society. *arXiv preprint arXiv:2303.17760*, 2023.
- [9] Enkelejda Kasneci, Kathrin Sebler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities and challenges

References

- of large language models for education. *Learning and Individual Differences*, 103:102274, 2023.
- [10] Daniel Russo, Paul HP Hanel, Seraphina Altnickel, and Niels Van Berkel. The daily life of software engineers during the covid-19 pandemic. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 364–373. IEEE, 2021.
- [11] Courtney Miller, Paige Rodeghero, Margaret-Anne Storey, Denae Ford, and Thomas Zimmermann. ”how was your weekend?” software development teams working from home during covid-19. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pages 624–636. IEEE, 2021.
- [12] Irum Inayat, Siti Salwah Salim, Sabrina Marczak, Maya Daneva, and Shahaboddin Shamshirband. A systematic literature review on agile requirements engineering practices and challenges. *Computers in human behavior*, 51:915–929, 2015.
- [13] Ronnie E de Souza Santos and Paul Ralph. A grounded theory of coordination in remote-first and hybrid software teams. In *Proceedings of the 44th International Conference on Software Engineering*, pages 25–35, 2022.
- [14] Justin Lavelle. Gartner cfo survey reveals 74% intend to shift some employees to remote work permanently. *Gartner Newsroom*, 3, 2020.
- [15] Aifric Nolan, Rachel White, Mahjabeen Soomro, Boluwatife Comfort Dopamu, Murat Yilmaz, David Solan, and Paul Clarke. To work from home (wfh) or not to work from home? lessons learned by software engineers during the covid-19 pandemic. In *Systems, Software and Services Process Improvement: 28th European Conference, EuroSPI 2021, Krems, Austria, September 1–3, 2021, Proceedings*, pages 14–33. Springer, 2021.
- [16] Foutse Khomh, Bram Adams, Jinghui Cheng, Marios Fokaefs, and Giuliano Antoniol. Software engineering for machine-learning applications: The road ahead. *IEEE Software*, 35(5):81–84, 2018.
- [17] Andreas Vogelsang and Markus Borg. Requirements engineering for machine learning: Perspectives from data scientists. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 245–251. IEEE, 2019.
- [18] Nadia Nahar, Shurui Zhou, Grace Lewis, and Christian Kästner. Collaboration challenges in building ml-enabled systems: Communication, documentation, engineering, and process. In *Proceedings of the 44th International Conference on Software Engineering*, pages 413–425, 2022.
- [19] Anders Arpteg, Björn Brinne, Luka Crnkovic-Friis, and Jan Bosch. Software engineering challenges of deep learning. In *2018 44th euromicro conference on software engineering and advanced applications (SEAA)*, pages 50–59. IEEE, 2018.

- [20] Hugo Villamizar, Tatiana Escovedo, and Marcos Kalinowski. Requirements engineering for machine learning: A systematic mapping study. In *2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 29–36. IEEE, 2021.
- [21] Henning Femmer. *Requirements engineering artifact quality: definition and control*. PhD thesis, Technische Universität München, 2017.
- [22] Fuyuki Ishikawa and Nobukazu Yoshioka. How do engineers perceive difficulties in engineering of machine-learning systems?-questionnaire survey. In *2019 IEEE/ACM Joint 7th International Workshop on Conducting Empirical Studies in Industry (CESI) and 6th International Workshop on Software Engineering Research and Industrial Practice (SER&IP)*, pages 2–9. IEEE, 2019.
- [23] Boris Scharinger, Markus Borg, Andreas Vogelsang, and Thomas Olsson. Can re help better prepare industrial ai for commercial scale? *IEEE Software*, 39(6): 8–12, 2022.
- [24] Hrvoje Belani, Marin Vukovic, and Željka Car. Requirements engineering challenges in building ai-based complex systems. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 252–255. IEEE, 2019.
- [25] Hiroshi Kuwajima, Hirotoshi Yasuoka, and Toshihiro Nakae. Engineering problems in machine learning systems. *Machine Learning*, 109(5):1103–1126, 2020.
- [26] Jian Huang, Junyi Chai, and Stella Cho. Deep learning in finance and banking: A literature review and classification. *Frontiers of Business Research in China*, 14(1):1–24, 2020.
- [27] M Colangelo. Mass adoption of ai in financial services expected within two years. *Mumbai: Forbes*, 2020.
- [28] Maryan Rizinski, Hristijan Peshov, Kostadin Mishev, Lubomir T Chitkushev, Irena Vodenska, and Dimitar Trajanov. Ethically responsible machine learning in fintech. *IEEE Access*, 10:97531–97554, 2022.
- [29] Todd Sedano, Paul Ralph, and Cécile Péraire. The product backlog. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 200–211. IEEE, 2019.
- [30] Paul Meredith, Peter Summons, Mira Park, and Bruce Cheek. What do employers expect from business analysts and is it captured by the “business analysis body of knowledge”(babok)? 2019.
- [31] Hrafnhildur Sif Sverrisdottir, Helgi Thor Ingason, and Haukur Ingi Jonasson. The role of the product owner in scrum-comparison between theory and practices. *Procedia-Social and Behavioral Sciences*, 119:257–267, 2014.

References

- [32] Matthew Honnibal, Ines Montani, Sofie Van Landeghem, Adriane Boyd, et al. spacy: Industrial-strength natural language processing in python. 2020.
- [33] Fabiano Dalpiaz, Ivor Van Der Schalk, Sjaak Brinkkemper, Fatma Başak Aydemir, and Garm Lucassen. Detecting terminological ambiguity in user stories: Tool and experimentation. *Information and Software Technology*, 110: 3–16, 2019.
- [34] Mike Cohn. *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.
- [35] Marco Brambilla, Jordi Cabot, and Manuel Wimmer. *Model-driven software engineering in practice*. Morgan & Claypool Publishers, 2017.
- [36] Maxim Bragilovski, Fabiano Dalpiaz, and Arnon Sturm. Guided derivation of conceptual models from user stories: A controlled experiment. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 131–147. Springer, 2022.
- [37] Jianzhang Zhang, Yiyang Chen, Nan Niu, and Chuang Liu. A preliminary evaluation of chatgpt in requirements information retrieval. *arXiv preprint arXiv:2304.12562*, 2023.
- [38] Alessandro Fantechi, Stefania Gnesi, Lucia Passaro, and Laura Semini. Inconsistency detection in natural language requirements using chatgpt: a preliminary evaluation. In *2023 IEEE 31st International Requirements Engineering Conference (RE)*, pages 335–340. IEEE, 2023.
- [39] Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR, 2021.
- [40] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.
- [41] Mary Brydon-Miller, Davydd Greenwood, and Patricia Maguire. Why action research?, 2003.
- [42] Mirosław Staron. *Action research in software engineering*. Springer, 2020.
- [43] Claes Wohlin and Austen Rainer. Is it a case study?—a critical analysis and guidance. *Journal of Systems and Software*, page 111395, 2022.
- [44] Martin Kalenda, Petr Hyna, and Bruno Rossi. Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, 30(10):e1954, 2018.

- [45] Thatiany Lima De Sousa, Elaine Venson, Rejane Maria Da Costa Figueiredo, Ricardo Ajax Kosloski, and Luiz Carlos Miyadaira Ribeiro. Using scrum in outsourced government projects: An action research. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 5447–5456. IEEE, 2016.
- [46] Alessio Ferrari, Gloria Gori, Benedetta Rosadini, Iacopo Trotta, Stefano Bacherini, Alessandro Fantechi, and Stefania Gnesi. Detecting requirements defects with nlp patterns: an industrial experience in the railway domain. *Empirical Software Engineering*, 23(6):3684–3733, 2018.
- [47] Davide Falessi, Giovanni Cantone, and Gerardo Canfora. Empirical principles and an industrial case study in retrieving equivalent requirements via natural language processing techniques. *IEEE Transactions on Software Engineering*, 39(1):18–44, 2011.
- [48] Liping Zhao, Waad Alhoshan, Alessio Ferrari, Keletso J Letsholo, Muideen A Ajagbe, Erol-Valeriu Chioasca, and Riza T Batista-Navarro. Natural language processing for requirements engineering: A systematic mapping study. *ACM Computing Surveys (CSUR)*, 54(3):1–41, 2021.
- [49] Davide Falessi, Natalia Juristo, Claes Wohlin, Burak Turhan, Jürgen Münch, Andreas Jedlitschka, and Markku Oivo. Empirical software engineering experts on the use of students and professionals in experiments. *Empirical Software Engineering*, 23:452–489, 2018.
- [50] Nicolò Paternoster, Carmine Giardino, Michael Unterkalmsteiner, Tony Gorschek, and Pekka Abrahamsson. Software development in startup companies: A systematic mapping study. *Information and Software Technology*, 56(10):1200–1218, 2014.
- [51] Eriks Klotins, Michael Unterkalmsteiner, and Tony Gorschek. Software engineering antipatterns in start-ups. *IEEE Software*, 36(2):118–126, 2018.
- [52] Maximilian Zorzetti, Ingrid Signoretti, Larissa Salerno, Sabrina Marczak, and Ricardo Bastos. Improving agile software development using user-centered design and lean startup. *Information and Software Technology*, 141:106718, 2022.
- [53] Usman Rafiq, Sohaib Shahid Bajwa, Xiaofeng Wang, and Ilaria Lunesu. Requirements elicitation techniques applied in software startups. In *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pages 141–144. IEEE, 2017.
- [54] Vebjørn Berg, Jørgen Birkeland, Anh Nguyen-Duc, Ilias O Pappas, and Letizia Jaccheri. Software startup engineering: A systematic mapping study. *Journal of Systems and Software*, 144:255–274, 2018.
- [55] James D Herbsleb and Deependra Moitra. Global software development. *IEEE software*, 18(2):16–20, 2001.

References

- [56] James D Herbsleb, Audris Mockus, Thomas A Finholt, and Rebecca E Grinter. An empirical study of global software development: distance and speed. In *Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001*, pages 81–90. IEEE, 2001.
- [57] James D Herbsleb. Global software engineering: The future of socio-technical coordination. In *Future of Software Engineering (FOSE'07)*, pages 188–198. IEEE, 2007.
- [58] Christof Ebert, Marco Kuhrmann, and Rafael Prikladnicki. Global software engineering: Evolution and trends. In *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)*, pages 144–153. IEEE, 2016.
- [59] Javier Portillo-Rodríguez, Aurora Vizcaíno, Mario Piattini, and Sarah Beecham. Tools used in global software engineering: A systematic mapping review. *Information and Software Technology*, 54(7):663–685, 2012.
- [60] Samireh Jalali and Claes Wohlin. Global software engineering and agile practices: a systematic review. *Journal of software: Evolution and Process*, 24(6):643–659, 2012.
- [61] Denae Ford, Margaret-Anne Storey, Thomas Zimmermann, Christian Bird, Sonia Jaffe, Chandra Maddila, Jenna L Butler, Brian Houck, and Nachiappan Nagappan. A tale of two cities: Software developers working from home during the covid-19 pandemic. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 31(2):1–37, 2021.
- [62] Paul Ralph, Sebastian Baltes, Gianisa Adisaputri, Richard Torkar, Vladimir Kovalenko, Marcos Kalinowski, Nicole Novielli, Shin Yoo, Xavier Devroey, Xin Tan, et al. Pandemic programming: How covid-19 affects software developers and how their organizations can help. *Empirical software engineering*, 25:4927–4961, 2020.
- [63] Darja Smite, Nils Brede Moe, Jarle Hildrum, Javier Gonzalez Huerta, and Daniel Mendez. Work-from-home is here to stay: Call for flexibility in post-pandemic work policies. *Journal of Systems and Software*, 195:111552, 2023.
- [64] Ryan Randy Suryono, Indra Budi, and Betty Purwandari. Challenges and trends of financial technology (fintech): a systematic literature review. *Information*, 11(12):590, 2020.
- [65] Long Chen. From fintech to finlife: The case of fintech development in china. *China economic journal*, 9(3):225–239, 2016.
- [66] Erki Kilu, Fredrik Milani, Ezequiel Scott, and Dietmar Pfahl. Agile software process improvement by learning from financial and fintech companies: Lhv bank case study. In *Software Quality: The Complexity and Challenges of Software Engineering and Software Quality in the Cloud: 11th International Conference, SWQD 2019, Vienna, Austria, January 15–18, 2019, Proceedings 11*, pages 57–69. Springer, 2019.

- [67] Sepehr Sharifi, Daniel Amyot, John Mylopoulos, Patrick McLaughlin, and Ray Feodoroff. Towards improved certification of complex fintech systems—a requirements-based approach. In *2022 IEEE 30th International Requirements Engineering Conference Workshops (REW)*, pages 205–214. IEEE, 2022.
- [68] Lan Cao and Balasubramaniam Ramesh. Agile requirements engineering practices: An empirical study. *IEEE software*, 25(1):60–67, 2008.
- [69] Eva-Maria Schön, Jörg Thomaschewski, and María José Escalona. Agile requirements engineering: A systematic literature review. *Computer standards & interfaces*, 49:79–91, 2017.
- [70] Kim Dikert, Maria Paasivaara, and Casper Lassenius. Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119:87–108, 2016.
- [71] Fabiano Dalpiaz and Sjaak Brinkkemper. Agile requirements engineering with user stories. In *2018 IEEE 26th International Requirements Engineering Conference (RE)*, pages 506–507. IEEE, 2018.
- [72] António MS Ferreira, Alberto Rodrigues da Silva, and Ana CR Paiva. Towards the art of writing agile requirements with user stories, acceptance criteria, and related constructs. In *ENASE*, pages 477–484, 2022.
- [73] Mohsin Ahmed, Saif Ur Rehman Khan, and Khubaib Amjad Alam. An nlp-based quality attributes extraction and prioritization framework in agile-driven software development. *Automated Software Engineering*, 30(1): 7, 2023.
- [74] Garm Lucassen, Fabiano Dalpiaz, Jan Martijn EM van der Werf, and Sjaak Brinkkemper. Improving agile requirements: the quality user story framework and tool. *Requirements engineering*, 21:383–403, 2016.
- [75] Sonja Dimitrijević, Jelena Jovanović, and Vladan Devedžić. A comparative study of software tools for user story management. *Information and Software Technology*, 57:352–368, 2015.
- [76] Yves Wautelet, Samedi Heng, Manuel Kolp, Isabelle Mirbel, and Stephan Poelmans. Building a rationale diagram for evaluating user story sets. In *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, pages 1–12. IEEE, 2016.
- [77] Abhimanyu Gupta, Geert Poels, and Palash Bera. Creation of multiple conceptual models from user stories—a natural language processing approach. In *Advances in Conceptual Modeling: ER 2019 Workshops FAIR, MREBA, EmpER, MoBiD, OntoCom, and ER Doctoral Symposium Papers, Salvador, Brazil, November 4–7, 2019, Proceedings 38*, pages 47–57. Springer, 2019.
- [78] Elizabeth Bjarnason, Michael Unterkalmsteiner, Emelie Engström, and Markus Borg. An industrial case study on test cases as requirements. In *Agile Processes in Software Engineering and Extreme Programming: 16th International*

References

- Conference, XP 2015, Helsinki, Finland, May 25-29, 2015, Proceedings 16*, pages 27–39. Springer, 2015.
- [79] Philipp Diebold, Jan-Peter Ostberg, Stefan Wagner, and Ulrich Zendler. What do practitioners vary in using scrum? In *Agile Processes in Software Engineering and Extreme Programming: 16th International Conference, XP 2015, Helsinki, Finland, May 25-29, 2015, Proceedings 16*, pages 40–51. Springer, 2015.
- [80] Emilio Insfrán, Oscar Pastor, and ROEL Wieringa. Requirements engineering-based conceptual modelling. *Requirements Engineering*, 7:61–72, 2002.
- [81] Neil AM Maiden, Sara V Jones, Sharon Manning, John Greenwood, and L Renou. Model-driven requirements engineering: synchronising models in an air traffic management case study. In *Advanced Information Systems Engineering: 16th International Conference, CAiSE 2004, Riga, Latvia, June 7-11, 2004. Proceedings 16*, pages 368–383. Springer, 2004.
- [82] Tao Yue, Lionel C Briand, and Yvan Labiche. A systematic review of transformation approaches between user requirements and analysis models. *Requirements engineering*, 16:75–99, 2011.
- [83] Yair Wand and Ron Weber. Research commentary: information systems and conceptual modeling—a research agenda. *Information systems research*, 13(4): 363–376, 2002.
- [84] J Mylopoulos. Conceptual modeling, databases, and case: An integrated view of information systems development; chapter conceptual modeling and telos, 1992.
- [85] Tao Yue, Lionel C Briand, and Yvan Labiche. Facilitating the transition from use case models to analysis models: Approach and experiments. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 22(1):1–38, 2013.
- [86] Bente Anda and Dag IK Sjøberg. Investigating the role of use cases in the construction of class diagrams. *Empirical Software Engineering*, 10:285–309, 2005.
- [87] Michel H Fortuna, Cláudia ML Werner, and Marcos RS Borges. Info cases: integrating use cases and domain models. In *2008 16th IEEE International Requirements Engineering Conference*, pages 81–84. IEEE, 2008.
- [88] Daniel Berry, Ricardo Gacitua, Pete Sawyer, and Sri Fatimah Tjong. The case for dumb requirements engineering tools. In *Requirements Engineering: Foundation for Software Quality: 18th International Working Conference, REFSQ 2012, Essen, Germany, March 19-22, 2012. Proceedings 18*, pages 211–217. Springer, 2012.
- [89] Garm Lucassen, Marcel Robeer, Fabiano Dalpiaz, Jan Martijn EM Van Der Werf, and Sjaak Brinkkemper. Extracting conceptual models from user stories with visual narrator. *Requirements Engineering*, 22:339–358, 2017.

- [90] Jennifer Horkoff. Non-functional requirements for machine learning: Challenges and new directions. In *2019 IEEE 27th International Requirements Engineering Conference (RE)*, pages 386–391. IEEE, 2019.
- [91] Khlood Ahmad, Muneera Bano, Mohamed Abdelrazek, Chetan Arora, and John Grundy. What’s up with requirements engineering for artificial intelligence systems? In *2021 IEEE 29th International Requirements Engineering Conference (RE)*, pages 1–12. IEEE, 2021.
- [92] Khlood Ahmad, Mohamed Abdelrazek, Chetan Arora, Muneera Bano, and John Grundy. Requirements engineering for artificial intelligence systems: A systematic mapping study. *Information and Software Technology*, page 107176, 2023.
- [93] Changho Shin, Seungeun Rho, Hyoseop Lee, and Wonjong Rhee. Data requirements for applying machine learning to energy disaggregation. *Energies*, 12(9):1696, 2019.
- [94] Dennis Weihrauch, Paul Anton Schindler, and Wilfried Sihm. A conceptual model for developing a smart process control system. *Procedia CIRP*, 67: 386–391, 2018.
- [95] Koji Nakamichi, Kyoko Ohashi, Isao Namba, Rieko Yamamoto, Mikio Aoyama, Lisa Joeckel, Julien Siebert, and Jens Heidrich. Requirements-driven method to determine quality characteristics and measurements for machine learning software and its evaluation. In *2020 IEEE 28th International Requirements Engineering Conference (RE)*, pages 260–270. IEEE, 2020.
- [96] Benoit Ries, Nicolas Guelfi, and Benjamin Jahic. An mde method for improving deep learning dataset requirements engineering using alloy and uml. In *9th International Conference on Model-Driven Engineering and Software Development*. SCITEPRESS, 2021.
- [97] Mark Hall, Daniel Harborne, Richard Tomsett, Vedran Galetic, Santiago Quintana-Amate, Alistair Nottle, and Alun Preece. A systematic method to understand requirements for explainable ai (xai) systems. In *Proceedings of the IJCAI Workshop on eXplainable Artificial Intelligence (XAI 2019), Macau, China*, volume 11, 2019.
- [98] Barbara Bruno, Fulvio Mastrogiovanni, and Antonio Sgorbissa. Functional requirements and design issues for a socially assistive robot for elderly people with mild cognitive impairments. In *2013 IEEE RO-MAN*, pages 768–773. IEEE, 2013.
- [99] Maike Schwammberger. A quest of self-explainability: when causal diagrams meet autonomous urban traffic manoeuvres. In *2021 IEEE 29th International Requirements Engineering Conference Workshops (REW)*, pages 195–199. IEEE, 2021.

References

- [100] Zhongyi Pei, Lin Liu, Chen Wang, and Jianmin Wang. Requirements engineering for machine learning: A review and reflection. In *2022 IEEE 30th International Requirements Engineering Conference Workshops (REW)*, pages 166–175. IEEE, 2022.
- [101] Fabiano Dalpiaz and Nan Niu. Requirements engineering in the days of artificial intelligence. *IEEE software*, 37(4):7–10, 2020.
- [102] Alessio Ferrari and Stefania Gnesi. Using collective intelligence to detect pragmatic ambiguities. In *2012 20th IEEE International Requirements Engineering Conference (RE)*, pages 191–200. IEEE, 2012.
- [103] Sofija Hotomski and Martin Glinz. A qualitative study on using guidegen to keep requirements and acceptance tests aligned. In *2018 IEEE 26th International Requirements Engineering Conference (RE)*, pages 29–39. IEEE, 2018.
- [104] Laura V Galvis Carreño and Kristina Winbladh. Analysis of user comments: an approach for software requirements evolution. In *2013 35th international conference on software engineering (ICSE)*, pages 582–591. IEEE, 2013.
- [105] Nasir Ali, Haipeng Cai, Abdelwahab Hamou-Lhadj, and Jameleddine Hassine. Exploiting parts-of-speech for effective automated requirements traceability. *Information and software technology*, 106:126–141, 2019.
- [106] Ayat Mashal Almanaseer, Wael Alzyadat, Mohammad Muhairat, Suleyman Al-Showarah, and Aysh Alhroob. A proposed model for eliminating nonfunctional requirements in agile methods using natural language processes. In *2022 International Conference on Emerging Trends in Computing and Engineering Applications (ETCEA)*, pages 1–7. IEEE, 2022.
- [107] Xusheng Xiao, Amit Paradkar, Suresh Thummalapenta, and Tao Xie. Automated extraction of security policies from natural-language software documents. In *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, pages 1–11, 2012.
- [108] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.
- [109] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [110] Preksha Nema, Pauline Anthonysamy, Nina Taft, and Sai Teja Peddinti. Analyzing user perspectives on mobile app privacy at scale. In *Proceedings of the 44th International Conference on Software Engineering*, pages 112–124, 2022.
- [111] Jitendra Singh Thakur and Atul Gupta. Anmodeler: a tool for generating domain models from textual specifications. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, pages 828–833, 2016.

- [112] Tobias Hey, Jan Keim, Anne Kozirolek, and Walter F Tichy. Norbert: Transfer learning for requirements classification. In *2020 IEEE 28th International Requirements Engineering Conference (RE)*, pages 169–179. IEEE, 2020.
- [113] Zhi Jin. Environment-centric self-adaptivity for autonomous systems in smart spaces. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 214–215. IEEE, 2019.
- [114] Jaspreet Bhatia, Pradeep K Murukannaiah, Nan Niu, and Fabiano Dalpiaz. The 6th international workshop on artificial intelligence for requirements engineering (aire’19). In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 213–213. IEEE Computer Society, 2019.
- [115] Anna Perini, Angelo Susi, and Paolo Avesani. A machine learning approach to software requirements prioritization. *IEEE Transactions on Software Engineering*, 39(4):445–461, 2012.
- [116] Siba Mishra and Arpit Sharma. On the use of word embeddings for identifying domain specific ambiguities in requirements. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 234–240. IEEE, 2019.
- [117] Alessio Ferrari, Sallam Abualhaija, and Chetan Arora. Model generation from requirements with llms: an exploratory study. *arXiv preprint arXiv:2404.06371*, 2024.
- [118] Paola Spoletini and Alessio Ferrari. The return of formal requirements engineering in the era of large language models. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*, pages 344–353. Springer, 2024.
- [119] Zhen Yang, Fang Liu, Zhongxing Yu, Jacky Wai Keung, Jia Li, Shuo Liu, Yifan Hong, Xiaoxue Ma, Zhi Jin, and Ge Li. Exploring and unleashing the power of large language models in automated code translation. *arXiv preprint arXiv:2404.14646*, 2024.
- [120] Pengyu Xue, Linhao Wu, Zhongxing Yu, Zhi Jin, Zhen Yang, Xinyi Li, Zhenyu Yang, and Yue Tan. Automated commit message generation with large language models: An empirical study and beyond. *arXiv preprint arXiv:2404.14824*, 2024.
- [121] Fang Liu, Yang Liu, Lin Shi, Houkun Huang, Ruifeng Wang, Zhen Yang, and Li Zhang. Exploring and evaluating hallucinations in llm-powered code generation. *arXiv preprint arXiv:2404.00971*, 2024.
- [122] Yuwei Zhang, Zhi Jin, Ying Xing, and Ge Li. Steam: simulating the interactive behavior of programmers for automatic bug fixing. *arXiv preprint arXiv:2308.14460*, 2023.

References

- [123] Angela Fan, Beliz Gokkaya, Mark Harman, Mitya Lyubarskiy, Shubho Sengupta, Shin Yoo, and Jie M Zhang. Large language models for software engineering: Survey and open problems. *arXiv preprint arXiv:2310.03533*, 2023.
- [124] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.
- [125] Mickel Hoang, Oskar Alija Bihorac, and Jacobo Rouces. Aspect-based sentiment analysis using bert. In *Proceedings of the 22nd nordic conference on computational linguistics*, pages 187–196, 2019.
- [126] Kai Hakala and Sampo Pyysalo. Biomedical named entity recognition with multilingual bert. In *Proceedings of the 5th workshop on BioNLP open shared tasks*, pages 56–61, 2019.
- [127] Hugh A Chipman, Edward I George, and Robert E McCulloch. Bart: Bayesian additive regression trees. 2010.
- [128] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- [129] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.
- [130] Shuo Liu, Jacky Keung, Zhen Yang, Fang Liu, Qilin Zhou, and Yihan Liao. Delving into parameter-efficient fine-tuning in code change learning: An empirical study. *arXiv preprint arXiv:2402.06247*, 2024.
- [131] Xiaoxue Ma, Jacky Wai Keung, Xiao Yu, Huiqi Zou, Jingyu Zhang, and Yishu Li. Attsum: A deep attention-based summarization model for bug report title generation. *IEEE Transactions on Reliability*, 2023.
- [132] Zhen Yang, Jacky Keung, Xiao Yu, Xiaodong Gu, Zhengyuan Wei, Xiaoxue Ma, and Miao Zhang. A multi-modal transformer-based code summarization approach for smart contracts. In *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*, pages 1–12. IEEE, 2021.
- [133] Sandeep Subramanian, Sai Rajeswar Mudumba, Alessandro Sordoni, Adam Trischler, Aaron C Courville, and Chris Pal. Towards text generation with adversarially learned neural outlines. *Advances in Neural Information Processing Systems*, 31, 2018.
- [134] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*, 2022.

- [135] Daniel Fried, Armen Aghajanyan, Jessy Lin, Sida Wang, Eric Wallace, Freda Shi, Ruiqi Zhong, Wen-tau Yih, Luke Zettlemoyer, and Mike Lewis. Incoder: A generative model for code infilling and synthesis. *arXiv preprint arXiv:2204.05999*, 2022.
- [136] Erik Nijkamp, Hiroaki Hayashi, Caiming Xiong, Silvio Savarese, and Yingbo Zhou. Codegen2: Lessons for training llms on programming and natural languages. *arXiv preprint arXiv:2305.02309*, 2023.
- [137] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.
- [138] Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. What language model architecture and pretraining objective works best for zero-shot generalization? In *International Conference on Machine Learning*, pages 22964–22984. PMLR, 2022.
- [139] Yue Wang, Weishi Wang, Shafiq Joty, and Steven CH Hoi. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. *arXiv preprint arXiv:2109.00859*, 2021.
- [140] Daya Guo, Shuai Lu, Nan Duan, Yanlin Wang, Ming Zhou, and Jian Yin. Unixcoder: Unified cross-modal pre-training for code representation. *arXiv preprint arXiv:2203.03850*, 2022.
- [141] Ipek Ozkaya. Application of large language models to software engineering tasks: Opportunities, risks, and implications. *IEEE Software*, 40(3):4–8, 2023.
- [142] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*, 2023.
- [143] Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023.
- [144] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. *arXiv preprint arXiv:2303.17580*, 2023.
- [145] Yihong Dong, Xue Jiang, Zhi Jin, and Ge Li. Self-collaboration code generation via chatgpt. *arXiv preprint arXiv:2304.07590*, 2023.
- [146] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

References

- [147] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.
- [148] Yishu Li, Jacky Keung, Kwabena Ebo Bennin, Xiaoxue Ma, Jingyu Zhang, and Zhen Yang. Agile requirements engineering in a distributed environment: Experiences from the software industry during unprecedented global challenges. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2024.
- [149] Claes Wohlin. Case study research in software engineering—it is a case, and it is a study, but is it a case study? *Information and Software Technology*, 133: 106514, 2021.
- [150] Claudia Melo, Daniela S Cruzes, Fabio Kon, and Reidar Conradi. Agile team perceptions of productivity factors. In *2011 Agile Conference*, pages 57–66. IEEE, 2011.
- [151] Agile Manifesto. Agile manifesto. *Haettu*, 14:2012, 2001.
- [152] Dennis Mancl and Steven D Fraser. Covid-19’s influence on the future of agile. In *International Conference on Agile Software Development*, pages 309–316. Springer, 2020.
- [153] Joaquín Nicolás, Juan M Carrillo De Gea, Bernabe Nicolas, Jose L Fernandez-Aleman, and Ambrosio Toval. On the risks and safeguards for requirements engineering in global software development: Systematic literature review and quantitative assessment. *IEEE Access*, 6:59628–59656, 2018.
- [154] Jonathan Lazar, Jinjuan Heidi Feng, and Harry Hochheiser. *Research methods in human-computer interaction*. Morgan Kaufmann, 2017.
- [155] Alisa Ananjeva, John Stouby Persson, and Anders Bruun. Integrating ux work with agile development through user stories: An action research study in a small software company. *Journal of Systems and Software*, 170:110785, 2020.
- [156] Graeme Shanks and Nargiza Bekmamedova. Chapter 7 - case study research in information systems. In Kirsty Williamson and Graeme Johanson, editors, *Research Methods (Second Edition)*, pages 193–208. Chandos Publishing, second edition edition, 2018. ISBN 978-0-08-102220-7. doi: <https://doi.org/10.1016/B978-0-08-102220-7.00007-8>. URL <https://www.sciencedirect.com/science/article/pii/B9780081022207000078>.
- [157] Matthew B Miles and A Michael Huberman. *Qualitative data analysis: An expanded sourcebook*. sage, 1994.
- [158] Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, et al. Manifesto for agile software development. 2001.

- [159] Angela Carell, Kim Lauenroth, and Dirk Platz. Using design thinking for requirements engineering in the context of digitalization and digital transformation: a motivation and an experience report. In *The Essence of Software Engineering*, pages 107–120. Springer, Cham, 2018.
- [160] Birgit Penzenstadler. When does design help thinking, and when does design thinking help? *IEEE Software*, 37(2):6–9, 2020.
- [161] Elizabeth Bjarnason, Krzysztof Wnuk, and Björn Regnell. A case study on benefits and side-effects of agile practices in large-scale requirements engineering. In *proceedings of the 1st workshop on agile requirements engineering*, pages 1–5, 2011.
- [162] Dat Thanh Nguyen, Dinh Hoang Bach Phan, Tee Chwee Ming, et al. An assessment of how covid-19 changed the global equity market. *Economic Analysis and Policy*, 69:480–491, 2021.
- [163] Zhen Yang, Jacky Keung, Md Alamgir Kabir, Xiao Yu, Yutian Tang, Miao Zhang, and Shuo Feng. Acomnn: Attention enhanced compound neural network for financial time-series forecasting with cross-regional features. *Applied Soft Computing*, 111:107649, 2021.
- [164] Paul Ralph, Sebastian Baltes, Domenico Bianculli, Yvonne Dittrich, Michael Felderer, Robert Feldt, Antonio Filieri, Carlo Alberto Furia, Daniel Graziotin, Pinjia He, et al. Acm sigsoft empirical standards. 2020.
- [165] LI Yishu, Jacky Keung, Kwabena Ebo Bennin, Xiaoxue Ma, Yangyang Huang, and Jingyu Zhang. Towards requirements engineering activities for machine learning-enabled fintech applications. In *2023 30th Asia-Pacific Software Engineering Conference (APSEC)*, pages 121–130. IEEE, 2023.
- [166] Keng Siau and Matti Rossi. Evaluation techniques for systems analysis and design modelling methods—a review and comparative analysis. *Information Systems Journal*, 21(3):249–268, 2011.
- [167] Steve Easterbrook. Empirical research methods in requirements engineering. In *15th IEEE International Requirements Engineering Conference*, 2007.
- [168] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [169] Leon O Chua and Tamas Roska. The cnn paradigm. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):147–156, 1993.
- [170] Robert K Yin. *Case study research: Design and methods*, volume 5. sage, 2009.
- [171] Hamza Hussein Altarturi, Keng-Yap Ng, Mohd Izuan Hafez Ninggal, Azree Shahrel Ahmad Nazri, and Abdul Azim Abd Ghani. A requirement engineering model for big data software. In *2017 IEEE Conference on Big Data and Analytics (ICBDA)*, pages 111–117. IEEE, 2017.

References

- [172] Jan Bosch, Helena Holmström Olsson, and Ivica Crnkovic. It takes three to tango: Requirement, outcome/data, and ai driven development. In *SiBW*, pages 177–192, 2018.
- [173] Ian Sommerville. Software engineering (ed.). *America: Pearson Education Inc*, 2011.
- [174] Organisation Internationale de Normalisation. *Systems and Software Engineering: System Life Cycle Processes*. Organisation internationale de normalisation, 2009.
- [175] Barry Boehm and Alexander Egyed. Software requirements negotiation: some lessons learned. In *Proceedings of the 20th international conference on Software engineering*, pages 503–506. IEEE, 1998.
- [176] Saleema Amershi, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi, and Thomas Zimmermann. Software engineering for machine learning: A case study. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 291–300. IEEE, 2019.
- [177] A Terry Bahill and Steven J Henderson. Requirements development, verification, and validation exhibited in famous failures. *Systems engineering*, 8(1):1–14, 2005.
- [178] Mona Rahimi, Jin LC Guo, Sahar Kokaly, and Marsha Chechik. Toward requirements specification for machine-learned components. In *2019 IEEE 27th International Requirements Engineering Conference Workshops (REW)*, pages 241–244. IEEE, 2019.
- [179] Alfredo Vellido, José David Martín-Guerrero, and Paulo JG Lisboa. Making machine learning models interpretable. In *ESANN*, volume 12, pages 163–172. Citeseer, 2012.
- [180] Yishu Li, Jacky Keung, Zhen Yang, Xiaoxue Ma, Jingyu Zhang, and Shuo Liu. Simac: Simulating agile collaboration to generate acceptance criteria in user story elaboration. *Automated Software Engineering*, 2024.
- [181] Allan Kelly. *The Art of Agile Product Ownership*. Springer, 2019.
- [182] Pallavi Pandit and Swati Tahiliani. Agileuat: A framework for user acceptance testing based on user stories and acceptance criteria. *International Journal of Computer Applications*, 120(10), 2015.
- [183] Jannik Fischbach, Andreas Vogelsang, Dominik Spies, Andreas Wehrle, Maximilian Junker, and Dietmar Freudenstein. Specmate: Automated creation of test cases from acceptance criteria. In *2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*, pages 321–331. IEEE, 2020.

- [184] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [185] Palakorn Achananuparp, Xiaohua Hu, and Xiajiong Shen. The evaluation of sentence similarity measures. In *Data Warehousing and Knowledge Discovery: 10th International Conference, DaWaK 2008 Turin, Italy, September 2-5, 2008 Proceedings 10*, pages 305–316. Springer, 2008.
- [186] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- [187] Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, May 2023. URL https://github.com/openlm-research/open_llama.
- [188] Bill Wake. Invest in good stories, and smart tasks. *Retrieved December*, 13: 2011, 2003.
- [189] Garm Lucassen, Fabiano Dalpiaz, Jan Martijn EM van der Werf, and Sjaak Brinkkemper. The use and effectiveness of user stories in practice. In *Requirements Engineering: Foundation for Software Quality: 22nd International Working Conference, REFSQ 2016, Gothenburg, Sweden, March 14-17, 2016, Proceedings 22*, pages 205–222. Springer, 2016.
- [190] Luigi Buglione and Alain Abran. Improving the user story agile technique using the invest criteria. In *2013 joint conference of the 23rd international workshop on software measurement and the 8th international conference on software process and product measurement*, pages 49–53. IEEE, 2013.
- [191] Erika Halme, Ville Vakkuri, Joni Kultanen, Marianna Jantunen, Kai-Kristian Kemell, Rebekah Rousi, and Pekka Abrahamsson. How to write ethical user stories? impacts of the eccola method. In *International Conference on Agile Software Development*, pages 36–52. Springer International Publishing Cham, 2021.
- [192] Philipp Lombriser, Fabiano Dalpiaz, Garm Lucassen, and Sjaak Brinkkemper. Gamified requirements engineering: model and experimentation. In *Requirements Engineering: Foundation for Software Quality: 22nd International Working Conference, REFSQ 2016, Gothenburg, Sweden, March 14-17, 2016, Proceedings 22*, pages 171–187. Springer, 2016.
- [193] Jefferson Rosa Cardoso, Ligia Maxwell Pereira, Maura Daly Iversen, and Adilson Luiz Ramos. What is gold standard and what is ground truth? *Dental press journal of orthodontics*, 19:27–30, 2014.
- [194] Zedong Peng, Prachi Rathod, Nan Niu, Tanmay Bhowmik, Hui Liu, Lin Shi, and Zhi Jin. Environment-driven abstraction identification for requirements-based

References

- testing. In *2021 IEEE 29th International Requirements Engineering Conference (RE)*, pages 245–256. IEEE, 2021.
- [195] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.
- [196] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.
- [197] Gail M Sullivan and Richard Feinn. Using effect size—or why the p value is not enough. *Journal of graduate medical education*, 4(3):279–282, 2012.
- [198] Jacob Cohen. *Statistical power analysis for the behavioral sciences*. Academic press, 2013.
- [199] Robert Coe. It’s the effect size, stupid. In *British Educational Research Association Annual Conference*, volume 12, page 14, 2002.
- [200] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. *Experimentation in software engineering*. Springer Science & Business Media, 2012.
- [201] Kieran Conboy and Brian Fitzgerald. Toward a conceptual framework of agile methods: a study of agility in different disciplines. In *Proceedings of the 2004 ACM workshop on Interdisciplinary software engineering research*, pages 37–44, 2004.
- [202] Nguyen Thien Khanh, Jirapun Daengdej, and Habibi Husain Arifin. Human stories: A new written technique in agile software requirements. In *Proceedings of the 6th International Conference on Software and Computer Applications*, pages 15–22, 2017.
- [203] Yishu Li, Jacky Keung, Xiaoxue Ma, Chun Yong Chong, Jingyu Zhang, and Yihan Liao. Llm-based class diagram derivation from user stories with chain-of-thought prompting. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2024.
- [204] Sergio España, Marcela Ruiz, and Arturo González. Systematic derivation of conceptual models from requirements models: a controlled experiment. In *2012 Sixth International Conference on Research Challenges in Information Science (RCIS)*, pages 1–12. IEEE, 2012.
- [205] Fabiano Dalpiaz, Patrizia Gieske, and Arnon Sturm. On deriving conceptual models from user requirements: An empirical study. *Information and Software Technology*, 131:106484, 2021.
- [206] Elena Planas and Jordi Cabot. How are uml class diagrams built in practice? a usability study of two uml tools: Magicdraw and papyrus. *Computer Standards & Interfaces*, 67:103363, 2020.

References

- [207] Wesley KG Assunção, Silvia R Vergilio, and Roberto E Lopez-Herrejon. Automatic extraction of product line architecture and feature models from uml class diagram variants. *Information and Software Technology*, 117:106198, 2020.
- [208] Tianshi Cao, Marc Law, and Sanja Fidler. A theoretical analysis of the number of shots in few-shot learning. *arXiv preprint arXiv:1909.11722*, 2019.
- [209] Odd Ivar Lindland, Guttorm Sindre, and Arne Solvberg. Understanding quality in conceptual modeling. *IEEE software*, 11(2):42–49, 1994.