systems

MDPI

Article

Practitioners' Perspectives towards Requirements Engineering: A Survey

Mert Ozkaya [1,*], Deniz Akdur [2], Etem Cetin Toptani [3], Burak Kocak [3] and Geylani Kardas [4]

[1] Computer Engineering Department, Yeditepe University, Istanbul 34755, Turkey
[2] ASELSAN Inc., Ankara 06830, Turkey
[3] DFDS, Istanbul 34746, Turkey
[4] International Computer Institute, Ege University, Izmir 35100, Turkey
* Correspondence: mozkaya@cse.yeditepe.edu.tr

Abstract: In this paper, we discuss the results of our survey among 84 practitioners in order to understand practitioners' perspectives towards requirements engineering. We asked 28 questions to learn the practitioners' motivations, the techniques and technologies used for different activities, practitioners' experiences with customer involvement, and any challenges encountered. Some important results are as follows: the practitioners' top motivations are the precise communication of requirements and analyzing the requirements to detect issues. Most practitioners (i) insist on using natural languages, (ii) specify requirements as the use case and scenario descriptions, (iii) neglect using/transforming requirements for making high-level decisions and reasoning about requirements, (iv) neglect the specifications of quality requirements and their reasoning while considering quality requirements important, and (v) neglect any technologies for facilitating requirements engineering (e.g., meta-modeling technologies, formal verification tools, and advanced tools). Practitioners are challenged by the cost and effort spent in specifying requirements, the omissions of errors, misinterpretations of requirements and their incorrect (manual) transformations, and customers' lack of technical knowledge. With the survey results, practitioners can gain an awareness on the general perspectives, academics can trigger new research addressing the observed issues, and tool vendors can improve their tools with regard to the weaknesses determined.

Keywords: practitioners survey; requirements engineering; specification; analysis; evolution; transformation; customers

Citation: Ozkaya, M.; Akdur, D.; Toptani, E.C.; Kocak, B.; Kardas, G. Practitioners' Perspectives towards Requirements Engineering: A Survey. *Systems* **2023**, *11*, 65. https://doi.org/10.3390/systems11020065

Academic Editor: Ed Pohl

Received: 28 December 2022
Revised: 14 January 2023
Accepted: 24 January 2023
Published: 27 January 2023

1. Introduction

Today, many software systems are either developed but not used, developed over budget and time limits, or do not work as expected. Such issues are considered as a software crisis or chaos [1]. As Standish group indicated that, in their recent chaos reports [2], 56% of the software projects have been developed over budget, 60% of them exceeded the planned time interval, and 44% of them failed to support the desired set of features and functions. Indeed, several big software failure incidents have occurred since the early nineties and led to catastrophic results, causing huge losses of money and human life. Some of the most remarkable ones are the Ariane 5 rocket failure [3], US soldiers' Precision Lightweight GPS Receiver failure [4], Therac-25 radiation therapy machine software failure [5], and Knight Capital Group's trading software failure [6]. Moreover, more recent failures include British Airways software failures [7], Google Plus software failure [8], and Amazon AWS software failure [9]. All such software failures are partially (or even fully) due to the wrong or inadequate applications of requirements engineering in software development projects, which affect the system boundaries, system architecture and design, implementation and testing, and therefore prevent the development of quality software systems on time and within budget.

Requirements engineering has been always considered as an indispensable part of the engineering process and is concerned with the systematic and disciplined application of techniques, technologies, and scientific approaches for gathering requirements and their specifications that can further be analyzed, changed, and even transformed [10–13].

Software requirements can be gathered through different requirements gathering techniques such as observation, interviewing, prototyping, and brainstorming [14]. The gathered requirements can then be specified for such purposes as documentation, versioning, communication among different stakeholders (e.g., customers, developers, testers, and analysts), and analysis. To specify the gathered requirements, different approaches can be preferred. These include using natural languages in a restricted or unrestricted form, office tools (e.g., PowerPoint and Word), de-facto general-purpose modeling languages (e.g., UML [15] and SysML [16]), and any domain-specific modeling languages (DSMLs) (e.g., AADL [17] and any in-house languages). The analysis of requirement specifications is highly crucial for determining the quality of the requirements before using the requirements in other stages of systems development (e.g., design, implementation, and testing) [18,19]. Note, however, that analyzing requirements may not always be possible depending on the approaches used for specification. Whenever the software requirements are specified and analyzed, the next thing to do is to use the requirements and produce some useful artifacts for the system to be developed. That is, the requirement specifications can be transformed into high-level design decisions (e.g., interaction and behavior decisions), skeleton program code, and test scenarios [20].

Requirements also inevitably evolve constantly throughout the development lifecycle due to many needs including the customer or user demands, platform changes, regulation changes, design issues, and time and budget constraints. If the necessary changes cannot be handled correctly by practitioners, this may even result in systems that fail to meet their quality properties and work incorrectly [21,22]. For requirements engineering to be performed effectively, one essential criteria is the involvement of customers [23,24]. With the customer involvement, requirements can be specified, analyzed, changed, and transformed in a way that satisfies the customer needs.

While requirements engineering is highly important for developing quality software systems, it is not clear how practitioners in industry approach requirements engineering. With the current literature, it is very difficult (if not impossible) to understand practitioners' motivations, the applications of different activities (e.g., gathering, specification, evolution, analysis, and transformation), the techniques, languages and tools used in different activities, the challenges faced by practitioners in different activities, and practitioners' relationships with their customers (e.g., the activities that customers are involved in and any challenges). While the survey studies in the literature help in understanding practitioners' experiences on some particular activities, none of those studies cover a diverse set of activities including requirements gathering, specification, change, analysis, and transformation. Indeed, the requirements transformation and analysis are such important activities for requirements engineering that are nevertheless not handled by most of the survey studies. Additionally, while some surveys address practitioners' relationship with customers, it is not easy to understand which particular activities customers are involved in and how challenging it is to work with the customers in those activities.

The goal of this paper is to conduct a practitioner survey so as to understand practitioners' perspectives towards requirements engineering by focusing on practitioners' *(i)* motivations for specifying and changing requirements, *(ii)* experiences with diverse activities of requirements engineering and challenges, and *(iii)* experiences with the customer involvement. We uniquely consider all the gathering, specification, analysis, transformation, and evolution activities of requirement engineering and aim to learn practitioners' knowledge and experiences and the tools and languages used for each activity. We also intend to learn their challenges for each activity.

We strongly believe that the survey results will be highly useful for *(i)* the industries who perform different activities of requirements engineering, *(ii)* tool vendors who develop

languages and tools for requirements engineering, and *(iii)* academia who conduct active research for improving requirements engineering. The industry practitioners could gain some awareness on the practitioners' general perspectives towards the requirements engineering, general trends on the techniques, languages and tools, and any challenges. The tool vendors could determine practitioners' challenges on different activities and improve their existing languages and tools accordingly. Lastly, academia could use the survey results to determine new research questions to be investigated about requirements engineering.

2. Related Work

In this section, we analyze fourteen different empirical studies that surveyed practitioners from different industries with regard to requirements engineering. In Table 1, we show the analysis results of those surveys for a set of concerns that are addressed by our survey study discussed in this paper. Those concerns are *(i)* practitioners' motivation for specifying and changing requirements, *(ii)* practitioners' experiences towards different requirements activities, *(iii)* the techniques, languages, and tools employed in different activities, *(iv)* practitioners' challenges, and *(v)* the customer involvement.

In the rest of this section, we discuss each survey study in a separate paragraph. At the end, we give a summary of all those similar surveys.

In [25], Franch et al. surveyed 153 practitioners to understand practitioners' point of views with regard to the current research on requirements engineering. Franch et al. studied and created short summaries of 435 different papers on requirements engineering. The summaries have been presented to the practitioners via an online survey to obtain their feedback.

In [26], Fernandez et al. surveyed 228 companies from 10 countries to understand practitioners' experiences and challenges on the requirements engineering. To this end, Fernandez et al. designed 33 different questions focusing on different aspects, including how practitioners gather, specify, and change requirements, any standards on requirements engineering followed by the companies, and practitioners' problems on applying requirements engineering and communicating with customers. Fernandez et al. aimed to analyze the problems that practitioners face while performing the requirements engineering in general and the causes of those problems.

In [27], Verner et al. surveyed 143 developers working in the U.S. or Australia so as to understand the developers' thoughts about how requirements engineering impacts on the project success. Verner et al. designed 42 questions categorized into three groups—the questions about the customers and users involved in the developers' projects, the questions about the requirements, and the question about how the development processes are managed. Verner et al. essentially aimed to analyze how the quality of requirements affects the project success and the project management impacts on the requirements quality.

In [28], Solemon et al. surveyed 64 practitioners from software development companies in Malaysia. Solemon et al. aimed to understand the external (e.g., customer related issues) and internal factors that impact on the problems faced while performing the requirements engineering practices. Solemon et al. considered the Capability Maturity Model Integration (CMMI) model to understand how CMMI impacts on the companies dealing with requirements engineering problems.

In [29], Kassab conducted three different surveys between 2003 and 2013, all of which attracted more than 500 practitioners. Kassab's participants are actually the former MSc Software Engineering students in the Masters of Software Engineering degree program at the Penn State Great Valley School of Graduate Professional Studies. In his surveys, Kassab focused on different activities related to requirement engineering, including requirements gathering, analysis, presentation, management, prototyping, tools and effort estimation. Kassab essentially aimed to understand to what extent the techniques and tools preferred by practitioners in those activities change over time.

In [30], Juzdago et al. conducted a survey with 11 different organizations, each of which has five to one hundred employees so as to learn practitioners' experiences

and challenges on requirements engineering from three different perspectives. These are adoption, the source of requirements, and dependability. Adopting new technologies has to do with how practitioners tackle using newly emerged tools and techniques and any challenges that they face. Dependability has to do with practitioners' experiences on managing the quality requirements (e.g., safety and security). The source of requirements has to do with the sources from whom/which practitioners gather the requirements and any challenges faced.

Table 1. The analysis of the similar survey studies.

Similar Works	Year	Survey Summary	Country	# of Res.	Practitioners' Motivation	Activities	Languages, Tools	Challenges	Customers
Franch et al. [25]	2017	- Online survey - Practs thoughts on 435 research papers	General	153	No	No	No	No	No
Fernandez et al. [26]	2017	- Company survey - Practs' challenges - Gathering, spec., changing req.	General	228	No	Ga, spe, evol	No	Yes	Yes
Verner et al. [27]	2005	- Survey - Practs' thoughts - Project man. impact on req. quality	U.S., Australia	143	No	No	No	No	Yes
Solemon et al. [28]	2009	- Survey - Practs' thoughts - Challenges (internal, external) - CMMI impact on companies	Malaysia	64	No	No	No	Yes	Yes
Kassab [29]	2015	- 3 surveys - Techniques, tools usage and changes over time	U.S.	500	No	Ga, spec, ana, man	Yes	No	No
Juzdago et al. [30]	2002	- Survey - Practs' experiences - Adopting new techs., source of reqs., dependability	General	11 *	No	Ga	No	Yes	No
Neill et al. [31]	2003	- Online survey - Req. gathering and spec. - Tools, notations	General	194	No	Ga, spec	Yes	No	No
Carrillo de Gea et al. [32]	2012	- Survey - Tool vendors - Tool capabilities	General	38	No	No	No	No	No
Liu et al. [33]	2010	- Online survey - Success, failure scenarios	China	377	No	Ga, spec	Yes	No	Yes
Nikula et al. [34]	2000	- Interview - Techniques, tools - Practs needs - Industry-Academia	General	12*	Yes	No	Yes	No	No
Memon et al. [35]	2010	- Student survey - Challenges on learning req. eng.	Malaysia	48	No	No	Yes	Yes	No
Jarzębowicz et al. [36]	2019	- Online survey - Practs' thoughts - Req. eng. and agile development	Poland	69	No	Spec.	Yes	No	Yes

Table 1. *Cont.*

Similar Works	Year	Survey Summary	Country	# of Res.	Practitioners' Motivation	Activities	Languages, Tools	Challenges	Customers
Agren et al. [37]	2019	- Interview - Automative - Practs thoughts - Req. eng. impacts on automative software	General	20	No	No	No	No	No
Palomares et al. [38]	2021	- Interview - Practs thoughts - Roles involved in gathering - Gathering challenges	Sweden	24	No	Ga	Yes	Yes	Yes
Our Work	2023	- Survey - Practs' thoughts - Several activities - Techniques, tools, languages in activities - Activity challenges - Customers	General	84	Yes	Ga, spe, ana, trac, evol, tran	Yes	Yes	Yes

Ga: Gathering / Elicitation, Spe: Specification, Ana: Analysis, Trac: Traceability, Man: Management, Evol: Evolution, Tran: Transformation, Practs: Practitioners, Req. Eng.: Requirements Engineering, *: Companies/Organizations, Req.: Requirements, Res.: Responses.

In [31], Neill et al. surveyed 194 practitioners so as to understand their knowledge and experiences on requirements gathering and specifications. Neill et al.'s survey consists of 22 questions on the techniques, tools, and notation used for the requirements gathering and specifications. Neill et al. aimed to understand the techniques used for requirements gathering and any modeling notations used for requirement specifications.

In [32], Carrillo de Gea et al. surveyed 38 different tool vendors that provide tools for requirements management so as to learn the capabilities of their toolset. To this end, Carrillo de Gea et al. used the ISO/IEC TR 24766:2009 framework, which provides the features for a requirements engineering tool, and designed a set of questions to be sent to the tool vendors.

In [33], Liu et al. surveyed 377 practitioners from China so as to understand their perspectives towards requirements engineering. Liu et al. focused on the techniques used for requirements gathering and specifications. Additionally, Liu et al. further asked the practitioners to learn their success and failure scenarios in applying the requirements engineering activities.

In [34], Nikula et al. interviewed 12 companies so as to determine the improvement needs for requirements engineering from the perspective of industry. Nikula et al. focused on the techniques and tools for requirements specification and management. Nikula et al. further focused on the general development needs for requirements engineering and industry's expectations from academia.

In [35], Memon et al. surveyed 48 students who have been educated in Malaysian universities and took the requirements engineering course. Memon et al. intended to understand the problems that the students face while learning the basics of requirements engineering and their suggestions for improving the requirements engineering courses.

In [36], Jarzębowicz et al. surveyed 69 practitioners from the Polish IT industry so as to understand practitioners' approach towards requirements engineering in agile software development projects. Jarzębowicz et al. considered the source of requirements in agile projects, customers' involvement in the agile projects, and the techniques used for documenting requirements.

In [37], Agren et al. conducted two-stage interviews with practitioners from the automotive domain so as to understand the impacts of requirements engineering on increasing the development speed of the automative software. In the first stage, Agren et al. conducted semi-formal interviews with 20 different practitioners, and the collected

data have been validated in the second-stage together with 12 practitioners. Agren et al. focused on how the current techniques employed for requirements engineering affect the development effort, any new techniques that could help reducing the development effort, and how the agile development process supports all those.

In [38], Palomares et al. conducted interviews with 24 practitioners from 12 Swedish IT companies. Palomares et al. aimed to understand practitioners' perspectives towards the requirements elicitation and focused more on the techniques employed for the requirements elicitation, practitioners' challenges, and the different roles that are involved in the elicitation processes (e.g., customers).

Summary

As shown in Table 1, none of the analyzed survey studies address all the concerns that are fully addressed in our study. Indeed, our survey study is considered to be unique, which sheds light on the reasons that encourage practitioners to specify and change requirements, practitioners' experiences on *(i)* a number of activities (gathering, specification, analysis, transformation, and evolution), *(ii)* any modeling languages and tools used for each activity, *(iii)* any challenges faced during each activity, and *(iv)* the customer involvement.

None of the existing surveys focus on understanding what motivates (or demotivates) practitioners for specifying and changing the software/system requirements—the only exception here is Nikula et al.'s survey [34]. Concerning the requirements engineering activities, none of the existing surveys consider the requirements transformation activity, which we believe is highly important for understanding practitioners' experiences on using the requirements for producing some other important artifacts including the high-level design decisions (e.g., system interaction and behaviors), skeleton program code, and test scenarios. Additionally, none of the existing surveys consider the analysis of requirement specifications for detecting errors early on, which we consider as another limitation of the existing studies. The only exception here is Kassab's work [29], which, however, does not give any information about practitioners' choice of performing manual or automated analysis and any challenges faced during the requirements analysis that our work considers as well.

Concerning the tools and languages employed, the existing survey studies either show the tools and languages used in general or the tools used for a particular activity only. It is not possible to understand what languages and tools practitioners use for the specification, analysis, transformation, and evolution activities. Additionally, one cannot easily understand any challenges faced regarding practitioners' language and tool usages and their experiences with different activities. Concerning the customer involvement, while a few studies asked some questions about the customer involvement, it is not easy to determine which activities of the requirements engineering customers are made to be involved in and any challenges faced with.

3. Research Methodology

We followed the online survey method in our study and intended to collect the responses online and analyze them in the quickest way possible [39].

3.1. Research Questions

In our survey, we investigate the following four research questions so as to achieve the paper goal introduced in Section 1.

RQ1: What motivate practitioners for specifying and changing the software requirements? In this research question, the goal is to understand the reasons that impact on the practitioners' motivation for specifying requirements and changing them later on. To this end, we aim to learn *(i)* any reasons that make practitioners specify and change requirements, *(ii)* the types of requirements that practitioners prefer to specify, *(iii)* the types of concerns that practitioners prefer to address in their requirements specifications.

RQ2: What techniques and technologies do practitioners use for performing different activities of requirements engineering? In this research question, the goal is to understand for each activity considered in our survey—i.e., gathering, specification, analysis, evolution, and transformation—any techniques and technologies used by the practitioners. To this end, we aim to learn any *(i)* techniques used for the requirements gathering, *(ii)* modeling approaches used for the requirement specifications, and *(iii)* technologies (e.g., languages and tools) used for changing (i.e., evolving), analyzing, and transforming requirements. To the best of our knowledge, there is no requirements modeling language or management tool used by practitioners for performing all the activities considered. Therefore, we decided to consider the languages and tools for each activity separately.

RQ3: What are the challenges that practitioners face in different activities of requirement engineering? In this research question, the goal is to understand any difficulties that practitioners face while performing the activities of requirements engineering. To obtain precise results, we aim to learn the challenges for different activities separately rather than learning the challenges in general.

RQ4: To what extent are the customers involved in the requirement engineering? In this research question, the goal is to understand to what extent practitioners involve customers in their requirements engineering activities and the activities that the customers are more or less involved with. We are also interested in learning any challenges that practitioners face while involving customers.

3.2. Survey Design

To design our survey, we *(i)* used our expertise and experiences on the requirements engineering and designing surveys [40–43], *(ii)* went through similar surveys conducted in the past, which are discussed in Section 2, *(iii)* examined the well-known guidelines on requirements engineering (e.g., [11,12]) and conducting surveys (e.g., [44]). Finally, we ended up with a draft of survey questions.

The survey draft starts with an introduction section, which introduces to any potential participants what the survey is about and the expected time that the survey takes. Then, if the participant agrees to continue, a profile section is presented to the participant, which consists of a set of questions for learning the participants' demographic information. Then, we provide a list of sections that each consists of a set of relevant questions for *(i)* each activity of requirements engineering (i.e., gathering, specification, analysis, transformation and evolution) and *(ii)* the customer involvement. The questions in the survey sections have been designed in such a way that the research questions described in Section 3.1 are addressed completely.

To improve our survey draft, we conducted a pilot study together with a group of practitioners and academics. Firstly, we sent our survey draft to two academics who conduct active research on empirical software engineering. With the feedback received, we revised the survey questions and research questions to ensure that the survey questions are complete and consistent with the research questions and well-structured into meaningful sections. Furthermore, we obtained some valuable feedback on the answer list of the questions and were directed to the well-known papers that could support the questions. In the second stage of our pilot study, we asked 10 different practitioners who are involved in the development of software systems in one of the largest logistics companies in the world, called DFDS [45], to share their feedback. To minimize any biases, we carefully selected the practitioners holding different positions including analysts, software developers, testers, scrum masters, and project managers. We asked the participants to fill in the survey questions and note any comments or issues that they spotted. So, we got valuable feedback about the ambiguous questions and the incomplete answer lists for the questions that could further be improved from the practical perspectives. We also obtained feedback on the questions' answer types, which can be either multiple-choice, single-choice, or free-text, and the average time that is required to fill in the survey.

Performing the pilot study on the survey draft, we finalized the questions as shown in Table 2. So, the survey consists of different types of questions depending on the answer types. While some questions are single-answered (Yes/No), some are multiple-answered and some are free-text questions. The single answer questions (Q4, Q6, Q12, Q15, Q17, Q19, and Q24) have been intended for learning precise data from the participants, e.g., the year of experiences, the frequency of performing some activities, etc. For instance, the question "Do you specify the requirements of any software system to be developed?" is answered with either one of the following answers: *always (100%)*, *much of the time (>70%)*, *often (>=50%)*, *sometimes (<50%)*, and *never (0%)*. The multiple-answer questions are supplemented with a pre-defined list of answers and further enable participants to type their own answers freely if the existing list of answers does not include the participants' possible answer(s). The multiple-answer questions are those for learning the techniques, languages, and tools used in different activities (Q8, Q10, Q11, Q20, Q21, Q25–26), the challenges faced in different activities (Q14, Q23, Q28), and concerns that motivate the participants (Q7, Q9, Q16). To determine the pre-defined answer lists for the multiple-answer questions, we considered well-regarded books and articles in the relevant areas and came up with a list of possible answers that have later on been revised and improved during the pilot study. The different types of requirements in Q8 have been determined from Lethbridge et al.'s seminal book on software engineering [14]. The different types of concerns in Q10 have been determined from Rozanski et al.'s book on viewpoint descriptions [46]. The different analysis goals in Q21 have been determined from Zowghi et al.'s highly-cited article on requirements analysis and validation [19]. The different approaches for specifying requirements in Q11 have been determined from Taylor et al.'s book on software architectures [47]. Note that the answer lists of the rest of the multiple-answer questions herein have been determined using our expertise in the field. Lastly, some of the survey questions (Q18, Q22, and Q27) ask the participants to type their own answers freely. Our free-text questions are the ones that ask for any languages or tools used by the participants. Since many alternative languages and tools are available, we did not want to provide participants with a huge list and rather asked the participants to type the names of the languages and tools that they use freely, as also suggested in the pilot study. The free-text answers provided by the participants are analyzed in detail using the coding strategy [48]. That is, each answer is initially checked to determine whether the answer makes sense for the question, and any answers that are difficult to understand are omitted. Those free-text answers that sound close to any of the answers given in the question's answer list (if any) are counted for that pre-determined answer of the list. Otherwise, the free-text answer is considered as a new answer to that question.

Table 2. The survey questions.

Res. Que.	Survey Questions	Multiple Answers	Free Text	Single Answer
Profile	1- Which industry(ies) do you work in?	X	X	
	2- What is (are) your current job position(s)?	X	X	
	3- What is your bachelor degree?	X	X	
	4- How many years of experience do you have in software development?			X
	5- Which software process models do you use for developing software systems?	X	X	
	6- Do you specify the requirements of any software system to be developed?			X
RQ1	7- Please rate the importance of the following motivations for the requirements specification for you.	X		
RQ1	8- Which type(s) of requirements do you focus on?	X	X	

Table 2. *Cont.*

Res. Que.	Survey Questions	Multiple Answers	Free Text	Single Answer
RQ1	9- Which of the following concerns are important for you in the requirements specification?	X	X	
RQ2	10- How do you gather the requirements to be specified?	X	X	
RQ2	11- Which approach(es) do you use for specifying the requirements?	X	X	
RQ4	12- How often do you involve customers in the requirements specification process?			X
RQ4	13- What aspect(s) of the requirements engineering do you involve the customers in?	X	X	
RQ4	14- What are the challenges that you face with while involving customers in the requirements specification?	X	X	
RQ2	15- How often do you need to change the requirements during the software development lifecycle?			X
RQ1	16- What makes you change requirements?	X	X	
RQ2	17- Do you use any traceability tools for determining the affected parts (e.g., design, code, test artefacts) upon changing a requirement?			X
RQ2	18- If you use any tools for changing and tracing requirements, please tell us which tools you use.		X	
RQ2	19- How often do you analyse the requirements during the software development lifecycle?			X
RQ2	20- How do you analyse the requirements?	X	X	
RQ2	21- Which of the following analysis goals are important for you?	X		
RQ2	22- If you use languages and tools for the requirements analysis, please give the language and tool names and describe how you use them for the requirements analysis.		X	
RQ3	23- What are the challenges you face with while analysing the requirements?	X	X	
RQ2	24- How often do you transform the requirements into some other artefact?			X
RQ2	25- What would you like to produce with the requirements transformation?	X	X	
RQ2	26- How do you perform the requirements transformation?	X	X	
RQ2	27- If you use language and tools for the requirements transformation, please describe which language(s) and tool(s) you use and how.		X	
RQ3	28- What are the challenges that you face with while transforming the requirements?	X	X	

3.3. Survey Execution

We made our survey available online via *Google forms*, which was accessible for 3 months in between December 2021 and March 2022. We shared the survey link with several people who may be involved in the activities of requirements engineering in diverse industries. In executing the survey, we initially used our personal contacts with whom we have collaborated on different occasions including the consultancy services, R&D projects with many industrial partners from Europe (e.g., ITEA Cluster programme), and past/present work experiences in different industries and countries. So, we sent e-mails to the groups of four different R&D projects and nearly 100 different practitioners whom we know individually.

We also e-mailed the practitioners whom we identified via Google Scholar and who contributed to the well-regarded conference and journal papers that are associated with many relevant topics such as requirements engineering, software and systems engineering, modeling, modeling languages, modeling toolsets, agile development applications/adaptations, etc.

Additionally, we posted messages to several active mailing-lists to attract further participation. These lists are the Eclipse modeling platforms (e.g., sirius-dev, graphiti-dev, papyrus-rt-dev, emf-dev, emft-dev, agileuml-dev, papyrus-ic, etc.), Netbeans, IEEE architecture description, and AADL. Another source of participants is social media platforms such as Linkedin. So, we shared our survey on several Linkedin groups (e.g., agile development, scrum, IT professionals, project management, modeling and modeling language, software testing, and software and systems engineering groups) to reach as many participants as possible. According to our observations, Linkedin particularly attracted many practitioners from diverse industries, where our posts received many likes and shares.

3.4. Survey Sampling

We were not able to reach every single stakeholder who is involved in requirements engineering activities in different industries. Therefore, we decided to perform the non-probability sampling technique [49]. To this end, we initially attempted reaching our personal contacts by e-mail, as we believed that this is the quickest way to attract participants to our survey. However, due to the non-random way of selecting participants, this could cause biases. So, we intended to mimic the probability sampling by sharing our survey link across some actively used mailing lists and Linkedin groups. By doing so, the survey could be reached by anyone who may have an interest in participating in the survey, and each participant who are enrolled in those mailing lists and Linkedin groups has an equal chance to participate as they view our post.

3.5. Data Analysis and Validation

After collecting data via our survey, we proceeded with the analysis of those data to eliminate any misleading ones. We have collected 95 different responses via Google Forms, which are made available online [50]. Among those responses, four responses did not include any data for the survey questions, which indicates that the corresponding participants submitted the survey form without filling it in. We therefore eliminated those four responses from our dataset. Additionally, we noticed that three participants answered some of the profile questions and then refused to answer the technical parts of the survey. We eliminated those three responses as well. So, we ended up with 88 different responses in our dataset. Note that among the 88 responses, four participants answered question 6 (see Table 2) with the "Never" choice, which thus indicates that those participants never specify the requirements of systems. To minimize any biases, we made the survey direct those participants with no experience in requirements engineering to submit the survey form without answering any questions. After the data validation, we ended up with 84 acceptable responses. Note also that we did not have the chance for offering any incentives for attracting participants to the survey, and thus we think that the number of participants reached here is not bad at all.

To analyze the 84 participants' response data, we performed the voting method and calculated the voting for each question's answers. By doing so, we were able to draw statistical inferences using the Google Forms and MS Excel tools. It should be noted that most of the similar surveys that we discussed in Section 1 also followed the same voting approach to analyze the survey data. We strongly believe that the results that we obtained and discussed in Section 4 aid in understanding practitioners' thoughts and perspectives on requirements engineering.

4. Survey Results

4.1. Profile

4.1.1. Work Industries

As shown in Figure 1, IT & telecommunications is the top-selected industry among the participants (35%), which is followed by the computer manufacturing (33%) and software outsourcing (15%) industries.

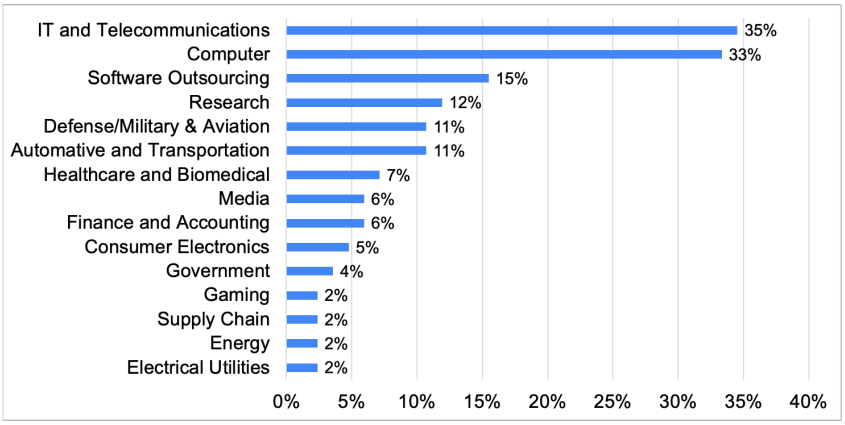


Figure 1. Participants' work industries.

4.1.2. Job Positions

As shown in Figure 2, the top job position is the software developer/programmer (44%), which is followed by the software architects (23%) and systems engineers (18%).

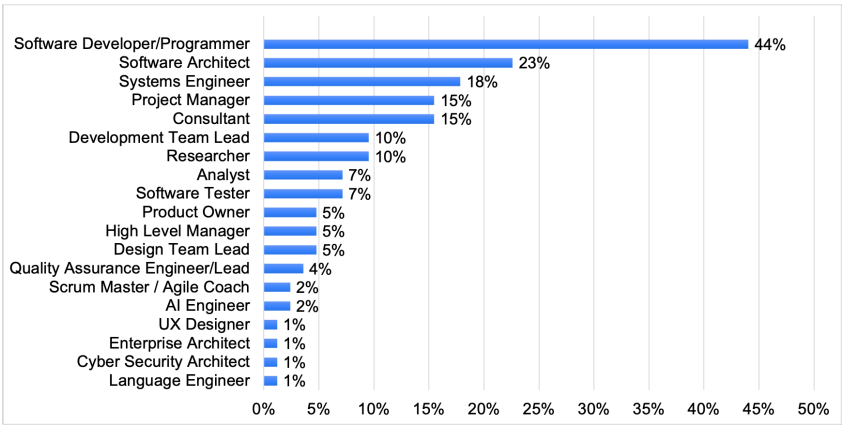


Figure 2. Participants' job positions.

4.1.3. Bachelor Degrees

As shown in Figure 3, the top-selected bachelor degree is computer engineering (51%), followed by the software engineering (16%) and electrical and electronics engineering (12%) degrees.

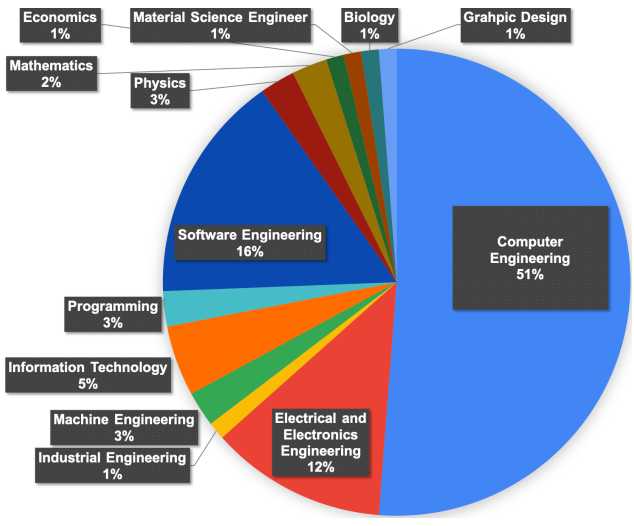


Figure 3. Participants' bachelor degrees.

4.1.4. Years of Experience

As shown in Figure 4, the greatest portion of the participants (43%) have 10+ years of experiences on software development, which is followed by the participants with less than 2 years of experiences (20%) and 2–5 years of experiences (20%).

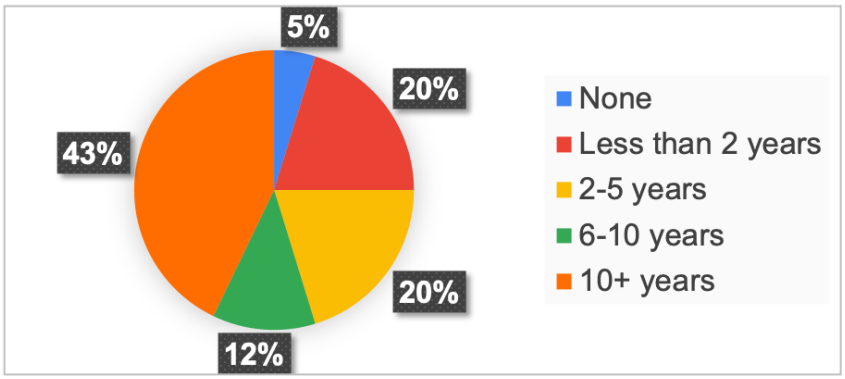


Figure 4. Participants' years of experience on software development.

4.1.5. Software Process Model

Figure 5 shows that agile software development is by far the most preferred software development process model by the practitioners (77%). The agile model is followed by the waterfall process model (35%).

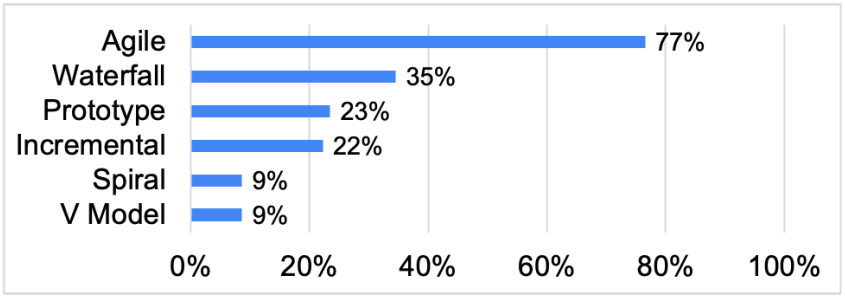


Figure 5. The software process models preferred by the participants.

4.1.6. The Frequency of Requirements Specification

As shown in Figure 6, many participants (64%) frequently specify the software requirements—19% chose "always" and 45% chose "much of the time". Note that those participants who never specify software requirements (5%) have been directed to submit the survey without filling in the rest of the questions.

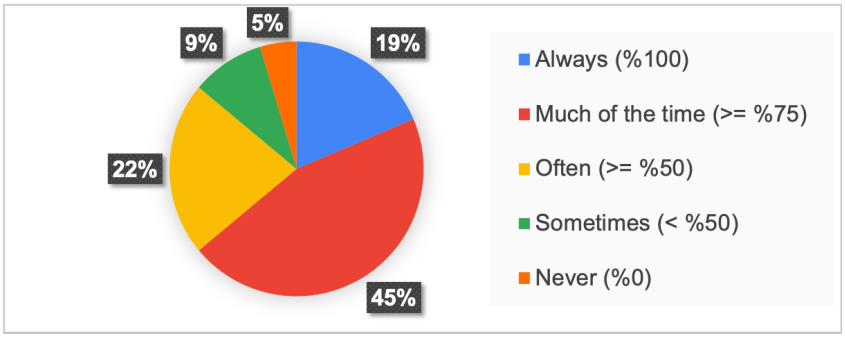


Figure 6. The participants' frequencies of requirements specifications.

4.2. Participants' Motivations for Specifying the Software Requirements

Figure 7 shows six different concerns that can motivate the participants for specifying software requirements. We provide a separate chart for each concern where the responses are displayed with their percentages and the mean value is displayed at the bottom of the chart using a red-colored circle.

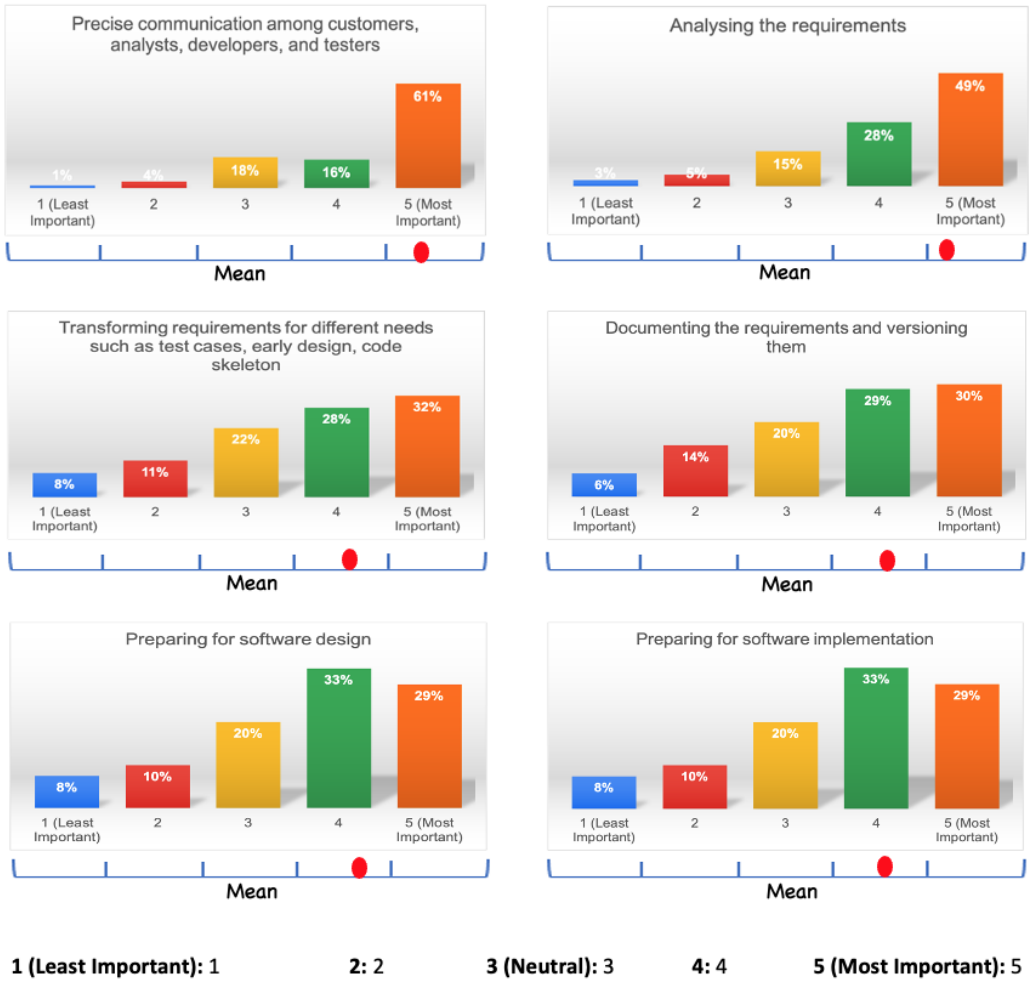


Figure 7. Participants' motivations for specifying the software requirements.

The results here reveal that "the precise communication of the requirements among different stakeholders" is the top-motivating factor for the participants, as the mean value here is in the middle of the "5 (Most Important)" interval. Also, "analysing requirements" is found to be very motivating, given its mean value located in the beginning of the "5 (Most Important)" interval. So, most participants put their greatest emphasis on specifying requirements in a non-ambiguous and precise way that can be understood clearly and reasoned for detecting issues (e.g., incomplete, inconsistent, incorrect requirements) early on.

The rest of the the concerns are transforming requirements, documenting and versioning requirements and using the requirements to prepare for the software design and implementation. While those concerns are also highly important for many participants, their mean value is not inside "5 (Most Important)" and rather remains inside "4".

4.2.1. The Types of Requirements

In this question, we aimed for learning what type(s) of requirements motivate practitioners more and less. To this end, we considered Lethbridge et al.'s seminal book on software engineering [14], which categorized requirements into four types—i.e., functional,

quality, process, and platform. As shown in Figure 8, almost all the participants (98%) prefer to specify the functional requirements of software systems. Surprisingly, the ratio of the participants who prefer to specify the quality (i.e., non-functional) requirements is quite high (64%).

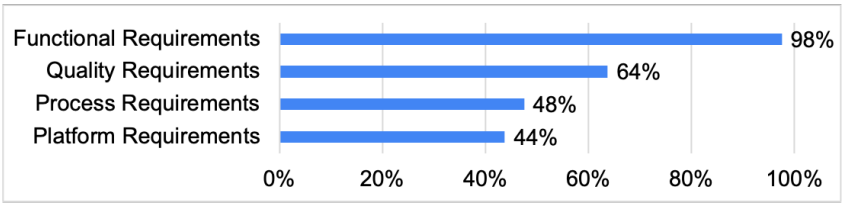


Figure 8. The types of requirements preferred by the participants.

4.2.2. The Requirements Specification Concerns

In this question, we aimed to learn what concern(s) of software systems are important for the participants and addressed in their requirement specifications. To this end, we offered a list of possible concerns that are inspired from Rozanski et al.'s viewpoint descriptions [46], which provide a nice separation of concerns (aka viewpoints) that can be addressed in the requirement specification and analysis of any software systems. Therefore, our list of concerns include the use-case related concerns, scenario related concerns (e.g., test scenarios), system interactions (e.g., protocols of interactions), system behaviors (e.g., how the component state changes depending on events occurring), information (e.g., data structures, data flow, data state changes), structures (systems' decomposition into logical elements), deployment (e.g., mapping physical elements with the logical elements), development (e.g., source file structures and source-code structures), and operational concerns (administering, installing/uninstalling, and operating systems).

As Figure 9 shows, the top-two concerns that have been selected by many of the participants are *(i)* use-case (70%) and *(ii)* scenario (e.g., test scenarios that could be used for testing the system to be developed) (65%). So, many participants specify requirements so as to document the use-cases of their systems and the scenario descriptions. Note that the rest of the concerns that are to do with using the gathered requirements and making high-level decisions about the system to be developed have been selected by less than half of the participants.

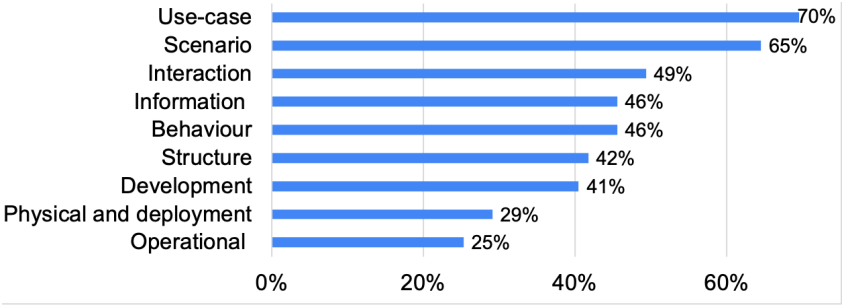


Figure 9. The concerns that the participants consider in their requirements specifications.

4.3. Requirements Gathering

Techniques Used for Gathering Requirements

As shown in Figure 10, the participants' most preferred techniques for gathering the requirements are the use-cases provided by the customers (60%) and performing surveys and interviews with the customers (54%).

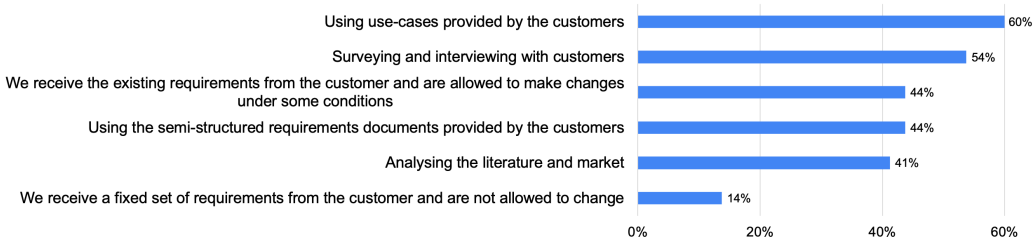


Figure 10. The requirements gathering techniques preferred by the participants.

4.4. Requirements Specifications

The Modeling Approaches Used for the Requirements Specifications

In this question, we focused on Taylor et al.'s categorization of modeling approaches into natural languages, PowerPoint-style modeling, and modeling languages [47]. As Figure 11 shows, the most preferred technique for specifying requirements is natural languages, which could be used in different formats (e.g., unrestricted natural languages, templates, etc.). Overall, 35% the participants only use natural languages, while 29% of the participants use natural languages together with the modeling languages and 13% of the participants use natural languages with the PowerPoint-style approaches. Note that 16% of the participants indicated that they use natural languages, PowerPoint, and modeling languages together.

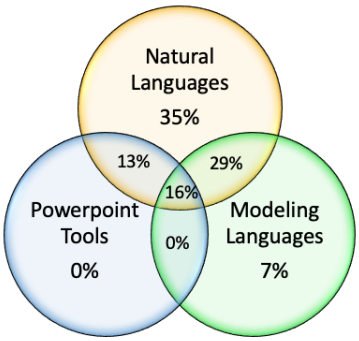


Figure 11. The modeling approaches used by the participants for the requirements specifications.

4.5. Customer Involvement

4.5.1. The Frequency of Involving Customers

As shown in Figure 12, more than half of the participants are highly interested in involving customers in their requirements engineering activities—29% chose "always" (100%) and another 26% chose "much of the time" (≥75%).

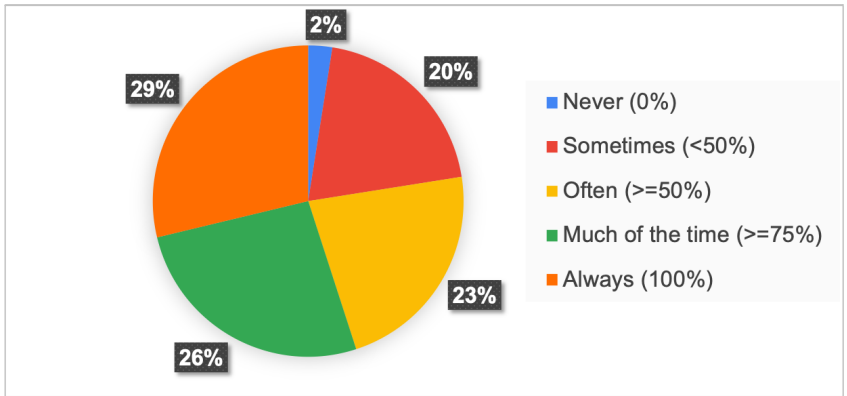


Figure 12. The participants' frequency for involving customers in their software requirements engineering activities.

4.5.2. Customers' Involvement in the Requirements Engineering Activities

As shown in Figure 13, the top activity in which the customers are involved is requirements gathering (82%), and that is followed by the requirements analysis activity (56%). Some participants (29%) indicated that their customers are involved in the requirements modeling, which is to do with specifying different types of requirements using different types of modeling approaches (e.g., natural languages and modeling languages) and addressing different concerns (e.g., use-case, scenarios, system interaction, and system behavior). Lastly, 22% of the participants involve the customers in the requirements transformation, which could be intended for different purposes, e.g., *(i)* producing useful artifacts including test scenarios and program code, and *(ii)* making high-level design decisions and reasoning about the requirements.

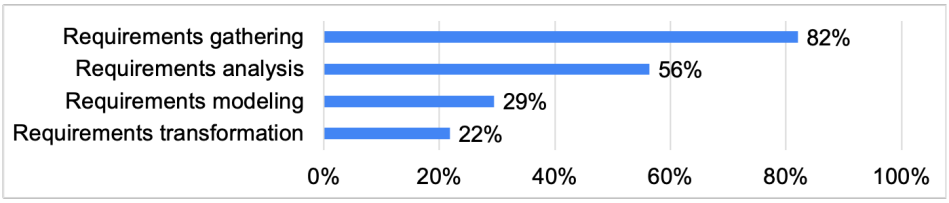


Figure 13. The requirements engineering activities that the customers are involved in.

4.5.3. Participants' Challenges on Involving Customers

As shown in Figure 14, the participants' top challenges are that *(i)* customers are not so willing to be involved due to the technical knowledge required in the requirements engineering activities (53%), and *(ii)* customers possibly do not easily understand the precise requirement specifications that are described with some modeling languages (51%). Some participants (42%) pointed out the challenge about the lack of a precise requirements specification language, which causes the customers and participants to misunderstand each other. A few of the participants (24%) face the challenge of a lack of transformation tool support that can transform the customer requirements into precise requirement specifications automatically for different purposes such as requirements analysis and transformation (e.g., test scenario generation and code generation). Some participants pointed out other challenges (i.e., used the free-text option). That is, a few of the participants complain that customers cannot separate an adequate amount of time for the requirements engineering activities due to such reasons as the management pressure. Additionally, some participants are concerned about some other issues including the customers' lack of domain knowledge, customers' tendency for changing the requirements regularly during their involvement, and the lack of traceability tools for tracing the impacts of changes made together with customers.

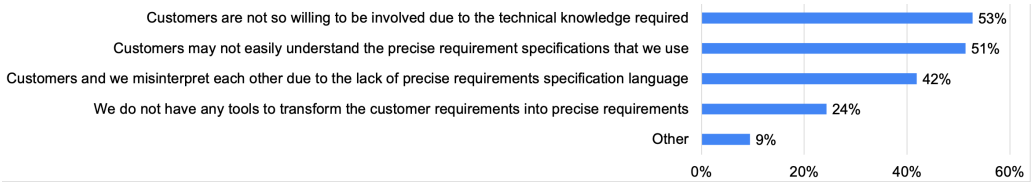


Figure 14. The challenges that the participants face while involving customers in their requirements engineering activities.

4.6. Requirements Evolution

4.6.1. The Frequency of Changing the Software Requirements

As Figure 15 shows, many participants do not frequently change the requirements throughout the software development life-cycle. Indeed, just 34% of the participants frequently change the software requirements, where 14% "always" change requirements and 20% do so "much of the time" ($\geq$75%).

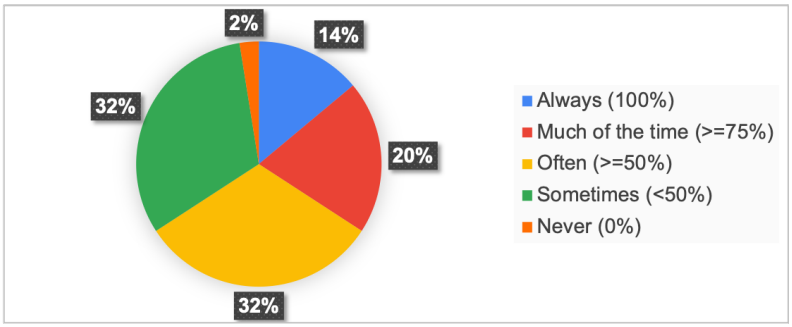


Figure 15. Participants' frequency of changing the software requirements.

4.6.2. Participants' Reasons for Changing the Software Requirements

In this question, we aimed to understand the source of changes by questioning the participants who stated in the previous question that they make changes to the requirements. As Figure 16 shows, 81% of those participants stated that the requirement changes are due to the customers who ask to change the requirements (e.g., adding/removing functionalities, support for new platforms or technologies, new/changed quality expectations, etc.). The second top reason has to do with the changes on the priorities that may occur throughout the software development life-cycles due to any technical or non-technical reasons (64%). The rest of the possible reasons listed in the question's answer list has been selected by less than or equal to 40% of the participants.

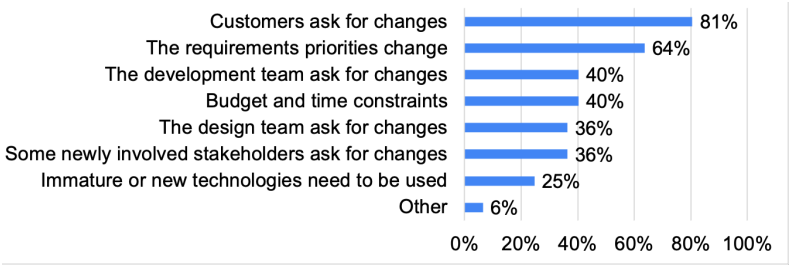


Figure 16. Participants' reasons for changing the software requirements.

A few participants pointed out some other reasons for changing the requirements, which are not given the question's answer list (i.e., the free-text typing). These are *(i)* corporate, governmental rule or policy changes, *(ii)* imprecise specifications of requirements at the beginning, and *(iii)* the technical limits (e.g., the lack of technical knowledge to design and implement the system for a particular requirement).

4.6.3. Participants' Usage Frequencies for the Traceability Tools

As Figure 17 shows, 40% of the participants "never" use any tools for tracking the changes made on the software requirements. Just 25% of the participants frequently use some traceability tools (10% chose "always" and 15% chose "much of the time").

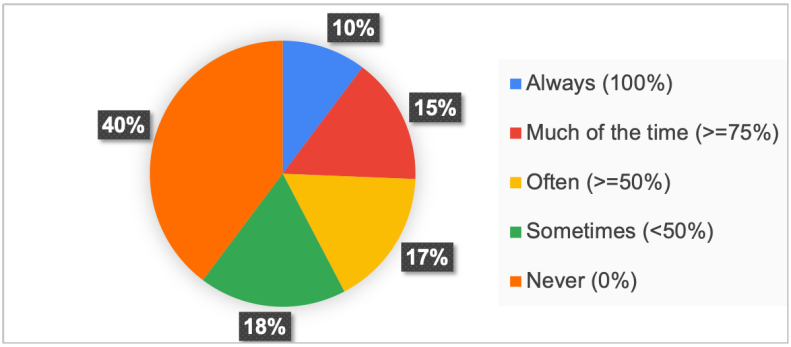
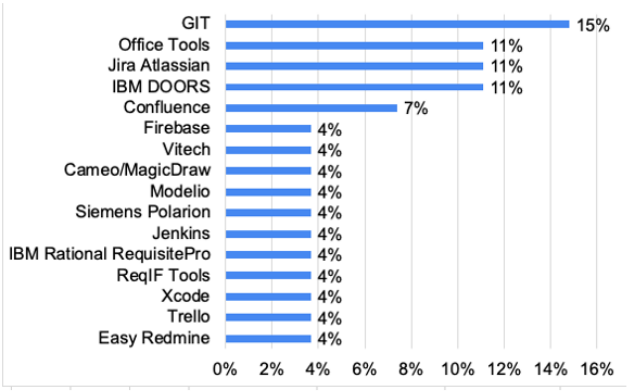


Figure 17. The participants' usage frequencies of the traceability tools for the requirements changes.

4.6.4. The Tools Used by the Participants for Tracing the Requirements Changes

Figure 18 shows the different tools used by the participants who stated in Figure 17 to use any traceability tools for the requirement changes. Some participants use the popular requirements management and traceability tools such as IBM Doors, Vitech, Siemens Polarion, and IBM Rational RequisitePro, which provide advanced features for specifying requirements, establishing their relationships and tracing those relationships to determine any affects on the requirement changes, and also creating reports and metrics about the requirement changes. On the other hand, some other participants prefer to use different kinds of tools for tracing the requirement changes. These include the Office tools (e.g., MS Word and Excel), versioning management tools (e.g., GIT), project management tools (e.g., Confluence, Trello, Easy Redmine, and Jira), modeling tools (e.g., Modelio, MagicDraw, ReqIF modeling tools), and integration tools (e.g., Jenkins).



Easy Redmine	https://www.easyredmine.com/
Trello	https://trello.com/
Xcode	https://developer.apple.com/xcode/
ReqIF Tools	https://www.omg.org/spec/ReqIF/1.2/About-ReqIF/
IBM Rational RequisitePro	https://www.ibm.com/support/pages/rational-requisitepro-714
Jenkins	https://www.jenkins.io/
Siemens Polarion	https://polarion.plm.automation.siemens.com/products/polarion-requirements
Modelio	https://www.modelio.org/
Cameo/MagicDraw	https://www.3ds.com/products-services/catia/products/no-magic/magicdraw/
Vitech	https://www.vitechcorp.com/requirements-management-and-traceability/
GIT	https://git-scm.com/
Confluence	https://www.atlassian.com/software/confluence
IBM DOORS	https://www.ibm.com/docs/en/ermd/9.7.0?topic=overview-doors
Jira Atlassian	https://jira.atlassian.com/
Office Tools	https://www.office.com/

Figure 18. The tools used by the participants for tracing the requirement changes.

Concerning the tool usage rates, GIT is used by relatively more participants, which is followed by the office tools, Jira, and DOORS. Confluence is also used by a few participants. The rest of the tools are each preferred by one or two participants at most.

4.7. Requirements Analysis

4.7.1. The Frequency of Analysing the Software Requirements

The participants show a high-level of interest to the requirements analysis. As shown in Figure 19, many participants (58%) frequently analyze their software requirements—26% chose "always" and 32% chose "much of the time".

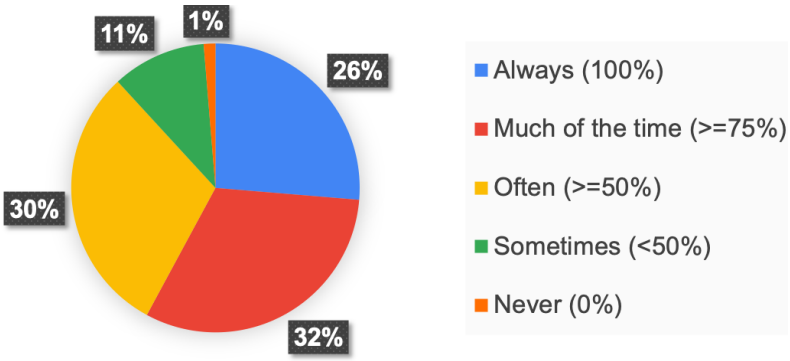


Figure 19. Participants' frequency of analyzing software requirements.

4.7.2. Analysis Techniques

As shown in Figure 20, among the participants analyzing requirements, most participants prefer manual inspection, which is concerned with analyzing software requirements without using any languages and tools. Overall, 61% of the participants only perform

manual inspection and 25% of the participants use manual inspection together with some languages and tools that promote the automated analysis of requirements. Only 14% of the participants perform automated analysis exclusively.

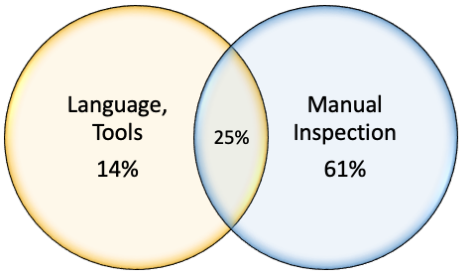


Figure 20. The techniques used by the participants for analyzing the software requirements.

4.7.3. Analysis Goals

In this question, we focused on Zowghi et al.'s approach for requirements validation [19], where three different properties for requirements validation are defined—consistency, completeness, and correctness. Consistency is concerned with if there exists any contradictions between any two requirements, e.g., name inconsistency between any two requirements that refer to the same service with different names. Completeness is concerned with any requirement specifications that are incomplete with regard to the language notation set or some design artifact (e.g., the software architecture design). Correctness is concerned with any requirement specifications that are incorrect with regard to some properties (e.g., the incorrect behavioral requirement specification with regard to the deadlock property).

As shown in Figure 21, a considerable number of the participants (35%) did not make any particular choices among the completeness, consistency, and correctness properties and chose all of them. That is, those participants are seemingly interested in analyzing requirements for all the three properties and detecting any relevant issues. Additionally, 34% of the participants are interested in analyzing requirements for just two of the properties together. Among those selecting a single property, the top-interest has been shown to be correctness.

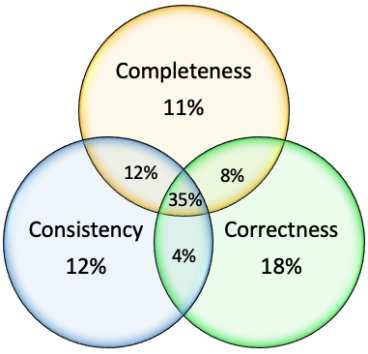


Figure 21. Participants' analysis goals for the software requirements.

4.7.4. The Languages and Tools Used for the Requirements Analysis

The greatest portion of the participants who use any languages and tools for the requirements analysis tend to develop their own analysis tools for specifying and analyzing requirements. Those participants seem to consider using programming technologies to develop their analysis tools. Only one of those participants considers using the meta-modeling technologies (i.e., Metaedit+ [51]), which support the easy and quick development of modeling editors for any domain-specific languages and their supporting toolset (e.g., analysis tools and code generators).

Two of the participants use tools such as Qvscribe [52] and Vitech [53], which provide advanced support for the requirements specifications and analysis. Another two participants use the UML/SysML modeling languages and their supplementary Object Constraint Language (OCL), and therefore possibly use some UML modeling tools to analyze the UML/SysML specifications with regard to the OCL constraint specifications. Additionally, another two participants use formal verification languages (e.g., Alloy [54], TLA+ [55], and VDM [56]), to formally specify the requirements and prove the formal specifications using the model-checking tools and theorem provers that support those formal languages. Lastly, another participant prefers to use the office tools (e.g., MS Excel) to analyze their requirements specifications.

4.7.5. Participants' Challenges on Analyzing Software Requirements

As shown in Figure 22, the participants' top challenge has to do with omitting the requirement specification errors due to the manual analysis performed (64%). The top-second challenge (30%) has to do with the inability of separating adequate amounts of time and budget for the requirements analysis.

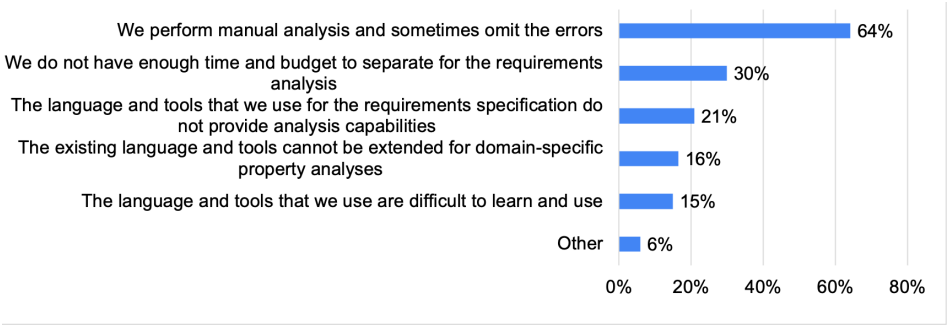


Figure 22. The challenges that the participants face with while analyzing software requirements.

A few participants indicated some other challenges that are different from the predefined challenge list discussed above (i.e., the free-text option). Two of the participants are concerned about the lack of skilled people for using the requirements analysis tools. Indeed, formal verification languages and tools that can be used to specify and analyze requirements require a steep learning curve. Additionally, one of the participants faced with another challenge on the inability of common, precise understanding of the requirements that prevents the effective analysis of requirements.

4.8. Requirements Transformation

4.8.1. The Frequency of Transforming the Software Requirements

Compared with the requirements change and analysis, the participants show a considerably lower level of interest in the requirements transformation. As shown in Figure 23, only 26% of the participants frequently transform the software requirements—7% chose "always" and 19% chose "much of the time". Additionally, 24% of the participants never transform their software requirements.

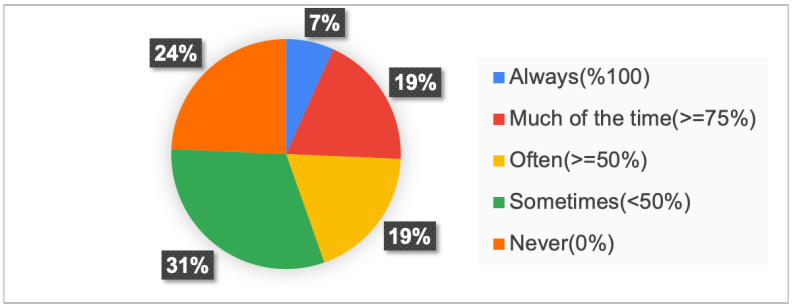


Figure 23. Participants' frequency of transforming software requirements.

4.8.2. The Types of Artifacts to be Produced

This question can be answered by all those participants who are interested in the requirements transformation—even those who have never performed requirements transformation before. As shown in Figure 24, the most considered artifact is the test scenarios/cases (57%), which indicates that most participants wish to transform requirements into test scenarios. The top-second artifact (46%) is the requirements documentations (e.g., Word, HTML, and RTF), which indicates that the participants could use the requirements gathered from customers (e.g., use-case scenarios) and produce formatted documentations in some standard formats. These two artifacts are followed by the early design decisions (44%), which has to do with transforming requirements into the high-level design decisions about systems to be developed (e.g., structural, behavioral and interaction decisions).

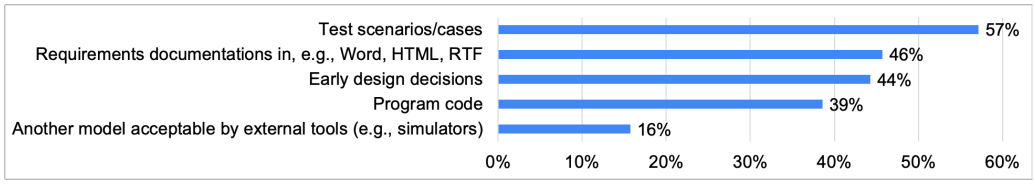


Figure 24. The types of artifacts that the participants wish to produce via the transformation of the software requirements.

4.8.3. Techniques Used for the Software Requirements Transformations

Among the participants who transform the software requirements, most of them (77%) use manual approaches for the requirements transformation. That is, those participants manually transform their requirements into, e.g., test scenarios, documentations, and high-level decisions and do not use any languages and tools that could enable specifying requirements and transforming them into a bunch of artifacts (e.g., scenarios) using a user-defined transformation algorithm automatically and correctly.

4.8.4. The Languages and Tools Used for the Software Requirements Transformations

The small number of participants who use some languages and tools for the requirements transformation process followed different approaches. These include *(i)* developing plugin tools for the existing modeling tools (e.g., MagicDraw) for transforming the requirements specifications, *(ii)* using the code generation frameworks of the meta-modeling technologies such as Metaedit+ [51], and *(iii)* developing in-house compiler applications that can process the requirements specifications for different purposes.

4.8.5. Participants' Challenges on Transforming Software Requirements

As shown in Figure 25, the greatest challenges occur due to *(i)* the considerable amount of time and budget that need to be separated for the manual transformation of requirements (54%) and *(ii)* the difficulties in ensuring the correctness of the transformation process (41%). Note that, regardless of the manual and automated transformation of software requirements, it is really difficult to check if the transformation works correctly and produce the intended output.

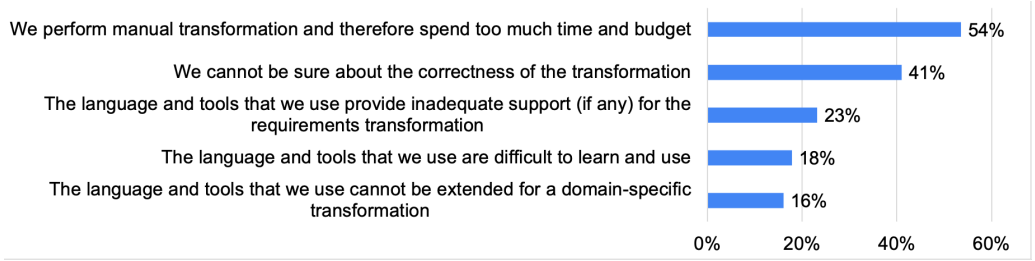


Figure 25. The challenges that the participants face while transforming the software requirements.

5. Discussion

5.1. Summary of Findings

Below are the summary of the findings for the survey results that are discussed with regard to the research questions of the survey study.

RQ1: Practitioners motivations for specifying and changing software requirements.

Practitioners' top concern is to specify the requirements precisely so that the collaborating stakeholders (e.g., customers, system engineers, software engineers, and testers) understand each other in the same way and analyze those requirements to detect any incompleteness, inconsistencies, and incorrectness issues. So, obtaining the right requirements the first time is essentially the main priority of the practitioners contributing to the survey.

Almost all the practitioners (98%) specify requirements for the functional aspects of systems. Indeed, the use-cases and scenarios are the top-concerns that are addressed in the requirements specification process (65–70%), which have to do with the functional requirements. It should also be noted that 64% of the practitioners specify the quality aspects of systems, which is quite surprising and indicates that practitioners have started to realize the importance of the quality requirements for developing the system functionalities in a quality way. Indeed, this is quite opposite to the past survey results that indicate practitioners' reluctance towards the quality requirements [57–59].

Another interesting outcome has to do with the practitioners (64%) changing requirements due to the requirement priorities. This can be attributed to the practitioners who prefer the agile software process model (77%), which actually promotes the software systems to be developed in multiple, iterative releases that differ from each other on the requirements implemented depending on their priorities [60]. The requirement priorities can change dynamically due to the time and budget constraints and cause the existing requirements to be changed, left for a later release, or even totally omitted.

RQ2: The techniques and technologies used by practitioners in performing different activities of requirements engineering.

The top techniques for requirements gathering are the use-case scenarios (60%) and surveys and interviews (54%), where customers are the source in both techniques. On the other hand, it is not so common for the practitioners to be provided with an existing list of requirements by the customers (44%) or asked to develop novel solutions by searching the literature and market (41%).

Concerning the requirements specification, 35% of the practitioners use solely natural languages for the requirements specifications, while just 7% of the practitioners solely use modeling languages. Most of the practitioners who use modeling languages indicated that they use modeling languages together with natural languages and PowerPoint tools.

Concerning the requirement changes, while many powerful requirements management and traceability tools are available (e.g., IBM tools, Siemens Polarion, Vitech, MagicDraw, Modelio, etc.), more than half of the practitioners (58%) rarely use any traceability tools to manage the requirement changes.

Concerning the requirements analysis, 58% of the practitioners frequently analyze their software requirements and many of them are highly interested in analyzing the requirements for all properties considered (i.e., completeness, consistency, and correctness). However, the tool usage for the automated requirements analysis is rare. This may be attributed to the fact that many analysis tools essentially require some specification languages to be learned and used [61,62]. Indeed, model checkers that can prove the correctness of the requirement specifications need to use process algebras, which are known too have a steep learning curve [63].

The requirements transformation is rarely performed by the practitioners—just 26% frequently transform requirements. The top artifact that wishes to be produced from the requirements is the test scenarios (57%).

RQ3: The challenges that practitioners face with in different activities of requirement engineering.

Many practitioners are well aware that performing requirements analysis and transformation manually causes some issues, despite the lack of tool usages for the necessary automation. Indeed, practitioners agree that the requirement specification errors are omitted with the manual inspection techniques and too much time and budget need to be spent when requirements are analyzed and transformed into some other artifacts (e.g., test scenarios) manually. While the requirements analysis and transformation could be performed more effectively and productively via the automation support of the different technologies available, we believe that practitioners are either not aware of many such tools and technologies or never use them. Indeed, tool-related challenges for the requirements analysis and transformation that are available in the answer lists of the respective survey questions have been rarely selected by the practitioners.

RQ4: The customer involvement in the requirement engineering.

Nearly all the practitioners (98%) involve customers in their requirements engineering activities to some extent. Most of those practitioners (82%) involve customers in the requirements gathering activity and more than half of those (56%) involve customers in the requirements analysis. Note also that 22% of the practitioners involve customers in the requirements transformation. Indeed, transforming requirements into test scenarios or high-level design decisions can be performed together with customers more effectively. Customers could even help the development team to transform the requirements into code (or low-level design decisions) [64]. However, the results show that practitioners' top challenges indicated in the survey have to do with the customers who have limited or no technical knowledge, which causes the customers to be reluctant to participate in the requirements engineering activities or fail to communicate precisely even if they contribute.

5.2. Lessons Learned

We have previously conducted several industrial surveys on different aspects of software engineering, including the surveys on practitioners' knowledge and experiences on software architecture [40], modeling experiences [65], software architecture modeling languages [41], meta-modeling technologies [42], and practitioners' model-driven engineering experiences in the embedded systems domain [43]. Those survey studies let us gain experience on designing survey questions effectively, finding participants, executing surveys, and analyzing the responses carefully. The survey discussed in this paper is our first attempt at conducting an industrial survey on the requirements engineering. Therefore, we learned many technical lessons regarding practitioners' knowledge and experiences towards the requirements engineering, which are discussed in the rest of this sub-section.

We learned that most of the practitioners show a considerable level of interest in each activity of requirements engineering (i.e., gathering, specification, analysis, change, and transformation). However, many practitioners still use the traditional approaches (i.e., informal natural languages) applied in the nineties for performing requirements specification, despite the fact that the natural language specifications are very difficult (if not impossible) to analyze, transform, and trace via some tool support. We believe that this outcome needs to be used by both the tool vendors and academia. Academia can perform some empirical research in the near future to understand what really makes practitioners stay away from the modeling languages and tools and any supporting technologies (e.g., meta-modeling technologies). Tool vendors can also improve their tools with regard to the results of those empirical studies. Any tool that supports the natural language-based specifications of requirements in a precise way for enhanced communications and their analysis and transformation could also be attractive for practitioners.

While almost all the practitioners wish the customers to be involved in the requirements engineering activities, practitioners' top challenge is the customers' lack of technical knowledge and therefore reluctance to become involved. This could actually be considered as one of the reasons that practitioners still use natural languages. Indeed, if any modeling languages were used, customers would probably have difficulties in understanding the specifications due to the language and tools' learning curves.

Many practitioners who use tools for the requirements analysis prefer to develop their own analysis tools for specifying requirements and checking them against such properties as the requirements completeness, consistency, and correctness. Actually, many meta-modeling technologies have existed for 10 (or more) years, including Metaedit+ [51], Eclipse Xtext [66], Eclipse Sirius [67], and MPS [68]. Meta-modeling technologies make it very easy and quick to develop modeling editors with analysis support for any domain-specific modeling languages [69,70]. However, we learned that none of the practitioners (except one participant using Metaedit+) participating in the survey use any meta-modeling technologies. The practitioners instead use programming technologies to develop requirements analysis tools, which, however, could take much more time and budget, and the resulting tools may not be so easy to extend.

The survey results indicate that practitioners essentially show a high-level of interest on the quality requirements. However, the results also show that practitioners tend to specify the use-case and scenario descriptions of systems, which essentially have to do with the user–system interactions and thus the functional requirements. Specifying high-level decisions about the system behaviors and interactions where the quality requirements can be addressed is not considered by many practitioners. To minimize any knowledge gap here, academia can consider opening more university courses at undergraduate and graduate levels such as software architectures, formal specification and verification, software modeling, model-based systems engineering for teaching the essence of modeling, meta-modeling, and their aid in specifying and analyzing early decisions about system requirements.

As discussed before, practitioners have been observed to rarely use any modeling languages and toolsets for their requirements engineering activities. One should, however, note that the existing modeling languages and toolsets suffer from a number of problems that we believe could make the practitioners reluctant here. While UML is considered as the most used general-purpose modeling language in the industry, UML is essentially an ambiguous language that consists of many confusing diagrams and thus leads to the requirement models that are open to different interpretations [71,72]. Note that as indicated in the survey results, practitioners' main motivations with the requirement specifications have to do with the precise communications among stakeholders and analyzing requirements to detect any issues. Moreover, while hundreds of tools support UML for the requirements modeling and analysis, the existing tools do not support reasoning about the requirement specifications adequately [73]. Indeed, one cannot check the UML models for any quality properties or define their own functional constraints for checking models easily. So, academia can consider teaching potential practitioners how to specify their own domain-specific modeling languages and toolset that could better suit their needs rather than teaching the general-purpose UML language solely. Additionally, formal specification languages have been existing for the formal specification and verification of system requirements and proving the correctness of the specifications for the quality properties. Some of the well-known formal specification languages include ProMeLa [74], Alloy [54], AADL [75], LOTOS [76], and TLA+ [55], and all these languages are supported with tools that support checking the specifications for the quality properties and proving their correctness as well. However, formal specification languages are rarely used in industry due to requiring steep learning curves with their algebraic notation sets [63]. Therefore, tool vendors for the formal specification languages could introduce an additional level of abstractions that can lower the learning curve with a higher level of notation sets and let practitioners use the formal verification tools easily.

Lastly, we learned another lesson about some missing profile questions. To reduce the time needed to fill in the survey and attract more participants, we did not include the questions for learning the participants' organization size, the types of software systems developed and the level of complexity for the projects. However, lacking that information prevented us from observing some interesting correlations such as the correlations between *(i)* the organization size and the motivation for specifying requirements, *(ii)* tool usages and the project complexity, and *(iii)* software types and formal modeling language usages.

5.3. Threats to Validity

In this section, we will discuss the potential threats to the validity of our survey results and how we intended to mitigate those threats.

5.3.1. Internal Validity

The threats to the internal validity of the survey results essentially have to do with the cause–effect relationships, which indicate the possibility of some unknown variables that affect the outcome and therefore lead to biased results [77–79]. The unknown variables may occur due to the selection of the participants for the survey, which was the non-random selection via convenience sampling in our case. Therefore, to minimize any potential threats herein, we tried to simulate the random sampling. That is, as discussed in Section 3.4, we did our best to spread our survey in as many platforms as possible (e.g., mailing lists, Linkedin groups) where any participants using those platforms may encounter our survey posts. We also sent three reminders in three months' time to attract any potential participants who may have omitted the previous posts. We further asked in each post to share/forward the post to any whom the participants know in their domain that may be interested in participating (i.e., the snowballing technique). So, we managed to attract 84 acceptable responses from the participants with diverse profiles, including different industries, job positions, and educations. Note here that while the participants enrolled in those platforms may access the survey link randomly, we attracted the interested participants only and that is actually considered as a nonrandom sampling. Lastly, we tried to minimize any threats due to the instrumentation biases by performing a pilot study as discussed in Section 3.2.

5.3.2. External Validity

The threats to the external validity of the survey results have to do with generalizing the results to the entire population who are involved in the requirements engineering activities [77–79]. While we are not able to reach every member of the population, we tried to minimize any external threats by sharing the survey on various platforms. So, we managed to reach practitioners with diverse profiles. Indeed, we received 84 acceptable responses from 16 work industries, 19 job positions, and 13 bachelor degrees.

5.3.3. Construct Validity

The threats to the construct validity of the survey results have to do with how well the research questions are investigated via the survey questions and practitioners' answers to those questions [77,79]. To minimize any threats here, we divided the survey questions into separate sections where each section targets particular research question(s) introduced in Section 3.1. Additionally, the answer lists for the survey questions have been defined carefully using well-cited books or articles in the corresponding areas and revised through a comprehensive pilot study involving academics who are highly experienced in the field and industry practitioners. To avoid any misunderstandings and thus unintended answers, we clearly explained the purpose of the survey to the participants before the participants are allowed to answer the survey questions and indicated that we do not collect any personal data—all collected data are kept anonymous. Lastly, to minimize any mono-operation bias, we accepted responses from different profiles with diverse industries, job positions, and educations.

6. Conclusions

Thanks to the survey study discussed in the paper, practitioners' perspectives towards the requirements engineering have been re-considered in a more comprehensive way. The survey study provides unique results with its consideration of multiple important aspects together including practitioners' motivations, *(i)* practitioners' experiences on the requirements gathering, specification, analysis, transformation, and evolution, *(ii)* the customer involvement, and *(iii)* the techniques and technologies used.

The survey results shed light on many crucial issues. Practitioners' top motivations for specifying requirements are *(i)* the precise communications among different stakeholders to obtain the right requirements and *(ii)* the analysis of the requirement specification to obtain the right requirements. Most practitioners specify the functional requirements to describe the use-cases and test scenarios of their systems to be developed. The quality requirements are important for many practitioners as well. However, most of those practitioners do not specify high-level decisions about system behaviors, interactions, and deployment that enable reasoning about the quality requirements. Most practitioners still use the traditional approaches for performing the requirements engineering activities. That is, practitioners use natural languages in any form to specify requirements and neglect the state of the art techniques and technologies (e.g., meta-modeling technologies and advanced requirements management tools, and formal specification and verification techniques). While practitioners give high importance to the requirements analysis, most of them manually analyze their requirements without using specialized modeling languages and their analysis tools. Transforming requirements to produce any useful artifacts is not as popular as the requirements analysis and the main focus is on using the requirements to produce test scenarios rather than program code or high-level design decisions. Again, most of those practitioners perform manual transformations. Concerning the requirements change, most practitioners change their requirements because of the customer demands and the requirement priorities that could change constantly throughout the development life-cycle. Practitioners also face some challenges while performing the requirements engineering activities, including *(i)* the time and effort that need to be spent in performing the requirements analysis and transformation activities manually, *(ii)* the issue of omitting many requirement specification errors due to the manual analysis, *(iii)* ensuring the correctness of the transformation. Lastly, most practitioners involve customers in their software development to some extent. The top activities in which customers are involved are the requirements gathering and requirements analysis. However, practitioners are challenged due to customers' limited technical knowledge, which causes the customers to be reluctant to participate in the activities or fail to communicate precisely.

In the near future, we are planning to validate the survey results in the DFDS company that produces software solutions in the shipping and logistics industry [45]. We will organize interview sessions with a large group of practitioners in DFDS and ask the survey questions. We will also conduct similar interviews with our partners from the automotive industry with whom we work together in the AITOC project [80] that is labeled by the European Union's EUREKA Cluster program ITEA (Information Technology for European Advancement).

We strongly believe that the survey results will be useful for both academia, industry, and tool vendors. Academia can use the survey results to shed light on their research on requirements engineering. Industry can use the survey results to learn the general perspectives of the practitioners on requirements engineering in terms of different activities and customer involvement. Lastly, tool vendors can improve their tools on the requirements management so as to better satisfy the needs of practitioners.

Author Contributions: Conceptualization, M.O. and D.A.; methodology, M.O. and D.A.; formal analysis, M.O. and D.A.; investigation, M.O, E.C.T. and B.K.; resources, M.O, E.C.T. and B.K.; data curation, M.O., D.A., E.C.T. and B.K.; writing—original draft preparation, M.O., D.A., E.C.T., B.K. and G.K.; writing—review and editing, M.O., D.A. and G.K.; visualization, M.O.; supervision, M.O. and G.K.; project administration, M.O.; funding acquisition, E.C.T. and B.K. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was funded by DFDS.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The survey raw data (including the survey questions and the data collected from the participants) are available on https://doi.org/10.5281/zenodo.6754676 (accessed on 15 November 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fitzgerald, B. Software Crisis 2.0. *Computer* **2012**, *45*, 89–91. [CrossRef]
2. Chaos Report 2015. Available online: https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf (accessed on 15 October 2022).
3. Dowson, M. The Ariane 5 software failure. *ACM SIGSOFT Softw. Eng. Notes* **1997**, *22*, 84. [CrossRef]
4. Jackson, M. Seeing More of the World. *IEEE Softw.* **2004**, *21*, 83–85. [CrossRef]
5. Leveson, N.G. The Therac-25: 30 Years Later. *Computer* **2017**, *50*, 8–11. [CrossRef]
6. Davidson, C. A dark Knight for algos. *Risk* **2012**, *25*, 32–34.
7. British Airways System Problem. Available online: https://www.theguardian.com/world/2017/may/27/british-airways-system-problem-delays-heathrow (accessed on 12 January 2023).
8. Google Plus Data Leak. Available online: https://www.usatoday.com/story/tech/2018/12/11/google-plus-leak-social-network-shut-down-sooner-after-security-bug/2274296002/ (accessed on 12 January 2023).
9. Amazon Software Problem. Available online: https://fortune.com/2021/12/10/amazon-software-problem-cloud-outage-cause/ (accessed on 12 January 2023).
10. Hsia, P.; Davis, A.M.; Kung, D.C. Status Report: Requirements Engineering. *IEEE Softw.* **1993**, *10*, 75–79. [CrossRef]
11. Cheng, B.H.C.; Atlee, J.M. Research Directions in Requirements Engineering. In Proceedings of the International Conference on Software Engineering, ISCE 2007, Workshop on the Future of Software Engineering, FOSE 2007, Minneapolis, MN, USA, 23–25 May 2007; Briand, L.C., Wolf, A.L., Eds.; IEEE Computer Society: Washington, DC, USA, 2007; pp. 285–303. [CrossRef]
12. Nuseibeh, B.; Easterbrook, S.M. Requirements engineering: A roadmap. In Proceedings of the 22nd International Conference on on Software Engineering, Future of Software Engineering Track, ICSE 2000, Limerick, Ireland, 4–11 June 2000; Finkelstein, A., Ed.; ACM: Boston, MA, USA, 2000; pp. 35–46. [CrossRef]
13. Rodrigues da Silva, A.; Olsina, L. Special Issue on Requirements Engineering, Practice and Research. *Appl. Sci.* **2022**, *12*. [CrossRef]
14. Lethbridge, T.C.; Lagamiere, R. *Object-Oriented Software Engineering–Practical Software Development Using UML and Java*; MacGraw-Hill: New York, NY, USA, 2001.
15. Rumbaugh, J.; Jacobson, I.; Booch, G. *Unified Modeling Language Reference Manual*, 2nd ed.; Pearson Higher Education: New York, NY, USA, 2004.
16. Balmelli, L. An Overview of the Systems Modeling Language for Products and Systems Development. *J. Obj. Tech.* **2007**, *6*, 149–177. [CrossRef]
17. Feiler, P.H.; Gluch, D.P.; Hudak, J.J. *The Architecture Analysis & Design Language (AADL): An Introduction*; Technical Report; Software Engineering Institute: Pittsburgh, PA, USA, 2006.
18. Boehm, B.W. Verifying and Validating Software Requirements and Design Specifications. *IEEE Softw.* **1984**, *1*, 75–88. [CrossRef]
19. Zowghi, D.; Gervasi, V. On the interplay between consistency, completeness, and correctness in requirements evolution. *Inf. Softw. Technol.* **2003**, *45*, 993–1009. [CrossRef]
20. Kos, T.; Mernik, M.; Kosar, T. A Tool Support for Model-Driven Development: An Industrial Case Study from a Measurement Domain. *Appl. Sci.* **2019**, *9*, 4553. [CrossRef]
21. Lam, W.; Shankararaman, V. Requirements Change: A Dissection of Management Issues. In Proceedings of the 25th EUROMICRO '99 Conference, Informatics: Theory and Practice for the New Millenium, Milan, Italy, 8–10 September 1999; IEEE Computer Society: Washington, DC, USA, 1999; pp. 2244–2251. [CrossRef]
22. Jayatilleke, S.; Lai, R. A systematic review of requirements change management. *Inf. Softw. Technol.* **2018**, *93*, 163–185. [CrossRef]
23. Kabbedijk, J.; Brinkkemper, S.; Jansen, S.; van der Veldt, B. Customer Involvement in Requirements Management: Lessons from Mass Market Software Development. In Proceedings of the RE 2009, 17th IEEE International Requirements Engineering Conference, Atlanta, GA, USA, 31 August–4 September 2009; IEEE Computer Society: Washington, DC, USA, 2009; pp. 281–286. [CrossRef]
24. Saiedian, H.; Dale, R. Requirements engineering: Making the connection between the software developer and customer. *Inf. Softw. Technol.* **2000**, *42*, 419–428. [CrossRef]
25. Franch, X.; Fernández, D.M.; Oriol, M.; Vogelsang, A.; Heldal, R.; Knauss, E.; Travassos, G.H.; Carver, J.C.; Dieste, O.; Zimmermann, T. How do Practitioners Perceive the Relevance of Requirements Engineering Research? An Ongoing Study. In Proceedings of the 25th IEEE International Requirements Engineering Conference, RE 2017, Lisbon, Portugal, 4–8 September 2017; Moreira, A., Araújo, J., Hayes, J., Paech, B., Eds.; IEEE Computer Society: Washington, DC, USA, 2017; pp. 382–387. [CrossRef]

26. Fernández, D.M.; Wagner, S.; Kalinowski, M.; Felderer, M.; Mafra, P.; Vetrò, A.; Conte, T.; Christiansson, M.; Greer, D.; Lassenius, C.; et al. Naming the pain in requirements engineering–Contemporary problems, causes, and effects in practice. *Empir. Softw. Eng.* **2017**, *22*, 2298–2338. [CrossRef]
27. Verner, J.M.; Cox, K.; Bleistein, S.J.; Cerpa, N. Requirements Engineering and Software Project Success: An industrial survey in Australia and the U.S. *Australas. J. Inf. Syst.* **2005**, *13, pp. 225–238*. [CrossRef]
28. Solemon, B.; Sahibuddin, S.; Ghani, A.A.A. Requirements Engineering Problems and Practices in Software Companies: An Industrial Survey. In Proceedings of the Advances in Software Engineering– International Conference on Advanced Software Engineering and Its Applications, ASEA 2009 Held as Part of the Future Generation Information Technology Conference, FGIT 2009, Jeju Island, Korea, 10–12 December 2009; Proceedings; Communications in Computer and Information Science; Slezak, D., Kim, T., Akingbehin, K., Jiang, T., Verner, J.M., Abrahão, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 59, pp. 70–77. [CrossRef]
29. Kassab, M. The changing landscape of requirements engineering practices over the past decade. In Proceedings of the 2015 IEEE Fifth International Workshop on Empirical Requirements Engineering, EmpiRE 2015, Ottawa, ON, Canada, 24 August 2015; Berntsson-Svensson, R., Daneva, M., Ernst, N.A., Marczak, S., Madhavji, N.H., Eds.; IEEE Computer Society: Washington, DC, USA, 2015; pp. 1–8. [CrossRef]
30. Juzgado, N.J.; Moreno, A.M.; Silva, A. Is the European Industry Moving toward Solving Requirements Engineering Problems? *IEEE Softw.* **2002**, *19*, 70–77. [CrossRef]
31. Neill, C.J.; Laplante, P.A. Requirements Engineering: The State of the Practice. *IEEE Softw.* **2003**, *20*, 40–45. [CrossRef]
32. Carrillo-de-Gea, J.M.; Nicolás, J.; Alemán, J.L.F.; Toval, A.; Ebert, C.; Vizcaíno, A. Requirements engineering tools: Capabilities, survey and assessment. *Inf. Softw. Technol.* **2012**, *54*, 1142–1157. [CrossRef]
33. Liu, L.; Li, T.; Peng, F. Why Requirements Engineering Fails: A Survey Report from China. In Proceedings of the RE 2010, 18th IEEE International Requirements Engineering Conference, Sydney, New South Wales, Australia, 27 September–1 October 2010; IEEE Computer Society: Washington, DC, USA, 2010; pp. 317–322. [CrossRef]
34. Nikula, U.; Sajaniemi, J.; Kälviäinen, H. *A State-of-the-Practice Survey on Requirements Engineering in Small-and Medium-Sized Enterprises*; Technical report; Telecom Business Research Center Lappeenranta: Lappeenranta, Finland, 2000.
35. Memon, R.N.; Ahmad, R.; Salim, S.S. Problems in Requirements Engineering Education: A Survey. In Proceedings of the 8th International Conference on Frontiers of Information Technology, Islamabad, Pakistan, 21–23 December, 2010; Association for Computing Machinery: New York, NY, USA, 2010; FIT '10. [CrossRef]
36. Jarzebowicz, A.; Sitko, N. Communication and Documentation Practices in Agile Requirements Engineering: A Survey in Polish Software Industry. In Proceedings of the Information Systems: Research, Development, Applications, Education–12th SIGSAND/PLAIS EuroSymposium 2019, Gdansk, Poland, 19 September 2019; Proceedings; Lecture Notes in Business Information Processing; Wrycza, S., Maslankowski, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2019, Volume 359, pp. 147–158. [CrossRef]
37. Ågren, S.M.; Knauss, E.; Heldal, R.; Pelliccione, P.; Malmqvist, G.; Bodén, J. The impact of requirements on systems development speed: A multiple-case study in automotive. *Requir. Eng.* **2019**, *24*, 315–340. [CrossRef]
38. Palomares, C.; Franch, X.; Quer, C.; Chatzipetrou, P.; López, L.; Gorschek, T. The state-of-practice in requirements elicitation: An extended interview study at 12 companies. *Requir. Eng.* **2021**, *26*, 273–299. [CrossRef]
39. Groves, R.M.; Fowler, F.J., Jr.; Couper, M.P.; Lepkowski, J.M.; Singer, E.; Tourangeau, R. *Survey Methodology*, 2nd ed.; John Wiley & Sons: Hoboken, NJ, USA, 2009.
40. Ozkaya, M. What is Software Architecture to Practitioners: A Survey. In Proceedings of the MODELSWARD 2016–Proceedings of the 4rd International Conference on Model-Driven Engineering and Software Development, Rome, Italy, 19–21 February 2016; Hammoudi, S., Pires, L.F., Selic, B., Desfray, P., Eds.; SciTePress: Setubal, Portugal, 2016; pp. 677–686. [CrossRef]
41. Ozkaya, M. Do the informal & formal software modeling notations satisfy practitioners for software architecture modeling? *Inf. Softw. Technol.* **2018**, *95*, 15–33. [CrossRef]
42. Ozkaya, M.; Akdur, D. What do practitioners expect from the meta-modeling tools? A survey. *J. Comput. Lang.* **2021**, *63*, 101030. [CrossRef]
43. Akdur, D.; Garousi, V.; Demirörs, O. A survey on modeling and model-driven engineering practices in the embedded software industry. *J. Syst. Archit.* **2018**, *91*, 62–82. [CrossRef]
44. Molléri, J.S.; Petersen, K.; Mendes, E. Survey Guidelines in Software Engineering: An Annotated Review. In Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM 2016, Ciudad Real, Spain, 8–9 September 2016; ACM: Boston, MA, USA, 2016; pp. 58:1–58:6. [CrossRef]
45. DFDS. Available online: https://www.dfds.com.tr (accessed on 1 October 2022).
46. Rozanski, N.; Woods, E. *Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives*, 2nd ed.; Addison-Wesley Professional: Westford, Massachusetts, USA, 2011.
47. Taylor, R.N.; Medvidovic, N.; Dashofy, E.M. *Software Architecture–Foundations, Theory, and Practice*; Wiley: Hoboken, NJ, USA, 2010.
48. Popping, R. Analyzing Open-ended Questions by Means of Text Analysis Procedures. *Bull. Sociol. Methodol. De Méthodol. Sociol.* **2015**, *128*, 23–39. [CrossRef]
49. Fricker, R.D. Sampling methods for web and e-mail surveys. In *The SAGE Handbook of Online Research Methods*; SAGE: Virginia, USA, 2008; pp. 195–216.

50. Raw Data of Our Survey on the Practitioners' Perspectives towards Requirements Engineering. Available online: https://doi.org/10.5281/zenodo.6754676 (accessed on 15 October 2022).
51. Kelly, S.; Lyytinen, K.; Rossi, M. MetaEdit+ A Fully Configurable Multi-User and Multi-Tool CASE and CAME Environment. In *Seminal Contributions to Information Systems Engineering, 25 Years of CAiSE*; Bubenko, J., Krogstie, J., Pastor, O., Pernici, B., Rolland, C., Sølvberg, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 109–129. [CrossRef]
52. QVscribe. Available online: https://qracorp.com (accessed on 15 October 2022).
53. Vitech. Available online: https://www.vitechcorp.com (accessed on 15 October 2022).
54. Jackson, D. Alloy: A lightweight object modelling notation. *ACM Trans. Softw. Eng. Methodol.* **2002**, *11*, 256–290. [CrossRef]
55. Lamport, L.; Matthews, J.; Tuttle, M.R.; Yu, Y. Specifying and verifying systems with TLA+. In Proceedings of the 10th ACM SIGOPS European Workshop, Saint-Emilion, France, 1 July 2002; Muller, G., Jul, E., Eds.; ACM: Boston, MA, USA, 2002; pp. 45–48. [CrossRef]
56. Plat, N.; Larsen, P.G. An overview of the ISO/VDM-SL standard. *ACM SIGPLAN Not.* **1992**, *27*, 76–82. [CrossRef]
57. Mairiza, D.; Zowghi, D.; Nurmuliani, N. An investigation into the notion of non-functional requirements. In Proceedings of the 2010 ACM Symposium on Applied Computing (SAC), Sierre, Switzerland, 22–26 March 2010; Shin, S.Y., Ossowski, S., Schumacher, M., Palakal, M.J., Hung, C., Eds.; ACM: Boston, MA, USA, 2010; pp. 311–317. [CrossRef]
58. Kopczynska, S.; Ochodek, M.; Nawrocki, J.R. On Importance of Non-functional Requirements in Agile Software Projects–A Survey. In *Integrating Research and Practice in Software Engineering*; Studies in Computational Intelligence; Jarzabek, S., Poniszewska-Maranda, A., Madeyski, L., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; Volume 851, pp. 145–158. [CrossRef]
59. Bajpai, V.; Gorthi, R.P. On non-functional requirements: A survey. In Proceedings of the 2012 IEEE Students' Conference on Electrical, Electronics and Computer Science, Bhopal, India, 1–2 March 2012; pp. 1–4. [CrossRef]
60. AL-Ta'ani, R.H.; Razali, R. Prioritizing Requirements in Agile Development: A Conceptual Framework. *Procedia Technol.* **2013**, *11*, 733–739. [CrossRef]
61. Erata, F.; Challenger, M.; Tekinerdogan, B.; Monceaux, A.; Tüzün, E.; Kardas, G. Tarski: A Platform for Automated Analysis of Dynamically Configurable Traceability Semantics. In Proceedings of the Symposium on Applied Computing, Marrakech, Morocco, 3–7 April 2017; Association for Computing Machinery: New York, NY, USA, 2017; SAC '17, pp. 1607–1614. [CrossRef]
62. Erata, F.; Goknil, A.; Tekinerdogan, B.; Kardas, G. A Tool for Automated Reasoning about Traces Based on Configurable Formal Semantics. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, Paderborn, Germany, 4–8 September 2017; Association for Computing Machinery: New York, NY, USA, 2017; ESEC/FSE 2017, pp. 959–963. [CrossRef]
63. Malavolta, I.; Lago, P.; Muccini, H.; Pelliccione, P.; Tang, A. What Industry Needs from Architectural Languages: A Survey. *IEEE Trans. Softw. Eng.* **2013**, *39*, pp. 869–891. [CrossRef]
64. Ramesh, B.; Cao, L.; Baskerville, R.L. Agile requirements engineering practices and challenges: An empirical study. *Inf. Syst. J.* **2010**, *20*, 449–480. [CrossRef]
65. Akdur, D. Modeling knowledge and practices in the software industry: An exploratory study of Turkey-educated practitioners. *J. Comput. Lang.* **2021**, *66*, 101063. [CrossRef]
66. Eysholdt, M.; Behrens, H. Xtext: Implement your language faster than the quick and dirty way. In Proceedings of the SPLASH/OOPSLA Companion, Reno, NV, USA, 17–21 October 2010; Cook, W.R., Clarke, S., Rinard, M.C., Eds.; ACM: Boston, MA, USA, 2010; pp. 307–309.
67. Viyović, V.; Maksimović, M.; Perisić, B. Sirius: A rapid development of DSM graphical editor. In Proceedings of the IEEE 18th International Conference on Intelligent Engineering Systems INES 2014, Tihany, Hungary, 3–5 July 2014; pp. 233–238. [CrossRef]
68. Pech, V.; Shatalin, A.; Voelter, M. JetBrains MPS as a Tool for Extending Java. In Proceedings of the 2013 International Conference on Principles and Practices of Programming on the Java Platform: Virtual Machines, Languages, and Tools, Stuttgart, Germany, 11–13 September 2013; Association for Computing Machinery: New York, NY, USA, 2013; PPPJ '13, pp. 165–168. [CrossRef]
69. Kärnä, J.; Tolvanen, J.P.; Kelly, S. Evaluating the use of domain-specific modeling in practice. In Proceedings of the 9th OOPSLA workshop on Domain-Specific Modeling, Orlando, FL, USA, 25–29 October 2009.
70. Nokia Case Study–MetaEdit+ Revolutionized the Way Nokia Develops Mobile Phone Software. Available online: https://www.metacase.com/papers/MetaEdit_{}in_{}Nokia.pdf (accessed on 14 July 2022).
71. Siau, K.; Loo, P. Identifying Difficulties in Learning Uml. *Inf. Syst. Manag.* **2006**, *23*, 43–51. [CrossRef]
72. Simons, A.J.H.; Graham, I. 30 Things that Go Wrong in Object Modelling with UML 1.3. In *Behavioral Specifications of Businesses and Systems*; The Kluwer International Series in Engineering and Computer Science; Kilov, H., Rumpe, B., Simmonds, I., Eds.; Springer: Berlin/Heidelberg, Germany, 1999; Volume 523, pp. 237–257. [CrossRef]
73. Ozkaya, M. Are the UML modelling tools powerful enough for practitioners? A literature review. *IET Softw.* **2019**, *13*, 338–354. [CrossRef]
74. Holzmann, G.J. *The SPIN Model Checker–Primer and Reference Manual*; Addison-Wesley: Boston, MA, USA, 2004; pp. I–XII, 1–596.
75. Feiler, P.H.; Lewis, B.A.; Vestal, S. The SAE Architecture Analysis & Design Language (AADL): A Standard for Engineering Performance Critical Systems. In Proceedings of the IEEE Intl Symp. on Intell. Control, 2006 Toulouse, France, 25–29 April 2006; pp. 1206–1211. [CrossRef]
76. Bolognesi, T.; Brinksma, E. Introduction to the ISO Specification Language LOTOS. *Comput. Netw. ISDN Syst.* **1987**, *14*, 25–59. [CrossRef]

77. Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M.C.; Regnell, B. *Experimentation in Software Engineering*; Springer: Berlin/Heidelberg, Germany, 2012. [CrossRef]
78. Siegmund, J.; Siegmund, N.; Apel, S. Views on Internal and External Validity in Empirical Software Engineering. In Proceedings of the 37th IEEE/ACM International Conference on Software Engineering, ICSE 2015, Florence, Italy, 16–24 May 2015; Bertolino, A., Canfora, G., Elbaum, S.G., Eds.; IEEE Computer Society: Washington, DC, USA, 2015; Volume 1, pp. 9–19. [CrossRef]
79. Shull, F.; Singer, J.; Sjøberg, D.I.K. (Eds.) *Guide to Advanced Empirical Software Engineering*; Springer: Berlin/Heidelberg, Germany, 2008. [CrossRef]
80. AITOC ITEA Project. Available online: https://itea4.org/project/aitoc.html (accessed on 15 October 2022).