University of Southampton

Faculty of Engineering and Physical Sciences

Electronics and Computer Science

Cyber-Physical Systems Group

Usage and Perceptions of Spikes in Agile Software Development: An Exploratory Study in an Industrial Context

By

Hussein Ali Al Hashimi

ORCID ID: 0000-0003-2148-8095

Thesis Submitted in Fulfilment of the Requirements for the Degree of Doctor of Philosophy in Computer Science

April 2023

This thesis is completely dedicated

To my parents for their endless love, support, and encouragement

To my wife Afaf and my daughter Rival

To my siblings

Without their constant support, this work would not have been possible

UNIVERSITY OF SOUTHAMPTON

Abstract

Faculty of Engineering and Physical Sciences

Electronics and Computer Science

Thesis for the degree of Doctor of Philosophy

Usage and Perceptions of Spikes in Agile Software Development: An Exploratory Study in an Industrial Context

Hussein Al Hashimi

New trends in the software development industry have led many organisations to find a viable solution to their client's needs while producing quality software in a short period of time using cost-effective techniques within a changing and unstable environment. The use of spikes in agile software development (ASD) can enable organisations to produce quality software by employing the required technical expertise, planning the entire development cycle, and ensuring that the client's requirements are adhered to. Spikes can be an essential component of the agile development cycle because they assist the teams to identify any uncertainty in a user story, leading to a more effective solution to the problem. A spike is a period of time spent investigating aspects of a project, especially relating to a user story, such as the appropriate architecture, user interface or technology to be used, so that these can be identified more precisely.

This study aims to examine the use of spikes in ASD empirically. It explores the role, efficiency, and effectiveness of spikes in various software development domains through the application of different agile methods. An exploratory research design is adopted to achieve this purpose, using mixed methods to collect qualitative and quantitative data concurrently from the agile practitioners recruited to the study. Moreover, this study explores the common spikes success factors (CSSFs) influencing their use. This is achieved using semi-structured interviews, focus groups, and questionnaires with software development practitioners. The findings are validated using case studies conducted with three focus groups from different software companies, as well as 17 individual agile practitioners from around the world.

The study findings showed the impact of spikes on risk reduction and their role in the estimation process, as well as how their use is related to the team's understanding, and the consistency and reliability of the story estimate. Also, the study demonstrated that the effective application of spikes might be influenced by various factors related to the project, process, organisation, people, and technical aspects of the software being developed. A variety of statistical test analyses were applied to provide quantitative evidence of the strength of the relationships between variables and detect outliers or other anomalous data points.

The study establishes that spikes are primarily used for risk management and are found to be efficient and effective, thereby improving the quality of software products, as demonstrated in Chapter 5 of this thesis. Furthermore, the study identified the CSSFs that enhance the application of spikes in ASD by ensuring that the spike is well-planned, structured, and focused, which aids in achieving the desired outcome. The CSSFs were evaluated by agile practitioners in phase three of this study (see Chapter 8).

Table of Contents

Table of Contents i

List of Tables vii

List of Figures viii

Research Thesis: Declaration of Authorship x

Acknowledgements xii

List of Abbreviations xiv

Chapter 1: Introduction 1

1.1. Background 3

1.1.1. Principles Behind Agile 3

1.1.2. Agile Methodologies 4

1.1.3. Roles in Agile Methods 8

1.1.4. Overview of Spikes and Prototyping 9

1.1.5. A Big Picture of Agile Enterprise 13

1.1.6. Rapid Prototyping and Agile Software Development 14

1.2. Thesis Motivation 15

1.3. Research Questions and Objectives 16

1.4. Contributions of this Research 17

1.6. Structure of the Thesis 17

Chapter 2: Literature Review 20

2.1. Inclusion and Exclusion Criteria 21

2.2. Risk-Based Spikes 23

2.3. Quality Assurance for Agile Software Development 25

2.4. Testing in agile software development (ASD) 27

2.5. Agile information technology (IT) security implementation 28

2.6. Agile Spikes in Different Development Domains 29

2.6.1. Agile development for big data 29

2.6.2. Agile development for data warehousing 30

2.6.3. Agile development for computer science education 30

2.6.4. Agile development for blockchain 31

2.6.5. Agile user experience (UX) design 32

2.6.6. Agile development in cloud computing 32

2.6.7. Agile development in the Internet of Things (IoT) 34

2.7. Research Gap 34

2.8. Summary 35

Chapter 3: Research Methodology 38

3.1. Research Philosophy 40

3.2. Research Approach 40
3.3. Research Strategies and Data Collection Methods..41
3.4. Time Horizon 45
3.7. Summary 45
Chapter 4: Preparation for Data Collection Concerning the Roles, Efficiency, and Efficacy of Spikes 48
4.1. Interview Method 48
4.1.1. Designing the interview questions...48
4.1.2. Interview sample size...50
4.1.3. Interview piloting 50
4.2. Questionnaire Method...51
4.2.1. Designing the questionnaire 51
4.2.2. Questionnaire sample size 51
4.2.3. Questionnaire piloting 52
4.3. Ethical Considerations...53
4.4. Data Analysis Instruments 53
Chapter 5: Results of Practitioners' Responses Concerning the Roles, Efficiency, and Efficacy of Spikes 56
5.1. Interview Data Collection Process 56
5.2. Qualitative and Quantitative Analysis of Interviews 57
5.2.1. Participants' demographic information 58
5.2.2. Roles of spikes in agile software development (ASD)...61
5.2.3. Efficiency of spikes in agile software development (ASD) 66
5.2.4. Effectiveness of spikes in risk management and software domains 72
5.3. Questionnaire Data Collection Process..82
5.4. Quantitative Analysis of Questionnaires 82
5.4.1. Participants' demographic information 82
5.4.2. Agile methodologies 85
5.4.3. Effectiveness of spikes in risk management 85
5.4.4. Agile spikes in various software domains 87
5.4.5. Effectiveness and Efficiency of spikes in ASD 87
5.4.6. Frequency of use of spikes in agile software development (ASD)..........................89
5.4.7. Likelihood of applying spikes to estimate user stories 90
5.4.8. The most effective type of spikes 91
5.4.9. Practitioners' perspectives on recommending spikes in agile software development (ASD) 96
5.5. Statistical Tests for Quantitative Analysis..97
5.5.1. Reliability analysis 97

5.5.2. Correlation analysis 97
5.5.3. Regression analysis 99
5.5.4. One-sample t-test 100
5.5.5. Univariate ANOVA 101
5.5.6. Scatter plot 102
5.6. Qualitative Analysis of Questionnaires 103
5.6.1. Practitioners’ opinions of the roles of spikes in ASD 103
5.6.2. Practitioners' perspectives on technical and functional spikes 104
5.6.3. Practitioners’ opinions of efficient use of spikes in agile software development (ASD) 105
5.6.4. Practitioners' perspectives on the use of spikes in estimating user stories 106
5.6.5. Practitioners' perspectives on the feasibility of risk management without the use of spikes during agile software development. 107
5.7. Discussion 108
5.7.1. Discussion of the interview findings 108
5.7.2. Discussion of the questionnaire findings 109
5.7.3. Threat to validity 111
5.8. Summary 112
Chapter 6: Preparation for Identifying Common Success Factors in the Use of Spikes and Case Studies 114
6.1. Identifying Common Success Factors in Applying Spikes 114
6.1.1. Interviews and focus groups 114
6.1.2. Questionnaire 115
6.2. Overview of the Case Study Approach 115
6.2.1. Embedded case study approach 116
6.2.2. Holistic case study approach 117
6.3. Case Study Objectives 117
6.4. Case Study Questions 118
6.5. Case Study Methods 119
6.5.1. Data collection methods 119
6.5.2. Data selection 120
6.5.3. Units of analysis 121
6.6. Case Study Process 122
6.6.1. Step 1: Preliminary evaluation of past projects 124
6.6.2. Step 2: Focus group meetings 125
6.6.3. Step 3: Case study planning 126
6.6.4. Step 4: Sprint planning and the application of spikes 126
6.6.5. Step 5: Sprint review and retrospective 126

6.7. Ethical Considerations....127
Chapter 7: Common Spike Success Factors (CSSFs)....129
7.1. Participant Demographics for Interviews and Focus Groups....130
7.2. Common Success Factors in Spike Applications (CSSFs)....134
7.2.1. Success factors based on participants' experiences....134
7.2.2. Factors that enhance the outcomes of spikes....140
7.2.3. Categories of success factors....145
7.2.4. Opinions of participants on the categorisation of factors....151
7.3. Common Success Factors of Spikes: Quantitative Analysis Results....153
7.3.1. Participants' demographic characteristics....153
7.3.2. Results of analysis of success factors....157
7.3.3. Participants' opinions on the categorisation of factors....165
7.3.4. Reliability analysis....168
7.3.5. Inferential analysis....168
7.4. Discussion....172
7.5. Summary....175
Chapter 8: Validation of the Roles, Efficiency, Efficacy, and Common Success Factors of Spikes using Case Studies....177
8.1. Organisation and Participant Selection....178
8.2. Use of Architectural, Design Spikes and Spike Solutions in Resolving Technical and Functional Issues....181
8.3. Roles of Spikes in Previous Software Projects....183
8.3.1. Risk management....183
8.3.2. Ensuring smooth and sustainable running of the project....184
8.3.3. Proof of Concept (POC)....185
8.3.4. Investigations....185
8.3.5. Decision-making....185
8.4. Effectiveness of Spikes in Fulfilling their Roles....186
8.5. Effectiveness of Spikes in Risk Management and Estimation....187
8.6. Skills Required for Development Teams to Apply Spikes Effectively....189
8.6.1. Technical skills and maturity....189
8.6.2. Being open-minded....189
8.6.3. Communication skills....189
8.6.4. Understanding the meaning and objectives of spikes....190
8.6.5. Time discipline....190
8.7. Objectives of Spikes in Recent Software Projects....191
8.7.1. Organisation A (FG1)....192
8.7.2. Organisation B (FG2)....193

8.7.3. Organisation C (FG3) 195
8.7.4. Individuals' experiences of spikes to achieve objectives 196
8.8 Total Number of Spikes Used in Sprint/Iteration and the Project 201
8.9. Use of Spikes during the Sprint/Iteration Retrospective 202
8.10. Defining the spike objectives in the project planning stage 203
8.11. Common Spike Success Factors (CSSFs) for Effective Application 204
8.12. Tools for Mitigating and Managing Risk in Agile Software Development (ASD) 206
8.13. Most Significant Challenges Impeding the Effective Use of Spikes 207
8.13.1. Lack of clarity in goals 208
8.13.2. Changes in requirements 208
8.13.3. Inadequate comprehension of spikes 208
8.13.4. Organisational culture 209
8.13.5. Lack of a cost-benefit analysis of spike application (no metrics to assess) 209
8.14. Summary 210
Chapter 9: Conclusions and Future Work 212
9.1. Conclusions 212
9.2. Research Contributions 217
9.2.1. Identifying the roles of spikes in agile software development (ASD) 217
9.2.2. Efficiency and effectiveness of spikes in ASD 217
9.2.3. Identifying common spike success factors (CSSFs) 218
9.2.4. Validating and evaluating the CSSFs 218
9.3. Research Implications 218
9.3.1. Implications for organisations 219
9.3.2. Implications for practitioners 219
9.3.3. Implications for researchers 219
9.4. Research Limitations and Challenges 220
9.5. Future Work 221
9.6. Closing Remarks 223
References 225
Appendix A: Research Ethics Forms 238
Appendix B: Interview Consent Form 240
Appendix C: Study Details 241
Appendix D: Practitioner Interviews for RQ1 to RQ3 244
Appendix E: Practitioner Interviews for RQ4 246
Appendix F: Codes and Themes 247
Appendix G: Study questionnaires for RQ1 to RQ3 255
Appendix H: Study questionnaire for RQ4 264
Appendix I: Outputs of Questionnaires Analysis for RQ1, RQ2, and RQ3 271

Appendix J: Outputs of Questionnaires Analysis for RQ4 287

List of Tables

Table 2.1: Overview of the domains discovered during the literature review 21
Table 2.2: Risk management plan (Douglass, 2016) 24
Table 3.1: Summary of the research methods....45
Table 4.1: Interview data requirements 48
Table 5.1: Interview duration in minutes....54
Table 5.2: Participants' experience, country of origin and agile roles....57
Table 5.3: Spike roles and applications in ASD 65
Table 5.4: Efficiency score for spikes in ASD....67
Table 5.5: Spike estimation roles, as deduced from participants....71
Table 5.6: Practitioners' opinion on the appropriate time to apply spikes 71
Table 5.7: The impact of spikes on project activities....71
Table 5.8: Effectiveness of spikes in reducing risk....73
Table 5.9: Practitioners' summary of spike factors that boost the quality of software products....76
Table 5.10: Practitioners' opinions on the most common uncertainty factors....76
Table 5.11: Spike applications in different domains....78
Table 5.12: Cross-tabulation of participants' roles with their work experience 84
Table 5.13: Summary distribution of scores for efficiency and effectiveness of spikes....88
Table 5.14: Type of spikes most used in ASD according to participants'/practitioners' opinions....92
Table 5.15: Likert items and summary statistics....95
Table 5.16: Reliability analysis for the efficiency and effectiveness of spikes in ASD 97
Table 5.17: Correlation between spike efficacy in domains, risk management, and efficiency in ASD 98
Table 5.18: Correlation matrix of Likert-scale items (Pearson's r) 98
Table 5.19: Analysis of variance....99
Table 5.20: Model summary 99
Table 5.21: Results of t-test for Likert-scale questions....101
Table 5.22: Test of difference between responses....102
Table 6.1: Summary of case study questions....119
Table 7.1: Demographic questions for interviews and focus groups 130
Table 7.2: The most common organisational factors for the successful application of spikes 147
Table 7.3: Summary of procedure/process factors 148
Table 7.4: Summary of project-related factors....149
Table 7.5: Summary of technical success factors 150
Table 7.6: Reliability analysis for the common success factors....168
Table 7.7: Summary responses of participants....169
Table 7.8: Correlation analysis for the common success factors....170
Table 7.9: One-sample t-test for the common success factors. 171
Table 8.1: Summary of focus groups who took part in the case studies....179
Table 8.2: Summary of individuals involved in the case studies....180
Table 8.3: Duration of each focus group session....180
Table 8.4: Duration of individual interviews....181
Table 8.5: Statistical measures of focus group durations....181
Table 8.6: Number of spikes used by the focus groups....202
Table 8.7: Number of spikes used by the individual participants....202
Table 8.8: Validation of common spike success factors (CSSFs)....205
Table 8.9: Other risk management tools/techniques used by the participants 206
Table 9.1: Summary of methods used in the study 213

List of Figures

Figure 1.1: Principles of Agile Manifesto (Beck et al., 2001) 4
Figure 1.2: Extreme programming process (Donovan Wells, J., 2000) 5
Figure 1.3: Overview of Scrum process (Schwaber, K., 2004) 7
Figure 1.4: Types and subtypes of spikes in ASD 12
Figure 1.5: Agile Enterprise Big Picture (Leffingwell, 2009) 14
Figure 2.1: Traditional process of risk management (Albadarneh et al., 2015) 23
Figure 2.2: Quality assurance and agile methods (Huo et al., 2004). 26
Figure 2.3: Strategies for agile testing (Isaacs, 2016) 27
Figure 2.4: Testing use over time (Hellmann et al., 2012) 28
Figure 2.5: Development time in days combining agile methods and cloud computing 33
Figure 3.1: Research methodology layers (adapted from Saunders et al., 2016) 38
Figure 3.2: Overview of the research process and stages 39
Figure 3.3: Inductive vs. deductive approaches....66
Figure 3.4: Research designs for mixed methods (Warfa, 2016) 42
Figure 3.5: Triangulation approaches used in the study....44
Figure 5.1: Themes identified in the interview transcripts....58
Figure 5.2: Participants' countries 59
Figure 5.3: Participants' roles 59
Figure 5.4: Agile methodologies used by practitioners 61
Figure 5.5: Likert rating scale for the effectiveness of spikes in ASD 66
Figure 5.6: Rating of spikes' efficiency in ASD 67
Figure 5.7: Employing spikes during sprint 69
Figure 5.8: Area of uncertainty in ASD....72
Figure 5.9: Rating of spikes' effectiveness in reducing risks....73
Figure 5.10: Most appropriate agile method to apply spikes....79
Figure 5.11: Practitioners' opinions of the effectiveness of spikes in reducing technical and functional risk....80
Figure 5.12: Participants experience of using agile software development and spikes....83
Figure 5.13: Participants' agile roles....83
Figure 5.14: Agile role in which respondents had the most experience 84
Figure 5.15: Most used agile methods by participants....85
Figure 5.16: Participants' opinions of the effectiveness of spikes in managing risk 86
Figure 5.17: Participants' views on the use of spikes to enhance the quality of software 86
Figure 5.18: Domains in which participants apply agile spikes....87
Figure 5.19: "Box and whisker" plot for efficiency and effectiveness scores of agile spikes 89
Figure 5.20: Frequency of using spikes 90
Figure 5.21: Likelihood of practitioners applying spikes in estimating user stories 91
Figure 5.22: The most effective type of spikes 92
Figure 5.23: Likert-scale item frequency summary 94
Figure 5.24: Likelihood of participants advising others to use spikes in ASD 96
Figure 5.25: Scatter plot of spike experience vs. agile experience 102
Figure 6.1: Embedded case study approach 117
Figure 6.2: Case study process scenarios....124
Figure 7.1: Interview process addressing RQ4....129
Figure 7.2: The experience in the agile and spike fields 131
Figure 7.3: The roles of agile practitioners 131

Figure 7.4: Agile methodologies used........132
Figure 7.5: Organisational sectors of the participants........132
Figure 7.6: Distribution of organisation size........133
Figure 7.7: Geographical distribution of the participants (FG and Individuals)........133
Figure 7.8: Common success factors for spike applications........144
Figure 7.9: People factors for the successful application of spikes........146
Figure 7.10: Participants' opinions on the categorisation of success factors........151
Figure 7.11: Common success factor categories........152
Figure 7.12: Participants' experience in ASD........153
Figure 7.13: Participants' experience in spike........154
Figure 7.14: Participants' roles in their organisations........155
Figure 7.15: Participant's organisational sectors........155
Figure 7.16: Size of participants' organisations........156
Figure 7.17: Agile methodologies used by survey participants........157
Figure 7.18: Participants' opinions of the existence of success factors for spike application........158
Figure 7.19: Proposed organisation-related factors for successful spike application........159
Figure 7.20: Proposed people-related factors for successful spike application........160
Figure 7.21: Proposed process-related factors for successful spike application........162
Figure 7.22: Proposed technical factors for successful spike application........163
Figure 7.23: Proposed project-related factors for successful spike application........165
Figure 7.24: Participants' perspectives on categorising factors........165
Figure 7.25: Opinions on the existence and importance of success factors........167
Figure 7.26: Summary of participants' responses........169
Figure 8.1: An overview of the case study........178
Figure 8.2: The agile methodology used by the participants in the case studies........182
Figure 8.3: Participants' opinions concerning the effectiveness of spikes in fulfilling their roles..186
Figure 8.4: Participants' opinions concerning the effectiveness of spikes in risk management and estimation........188
Figure 8.5: The participants' perspectives on spikes achieving their objectives........191
Figure 8.6: Example of a user story – FG1........192
Figure 8.7: Example of spike – FG1........193
Figure 8.8: Example of a user story – FG2........194
Figure 8.9: Example of spike – FG2........194
Figure 8.10: Example of a user story – FG3........195
Figure 8.11: Example of spike – FG3........196
Figure 8.12: Example of functional spike – CS_P6 project........199
Figure 8.13: Example of technical spike – CS_P6 project........200
Figure 8.14: Participants' perspectives on the use of spikes in project planning........203
Figure 8.15: Use of other risk management techniques by participants........206
Figure 9.1: Overview of the study process and methods........216

Research Thesis: Declaration of Authorship

Name: Hussein Al Hashimi

Title of thesis: **Usage and Perceptions of Spikes in Agile Software Development: An Exploratory Study in an Industrial Context**

I declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

I confirm that:

1. this work was done wholly or mainly while in candidature for a research degree at this University;
2. where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
3. where I have consulted the published work of others, this is always clearly attributed;
4. where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
5. I have acknowledged all main sources of help;
6. where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
7. parts of this work have been published as:

   Al Hashimi, H. and Gravell, A. 2022. Common spike success factors: An industrial investigation within agile software development. 12th International Conference on Software Technology and Engineering (ICSTE 2022). IEEE, Osaka, Japan.

   Al Hashimi, H. and Gravell, A. 2022. Validation of the roles, efficiency, efficacy, and common success factors of spikes in agile software development using case studies. In *IEEE International Conference on Advanced Aspects of Software Engineering* (ICAASE). IEEE, Constantine, Algeria. (Accepted).

   Al Hashimi, H. and Gravell, A. 2021. Spikes in agile software development: An empirical study. *The International Conference on Computational Science & Computational Intelligence* (CSCI2020). IEEE, Las Vegas, USA.

   Al Hashimi, H., Altaleb, A. R. and Gravell, A. 2020. An empirical investigation of spikes in agile software development. *International Conference on European Symposium on Software Engineering* (ESSE2020). ACM, Roma, Italy.

Altaleb, A. R., Al Hashimi, H. and Gravell, A. 2020. A case study validation of the pair-estimation technique in effort estimation of mobile app development using agile processes. In *2020 10th International Conference on Advanced Computer Information Technologies* (ACIT). IEEE. pp. 469-473. (doi:10.1109/ACIT49673.2020.9208985).

Al Hashimi, H. and Gravell, A. 2019. A critical review of the use of spikes in agile software development. In *ICSEA 2019: The Fourteenth International Conference on Software Engineering Advances*. IARIA. pp. 154-162.

Signature: Date: 04 April 2023

Acknowledgements

First and foremost, I would like to thank Almighty Allah, who has granted me wisdom, strength, and good health and provided me with faith and patience to complete this thesis.

I'm deeply grateful to everyone I came across during the PhD who offered me assistance, encouragement, and concern. Among those people, I would like to convey my special thanks and appreciation to my wonderful supervisor Dr Andrew Gravell for his invaluable guidance, motivation, patience, support, and endless encouragement to complete this research, especially during difficult times. This work would not have been possible without his feedback and constant supervision. Also, I would thank my second supervisor Professor Steve Gunn for his guidance and support throughout my doctoral journey.

I would like to express my sincerest gratitude to Professor Michael Butler, Dr Andrew Sogokon, Dr Reza Rezazadeh, and Dr Tim Storer for their valuable feedback and recommendations. Their insightful contributions have been invaluable in helping me to refine and improve the overall quality of my thesis. I'm thankful for having the opportunity to benefit from their expertise and knowledge.

Likewise, I would like to extend my deepest gratitude to Mrs Josephine Axtell CPS Research Group Administrator, for her help, support, and her dedication to serving the members of this research group. Also, I would like to express my thanks to the University of Southampton for providing us with all the necessary tools to keep us on board during Covid 19.

Many thanks and appreciation are extended to all organisations and practitioners who contributed to the research's successful completion.

I would also like to convey my heartfelt gratitude to my friends and colleagues at the CPS Research Group, Dr Abdullah Altaleb and Dr Ahmed Alenazi, for their guidance and collaboration during my study. Special gratitude also goes to Professor Alaaedin Hafez, Professor Hassan Mathkour, and Professor Abdulmenim Artoli for their continuous support over the past years.

Additionally, I would like to express my sincere appreciation to the Saudi Arabian government, represented by the Cultural Bureau in London (SACB) and King Saud University, for granting me the opportunity to pursue my academic endeavours and obtain a doctoral degree.

Finally, I would like to acknowledge my family, particularly my mother and father, who motivated and unconditionally supported me in achieving my life's ultimate goals. I also would like to express my gratitude to my wife for her endless support during my PhD. In addition, I am grateful to my brothers Mansour, Sultan, and Abdullah and sisters Badriyah, Ohood, Afaf, and Weaam for their unparalleled guidance, support, and encouragement.

List of Abbreviations

AABA	Architecture-centric agile big data analytics
API	Application programming interface
ARTs	Agile release trains
ASD	Agile software development
ATDD	Acceptance test-driven development
AWS	Amazon Web Services
BA	Business analyst
BBS	Blockchain Based Software
BDA	Big Data Analytics
BI	Business intelligence
BLE	Bluetooth low energy
CFA	Confirmatory factor analysis
CI/CD	Continuous integration/continuous delivery
CSSFs	Common spike success factors
DB	Database
DPA	Data Protection Act
DWBI	Data warehousing/business intelligence
EFA	Exploratory factor analysis
ERGO II	Ethics and Research Governance Online
ETA	Estimated time allocation
ETL	Extract, transform, and load
FMEA	Failure modes and effects analysis
IDE	Integrated development environment
ID&V	Identity and verification
IOS	iPhone Operating System
IoT	Internet of Things
IPCMS	Integrated Process Control and Maintenance System
IVR	Interactive Voice Response
LESS	Large-scale scrum
MVP	Minimum viable product

NHS	National Health Service
OS	Operating system
OTP	One time password
PBI	Product backlog item
PBX	Private branch exchange
PCL	Portable class library
PI	Program increment
PM	Project manager
PMI	Project Management Institute
PMO	Project Management Office
PO	Product owner
POC	Proof of concept
RAD	Rapid application development
RAID	Risks, assumptions, issues, and dependencies
RDS	Relational database service
RI	Reference implementation
ROI	Return on investment
SAs	Solution architects
SAFe	Scaled agile framework
SDLC	Software development life cycle
SLR	Systematic Literature Review
SME	Subject matter expert
UAT	User acceptance testing
UX	User experience
UI	User interface
WIP	Work in progress
XP	eXtreme Programming

Chapter 1: Introduction

Agile is a philosophy for guiding teams and software projects towards the creation and release of software products. Agile is a collection of software engineering procedures constructed on common rules and concepts, and it provides a platform to help teams, offering a continuously changing functional and technical environment. A focus on speedy delivery of business value is maintained (Moniruzzaman and Hossain, 2013). As a result, it significantly mitigates the risks associated with software development.

Agile Software Development (ASD) is a blanket term for a set of processes and techniques based on the principles and values described in the Agile Manifesto, which was published in 2001 by a group of autonomous experts and practitioners in software development (Beck et al., 2001). Agile development is about enhancing the collaboration of teams ranging from two to 20 people. Software products and artefacts develop gradually through mutual cooperation among these self-organising, cross-functional teams, using acceptable and appropriate practices for their respective domains (Hoda, Noble and Marshall, 2013).

There are four values in ASD, commonly known as the Agile Manifesto (Beck et al., 2001):

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan.

The Manifesto clarifies these items by concluding: ‘That is, while there is value in the items on the right, we value items on the left more’.

The agile approach provides several advantages, and customer satisfaction and less miscommunication are the two major ones. Working software is developed and delivered frequently in agile, and it is adaptable to changing requirements. Another benefit is good team cohesion and including stand-up meetings as practice that can be adopted to enhance communication in Scrum methodology.

Despite the benefits of agile, it has some flaws too. The approach has less emphasis on documentation. Thus, problems can arise if the outcome is not clear from the customer representative’s side. In addition, using the agile approach at the wrong time can cause problems. Agile is appropriate when new changes need to be added. According to Hoda and others, when there is a need to implement a new feature in a very short span of time and when there is a dynamic change in user needs (Hoda, Noble and Marshall, 2013).

In the software world, risk is the factor that most influences the project's success. Due to the many risks that can result in software not working properly, risk management is required in software engineering and development (Boehm, 1991).

In agile methodology, spikes are used in identifying an issue and providing a short confirmation of an idea to examine an issue further. Spikes also incorporate the testing of distinctive strategies to accomplish a similar outcome, just like testing to affirm that the ideal outcome is achievable through the present ventured approach. For instance, a group may play out a spike to check whether the members should code an application in one language rather than another (Moran, 2014). This concept is defined and explained further in section 1.1.4.

New software development projects will usually include novel elements whether in the software application or the knowledge required to complete it. These novel elements pose challenges to the project's timely and successful completion. These risks can be divided into two categories: a project team cannot provide an accurate quote or allot team resources if it does not know how some element of a project should be completed, and a project will most likely be delayed or fail. Therefore, spike is a risk-reduction activity in agile, where skill, gaps in knowledge, and technology can be spotted and addressed early on (Woodward et al., 2013).

The primary goal of this study is to investigate the various aspects of spikes in agile, including how they are used and how their behaviour differs from traditional agile user stories. Moreover, to measure the effectiveness, efficiency, and productivity of spikes and their different roles. Furthermore, establishing the most common factors that may help practitioners improve their agile process to employ spikes effectively.

The review highlights spikes' usage in agile development processes and covers several techniques and approaches. It addresses the basic concepts of agile spikes with respect to their implementation in agile methodologies by considering how they are used in risk management.

1.1. Background

This section aims to provide an overview of the main area of the research and related topics. Firstly, the background of agile principles, methods, and roles is discussed. This gives the overall context for this research since the agile development practices provide a disciplined, production-like ability to fulfil obligations and rapidly evolve a system to meet customers' needs (Leffingwell, 2010). Secondly, an overview of spikes is presented. This involves definitions of spikes and the importance of their forms in eliminating or reducing the risks associated with project facets. Thirdly, a big picture of agile enterprise is provided to illustrate the spikes within it (Leffingwell, 2010). Finally, a comparison between rapid prototyping and agile development is offered to highlight the differences and how they might be combined.

1.1.1. Principles Behind Agile

There are “twelve principles behind Agile Manifesto” (Beck et al., 2001), as shown in Figure 1.1. It is necessary to achieve customer satisfaction by repeated, one after the other, continuous delivery of a ready-to-use product. Changes to the requirements must be considered in the development, even if they arrive late. Working software is delivered frequently, at intervals of two weeks to two months, with preference to the shorter timescale. The principal measure of success or failure is the working software. Developers, business and technical personnel must work together daily throughout the period of the project. The medium of communication is face-to-face conversation, as it is the most efficient. Projects are built around individuals who are motivated, and they are provided with the support and environment that they require and trusted to get the job done (Fowler and Highsmith, 2001).

Proper, continuous attention is given to good design and technical excellence. Self-organised teams are preferred, because the best design, architecture and requirements emerge from these self-organising teams. Teams reflect on how to be more efficient and effective and adjust their behaviour accordingly. The agile process promotes development that can be sustained, and the users, sponsor and developers should be well aware of and able to maintain the momentum indefinitely. Last but not least, Agile encourages making software development process in a simple way, because agile is all about simplicity (Fowler and Highsmith, 2001).

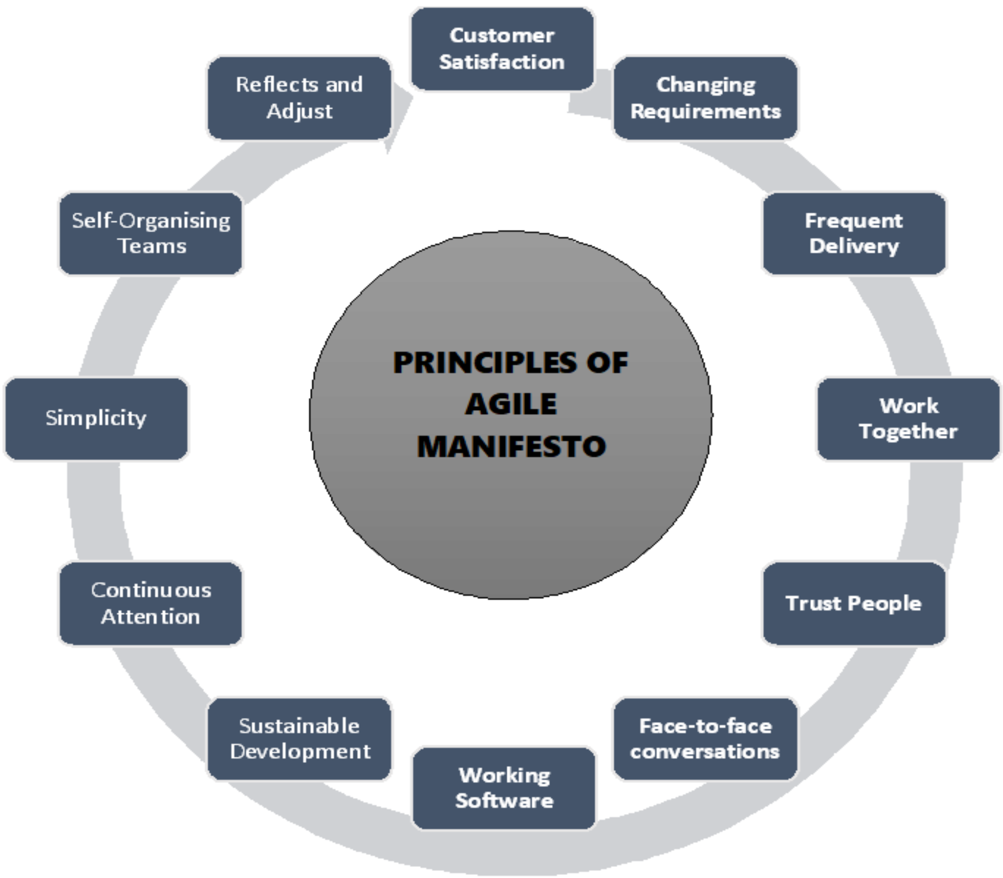


Figure 1.1: Principles of Agile Manifesto (Beck et al., 2001)

1.1.2. Agile Methodologies

There are several agile methodologies. The most widely used are: eXtreme Programming (XP), Scrum, Dynamic System Development Methodology (DSDM), Lean, Crystal, Feature Driven Development (FDD), and the Kanban method (Moniruzzaman and Hossain, 2013). This section describes the most common methodologies.

Extreme Programming (XP) is a disciplined approach to producing high-quality software rapidly and consistently. Kent Beck introduced the approach in 1996 while working at Chrysler, heading the establishment of its payroll system (Abbas, Gravell and Wills, 2008). XP supports high customer engagement, rapid feedback loops, consistent planning, consistent testing, and the production of working software at regular intervals within very short periods of time. Clients work in partnership with the development team to define and prioritise the user stories (Beck and Gamma, 2000).

XP is applied particularly in small teams comprising fewer than 10 developers. Customers are normally part of the team, as they help in the approval of the process after meeting the end user's needs. In XP, the role of a spike is to secure access to the information required to mitigate risks of this technology-based approach and validate estimates (Canty, 2015). According to Leffingwell, the XP-originated user story was introduced and adopted as the primary currency for expressing application requirements in agile development practices (Leffingwell, 2010).

The XP process often begins with preliminary requirements that are used to generate an architecture spike of the system (see Figure 1.2). The following step in the process is to build a plan for the iterations in which elements of the system are created. Each iteration begins with an activity to plan the iteration, followed by a simple design, create unit tests, pair programming, continuous integration with the rest of the system, and unit testing. The final step in the process is the customer acceptance test (Wang, A., Sørensen, C.F. and Conradi, R., 2004).

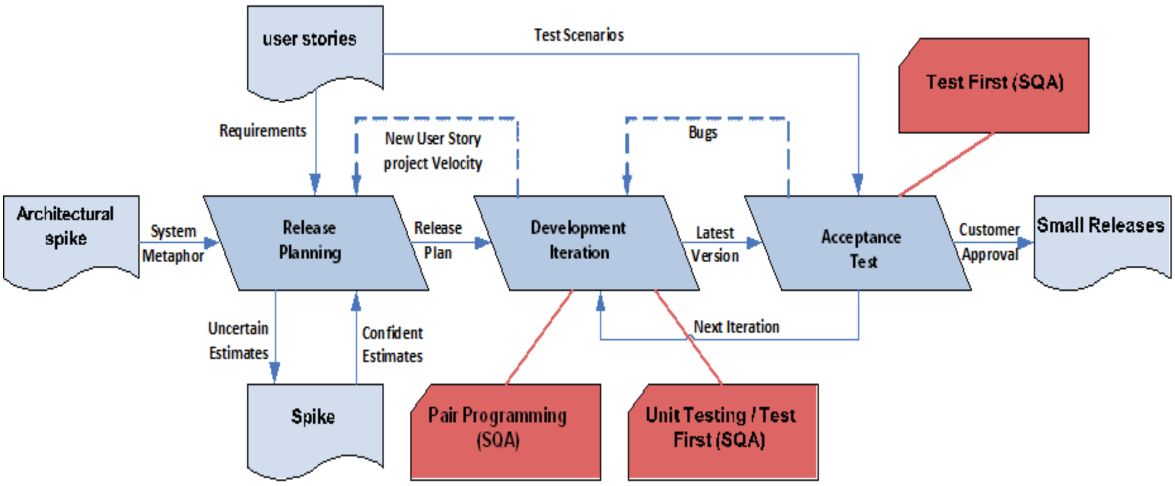


Figure 1.2: Extreme programming process (Donovan Wells, J., 2000)

Scrum is a framework that people can use to identify and resolve adaptive problems that are complex while creatively and productively producing software of enhanced value. It can assist development teams to create products that evolve with the ever-changing needs of users, which makes it a good choice for software development solutions. Scrum was first noted during a business object design and implementation workshop run in 1995 by two software industry pioneers, Ken Schwaber and Jeff Sutherland (Abbas, Gravell and Wills, 2008). The Scrum framework is usually a Scrum team with the affiliated functions, rules, artefacts, and regular events that put them together (Hoda, Noble and Marshall, 2013). Such a team usually includes: a Product Owner (PO), who can be a Project Manager (PM) or project sponsor; a Scrum Master as a team mentor or facilitator; and a project team of 5 to 10 developers, analysts, User Interface/Experience (UI/UX) designers and testers.

Schwaber (2004) stated that Scrum is an agile method used in project management. Backlogs can be described as the list of all things that need to be accomplished within a project (Schwaber, K., 2004). Backlogs are used by scrum teams to identify product features and manage a project, and product backlogs often contain technology and business features visualised for a product (Highsmith, 2002). The product backlog is a list of desired product features. Whereas the sprint backlog is a list of tasks that need to be performed during a sprint (Schwaber, K., 2004).

Figure 1.3 illustrates the essential elements of utilising scrum for agile software development. The product backlog should be prioritised by the PO and include everything that is desired in the product. The PO and development team are tasked with establishing sprint backlogs. Artefacts are by-products of Scrum activities that aid in transparency and act as guideline to the team. Scrum has three main artefacts, namely sprint backlogs, product backlogs and burn-down charts (Blankenship, Bussa and Millett, 2011).

At the beginning of each sprint, the team chooses some work from the product backlog and commits to finishing it during the sprint (2 to 4 weeks). The chosen work is called sprint backlog, which is the list of tasks required for the specified set of product backlog items to be performed in the sprint. Team members meet each day during the sprint to discuss their progress and any obstacles to completing the task for that sprint. This is referred to as the daily scrum, and it is shown in Figure 1.3 (24 hours). At the end of each sprint, the team produces a potentially shippable product increment.

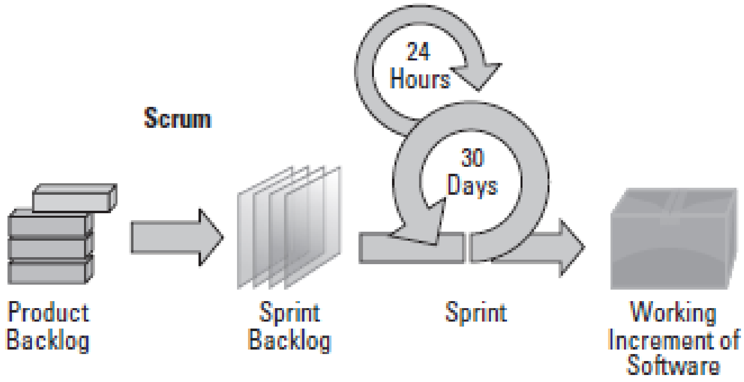


Figure 1.3: Scrum process overview (Schwaber, K., 2004)

Implementing scrum involves the following steps:

1. Splitting the organisation into smaller self-organising and cross-functional groups.
2. Splitting the work into a list of deliverables, i.e., a list of smaller and concrete products, and then sorting that list by prioritising the items.
3. For each team, estimating the relative effort and dividing the time into short, fixed-length iterations (called 'sprints', in agile terminology) of usually two to four weeks, producing potentially shippable code that can be demonstrated after each iteration.
4. In collaboration with the customer, creating an optimised release plan and prioritising the deliverables. These deliverables and work priorities are based on the insights obtained by inspecting the release after each iteration; they provide a good retrospective for process optimisation (Maximini and Rauscher, 2018).

Kanban is a Japanese word, and its meaning is linked to the 'just in time' theory. David J. Anderson was a pioneer in the use of Kanban in software development, having employed it for the first time in 2004 when he was asked to improve Microsoft's software maintenance department. He accomplished his mission in 15 months by tripling the delivery rate and cutting lead time by up to 90%, with on-time delivery increasing from 0% to 98 % (Anderson, 2010).

Although Scrum and Kanban are agile methodologies, they differ in several aspects. For instance, Scrum requirements are managed in the form of artefacts through sprint backlog and product backlog, whereas Kanban uses its boards. The product delivery is continuous in Kanban, whereas it is per time-boxed in Scrum. Changes are not allowed in sprints. In contrast, it is permitted at any time in Kanban. Scrum has no formal testing approach, but Kanban does it after each work product is implemented (Matharu et al., 2015).

SAFe is a scalable and adaptable framework that enables organisations to deliver the world's most critical systems with the shortest feasible time lag while maintaining the highest quality and value for lean enterprises. It synchronises cooperation, alignment, and delivery across various agile teams as well as larger-scale programmes (Knaster and Leffingwell, 2018). SAFe was introduced to the public in 2011 and it is a combination of systems thinking and the capabilities of agile with lean product development. Its vast collection of knowledge is based on lean-agile values and principles, which guide the responsibilities, necessary activities, roles, and artefacts to attain better business outcomes (Knaster and Leffingwell, 2018).

1.1.3. Roles in Agile Methods

An agile team is a multifunctional team of professionals with the resources necessary to develop a working, properly tested release of a product. Each agile team member has a specific role in the team, and therefore functions are well defined to avoid overlap and duplication of responsibilities. A Scrum Master mentoring the Scrum team, who also facilitates with resources and helps when required, guides the team members to follow Scrum framework and enables close cooperation between all team roles and their respective functions (Hoda, Noble and Marshall, 2013). The Scrum Master is in charge of promoting and supporting Scrum in accordance with the Scrum Guide. Scrum Masters accomplish this by assisting all team members in understanding Scrum theory, practises, values and rules (Schwaber and Sutherland, 2011).

Another role – the Product Owner (PO) – drives the product from the business angle by defining the requirements, evaluating their priority, and determining the date and content of each release. This person takes an active role in planning the iteration and release meetings, as the client’s voice. The PO accepts and evaluates user stories that meet the defined acceptance criteria and definitions of ‘done’ (Lenarduzzi et al., 2018).

An agile team is self-organising, with 2 to 20 members who have an average working experience of around 6 to 10 years. Typically, a team comprises three or four developers, a tester, a technical lead, a PO and a Scrum Master (when the team follow the Scrum methodology). The agile team uses its expertise to work on tasks and to decide and plan the scope of the work (Hoda, Noble and Marshall, 2013).

1.1.4. Overview of Spikes and Prototyping

The term “spike” was introduced at the XP Universe conferences in the early 2000s as a type of test in coding and programming, now used to solve a technical problem or design in user stories or project facets. Spikes can be defined as a particular type of story, involving activities such as research, investigation, exploration, prototyping and design with the aim of reducing or driving out uncertainty or technical risks associated with either the user story or other project facets (Leffingwell, 2010). In addition, spikes in agile are used to determine the uncertainty of the project by collecting the relevant and required pieces of information to help in understanding its technical or functional requirements. Similarly, Leffingwell (2010) also claims that spikes are required when an agile team needs to resolve a specific technical problem or does not have enough information for user story estimation.

A formal definition of a spike by Cohn (2005) states that it is “a task included in an iteration plan being undertaken specifically to gain knowledge or answer a question”. Leffingwell (2010) describes spikes as a special story that drives out risk and uncertainty within a user story, specifically where knowledge is light or can be used as the basis of research to mitigate risk. In agile terminology (Miranda, Bourque and Abran, 2009), a spike is an experiment designed to learn something. In this case, the spike is to develop a user story to track how much effort is needed.

On the other hand, Bernhard Boar (1984) defined prototyping as a particular strategy for undertaking requirements definitions in which user needs are obtained, introduced, and sequentially refined by rapidly building a working model of the overall system in its operational context. Similarly, Cobb (2015) defined a prototype as a model that is constructed for the purpose of testing a potential notion or concept.

Pomberger (1994) claims that prototypes need to be made rapidly and at low cost. The idea of prototyping emerged in the 1980s because tools for efficient prototype production are difficult to produce and consequently were not available before then in a sufficient quantity and quality (Pomberger and Weinreich, 1994).

Despite its prominence in academic and business development literature, prototyping seems to be a technique that some commercial software engineers are unwilling to acknowledge or utilise. There are pressures on companies developing increasingly complex applications to gain a competitive edge by releasing software in ever-shortening time frames. Changes made during development must be reflected in the final product, which means the software must be flexible and adaptable (Kimmond, 1995).

Prototypes are designed to assess the viability of an idea and subsequently reproduced, enhanced, or scrapped partially or fully. The key distinction between prototyping and spikes is the level of isolation. Spikes are intended to resolve uncertainty associated with specific aspects of software development. They provide answers to specific questions concerning particular components of the system. In contrast, a prototype is intended to be a conceptual model of a system (Cobb, 2015).

The use of spikes in agile software development (ASD) was initially defined in the eXtreme Programming (XP) approach because spikes represent prototyping, exploration, investigation, design, and research activities (Beck and Gamma, 2000). According to Leffingwell, spikes in agile are demonstrated and estimated in the same way as other stories at the end of an iteration.

Leffingwell (2010) claims that spikes are responsible for providing the workflow and protocol used by Agile Release Trains (ARTs). These ARTs mentioned by Leffingwell were developed by the Scaled Agile Framework (SAFe), which are virtual institutions (between 50 and 125 people) constituting self-organising teams of the experts needed to determine and deliver value to the end-user (Knaster and Leffingwell, 2018). ARTs are a long-lived group of agile teams that develop and deliver solutions progressively, employing a succession of fixed-length iterations within a Program Increment (PI) timebox in collaboration with other stakeholders. The authors further stated that the ARTs help to determine the viability and feasibility of epic user stories through a process of evaluation, estimation, and prioritisation. First, the team should evaluate each user story to ensure that it meets the criteria for the project, such as scope, timeline, budget, and other constraints. Next, the team should estimate the effort required to complete the user story, including any dependencies and risks. Finally, the team should prioritise the user stories based on their estimated value versus the effort required. This process helps the ART to better understand the potential impact of each user story and ensure that the most valuable user stories are addressed (Leffingwell, 2010). Thus, the primary goal of integrating spikes in agile is to enhance the feasibility of user story estimates and minimise technical problems by providing a framework for breaking down a user story into smaller chunks of work that can be more easily estimated and allowing the team to identify and address potential problems before they become major issues. (Knaster and Leffingwell, 2018).

There are two major types of spikes: technical and functional. The technical spike is commonly used to assess the impact of new technology on present implementation, and it has two forms: architectural spikes and spike solutions, as depicted in Figure 1.4. A functional spike is used to determine the interaction with a new feature or implementation when the team is mainly concerned with the user's interaction with the system and has one form called a design spike (Leffingwell, 2010).

Technical spikes are used when conducting technical approaches to the solution domain (Leffingwell, 2010). Specifically, they determine the following: defining the processes in the decision over buy vs build; accessing technical approaches to a specific implementation; and building assurance for the chosen solution path (Morris, 2018). An example of the use of a technical spike is where the project team is looking to implement rule engines for a course-planning tool. An open-source framework provides the best solution, based on a rule set and evaluation criteria. The team discusses the best choice to gain sufficient insights to make this technical decision (Proestakis, 2018). On the other hand, functional spikes are used when a degree of uncertainty is identified and where there is a lack of understanding of the interaction between the software and its associated users to consolidate its benefits. Agile process permits the generation of documentation to evaluate the spike through user interface (UI) design mock-ups, process flow diagrams and wireframes. From such a spike, the project team may seek further details to mitigate a particular risk arising from the customers or stakeholders (Leffingwell, 2010).

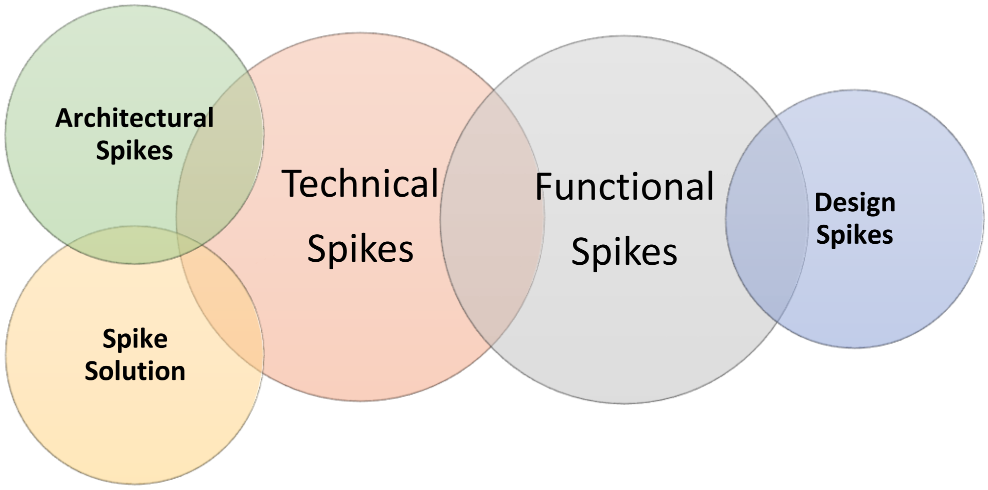


Figure 1.4: Types and subtypes of spikes in ASD

The use of architectural spikes is a time-fixed, technical risk-reduction technique, originating from XP, used to create a simple overall design or write sufficient code for a software program to explore its use (Wells, 2019). Sometimes, there will be stories involving risk or a technical issue that cannot be resolved at the time, and an architectural spike may be required (Hunt, 2018). An example provided by Wirfs-Brock (2015) shows that the current architecture is inadequate and requires support or assistance. Adding an architectural spike applies a backlog task for a member (or members) of the project team to perform investigations to make an architectural decision. The result of such a spike can be either a solution or the identification of further information to put into the project; however, it may also be an alternative solution, thus not exactly resolving an existing issue (Fuqua, 2016).

A spike solution is a simple program that explores all potential solutions, with the spike built to address the specific problem and devise a solution to resolve it. The best method for implementing a spike as a solution is to demonstrate the feature in question in order to obtain further information. The solution can be a small program or test to demonstrate the feature. Resolving spikes will always be from a practical point of view, not a theoretical perspective. The focus is purely on getting something working as a solution. The focus is not to write good code; rather, it is simply to focus on short-term results and identify a basic solution for the spike area in question (Shore and Warden, 2008).

A design spike is a timeframe in which team members are focused mainly on design questions. Design spikes can occur at the beginning of a project or at any time during the normal development process. Also, can be defined as a method that allows design teams to address complex User Experience (UX) issues within the constraints of the agile methodology. The timeframe of the design spike is not typically time boxed. Nevertheless, the team should strive to end the spike as quickly as possible so that team resources are utilised. Although it can be difficult to predict how long it will take to work through complex design issues, the PO could choose to time box the design spike if it becomes necessary. Each spike's objective is to deliver actionable design decisions in the form of wireframes, mockups, prototypes, or research, rather than potentially shippable product (Dimmick, 2012).

1.1.5. A Big Picture of Agile Enterprise

The Big Picture of Agile Enterprise was first published as a white paper by Leffingwell in 2009. The white paper claims that it is a model for implementing agile methods at enterprise scale (Leffingwell, 2009). The Big Picture acts as a process and organisational model for practising agile requirements (see Figure 1.5). The team level is where agile teams define, build and test user stories through iterations and releases. The team often differs depending on the size of an enterprise; small enterprises have few teams, while large enterprises have many that work hand in hand in building larger functionalities (Leffingwell, 2010). At the program level, multiple teams are tasked with developing large-scale systems' functionality through time-boxed iterations known as ART. At the portfolio level, investment themes are used to drive investment priorities and portfolio visions of an enterprise. Agile epic can be described as a large volume of work that can be broken into specific functions, usually referred to as user stories, based on the client's requirements (Conboy et al., 2013).

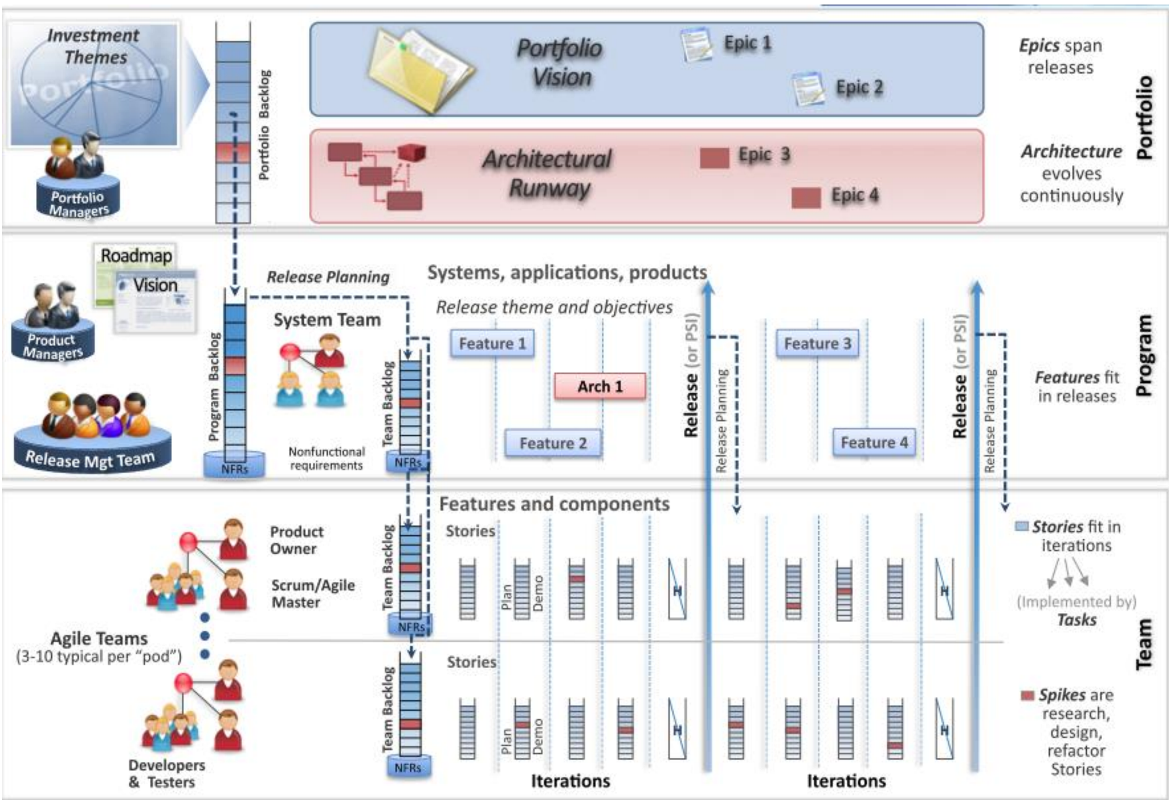


Figure 1.5: Agile Enterprise Big Picture (Leffingwell, 2009)

Agile teams are particularly self-organising in delivering software components or features and can reorganise themselves due to the work in program backlogs. There are three to 10 or so teams that are tasked with making sure that they build a system. This team comprises the PO, scrum or agile master, developers and testers, numbering between four and six individuals. There can also be tech lead and test automation experts. The POs determine and prioritise the user requirements. Spikes are usually accepted by POs once there is fulfilment of the acceptance criteria for spikes but should be used with caution since they do not deliver user value. They also maintain the product backlog; that is, the catalogue of tasks that need to be done in a project. The PO also works hand in hand with other team members by organising the team for the product backlog (Leffingwell, 2010).

1.1.6. Rapid Prototyping and Agile Software Development

This section highlights the benefits of adopting rapid prototyping to agile software development. Rapid prototyping refers to a set of techniques to visualise and model the working part of a product or system. It is helpful in quickly checking the functionality and testing a system for multiple ideas using updated prototypes and multiple short cycles. Usually, it is produced using a rapid prototyping tool for cost-effective and fast turnaround (Merrill, 2018). In the past few years, the excitement generated by rapid prototyping has led to software engineers and technologists creating more advanced rapid prototyping machines with improved features and declined cost. Because of these enhancements, many organisations have started adopting rapid prototyping as a development technique to compress the product development time (Vinodh et al., 2010).

Many technologists and developers have become experts in development using such technologies, especially in making models and developing prototypes; however, not many are familiar with recent enhancements in rapid prototyping and manufacturing methodologies, which include the implementation of agile alongside rapid prototyping. Agile manufacturing can help these prototypes to work better in iterations and to provide working software as early as possible (Vinodh et al., 2010).

1.2. Thesis Motivation

The increasing use of agile methodology has been triggered by the variety of its benefits for software development. Agile could save time and cost, as minimal documentation is involved when this methodology is used in comparison to traditional software development methods (Hoda, Noble and Marshall, 2012). Douglass claims (2016) that the agile approach helps in verifying and validating the requirements by allowing customers or stakeholders to be involved regularly in the development process to get their feedback at the end of each iteration. Moreover, agile teams experience lightweight processes when using agile methodology. Accordingly, rapid delivery is witnessed, enhancing the business value. In addition, the users are able to see the benefits of this approach through real-time updates during the project since agile development offers significant transparency (Shankarmani et al., 2012).

Spikes are used in agile software development to aid in research and drive out risk and uncertainty associated either with the user story or with project facets (Hunt, 2018). For this reason, several benefits of spikes are witnessed by agile teams such as allowing teams to familiarise themselves with a new domain or technology. Sometimes the user's stories might be too big to be estimated thus spikes become essential in breaking those stories into estimable pieces. Furthermore, some stories may have significant risks, and spikes are applied here to enable prototyping, investigation, and research that will be imperative to mitigate these risks (Leffingwell, 2010).

The potential for confusion in spike practices and other prototyping techniques in agile software development among some development teams motivated the researcher to conduct this study (Layton, M. 2019). A common belief in the industry is to treat spikes as stories. Nicolette (2014) claims this could impede the team's ability to maintain a smooth stream of productivity. A “story” has certain characteristics that differ from those of spikes. The differences are significant enough that it makes sense to treat spikes quite differently from stories. Using the word "story" tends to confuse teams who are not yet well-versed in agile thinking (Nicolette, 2014). Furthermore, the efficiency and efficacy of spikes have not been demonstrated previously. In addition, insufficient information in the literature review about spikes concepts, roles and implementation prompted the need to investigate these aspects. Such information will provide practitioners and researchers interested in software engineering with a better understanding of spikes and best practices for reaching the desired outcomes.

1.3. Research Questions and Objectives

Research questions are a key component of any research project or study. They provide structure and direction for the research process and help identify its purpose. They are used to focus the research process, guide the selection of relevant data, and organise and interpret findings. Research questions also help to determine the methodology and design of the research project and provide a framework for analysing and interpreting the data. They can be a powerful tool for understanding a particular problem or phenomenon (Saunders, Lewis and Thornhill, 2016). As derived from the literature review in the following chapter, this study’s research questions can be expressed as follows:

RQ1. What are the basic roles of spikes in agile software development?

RQ2. What are the agile teams’ opinions on the efficiency of spikes in ASD?

- (a) In what ways can spikes estimate user stories, effort and delivery time in agile software development?
- (b) When should agile teams use spikes during software development?

RQ3. What are agile teams’ thoughts on the effectiveness of spikes in managing risk?

- How are spikes used to identify uncertainty in agile software development projects?

RQ4. What are the most common factors that help agile teams to use spikes successfully?

The following research objectives were formulated in response to the aforementioned research questions and sub-questions in order to advance the study's aims:

- To identify the roles of spikes in ASD and risk management.
- To explore the use of spikes in various software development domains and methods.
- To explore the efficiency and effectiveness of spikes within agile software development.
- To identify the causes of uncertainty in agile software development.
- To establish the common success factors that help to apply spikes effectively.
- To evaluate the practices and implementations of spikes in different software projects.

1.4. Contributions of this Research

This research adds to the existing knowledge of spike applications through four main contributions based on the research gaps identified in Chapter 2. First, the roles of spikes were determined during the first phase of this study to answer the first research question. Second, the efficiency and effectiveness of spikes within ASD have been demonstrated. Third, the list of factors identified by participants as influential in the successful application of spikes was the most notable contribution determined during the second phase of this study. Finally, validating the findings obtained in phase one and two addressing the four research questions by seeking software development practitioners' opinions and perspectives on roles, efficiency, effectiveness, and common success factors for spikes (CSSFs) covered in this research.

1.6. Structure of the Thesis

This thesis comprises 9 chapters. Chapter 1 gives an introduction to this research, setting out the motivations for conducting it and its objectives. Moreover, it provides a brief background to agile principles, methods, roles, spikes and prototyping, and agile enterprise. Chapter 2 presents a review of the literature on the domains associated with agile spikes as well as the gaps identified as a result of this review. Chapter 3 outlines the research methodology and approaches used in this study. Chapter 4 presents the preparation for data collection, going through the design of the study tools and obtaining ethics committee approval to answer Research Questions (RQ) RQ1, RQ2, and RQ3. This is followed in Chapter 5 by an analysis and interpretation of the interviews and questionnaires that answered the three questions. Chapter 6 provides the details of the preparation for identifying Common Success Factors (CSSFs) in the use of spikes (RQ4) and case studies for this research. The results of the analysis addressing RQ4 concerning the most common factors in the successful application of spikes are presented in Chapter 7. Chapter 8 discusses the findings of the case studies conducted with three focus groups and 17 individual participants from various companies and sectors. Chapter 9 summarises the study's overall findings based on the research questions and then discusses the contributions, implications, challenges, and limitations. Finally, potential future work avenues are discussed in light of the study findings.

Chapter 2: Literature Review

The previous chapter discussed the agile methods, prototyping and spikes to provide fundamental knowledge to the reader before delving into the literature. As this research aims to investigate the spike applications in ASD, it is essential to review the available literature to identify the gaps and thus, the research questions and objectives can be formulated.

This chapter presents a narrative review conducted following (Dudovskiy, 2018) and (Snyder, 2019) guidelines to synthesise relevant studies and identify research gaps. The review sought to determine and comprehend all potentially relevant research domains connected to the studied topic and summarise and present the available research on spikes and related work. Finally, the chapter will discuss the research gaps in the literature from which the research questions and objectives arose, alongside the initial motivations outlined in section 1.2.

In this literature review, the focus will mainly be on spikes to achieve two important objectives.

- To establish how spikes are used in different domains to mitigate risks, estimate user stories, and provide a quality assurance platform for the development teams.
- To identify gaps that have not been explored extensively regarding the application of spikes in agile software development.

Table 2.1 summarises the literature review, focusing on the themes or domains of agile spikes across the included articles. This table is discussed in section 2.8.

Table 2.1: Overview of the domains discovered during the literature review

Spike Theme	Source	Purpose of Study	Research Design	Conclusion
Risk-based spike	Chen, Kazman, and Haziyev (2016)	To use spikes for addressing the risks of a project by breaking down the story into smaller components	Descriptive and Exploratory (10 case studies)	An alternate programming language can accelerate advancement or reveal that the chosen language cannot be used. These spikes would be added to the excess as alleviation activities.
	Albadarneh (2015)	To use spikes in testing to affirm that the ideal outcome is achievable	Comparative Research	Spikes incorporate testing distinctive strategies to accomplish a similar outcome, just like testing to affirm that the ideal outcome is achievable through the present ventured approach.
Quality Assurance	Huo et al. (2004)	To contrast the agile process with the waterfall model to demonstrate how agile methods can produce high-quality software despite the presence of time constraints and unstable requirements.	Comparative Research	The agile methodology does include QA-capable practices. Agile QA practices occur more frequently than their waterfall counterparts. The characteristics of an agile process make it possible for QA practices to be implemented at an early stage of the process.
Spikes in agile development for testing	Hellmann et al. (2012)	To integrate agile development testing in the SDLC	Review (SMR)	This paper conducted a systematic mapping of agile testing to investigate five research questions in order to provide guidance for future work in this field.
Spikes in agile development for security implementation	Rindell, Hyrynsalmi and Leppänen (2017)	To understand the benefits and drawbacks of using agile software methodologies in security-sensitive development environments.	Exploratory and Experimental (3 case studies) 18 participants	In order to reduce overhead costs and uncertainties during agile software development, proper security engineering planning, mechanisms and measures should be put in place and incorporated with various methodologies best suited for implementation, in order to assist with software development and provide a robust and secure end product.
	Siponen, Baskerville and Kuivalainen (2005)	To incorporate automated techniques to ensure that secure programming practices are implemented to ease the burden by building efficient, effective, and secure systems	Comparative Research	Although several issues of integrating security into agile are solved, those methods have many limitations. The combination of related methods can eliminate some weaknesses and improve the existing methods.
Spikes in agile development for big data	Chen, Kazman, and Haziyev (2016)	To effectively implement agile approaches on big data projects for reducing the risk exposure	Review (SMR)	The present agile analytics development practices do not possess architectural reinforcement for big data analytics, but they can help to tame project complexity, reduce uncertainty, and hence reduce project risk.
	Grady, Payne and Parker, (2017)	To discuss the implications of an agile process for Big data analytics (BDA) in cleansing, transformation, and analytics.	Descriptive and Exploratory	The proposed process model for BDA provides the additional agility to adjust to differences in architecture derived from data characteristics such as volume, velocity, variety, or variability.
Spikes in agile development for data warehousing	Rahman, Rutz and Akhter (2013)	To follow agile approaches on data warehousing projects and incorporate all the changes without changing the basic architecture of the software system	Descriptive	Agile methodologies are best known for identifying the inefficient areas for each phase in a data warehousing project. Finding different ways for reducing redundancy, wasted time, and inefficiency can be best serviced with system metrics development.
Spikes in agile development for computer science education	Woodward et al. (2013)	To take a potential benefit for computer science education by incorporating the techniques of agile development	Experimental (3 case studies) groups of students	The spike approach is appropriate if a study area can be broken down into highly specific and fairly isolated topics, even if it is outside the domain of software development.
Spikes in agile development for blockchain	Lenarduzzi et al. (2018)	To incorporate the potential advantage of the strengths of a blockchain to augment the vulnerability of the Agile/Lean approaches	Observational and Experimental (one case study)	Agile blockchain might be a good way to record the workflow and to track the enhancements of the product under work as well as the productivity of developers by using Smart Contracts as a payment support.
	Fahmideh et al. (2021)	To provide a complete review of the state-of-the-art of Blockchain-Based Software development.	Review (SLR)	The software engineering for BBS is not explicit as for many non-BBS ones. Ad hoc approaches are rife with partiality and subjectivity, making them unfit for business-critical BBS. To improve the maturity of this research field and make the transition from ad-hoc BBS development to more disciplined BBS engineering, more research is required.
Spikes in agile development for UX design	Da Silva et al. (2012)	To establish a framework for incorporating agile and user experience.	Semi-Experimental (Field Study) for 45 days (3 iterations) seven participants	The framework proposed aims at addressing different aspects of this integration, providing alternatives to the UX designer inserted in the agile context.
Spikes in agile development for cloud computing	Younas et al. (2018)	To ascertain the methods used in a cloud computing platform that are appropriate for agile development using systematic literature review.	Review (SLR)	Of all studies in the SLR, the techniques using existing tools were reported in 35%, simulations in 20%, and applications developed in 15%.
	Kalem, Donko and Boskovic (2013)	To illustrate the association between agile methods for software development with the cloud computing platform.	Descriptive	Software development with agile methods is compared with software development with agile methods using cloud computing. All advantages of the second approach are pointed out.
Spikes in agile development for IoT	Cheng et al. (2018)	To showcase service communication of agile IoT and orchestration platform using an event-driven service-oriented architecture (SOA) paradigm	Exploratory	The demo (Internet of heating network in smart city) shows that the IoT service communication and orchestration platform responds quickly to the dynamic changes in the physical world.

2.1. Inclusion and Exclusion Criteria

The definition of inclusion and exclusion criteria determines the validity and scope of a literature review's results (Meline, 2006). A key consideration is that if the inclusion criteria are too broad, the results may include poor-quality studies, decreasing confidence in the yield; where the criteria are too restrictive, the results are based on fewer studies than expected, thus they may not be generalisable (Meline, 2006). In obtaining the relevant materials to build the literature review, the Search terms highlighted below were used to come up with literature related to the study topic. However, some articles beyond the peer-reviewed academic literature which are nonetheless well-cited and authoritative sources were included in this thesis to expand the understanding of the research issues (e.g. the article by Leffingwell, 2009, blogs by Dimmick, 2012 and Layton, 2019, are cited in chapter 1, and Ogle, 2019, is cited in chapter 9). The inclusion and exclusion criteria in this research have been adopted based on Kitchenham and Charters (2007) guidelines to select the studies that provide evidence about the research question. The databases used to source the papers used in the literature reviews consisted mainly of those dominated by computer science publications. They included Scopus, Web of Science, IEEE Xplore, and ACM Digital Library as well as Google Scholar.

The inclusion criteria are as follows:

- All studies related to spikes in ASD:
    - ➢ Studies relating to risk management in ASD
    - ➢ Studies relating the roles of spikes in ASD.
- All studies contain the terms "user story", "spike or spikes", "agile spike or agile spikes", "spike in agile", "spike prototyping or spikes prototyping", "spike risk management", "agile testing", "spike testing", and "agile risk management".
- Studies from 2000 onwards. Studies were filtered by year specifically because the term "spike" was introduced at the XP Universe conferences in the early 2000s, as mentioned in previous chapter (see section 1.1.4)

The exclusion criteria are as follows:

- Spike articles in a field unrelated to ASD, such as Medicine, Electrical and Communication Engineering, Physics or Neurobiology.
- Any related article not published in credible sources such as:
    - ➢ Scholarly peer-reviewed articles and books.
    - ➢ Websites and blogs of well-known subject matter experts.
- Relevant articles involving difficulties in accessing it properly.
- Research articles not presented in the English language.

Other than the inclusion-exclusion criteria, snowballing was also utilised to get more articles in this literature. Snowballing is the process of identifying additional papers by using a paper reference list or citations (Wohlin, C., 2014). Due to the scarcity of literature on agile spikes, snowballing (backward and forward) was employed to obtain more relevant articles in this study.

2.2. Risk-Based Spikes

Risk management begins with risk identification, wherein a list of the threats to the project is compiled. The probability and severity of each potential threat are then evaluated through a process known as risk analysis. The significant risks are identified during risk prioritisation step based on their likelihood and impact. The next step in risk management planning is developing a strategy to address each major risk, and the implementation of this strategy is developing in risk resolution step. At last, the plan undergoes continuous risk monitoring (Albadarneh, Albadarneh and Quesef, 2015). Figure 2.1 depicts this process.

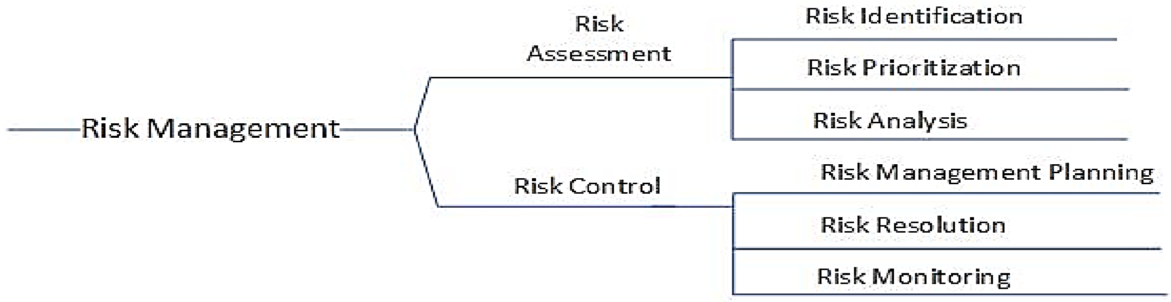


Figure 2.1: Traditional process of risk management (Albadarneh et al., 2015)

A risk-based spike is completed in light of a known risk or potential project risk (Moran, 2014). The group may discover that using an alternative programming language will accelerate advancement or that they cannot use the one originally chosen. These spikes are added to the backlog as alleviation activities (Chen, Kazman and Haziyev, 2016).

For projects on agile approaches, formal documentation and meetings are not required for risk management: the task is achieved by splitting it into scrum roles, artefacts and events. Many risks in agile projects can be eliminated by following agile principles. These significantly mitigate and eliminate project challenges and future failures (Moniruzzaman and Hossain, 2013; Moran, 2014). However, Moran (2014) claims that risk-based spikes should be used with a focus on 'fast failure' (that is, if a spike fails under every available approach, an early failure will cost far less than a later failure) to assist agile teams in eliminating or minimising significant risks. The author gives an example to support this claim, a software team can use a risk-based spike to find that the methodology the members arranged will not work and that they need another method. This revelation may, at first, appear to be an enormous issue; however, the sooner this issue is revealed, the sooner it tends to be resolved, with the goal that the group restores its concentration on conveying confidence to the client (Moran, 2014).

All potential risk to the project that are evaluated to be more severe than others are catalogued in the Risk Management Plan or Risk List. Typically, this is represented in a spreadsheet containing fields similar to those in Table 2.2.

Table 2.2: Risk management plan (Douglass, 2016)

Project Management Plan												
Risk ID	**Security (1-10)**	**Probability (0.0-1.0)**	**Risk (S*P)**	**Consequence Cost ($)**	**Probability Cost (S*P)**	**State (Open, Closed)**	**Priority**	**Occurrence date**	**Planned Iteration**	**Impacted Stakeholder**	**Owner**	**Mitigation Strategy**
1			0%		0.0							
2			0%		0.0							
3			0%		0.0							
4			0%		0.0							
5			0%		0.0							
6			0%		0.0							
7			0%		0.0							
8			0%		0.0							
9			0%		0.0							
10			0%		0.0							

A risk management plan is needed to ensure that the evaluation of risks is done accurately and in detail enough to mitigate risk effectively. Table 2.2 adapted from Douglass (2016) illustrates the sample risk management plan table that can be used to assess risks, their severity, costs and types. In the first six columns of the table, the details of the risk are recorded. This includes when it was identified, the type, nature severity and probability of occurrence. From this information, the risk itself is calculated based on severity and probability. Subsequently, Douglass points out that the probability cost of the risk can also be evaluated based on the consequence of the risk. The table further documents the status, priority, occurrence date, planned iteration, impacted stakeholder, owner, and mitigation strategy. The time the spike (the risk-reduction action) was completed is recorded as the occurrence data. Whereas in which the spike is planned to be implemented is referred to as the "planned iteration." The term "impacted stakeholder" is used to determine the various parties who may be affected. The owner is the person responsible for undertaking the spike. Finally, the mitigation strategy column can include the plan to reduce the risk (Douglass, 2016).

2.3. Quality Assurance for Agile Software Development

Software quality assurance is used to govern the process of building the desired quality into software products. Quality assurance may be divided into two main types: dynamic and static (Huo et al., 2004). The organisation, objectives, and selection of the specific techniques depend on the nature and requirements of the software project. In addition, the selection of these methodologies depends on the project criteria. The static technique includes an examination of project documentation by groups or individuals using several tools, such as project inspection of the requirements and reviewing the code technically. By contrast, the dynamic technique involves the execution of code and is generally used in agile development and processes (Huo et al., 2004).

Coram and Bohner (2005) provide an example to support their claim that quality assurance provides in-depth coverage of significant trends, technologies, concepts, and issues in agile software by focusing on the continuous integration, improvement, and automation aspects that will ensure a high level of customer satisfaction and an exceptional user experience.

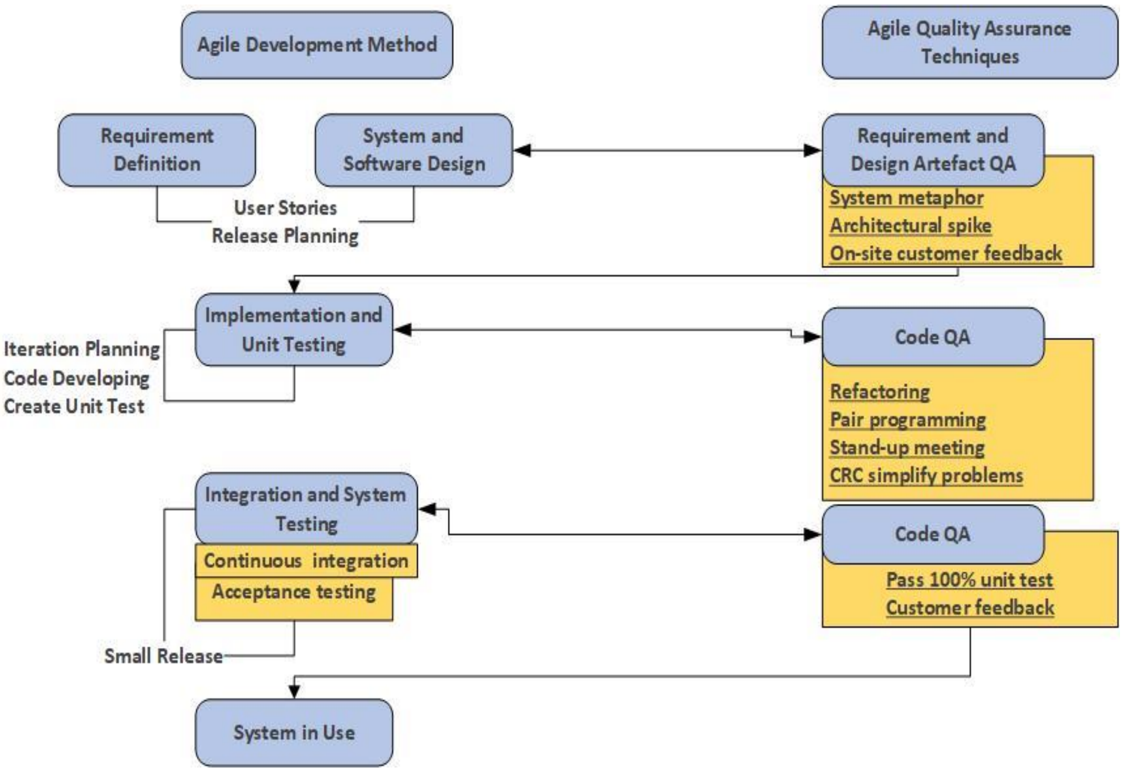


Figure 2.2: Quality assurance and agile methods (Huo et al., 2004)

In Figure 2.2, the generalised development lifecycle approach of agile methodology shows that some of the agile stages overlap. In ASD, some techniques integrate both quality assurance and agile functionality, which means that it has shifted some quality assurance responsibilities onto developers.

Agile development is a methodology that emphasises rapid iteration and responsiveness to customer feedback. In this approach, only a small amount of the total output is sent to quality assurance for testing and review in order to obtain rapid feedback. This allows the team to make changes and improvements quickly, responding to customer needs and expectations in the process (Huo et al., 2004).

The architectural spike is a fixed variable/time scope Product Backlog Item (PBI) that is incorporated to inform the software team that more investigation is required to maximise velocity. The effective implementation of architectural spikes helps the software team to achieve optimum estimates. Spikes are a series of investigations designed to solve as many problems as possible. The architectural spike technique is integrated to reduce the risks posed by XP (Ambler, 2002; Williams, 2012).

2.4. Testing in agile software development (ASD)

In ASD, testing is considered as the cornerstone, as most agile practices depend entirely on effective software testing. The efficacy and efficiency of agile methodology help in determining the ASD outcomes. In agile development, a test plan is updated and written for every release. This involves various testing types that are executed in a specific iteration, such as test results, test environments, infrastructure, and test data requirements (Isaacs, 2016). The general test plan in agile development includes the following: the testing scope; the new functionalities that need to be tested; the types or levels of testing, depending on complex features, performance, and load testing; the infrastructure considerations, risks, or mitigation plan; and the resourcing, milestones, and deliverables. Figure 2.3 illustrates the agile testing life cycle consists of four stages.

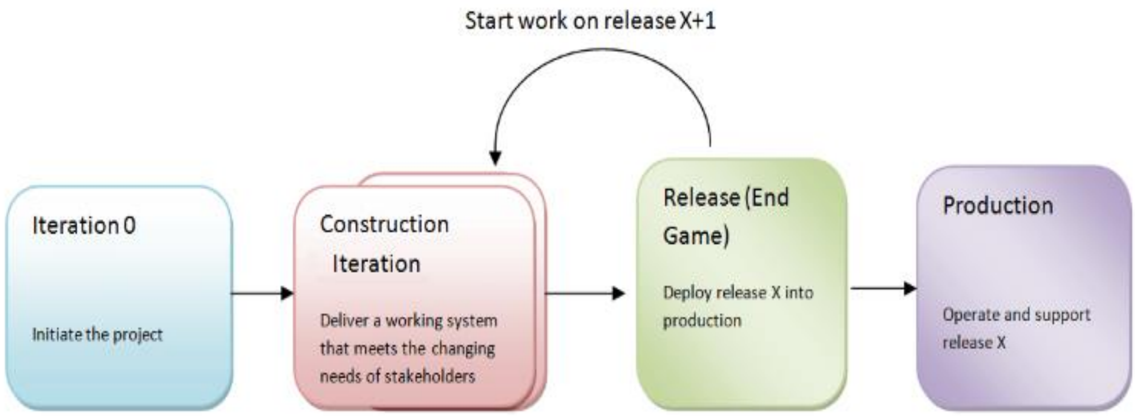


Figure 2.3: Agile testing stages (Isaacs, 2016)

Figure 2.4 provides a better understanding of how agile testing techniques are used over time. Interest in Test-Driven Development (TDD) continues to increase, highlighting its central role in agile testing; however, another mode of agile testing represents “distinct spikes”. For instance, there are gaps in performance, publication database record, acceptance and Graphical User Interface (GUI) testing, illustrating the subfields with respect to distinct agile testing. For example, in agile development, GUI testing was not included from 2006 to 2009, yet interest was shown in database testing at that time (Hellmann et al., 2012).

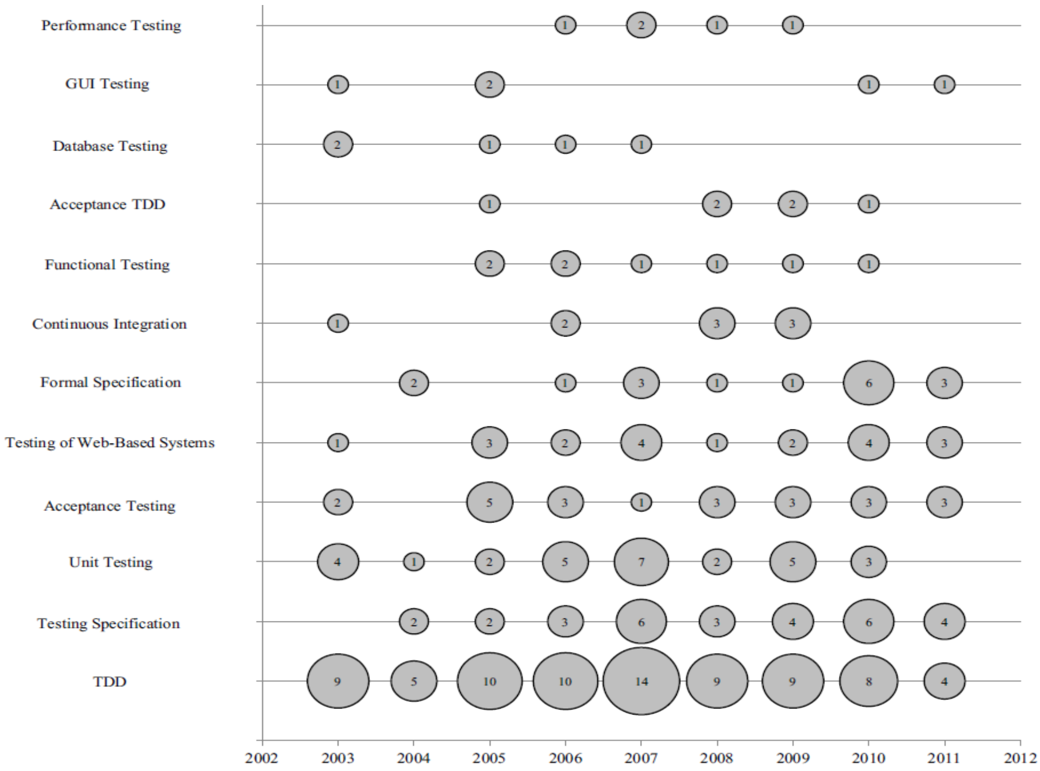


Figure 2.4: Testing use over time (Hellmann et al., 2012)

2.5. Agile information technology (IT) security implementation

In all software products, there are potential vulnerabilities that can cause considerable damage. Therefore, software developers are required to develop more efficient and secure systems by conducting every phase of the software development life cycle (SDLC) (Rindell, Hyrynsalmi and Leppänen, 2017). At each stage, developers must consider and incorporate all the security aspects to ensure that there is no vulnerability. Security is considered the most important component when developing any software product (Siponen, Baskerville and Kuivalainen, 2005). Agile IT security goes beyond safeguarding the software, also focusing on its efficiency in performing tasks. This is facilitated by involving spikes, for instance to enhance the testing process to abolish uncertainty and inconsistencies, boosting the software lifecycle and its ability to function efficiently, thus it resolves security issues that may arise from unresolved complexity (Rindell, Hyrynsalmi and Leppänen, 2017). Furthermore, the involvement of spikes covers issues arising from software developers' lack of knowledge about the future implications of a particular feature. This lack of knowledge can leave developers unaware of which areas to address to secure the software, and the use of spikes may cover this uncertainty, enhancing agile IT security (Nemati, 2007).

2.6. Agile Spikes in Different Development Domains

Software teams use agile spikes to investigate, close gaps and reduce risk. Spikes should be estimated during the sprint/iteration process in the form of sprint/iteration tasks. The duration of each task should be used to develop and research the deliverables in the state of workflow or working items of software or documentation (Leffingwell, 2010). Before execution, the objectives and duration of spikes in agile should be agreed between the development team and the PO. Defining the spike in terms of end-user functionality is essential, especially when attempting to accomplish the requirement for which the spike is being performed. In addition, some acceptance criteria should be defined to determine and identify the outcomes and results of the spike. A spike is a good solution for these situations because it provides a proper estimate, including the sprint backlog (Rindell, Hyrynsalmi and Leppänen, 2017). More details about spikes and prototyping are provided in section 1.1.4.

In this section, we review the most significant domains identified through the literature review during the exploratory phase of this study.

2.6.1. Agile development for big data

In agile development, spikes can play a major role in data development as it has guidelines to make sure that user stories and data are quantifiable, demonstrable and acceptable. Big data represents parallel processing and data distribution to make sure that data analytics, algorithms and storage lifecycles are not separated from big data technologies. For this reason, architecture-supported agile spikes were introduced to ensure that rapid technological changes and new requirements are addressed effectively (Grady, Payne and Parker, 2017).

Big data systems are difficult to maintain and handle; however, the practices and principles collected under the agile umbrella focus on validating the assumptions in the lifecycle delivery. Effective implementation of the agile approach reduces the risk exposure after a project is initiated. As per the methodology proposed by Chen, Kazman and Haziyev (2016), Architecture-centric Agile Big Data Analytics (AABA) is used primarily for addressing the rapid, organisational, and technical technology change challenges of both the agile delivery of big data analytics for web-based systems and big data system development. Chen et al. (2016) claim that agile big data analytics development is greatly aided by the use of design concept catalogues and architectural spikes, both of which are examples of improvements to architecture design methods (Chen, Kazman and Haziyev, 2016).

2.6.2. Agile development for data warehousing

Traditional data warehouse projects follow the waterfall development model, where they complete all six phases: gathering requirements; designing; developing; testing; deploying; stabilising. In this model, both the technology and business requirements are complex and critical by nature, and it takes approximately six to nine months to ensure full implementation (Rahman, Rutz and Akhter, 2013). After advances in technology or any changes to the requirements, it becomes difficult and challenging for the software team to incorporate these updates without altering the fundamental architecture of the system, resulting in frustrated development teams and disappointed stakeholders.

By contrast, agile development integrates the solution in an iterative fashion, which is why it is known as the '60% solution' (Rahman, Rutz and Akhter, 2013). The approach helps to deliver the user requirements in the initial release, with refinements in the subsequent regular scheduled release series. To accommodate this task, the agile data warehousing approach increases successful implementation within the specified budget and on time. For data warehouses, incorporating spikes allows teams to hone their skills and comprehension to gain traction on a problematic part of the project (Hughes, 2012). In some cases, the parameters required for an essential business rule in the data transformation modules are not always provided by the source system, even when using a standard ETL (Extract, transform, and load) tool or the DWBI (Data warehousing/business intelligence) team discovers that it will need to resort to a service-oriented piece of middleware that none of them has worked with before. In this case, the developers need to pause the current construction sprints in order to use spikes to address this issue. Hughes (2012) claims that the only part of the warehouse architecture required to solve the problem with the new technology or data is the layer responsible for transporting a small sample of data between the various application layers (Hughes, 2012).

2.6.3. Agile development for computer science education

The primary goals of computer science education are to assist students in developing meaningful knowledge and relevant skills (Bergin et al., 2005). Computer science education can benefit from incorporating agile development techniques, similar to how software project teams can use agile spikes to reduce risk, close gaps, and investigation in software development (Woodward et al., 2013). In the educational setting, students commonly try spikes in the order presented, especially if they lack a thorough understanding of the domain. Students can learn from the industry's best practices for task allocation and source code management. Even for an independent student working with spikes, it is beneficial to record and store the execution of a spike plan, as well as its preliminary description and final outcomes. The outcomes should be in a format that would bridge the same gap for another student with a similar background when placed in a specific educational context (Woodward et al., 2013). The authors claim that spikes are compatible with a wide range of educational objectives. They used spikes in a variety of curriculum approaches, including some cases where spikes were used in place of traditional assignments. To back up this claim, certain criteria and checklists were developed for assessment purposes. Furthermore, Woodward et al. (2013) suggest that spikes are unsuited to first-year students because, at this early stage, the overheads of compound skills (e.g., the software development approaches) involved in using spikes overshadow the potential benefits.

2.6.4. Agile development for blockchain

Blockchain was initially developed to ensure the decentralised and secure operation of the Bitcoin cryptocurrency. As a result, programmers quickly realised that the Blockchain can also be used as a decentralised computer running Smart Contracts, which are essentially pre-written computer programmes that can be used as the basis for automating the enforcement of contracts (Marchesi, M., Marchesi, L., and Tonelli, R., 2018).

Agile methodology is integrated into adaptive planning because it provides potential support for continuous improvement, allowing the lifecycle to respond to changes easily and quickly. These procedures rest on key principles organised into phases. Together, the decentralised technologies and blockchain approach allow for new possibilities, offering users value from their digital and software products. Agile development involves integration and transition, whereas blockchain offers a wide range of system design options. This creates a platform of uncertainty when a company incorporates new technology; however, the involvement of spikes may mitigate that level of insecurity (Lenarduzzi et al., 2018). Nonetheless, the supporting evidence in this paper is weak (see Table 2.1), as there is only one case study.

Researchers in the blockchain have adopted agile methodologies for developing Blockchain Based Software (BBS). Fahmideh et al. (2021) give various purposes to support their claim that spike in agile has been found to be a valuable technique at the preliminary phases of BBS development, such as (1) requirements gathering, specification, and so far from intended blockchain platform, (2) establishing a base target BBS architecture independent of reliance to a specific platform, and (3) determining uncertainties in system quality aspects like security, transaction execution performance and the trade-off between them (Fahmideh et al., 2021). Although the studies cited by Fahmideh et al. to support the significance of spike were included in their Systematic Literature Review, the importance was supported indirectly through those papers.

2.6.5. Agile user experience (UX) design

The Agile Manifesto neither includes UX professionals nor addresses the resources, research and time required by them to create excellent design. Agile and UX methods coexist well only if the management of the organisation supports and understands UX work; the UX practitioners spend time and show leadership in reaching out to their colleagues; the agile workflows have the flexibility to accommodate UX needs; and the product teams are composed of UX professionals, so they can build rapport and respect with the developers (Da Silva et al., 2012).

Spikes in agile can be used to incorporate UX design work or to monitor user research, but their primary purpose is to manage risk in implementation solutions. Normal spike planning should easily predict and anticipate design and research events, and spikes should be responsible for managing risk problems, such as a design task that requires an inquiry into available technology before it can be estimated, or any uncertainty that arises and necessitates user research (Brown, 2012).

2.6.6. Agile development in cloud computing

Agile software development encounters several challenges, including face-to-face communication, scalability, streamlined development control, transparency, management of resources, and the capacity to develop applications from different locations. To deal with such challenges, cloud computing offers a platform for quickly testing new ideas in the market while also lowering the cost of agile development through data sharing, task prioritisation, distributed applications, and infrastructure provision (hardware and software). Cloud computing speeds up development by removing the need for setup process, software patches, and re-installation. Moreover, it can enhance the agile process by allowing for faster delivery, and elevating software quality (Younas et al., 2018).

Integrating agile spikes into cloud computing may resolve problems of quantifying risks and uncertainties over timing. Combining the agile approach with cloud computing eliminates a risky infrastructure investment plan. In addition, the risk elements associated with unapproved prototypes are far fewer than in traditional implementation using the on-premises model, in which significant investment is made to access the prototype (Kalem, Donko and Boskovic, 2013). The advantages of using cloud computing for agile development can be observed throughout the various steps of the agile development process. These enhancements influence multiple aspects of agile development, allowing for a quicker and higher-quality application development process. Figure 2.5 depicts the overall project development cycle in the agile development framework. Timelines for each step of agile development are displayed in this Figure. The length of time spent in each phase shown in Figure 2.5 is affected by the benefits stated above in this section (Kalem, Donko and Boskovic, 2013).

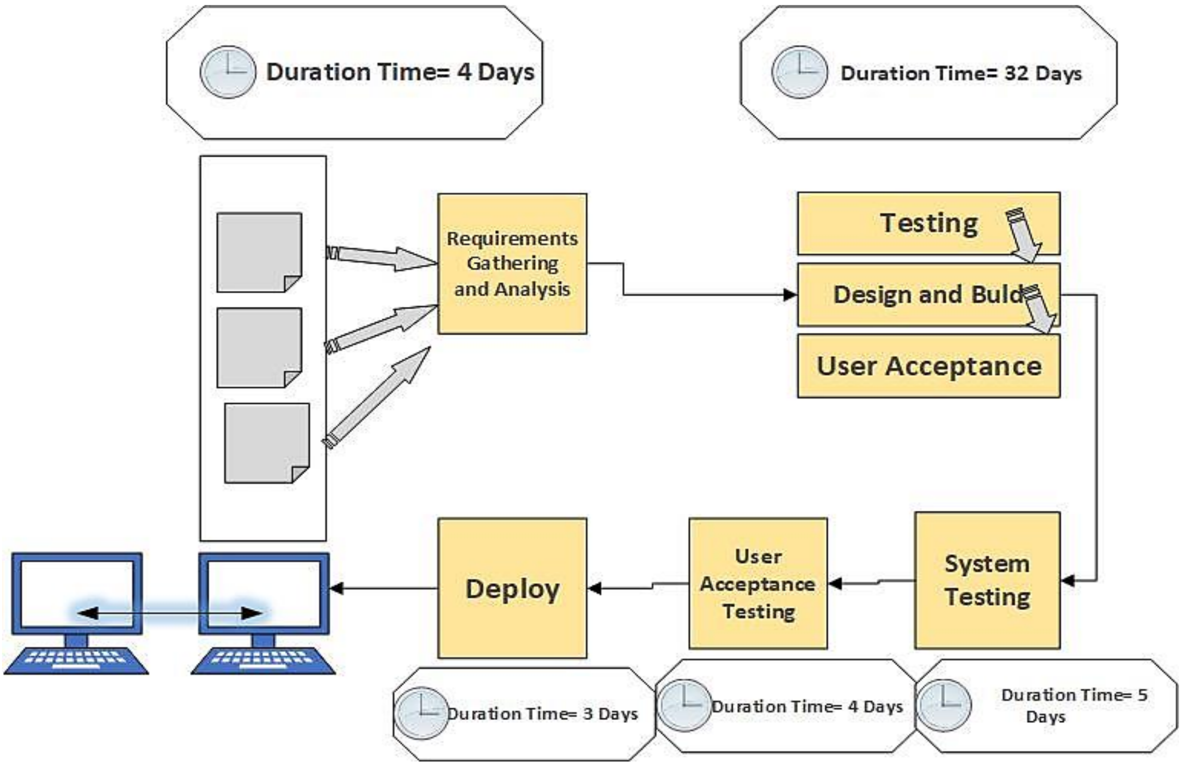


Figure 2.5: Development time in days combining agile methods and cloud computing (Kalem et al., 2013)

2.6.7. Agile development in the Internet of Things (IoT)

Integrating the IoT approach into current software development, manufacturing, and engineering processes requires a value-driven and streamlined framework. The relevance of IoT to agile development rests on three main areas: responsiveness to the requirements that are changing continuously; cycle management; and frequent updates. As more devices are connected to the Internet, cycle management for both hardware and software is becoming increasingly relevant and requires an iterative or incremental approach to ensure project delivery (Cheng et al., 2018).

In IoT devices, customer responsiveness is considered the main factor, yet with fast-growing technology, responding to customer responsiveness has become the end solution for all organisations. Agile methods have the ability to accommodate changing requirements. The effectiveness of the agile framework must be linked to the value stream of the agile process (Cheng et al., 2018). Additionally, for continuous improvement and sustainability of practise, regulations, competition, volatility, uncertainty, complexity, and ambiguity must be addressed (Akem, 2018). It is also important to factor in the usage of spikes in agile development as a way to facilitate the integration of IoT during software development. Although agile development aims to deliver a high-quality product rapidly, the involvement of spikes may represent a good opportunity to reduce the risks to IoT in this environment.

2.7. Research Gap

Although the use of spikes in agile software development was defined in XP more than 20 years ago, the literature has revealed that little is known about the use of spikes in ASD. As a result, researchers and some developers are struggling to comprehend the spike concept and its applications, as well as what sets it apart from other prototyping concepts that aim to improve the quality of software before it is delivered to the end user (Layton, M. 2019).

As this literature review has shown, spikes are applicable in various domains. However, there is still a dearth of information regarding their exact roles and their effectiveness or even efficiency. Furthermore, the precise reasons for spike applications in agile have not been highlighted, studied, or empirically evaluated. The common spikes success factors had not been identified as well. It was also unclear what factors contributed to the success of the spike applications. Thus, the main aim of the study is to bridge the gap in the literature by conducting in-depth analysis and investigation of spikes in ASD. Triangulation will be used to obtain reliable results by completing three steps: a literature review, the agile practitioners review, and a case study, as shown in the subsequent chapter.

2.8. Summary

This chapter provided a review of the literature on agile spikes and related domains found during the exploratory phase of this study. However, the use of spikes should not be limited to these domains as illustrated in Chapter 5.

Table 2.1 studied agile development in various domains and its contribution to developing working software that satisfies the end-user. Among the 16 papers listed in this table, only five of these articles provide empirical data supporting their findings. It should be noted that only one article focuses solely on spikes, whereas the others include spikes as one of several techniques covered. The table showed the limited information on spike usage by listing the various software domains identified based on the application of spike found in the literature review. The use of spikes plays a critical role in ASD by minimising unforeseen risks and uncertainties in the development cycle by getting the agile team to understand the risks involved and find a viable solution. Based on the review conducted in this study, there seems to be less concentration on the issue of spikes application in agile software development. Furthermore, the review showed that the spike had not been surveyed previously.

Big data, data warehousing, computer science education, blockchain, security, cloud computing, UX design, and IoT are among the domains covered in this review. It is unclear what roles the spike appears to serve in each of these contexts. Besides, the concept of spikes application in agile software development seems to be underexplored based on the research. Thus, the literature review elicits a potential area of focus for studies seeking to deepen the understanding of spike application. On the other hand, the potential confusion in spike implementation among the development teams still creates concerns for many development teams, as Nicolette (2014) and Layton (2019) indicated earlier in sections 1.2 and 2.7.

There are clearly challenges in gathering relevant information as there is insufficient information concerning spikes application in agile. The studies examined lacked information that could significantly support the idea that spikes are useful in ASD projects and different domains. As a result, the information gathered cannot expressly confirm the quality and efficacy of spikes in the software produced. Therefore, more research and experiments are required to determine the nature and effectiveness of spikes in agile software development. Although the potential for implementing spikes is undeniable, it remains unclear at what point they are necessary and for what purposes they can be used, beyond risk management and the little-known roles of spikes in investigation and research. In addition, how to make the most of spikes by considering the factors that affect their implementation. Nonetheless, reaching this milestone requires a solid research approach/methodology that is specific and all-inclusive of various aspects of spikes application in ASD.

Chapter 3: Research Methodology

Research methodology is the systematic plan for conducting research. This chapter mainly focuses on how the research objectives identified in chapter 1 will be achieved.

The proper selection of research methodology contributes to the success and overall quality of the research study and its documentation. Furthermore, becoming acquainted with the research methods employed by a field of study allows one to comprehend it more effectively (Singh, 2006). The research methodology design in this study was inspired by Saunders's (2016) framework to provide a clear and extensive plan to answer the research questions with considering Kitchenham et al. (2002) guidelines in conducting empirical research in software engineering. Figure 3.1 depicts the methodology framework adapted from the Saunders et al. (2016) model, which consists of multiple layers that can be used from the outer to the inner layers.

Figure 3.1: Research methodology layers assigned to this study (adapted from Saunders et al., 2016)

The research objectives and questions are crucial in determining the type of research methodology and design. According to Cohen et al. (2007), "the purposes of the research determine the methodology and design of the research". This research will employ an exploratory research method for the research questions. Exploratory research is used to investigate a problem that is not clearly defined. It is carried out to gain a better understanding of the current situation. However, this kind of research will not provide conclusive results (Dudovskiy, 2018). The exploratory method is significant to the study of the research questions, examining the roles of spikes and how they can be employed to manage risk efficiently. Furthermore, to explore the common success factors that can enhance the spike outcomes. Figure 3.2 illustrate the research process and investigation stages with strategies used to answer the research questions (RQ1, RQ2, RQ3, and RQ4) and validate the study findings.

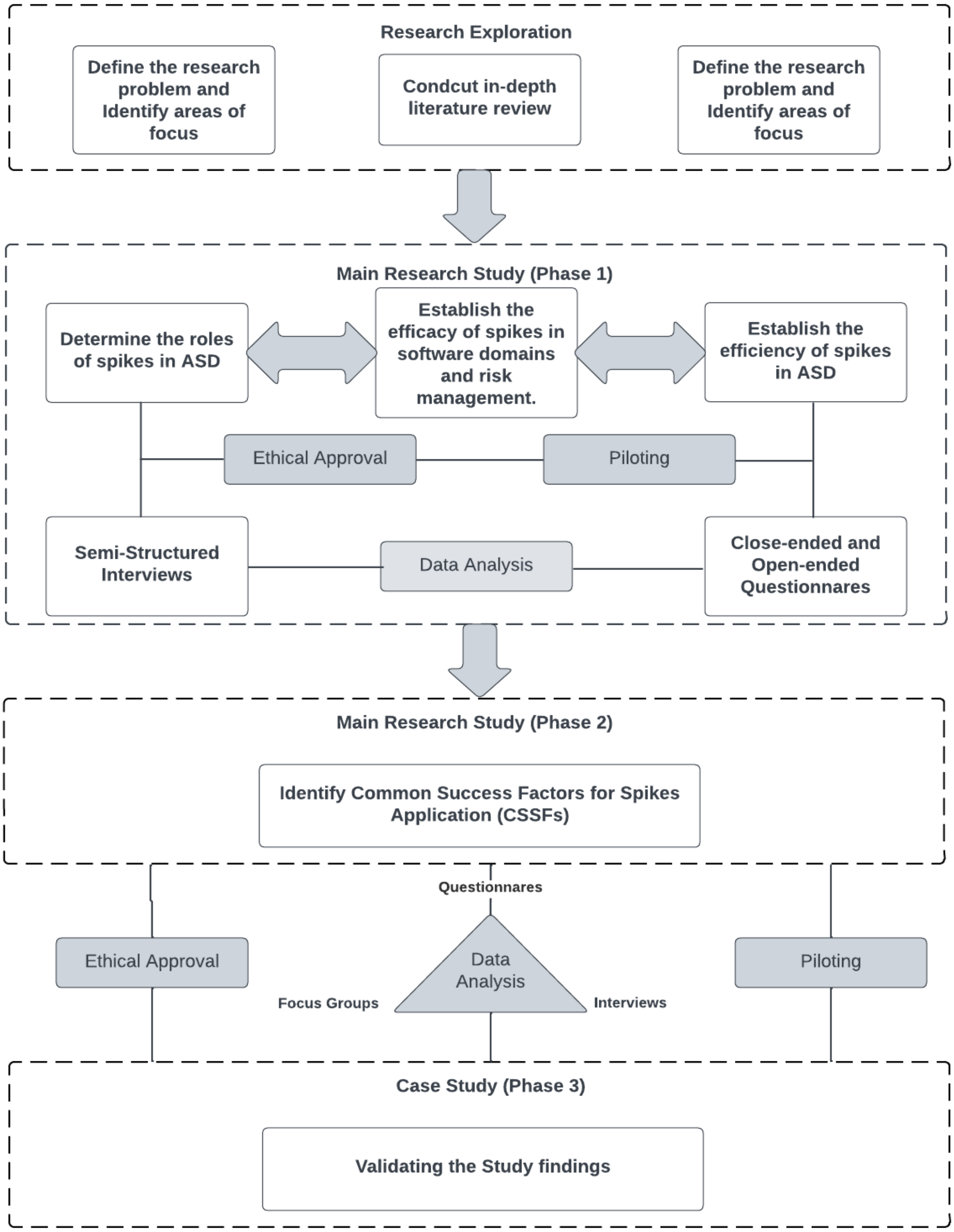


Figure 3.2: Overview of the research process and stages

3.1. Research Philosophy

A research philosophy is a set of beliefs about the essence of the reality under investigation. It is the fundamental definition of what knowledge is. A research philosophy's assumptions provide rationale for how the research will be conducted (Saunders, Lewis and Thornhill, 2016). Different research philosophies may have various ideas about the research purpose and the most effective means of achieving that purpose. The type of knowledge being explored in a research project ultimately determines which research philosophy to employ. Thus, knowing the research philosophy can shed light on the assumptions made during the research process and how they relate to the methodology employed (Becker, Bryman, and Ferguson, 2012). The four primary trends of thought within research philosophy are positivism, interpretivism, pragmatism, and realism. The pragmatic philosophy used in this study since it is congruent with the strategies that can be used to achieve the research objectives.

Pragmatism is a philosophical movement emphasising the practical application of ideas by engaging in thought experiments, deriving principles from actual human experiences, and exploring the consequences of various theories. It seeks to balance objectivism and subjectivism, facts and values, precise and rigorous knowledge, and various contextualised experiences. Pragmatism holds that the truth of any idea or proposition is determined by its practical consequences rather than its logical structure and that the purpose of thought is to guide action. It is often considered the opposite of rationalism and traditional philosophy, which focuses on truth and the abstract principles of reality (Saunders, Lewis and Thornhill, 2016).

3.2. Research Approach

The research procedure includes several detailed steps to collect the data and then undertake analysis and interpretation. Based on the nature of the problem addressed in the research, the approach is usually divided into two:

- Data collection
- Data analysis or reasoning

The philosophy chosen will determine the approach used for the development of the theory or the reasoning behind the research findings. Furthermore, the approach chosen will have an impact on the research design and methods used. Research approaches can also be determined by the researchers' prior knowledge and expertise, the study's target demographics, and the nature of the problem or issue being investigated (Babbie 2021).

There are three main approaches to conducting research, as described by Saunders et al. (2016) Inductive, Deductive, and Abductive approach. The study uses both inductive (qualitative) and deductive (quantitative) approaches with a pragmatic philosophy that seeks to incorporate only what works best in finding the answers to the research questions. According to Mitchell, mixed methods draw out the strengths and minimise the weaknesses associated with the traditional, single approach. The approach usually encompasses many complementary aspects; thus, its outcomes are conclusive. The inductive approach typically starts with specific observations and proceeds to theories and broader generalisations, whereas the deductive approach works from the more general aspects to the more specific ones (Mitchell, 2018). Figure 3.3 illustrate the sequential process of inductive and deductive approaches.

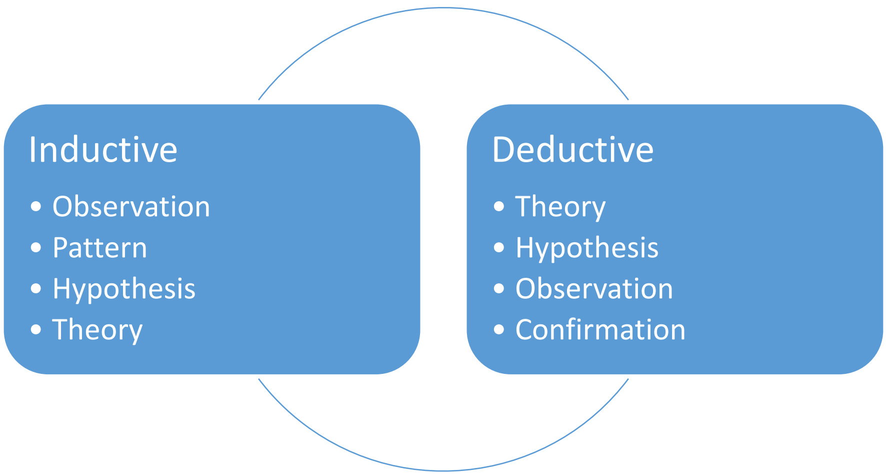


Figure 3.3: Inductive vs. deductive approaches

In this research, the deductive approach was used to test the hypotheses and concepts concerning the roles of spikes in agile software development and their effectiveness. On the other hand, the inductive approach begins by observing the collected data and then formulating a theory based on the results of data analysis.

3.3. Research Strategies and Data Collection Methods

A research strategy is a plan by which to answer research questions. This step-by-step procedure provides a path for the researcher’s thoughts, and it helps in conducting the research and producing quality documents and detailed results. The choice of research strategy rests on how a researcher wants to achieve a reasonable level of consistency throughout the process. The research method undertaken in this study are mixed method (qualitative and quantitative). Mitchell (2018) states that mixed methods draw out the strengths and minimise the weaknesses associated with the traditional, single approach. The approach usually encompasses many complementary aspects; thus, its outcomes are conclusive. In this case, we want to review the basic concepts of implementation in agile methodologies by considering how agile spikes are used in ASD. This can be done by employing mixed-methods research, a combination of qualitative and quantitative data collection techniques that can be used either concurrently (i.e., at the same time) or sequentially (i.e. after the other data collection technique has been applied) (Saunders, Lewis and Thornhill, 2016), as shown in Figure 3.4

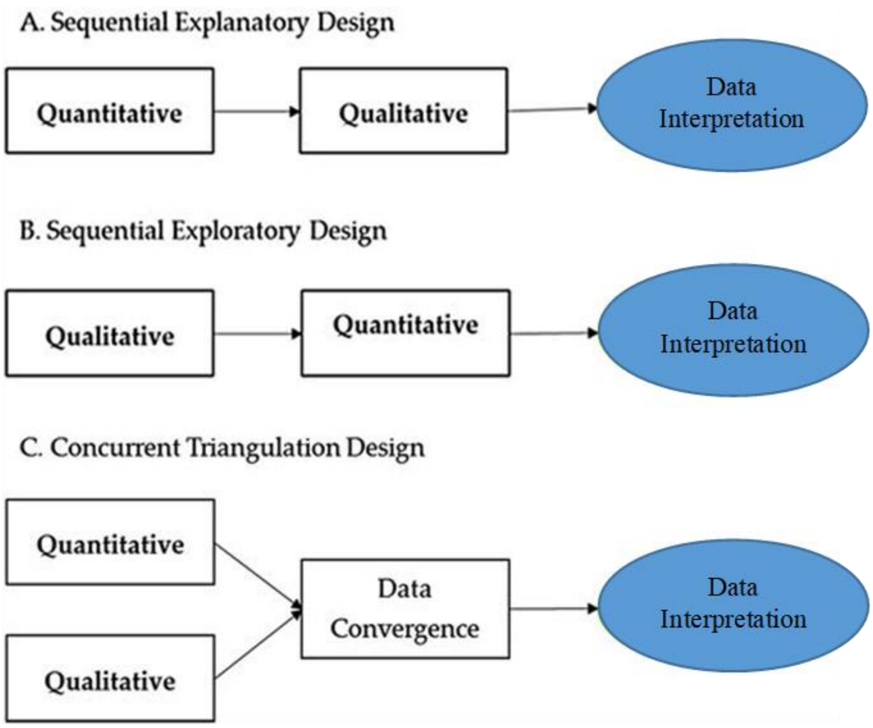


Figure 3.4: Research designs for mixed methods (Warfa, 2016).

Concurrent mixed-method research involves using both quantitative and qualitative techniques in data collection and analysis, known as empirical evidence gathered. This data can be collected from primary sources such as surveys, observations, and experimentation conducted by the researcher. In this study, the concurrent research design was employed to address the four research questions, whereas the qualitative method was used for the case studies that validated the findings of the research questions, as shown in chapter 6.

Saunders et al. (2016) stated that the exploratory research questions are likely to begin with 'What' or 'How.' Questions about an issue, problem, or phenomenon that should be asked during data collection are likely to start similarly. The same approach was followed to investigate the basic concept of agile spikes concerning their implementation by considering how they are used in ASD. Mixed methods are more interactive than other approaches, as each phase of data collection and analysis directs the next (Saunders, Lewis and Thornhill, 2016).

The research approach was focused on interviews, online surveys, and observations from a pilot study, as discussed in chapters 4 and 6.

Open-ended interviews were used to gather in-depth information from respondents, collecting their actual thoughts. Observations were used to ensure that we gathered information on how agile spikes work in a specific setting. The researcher adopted random sampling, as Kitchenham et al. (2002) pointed out in their preliminary guidelines to minimise the biased allocation. The participants' selection was based on inclusion and exclusion criteria, as illustrated in chapter 4 and 6.

The qualitative approach was employed to deduce the information to answer the research questions and meet the objectives of this study. This method was access and collect vital data and information from interviews with individuals and focus groups, observations during the case studies stage.

The quantitative method of research was used effectively to find statistical and numerical findings on how spikes function and deliver results. Apart from analysis of existing knowledge, the quantitative method was of great importance to the collection of new numerical data from online questionnaires.

From the interviews, the researcher deduced how agile spikes have been significant to the respondents and obtained their views on their efficiency. To achieve more concrete findings on the efficacy of agile spikes, the researcher conducted an online survey to gain further opinions from various agile teams on their experiences with agile spikes. These opinions concern how agile spikes can be used effectively, the roles of spikes in different agile methods, and the factors that help to complete spikes successfully.

The researcher used case studies, multiple focus group discussions, and observations with various agile teams working in different companies around the world to further consolidate the study findings (see chapter 8). This was focused on how various agile teams use agile spikes. The focus was on new technology, system architecture and related fields. The observation was also employed in the process of studying and analysing how agile teams can use spikes to achieve minimal risk effectively. The observation method provides the researcher with first-hand experience, so the data collected during the observation can be analysed and deduced to be entirely sound, concrete findings.

In such a study, triangulation is required to develop a more comprehensive understanding of the application of spikes in ASD (Carter et al., 2014). For this reason, this study used methodological triangulation to collect as much information as possible on the application of spikes in ASD. These methods included reviewing the literature, and conducting interviews, focus groups, and questionnaires, as shown in Figure 3.5. Combining these methods effectively collected sufficient data to answer the research questions. Furthermore, this research used data source triangulation by varying the participants among whom the interviews and questionnaires were administered. For instance, interviews were conducted with individual participants and groups/teams. Similarly, the various questionnaires used to collect the data were distributed to people from different countries. This triangulation was necessary for the data to have multiple perspectives and provide some sort of validation, especially when the findings from various sources might be similar and consistent (Carter et al., 2014).

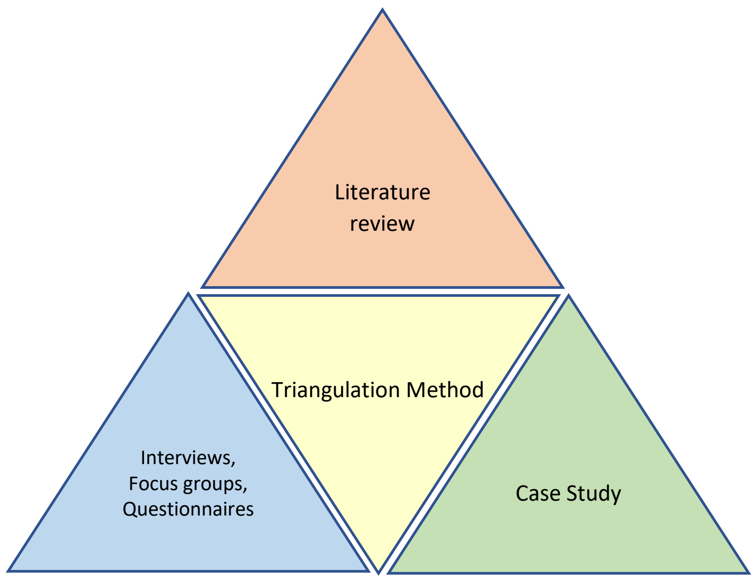


Figure 3.5: Triangulation approaches used in the study

3.4. Time Horizon

To plan a research time horizon, two research methodology designs can be employed. These are cross-sectional and longitudinal research, respectively. Cross-sectional research collects study data at a single point in time, whereas longitudinal research collects study data over time (Saunders, Lewis and Thornhill, 2016). Therefore, this study utilised cross-sectional since it aimed to investigate the roles of spikes, their efficacy, and factors that aid in effective implementation through interviews and surveys.

To help understand the various research methods and avoid confusion, Saunders et al. (2016) summarise the significant approaches, designs, and strategies and their relationships to one another in the following table (Table 3.1).

Table 3.1: Summary of the research methods (adapted from Saunders et al., 2016)

Research Philosophy	Research Approach	Research Design	Type of Study
Positivism	Deductive	Quantitative	Survey, Experiment, mono-method quantitative study
Interpretivism	Inductive	Qualitative	Case studies, action research, ethnographic, phenomenological
Pragmatism	Inductive & Deductive	Mixed	Explanatory, Exploratory, Descriptive

3.7. Summary

This chapter presented the research methodology used in this study based on research questions that need to be investigated. A plan of research methods, approaches, and strategies has been developed in order to address the gaps identified in the previous chapter effectively. This study employed concurrent mixed methods including both qualitative and quantitative, to gain a comprehensive understanding of the spike's usage in ASD. The nature of the research questions that need to be investigated identifies the study design as illustrated in this chapter. Therefore, an exploratory approach was employed to answer the research questions. The research approaches were focused on interviews, an online survey (questionnaires) and observations to achieve the aims and objectives of this study. The interviews and surveys were piloted to assess the feasibility of the study before launching it to participants, as shown in chapters 4 and 6. The investigation framework consists of three stages besides the literature review. The first stage was concerning the first three research questions to establish the roles of spikes, efficiency, and effectiveness. The second stage was focused on RQ4 to identify the Common Spikes Success Factors (CSSFs). Finally, the third stage of this study comprised case studies were undertaken to validate the findings of the previous two studies, as illustrated in Figure 3.2.

Chapter 4: Preparation for Data Collection Concerning the Roles, Efficiency, and Efficacy of Spikes

The previous chapter described the methods and approaches adopted by this study. In this chapter, the dynamics of the process followed before data collection is discussed in detail to address RQ1, RQ2, and RQ3. It has four main sections, and the first two have three sub-sections each. Section 4.1 discusses the interview method used in this study. It covers the design of the questions, the interview sample, and the piloting to test the instrument for data collection. Section 4.2 describes the questionnaire method used in the study. In particular, it covers the design of the questionnaire, the estimation of sample size and how the questionnaire method of piloting was used. Section 4.3 describes the ethics approval that the research underwent prior to the collection of data. Lastly, section 4.4 discusses the data analysis instruments used in this research study.

4.1. Interview Method

In this study, interviews were used to answer the research questions that needed complex explanations that the questionnaire could not accommodate. Importantly, the qualitative information needed in this research was obtained through interviews with practitioners in ASD who have had the chance to witness or use agile spikes in software development projects. In most cases, to collect qualitative data that need follow-up questions to obtain the information from respondents, interviews are the preferred method (Hennink, Hutter and Bailey, 2010). In this study, semi-structured interviews were used. A total of 14 questions, excluding demographic questions, were used to collect more information on how agile spikes are used in ASD, as shown in appendix D. The 16 questions were subdivided into three, with each section covering a common theme: the roles of spikes; their efficiency in ASD; and their effectiveness in managing risks of ASD. This separation was to enable a more focused and in-depth interview process (Hennink, Hutter and Bailey, 2010). Furthermore, some of the questions are interrelated, hence the need for a systematic approach to asking the questions.

4.1.1. Designing the interview questions

An interview approach is a ‘one-chance-only’ approach to collecting data. In the event that all the information needed is not obtained, it is difficult for a researcher to go back or reschedule the session to get it. To ensure that the questions are focused and precise, the researcher first identifies the key areas or questions that directly answer the study's research questions. In one-to-one/face-to-face interviews, the interviewer has a chance to seek clarification on the answers provided (Ryan, Coughlan and Cronin, 2009). In this study, a data requirement table (Table 4.1) derived from Saunders's book (2016) was used to guide how the questions in the interview were designed and their focus to ensure the three research questions were answered comprehensively. This is not always a standard approach when preparing questionnaires. However, it was decided to use such a table to guide the structuring of the interview questions with a view to gaining as much information as possible through the investigative questions outlined and thus shed light on the research questions (Saunders, Lewis, and Thornhill, 2016). The 'variable required' column addressed the types of variables that could comprehensively answer the investigative questions. For instance, investigative questions concerning 'The roles that spikes play in agile software development' could be covered by two types of variables: behavioural (how the practitioners use spikes) and opinion (what they perceive spikes are used for in most cases). Table 4.1 thus indicates the key areas that the interviews needed to cover.

Table 4.1: Interview data requirements

Research Questions: What are the primary roles of spikes in agile software development? What are the practitioners' opinions on the efficiency of spikes when used? What are practitioners' opinions on effectiveness of spikes to manage risk?		
Type of Research: Exploratory research across different domains where spikes are applied during agile software development projects		
Investigative Question	**Variable Required**	**Details in which data is measured**
The roles that spike play in agile software development	Behavioural and opinion variables: What the agile team does when they use the spikes in agile software development.	List of agile methods used, and roles that spikes play, e.g., estimation and system exploration in scrum.
The efficiency of spikes when applied in ASD	Opinion variable: The accounts of practitioners on how efficiency they have seen spikes be in their software development projects.	• efficiency score, and rationale for the scores • Feelings like, very efficient or not.
The effectiveness of spikes in risk management	Behavioural and opinion variables: The scores of effectiveness, account of how effective spikes are in solving uncertainties in ASD.	• Effectiveness scores and their rationales. • Feelings like, very effective or not.
Years of experience in both spikes and agile software development	Attribute	The practitioners are allowed to enter their years of experience.
Agile roles	Attribute	Agile practitioners state their different roles in agile teams
Agile methods	Attribute	Allow agile practitioners to state the methodologies that they use

Based on these requirements, a list of 14 questions (including demographics) was created to guide the interview procedure and satisfactorily answer the research questions. Multiple questions were devised to satisfy each requirement properly (investigative question).

4.1.2. Interview sample size

It is practically impossible to interview the entire targeted population in some studies and, in this case, sampling becomes appropriate. In research methods, there are two types of sampling: probabilistic and non-probabilistic. The latter refers to sampling techniques in which not all individuals in the population have an equal chance of being included in the study sample, while the former refers to random sampling criteria under which everyone has an equal chance of inclusion (Adams and Lawrence, 2019). This study used a non-probabilistic sampling technique known as convenience sampling. This approach only included those practitioners identified through colleagues and social networks (LinkedIn and Slack) who agreed to participate in the interview, whether face-to-face or virtually. Through this approach, the researcher obtained a sample of 22 participants scattered across many world countries. The sample size is between the recommended limits when an interview approach is used, namely five to 25 participants (Dudovskiy, 2018). When a random sample has been obtained, the selection method should be defined, as Kitchenham et al. (2002) pointed out. Therefore, the inclusion criteria have been identified to ensure that the findings of the study are valuable and can be generalised to the population of interest as follows:

- The practitioner must be 18 years old and above.
- The practitioner should have experience in agile software development.
- The practitioner has used the spikes technique before.
- The practitioner should hold at least one agile role.

4.1.3. Interview piloting

Testing data collection instruments for suitability and appropriateness is crucial in any study. It provides more information on comprehensibility, language appropriateness and any potential ambiguity in the questions (Hennink, Hutter and Bailey, 2010). Furthermore, it allows the researcher to estimate the time required to complete a single interview without hurrying interviewees and practice how best to present the questions. To achieve these milestones, the questions used in the data collection instruments were validated by reviewing the questions with two practitioners in ASD. The process also involved piloting the questions with five people (four PhD students and one agile practitioner) to ascertain whether they were reliable in collecting the needed information. Lastly, the questions were reviewed by the ethics committee at the University of Southampton before being applied in the data collection process.

4.2. Questionnaire Method

As mentioned earlier, the study used a questionnaire to collect most of the quantitative data answering some research questions. The questionnaire approach involved practitioners with experience in both spikes and ASD. As part of the data collection instrument, the questionnaire has 30 questions, mostly comprising close-ended questions but with a few open-ended questions, as shown in appendix G. The criteria for inclusion in the questionnaire considered the convenience and willingness of the invited participants.

4.2.1. Designing the questionnaire

Questionnaire design largely depends on the objectives or the research questions to be addressed. The questions need to be directly and collectively to address the research goals adequately. Without a proper design, a questionnaire may collect redundant or irrelevant information, which may change the direction of the study or impede the fulfilment of the objectives altogether (Boynton and Greenhalgh, 2004). Although there are instances where standardised questionnaires are employed, unless it is a replication every study requires an entirely new questionnaire.

In this study, the questionnaire was based on the focus of the research. Thirty questions were formulated: demographic details; the efficiency of spikes in ASD; and the efficacy of spikes in risk management. Each of these sections, mainly the last two, contained a mix of closed- and open-ended questions. Together, they answered RQ1, RQ2, and RQ3. Thirteen required a Likert-scale response, being statements that prompted the participants to either agree or disagree on a five-point scale.

4.2.2. Questionnaire sample size

As with interviews, it can be challenging to survey the entire population of a study by questionnaires. Apart from the cost, it can be a time-consuming process. Conventionally, a sample is selected, whether through a probability or non-probability sampling approach, to refer to the entire population. However, if the findings are to be generalised, the internal and external validity of the study needs to be high (Adams and Lawrence, 2019). Estimates of the necessary sample size should be made to establish the minimum number of participants to achieve a specific power for the results in the tests being run (Hennink, Hutter and Bailey, 2010).

Since the population of practitioners in ASD is not known, it was not possible to use conventional formulae for calculating the sample size to be studied based on the population size. In such a case, the size has to be based on other parameters, such as the confidence level and an attribute's estimated proportion (Cochran, 1963). Thus, it was necessary to use the G*Power tool to approximate the appropriate sample size for the study, given that the conventional effect size and confidence level were known. This tool takes into consideration the tests that will be performed on the data collected (in this case the t-test). It also considers the confidence level (which in this study was the conventional 95%, or simply stated as an alpha level of 0.05). Finally, it takes into account the expected effect size, in most cases the strength of relationships between variables (always between 0 and 1). This effect size is not related to the number of questions but the expected correlation coefficient between variables of interest (in this case efficiency and effectiveness). Since the relationship was expected to be moderate, the average value of the possible effect size was used, i.e., 0.5. Based on these items and the type of test that would be undertaken on the data, it was possible to estimate the minimal sample size using the G*Power tool.

According to Cohen (Mayr et al., 2007), sample size estimation using G*Power depends on the effect size that is established; however, the estimate used a hypothesised effect size of 0.5 to obtain a sample size of 54 participants. Due to the need to clean the data and eliminate outliers, a larger sample size was used. In the end, after eliminating non-responders and those who recorded no experience in either ASD or spikes a sample of 72 participants was obtained. Similar to the interview approach, non-probabilistic sampling was employed, and only those invited by the researcher could access the questionnaire based on the inclusion criteria mentioned in section 4.1.2.

4.2.3. Questionnaire piloting

Piloting a survey goes beyond checking the validity of its questionnaire: it includes testing for the scale's reliability (Landsheer and Boeije, 2008). In this study, ten individuals were involved in the piloting exercise. Seven were PhD students in the School of Electronics and Computer Science at the University of Southampton, and three were agile practitioners recruited through the researcher's social network. They all completed the questionnaire without any non-response to a question. A reliability test was undertaken from the data obtained to determine whether the measurement scale was appropriate for the study. Furthermore, the participants pointed to a few areas that needed improvements to the phrasing to eliminate ambiguity and thus enhance the comprehensibility of the statements presented.

4.3. Ethical Considerations

Ethics issues inevitably arise when dealing with humans. Respect and confidentiality are needed whenever one is conducting a study using humans as subjects (Oliver, 2010). For this reason, research needs to undergo an approval process before being conducted. In most cases, an institutional review board is mandated to approve research proposals that involve the collection of data from humans. In this study, ethics approval to proceed with the interviews and questionnaire approaches was awarded by ERGO II. To complete the process, six items had to be submitted to the approval body for consideration:

- Ethics application form
- Consent form
- Participant information sheet
- DPA (Data Protection Act) plan
- Risk assessment form
- Interview questions and questionnaire

After submitting these documents, ethics approval was obtained on 09/01/2020 after the ethics committee at the University of Southampton checked the submitted items, and the ethics approval reference number is 53962.

4.4. Data Analysis Instruments

To simplify the calculations and ensure accuracy, IBM SPSS v.24 was used to analyse the quantitative data obtained from the questionnaire. The data were coded to ensure that calculations were easy and precise. Qualitative data from the interviews were analysed using NVivo to detect thematic patterns. From the analysis of the transcripts, a total of five themes were identified. Microsoft Excel 2016 was used to plot various figures from the questionnaire data. The adoption of these three approaches made it possible to analyse data from both the interviews and the questionnaires to answer the study’s research questions.

Chapter 5: Results of Practitioners' Responses Concerning the Roles, Efficiency, and Efficacy of Spikes

This chapter presents the findings from the participant interviews and questionnaires. It reports on the main themes identified in the interview transcripts, both qualitatively and quantitatively. On the other hand, the questionnaire was devised to collect data from software development practitioners who were invited to complete it based on the inclusion criteria outlined in Chapter 4.

We need to understand the difference between the efficiency and efficacy of spikes since this chapter is mostly about them. The efficiency of a spike in ASD is the amount of time and resources needed to achieve the desired outcome. This can be measured objectively by looking at the time taken to complete the task and the resources used, such as the number of people involved in the task, the number of hours spent on the task, and the number of tasks completed (Hossain, Kashem, and Sultana, 2013; Douglass, 2016). On the other hand, the effectiveness of a spike in ASD is the ability to test a potential solution or approach to a problem before committing to a full implementation. By having a short, focused effort to test the approach, teams can decide whether or not to proceed with the work, saving time and effort. The effectiveness of a spike can be measured by assessing how much knowledge and insight was gained through the spike. This could include the amount of time saved by avoiding long and costly development cycles, the accuracy of the decisions made by the team as a result of the spike, and the quality of the final product. Additionally, a team could measure the amount of knowledge gained relative to the cost and time invested in the spike (Leffingwell, 2010; Hunt, 2018).

5.1. Interview Data Collection Process

Due to the difficulty in conducting all the interviews face to face and to accommodate the participants' preferences and convenience, 18 interviews were conducted online by using available applications and just four were held in person. The participants were carefully selected based on colleagues' recommendations, social networks (LinkedIn and Slack) based on specific criteria stated in the previous chapter, and the practitioners' online home pages. The duration of the interviews that was estimated by the pilot was approximately 60 minutes. In the actual interviews, this trend was maintained: the minimum was 46 and the maximum was 72 minutes. The duration was dependent on my follow-up questions and the extent of the responses by the participants. Some participants kept their responses short, and with fewer follow-up questions, they spent minimal time answering the 14 questions. However, the interview took more than one hour with others who were very detailed in their responses. The mean duration of the interviews was 62.78 minutes, with a standard deviation of 6.09 minutes, as presented in Table 5.1.

Table 5.1: Interview duration in minutes

Interview duration	Minutes
Average	62.78
Standard deviation	6.09
Minimum	46
Maximum	72

5.2. Qualitative and Quantitative Analysis of Interviews

This section presents the results of the interviews conducted with participants from various countries. The criteria for deciding if a participant was an expert in ASD was through the preliminary questions that needed them to state their roles in the team and their experience with agile methodologies and spikes. However, some participants did not provide their domains for privacy and confidentiality reasons. The questions mostly focused on answering the 'how' and 'what' questions of the research. The questions were designed to be comprehensive and required practitioners' personal opinions rather than a predetermined response. From an inductive perspective, the interviews were designed to collect the data needed to explore various themes regarding the application of spikes in ASD. The results were from 22 participants from seven countries. The participants were selected through colleagues and social networks such as LinkedIn and Slack. They are all practitioners of ASD with varying experience in several roles. In the interviews, a total of 16 open-ended questions were put to the participants, who responded from their own perspective and experience.

From the interview questions, a total of five themes were identified as addressing the research questions. These themes are based on a combination of participant responses that collectively point to specific issues. The researcher used triangulation to validate the thematic analysis and support the same findings, which is the process of gathering multiple sources of evidence to support a conclusion. This can include collecting data from multiple sources, such as interviews, surveys, documents, and observations; comparing the results of different methods; and having multiple researchers analyse the data independently. In our case, involving multiple researchers was challenging as the research was conducted by a single researcher studying for the award of a PhD. Moreover, the cost and time associated with obtaining multiple researchers to validate the thematic analysis can be prohibitive.

Although the responses are varied, a synthesis of the transcripts was undertaken to construct common themes supported by the responses. Some responses are quoted with the results to emphasize and authenticate the themes being discussed. According to Corden and Sainsbury (2006), using verbatim quotes from transcripts serves as evidence, explanation, and illustration, enhances readability, gives participants a voice and deepens the reader's understanding. Accordingly, this thematic analysis will use quotes from the participants' responses to secure these benefits.

Figure 5.1 summarises the themes identified in the interview transcripts that are covered in the thematic analysis.

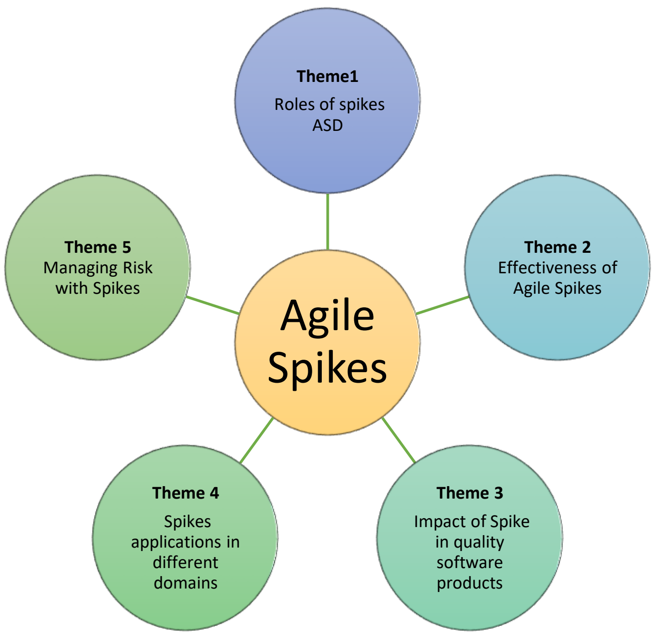


Figure 5.1: Themes identified in the interview transcripts

5.2.1. Participants' demographic information

The interviews commenced by gathering participants' demographic details, such as country of origin, practitioner experience in both ASD and using spikes and the agile role. Of the 22 participants, 10 were from the United States (US), four each from the Kingdom of Saudi Arabia (KSA) and the United Kingdom (UK) and one each from South Africa, Ireland, Portugal and Germany, as shown in Figure 5.2.

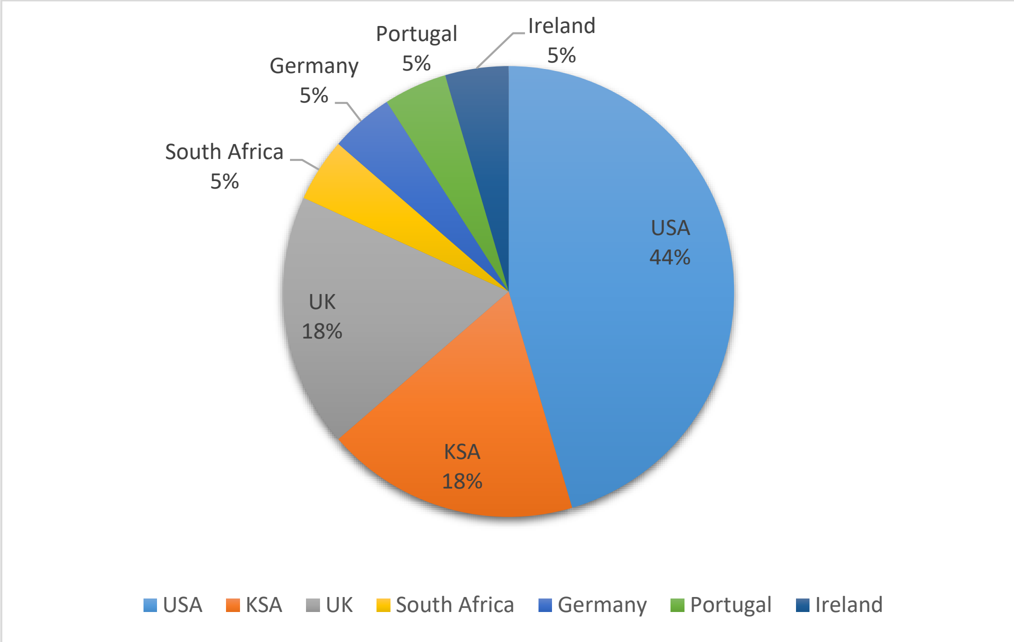


Figure 5.2: Participants' countries

The participants' experience varied. The majority of the 22 participants, 10 in total, were Scrum Masters, while six were POs and six were developers, together representing 42.8%. There were five agile coaches among those interviewed, accounting for 17.9% of responses, and the most seldom seen role was that of tester (QA), just one, representing 3.6%, as presented in Figure 5.3.

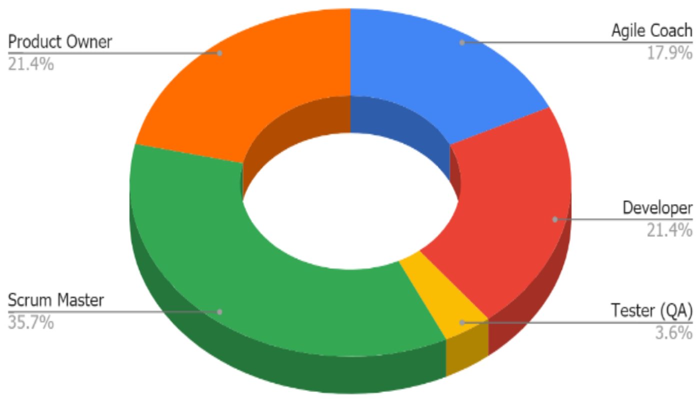


Figure 5.3: Participants' roles

The total of responses (28) is greater than that of participants (22), since some reported multiple roles. For instance, P1, P7, P11, P12, and P14 are at the same time both Scrum Masters and agile coaches, while P14 is both a developer and an agile coach.

From these roles, the participants had experience ranging between one and 20 years. In terms of ASD, the majority had between 6 and 10 years of experience. Notably, 9 of the interviewees had experience in this range. In terms of using spikes, most participants (14) reported having between zero and five years of experience, as shown in the summary in Table 5.2.

Table 5.2: Participants' experience, country of origin and agile roles

Participant	**Experience in Agile**				**Experience in Spikes**				**Country**	**Agile Role**
	0-5	6-10	11-15	16-20	0-5	6-10	11-15	16-20		
P1	✓				✓				US	Scrum Master & Agile Coach
P2		✓				✓			Ireland	Product Owner
P3	✓				✓				US	Scrum Master
P4		✓			✓				UK	Scrum Master
P5				✓	✓				UK	Developer
P6				✓				✓	South Africa	Scrum Master
P7		✓			✓				US	Scrum Master & Agile Coach
P8			✓			✓			Portugal	Product Owner
P9		✓				✓			UK	Scrum Master & Agile Coach
P10		✓			✓				Germany	Scrum Master & Agile Coach
P11			✓				✓		US	Scrum Master & Product Owner
P12		✓				✓			US	Scrum Master & Agile Coach
P13	✓				✓				US	Product Owner
P14		✓				✓			US	Scrum Master
P15	✓				✓				US	Developer
P16		✓			✓				US	Developer
P17		✓			✓				US	Developer
P18			✓			✓			UK	Product Owner
P19	✓				✓				KSA	Developer
P20	✓				✓				KSA	Developer
P21	✓				✓				KSA	Tester (QA)
P22	✓				✓				KSA	Product Owner
Totals	**8**	**9**	**3**	**2**	**14**	**6**	**1**	**1**		

5.2.2. Roles of spikes in agile software development (ASD)

Spikes have been lauded for addressing functional and technical issues arising in software development projects. This section answers RQ1, which seeks to find out the specific roles that spikes play in various agile methods. To answer this research question exhaustively, the theme is subdivided to cover three other sub-themes: the agile method used; the experience of the practitioners in using spikes; and the roles of spikes in ASD.

Agile methods used

Question
What agile methodologies is your team or company using in software development?

While all 22 responses acknowledged that Scrum is their preferred agile method, others articulated further methods that their companies/teams use. c was the second-most mentioned, and seven participants confirmed that it is employed in ASD, as shown in Figure 5.4. Other methods mentioned were XP, LESS (Large-Scale Scrum), TDD (Test-Driven Development), SAFe (Scaled Agile Framework), ATDD (Acceptance Test-Driven Development), RAD (Rapid Application Development) and Lean, among others. The bottom line is that Scrum and Kanban are the most widely used agile methods in ASD.

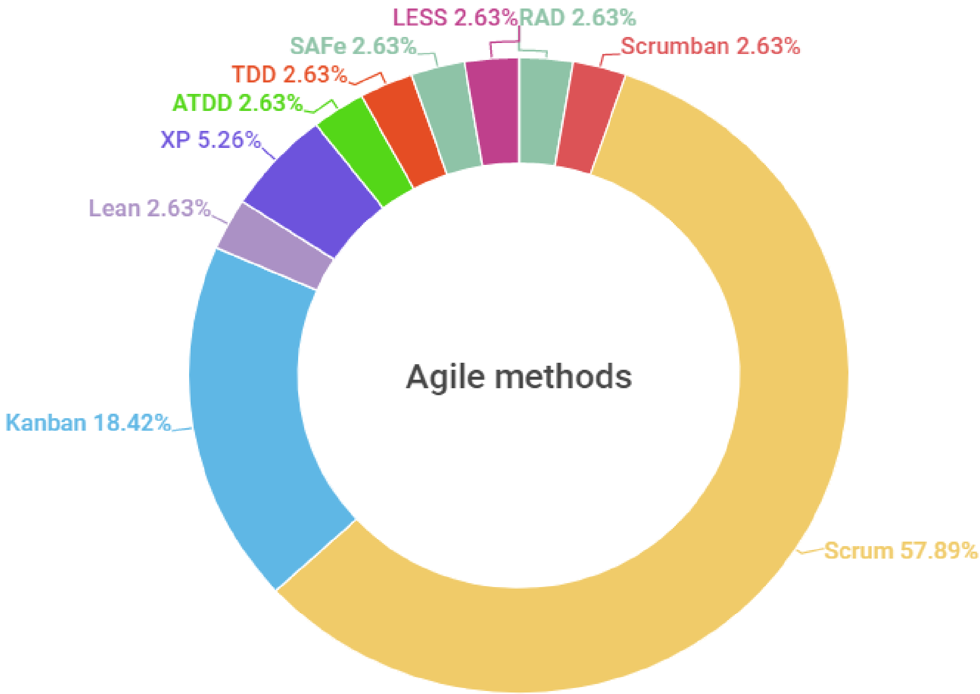


Figure 5.4: Agile methodologies used by practitioners

Experience of spike usage

Question
Based on your experience, what can you say about using spikes in agile software development?

As shown in Table 5.2, the experience of participants in using spikes in ASD varied from zero to 20 years. From their experience, they acknowledged that spikes are used for various purposes in agile developments.

Participants P4, P13, P14, P15 and P16 stated that spikes are used to provide a greater understanding of the project scope. P4 stated that due to their effectiveness, “Spikes allow the developer to add value towards the development and delivery of the feature whilst de-risking it within either a time-boxed period as agreed with team and or for as long as it takes”. The responses of all five participants supported the idea that spikes are useful in understanding the scope of a software development project.

Additionally, participants P8, P20, P21 and P22 stated that spikes are useful in clarifying and eliminating ambiguity in new technologies. According to P4, spikes provide an opportunity for developers to carry out investigations and/or research and development/prototyping based on a feature, epic or story that cannot currently be estimated. P20 added to this, stating that, “Spikes are useful techniques to do more investigation for unclear user story in order to estimate the original story”. This response further supports the idea that spikes are useful in solving uncertainty and ambiguity in ASD projects and new technologies. P5, P8, P14, P21, and P22 support the concept and based on their responses and experience; it is apparent that using spikes has become a useful tool in ASD with respect to reducing uncertainty and ambiguity.

The top roles of spikes in ASD

Question
How can spike help you in agile software development? What are the roles of spikes that may help in ASD?

After evaluating the transcripts of the interviews, it was found that the practitioners’ opinions point to five major roles for spikes in ASD, regardless of the methodology: providing a better understanding of user stories (19 responses); allowing a better estimation (17 responses); reducing uncertainty (15 responses); exploring new/unfamiliar technologies (15 responses) and understanding the business necessity (14 responses). Of the 22 participants, 19 opined that spikes enable agile software developers to produce better estimations. These practitioners revealed the various roles that spikes perform in ASD. For instance, P20 stated, “A spike can provide a presentable result, still being able to provide an actual piece of software – even only a small part provides significantly more value to customers. So, my recommendation would be to use spikes only as a last resort should all other methods to split part of the requirement and reduce risk/complexity fail.” In this regard, other participants added to the information by suggesting roles besides enabling better estimations.

About 15 participants said that spikes are an effective way to reduce uncertainty in ASD. According to P2, despite the effectiveness of spikes in reducing uncertainty, the time spent on the spike itself should be limited. The response by this participant stated, “…during a spike you learn a lot and experiment a lot, but some uncertainty remains until you develop the product. To reduce this uncertainty more, you can use the spike to create a POC, but the risk is to spend too much time on the spike itself.” The other 14 participants supported the idea that part of spikes’ role is to reduce uncertainty in software development projects.

About 15 participants stated that spikes’ responsibility in software development is the exploration of any new/unfamiliar technologies being applied, as shown in Table 5.3. P14 claimed that spikes are useful in understanding a third-party system, “Spikes help you analyse a problem before starting on it. You can break it down into the smallest parts and decide how each part should be handled. They give you an opportunity to fully understand a third-party system and what a possible integration with that system will take.” Further responses add to the evidence that spikes are used as a tool to explore unfamiliar technologies or a third-party system.

Nine participants added that spikes provide prior knowledge of the event of a project. P3, for instance, stated that spikes are a valuable tool in software development as follows, “Originally, we considered work that was noncustomer facing as a spike, but we eventually equated spikes with research or analysis with a result. Our implementation is to use spikes to research problems and create specific tasks to address those problems. We also use spikes to analyse application performance or behaviour and create tasks for improving these elements.”. P13 supports this sentiment and adds, “As the product owner, I’ve found spikes useful in better understanding the scope of implementing something while still in the requirements phase. I use the information gathered from the spike to fully layout requirements that I didn’t know beforehand”. In this response, the participant reveals the usefulness of spikes in obtaining prior insight about new technology. Although respondents stated the ideas differently, they all supported the finding that spikes are essential to gathering information on a project prior to commencement. All the participants were of the same opinion regarding the applicability of spikes in new technologies and gathering preliminary information about a project. In this context, the idea supports the use of spikes in reducing uncertainty and ambiguity in projects.

Based on the uses stated by the 22 participants of the agile practitioners, spikes are concluded to have limited use in minimising the gap between technical and business solutions and prototyping solutions. Although some responses pointed to them as useful in this function, others stated other uses. The roles of spikes in decision-making and researching better ways to accomplish tasks received six responses each (see Table 5.3).

Further uses of spikes mentioned by the participants included providing extra information through exploratory work and learning purposes, as well as use in the User Interface (UI) to develop the first POC and investigation, and better estimate delivery. Providing more information to developers was supported by P1, P2, P9, P12, P17, P18, and P20. They attributed the enormous information spikes provided by exploratory works to improvements in ASD projects and resolving uncertainty. Furthermore, P2, P5, and P6 stated that spikes are not only applied to solve uncertainties, reduce risks, and provide information in the scrum, but also for learning. As agile coaches, they perceive that spikes are essential to learning, pointing out that spikes have nine different roles and applications in software development. As shown in Table 5.3, the top roles of spikes in ASD are providing a better understanding of user stories (20 responses), allowing better estimations (17 responses), reducing project uncertainty, and exploring new/unfamiliar technologies (each with 15 responses). Therefore, spikes improve the accuracy of user story estimation, reduce uncertainty, and enhance how an agile team explores new technologies.

Table 5.3: Spike roles and applications in ASD

Spike Roles	P 1	P2	P3	P4	P5	P6	P7	P8	P9	P 10	P 11	P 12	P 13	P 14	P 15	P 16	P 17	P 18	P 19	P 20	P 21	P22	Total
Explore new/unfamiliar technologies, features, and tasks	●	●		●	●	●			●	●	●	●		●			●		●	●	●	●	**15**
Provides a better understanding of user stories to minimise the gap between technical and business solutions	●	●	●	●	●	●	●	●	●	●		●	●	●	●		●	●	●	●	●	●	**20**
Prototyping a workaround									●	●												●	**3**
Reducing uncertainty and complexity	●	●	●	●				●	●	●	●	●		●			●		●	●	●	●	**15**
Understanding business needs	●	●	●	●					●	●	●	●	●			●	●		●	●	●	●	**14**
Help in decision making	●				●		●					●	●		●								**6**
Researching better ways to accomplish tasks									●		●	●	●			●	●						**6**
Allows for better estimation	●	●	●	●				●	●	●	●	●	●	●	●	●	●		●	●	●	●	**17**

5.2.3. Efficiency of spikes in agile software development (ASD)

Spikes continue to be used by many software practitioners in various domains. Their application in projects depends on their effectiveness and efficiency. In this section, RQ2 is answered, together with its two sub-questions, particularly how agile spikes can be used efficiently in ASD. The section answers the question of how spikes estimate user stories, effort, and delivery time. Lastly, it reports on the appropriate time to apply spikes during an ASD project.

Question
On a scale of 1-5 where 1 being the lowest, how would you rate the efficiency of spikes in agile software development? Why this score?

The efficiency and effectiveness of spikes in ASD generally and specifically in reducing risk, was measured on a five-point Likert scale. The participants were asked to score the efficiency of spikes in ASD. A score of 5 represented 'Very efficient, 4 'Partially efficient, 3 'Neutral', 2 'Not efficient, and 1 'Completely inefficient, as presented in Figure 5.5. The participants were asked to provide a justification for their scores.

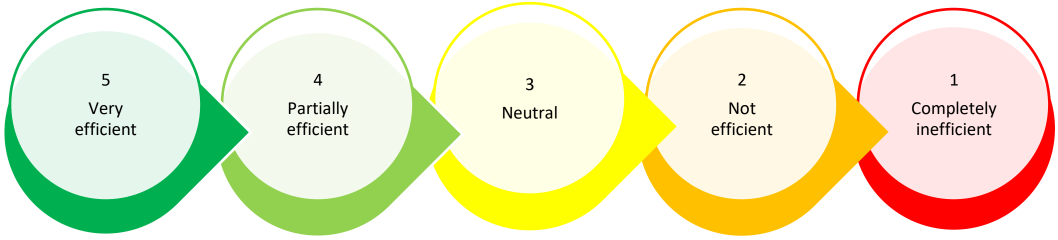


Figure 5.5: Likert rating scale for the effectiveness of spikes in ASD

From the responses, 11 participants scored spikes' efficiency 4, nine scored it 5, only three participants scored it 3 and none scored it at either 1 or 2, as shown in Figure 5.6. The mean score of the responses was found to be 4.31 and the standard deviation 0.63, as shown in Table 5.4.

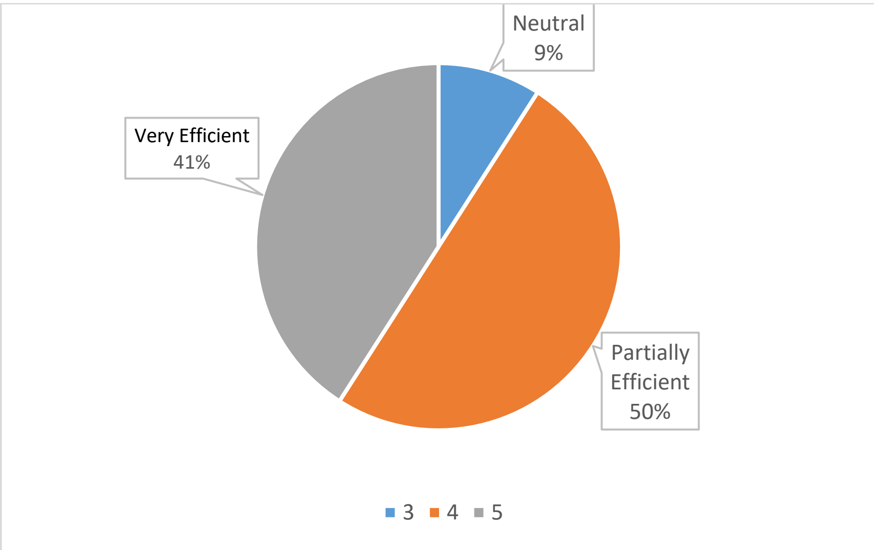


Figure 5.6: Rating of spikes' efficiency in ASD

Table 5.4: Efficiency score for spikes in ASD

	Rating	Count
Score	5	9
	4	11
	3	2
Mean Score	4.31	
Standard Deviation	0.63	

The participants gave varied reasons for their scores. P1 and P5 said that the efficiency of spikes depends on the teams using them in ASD. In particular, P1 said, "I think it depends on the team that is using them. How efficient is the team at defining their spike(s), completing them, and disseminating what they have learned to the other members of the team". P4, P6, P8, and P13 associated their high scores with quickly testing out a hypothesis or exploring an idea, while keeping costs and time to a minimum, so their efficiency is very high, as they asserted. P8 said, "The efficiency of spikes is highly dependent on the particular context and goals of the project. Generally, agile spikes are useful for quickly testing out a hypothesis or exploring an idea, while keeping costs and time to a minimum". P13 support this claim by saying, "Spikes can help teams to quickly answer questions, such as whether a certain feature is feasible, and whether a certain approach would be beneficial". In the same context, P7, P14 and P21 associated their high scores with the experimentation and exploration period. P10 and P16, who scored the efficiency of spikes at 3, held the opinion that spikes' efficiency depends on the project complexity and team undertaking this project.

Roles of spikes in estimation

Question
What roles would you say spikes have in estimating user stories in agile development? How do spikes estimate user stories in agile?

The responses of the participants revealed that spikes are used in agile software projects to estimate user stories, and work on the product backlog and time required. Of the 22 participants, 16 stated that the most crucial role of spikes is estimating user stories. P4, P8 and P10 described spikes as effective in estimating user stories. The same sentiments were voiced by P19, P20 and P21, as shown in Table 5.5. Quoting P14 from the transcript, "Spikes themselves should not be estimated. However, once a spike is completed then the story is able to be estimated, and that estimate should be a more correct estimate". P2, P3, and P5 stated that spikes improve clarity and estimate tasks when applied research is conducted. According to P3, "Spikes provide us more clarity regarding specific tasks so we can then more accurately estimate those tasks and the effort required". Other practitioners, including P11, P15, and P17, mentioned that part of spikes' role in ASD is estimating the effort required to complete a story. P15 stated, "Spikes are very important for estimating user stories, mostly when there is a need to either integrate with third-party API providers OR identify precise user requirements. The development team can closely estimate the time required to integrate with a third-party API by investigating their documentation (or lack thereof) and drafting in their mind a possible implementation so they can estimate the effort required to complete it".

Participants also mentioned that spikes are used in estimating the work in the product backlog. P1, P3, P4, and P11, for instance, stated that spikes allow a team to plan and get an idea of the intensity of the work ahead. P1 stated, "Spikes, by themselves, don't estimate anything. They allow the team to plan a small piece of work that can be completed within one sprint… By learning more, the team is able to ask better questions of stakeholders and as these questions get answered, can more effectively refine, decompose, and estimate work in the product backlog". As demonstrated in Table 5.5, the participants who gave a similar response were P16, P20, P21, and P22.

Six participants mentioned that spikes are useful in estimating the time required to complete a task. P2 and P10 asserted that because spikes help in solving uncertainty and understanding project necessities, the time required for completion can easily be estimated. Specifically, P2 opined that, "spikes are also useful in understanding the business necessity so that we can develop what is useful for the company and do not waste time". Other participants, including P9, P15, P16 and P22, gave similar responses.

Appropriate time for spike usage

Question
What is the appropriate time to use spikes in agile software development?

The best time to apply a spike depends on the function that it is intended to perform. According to the practitioners' responses, 18 preferred to use spikes during a sprint, as shown in Figure 5.7. The answers given by 18 participants support this usage; however, the participants also reported using them between backlog grooming (refinement) and sprint planning (13 responses), during sprint planning (7 responses), and when faced with uncertainty (9 responses), as presented in Table 5.6.

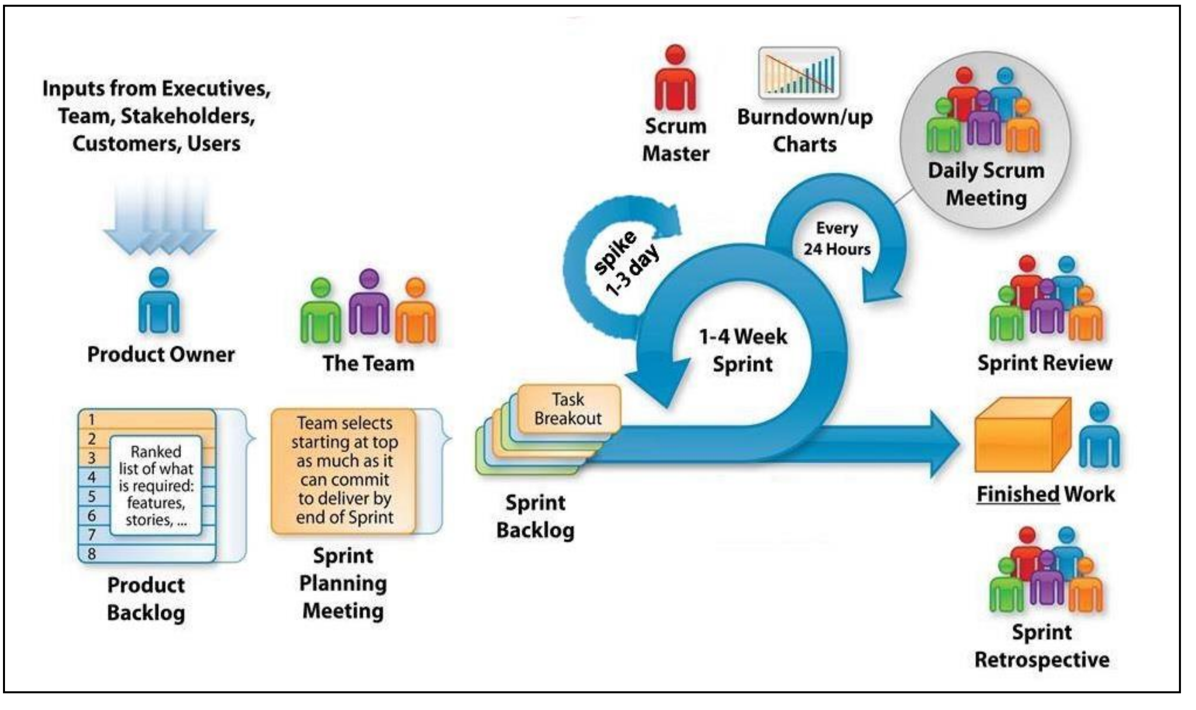


Figure 5.7: Employing spikes during sprint

The influencing of spikes on project activities

Question
Do you believe agile spikes are influencing project activities? If yes, how?

The influence of spikes on project activities defines their overall impact on product quality. All participants agreed that spikes influence the project's activities, but they differed in how this could influence. Of the 22 participants, nine said that spikes affect projects' timelines. P8 responded to the question of whether and how spikes influence project activities, saying "Yes, because they affect the tasks' number, time, complexities, sprint and velocity". In the same context, P1, P10, P12, P14 and P15 agreed that spikes indeed influence project activities and affect the timelines. P15 asserted, "I would just say that it affects the entire sprint in terms of speed and more value-driven sprints". P12 also said, "Yes. The knowledge garnered from the research can alter the project if the outcome is not what the product or development team was expecting". P10 affirmed that spikes delay development activities. P14 supported this claim by saying, "I believe agile spikes have an impact on project activities. In my experience, while spikes can be a helpful tool, having too many spikes on a project indicates that the PO/stakeholder does not fully understand the project. Excessive spikes can disrupt the project's timeline. They can have an impact on backlog grooming because there are too many questions that need to be answered. They can impact stakeholder interactions because the issues raised by the spikes must be addressed". However, one participant did not provide information that would apply to this case. P11 stated, "Spikes are just another work item, they don’t influence other work".

Other influences mentioned by participants included clarifying the required tasks (8 responses), easing the planning process (9 responses), understanding the alternative approaches (3 responses), dependencies and feasibility (2 responses), as summarised in Table 5.7.

Table 5.5: Spike estimation roles, as deduced from participants

Spike Roles	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	Total
Estimating the size of user stories		●	●	●	●		●	●	●	●		●	●	●	●		●		●	●	●		16
Estimate the work in the Product Backlog	●		●	●							●					●				●	●	●	8
Estimate the delivery time		●							●	●					●	●						●	6

Table 5.6: Practitioners' opinion on the appropriate time to apply spikes

Appropriate time	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	Total
(Before sprint planning) Between backlog grooming and sprint planning		●			●	●	●	●		●			●	●	●	●		●	●	●			13
During sprint planning				●		●	●	●	●	●						●						●	7
During sprint	●	●	●	●	●	●	●	●	●		●	●		●	●		●		●	●	●	●	18
When faced with uncertainty											●	●			●		●	●		●	●		9

Table 5.7: The impact of spikes on project activities

Outcomes of spikes	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	Total
Affect the timeline		●				●		●	●	●		●		●	●						●		9
Clarify the required tasks	●		●	●									●				●			●	●	●	8
Ease the planning process	●			●	●			●							●	●		●	●	●			9
Understand alternative approaches		●														●							2
Understand dependencies and feasibility	●	●																		●			3

5.2.4. Effectiveness of spikes in risk management and software domains

As reported by most participants, agile spikes are useful in mitigating risks in ASD projects. In this section, RQ3 will be answered through the results. Notably, the question of how spikes can be used to manage risk effectively in ASD is covered in this section, and it addresses how uncertainties are defined in ASD. It explains the three areas of uncertainty: knowns, known unknowns, and unknown unknowns (see Figure 5.8).

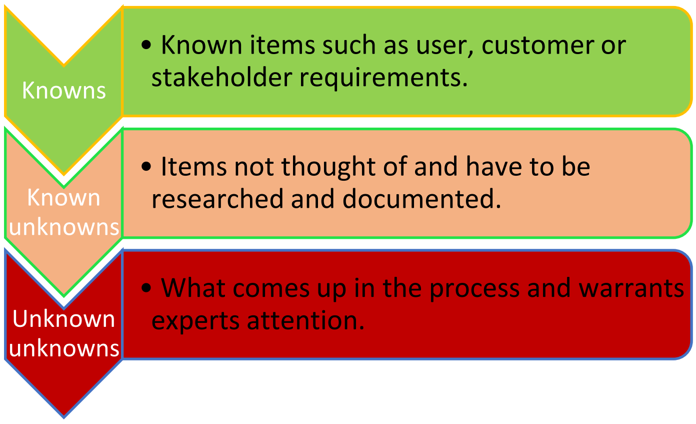


Figure 5.8: Area of uncertainty in ASD

Based on P5's and P12's responses, there are three areas of uncertainty in ASD: first, the "known knowns", pointed out by P12, which are items that the client, customer, or stakeholder knows that they want in the software, and which are planned for. Second, the "known unknowns" are items that a team or the software developers do not usually think of but for which the team will have to conduct spikes or document. Lastly, the "unknown unknowns" that arise throughout development and need the attention of the team to use spikes to clarify. P5, too, mentioned these three areas as categories of uncertainty in ASD.

The rate of effectiveness of spikes in risk reduction

Question
On a scale of 1-5 where 1 being the lowest, how will you rate the effectiveness of spikes in reducing risk? Why this score?

The responses of the participants on the effectiveness of spikes in reducing risks in ASD revealed differences in perception. Of the 22 practitioners interviewed, most rated it as 4 and 5. Notably, nine participants scored the effectiveness at 4, eight at 5, and only five at 3, and none scored it 1 or 2, as shown in Figure 5.9.

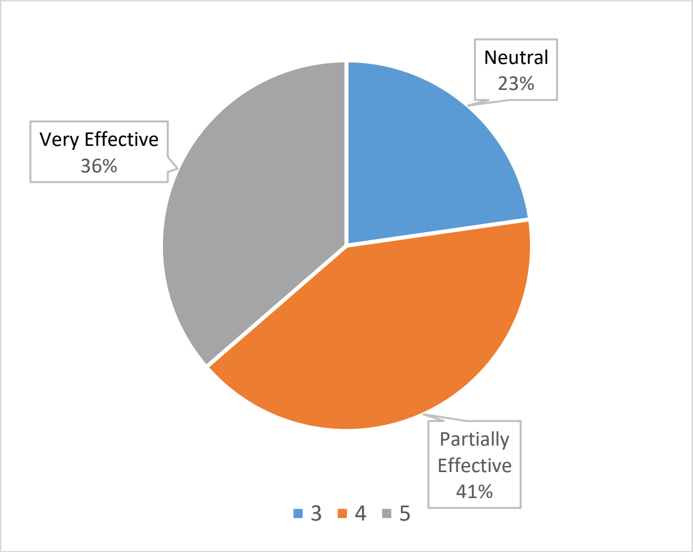


Figure 5.9: Rating of spikes' effectiveness in reducing risks

The participants were asked to score this on a scale of zero to 5. From the data, the mean score was found to be 4.14 with a standard deviation of 0.76, as shown in Table 5.8.

Table 5.8: Effectiveness of spikes in reducing risk

	Rating	Count
Score	5	8
	4	9
	3	5
Mean Score	4.14	
Standard Deviation	0.76	

The rationale given by participants for the scores varied depending on the scores. P1 and P17 believe that reducing risk depends on the team and their implementation of agile. P1 said, "The ability to reduce risk depends on the team's effectiveness in planning, executing, and communicating the results of a spike. Doing the wrong spike is not very effective at reducing risk. Doing the right spike, but not spreading the knowledge across the team, just changes the risk. The right way to execute the spike depends on the team and the spike". Whereas P7 said, "Anything you don't know is a risk. None of us knows the future perfectly. Spikes are all about finding solutions, solving problems, and answering questions to help the project succeed". P4 and P11 believe spikes reduce risk by giving the team additional time to investigate and research potential solutions.

Other responses centred on the usefulness of spikes in solving uncertainties and the impact on the outcomes and resource and time management through the knowledge and insight gained, as articulated by P5, P9, P10, P15, and P19.

On the other hand, the participants who gave a rating of 3 to the spikes reducing risk have a different opinion. For instance, P10 said: “Scrum itself reduces risk by frequent inspection and adaption of the shippable software at the end of each sprint. The risk of maldevelopment is reduced by showing the software frequently to the stakeholders, risk of low quality is reduced by constant testing of each increment. A spike reduces risk but developing a slice of a feature does more”. While P3 stated that spikes do not necessarily reduce risk, they do make risk more visible to development teams.

The influence of spikes on the quality of a software product

Question
What do you believe in the influence of agile spikes on the quality of software products?

The quality of the final product is influenced by spikes’ impact on project activities. According to their responses, 20 participants agree that spikes affect the quality of software products positively, while two said that they have no impact on quality. In the practitioners’ view, spikes improve the quality of the product by providing better planning and decision-making (7 responses), understanding the project (6 responses), providing extra time (3 responses) and understanding the risks and complexity (16 responses). Table 5.9 provides a summary of how spike factors enhance product quality by influencing project activities. P1 said that “…completing spikes (or, even more generally, gaining a deeper understanding of the product, its context, and the work before the team) absolutely increases the quality of a software product”. In the same context P2 stated, “spikes significantly improve software quality and the time required to develop new features”. Similarly, P3 said “spikes have a positive influence on the quality of our products because they help the team understand the complexity and risk more clearly prior to actually developing”. P11 asserted that spikes improve quality significantly because they give the team extra time to ensure they are writing code in the best way to implement the new work. The team aren’t rushed to complete all the work in the sprint because much of the initial research was conducted in a previous sprint. In the experience of P12, "when we allow the team to conduct research, we are giving them the power to determine the path forward, which gives them ownership of what they are working on and, thus, makes the products better".

P10 and P13 have slightly different perspectives, where P13 stated, "I believe spikes positively influence the software products, but I would argue to say the software would probably come out the same without the spike. I believe the spikes influence is more noticeable in the requirements phase as the information is gathered and documented before the relevant PBI is pulled into a sprint".

The practitioners reveal that almost all had witnessed the importance of spikes to the overall quality of software development.

Identification of project uncertainty

Question
How is project uncertainty defined in agile software development? Can you list some causes of uncertainty in agile software projects?

Most issues in ASD are caused by uncertainty. In trying to identify the common causes of uncertainty in software development, one of the interview questions asked participants to list the causes that they were aware of. Most cited undefined/unclear user, stakeholder, or customer requirements as the major cause of uncertainty in ASD. For instance, P20 states that the causes of uncertainty in software development are:

- Stakeholders don't know what they want and cannot specify it up-front.
- The development team lacks knowledge of new technology that needs to be developed.
- The development team's user stories and tasks are unclear.

Other participants, such as P9, P10, P21, and P22, revealed that cross-functional dependency in teams creates a level of uncertainty. As shown in Table 5.10, other causes mentioned are the team’s experience and skill (10 responses), unfamiliar technologies (12 responses), poorly written stories (7 responses), misuse of the agile process (5 responses) and project complexity and compliance (10 responses).

Table 5.9: Practitioners' summary of spike factors that boost the quality of software products

Spike quality factors	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	**P17**	**P18**	**P19**	**P20**	**P21**	**P22**	**Total**
Provide better planning and decision making	●				●		●					●	●		●		●						7
Provide better understanding of projects	●									●			●			●	●				●		6
Provide extra time				●					●		●												3
Explore new/ unfamiliar technology and tasks	●	●		●	●	●			●	●	●	●		●	●				●	●	●	●	15
Understanding risks and complexity	●	●	●	●				●	●	●	●	●		●			●	●	●	●	●	●	16

Table 5.10: Practitioners' opinions on the most common uncertainty factors

Uncertainty factors	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	**P17**	**P18**	**P19**	**P20**	**P21**	**P22**	**Totals**
New/unfamiliar technologies or tasks	●	●	●	●				●		●	●		●						●	●	●	●	**12**
Lack of experience for the agile team	●	●				●		●		●	●				●				●	●		●	**10**
Unclear/incomplete (customer/stakeholders) requirements	●	●	●	●				●					●	●	●		●		●	●	●	●	**13**
Poorly written stories				●				●									●		●	●	●	●	**7**
Misuse of agile processes	●											●					●			●	●		**5**
Project complexity and Compliance	●	●	●	●				●					●						●	●	●	●	**10**
Cross functions and dependencies									●	●											●	●	**4**

Spike applications in different domains

Question
Spikes are known to be used in different domains during agile software development. What are some of these domains that you have utilised or have witnessed spikes being used?

The opinions of the participants on the domains in which spikes are applied depending on their role and experience. While some reported only one domain, others reported multiple. Spikes can be used in all software development domains. From the interviews, IT and software development had the greatest support, with eight participants, followed by web development with seven participants, and API and mobile apps domains, each with six responses, as shown in Table 5.11. Further analysis of spike applications revealed usage in the areas of the supply chain, finance, marketing, retail, human resources, and embedded software. The SaaS domain was mentioned by P1, P13, and P21 as an area in which they had seen a tremendous application of spikes. Data analysis and integration is also an area in which spikes have been extensively applied. According to P22, spikes are used in data warehousing, and P7 observes that spikes are used in investigating the best research tools and methods in domains such as online marketing and supply chains.

From the responses of participants, the domains that emerge include UX design, big data, cloud computing, the Internet of Things and IT. Areas within the bid data domain that were mentioned by the participants include online marketing, supply chain software, human resource management, retail and finance software, medical and healthcare application, and communications. Other identifiable domains include cloud computing, mentioned by P3, P7, and P17. The information security domain was mentioned by P5 regarding the use of spikes in developing software for air traffic control and antivirus protection.

Table 5.11: Spike applications in different domains

Spike applicability in domains	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	Totals
New visual interface and business analysis		●																					1
Cloud applications			●																				1
UX design				●																			1
Air traffic control					●																		1
Antivirus software					●																		1
Supply chain software							●																1
Medical and health care	●									●													2
Server or database migrations, development, and warehousing			●				●										●						3
Communications					●	●				●													3
HR software					●		●																2
SaaS	●												●								●		3
Data analysis and integration		●	●				●															●	4
API development												●		●		●	●	●	●				6
Web development												●			●		●	●		●	●	●	7
Mobile apps												●			●		●	●			●	●	6
Embedded software												●											1
IT and software development		●					●	●	●		●					●				●	●		8

Most appropriate agile methods for spike usage

Question
In your opinion, what are the most appropriate agile methods that can use spikes effectively?

Although all the participants said that scrum is their company's preferred agile method, individual opinions varied significantly. Participant 12 believed that, when using spikes, there is no specific method that is better than others. Similarly, P7 stated, "I don't think any agile methodology should prohibit the use of spikes. If your company has a team outside the agile team that can or usually performs this work, which is fine, too". Despite using Scrum, P1 supports P7 and P12 claims by stating, "Any agile methods - including homegrown methods and frameworks - can take advantage of spikes".

Eleven other participants were of the same opinion, while ten regarded Scrum as the most appropriate methodology for the use of spikes. In multiple responses, others included XP and Kanban. Notably, P9 responded, "I have used them in Kanban and Scrum. However, I believe it can be used in all agile methods". Only one respondent, P17, supported XP as an appropriate method, while three, P19, P2, and P8, claimed that Kanban is the best method to apply spikes in ASD, as shown in Figure 5.10.

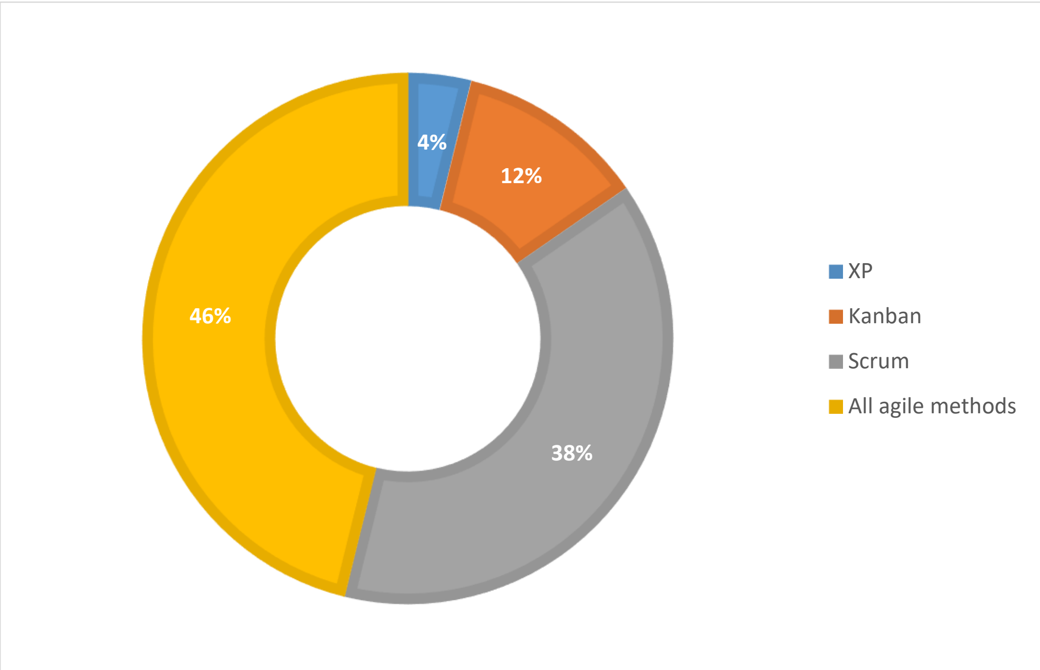


Figure 5.10: Most appropriate agile method to apply spikes

The usefulness of spikes in mitigating technical and functional risks.

Question
How useful do you find spikes to reduce technical and functional risks?

The effectiveness of spikes in reducing technical and functional risks was discussed with the participants. Most, at 15 responses, agreed that spikes reduce both technical and functional risks, while five participants said that they are effective in minimising only technical issues. Only one respondent (P8) said that spikes are effective in reducing functional issues. P18 held the opinion that spikes are not useful in reducing either issue. Among the responses given by the participants on the issue, P14 said “they are very useful and allow us to time-box time spent investigating issues”. Other participants, such as P12, P17 and P22, further insisted that spikes are essential in reducing both types. P1, P2, and P5 stated that spikes are helpful when the development team use them properly and vice versa. P 5 said, “It depends on the development team. The biggest issue is how the team works. If the team use the spikes properly, they can surely reduce the functional issue. The technical issues that are related to missing knowledge can be reduced too. While the spikes do not much impact the technical problems due to mistakes or technical debt”. Figure 5.11 shows the percentage of responses affirming the spikes' effectiveness in reducing technical and functional risks.

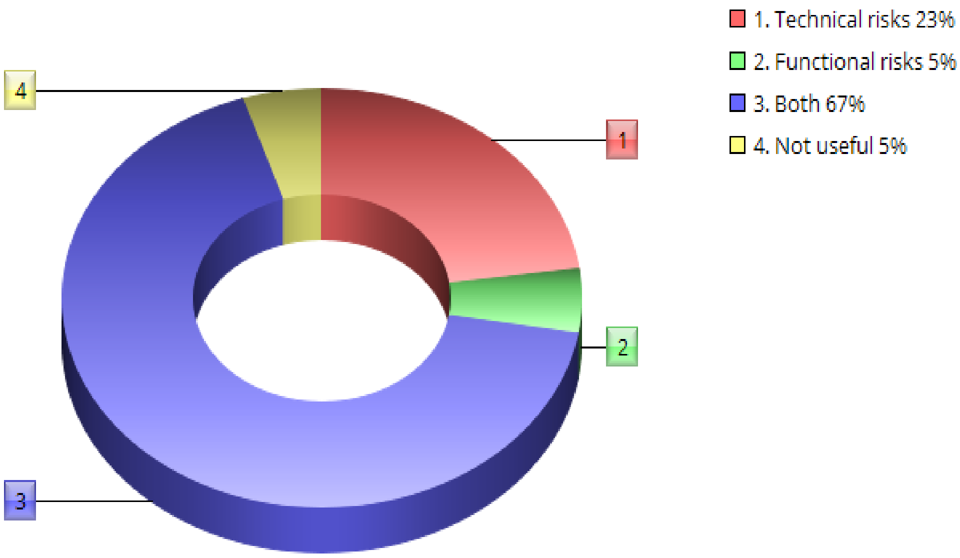


Figure 5.11: Practitioners’ opinions of the effectiveness of spikes in reducing technical and functional risk

Spikes in risk management

Question
How did you find the use of spikes in risk management?

The respondents highlighted various types of risk that are manageable using spikes. For instance, P1 said “Spikes are a form of risk management. There are multiple forms of risk: technical, business, and operational risk (sometimes schedule and budget risks). Spikes can address all of these.”

Due to the presence of such risks, spikes are often applied in software development in mitigation. Based on the responses, a reduction in the risk of maldevelopment is achieved through the proper application of spikes. According to P7, “Spikes is the realm of risk management, which entails identifying risks that threaten the project's success and either accepting, mitigating, or developing contingency plans. Uncertainty exists in all projects, including software development. One way agile adds value is by removing uncertainty earlier (as compared to waterfall development)”.

In addition, P22 asserted that spikes are useful in solving both functional and technical risks. To quote from P22’s response, “Spikes can reduce the risk of technical debts and maldevelopment. For instance, when we are planning the sprint, we usually add 2-3 days extra time to do spikes in case we face any technical or functional issues during the sprint.” P2 pointed out that spikes reduce risk by reducing the unknown, creating POC (Proof of Concept), and dedicating time to test incremental features.

In the experience of P12 to use spikes in risk management, "When new regulations are released, we have had our team test a new API to incorporate (CCPA related), or how we would gather data for our customers. A couple of our team members spend time finding and researching the implications of this new regulation without knowing precisely what data was being collected. Another example was security related and how we handled cross-site scripting (XSS) and how to make our products more secure".

Although participants expressed varied opinions, they all held that spikes are effective in managing risk in ASD.

5.3. Questionnaire Data Collection Process

The questionnaire consists of 30 questions, comprising both closed- and open-ended questions. Participants were carefully selected from the researcher's professional relationships as well as social networks such as LinkedIn and Slack based on the inclusion criteria identified in chapter 4. Of the total targeted, 83 consented to participate in the survey, and its link was provided to them; however, after data cleansing to eliminate some random responses, data from only 72 participants were used in this research. The cleaning involved discarding participants who made random or non-responses to critical questions, including on agile roles, and those who had no experience in spikes. In the data collected, there were no outliers since the quantitative questions used a predetermined scale. Those participants included in the final sample had at least one year of experience and completed all the questions in the questionnaire. The responses to the closed-ended question were coded in SPSS for analysis.

5.4. Quantitative Analysis of Questionnaires

This section presents the findings of questionnaires distributed to participants from various countries. The questionnaire contains 30 questions, including both closed-ended and open-ended questions. The questions were designed to be comprehensive to elicit practitioners' opinions and perceptions about the spike's usage in ASD.

5.4.1. Participants' demographic information

In the questionnaire, the participants were asked to provide the following demographic information (see appendix G):

- Experience of agile
- Experience in using spikes
- Role(s) in the use of agile
- Agile role in which the participant had the greatest experience.

As mentioned earlier, data cleansing excluded participants without experience in using agile spikes or agile development. Therefore, all 72 participants have either used or witnessed spikes' use in ASD. The respondents' average experience with agile developments was 6.69 years, with a standard deviation of 3.1 years. In the same context, their experience with spikes had an average of 5.19 years and a standard deviation of 3.2. This implies that, generally, the participants had more experience in agile development than in spikes. It also suggests that the application of spikes in agile development tends to be adopted after a practitioner has gained experience using ASD strategies. A summary of the participants' responses is represented in Figure 5.12.

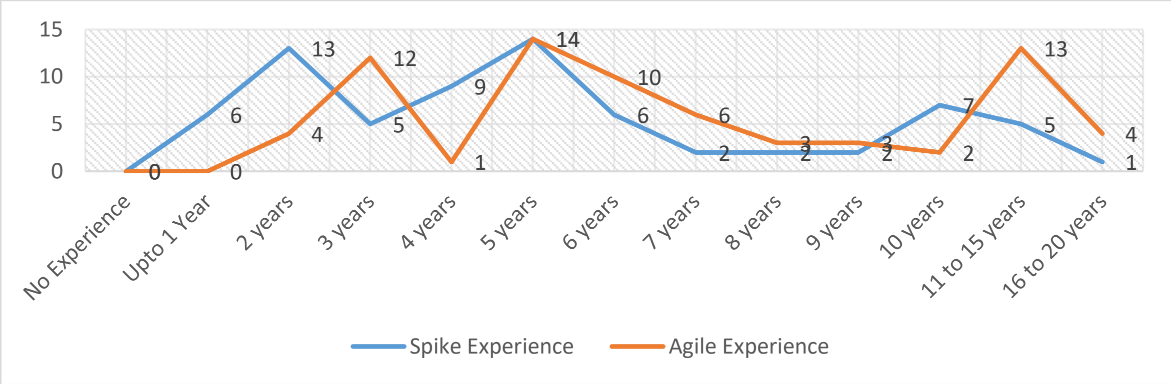


Figure 5.12: Participants' experience of using agile software development and spikes

Of the participants, there were 63 Scrum Masters, 18 POs, eight developers, eight testers (QAs), 40 agile coaches, 2 PMs, two business analysts (BAs) and one subject matter expert (SME), as shown in Figure 5.13. Some participants had more than one role, making the tally more than the sample size of 72.

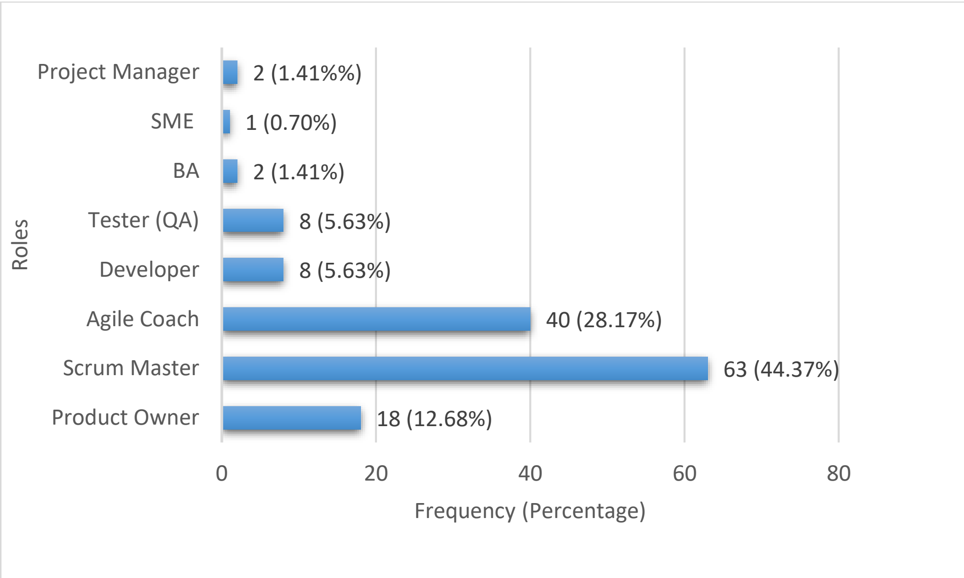


Figure 5.13: Participants' agile roles

After cross-tabulating the agile roles and experiences in ASD and spikes, it was found that most participants had an experience of one to five years. Around 49 participants had 1–5 years of experience in ASD, and 81 had the same in utilising spikes in ASD. Similarly, 48 had an experience of 6–10 years in ASD, and 47 had the same experience with agile spikes in

ASD. Notably, only 10 participants had an experience between 16 and 20 years in ASD, and only two respondents had an experience with agile spikes, as shown in Table 5.12.

Table 5.12: Cross-tabulation of participants' roles with their work experience

Agile Role	Experience in Agile				Total	Experience in Spikes				Total
	1-5	6-10	11-15	16-20		1-5	6-10	11-15	16-20	
PO	6	3	9		18	9	8	1	0	18
Scrum Master	27	22	11	3	63	41	16	5	1	63
Agile coach	10	18	8	4	40	22	12	5	1	40
Developer	1	2	3	2	8	1	6	1	0	8
Tester (QA)	2	3	2	1	8	4	4	0	0	8
BA	1	0	1	0	2	2	0	0	0	2
SME	0	0	1	0	1	0	1	0	0	1
PM	2	0	0	0	2	2	0	0	0	2
Total	49	48	35	10	**142**	81	47	12	2	**142**

Some respondents held more than one agile role and had more experience in one than in the other. Figure 5.14 shows that the respondents had more experience as Scrum Masters and agile coaches, with 40 and 15 responses respectively. Only a few, 12, had more experience as POs. Other roles reported included one tester (QA), one PM, one BA and two developers. This implies that most participants who use spikes are Scrum Masters and agile coaches.

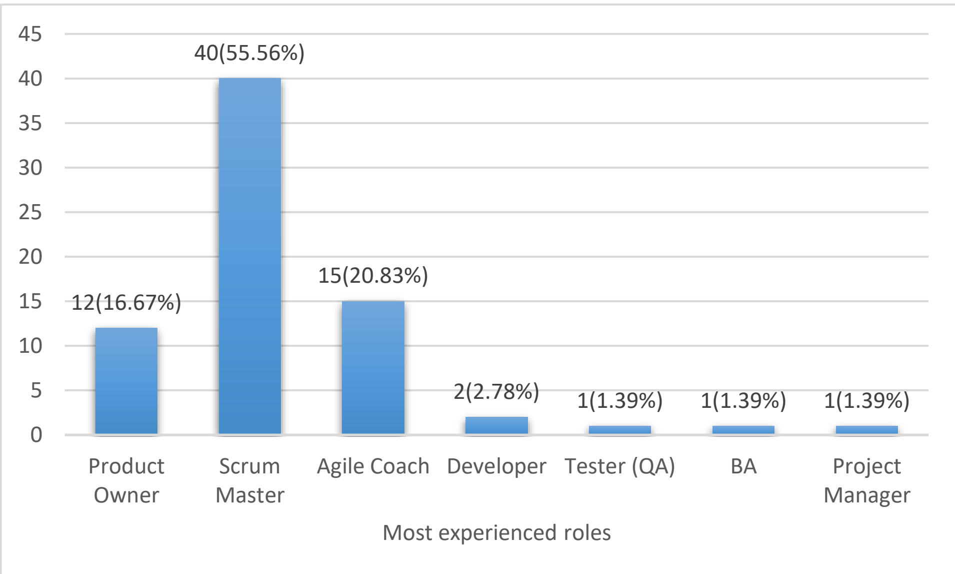


Figure 5.14: Agile role in which respondents had the most experience

5.4.2. Agile methodologies

In the questionnaire, the participants were asked the following question:

In your experience of agile development, which is the agile method you have used the most?

In evaluating the agile methodologies, the questionnaire required participants to select/state the agile method they mainly employed. The most common was Scrum, at 90.28%, and the least was the Dynamic System Development Method (DSDM), with 1.39% of participants. The dominance of Scrum explains why most participants are Scrum Masters and have the most experience in this role. As shown in Figure 5.15, other responses accounted for 1.39% of participants. This response points to a mix of practices utilised in agile development.

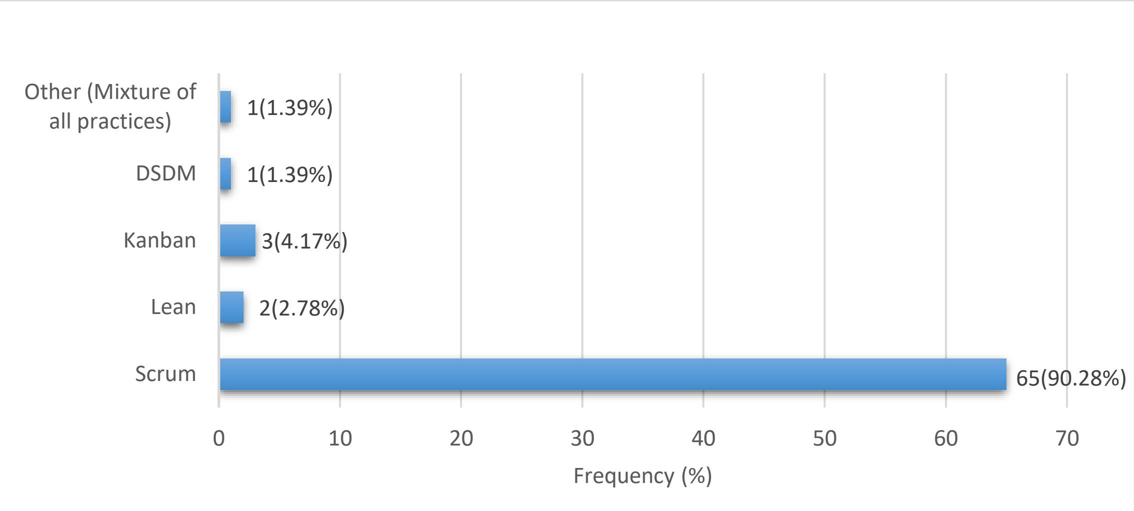


Figure 5.15: Most used agile methods by participants

5.4.3. Effectiveness of spikes in risk management

Concerning the effectiveness of spikes, the participants were asked the following questions in the questionnaire.

Based on the agile role you have the most experience of, to what extent do you agree with the statement that agile spikes are effective in risk management?

In your experience with agile software development, do you think that spikes can increase the quality of the product?

In response, 54.2% of participants agreed that spikes are effective in managing risk, and 30.6% strongly agreed with the same question. However, about 15% disagreed or were neutral about the effectiveness of spikes in risk management.

As shown in Figure 5.16, the high number of respondents agreeing that spikes are effective in risk management indicates the technique's prevalence in different domains. It also implies that risk management has driven more software developers to use spikes.

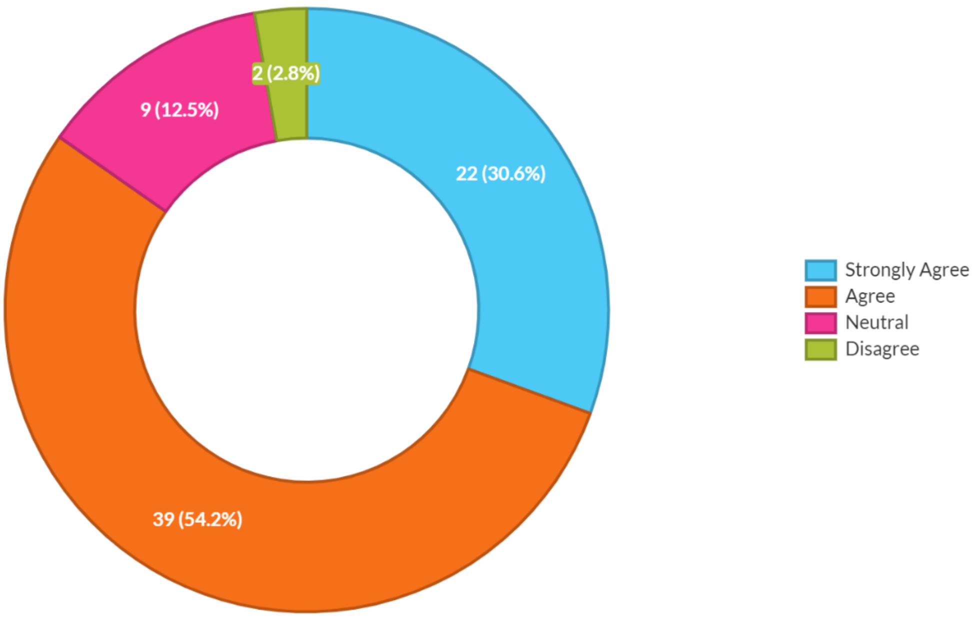


Figure 5.16: Participants' opinions of the effectiveness of spikes in managing risk

Concerning the efficacy of spikes in improving software products' quality, most respondents, 91.67% (66), confirmed that spikes are helpful. Although 6.94% (5 participants) denied that their application does help, the majority responded in the affirmative, providing further evidence of the usefulness of spikes in risk management in ASD. Nevertheless, about 1.39% of respondents were unsure about the effectiveness of spikes in improving the quality of end products of ASD, as depicted in Figure 5.17.

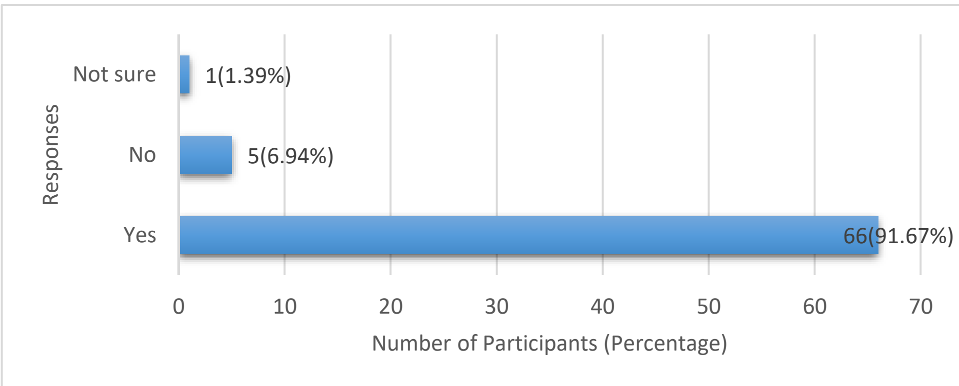


Figure 5.17: Participants' views on the use of spikes to enhance the quality of software

5.4.4. Agile spikes in various software domains

To identify the domains in which the participants had applied or seen agile spikes being used, the following question was asked:

In which of the following domains have you applied or seen spikes being applied more frequently?

The application of spikes in ASD is known in multiple domains, using various methods. According to the data obtained, user experience design (UX), at 22.73% (35 responses), and cloud computing, at 15.60% (24 responses), lead in the domains that utilise spikes frequently. Computer science education is the domain that least uses spikes, with only 2.6% (4 responses) of the respondents affirming it, as shown in Figure 5.18. The other domains, such as e-commerce, SaaS, web development, mobile apps and healthcare software are critical areas in which spikes are extensively applied. Most participants reported using agile spikes or having seen them used by others in more than one domain, hence the large number of responses exceeding the sample size.

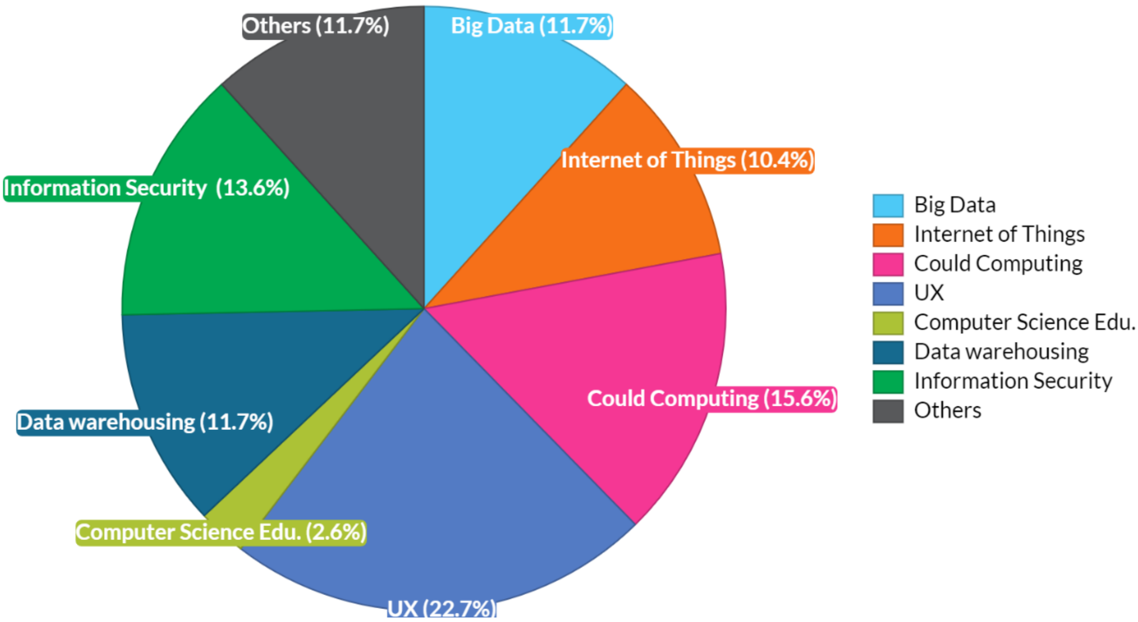


Figure 5.18: Domains in which participants apply agile spikes

5.4.5. Effectiveness and Efficiency of spikes in ASD

In the questionnaire, the respondents scored the efficiency of spikes in ASD and their effectiveness in domains, as well as in risk management, through the following questions.

Out of the domains you selected or stated above, kindly rate out of 5 points the efficacy of applying spikes in these domains, where 1 is low, and 5 is high.
On a scale of 1–5, where 1 is low and 5 is high, please rate the efficiency of spikes in agile software development.
On a scale of 1–5, where 1 is low and 5 high, how do you rate the effectiveness of spikes in risk management with regard to agile software development?

The efficacy of spikes in the domains according to the participants is presented in Table 5.13. About 56.94% of participants scored the effectiveness of spikes in the domains they mentioned at 4 out of 5. Only 13 participants rated it below 4; 12 participants rated it 3/5, and just 1 rated it 2/5. Overall, 81.94% (59 participants) scored the effectiveness of spikes in domains above 4/5.

Similarly, participants were asked to rate the efficiency of spikes in ASD. Around 48.61% of the participants scored it 4/5, while 31.94% scored it 5/5. The percentages represent 35 and 23 participants, respectively. However, 19.44% (14 responses) scored the efficiency at 3/5, as shown in Table 5.13.

To ascertain the effectiveness of spikes in risk management, the participants were asked to rate it out of 5 possible points. The most significant proportion of participants (54.17%) rated it 4 out of 5, and 19.44% (14 responses) scored it 5/5. Only one participant scored it 2/5 regarding risk management, as shown in summary in Table 5.13.

Table 5.13: Summary distribution of scores for efficiency and effectiveness of spikes

Measured item	Score (out of 5 points)	Frequency	Percentage (%)
Efficacy of spikes in different domains	5	18	25.00
	4	41	56.94
	3	12	16.67
	2	1	1.39
Efficiency of spikes in ASD	5	23	31.94
	4	35	48.61
	3	14	19.44
Effectiveness of spikes in risk management	5	14	19.44
	4	39	54.17
	3	18	25.00
	2	1	1.39

On average, the respondents scored spikes' efficacy in specific domains at 4.06, with a standard deviation of 0.68, while their efficiency in ASD scored 4.13 on average with a standard deviation of 0.71. Similarly, spikes' effectiveness in risk management scored on average 3.92 with a standard deviation of 0.70. As shown in Figure 5.19, the median response for all three variables was found to be approximately 4.0. This value also represents the computed mean response of the scores for each variable. The upper cut-off (maximum score) across all the three variables was 5.0; however, the lower cut-off (minimum score) was 2.0 for efficacy of spikes in domains, 3.0 for efficiency of spikes in ASD and 2.0 for the effectiveness of spikes in risk management. The lower cut-off for spikes' efficacy in domains was treated as an extreme outlier in the data, despite falling in the range of 1 to 5, as shown in Figure 5.19.

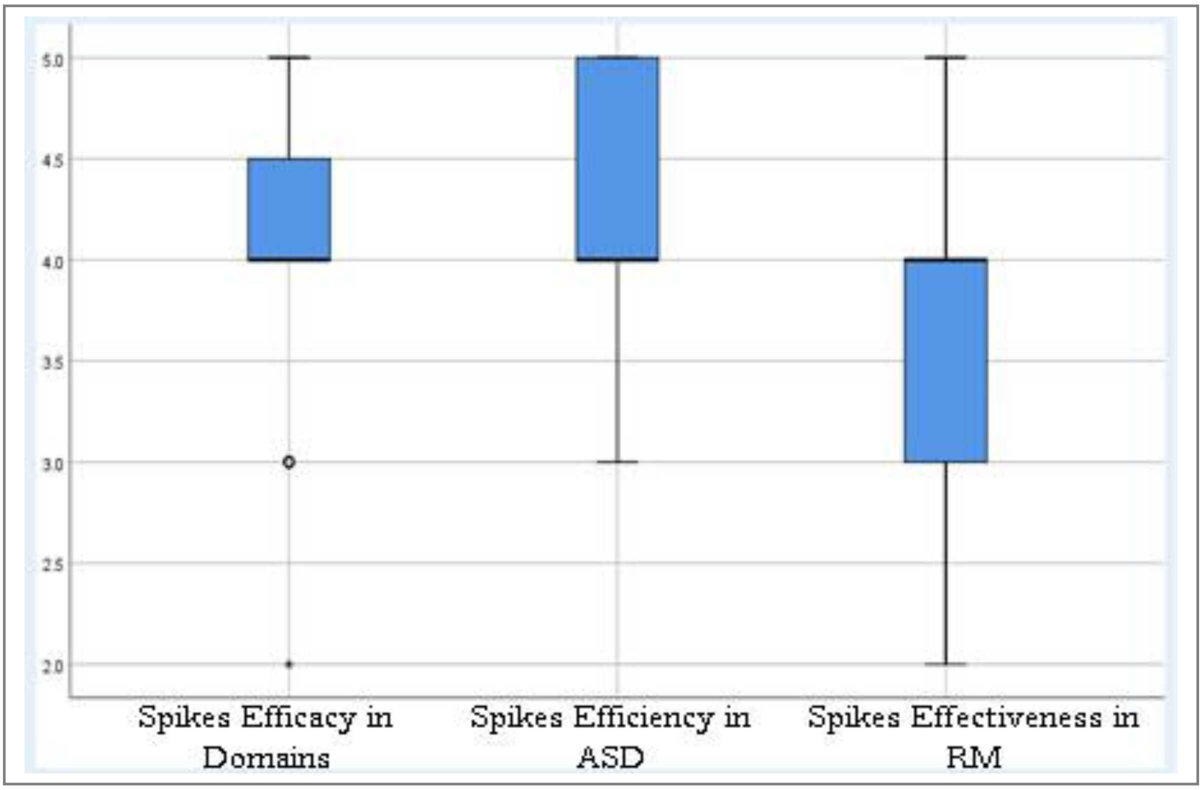


Figure 5.19: "Box and whisker" plot for efficiency and effectiveness scores of agile spikes

5.4.6. Frequency of use of spikes in agile software development (ASD)

In the questionnaire, the participants were asked to respond to the following question to ascertain how frequently they use spikes.

In your opinion, how frequently has your team or organisation utilised agile spikes to minimise risks in your software development projects?

Due to their effectiveness, spikes were reported to be used frequently in ASD to minimise risk. According to the data collected, 36 of respondents (50%) confirmed that they often used spikes in their roles in agile development to minimise risks. Nine (12.5%) asserted that they always used spikes or had seen them used in ASD. As can be seen in Figure 5.20, only five (6.94%) said that they rarely used spikes, while 22 (30.56%) stated that they used them sometimes.

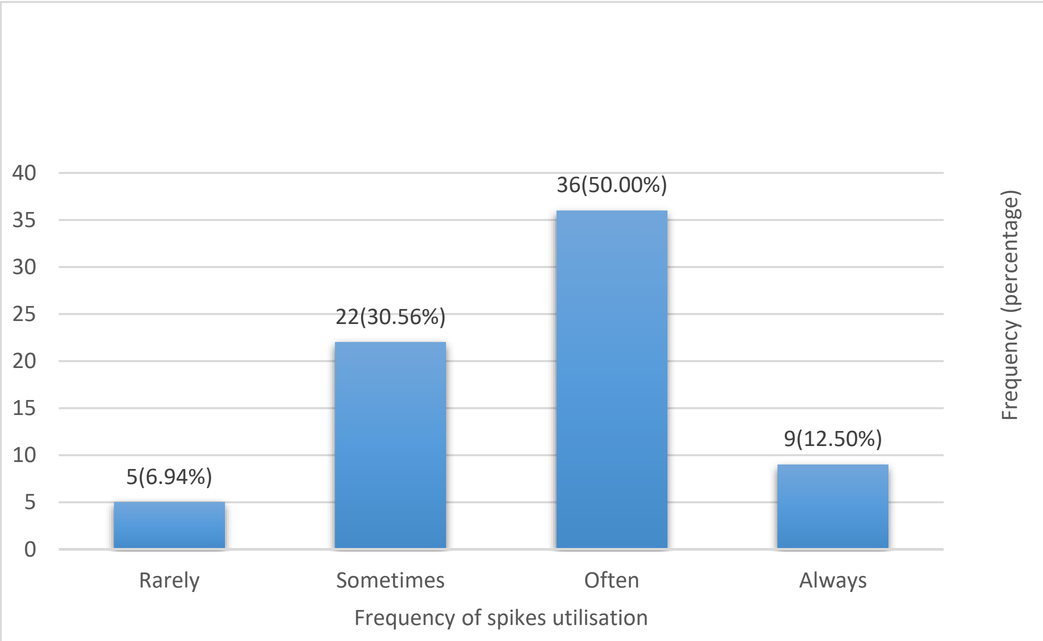


Figure 5.20: Frequency of using spikes

5.4.7. Likelihood of applying spikes to estimate user stories

To inquire further on use of spikes, the participants were asked the following question:

Is it likely that software development experts/practitioners apply spikes to estimate user stories?

The question's phrasing was meant to assess the likelihood of software development practitioners applying spikes. Since it is not guaranteed that they use spikes in their development projects, it was decided that likelihood rather than frequency is best placed to gauge the extent of using spikes in estimating user stories.

According to the responses, 34.72% (25 participants) feel that spikes are somewhat likely to be applied to estimate user stories in ASD projects. Cumulatively, 54.16% (39) mentioned that it is likely that spikes are applied in estimating user stories. However, 20.84% (15)

argued that spikes are very unlikely to be applied specifically to estimate user stories. About 25% (18 responses) were unsure if experts/practitioners are likely to use spikes in estimating user stories, as shown in Figure 5.21.

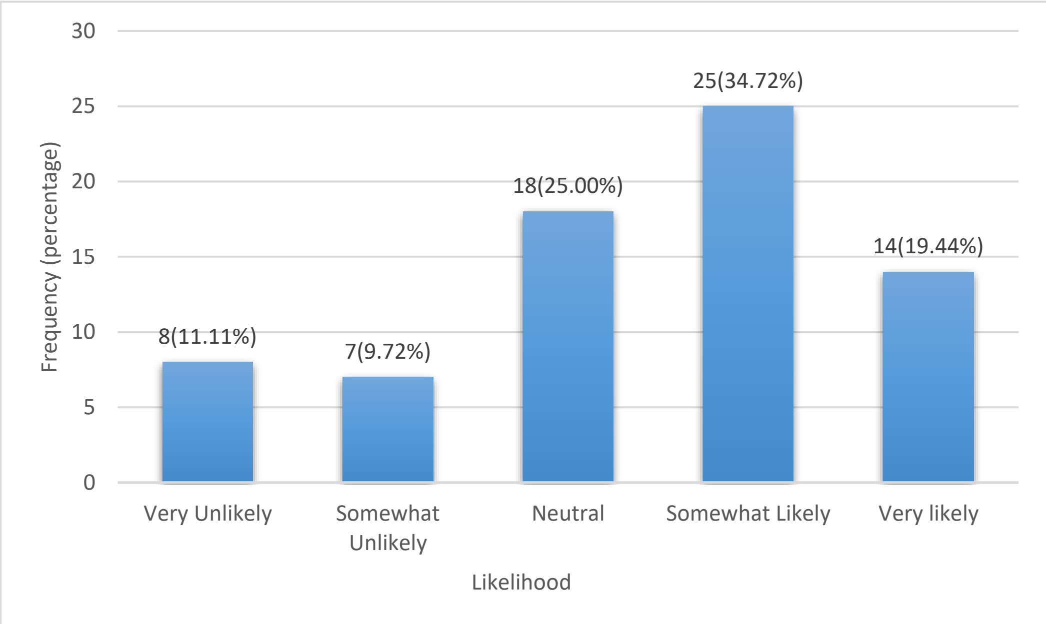


Figure 5.21: Likelihood of practitioners applying spikes in estimating user stories

5.4.8. The most effective type of spikes

The participants were asked the following questions to assess the effectiveness of spikes in risk management:

Which type of spikes do you use mostly in agile software development?
Which of the two types of spikes do you find more effective when used?

There are two types of spikes that this study seeks to explore. Technical spike was claimed by 79.17% of the 72 participants are the most common type of spikes, while functional spike was claimed by 20.83%, as presented in Table 5.14.

Table 5.14: Type of spikes most used in ASD according to participants'/practitioners' opinions

Type of spike	No. of Participants	Percentage
Technical	57	79.17
Functional	15	20.83
Totals	72	100

Regarding effectiveness, half the respondents affirmed that both technical and functional spikes are effective when utilised in ASD. However, 43.1% claimed that technical spikes were the most effective type when applied in agile development, as shown in Figure 5.22.

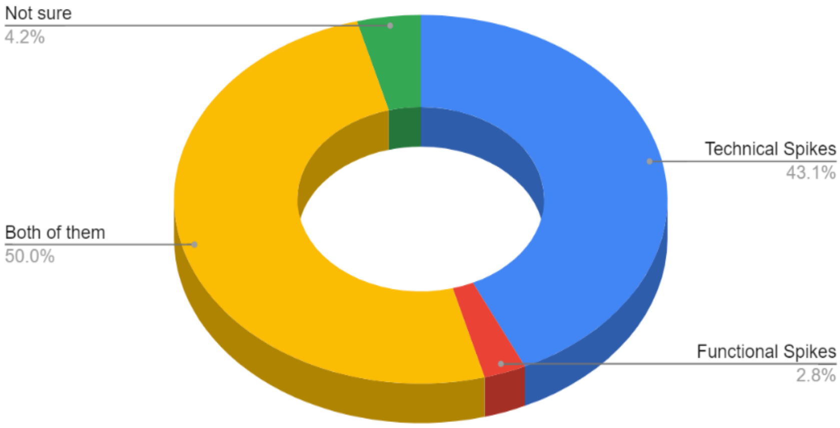


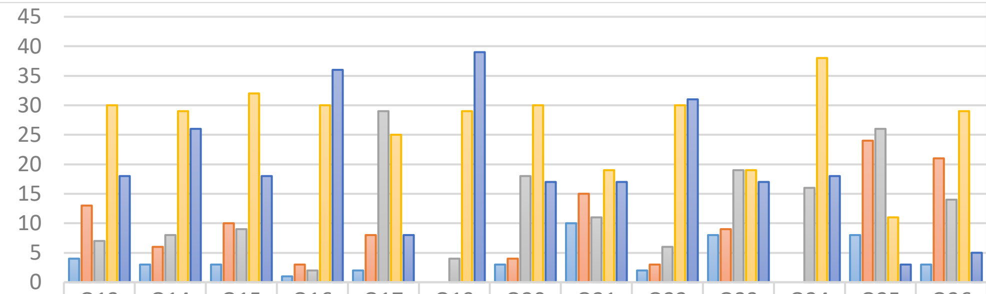
Figure 5.22: The most effective type of spikes

The results show the use of technical spikes in agile development in various domains. Although most believe both types are effective, technical spikes are more commonly used, and the response best represents it in the various domains.

For Q13 to Q26, apart from Q18, a Likert scale measured the level of participants' agreement regarding spikes' efficacy, their application's usability in estimating user stories and their overall effectiveness in risk mitigation. To all the statements measured, most respondents either agreed or strongly agreed with the claims. From the responses, 18 participants strongly agreed that the efficiency of spikes in ASD depends on the spikes applied, while 30 agreed to the same. However, 13 disagreed, and four strongly disagreed (Q13). In the same context, 26 strongly agreed that efficiency depends on the team applying the spike, while 29 agreed; six disagreed, and three strongly disagreed (Q14).

A total of 50 participants agreed that spikes sometimes do not lead to the desired solution (Q15). Only 13 disagreed, while nine were uncertain. In stating the roles of spikes, in total 66 participants agreed that they are used in prototyping, exploration, investigation, and design and research activities. Only four participants refuted this claim, while two were neutral. Based on Q17, 33 participants agreed that spikes are the best way to mitigate risk. Only ten disagreed, while 29 were neutral on the issue.

In Q19, all participants apart from four, who were neutral about the statement, agreed that spikes are effective in reducing uncertainty in ASD. More than half of the participants, 47, agreed that spikes could estimate user stories more precisely in ASD projects (Q20). Only seven disagreed, while the rest were neutral. In Q21, 36 participants agreed that spikes should be used sparingly, and 25 disagreed: the other 11 were neutral. A total of 61 agreed that spikes could be used when there is uncertainty about a process, and only five respondents disagreed. In Q23, 36 participants agreed that spikes could potentially increase risk in projects when misapplied; 17 disagreed, and 19 were neutral about the matter. More than two-thirds, 56 participants, agreed that spikes address risks and uncertainties in new systems (Q24). None disagreed, yet 16 were neutral about the matter. Regarding Q25, a total of 32 respondents disagreed that spikes are more convenient than any other approach. Only 14 individuals agreed, while 26 were indecisive. Finally, 34 participants agreed that risks in ASD can be managed without applying spikes (Q26), 24 disagreed, and 14 were neutral. Figure 5.23 is a detailed summary of responses to the questions.



	Q13	Q14	Q15	Q16	Q17	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26
Strongly Disagreee	4	3	3	1	2	0	3	10	2	8	0	8	3
Disagree	13	6	10	3	8	0	4	15	3	9	0	24	21
Neutral	7	8	9	2	29	4	18	11	6	19	16	26	14
Agree	30	29	32	30	25	29	30	19	30	19	38	11	29
Strongly Agree	18	26	18	36	8	39	17	17	31	17	18	3	5

Figure 5.23: Likert-scale item frequency summary

Of the responses on the Likert scale, 12 of the 13 questions had a mean score above 3. This implies that most of the respondents agreed with the statements. However, Q25 had a mean below 3 (M=2.68, SD=1.01), as shown in Table 5.15. This indicates that most participants disagreed with the statement that ASD projects can be managed without applying spikes. Therefore, based on the responses recorded, they feel that spikes are essential in ASD. This trend will be used as the basis for undertaking the one-sample t-test if the mean score for the 13 responses is equal to or greater than 3.0, as discussed in section 5.5.4. Since 3.0 is the mid-value of the possible score, it is used as the test value in the t-test.

Table 5.15 summarises the responses to the Likert-scale questions in the questionnaire. The question number will be used in the subsequent analysis rather than the statement.

Table 5.15: Likert items and summary statistics

No.	Item	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree	Mean	Std. Dev.
Q13	The efficiency of spikes in agile software development depends on the type of applied spikes	4	13	7	30	18	3.63	1.20
Q14	The efficiency of spikes in agile software development depends on the team applying the spikes	3	6	8	29	26	3.96	1.09
Q15	In some instances, spikes do not lead to the desired solution when applied	3	10	9	32	18	3.72	1.12
Q16	In most cases, spikes are used in prototyping, exploration, investigation, design, and research activities in agile development	1	3	2	30	36	4.35	0.84
Q17	Spikes are the best approach for risk management in agile software development	2	8	29	25	8	3.40	0.93
Q19	Spikes are effective in reducing uncertainty in agile software development	0	0	4	29	39	4.49	0.61
Q20	Spikes can estimate user stories more precisely during the software development process	3	4	18	30	17	3.75	1.02
Q21	Spikes should be used sparingly as solutions to problems since they do not yield direct value to the customers	10	15	11	19	17	3.25	1.39
Q22	Spikes can be used when uncertainty about a process, system or operation exists	2	3	6	30	31	4.18	0.95
Q23	Spikes can potentially increase risk in a project when wrongly applied	8	9	19	19	17	3.39	1.28
Q24	I find spikes to be effective in addressing software development risks and uncertainties in new systems	0	0	16	38	18	4.03	0.69
Q25	I find spikes more convenient than any other approach when estimating user stories	8	24	26	11	3	2.68	1.01
Q26	I believe that risks in agile software development projects can be managed without any application of spikes	3	21	14	29	5	3.17	1.06

5.4.9. Practitioners' perspectives on recommending spikes in agile software development (ASD)

To seek the opinions of the participants/practitioners on recommending that others use spikes, they were asked the following ensuring question:

Based on your agile role, would you advise other practitioners with a similar role to utilise spikes in their agile software development?

From the responses, 61.11% (44) claimed that they would definitely advise those in similar roles to utilise spikes. About 27.78% (20) said they would probably advise others to use spikes in ASD, while only eight said they would only possibly do so. Figure 5.24 shows that spikes' effectiveness is why most participants are willing to encourage others to use them in ASD.

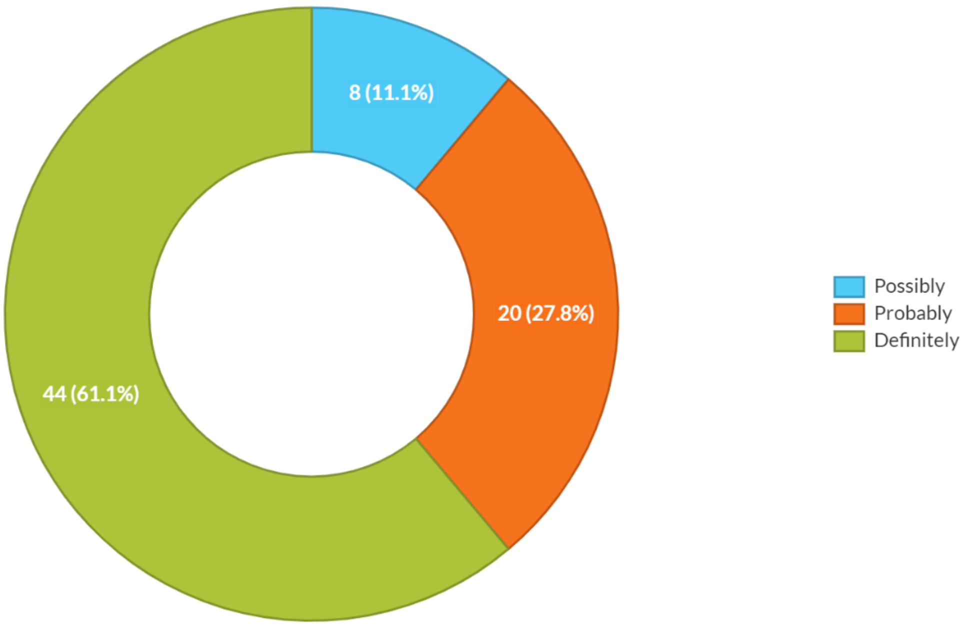


Figure 5.24: Likelihood of participants advising others to use spikes in ASD

5.5. Statistical Tests for Quantitative Analysis

The data collected were analysed to check the relationships between spike efficiency and effectiveness in minimising risk in ASD. Correlations, regressions, and t-tests were applied to synthesise the data thoroughly.

5.5.1. Reliability analysis

Reliability analysis reveals whether variables are measured adequately by the scale. Cronbach's alpha is the most efficient test to check the reliability of the measurement (Cronbach and Shavelson, 2004). A value of 0.7 is considered sufficient and implies high internal consistency. Based on an analysis of questions using a five-point Likert scale, Cronbach's alpha is 0.508 across the 13 items, implying low internal consistency, as shown in Table 5.16.

Table 5.16: Reliability analysis

Cronbach's alpha	No. of items
0.508	**13**

By contrast, the measure for scale variables – that is, spikes' effectiveness in domains, risk management and efficiency in ASD (3 items) – showed a high internal consistency, with a Cronbach alpha value of 0.778.

5.5.2. Correlation analysis

After undertaking a correlation between the efficiency of spikes in ASD and their effectiveness in specific domains, and risk management, strong positive correlations were revealed. Notably, spikes' efficacy in domains showed a statistically significant positive correlation to its effectiveness in risk management at the 99% confidence level ($r=0.356$, $p=0.002$). This implies that as the effectiveness of spikes in managing risk increases, their subsequent efficacy in software domains increases during software development increases. In the same context, the correlation analysis between spikes' efficiency in ASD and their effectiveness in risk management showed a stronger positive correlation. The relationship is statistically significant at the 99% confidence level ($r=0.413$, $p<0.01$), as presented in Table 5.17. Based on the participants' perception that as spikes continue to become more efficient, their ability to mitigate risk in software development increases.

Table 5.17: Correlation between spike efficacy in domains, risk management, and efficiency in ASD

		Spike efficacy in domains	Spike efficiency in ASD
Spike effectiveness in risk management	Pearson correlation	0.356**	0.413**
	Sig. (2-tailed)	0.002	0.000
	N	72	72

**. Correlation significant at the 0.01 level (2-tailed).

Table 5.18 depicts the correlations between the Likert-scale items in the questionnaire, based on data from all 72 participants. The highlighted values are all significant at the 95% confidence level. There are correlations between items that are rated on the Likert scale. At a 95% confidence level, there are statistically significant relationships between the participants' responses in the set of 13 questions. For instance, Q15 is positively correlated with Q13 (r=0.288, p=0.014) and Q14 (r=0.267, p=0.023).

While some are positively correlated, others have statistically significant negative associations. For instance, Q23 shows a significant negative correlation to Q19 (r=-0.283, p=0.016) and Q20 (r=-0.313, p=0.008). These statistics imply that, while most participants/practitioners find the spikes effective in resolving uncertainty during ASD (Q19), they do not find them convenient in estimating user stories (Q25). From the table, a correlation with a p-value less than 0.05 implies that the relationship, whether positive or negative, is statistically significant.

Table 5.18: Correlation matrix of Likert-scale items (Pearson's r)

	Q13	Q14	Q15	Q16	Q17	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26
Q13	1	0.202	0.288*	0.130	0.313**	0.157	0.428**	0.259*	-0.136	-0.050	0.097	0.202	0.039
Q14	0.202	1	0.267*	0.138	0.169	-0.075	0.142	0.248*	-0.033	0.232*	-0.017	0.065	0.127
Q15	0.288*	0.267*	1	-0.106	-0.013	-0.006	-0.161	0.163	-0.084	0.145	-0.154	-0.080	0.337**
Q16	0.130	0.138	-0.106	1	0.071	-0.004	0.119	-0.039	-0.097	0.108	0.032	0.000	-0.066
Q17	0.313**	0.169	-0.013	0.071	1	0.273*	0.346**	-0.003	0.060	-0.109	0.355**	0.396**	0.074
Q19	0.157	-0.075	-0.006	-0.004	0.273*	1	0.383**	-0.230	0.285*	-0.283*	0.405**	0.190	-0.018
Q20	0.428**	0.142	-0.161	0.119	0.346**	0.383**	1	0.045	0.018	-0.313*	0.370**	0.293*	-0.117
Q21	0.259*	0.248*	0.163	-0.039	-0.003	-0.230	0.045	1	0.019	0.292*	-0.066	0.038	-0.019
Q22	-0.136	-0.033	-0.084	-0.097	0.060	0.285*	0.018	0.019	1	0.011	0.248*	-0.203	-0.086
Q23	-0.050	0.232*	0.145	0.108	-0.109	-0.283*	-0.313**	0.292*	0.011	1	-0.314**	-0.066	0.210
Q24	0.097	-0.017	-0.154	0.032	0.355**	0.405**	0.370**	-0.066	0.248*	-0.314**	1	0.378**	-0.217
Q25	0.202	0.065	-0.080	0.000	0.396**	0.190	0.293*	0.038	-0.203	-0.066	0.378**	1	0.011
Q26	0.039	0.127	0.337**	-0.066	0.074	-0.018	-0.117	-0.019	-0.086	0.210	-0.217	0.011	1

** Correlation significant at the 0.01 level. * Correlation significant at the 0.05 level

5.5.3. Regression analysis

Based on the correlation analysis, both dimensions (efficacy and efficiency) of spike reveal a significant positive correlation with effectiveness in risk management, as evident in questions (Q3, Q4, Q7, Q10, Q11, and Q13 to Q29). Therefore, it is possible to predict how, collectively, the two dimensions relate to the effectiveness of spikes in risk management. After performing a regression analysis between these variables, the model was found to be significant at the 95% confidence level (F=7.114, df= 2, p=0.002), as shown in Table 5.19, which presents the ANOVA results.

Table 5.19: Analysis of variance

ANOVA[a]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	**Regression**	6.069	2	3.034	7.114	0.002[b]
	Residual	29.431	69	0.427		
	Total	35.500	71			

a. Dependent variable: spike effectiveness in risk management
b. Predictors: (Constant), spike efficiency in agile development, spike efficacy in domains

The results imply that the model is adequate and can accurately predict whether the application of spikes is effective in risk management based on its efficacy in domains. Furthermore, the model summary (see Table 5.20) shows an R-value of 0.413 and an R-squared value of 0.171, implying that the independent variables (efficiencies) are strongly and positively related to the dependent variable (spike effectiveness) and that 17.1% of the variation in the effectiveness of spikes in risk management is explained by their influence on efficacy in software domains and efficiency in ASD.

Table 5.20: Model summary

Model	R	R squared	Adjusted R squared	Std. error of the estimate
1	0.413[a]	0.171	0.147	0.653

a. Predictors: (Constant), spike efficiency in agile development, spike efficacy in domains
b. Dependent variable: spike effectiveness in risk management

The regression analysis results showcase a scenario where the development teams can benefit from spikes being effective in specific software domains. When such effectiveness is obtained, the spikes become effective in helping the teams to mitigate risks in general. On the other hand, efficiency is the extent to which a spike is completed in the least amount of time, while effectiveness is the extent to which a spike achieves its desired outcome. In order for both efficiency and effectiveness to be achieved, it is essential to have a clear understanding of the goal and to plan the process accordingly. Thus, applying the regression analysis helps uncover the connection between the efficiency of spikes in ASD and its effectiveness in risk management.

5.5.4. One-sample t-test

Applying the t-test to check whether the responses of the Likert-scale questions scored more than 3/5 revealed that the mean response to most questions was indeed higher than 3 (neutral). This is true apart from one of the items, Q25. Therefore, it indicates that most participants agreed that the application of spikes is effective in estimating user stories, resolving uncertainties in ASD, and improving risk management during software development. All the responses to questions Q13 to Q17 showed statistical significance when the means were tested to ascertain whether they are above 3, using a one-tailed t-test. The hypotheses tested in these cases were as follows:

> H_o: The mean response of the variable is less or equal to 3 (H_0: $\mu \leq 3.0$)
>
> H_a: The mean response of the variable is greater than 3 (H_1: $\mu > 3.0$)

However, this test did not incorporate Bonferroni correction since the data are from one sample and there are no multiple relationships being studied. Ideally, Bonferroni correction is applied when (i) comparing different groups at baseline, (iii) studying the associations between variables, and (iii) when examining more than one endpoint in trials (Armstrong, 2014). The one-sample t-test does not fit in any of these three scenarios and therefore Bonferroni correction was not applied.

Based on the statistics presented in Table 5.21, there is sufficient evidence to reject the null hypothesis and conclude that the mean responses of the variables (Q13-Q17) are above 3 at the 95% confidence level ($p<0.05$). For instance, in the mean response to (Q13), the efficiency of spikes in ASD depends on the type of applied spikes is greater than 3 ($t=4.406$, $df=71$, $p<0.001$). By contrast, the mean responses to Q21 and Q26 showed a lack of statistical significance to reject the null hypothesis, as shown in Table 5.21. For instance, the mean response to Q21 is not greater than 3 ($t=1.524$, $df=71$, $p=0.132$). Hence, the null hypothesis cannot be rejected.

Table 5.21: Results of t-test for Likert-scale questions

	One-Sample t-Test Test Value = 3					
	t	df	p-value	Mean difference	95% CI Lower	95% CI Upper
Q13	4.406	71	0.000	0.625	0.34	0.91
Q14	7.437	71	0.000	0.958	0.70	1.22
Q15	5.491	71	0.000	0.722	0.46	0.98
Q16	13.579	71	0.000	1.347	1.15	1.55
Q17	3.678	71	0.000	0.403	0.18	0.62
Q19	20.844	71	0.000	1.486	1.34	1.63
Q20	6.255	71	0.000	0.750	0.51	0.99
Q21	1.524	71	0.132	0.250	-0.08	0.58
Q22	10.498	71	0.000	1.181	0.96	1.40
Q23	2.569	71	0.012	0.389	0.09	0.69
Q24	12.613	71	0.000	1.028	0.87	1.19
Q25	2.698	71	0.009	-0.319	-0.56	-0.08
Q26	1.332	71	0.187	0.167	-0.08	0.42

5.5.5. Univariate ANOVA

On performing a univariate ANOVA on the use of spikes in risk mitigation during software development and its effectiveness in risk, there were statistically significant groups (levels of agreement) ($F(1, 4)=2.927$, $p=0.027$), as shown in Table 5.22. A post hoc test using the Bonferroni correction revealed that the use of spikes in solving uncertainties elicits a slight difference in the agreement of participants on the efficacy of spikes in risk management, from disagreeing to strongly agreeing (3.5 ± 0.926 vs 4.50 ± 0.535) respectively, which is statistically significant ($p=0.04$), as shown in Appendix I. The Bonferroni correction is mainly used to avoid committing a Type 1 error, rejecting a true null hypothesis (Shaffer, 1995). Furthermore, Cohen's d (partial eta squared) is small ($d=0.149$). According to Cohen (2007), a small Cohen d value indicates that the difference is trivial. In this case, the Cohen d value (partial eta squared) measures the degree of the effect size of the use of spikes in ASD to its effectiveness in managing the risks. As established by Cohen, any d-value equal to below 0.2 is considered negligible or "small" and does not show any variation in the dependent variable (in this case, spike effectiveness) that is explained or accounted for by the independent variable (spikes use in risk mitigation). Therefore, it can be concluded that there is a difference between the two levels of agreement on using spikes as the best approach to mitigate risks in ASD.

Table 5.22: Test of difference between responses

Tests of Between-Subject Effects Dependent variable: Spike effectiveness in risk management						
Source	**Type III sum of squares**	**df**	**Mean square**	**F**	**Sig.**	**Partial eta squared**
Corrected model	5.281[a]	4	1.320	2.927	0.027	0.149
Intercept	453.337	1	453.337	1005.129	0.000	0.938
Q17	5.281	4	1.320	2.927	0.027	0.149
Error	30.219	67	0.451			
Total	1140.000	72				
Corrected total	35.500	71				

[a] R-squared = 0.149 (adjusted R-squared = 0.098).

5.5.6. Scatter plot

A positive relationship was established in assessing participants' experience of spikes and agile development. As seen in Figure 5.25, the relationship showed a strong (almost perfect) relationship between the two variables. Based on the plot, the greater the experience a person has in agile development, the greater their expertise in spikes in ASD. Therefore, it can be concluded that from the participants' opinions, the experience of using spikes grows exponentially with increasing agile experience. Thus, practitioners with extensive experience in agile methods are likely to be adept at using spikes.

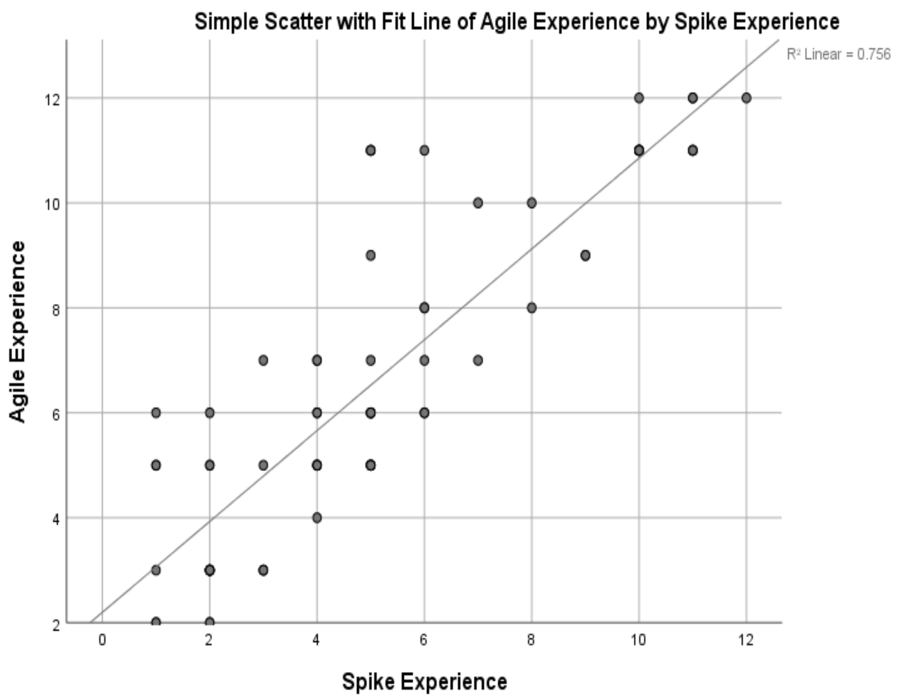


Figure 5.25: Scatter plot of spike experience vs. agile experience

5.6. Qualitative Analysis of Questionnaires

This section presents the results of the open-ended questions, where the participants were asked to give their own opinion or rationale for their answers. The section touches especially on roles, types, and efficacy of spikes in ASD.

5.6.1. Practitioners' opinions of the roles of spikes in ASD

At the end of the questionnaire, the participants were asked the following question:

Apart from earning some knowledge about a specific topic, what other roles do spikes play in agile software development?

While some were unsure of what other roles spikes play, others mentioned various functions, including:

- Helping in better understanding requirements.
- Driving a risk-aware culture, embedding more collaboration with accountability.
- Helping in estimation and risk management.
- Exposing complexity/simplification.
- Providing familiarisation among team members.
- Experimenting with an approach to delivery of a specific product or story to the client.
- Solving technical debts.

Responses provided by participants included:

"It allows team members to explore new ideas and sparks creativity when used properly, but of course, it depends on how much liberty the team has in such things". In addition, participants argued: "Spikes can improve developers' domain and architecture knowledge and improve team members' connections". According to one participant, spikes create an opportunity for team members to grow together. In particular, the participant said:

"…when team members work on the same spikes in parallel and when spikes are properly documented, they can allow team members to grow together, have better feature insight and learn how to improve their cooperation and communication".

5.6.2. Practitioners' perspectives on technical and functional spikes

As shown in Figure 5.22, the participants were asked which type of spike is most effective: technical or functional. To provide a more detailed response, the following question was asked.

What is the rationale for your selection?

The majority said both, while a few sided with technical spikes. Their rationale varied according to their response. For those who stated that both types are effective, their main reason was that either of the two is helpful in ASD when addressing either technical or functionality uncertainties in a software project. One of the responses was: “Spikes are there to answer complex questions on unknowns or seek clarity, and it is neither exclusively in tech nor function that these questions arise”.

For those who stated that technical spikes are more effective, the rationale revolved around usefulness in providing solutions. One of the respondents stated: “Technical spikes are often needed where POs cannot address the requirements of system components or underlying architecture, so the technical spikes help identify potential solutions and solicit feedback from technical stakeholders like architects, networking, etc. Functional spikes aren’t needed quite as often, as some up-front work can be done at the BA or PO level. They can still be useful, but in our org, technical spikes are more common”. In addition, a participant asserted that: “Technical spikes can be used to determine architecture changes that are necessary prior to the start of development. Therefore, it does not slow down or even block development effort”. In the same context, a participant opined that technical spike could be used to design parts of a system: “Technical spikes can be used to design parts of the system - meaning the team have more of an understanding of what they are building when refining Use Stories. Plus, it gives the development team the technical acceptance criteria for the solution”.

Some who perceived technical spikes to be effective mentioned that the PO can instead address functional spikes through user stories across sprints. Furthermore, other methods such as mock testing, MVP, A/B testing, and usability testing can be used when addressing functionality concerns.

For those who said that functional spikes are more effective when used, some responses pointed out that functional issues have less clarity than technical issues. When this type of spikes is applied, it clarifies the problem and allows teams to continue with the development. Expressly, the respondent stated: "Functionality can have less clarity than technical issues, so functional spikes are of greater value".

5.6.3. Practitioners' opinions of efficient use of spikes in agile software development (ASD)

To obtain participants' opinions on the efficiency of spikes, they were asked the following question:

In your opinion, how can agile spikes be used efficiently in software development?

The responses pointed to various suggestions. Common among them were:

- By frequently inspecting the goal and adaptation of the spike in ASD.
- By experimentation and research.
- By linking spikes to a defined business outcome.
- By using them for upcoming planning tasks.
- By trying to keep them at a minimum towards the end of the project and at a maximum at the beginning.
- When there is a fixed scope and objectives.
- By doing a spike in a time-boxed period.
- When they are refined just like any other work so that they have clear, measurable objectives.
- By using agile spikes only when needed to answer unknowns.

Some of the most interesting responses were as follows:

"Try to keep them to a minimum, but more of them at the beginning of the product. They should be used to increase the knowledge inside the team. Time-boxed rather than outcome-oriented. These would be the top recommendations". another said, "We should potentially create a spike in the backlog, and each spike should have some acceptance criteria like other user stories". Other responses included:

> "For our teams, spikes are used most efficiently when you set fixed scope and objective. It helps us remove the level of uncertainty when developing new user stories (features)".

> "We are a way to create time to properly scope out future work before adding it to a sprint. In this way, we used spikes to minimise surprises which might impact our

ability to complete a sprint. It often worked well, but of course there are always things that cannot be foreseen".

"Getting knowledge that can speed up development or help in decision-making. Which is not present at the moment you plan the spike".

"Spikes are about learning. I often challenge developers to do a spike for a time-boxed period (say 1-3 days). After that, the developers will present what they have learned; if this is sufficient for making a technical decision, then the spike is finished; if more time is required to come to the necessary learning, then we decide whether it is worth continuing the spike or not".

5.6.4. Practitioners' perspectives on the use of spikes in estimating user stories

In the Likert-scale questions, the participants were asked to respond to the statement that agile spikes are more convenient when estimating user stories. To follow up on the responses, the participants were further asked to provide the rationale for their responses. For varied reasons, the majority disagreed, as shown in Table 5.15. Among the common reasons, a participant stated: "You can use the spike results to estimate stories, but it's not the most convenient. The most convenient is to use the current knowledge of the team rather than a spike. If the estimate is too big and slicing can't occur effectively, then a spike should help, but it's the third option for us". Another participant, who disagreed with the statement, mentioned: "I do not totally agree with the statement since we should not hold spike as a tool for estimating stories. Yes, it could help provide insight into a solution or mitigate risk, which would help with better estimation. But using spikes directly as an approach, I don't think that is right". Other responses by participants who disagreed were along the following lines:

- Not every question needs a spike; sometimes a grooming session with stakeholders can answer the questions.
- The bucket system works quite well in estimating user stories.
- There are other alternative techniques, and the use of spikes depends on the context and situation.
- They are more for addressing uncertainty. Sometimes risks are obvious and using spikes to determine them is not necessary.
- Spikes should only be used when stories are too large, with many complexities.

From the few participants who agreed that spikes are convenient came varying responses closely related to the usefulness of spikes in gathering information. One stated: "The spike allows the team to dedicate time to understanding the level of complexity for a story without being concerned about delivering a potentially shippable increment". Another of the same opinion asserted: "Spikes reduce the unknown and help the team to estimate with more confidence". Lastly, another said that "using one or more spikes to answer specific questions about one or more other user stories (or use cases or....) helps to ensure that the answers are achieved when they are needed, or that delays become visible to stakeholders (such as the team, product manager, etc.)".

5.6.5. Practitioners' perspectives on the feasibility of risk management without the use of spikes during agile software development.

The final Likert-scale question, Q26, sought the opinions of participants on whether risks in ASD can be managed without the application of spikes. They were asked to agree or disagree with the statement and provide their rationale for the response. Based on the answers provided, most participants agreed with the statement, and only 24 disagreed, as shown in Table 5.15 above. Those who agreed that risks could be managed without using spikes gave various reasons. One of the participants mentioned: "When the risk is related to the release deadline, we do not have enough time for spikes and having a lot of spikes can even enhance the risk by slowing down the development process and delaying the deployment to the client. On the other side, when we have a flexible deadline, and we are able to focus on quality, spikes can mitigate or even eliminate risks in the software development process". A participant/practitioner with a similar opinion on spikes and risk management in ASD asserted: "If a PO/stakeholder does all the necessary research before writing the user story, then there is no need for the development staff to use spikes to estimate those stories because there should be no outstanding questions".

Other responses included as rationales for agreeing with the statements were as follows:

- Spikes are not the only way of mitigating risks in ASD.
- It is possible to push story point estimates higher to capture risk, which leads to fewer stories being taken into a sprint.
- Sometimes there is limited time to apply spikes, and too many can slow the development process, increasing the risk.
- Better application of scrum in the agile environment can make spikes unnecessary.

Although most participants agreed that projects could be completed without the use of spikes, others disagreed. One participant in particular stated: "Without a spike, the team leaves themselves exposed to prolonging the time spent on a specific story without getting close to a solution". Another mentioned: "Sometimes, no matter how experienced the team is or how easy the project is to implement, there are requirements for which a technical solution is not easy to decide. In this case, a spike is an easy way to overcome the problem”. Finally, a participant stated that a “spike is the best option to take time to work on identified risk”.

5.7. Discussion

The chapter presented the findings from participant interviews and questionnaires. The interviews and questionnaires with the participants provided detailed answers to the first three research questions through the first phase of this study (see figure 3.2). This section discusses the most significant findings of both practitioners’ interviews and surveys. Additionally, it discusses the threat to validity due to the selection of participants.

5.7.1. Discussion of the interview findings

Five themes were identified as addressing the research questions based on the interview questions. These themes are based on participant responses that point to specific issues. Despite the diversity of the responses, a transcript synthesis was carried out in order to construct common themes supported by the responses. Some responses are included with the results to highlight and validate those themes. The selection of the participants used purposive and snowballing sampling, yet the representation was from seven countries. Spikes continue to play a significant role in software development projects across all domains. Based on the findings, the primary functions of spikes in ASD relate to estimation, risk management, research and design, exploration, and decision-making. Based on the interviews, most practitioners believe that spikes are most effective in their roles of reducing uncertainty and complexity in software projects and in estimating user stories, also in the exploration of new technologies. Through these roles, spikes remain one of the preferred agile solutions development teams try to incorporate into their ASD plans. Although some participants admitted that it is unnecessary and should be used only as a last resort, the technique helps most teams uncover unknown uncertainties that might jeopardize a software development project’s progress later. The responses of the 22 participants illustrate that spikes are efficient in ASD (M=4.31, SD=0.63) on a five-point scale. The efficiency measure was based on participants' perceptions and experiences of spikes. The mean score indicates that the spikes are partially efficient in agile software projects. Similarly, they are partially effective in reducing risk and software domains (M=4.14, SD=0.76).

In software development projects, various issues can trigger uncertainty. According to the results of the interviews, in ASD projects, unclear user or stakeholder requirements, unfamiliar technologies, project complexity, and poorly written stories are the primary causes. If the development team is uncertain about the project’s requirements, the final product may not satisfy all the client’s needs. The inclusion of unfamiliar requirements makes it challenging for agile teams to make informed decisions that are pivotal to maintaining the quality of the end product.

Based on participants’ responses, it is apparent that the effectiveness of spikes in ASD increases when the development team works together to better understand the problem and develop an effective solution. This includes having a shared understanding of the problem and the goals, breaking down the problem into smaller pieces that can be tackled independently, and setting a clear timeline for the spike. Additionally, utilising tools such as user stories, estimation techniques, and retrospective meetings can help the team to be better equipped to make decisions and complete the spike on time. On the other hand, the efficiency of spikes increases as a team becomes more experienced in using the technique. Agile spikes allow teams to quickly test and validate ideas and hypotheses, which helps to reduce time wasted on ineffective approaches. Additionally, as teams become more familiar with the process, they are better able to identify and address potential problems ahead of time, leading to fewer costly missteps.

5.7.2. Discussion of the questionnaire findings

The questionnaire was to extensively investigate the RQ1, RQ2, and RQ3 and identify whether the efficiency and efficacy of spikes were statistically significant. Below, the discussion focuses on the findings of questionnaires obtained during the first phase of this study.

The questionnaire was designed to ask open-ended questions to gather more information on spikes' roles in ASD. As reported in the results, most roles mentioned are similar to those cited by the participants interviewed in this study. The key roles revealed in the questionnaire centred on estimation, investigation, experimentation, and risk mitigation.

In particular, spikes are essential in prototyping and investigating the dynamics of a new system or technology being applied in a project.

Negative options were not offered for some questionnaire items since they would need to be reverse coded before being analysed. That is, "definitely not" would be coded as 1, while "possibly not" would be coded as 3. However, when the items are positive, there is no need for reverse coding, and the coding process follows the usual pattern in terms of high to low. Furthermore, reverse coding negative items would not yield any more significant impact regarding the results since the potential response trend would be maintained through the coding procedure. Therefore, negative options were unnecessary as they would not have added any meaningful value because of the reverse coding procedure (Chyung, Barkin, and Shamsy, 2018).

The usefulness of spike efficiency and effectiveness are core reasons why professionals in software development find them suitable. As the participants pointed out in the questionnaire, the roles of spikes in reducing risk, estimating user stories and research are the primary reasons they are used in software development projects. On average, participants had an experience of over six years in ASD and over five years in using agile spikes in their various agile roles. The information that they provided is relative to their experience in the field. The results of the questionnaire addressed RQ1, RQ2 and RQ3 in detail, providing both quantitative statistics and qualitative information to answer the research questions succinctly.

The correlation has shown that the precision of spikes in estimating user stories (Q20) is positively correlated to their efficiency in ASD (Q13) ($r=0.43$, $p<0.001$). Furthermore, the correlation shows that precision positively correlates to effectiveness in risk management (Q17), as shown in Table 5.18. Since spikes have been found to be effective in various domains and efficient in ASD, their effectiveness in risk management has been reported to be high. At the 99% confidence level, the efficacy of spikes in domains was found to be significantly correlated to their effectiveness in risk mitigation ($r=0.36$, $p<0.001$). Similarly, spikes' efficiency positively influences their risk management effectiveness ($r=0.41$, $p<0.001$).

The regression analysis showed that the model developed is adequate to predict whether the efficacy of spikes in domains and efficiency in ASD can tell if spikes are effective in mitigating any uncertainty ($F(2, 71) = 7.11$, $p<0.05$). Based on the five-point scale for the effectiveness of spikes in risk management, it was revealed that most participants/practitioners find the spike technique effective, giving an average score of 3.92/5. From this result, it is apparent that the effectiveness of spikes depends on how effective they are in domains and their efficiency in ASD. Although the variable may be a standalone item, it has been established by the questionnaire data that the three items have significant positive interrelationships.

Although the agile teams had diverse roles, most interview and questionnaire respondents were Scrum Masters. While this does not precisely reflect the makeup of agile teams, Scrum Masters still outnumbered any other agile roles based on the organisations from which the participants were recruited. Thus, it is not possible to assert with certainty that the other groups were under-represented in this research. The sampling technique used was convenience sampling. Only those participants who agreed to be part of the study were included, regardless of their roles. Coincidentally, the majority were Scrum Masters. It would have been unethical to drop some of the responses just to balance the number of participants from each group. The sampling procedure paved the way for inequality in the representation of groups, but this was beyond the researcher's control.

5.7.3. Threat to validity

As with all empirical research, there are several threats to validity due to the selection of participants based on the interviews and questionnaires conducted. These threats include selection bias and self-selection bias. Selection bias occurs when the sample of participants is not representative of the population of interest, meaning that the results of the study may not be generalisable beyond the research participants. Self-selection bias occurs when participants can choose whether or not to take part in the study, and those who do so may be significantly different from those who do not. These biases can lead to inaccurate results, as the research participants may not represent the population of interest. Selection bias is a potential threat to the validity of this study since the participants in this study were self-selected. It is possible that those who chose to participate were already more likely to have positive opinions about agile spikes, which would skew the results of this study. Another potential problem is that those who did not choose to participate may have completely different opinions about agile spikes than those who did, which could potentially influence the results. The same applies to participants in the second and third phases of this study.

5.8. Summary

This chapter provides an analysis of quantitative and qualitative results and a discussion of the interviews and questionnaires adopted by the study. The interview method focused on answering three research questions through the responses of the 22 practitioners, whereas the questionnaire answered the same questions through a survey of 72 participants. The roles, efficacy and efficiency of spikes in ASD projects were reported and discussed. Furthermore, various agile methods were evaluated, whereby spikes were applied. The scrum was reported to be the most widely used of all the methods. Key among the results reported in the chapter are the efficiency and effectiveness scores of agile spikes. Supported by the literature review, the findings show that spikes effectively reduce risk and enhance the overall usability of ASD processes. The chapter describes the various domains in which spikes are used. In general, all ASD projects, regardless of the domain, can use agile spikes. The results show that despite the many causes of uncertainty that may introduce risk in an ASD project, the careful application of spikes can help teams investigate, gather more information, or develop prototypes to understand how the end product should look. Broadly, this chapter has compiled the opinions of agile development teams to describe how spikes are used, their roles, efficiency, and effectiveness in ASD.

Chapter 6: Preparation for Identifying Common Success Factors in the Use of Spikes and Case Studies

Previous chapters of this study have covered interviews conducted with 22 agile practitioners from various countries and organisations around the world, as well as a survey of 72 agile practitioners. The study has sought to determine the roles of spikes in agile methods, and how they can be used efficiently and effectively to manage risks in ASD projects. The study findings provide significant information addressing the research questions, but they need to be validated through case studies to ascertain whether the agile practitioners' opinions reflect what is happening in organisations and the software industry. This chapter highlights the design and preparation for investigating the common success factors in spikes (RQ4) and the case studies that were undertaken to validate the information obtained concerning RQ1, RQ2, and RQ3. Furthermore, the findings related to RQ4 were validated through case studies. The chapter comprises seven main sections describing the study design for identifying the common success factors in applying spikes and the approach to the case studies in terms of the objectives, questions, methods, processes, scenarios, and ethical approval.

6.1. Identifying Common Success Factors in Applying Spikes

Determining the common success factors in using spikes would significantly help in effective application and spotting glitches that may be caused by a lack of familiarity with them, which in turn will help in exploring uncertainty and risks associated with software projects. The following sub-sections set out the methods and approaches that used to answer RQ4 in this study, identifying the common success factors in the application of spikes.

6.1.1. Interviews and focus groups

Interviews are the preferred approach when undertaking qualitative research. Both semi-structured interviews and focus groups can be utilised. According to Archibald et al. (2019), video conferencing is one of the best approaches for undertaking interviews if the sample population covers a large area. Since this study involved participants from all over the world, semi-structured interviews and focus groups were used.

Piloting was initially thought to be associated only with quantitative research. However, its importance has also been recognised in qualitative research, and researchers are finding it important to test their structured or semi-structured interview protocols before starting to collect data (Majid et al., 2017). To meet the needs of this study, the interview questions were piloted with two PhD students and four practitioners in agile (one PO and three software developers). The outcome of the piloting showed that the methods were reliable in collecting the information needed from the participants. As stated in section 5.2, the researcher used triangulation to validate the thematic analysis and support the same findings, which is the process of gathering multiple sources of evidence to support a conclusion. This can include collecting data from multiple sources, such as interviews, surveys, documents, and observations; comparing the results of different methods; and having multiple researchers analyse the data independently. In our case, involving multiple researchers was challenging as the research was conducted by a single researcher studying for the award of a PhD. Moreover, the cost and time associated with obtaining multiple researchers to validate the thematic analysis can be prohibitive.

6.1.2. Questionnaire

In addition to the qualitative approach, the study includes a quantitative aspect (reported in section 7.3), administering a questionnaire with a mixture of closed-ended and open-ended questions to selected participants. The questionnaire was completed entirely online to ensure that participants from around the world could participate in the study.

When undertaking a quantitative study, it is necessary to test the research instrument (Majid et al., 2017). Thus, it was critical to test the questionnaire to ensure its reliability in collecting the information required. Two PhD holders, two PhD students, and three agile practitioners (one PO and two software developers) participated. They completed the questionnaire and were given the opportunity to provide feedback on the design of the questionnaire and the clarity of the questions, reporting any ambiguity in the questions. The results of the piloting verified the dependability of the questionnaire before using it in the main study to collect data from agile practitioners or experts.

6.2. Overview of the Case Study Approach

Case studies in software engineering concentrate on specific phenomena, systems, or organisations. They are forms of field studies that take an exploratory approach and provide limited modifications to the environment being studied (Stol and Fitzgerald, 2015). Furthermore, case studies provide researchers with the opportunity to undertake practical tests of assumptions or theories by examining the practices of software developers, the functionalities of the development process, and the techniques used in the field (Runeson et al., 2012). To comprehend the software engineering context fully, researchers then need to scale up initial case studies to include multiple large-scale cases. The primary goal of scaling up is to broaden the scope of the research and improve the validity of the empirical data and study conclusions, which are inherently limited to their contexts (Runeson et al., 2012; Stol and Fitzgerald, 2015). The approach aims to provide extensive empirical data based on which inferences about the real-world software engineering context can be made. The case studies in this research (reported in Chapter 8) comprise an empirical investigation carried out to validate the established roles of spikes in agile methods, and their efficiency and effectiveness in risk management, as well as to establish the most common factors that may help practitioners improve their agile process, particularly when using spikes. Notably, the case studies aim to reveal how common success factors aid in the effective application of spikes in software development within organisations.

Case studies, according to Yadav et al. (2007), are the best approach for answering the 'how' and 'what' questions in research. Moreover, Yin (2014) proposed that the case study approach is appropriate when the phenomenon being studied is not clearly or sufficiently theorised. There are two types of case study design: embedded and holistic (Yin, 2014). The context of the case study determines which of the two is most appropriate to explore a particular phenomenon.

6.2.1. Embedded case study approach

When a case study analysis focuses on multiple sub-units, the embedded case study approach is appropriate, assessing the subject matter through different units of analysis. In this case, both qualitative and quantitative analyses are supported. The embedded case study approach is useful not only for putting all the units of analysis into perspective as one case, but also for confronting rival interpretations (Yin, 2014). Researchers can use this method to ask multiple research questions requiring quantitative and qualitative responses. Ideally, they will conduct 'embedded' studies under the main case study. The main case study in this research concerns investigating the use of spikes in ASD. This then includes multiple sub-units exploring the role of spikes, their efficiency and effectiveness, and the success factors that enhance their application. These sub-units are intended to provide sufficient data to inform the primary case study as a whole (Scholz and Tietje, 2002).

6.2.2. Holistic case study approach

In the holistic case study approach, the case itself is the only unit of analysis in which the researcher is interested. In most cases, a systematic approach is taken to the phenomenon being studied (Yin, 2014). However, the approach can also involve multiple cases, each with a single unit of analysis. In the context of this research, it would be a holistic study if the three units of analysis presented in Figure 6.1 examined separate contexts or if the case only aimed to evaluate the role of spikes in ASD. However, the research design incorporates more than one unit of analysis, as shown in Figure 6.1, and thus the case study is not holistic in nature.

Using the embedded case study approach makes it possible to avoid the lengthy procedures associated with a holistic approach. Furthermore, the embedded case study approach does not require the investigator to access the software projects to measure the aspects being studied (Scholz and Tietje, 2002). It enables the researcher to make inferences about the units of analysis based on reported and observed data, making it the most appropriate approach for this research.

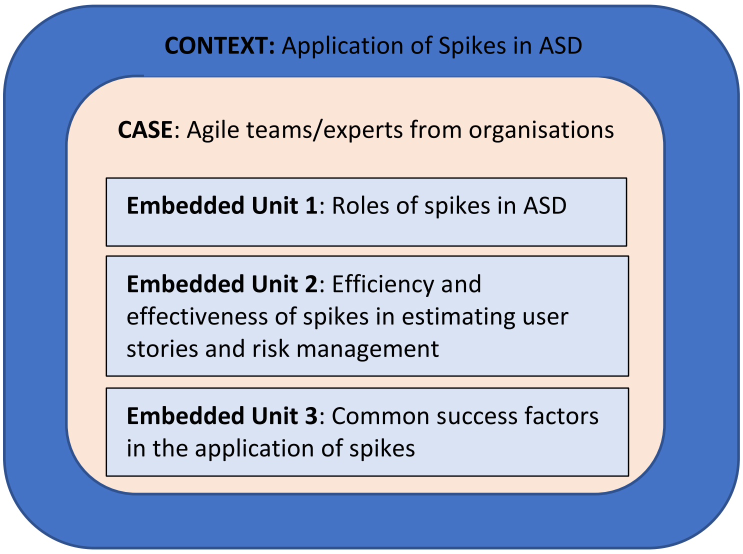


Figure 6.1: Embedded case study approach

6.3. Case Study Objectives

Case studies do not have a solid theoretical basis, and it is necessary to define specific objectives to ensure meaningful results are obtained (Kitchenham et al., 1995). The objectives define researchers' focus on studying a particular phenomenon (Hancock and Algozzine, 2016). In most cases, such studies are used to go beyond reported information and demonstrate a real-world situation based on empirical evidence from surveys or thematic analysis. Like other study designs, case studies are confined to the set objectives that define and set the basis for the research. As stated in section 6.2, the primary goal of this case study is to verify and authenticate the information developed in response to the research questions. In this research, the case studies had the following objectives:

- Examining the reasons why organisations use spikes in their agile processes.
- Examining how development teams assess risks and use spikes to mitigate them.
- Examine the effectiveness of spikes in an industrial context.
- Demonstrating the validity of the success factors developed in relation to RQ4 and their application in various organisations.

These objectives were achieved through direct and indirect interaction with development teams. Interaction with the product owner and development teams shed more light on these objectives when access to project documentation was not feasible.

6.4. Case Study Questions

Research questions are helpful in giving a case study the direction it needs to attain better and more valuable results (Lethbridge et al., 2005). The data gathered from practitioners and observations when spikes are employed in projects (previous or ongoing), as well as during the focus groups, were used to answer the following broad questions in this case study.

- Has the team used spikes on any of your recently completed projects?
- What were the spikes supposed to achieve? Did they meet their goals?
- Does your team use spikes as a risk management technique?
- Have the spikes you have used been effective? How?
- Do you use spikes during sprint retrospectives?
- Do you consider any particular factors when you are going to use spikes? If so, what?
- What success factors do you or your organisation consider when using spikes in software development?

Table 6.1 highlights the questions that led to the various stages and methods in data collection and the persons involved.

Table 6.1: Summary of case study questions

Case Study Questions	
Method of data collection	**Participants involved**
Direct: Focus groups with 2–4 agile practitioners (Online interviews)	**Researcher/PO/PM/development team**
1) Is it possible to solve technical and functional risks using architecture, design, and solution spikes? Please provide details on your answer.	
2) What roles have spikes played in past projects completed by your organisation?	
3) How effective were the spikes in fulfilling their roles?	
4) How effective were the spikes in managing risks?	
5) Are there specific skills that development teams need to apply spikes properly? If yes, please state them.	
6) Can you explain the objective(s) of the spikes used in your recent software project?	
7) Did the spikes achieve the objective(s)?	
8) How many spikes did you use in each sprint/iteration for the recent project?	
9) How many spikes did you or your team employ in total for the recent project?	
10) Did you use spikes during the sprint retrospective?	
11) What are the factors that most help in the effective application of spikes?	
12) Does your organisation use other risk management techniques? If yes, please state them.	
13) Are spikes used to carry out roles specified in project planning?	
14) What are the greatest challenges that may hinder the proper application of spikes?	
Indirect (online observation)	**Researcher**
1) Are there specific roles for spikes in the company's projects?	
2) How effective have spikes been in estimation and risk management in the company's past projects?	
3) Does the company have specific factors it considers in enhancing the application of spikes in ASD projects? What are they?	
4) How do they compare with those identified in RQ4?	
5) Have the spikes fulfilled the roles they are expected to perform?	
6) How does this compare to the findings of RQ1?	
7) What make the spikes more effective?	

6.5. Case Study Methods

This section considers data collection for the case studies in terms of the process, data selection, and the units of analysis on which the study has been conducted.

6.5.1. Data collection methods

In most cases, the data collection method is influenced by the source of the data (Runeson et al., 2012). In this study, the data was obtained from software development teams/practitioners in various organisations and their documents concerning their past projects. Lethbridge et al. (2005) define three data collection methods commonly used in software engineering, comprising direct, indirect, and independent approaches. The direct approach primarily involves interviews and focus groups, whereas the indirect approach entails obtaining raw data without making inquiries of practitioners, mostly through observation. Finally, the independent approach involves document analysis (Lethbridge et al., 2005), but this is not always possible due to privacy and confidentiality concerns on the part of participants.

In this study, both direct and indirect approaches were used to collect data from the agile practitioners and through observation. The focus groups provided various forms of data, as discussed in section 6.6.2. Furthermore, the indirect approach was used when the agreed procedure was employed in an organisation's previous or existing project. Here, observation was the best approach to utilise while collecting the information. As mentioned in section 6.6, The observations were conducted online and in an overt approach, with the objectives of data collection known to those observed (Nrskov and Rask, 2011). The observations entail observing the practices and behaviours of the practitioners who engaged in the case studies. However, the researcher was observing specific items during the observation as listed in Table 6.1. The online observations were carried out during some development sessions the researcher was allowed to attend with the development teams, as discussed in chapter 8. Observation in this study was useful in collecting information on which factors might ensure the successful application of spikes in executing the roles intended and effectively managing project risks, validating the findings related to RQs.

6.5.2. Data selection

Data were acquired using two methods, as stated in section 6.5.1. The data used were chosen based on practitioners' advice and past projects completed by the organisation or the development team. During the focus groups, the practitioners present were asked questions as a group, seeking their individual perspectives before reaching a consensus. The information gathered at this stage was used to determine when and how spikes might be integrated into the development process to achieve the case study objectives. The focus group participants contributed information based on their experiences with employing spikes in software development.

In addition, the case studies adopted an indirect approach based on observation, aiming to gather information in real-time when the development team uses spikes in ongoing projects. The data collection took approximately 60 days. There was sufficient time to select all valuable data from the observation, particularly regarding success factors and the effectiveness of spike use. Based on the approved estimates/assumptions from the focus group meetings described in section 6.6.2, only data addressing the three embedded units were selected and analysed. Moreover, during the focus groups, observations were conducted to determine the roles of spikes in software development projects.

6.5.3. Units of analysis

When conducting a case study, an essential factor is a context, which serves as the foundation for the entire study. Cruzes et al. (2015) recommend breaking down a case study concept into units when it is too complex to analyse holistically. The units of analysis thus formed assist researchers in determining which aspects of the context to focus on when conducting the case study. This approach also streamlines the creation of study objectives based on the units to be analysed (Hancock and Algozzine, 2016). In this research, the context concerns the application of spikes in ASD by organisations or freelance software development teams.

Identifying the units of analysis is critical in any study since they define the bounds of the research, encompassing what is analysed (Barquero et al., 2019). They explicitly outline what should be researched based on the primary unit depending on the context of the study, as well as determining from whom/where the information will be gathered. In this context, the unit of analysis can be individuals, groups, artefacts (e.g., documents), geographical units, or procedures.

In this study, the primary unit of analysis comprises individuals and software development groups/teams from organisations. Due to COVID-19 restrictions, meetings with these groups were conducted virtually rather than in person. The context of the case study is the application of spikes in ASD. Within the main unit, three embedded units were analysed, as set out in the case study design discussed in section 6.2.2 and presented in Figure 6.1, focusing on the following:

- The role of spikes in ASD projects.
- The efficiency of spikes in estimating user stories and their effectiveness in risk management.
- The most common success factors that can enhance the effective application of spikes.

To ensure the value of the case study results, agile teams/experts were selected based on certain criteria, as follows:

- They have used spikes in some of their previous projects/software development.
- The team in charge of the agile projects is well-versed in the use of spikes.
- The projects using spikes are on software development.
- The company/development team have experience in applying agile methods in their projects.

6.6. Case Study Process

Since the case studies were based on agile practitioners interviewed virtually (in focus groups or individually), a systematic approach to gathering information for each unit of analysis is required. In this regard, the case study procedures followed a five-step process with each of the agile teams/practitioners chosen, as shown in figure 6.2, to ensure that valuable information is obtained to authenticate the findings from RQ1 to RQ4.

There was only one option in executing the case study process due to the COVID-19 pandemic, and most software organisations opted to work remotely. Namely, the focus groups and interviews were virtual since face-to-face meetings were not possible. These virtual meetings with software development teams/practitioners were only executed following mutual agreement between myself as the researcher and the teams/ practitioners. Ideally, the case studies were implemented entirely online.

The emergence of COVID-19 has forced many researchers to redesign their data collection approach and shift to online-based methods. For a study that initially planned to use in-person observation or interviews, it is more complex to gather the necessary data virtually. The problem deepens when technical information is required. In this case study, data regarding the application of spikes in ASD needs to be collected from agile teams. However, the entire process was online to conform to the pandemic regulations. Nørskov and Rask (2011) pointed out that observations can be overt or covert. In overt observation, the researcher makes the objectives known, and the team is aware of the observer's intentions. In contrast, in covert observations, the researcher acts as an insider, and the team is unaware that they are being observed. An overt approach was used in this case study, and the teams involved were aware of the research objectives.

In addition, online methods were used to collect pre-existing data (collecting materials without involving participants directly) or to elicit data from participants via interviews, focus groups, and online observations (Salmons, 2015). Since this research involves interactions with the participants at various points throughout the case studies, the approach is one of eliciting data.

Various methods other than interviews can be used to collect the data required in a case study, including questionnaires, documents, and observations. Questionnaires involve using a structured form containing questions that aim to collect relevant information from the population or sample of the population being studied (Yin, 2014). In addition, relevant documents can be used to provide the data needed. This involves using documentation from an archive of the organisation or team being studied to gather information. Finally, observations are also used in case studies to collect information. In this approach, the researcher focuses on assessing certain aspects, behaviours, or processes of interest and documenting the findings from the observations. However, these approaches have various shortcomings. For instance, none of them offers the opportunity to follow up and gather more evident data. Furthermore, questionnaires tend to have a lower response rate than interviews. Moreover, travel restrictions were imposed due to the pandemic, and it would not be easy to visit the teams physically to collect their documents. Accordingly, virtual interviews were selected as the optimal and most appropriate approach for collecting data in the case studies.

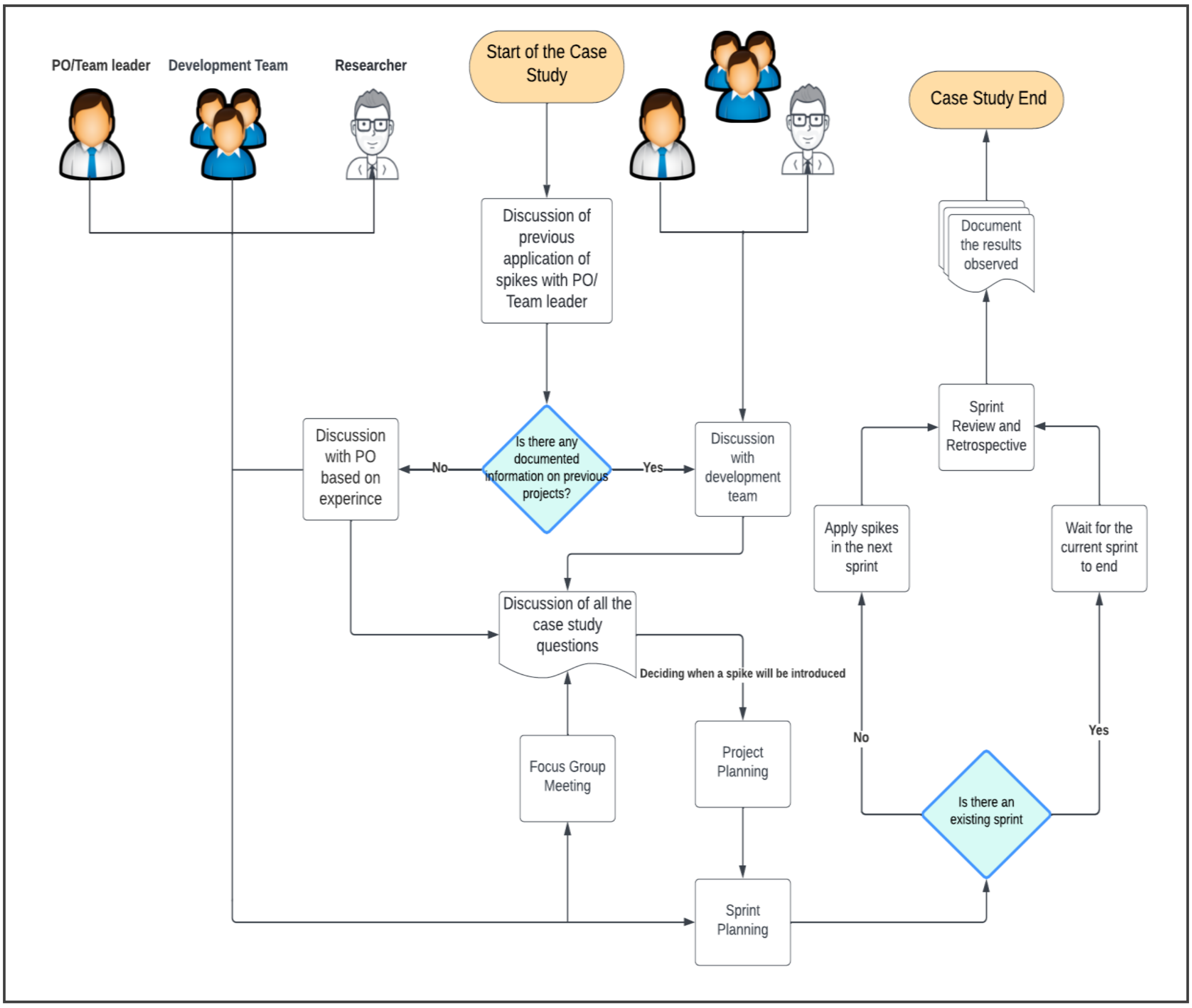


Figure 6.2: Case study process scenarios

6.6.1. Step 1: Preliminary evaluation of past projects

A quick evaluation of recent projects is necessary to gain insight into the application of spikes in the organisations' ASD projects. This step is intended to establish the three embedded units for the specific organisation and guide future discussions. At the beginning of the case study, the following aspects of previous projects were discussed with the PO or project team leader:

- The roles that spikes have played in their past agile software development projects.
- Historical accuracy and efficiency indices of spikes in the previous projects.
- From the documents, the researcher and the PO or PM discussed the success factors considered for the effective use of spikes.
- The researcher and the PO/PM discussed the success factors considered for the effective application.
- How the spikes contributed to the reduction of risk in their projects.

If no previous data or documentation are available, the PM provides details concerning aspects of the projects based on his/her experience. Furthermore, the PO/PM was asked to invite team members with knowledge of the specific aspects being discussed to take part. The consultation with the team members is intended to ensure the development team provides reliable and accurate information.

6.6.2. Step 2: Focus group meetings

This step entails holding a focus group with the agile practitioners and the project owner/team leader to discuss the roles of spikes, their efficiency and effectiveness in reducing risks, and the common success factors that improve the application of spikes in software development. The focus groups were held online, with 2–4 agile practitioners from various software organisations participating. A key consideration was that each focus group included practitioners from the same organisation working as a team to ensure consistency in the information provided and to make it easier to interview them all simultaneously. The discussion was based on previous projects completed by the organisation using spikes and agile processes. The focus group results identified the estimates for the three units of analysis, practically applied in previous or ongoing projects undertaken by the organisation or development team. There were two possibilities based on the previous projects completed by the team.

The following were discussed at the meeting if spikes already have set up roles in the organisation's projects, with established effectiveness scores and success factors:

- How the spikes have achieved the roles they were intended to fulfil.
- If the current efficiency and effectiveness scores of the spikes in risk management are satisfactory.
- How success factors enhance the application of spikes in ASD projects to attain satisfactory results.

If the development team/experts do not have any specific insights or information regarding the three units of analysis, the discussion was centred on whether or not the organisation/development team uses spikes to fulfil the roles previously identified (RQ1), and to address efficiencies (RQ2) and effectiveness (RQ3), as well as the impact of the factors identified in the findings related to RQ4 on the successful application of spikes.

6.6.3. Step 3: Case study planning

Following the focus group meeting, the PO/team leader and I discussed the process/procedure for implementing spikes during software development to support the discussions that took place during the focus group sessions. Incorporating spikes in the development process demonstrated their roles in risk management, estimating user stories, efforts, and delivery time. The success factors were also assessed to check whether they helped in the effective application of spikes. If the development team is engaged in an existing project, the current sprint/iteration would be completed before applying the agreed process in the next sprint. At this point, the researcher was an observer in some stages of the development process since the study was undertaken online.

If the team/practitioner was beginning a new software development project, spikes would be applied during the sprint/iteration to address any uncertainties or complexities. Specific development processes and the application of spikes were monitored to establish whether there are consistencies with or differences from the results for RQ1, RQ2, RQ3, and RQ4. Understandably, the development teams/experts may not wish to provide some data for privacy and confidentiality reasons. Therefore, the study relied primarily on the information collected through the semi-structured interviews and focus groups, which were undertaken at a mutually convenient time.

6.6.4. Step 4: Sprint planning and the application of spikes

The development teams carried out sprint planning based on the product backlog. During the team's development process, the procedure agreed upon with the PO/team leader in step 3 was implemented. A portion of the process was observed online in order to gather more information necessary to obtain valid and valuable results. The sprint was launched after the sprint planning was completed and a sprint backlog was identified. Only after the sprint was completed was the application of spikes based on the three embedded units evaluated.

6.6.5. Step 5: Sprint review and retrospective

At the end of the sprint, a virtual sprint review was held with the development team and the PO/team leader. They showcase what they accomplished in the previous sprint, and there was a particular focus on the following aspects:

- Obtaining actual data regarding the three units of analysis.

- Comparing the actual results with those proposed during the focus groups/individual interviews with practitioners who took part in the study.
- Developing a list of the success factors that helped the development teams effectively apply spikes to obtain the desired outcomes or to mitigate or eliminate the uncertainty.
- Comparing the findings to those obtained for RQ1 to RQ4.

Following the review, a sprint retrospective was conducted to generate ideas on how the process could be improved. The Scrum Master guided the team in brainstorming ideas for what they should begin, continue, or discontinue doing in the next sprint. At this point, my role was solely that of the observer to gain further information about the use of spikes in ASD. Figure 6.2 depicts the entire process of these steps.

6.7. Ethical Considerations

When dealing with human subjects, ethical considerations must be addressed. It is necessary to ensure the process respects the rights of the participants and that ethical standards are upheld throughout the process. This case study is designed to be ethical. The prospective participants in the focus groups and interviews were given a consent form, and only those who agreed to participate were included in the case studies. Furthermore, approval was sought from the organisations before carrying out the case studies. An assurance was given that all the information provided would be confidential and only be used for the study. All data obtained were stored in a password-secured computer and discarded after the study was completed.

As an institutional requirement, approval for this study was obtained from the Ethics Committee of the University of Southampton (Reference No. 62395, December 11, 2020). The approval indicates that all ethical standards have been satisfied regarding the materials presented to the committee for review and gives authorisation for the study to proceed.

Chapter 7: Common Spike Success Factors (CSSFs)

This chapter presents the results of the analysis addressing RQ4 concerning the most common factors for the successful application of spikes. The data were collected from 16 individuals and 3 focus groups consisting of 3–4 members, as shown in Figure 7.1. In all, four interview questions were used to guide the collection of the necessary information from the individuals and the focus groups. The four questions sought succinct responses from the participants, with the interviewer having the option to probe further and get more details from them. The chapter also discusses the results to answer the research question (What are the most common factors that help agile teams to use spikes successfully?) in detail.

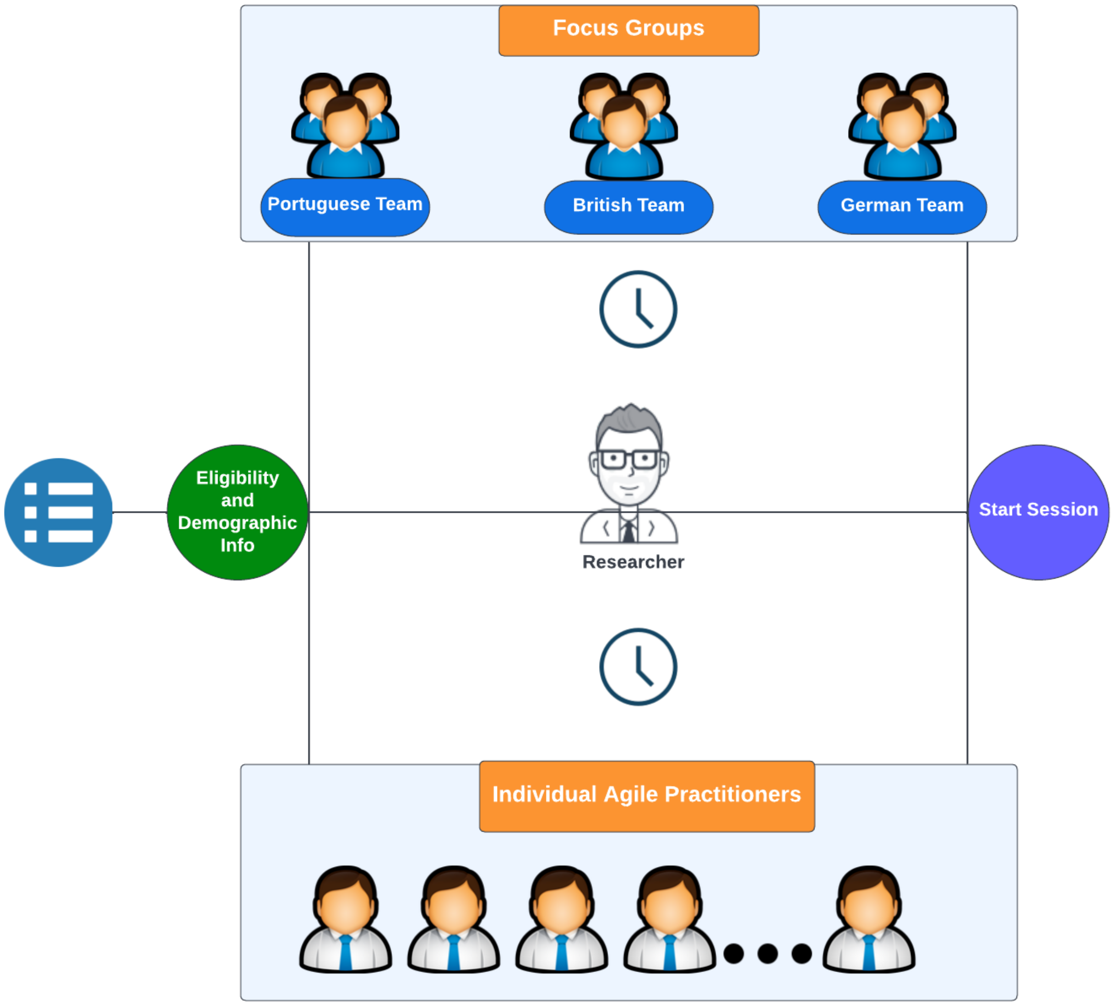


Figure 7.1: Interview process addressing RQ4

7.1. Participant Demographics for Interviews and Focus Groups

In total, 26 participants were involved in this study, including 16 in individual interviews and 10 participants in 3 focus groups. The participants who identified the common success factors in this chapter are not the same as those defined in Chapter 5, except for one participant from the Portuguese team. During the interviews, the participants were asked to state their length of experience (in years) in agile and spikes, their roles, the agile method they used, the type of organisation, and their country of origin (see Table 7.1).

Table 7.1: Demographic questions

Experience in agile?
Experience in spikes?
Agile role?
Agile method?
Organisation sector?
Organisation size?
Country?

Among the 26 participants involved in the study, 12 reported having between 16 and 20 years of experience in agile methodologies, while only 2 reported having 5–10 years of experience. As shown in Figure 9-2, 7 participants had 1–5 years of experience in agile, while 5 had between 11 and 15 years of experience.

Figure 7.2 also shows that half of the practitioners interviewed had between 1 and 5 years of experience in using spikes, and none reported having experience between 11 and 15 years. The rest, 50%, had 6–10 or 16–20 years of experience in utilising spikes. However, all the data were self-reported, and there was no way of verifying the authenticity of their experience in the different roles they fulfilled in their organisations.

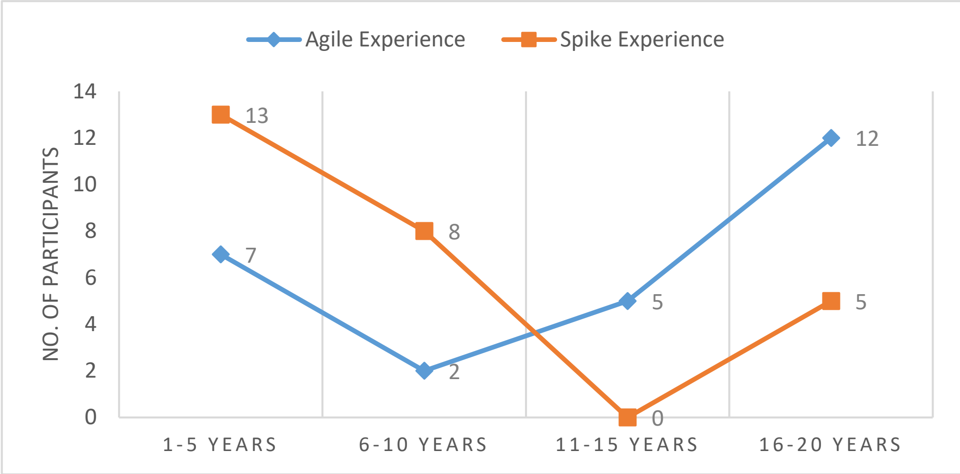


Figure 7.2: The experience in the agile and spike fields

Regarding the roles of the participants involved in the interviews, 27% of the total participants (10 participants) were developers, 27% (10 participants) were agile coaches, and 21.60% (8 participants) were Scrum Masters. As illustrated in Figure 7.3, some participants reported having multiple roles in their organisations. For instance, P15 was both a developer and an agile coach, and P6 was a developer and a Scrum Master. Other roles reported during the interview were tester (2 participants), PO (5 participants), and engineering manager (2 individuals). Whether the participant was interviewed alone or part of the focus group, their agile roles and experiences were recorded separately.

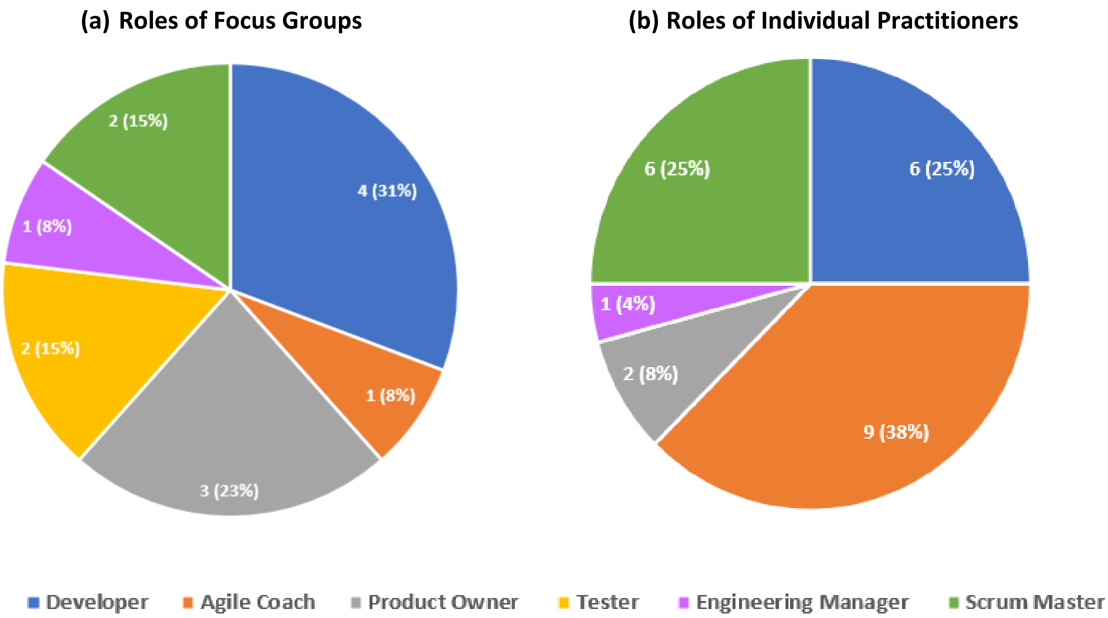


Figure 7.3: The roles of agile practitioners

With the participants playing different roles in their respective organisations or teams, their agile methods also differed. From the information collected, two major methods were common among all the 26 participants in the study. About 96% of the participants reported using Scrum as their primary agile method in software development, and only 4% (P6) reported using Kanban, as shown in Figure 7.4.

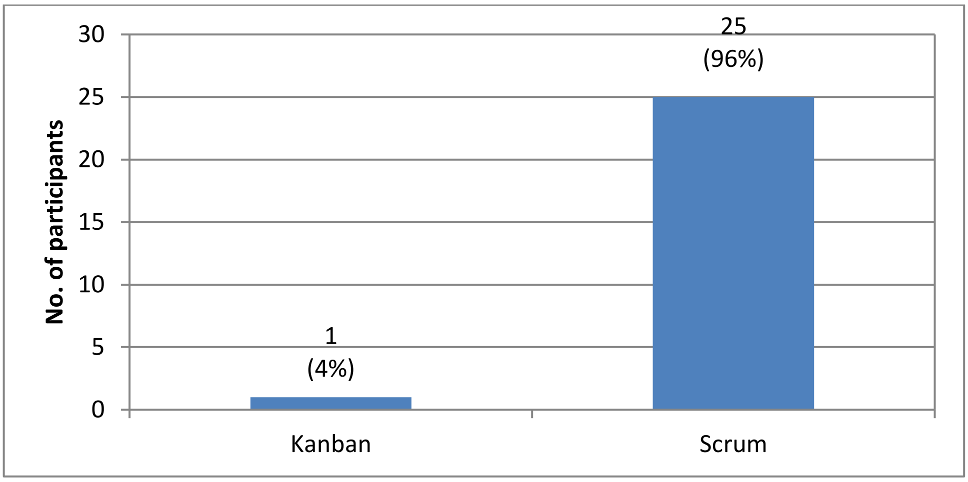


Figure 7.4: Agile methodologies used

For the sake of obtaining diverse opinions, practitioners from different sectors dealing with software development were involved in the interviews. About 75% of the participants, representing 20 practitioners, were from the IT and software development sector. Practitioners from other sectors, such as consultancy (11%), government (7%), and the energy and finance sectors (each 4%), were also interviewed, as presented in Figure 7.5. The diverse opinions of the participants from sectors other than IT and software development provide a deeper understanding of the extent to which spikes are being used in different industries when developing software.

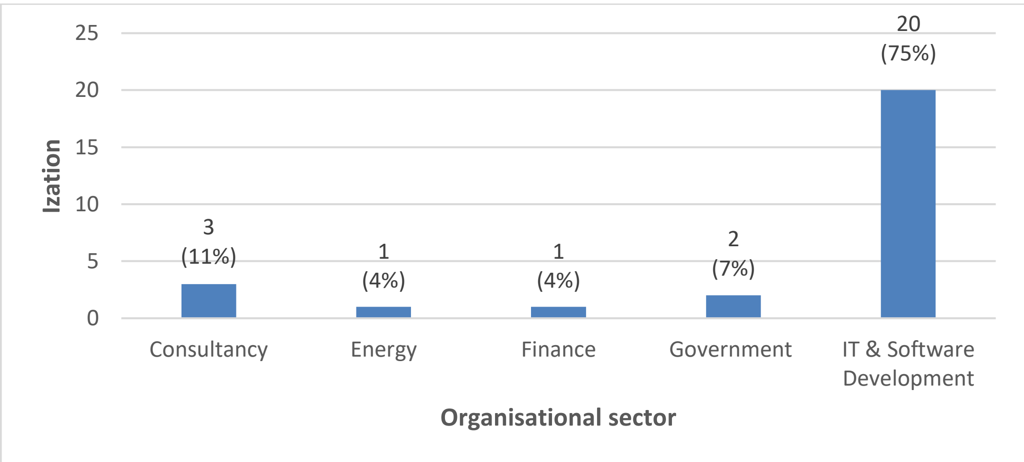


Figure 7.5: Organisational sectors of the participants

In addition, different sizes of organisations were considered when recruiting agile practitioners for the interviews. Of the individuals who participated in the study, 17 (65%) were from large organisations (more than 250 employees). This number includes seven participants in two focus groups, FG1 and FG3. Medium-sized organisations (51–250 employees) were represented by four participants, three of them from FG2, while three participants represented micro-organisations (fewer than 10 employees). The small organisations (10–50 employees) were represented by 2 participants, as shown in Figure 7.6.

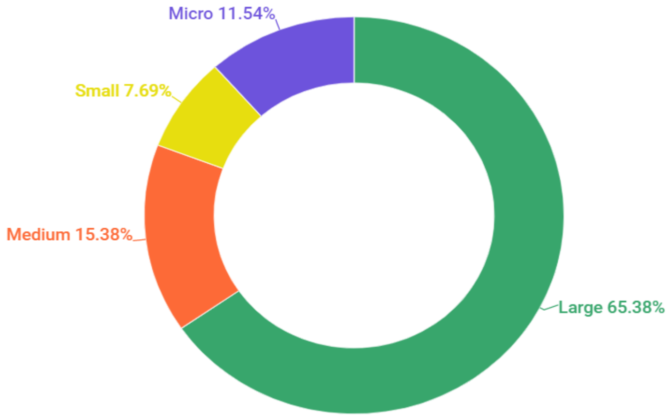


Figure 7.6: Distribution of organisation size

The agile practitioners were from countries all over the world. In the focus groups, there were 15.38% of the total participants (4 participants) from Portugal, while the UK and Germany each accounted for 11.53%. (3 participants). The Individual participants were from 10 countries distributed as shown in Figure 7.7

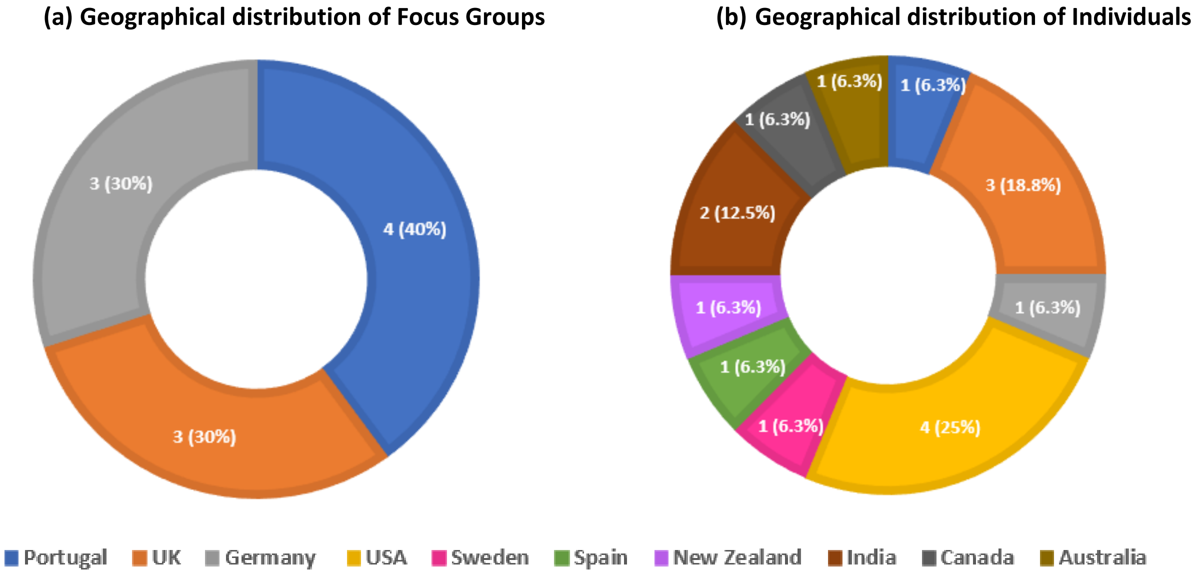


Figure 7.7: Geographical distribution of the participants (FG and Individuals)

7.2. Common Success Factors in Spike Applications (CSSFs)

A successful application of agile spike is a short period of focused work with clear objectives to answer a specific question. It is used to gain insight or knowledge to determine whether a proposed solution is feasible and viable. The outcome of a successful spike typically is a clear understanding of the problem, a potential solution, and an estimated timeline for implementation (Hunt, 2018).

The preliminary results indicate that there is no single universally agreed factor that should be considered when applying spikes in a software development process. Each development team has its own considerations, depending on the roles the spikes are intended to play, the type of software being developed, and the managerial support in place. However, the interviews revealed several categories of success factors, including people, organisational, procedural, project-related, and technical. All these categories contain different factors depending on the practitioner or team preferences.

7.2.1. Success factors based on participants' experiences

Question
1. From your experience in using spikes, have you considered some factors that may help in the effective application of spikes? If yes, please state them.

When asked this question, all participants agreed to consider some factors before using spikes, except for one (P1), who said, "No. Spike, in my opinion, is a type of work unit that helps to make work visible/transparent and helps the team focus." The participants pointed out several factors that they considered previously in their software development process.

Timeboxing

Timeboxing is the process of allocating a fixed time period within which planned activities are executed (Jalote et al., 2004). Most developers, if not all, use timeboxing as a personal time management strategy. In the interviews with individual practitioners and focus groups, 11 mentioned having considered timeboxing when they applied spikes in their software development processes. Notably, P6 stated that “A timebox for how long to spend answering the question is what helps the successful application of spike”. The same considerations were referenced by P4, P7, P9, and P10. Their answer to this question collectively illustrates that timeboxing is an important consideration when a developer needs to apply spikes in the software development process. Further, according to P11, a timebox is essential to ensure the spikes do not run over multiple sprints. A similar response was given by P7, who said that "Timeboxing is essential to ensure the spike does not continue forever".

In the focus groups, at least one group member mentioned timeboxing as an essential factor when applying spikes. For instance, FG3-3 (third member in focus group 3) mentioned that "to be successfully applied, spikes should always have a time limit". A member of focus group 1 and 2 members of focus group 2 reported similarly. FG2-2 said that "A short-time span is needed if a spike is to be successfully applied". FG2-3 added "timeboxing the spike is important for effective application". These suggestions by the focus groups further cement the importance of considering timeboxing as a key determinant of the successful application of spikes when developing software. Regardless of the role, each spike is intended to be achieved through, which is essential to ensure meaningful results and that it does not run over multiple sprints, as alluded to by P7 and P11.

Setting specific goals and expectations

Setting goals and expectations during a software development project aligns the team in working toward fulfilling the requirements within the budget and time allocated. When applying spikes, this procedure is equally important as it ensures the development team designs specific spikes for the intended task.

In response to the question, 8 participants pointed out that having a clear and specific goal or expectation is fundamental in spike application. Notably, P6 responded that "A spike should answer a specific question, e.g., how can we best integrate the ABC system? Or will event sourcing allow our application to scale by adding nodes to our cluster?". This illustrates that for a spike to be successfully applied, its goal in the process and what it is expected to deliver should be known. Only then can the team know if the spike has successfully fulfilled the goal it was intended to accomplish. As presented by FG2-3, "expected tangible output defined upfront, such as a document of recommendations, is needed for the effective application of spikes". This consideration was further mentioned by another member, FG1-2, who stated that having a clear purpose for the spike will help the successful application.

Addressing the same issue, P6 succinctly explained the need for clear goals and expectations when applying a spike in the software development process. This practitioner mentioned that “The goal must be described with a yes/no question with the outcome of the spike being to answer the question posed, generally owned/completed by one person. An example: Can we use Sockets to provide real-time updating of status on our main webpage through a browser?”. By setting up such a question before applying a spike, the expectation and goal are defined. If it provides the answer to the question, its application is considered successful. In a similar vein, FG2-2 mentioned that for spikes to be applied successfully, there should be a specific focus, i.e., solving a given problem, not a general issue. Overall, the eight members noted that in their past experience of spikes, having clear goals and expectations was necessary for their successful application.

Documentation

The participants' perspectives on agile spike documentation are divided into two aspects. The first is about the spikes, and the second is about what can support the spikes. They are addressed in the following two paragraphs.

As part of the factors considered for the successful application of spikes, documentation was found to be a common aspect among 4 of the 26 members. P5 for instance, stated that “to succeed in applying spikes, some kind of documentation should be produced in the end”. The documentation is meant to provide a history of the spike application and the results obtained. In support of this statement, one member in a focus group discussion pointed out the same consideration. FG2-1 said that “technical documentation is needed to give the best chance to the team picking up the spike and to produce the most value from the spike to inform the next steps and wider solution”. In his response, the participant added that teams rely on documentation to pick the best procedure in applying spikes for the same or similar problems. Thus, documented information is vital to apply spikes effectively.

Another agile practitioner who mentioned documentation was FG3-1, who stated that “irrespective of the result, we would have thorough and proper documentation of the spike so that we can succeed in applying spikes”. This sentiment adds to the points made by the other participants suggesting that documentation plays a critical role for spikes to be applied and has a significant impact. Moreover, as mentioned by FG3-3, the documentation should comprise two parts: "the intent of the spike and the reason for it, and the result of the spike. This can be done in code if you have a small paragraph and a demo or code result". With such documentation, it is easier to track whether the spike was successfully applied or failed to meet the goals or expectations of the team.

Providing clarity and spike creation for the unknown

As discussed with the participants, the creation of spikes to address an unknown issue needs to be clear to the development team. According to most focus group members, spikes are used for a specific purpose or role in the development process. Four participants alluded to this factor as part of the consideration for the successful application of spikes. FG2-3 stated that "specifying anything out of scope will lead to the unsuccessful application of spikes". In addition, FG1-3 pointed out that "spikes are used to reduce risk and try to discover the unknowns". As a result, the spikes applied should be able to discover the unknown to be considered successful in the development process. Although different teams might consider other approaches to discovering unknowns, P16 said that "creating spikes for every unknown will help spike application to be effective". Similarly, P7 suggested that the "The success of the spikes application can be increased if they are developed to investigate unknown (uncertainty) in order to be clarified". Thus, it is essential for software development teams to be clear on which unknowns the spikes are meant to mitigate or discover in the ASD process.

Enabling clear communication

According to some participants, communicating critical information to the rest of the development team when spikes are being implemented is vital. In particular, P3 said that "those leading the spike need to communicate clearly to get the information they need and provide the information necessary to complete the tasks resulting from the spike". Such communication is not only about the details of the spike but the rationale for applying the spike, the role it is supposed to play, and the expected outcomes of its application in the process. In this regard, P10 noted that team members should have conversations concerning whether or not spikes are needed. These conversations are essential in bringing the entire team on board when applying spikes to ensure their execution during a sprint is known by everyone in the team. The proposition that clear communication needs to be considered when using spikes for a successful result to be obtained was supported by three participants.

Setting specific goals and expectations for agile spikes differs from enabling clear communication among the development team members in that it focuses on the desired outcome of the project, while communication focuses on the collaborative process of working together (Hunt, 2018).

Validating the application of the spikes

When applying spikes, their proper use is paramount for the successful attainment of the roles they are intended to fulfil. According to FG1-1, there needs to be a discussion between all the team members to identify if the creation of the spike makes sense. This procedure is vital in ensuring spikes are used for the right reasons. Further, P2 stated that spikes need to be used for the right context; a random application of spikes may not yield satisfactory or meaningful results for the team. According to P15, spikes are helpful in addressing unknowns. In particular, the participants stated that when there is insufficient evidence available to resolve contradictory hypotheses or simply inadequate information to make an informed decision on investment, risk assessment is carried out to ascertain the likely impact of each decision. When the risk is unknown or highly uncertain, a spike is useful. In short, the "definition of ready" for a spike is "We don't know enough to make a decision", the "definition of done" for a spike is "We can now make a decision that we're satisfied with". In these responses, the three participants suggested that using spikes for a justifiable course is a factor to consider when a developer intends to employ them.

Creating a motivated or passionate team

The factor of motivation was mentioned by two participants in responding to the question. The morale and motivation of the team define how well they craft the spike before applying it. Without motivation and the drive to use spikes to discover unknowns, it is challenging to realise their successful execution. According to P7, the team's capacity and motivation to take up the responsibility for developing and executing spikes is a critical factor that defines how successful the application of the spikes will be. Further, FG2-2 stated that "people with a passion for the problem are needed for successful application of spikes". Without the commitment of the team, it might be difficult to apply spikes successfully in the ASD process.

Developing organisational/PO support

For spikes to be applied effectively in an ASD process, there needs to be coordination and support from the PO or the organisation. According to two of the participants interviewed, P8 and FG1-3, support from the management or the PO is critical. According to P8, "support from business/senior management is needed for experimentation and learning is needed". FG1-3 added that having the buy-in from the PO and company is fundamental for the successful application of spikes. These two opinions indicate that organisational support is essential when introducing and applying spikes in the development process.

Allowing a shared technical understanding of the issue to be addressed

Based on the results of discussions with the individual participants and the focus groups, it was established that a shared understanding of the issue necessitating the spike's application is important. This involves the team having significant knowledge of the issue that the spikes need to address. For spikes to be applied effectively, the issue to be solved should be known to most of the team members.

According to two of the participants interviewed, P4 and P7, a team should have a shared understanding of the issue to be addressed, the latter stating that there is a need for "technical understanding among the team about the topic to be addressed". In the same line of thought, FG3-3 and FG1-4 noted that spikes need to be implemented by more than one person in a team to attain valuable outcomes. Notably, FG1-4 said that "for successful spike application, more than one individual should be involved in it". Also, P5 and FG1-4 mentioned the need to consider team collaboration when applying spikes. P5 also said that "collaboration amongst team members will lead to the effective application of spikes". These opinions among the participants confirm that having a shared understanding of the issue to be addressed is as essential as other factors.

Assuring sufficient time

When using spikes, it is essential to consider timeboxing, as previously stated. Having enough time, according to the participants, is an important factor when applying spikes. Once spikes have been initiated, they need to run to the end for informative outcomes to be obtained. According to two of the participants, P16 and FG1-3, providing sufficient time in every sprint that has a spike is a necessary precondition for success. P16 emphasised the importance of "creating spikes for every unknown, having time to understand the problem and apply it if it is usable for spikes to be successfully applied". Similarly, FG1-3 said that based on his experience, spikes are used to reduce risk and try to discover unknowns. However, estimating what you do not know is extremely difficult. What you need is approval from stakeholders to allow you the time you need to investigate and work on the spikes. As a result, having enough time to execute and comprehend the type of issue, as well as its dimensions, is essential to set the goals and expectations for conducting the spikes effectively.

Understanding stakeholder expectations

As some participants noted, understanding stakeholder expectations is one of the most critical factors that need to be taken into account for spikes to be successful. Two participants believed that the stakeholder's expectations of the end product and the cost and time invested in the software development should be considered before introducing spikes. According to P7, "knowing the stakeholder's expectations, long-term aim, and priority (the effort, time, and expense that can be invested) is important". On the same issue, FG2-1 stated that "before any spike, we have a kick-off session with stakeholders to review the deliverables, solution, risks, issues, and pull any further artefacts together in terms of API documentation". These responses illustrate the need to consider stakeholder expectations to apply spikes effectively in the ASD.

7.2.2. Factors that enhance the outcomes of spikes

Question
In your opinion, what should agile teams, particularly project/product managers, do to enhance the spikes' outcomes?

In responding to this question, the participants interviewed provided significant information based on their opinions concerning the factors that can enhance spike outcomes. In addition to the factors they identified earlier, the participants further highlighted what they believed could enhance the success of spikes in ASD projects. Some issues agile teams need to identify include those in the following paragraphs.

Clear goals and expectations

Agile teams need to have clear goals and expectations of the spikes they intend to apply to enhance their outcomes. P1, P2, P3, and P6, together with four others, indicated that there is a need for development teams to have a clear understanding of what they want to accomplish through the spikes. For instance, P3 noted that the "PM has to realise that spikes most often need the involvement of multiple team members, so the goals of the spike need to be clear". This view was shared by multiple other participants interviewed. P1 stated that the most crucial thing for agile teams is to understand what outcomes the spike is supposed to yield and the goal it is supposed to accomplish. Understanding these aspects makes it easier for the team to develop a precise spike that executes the role for which it was built. According to FG3-2, "the PM and team need to have a clear definition of what outcomes are expected and how those should provide good value". Thus, the participants' opinions indicate that having clear goals and expectations about the spikes is needed for the spike outcomes to be enhanced.

Spike objectives in risk reduction and discovery

For most participants, identifying that the objective of spikes is risk reduction and discovery can enhance the outcomes of their application in an ASD project. Seven of the participants mentioned this as one of the factors that can enhance spike outcomes. For instance, FG3-1 stated that "in my opinion, the PM or POs need to realise that by doing a spike we are not committing to the project, we are only investing some time to understand the risks/knowledge gain or even if it is evaluating the impact of new technology". In the same vein, FG2-3 referred to the objective being about discovery and risk reduction. No development output should be expected, though this might depend on context. The scope should be left open as possible, rather than only exploring pre-conceived solutions. In this regard, the participants' opinions point to the importance of knowing that spikes are primarily for risk reduction and discovery.

In addition, other participants in the interview held similar views and opinions. P9 said that "spikes never produce deployable functionality; they show us how to design and implement that functionality. By definition, spikes are to resolve uncertainty". They only provide information, and thus the objectives of the spike should be known to managers beforehand for the outcomes to be achieved. Further, a participant opined that a spike is just a tool to find some answers making it possible to move forward with the work. According to FG3-3, spikes result from a conversation about a feature or task. Therefore, spikes clarify a technical solution, not a business one.

Regular meetings

Addressing pending questions in the spike process is essential for any development team. The participants interviewed expressed the view that regular meetings are crucial in enhancing the outcomes of spikes. In line with the responses of seven participants involved in the study, P5 said "the PM and team need to realise that there should be regular meetings to address any pending questions. Possibly involving all dependencies with other teams". The meetings not only highlight the spike itself but also the results obtained. FG1-1 mentioned that to establish if a spike is beneficial or not, the project/iteration managers must ensure that, in most cases, a presentation is given after the spike. This is a good way of measuring if the spike was worth it or not. If not, the PMs/iteration managers should suggest some improvements for the next spike presentations. As P7 stated, "regular meetings facilitate regular discussion with stakeholders to keep them updated on the team's progress". With such arrangements, the outcomes of the spike are enhanced, and the discoveries made through it are implemented more precisely by the entire team. P16 said that,"whenever a spike is applied, the PM needs to end it with a presentation to the whole scrum team".

Running safe-to-fail experiments

According to one of the interview participants, it is often possible the outcome of a spike may not be satisfactory. Running a safe-to-fail experiment entails trying new things on a small scale, expecting that some of them might fail (Appelo, 2016). With this approach, development teams use spikes in a safe-to-fail environment to try and resolve any issues in the development process. By trying out several new things, a team can increase the chances of obtaining a better solution to the issue being investigated using spikes. Thus, the team should not be judged on their speed and should be supported regardless of the outcome. This was the opinion aired by P7 regarding the need for PMs to understand why running safe-to-fail experiments is vital in enhancing spike outcomes. P15 further stated that it is good scientific practice to run a parallel and independent safe-to-fail experiment using spikes. The experiment is meant to test the criteria established and ascertain whether the direction being explored can produce meaningful outcomes for the teams. According to FG1-3, "the PM and team need to know it might fail. So, it should create a POC and workshops to spread the knowledge and the results". This further illustrates why managers need to realise that running a parallel safe-to-fail experiment is essential for improving spike outcomes. Without such an arrangement, spike applications may fail to provide the solution or discovery needed. FG2-2 also pointed out that we should be prepared for a "failed spike" – a case in which the team identifies no further way forward with the spike topic.

Team autonomy

Team flexibility is essential for planning and executing spikes during an ASD process. In the focus group discussions, two members supported the idea that team autonomy enhances the application of a spike and its subsequent outcomes. According to P4, if the team thinks they need a spike, then they implement one. Managers need to point the team toward well-formed business problems or hypotheses to test and then get out of the way. The team self-organises around the work, supported by the Scrum Master. Further, to ensure better outcomes, FG2-2 asserted that managers should "trust the team – they don't need to check up on them or worry about progress; keep out of the way – the experts are working on it and don't need distractions". Thus, it is the participants' opinion that managers should realise team autonomy is required for spike outcomes to be optimised.

Stakeholder collaboration

A fundamental issue in software development is collaborating with the stakeholders to meet the deliverables and expectations. In the same way, the participants proposed that for agile teams to improve spike outcomes, they should be aware of the importance stakeholder collaboration has for the entire process. In this context, P6 opined that the PM and agile team should recognise that the best spikes can be explained to business stakeholders, including the PO, to clarify the impact that the answers to the question may have on the overall product delivery. This will drive business engagement with the spike and provide feedback on the size of the timebox and the number of resources to be applied. In support of this, P7 added that "the PM and agile team need to facilitate regular discussion with stakeholders to keep them updated on the team's progress. Any decision taken should be the team's decision and not something imposed on the team". In this regard, managers need to understand that stakeholder collaboration is a pivotal issue for better spike outcomes to be realised.

Two questions were asked of the participants to understand the various success factors. Although the questions took different angles, the intention was to gather as much information on success factors as possible. As discussed in sections 7.2.1 and 7.2.2, there were 16 success factors that the participants mentioned. However, some were not supported or mentioned by most of the participants. As a result, the common factors in the responses from the two questions yielded 11 factors. The remaining five factors were only mentioned by one or two respondents or only argued vaguely. Based on the consideration of the responses from the two questions, the factors are summarised in Figure 7.8.

Figure 7.8: Common success factors for spike applications

7.2.3. Categories of success factors

The purpose of categorising success factors is to associate them with their respective contexts. The various categories were determined using Chow and Cao's (2008) proposed model for identifying critical success factors in agile software development projects. This model classified success factors in software development into five categories: people-related, project-related, organisation-related, procedural and technical factors.

Question
The success factors for the correct application of spikes may take different dimensions. Based on your experience and your opinion, can you provide some of the factors based on the following categories? **a) People factors** **b) Organisational factors** **c) Procedural factors** **d) Project-related factors** **e) Technical factors**

When the participants were asked to categorise the factors into the five categories, their responses varied, but some common responses were also obtained. The following factors were placed under each category.

People factors

Several participants pointed out that team skills, moral expertise, and motivation are important factors that fall under the category of the people required for the effective application of spikes in ASD projects. Among the notable responses, FG1-1 stated that the experience of the team members is important, but this depends on the complexity of each spike. So, the expertise of each team member should be considered when assigning a complex spike. P14 also pointed out that “people must be eager to explore/investigate/understand even if this may cause additional problems short-term”. In general, the participants’ responses suggested that there is a need for them to have the necessary skills, expertise, motivations, and morale when applying spikes.

In addition to these factors, the participants also categorised team autonomy, maturity, and inclusiveness as people factors. FG1-1 noted that “in order to apply spikes properly, the team members responsible for the spikes should be autonomous and motivated”. P6 also noted that applying spikes should not be left only to the “smart developers” but instead should involve all the team members regardless of their roles. In support of this, FG2-1 noted that substantial maturity is required of the team or expertise of members within their field and skillset to allow spikes to be truly successful and to drive the outcome and output developers want to obtain. Other than these factors, the participants also categorised clear communication, cross-functional teams, collaboration, and willingness to seek help as people-related factors that should be considered when applying spikes for instrumental outcomes. The outcomes are summarised in Figure 7.9

Figure 7.9: People factors for the successful application of spikes

Organisational factors

Similar to the previous category, the participants had varying opinions about the factors considered to be organisational in nature. Among the notable factors that were commonly identified by the participants were acceptance of the agile methodology being used, willingness to invest time for future value, and organisational diversity. In support of categorising these factors as organisational, P12 noted that an organisation adopting agile is a factor for the successful application of spikes. To support the classification of organisational diversity as a success factor, FG3-3 noted that “spikes are explorative tasks that will sometimes lead to trial-and-error processes, and without the freedom to fail even miserably, innovation and creativity will not thrive”. P11 also opined that organisation should allow time to conduct spikes with the knowledge that they will not deliver anything but allow for better fact-based decisions. This makes it possible for spikes to be applied despite having no specific direct value to the project. Other factors, including support, policies, stakeholder engagement, and team autonomy, were also categorised as organisational. Table 7.2 summarizes the participants' opinions concerning which factors are classified as organisational.

Table 7.2: The most common organisational factors for successful application of spikes

Participants	Organisational Factors									
	Acceptance of agile methodology	Willingness to invest time	Expectation	Cross-functional team	Organisational diversity	Running safe-to-fail experiments	Stakeholder engagement	Policies	Support	Team autonomy
P1				●						
P2			●							
P3										●
P4					●					
P5									●	
P6	●						●			●
P7					●		●			
P8	●									
P9		●								
P10						●				
P11		●								
P12	●									
P13				●						
P14		●								
P15		●			●					
P16	●									
FG1-1	●								●	
FG1-2								●		
FG1-3	●		●							
FG1-4		●								
FG2-1				●			●			
FG2-2		●						●	●	●
FG2-3						●	●			
FG3-1		●								
FG3-2					●					
FG3-3					●					

Process/procedural factors

When the participants were asked to identify process-related factors, six common factors were highlighted. Key among them were following the agile-oriented process, timeboxing, and having clear goals. For instance, P5 noted that "following the agile-oriented project management process is one of the factors needed for the successful application of spikes". In support of this, P7, P11, P12, P13, and P16 also mentioned following the agile process as a procedural factor that defines the successful application of spikes. Further, FG3-2 also noted that timeboxing is a critical factor that needs to be well thought out in the ASD process when applying spikes. Developers will always question whether the extent of the spike is limited to a specific timeframe. As noted earlier, clear goals are also an essential factor that some participants categorised under the procedural/process category. Table 7.3 illustrates some of the common factors under this category identified by the participants.

Table 7.3: Summary of procedure/process factors

	Process/Procedural Factors					
Participants	Following the agile-oriented project management process	Timeboxing	Focus on specific questions	Clear definition of 'done'	Clear goals	The right use of spikes
P1					●	
P2		●	●	●	●	
P5	●					
P6			●	●		
P7	●					
P8		●				
P10						●
P11	●					
P12	●					
P13	●					
P16	●					
FG1-1	●					
FG1-3					●	
FG2-2						
FG2-3						●
FG3-2		●				

Project-related factors

The interviews with the individual participants and focus groups revealed seven common project-related factors for the effective application of spikes in ASD. The most common were realistic goals and timelines, sufficient budget, and time. According to the participants, a development team intending to apply spikes must have realistic goals that act as a guide for what type of spikes can be used. P7 noted that realistic goal setting is needed for an agile team to succeed. By setting timelines, spikes are applied at the appropriate point in the development process. P16 further stated that "project metrics such as budget and quality are essential success factors when applying spikes". Other experts such as P8 noted that time for discovery is needed to help assess outcomes and drive decisions.

Further, the participants noted that training and access to resources influence the effectiveness of spike applications. According to P7, training for the team when they need it and availability of or access to the right tools (software/hardware) can help the team succeed in applying spikes to investigate unclear issues in the project. Also, an appropriate environment was raised as a potential project-related factor. Notably, P1 suggested that teams need sufficient scheduling room to conduct the spike without interference from other processes. Other factors highlighted by the participants included the right use of spikes and using spikes as an investment. P9, for instance, opined that before using a spike, the team should ask questions like “What are the benefits of spikes? Are we getting sufficient ROI (Return on Investment) on spikes to continue using them? Or is the team using them to ‘cheat’ by not having spikes count as an impact on throughput?” P10 and FG3-2 also pointed out the same. Table 7.4 summarizes the participants’ responses concerning the seven project-related factors.

Table 7.4: Summary of project-related factors

	Project-Related Factors						
Participants	**Realistic goal setting and timeline**	**Sufficient budget**	**Sufficient time**	**Training and access to resources**	**Appropriate environment**	**Right use of spikes**	**Use of spikes as an investment**
P1					●		
P3	●						
P5	●						
P6	●	●		●	●		
P7	●			●	●		
P8			●				
P9						●	
P10							●
P11	●	●					
P12	●						
P14		●	●				
P15	●						
P16		●					
FG1-1		●	●				
FG1-2	●						
FG1-3		●	●				
FG1-4		●					
FG2-1				●			
FG2-3	●					●	
FG3-1							●
FG3-2		●				●	

Technical factors

These factors include those related to the engineering of the software that is being developed by the team. Aspects such as the coding standards, tools, sample design, testing, delivery strategy, and team training are among the factors categorised and considered technical (Arcos-Medina and Mauricio, 2020). When an ASD project is started, all these technical factors are considered, especially those relating to the agile methodology used and the team training. Table 7.5 summarizes the responses of the participants in categorising success factors considered to be technical.

Table 7.5: Summary of technical success factors

Participants	Technical Factors				
	Team's technical ability and expertise	Access to appropriate tools and resources	Team's knowledge	Technical training	Understanding architecture limitations and guidelines
P1		●			
P2			●		
P3					●
P4		●			
P6	●				
P7	●			●	●
P8		●			
P11	●		●	●	
P14	●				
P15	●				
P16	●				
FG1-2			●		
FG1-3		●	●		
FG1-4	●				
FG2-2		●		●	
FG2-3	●	●			
FG3-1		●			
FG3-3	●				

As can be seen, five technical factors were listed by more than one participant in the interviews. Those most commonly mentioned were the team's technical ability and expertise and access to appropriate tools and resources. P11 noted that the developer's knowledge of the area and/or technology that is needed is a fundamental technical factor for the successful application of spikes. In addition, P1 opined that "the team needs appropriate tools, environments, and infrastructure to perform research and conduct experiments". These views were also supported by five other participants. Other success factors suggested as technical in nature included understanding architecture limitations and guidelines, team knowledge, and technical training, each being supported by three or more participants. In terms of understanding the architectural limitations of the product, P7 noted that a grasp of architectural guidelines (design before implementation), coding guidelines, quality assurance (Sonar rules), Continuous integration and continuous deployment CI/CD (with mandatory reviewers), unit testing, automation testing, branching strategies (major, patch, minor releases), are necessary for the successful application of spikes. Regarding team knowledge, P11 believed that developers' understanding of the area and/or technology is needed to apply spikes properly. Lastly, as supported by P7, P11, and FG2-2, team training was also noted as one of the success factors that might be considered when applying spikes.

7.2.4. Opinions of participants on the categorisation of factors

After categorising the factors into five categories, participants were asked whether the categories were a good fit with the actual factors, which should be taken into account when applying spikes. From the responses, 17 participants (71%) affirmed that the categorisation did provide a fit. However, seven participants (29%) disagreed, and two others did not have anything to say on this issue and skipped the question. Figure 7.10 presents the responses obtained.

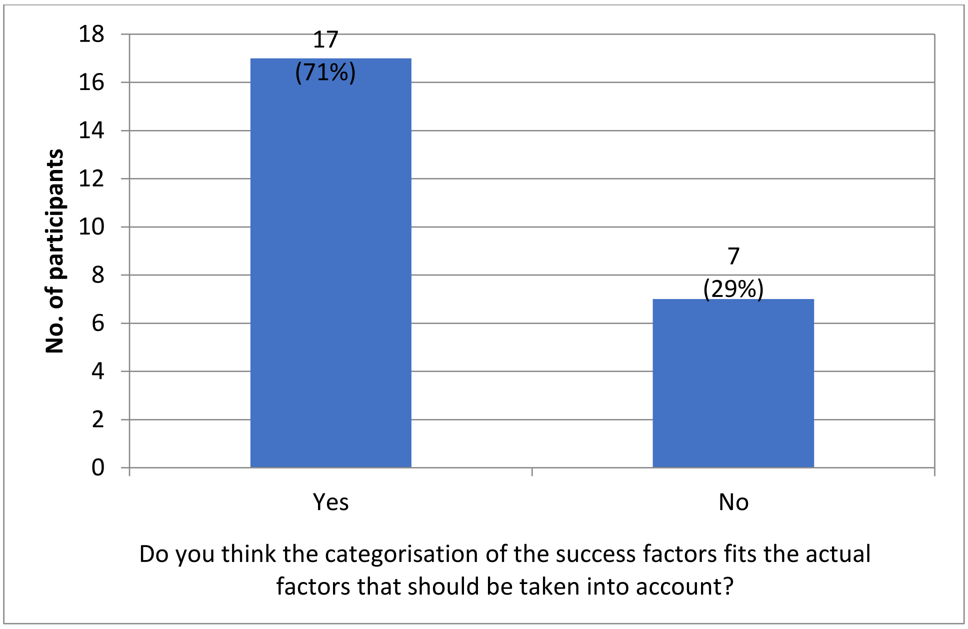


Figure 7.10: Participants' opinions on the categorisation of success factors

The seven participants who said the categories were not a fit were asked to provide a rationale for their opinion. Three said that the list was not exhaustive and that the categories did not entirely cover everything that should be considered. For instance, P2 stated that, "we can use spikes for many things and these categories narrow the scope too much". The categories might rarely be relevant, or maybe only one or two would be relevant in a specific case. It represented a start, but an attempt to categorise implies trying to constrain, and that reduces applicability. This rationale seems plausible since most of the developers had their own views that were not consistent all the time. P7 noted that, "the list should act as a guide for the team to plan for a spike. However, any additional steps taken toward the successful implementation of the spike can be added to this list. This way, the categorisation will cover more factors covering many possibilities".

The remaining four participants cited that the factors were subjective and that no single criterion can be applied in all spikes. According to FG2-2, "not all the factors will be applicable and will depend on the spike topic. For example, if the topic is focused on understanding more technical content with the aim of providing better estimates for the PO to weigh up the suitability of a product backlog item to be put forward to the next sprint, or whether it should be ditched entirely, support from senior management is not required. Under these circumstances, the category seems to constrain the factors that developers need to consider when applying spikes". Further, FG3-1 noted that "the categorisation is very subjective as each organisation handles it differently". In this regard, some of the participants believe that the categorisation does not fit the actual factors to be considered to successfully apply spikes in an ASD.

As illustrated in Figure 7.11, categorising these factors yielded a significant number of success factors related to each category. Unlike in Figure 7.8, all factors were aggregated together, resulting in fewer factors.

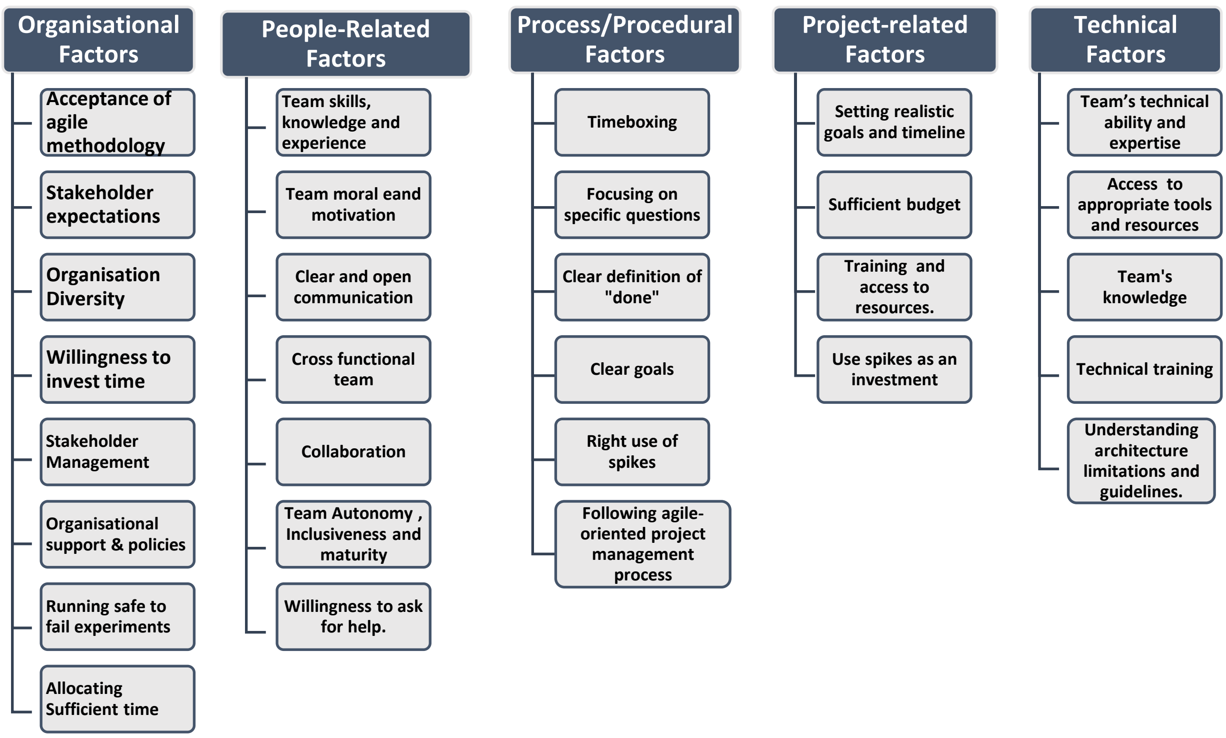


Figure 7.11: Common success factor categories

7.3. Common Success Factors of Spikes: Quantitative Analysis Results

To complement the qualitative information collected from the participants through interviews, a survey was also conducted to collect more data on success factors in the application of spikes in ASD. The survey was conducted using a questionnaire consisting of closed-ended and open-ended questions. A total of 64 agile practitioners participated in the questionnaire. This section highlights the results obtained from the questionnaire conducted to answer RQ4.

7.3.1. Participants' demographic characteristics

A total of 64 practitioners were recruited and participated in the research study. The participants came from different organisations operating in different industries, but mostly IT and software development. From the participants, it was established that the majority (15) had experience of between 11 and 15 years in ASD. In addition, there were eight participants who reported having experience ranging between 16 and 20 years. Very few participants (only three) had experience of less than two years. The distribution of the experience in ASD is shown in Figure 7.12. Given that the majority of those who took part in the questionnaire had expertise with ASD projects spanning over five years, it is safe to say that the information they supplied was grounded in extensive knowledge of software development. Notably, the questionnaire participants were not the same as those in the focus groups and individual participant interviews described in section 7.1.

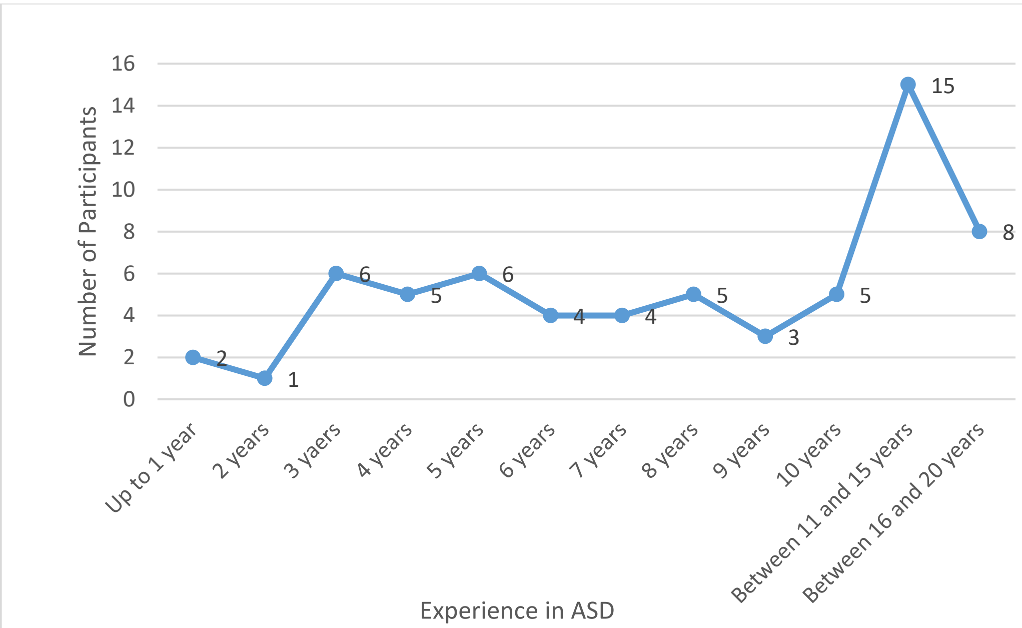


Figure 7.12: Participants' experience in ASD

Unlike the experience in ASD, most participants reported having less than six years of experience in spike applications. Notably, 15.6% (10 participants) reported having experience of only up to one year, while 23.4% of participants reported having over 11 years of experience in spikes application. As shown in Figure 7.13, most of the participants involved in the survey had experience of between 2–6 years, an adequate amount of time for an individual to have an idea of the success factors for the application of spikes.

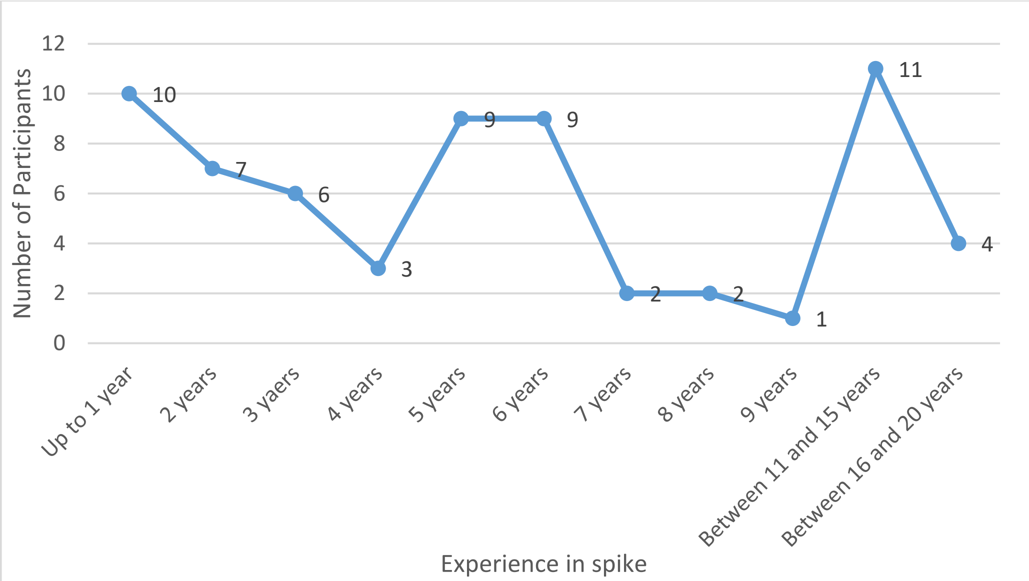


Figure 7.13: Participants' experience in spike

Among the participants involved in the questionnaire, different roles in ASD were represented. From the responses obtained, the participants reported more than 10 roles in agile. The majority of them (43 participants) reported being Scrum Masters. Second were agile coaches (28). However, most of the participants reported having more than one role. For instance, some POs were also Scrum Masters and agile coaches at the same time. The diversity of the agile roles in the population sample implies that the practitioners are flexible and can operate in more than one role. From the responses obtained, other than the common agile roles, including developer, tester, PM, Scrum Master, PO, and agile coach, there were other roles stated by the participants. These included solution architect, data scientist, trainer, and delivery manager, among others, as shown in Figure 7.14.

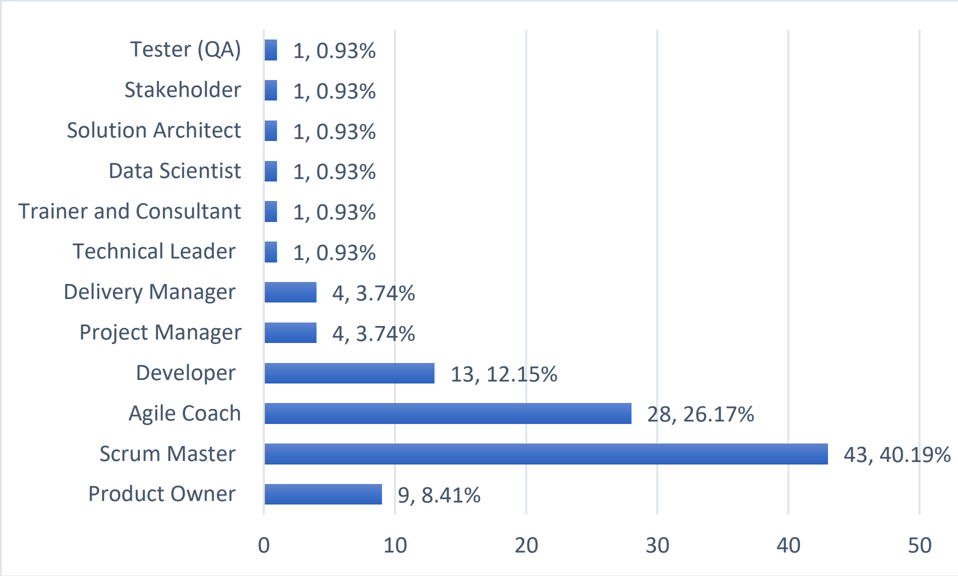


Figure 7.14: Participants' roles in their organisations

The participants involved in the questionnaire were from nine different sectors, with the majority coming from the IT and software development sector. More specifically, 67.2% of the individuals came from organisations belonging to the IT industry and focusing majorly on software development. Around 11% of those surveyed were from consultancy firms, while relatively few came from the aviation, logistics, education, retail, product management, finance and government sectors, as presented in Figure 7.15.

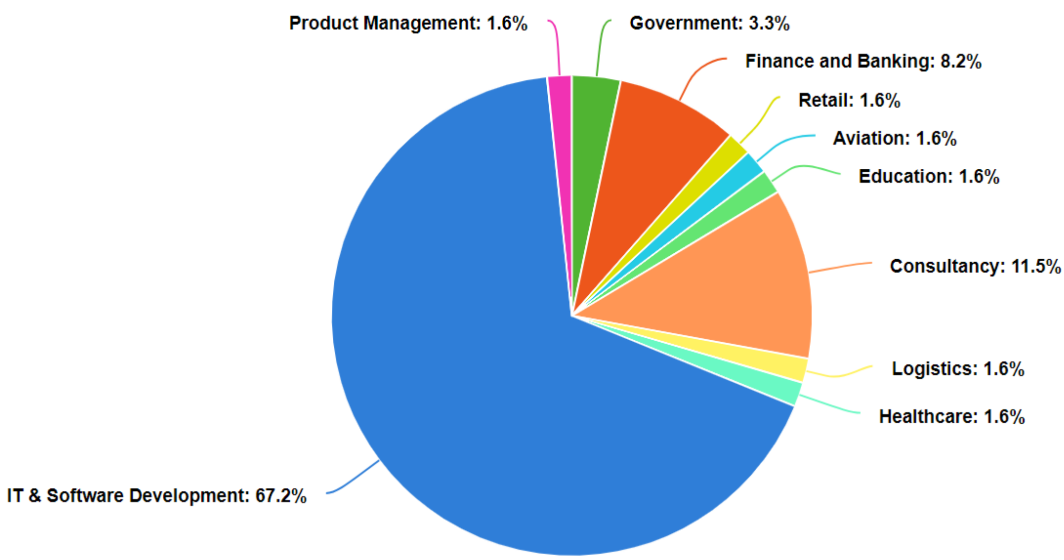


Figure 7.15: Participant's organisational sectors

The organisations from which the participants were recruited ranged in size from micro (fewer than ten employees) to large (more than 250 employees). From the data collected, 63% of the participants were from large organisations in different industries shown in Figure 7.16. This proportion represents 40 of the 64 participants surveyed. A considerable number of participants also came from medium-sized organisations (51–250 employees). Small (10–50 employees) and micro-organisations were represented by 6 participants each, as shown in Figure 7.16. It is thus apparent that most of the participants serve a large clientele based on the size of the organisations.

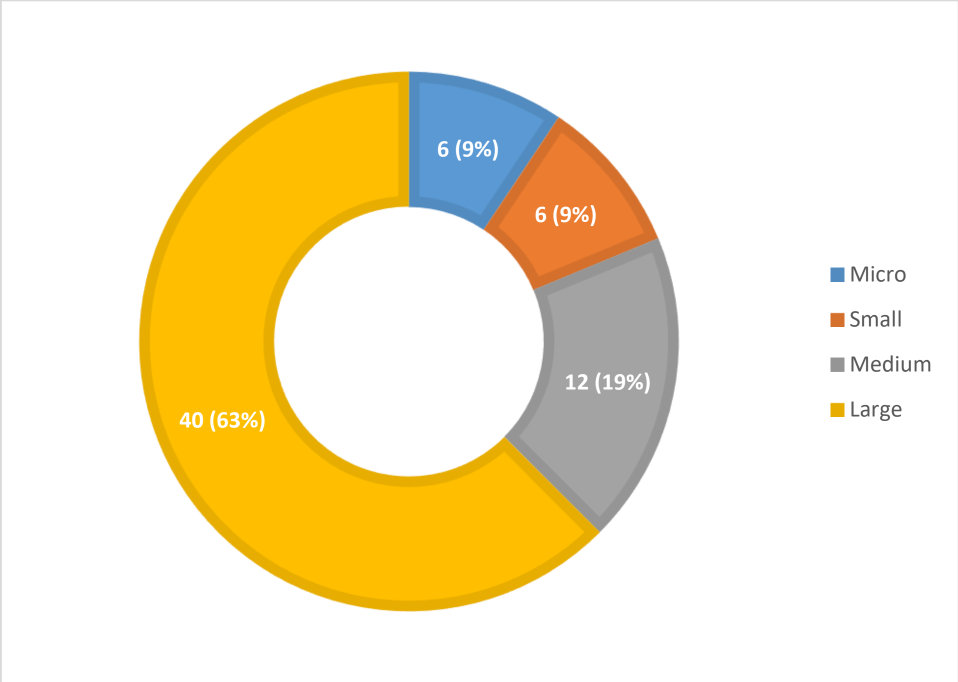


Figure 7.16: Size of participants' organisations

The last characteristic of the participants examined considered the agile methodologies they used. Conventionally, there are several methodologies, with scrum being the most used currently. Based on the data collected, 76.6% (42) of the participants reported using scrum as their primary methodology. Other methodologies, such as Kanban (4.7%), extreme programming (3.1%), and SAFe (4.7%), were less prominent among the practitioners surveyed. However, some of the respondents (10.9% of the sample) confirmed that they employ more than one agile methodology in their ASD projects. Some of the respondents revealed that they use multiple agile methods, such as a combination of Scrum, Kanban (Scrumban), and SAFe. The participants used these agile methods to achieve different objectives. Also, clients' preference was considered when completing their projects. The breakdown of the agile methodologies used by participants is shown in Figure 7.17.

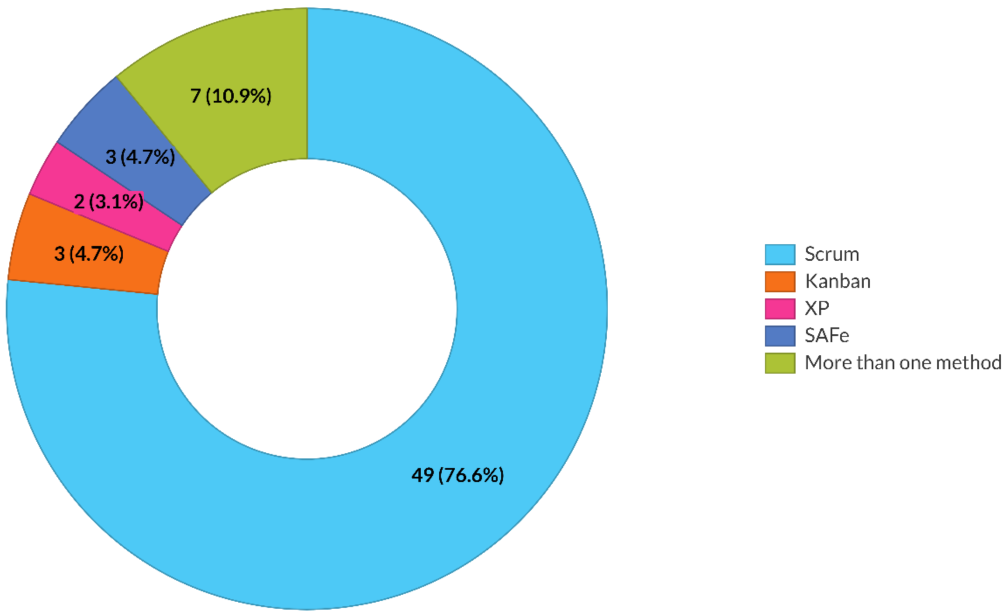


Figure 7.17: Agile methodologies used by survey participants

7.3.2. Results of analysis of success factors

Question
Based on your experience with agile software development and spikes, do you think that there are factors affecting the application of spikes?

As practitioners who have been actively engaged in software development and the use of spikes in their processes, the survey asked their opinions concerning whether there are known factors affecting their application. The vast majority, 92.2% of the participants, believed that there are factors that determine the successful application of spikes. However, about 5% were not sure, and 3% said they believed there were no factors affecting the application of spikes. As shown in Figure 7.18, 59 participants expressed confidence that certain factors could influence the application of spikes in ASD, whether people-, process-, project-, organisational-, or technical-related.

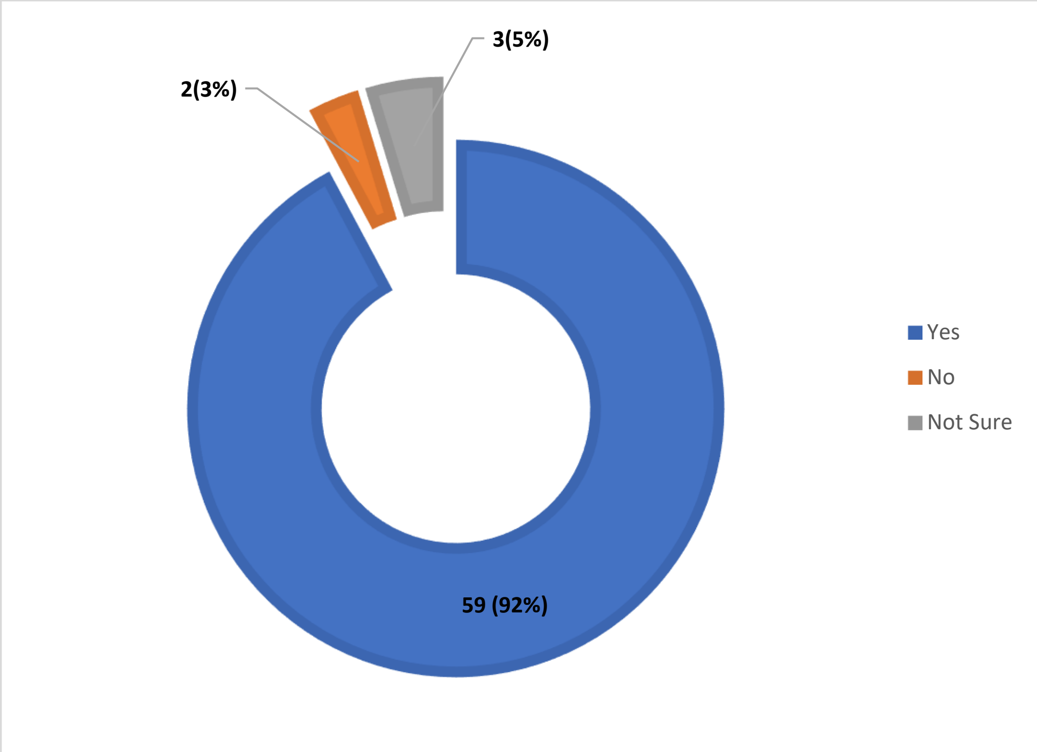


Figure 7.18: Participants' opinions of the existence of success factors for spike application

As in the qualitative section, the success factors considered in the application of spikes in ASD are grouped into five categories: people, organisational, process, technical, and project. However, earlier results showed that some participants did not consider the categories applicable as the factors vary based on the type of spike and nature of the project. The survey revealed additional factors in each category, discussed in the following paragraphs.

Organisational factors

There are various factors related to the organisational structure of the teams undertaking agile software development projects (Arcos-Medina and Mauricio, 2020). These factors include items such as support, commitment, collaboration, and working environment, among others. For any project to be undertaken successfully, these factors are critical. Moreover, they streamline and support teams undertaking the project by providing a suitable environment for the execution of agile projects. With respect to agile spikes, the same factors are essential for successful application. They offer organisational support to the teams by applying spikes to estimate stories or minimise project risk.

A significant number of respondents were identified from the survey responses (44 participants, 39.6%) believe that the team environment provided by organisations is a critical success factor in the application of spikes. Similarly, management commitment also proved to be a significant success factor in the application of spikes. Around 28.83% of the participants (32) selected this as an organisational factor influencing spike application and success. Also, the organisational environment was mentioned by 24 participants. The organisational environment can hamper or facilitate access to resources, thus posing either an opportunity for success or a threat to projects. Besides these three factors, 11 participants also mentioned others, including a large degree of autonomy, organisational processes, and technology selection approved by the team organisation. However, other participants considered that there were no organisational factors that applied to their companies. Figure 7.19 shows the responses of the participants regarding organisational factors.

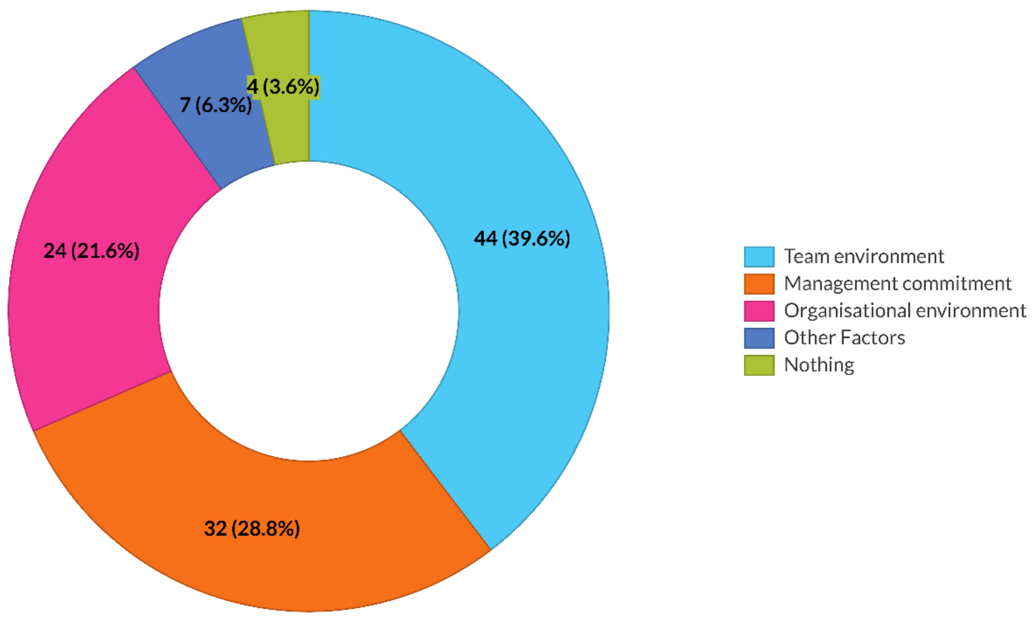


Figure 7.19: Proposed organisation-related factors for successful spike application

People-related factors

The people-related category of factors concerns individual or team capabilities influencing ASD. The expertise, motivation, and even the size of the team are crucial factors that can make the application of spikes in ASD succeed or fail in terms of accomplishing the role intended. Conboy et al. (2011) note that the people factor in ASD is essential due to the changing boundaries of agile methods, which require teams to get out of their comfort zones to succeed. The skills of the team in brainstorming solutions and integrating spikes to estimate user stories are critical to the success of spike application in ASD. Not ensuring there are sufficient skills to apply spikes in an ASD project will lead to probable failure. Again, the timing of applying spikes is not random but rather guided by the need to apply them (Conboy et al., 2011). In this case, people-related factors, such as expertise and competencies, are essential in the successful application of spikes at the right time and for the right reason.

From the data collected, 47 participants mentioned team capability as a people-related factor to be considered when applying spikes. More than half of the respondents surveyed (37) also identified expertise as an important people-related success factor. This factor concerns the skills and competencies of the practitioners involved in applying spikes in ASD projects. Furthermore, the results showed that 16.67% (26 participants) support the idea that motivation is a critical success factor in spike application. Customer involvement and team size were also noted as factors by 18 and 14 participants respectively. Moreover, some participants mentioned other factors. Notably, 14 participants mentioned that factors such as team maturity, prioritisation, SME availability, and confidence among others, also need to be considered when applying spikes. Figure 7.20 summarizes the responses of the participants.

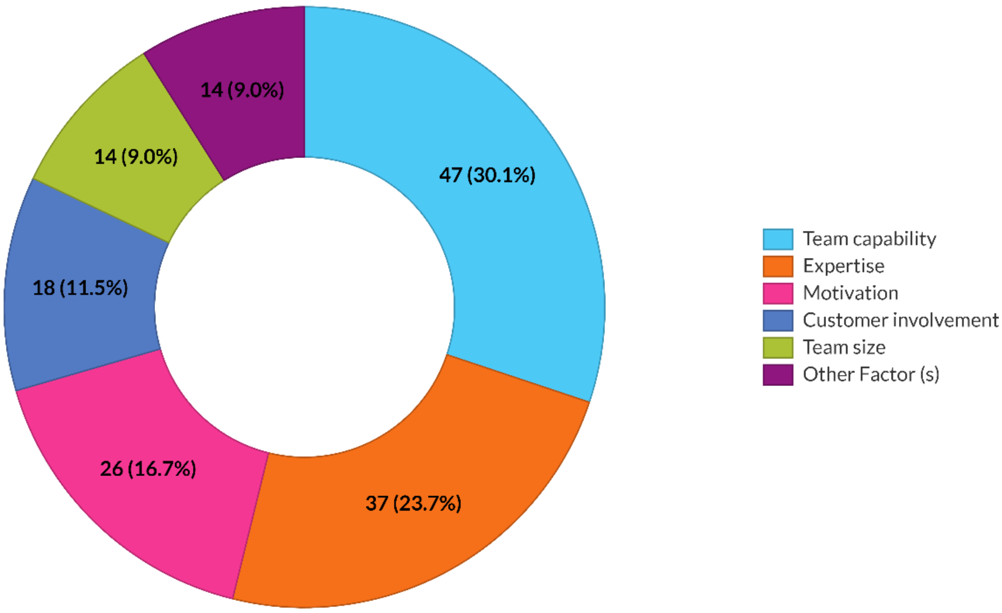


Figure 7.20: Proposed people-related factors for successful spike application

Process-related factors

Factors categorised as process-related are tied to the internal project management framework, which includes the schedule, configuration of management, and detailed plans of action when undertaking the project (Conboy et al., 2011). The process design is considered when making any adjustments to the project. Thus, when applying spikes in ASD projects, it is necessary to consider this category of factors. Essentially, the project management process is a crucial consideration as it gives direction and a framework for how the project should be undertaken. Furthermore, process-related factors such as work scheduling need to be considered when applying spikes. According to Arcos-Medina and Mauricio (2020), timeboxing is essential in ensuring the project remains within its timeline. Thus, it is important to consider such factors when applying spikes in ASD projects.

Responding to the questionnaire, the participants opined that the project management process and work schedule are the process-related factors most considered when applying spikes. Notably, each of these was supported by more than 30 practitioners, indicating the significance of these two aspects for the successful application of spikes. Furthermore, around 19.8% (22 participants) noted that the project definition process is a key area for applying spikes successfully. The project definition process involves identifying the expectations of all stakeholders regarding the outcome of the ASD project. Configuration management was also mentioned as one of the factors to be considered when applying spikes. It is critical to ensure that the spike is completed, as well as to ensure that the results of the spike are trackable and reproducible (Hunt, 2018). A total of 16 participants argued for this point, which primarily involves the process of establishing and maintaining consistency within a project. Finally, some 10 participants mentioned other success factors not included in the questionnaire options. These included factors such as the SDLC methodology used, the agile implementation process, and how the roles of each team member are stipulated in the project. In addition, clear understanding or empiricism was mentioned as a success factor to be considered. This involves understanding the view that all concepts originate from experience. However, some participants among the 10 stated that process-related factors are not relevant when applying spikes in an ASD. Figure 7.21 summarises the responses of the participants.

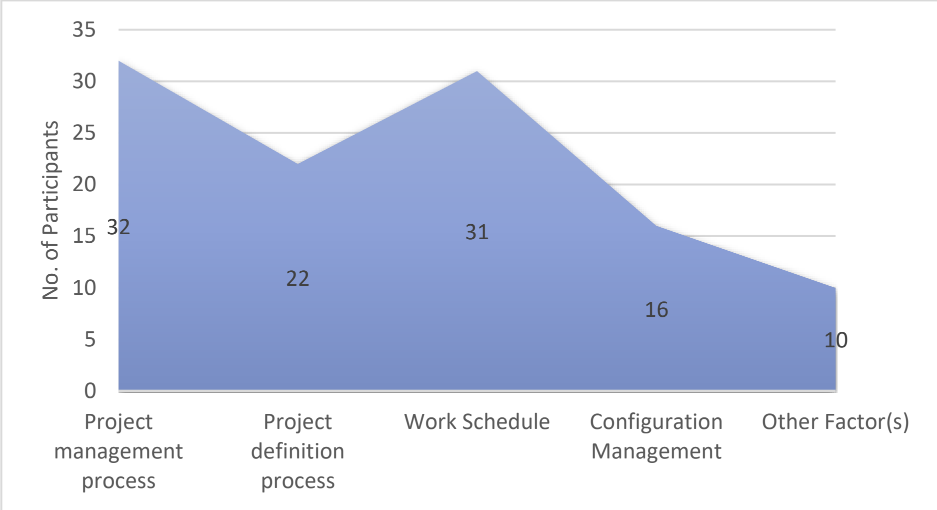


Figure 7.21: Proposed process-related factors for successful spike application

Technical factors

Technical factors include those related to the engineering process of the software that is being developed by the team (Arcos-Medina and Mauricio, 2020). Aspects such as the coding standards, tools, sample design, testing, delivery strategy, and team training are some of the factors categorised as and considered to be technical. Often, when an ASD project is started, all these technical factors are considered, especially those related to the agile methodology being used and the team training. Similarly, in applying spikes in ASD projects, these factors should be considered for two reasons. First, the technical factors dictate the type of spike to be applied. Second, the factors define whether a team will be able to apply spikes successfully given the technical training the team has had in spike application. Therefore, it is sensible to consider technical factors when applying spikes in ASD projects.

The participants predominantly mentioned several of the above techniques in the questionnaire responses. In particular, 30.48% (32 participants) pointed out that the agile software method adopted for the project should be considered for a spike to be successfully applied. The delivery strategy was also frequently mentioned by the participants, with 27 out of the 66 indicating that it is among the factors that should be considered when applying spikes in an ASD. The technical training of the team was referenced by 34 (32.3%) participants as a success factor of importance when using spikes. The ability of the practitioners to handle specific spikes in the project should be taken into account before applying a spike to enhance the chances of success. In addition, 11 participants indicated other probable success factors that they considered in their ASD projects. Among them were programming languages, architectural design, coding guidelines, and unit testing. Also, the participants suggested factors such as the feasibility of the proposed scope and definition of what a successful spike encompasses. However, others opined that this category did not apply to their organisation/team when implementing spikes in ASD projects. Figure 7.22 illustrates the summary responses of the participants.

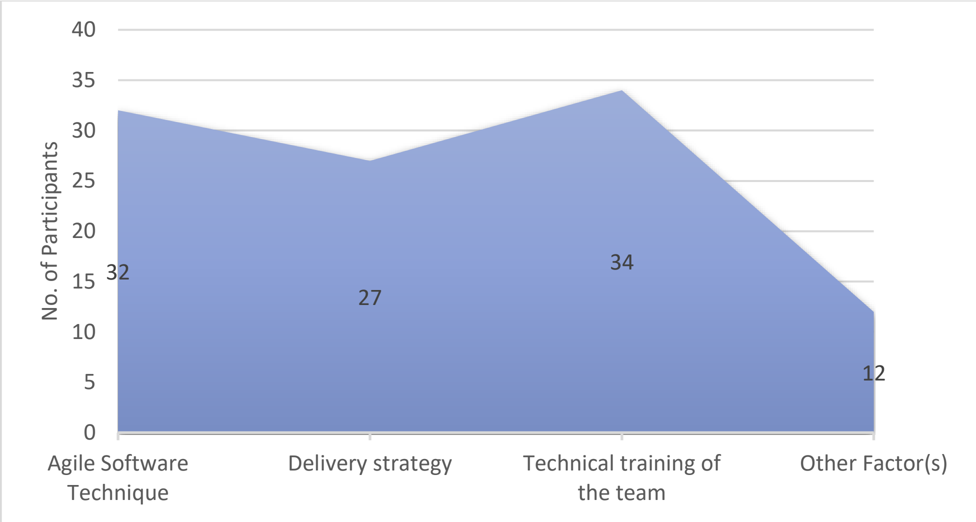


Figure 7.22: Proposed technical factors for successful spike application

Project-related factors

Project-related factors concern the nature of the software project during its development. These include issues such as the size of the project, the schedule, certainty of risks, and the team's independence, among others (Arcos-Medina and Mauricio, 2020). These factors are important not only when applying spikes, but also when undertaking the entire project. Neglecting them has the potential for severe consequences due to unknown risks that have not been investigated. Furthermore, by understanding the project-related factors, Arcos-Medina and Mauricio (2020) argue that collaboration between developers and users becomes more robust. As a result, risks can more easily be identified. Thus, the project-related factors are meant to ensure that despite disruption to the project schedule, the project can stay within the allotted timeframe.

The questionnaire data indicated that the project schedule is the primary consideration for development teams when aiming for the successful application of spikes. Notably, 34 (60.70%) of the participants believed the project schedule should be considered when applying spikes. Ideally, spikes are timeboxed and depend on the functions they are meant to serve. Conventionally, spikes cannot run forever. Instead, they are applied for a specific period to estimate the intended user story and help in preventing a known or unknown risk by providing more information to the developers. In addition, the nature of the project was referenced by 32 (57.10%) participants as one of the critical factors for applying spikes successfully. Project size was also mentioned as among the important factors. When the project is large, there is a likelihood that multiple spikes will be applied for different purposes. However, some small projects may not need to use spikes. In general, the size of the project being undertaken may also dictate whether spike application will be successful or not, depending on the time allocated for each spike. However, large projects may have multiple development teams, artefacts, and requirements, which makes compatibility more of a challenge than in small projects, and thus the application of spikes becomes more complex (Lalsing et al., 2012). Other factors not covered in the questionnaire were mentioned by around 13 of the participants. These included realistic goal setting, willingness to accept change, and grade of innovation. Other participants also proposed that technical complexity and the iterative ability of the project should be considered for the successful application of spikes.

When the project being undertaken is complex, spikes are needed to harness more information about aspects of the process that the team is uncertain about. The type of spike applied should be determined by the complexity of the issue it is intended to resolve (Conboy et al., 2011). Thus, it is the case that the technical complexity of an ASD project needs to be considered to apply spikes successfully, such as investigating unfamiliar technologies and frameworks, estimating the effort required for a specific feature or task, determining the feasibility of a feature or task, identifying the best technical approach for a feature or task, refining implementation approaches, experimenting with different solutions to a feature or task, developing prototypes to confirm assumptions, refactoring existing code to improve maintainability, and automating manual processes (Hunt, 2018). Nonetheless, a couple of participants said that project-related factors were not relevant to the application of spikes in their projects. Figure 7.23 presents the responses of the participants concerning the project-related factors that should be considered for the successful application of spikes.

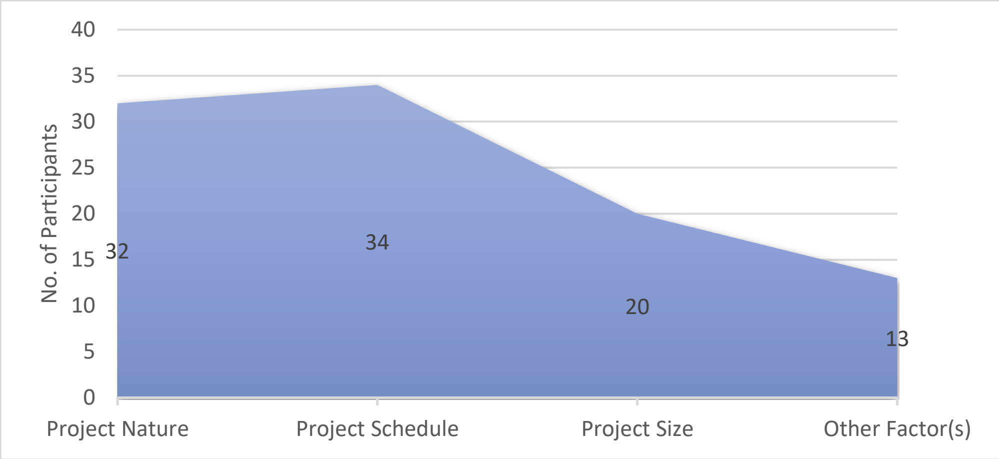


Figure 7.23: Proposed project-related factors for successful spike application

7.3.3. Participants' opinions on the categorisation of factors

Question
Do you think the categorisation of the success factors above is a good fit with the actual factors that should be taken into account when applying spikes?

Following on from their responses concerning the categories of success factors, the participants expressed mixed views about their suitability in guiding which factors should be considered. However, most respondents agreed that the categorisation corresponded with the actual factors they would consider whenever using spikes in their ASD projects. Notably, 72% of the participants responded in the affirmative, whereas only 28% said the categories were not a good fit. Although almost all the participants agreed that there were some factors that should be considered when using spikes in ASD projects, not all agreed on the various categories under which they were placed, as shown in Figure 7.24.

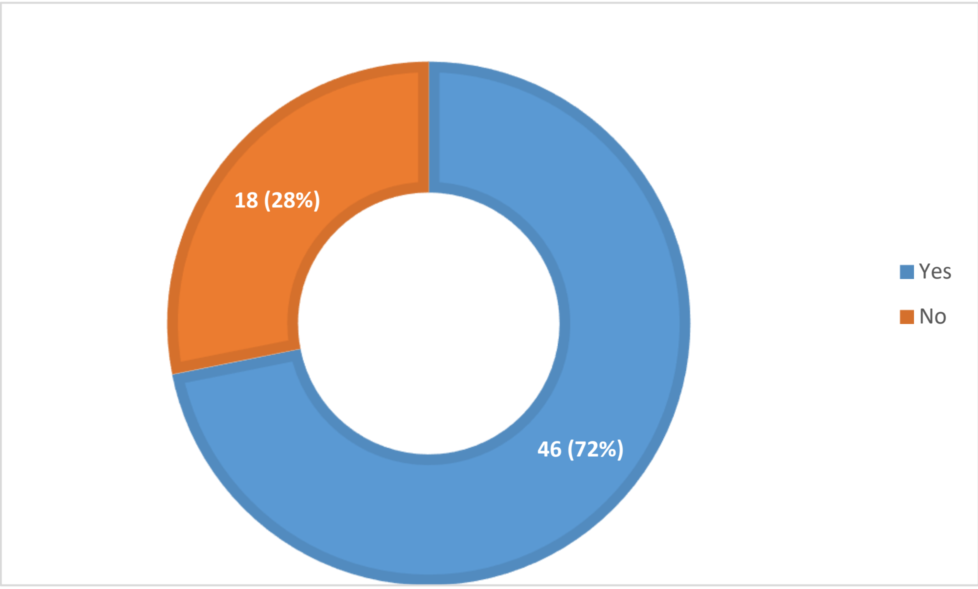


Figure 7.24: Participants' perspectives on categorising factors

The participants who opined that the categories were not a good fit gave various reasons for their responses. According to one: “The goal of spikes is to learn about the unknown from a delivery standpoint. Obtaining certainty in the face of uncertainty. So that when stories are chosen in sprints, they can be done with as little uncertainty as possible. So, it is entirely up to the team to decide how to select these items”.

From this rationale, the participants suggests that the success of spikes depends on the team and how they pick which uncertainties to solve. Not all the factors categorised necessarily need to be considered. Furthermore, some participants acknowledged that most of the factors would have an impact on spike application, but not to the extent of determining success. One of the participants, for example, stated: “I feel like these factors all have an impact but aren’t really specifically related to the success of my spikes. They may have an impact on my overall agile process, and thus the project’s success, but spikes can still be used effectively in a variety of situations”.

In this regard, the spike categories are irrelevant and are not entirely success factors that are universally applicable to all ASD teams. Other participants responded that each spike topic must be considered independently, and that the categories of success factors did not matter or fit the actual aspects they would take into account when using spikes in their ASD processes.

Question
Based on your experience, should the product owner or development team take into account certain factors that help to apply and complete spikes successfully?

This question was posed to the participants after presenting a list of factors and their categorisation to have a good understanding of the success factors of spikes so that they could provide more accurate responses to this question (see appendix H). When the participants were asked to respond to the question of whether certain factors might be relevant based on their experience with ASD and spike application, the vast majority (around 92%) agreed that some factors have to be taken into account to ensure the success of spikes. However, 8% (5 participants) disagreed, stating that no factors needed to be considered when using spikes in ASD projects. Overall though, Figure 7.25 shows the clear view among participants that certain factors cannot be ignored when using spikes.

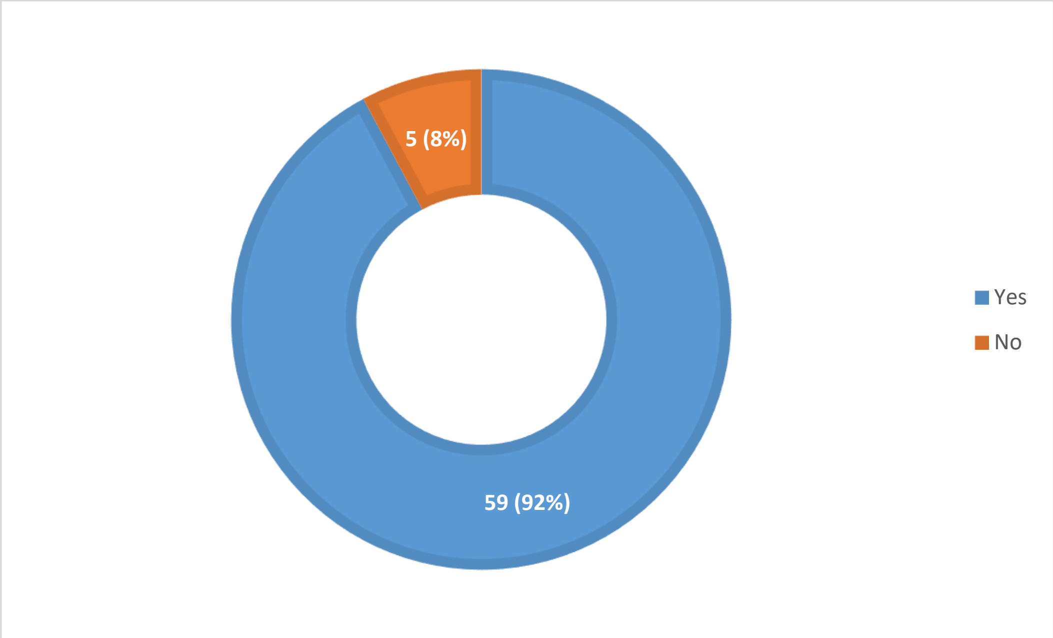


Figure 7.25: Opinions on the existence and importance of success factors

The 59 participants who answered yes were asked to justify their responses by naming some specific factors they believed would aid in achieving the aims of spikes. They referenced several factors, among which were the following:

i. **Clear scope:** Each spike should answer a specific question or clarify a specific function.
ii. **Timebox:** This includes the maximum effort that should be put into finding the answer before reporting back to the PO and the rest of the team. It is also the time limit for running the specific spike before the outcomes are evaluated.
iii. **Product owner engagement:** The PO should understand the importance of the question, as well as the impact that various answers may have on the overall project or product lifecycle and quality. Only then can the application of spikes be successfully completed, and the outcomes evaluated for accuracy or importance.
iv. **Time and project size:** The time allotted for the team to run spikes should be sufficient for them to be successful. Furthermore, project size is important, with a large project differing from a small one in terms of the number of development teams, project requirements, artefacts, and complexity.
v. **Clear communication:** The PO should ensure throughout the lifecycle that the development team is clearly communicating expectations and that stakeholders are kept up to date on current progress and associated risk.

vi. **Clear goal:** Prior to beginning the process, the goals of the spike should be clearly defined and related to what is to be unblocked. In addition, the amount of time invested should be agreed upon within the team.

vii. **Necessary preparations:** Preparations include all aspects of team development, as well as the ability of the teams to adapt to new settings, new technology, and new organisational structures.

According to the participants, the majority of the factors are organisation-specific, and each team has its own priorities in terms of which factors to consider. While some may prioritise organisation-related factors, others will focus on success factors related to the team and the nature of the project being undertaken as significant. One of the participants stated that "the development team should try their best to be self-organising while working". In a nearly identical response, a participant mentioned that technical capability should be considered before beginning to implement spikes in ASD projects.

7.3.4. Reliability analysis

In assessing the reliability of the scale used to measure the five statements pertaining to success factors, Cronbach's alpha was used. The five-point Likert scale yielded a Cronbach reliability coefficient of 0.57 for the five items. Although not high, this value is acceptable considering the small number of items and implies that the scale used measures the statements provided with a 57% precision (see Table 7.6).

Table 7.6: Reliability analysis

Reliability Statistics		
Cronbach's alpha	Cronbach's alpha based on standardised items	No. of Items
0.570	0.575	5

7.3.5. Inferential analysis

The last five questions in the questionnaire used a five-point Likert scale to measure the level of agreement among the participants concerning different success factors for spike applications. The summary statistics indicate that the participants generally agreed with four of the five statements and were neutral concerning one. Table 7.7 lists the five statements and their respective mean responses and standard deviations.

Table 7.7: Summary responses of participants

Item	Mean Response	Std. Dev.
Q5. Openness to the problem statement and the solution is a factor to consider when applying spikes	4.50	0.713
Q6. Experience of the software development team is important for the correct application of spikes	3.94	0.957
Q7. Awareness of time constraints and investment are important factors when applying spikes	4.22	0.951
Q8. Clarity of the issue to be solved by the spikes is a vital factor that determines the successful application	4.41	0.886
Q9. Project metrics such as budget and quality are essential success factors in the application of spikes	3.22	1.161

As can be seen from Table 7.7, most of the participants agreed that openness to problem statements and solutions is an important factor to be considered when using spikes. Notably, 39 participants strongly agreed with the statement and 19 agreed. Five participants were neutral on the issue, and only one disagreed. Similarly, 44 out of the 64 participants agreed that experience in software development is important when applying spikes. Another 16 participants were neutral and 4 disagreed. Considering awareness of time constraints, 55 participants agreed that it is an important factor, while 5 were neutral, and 4 disagreed. As shown in Figure 7.26, 55 participants also agreed that clarity of the problem to be solved is important when using spikes to achieve beneficial outcomes, in contrast to 5 who were neutral and 4 who disagreed. Finally, only 15 participants agreed that project metrics, such as the budget and quality, were a crucial factor for spike applications, with most being either neutral or disagreeing.

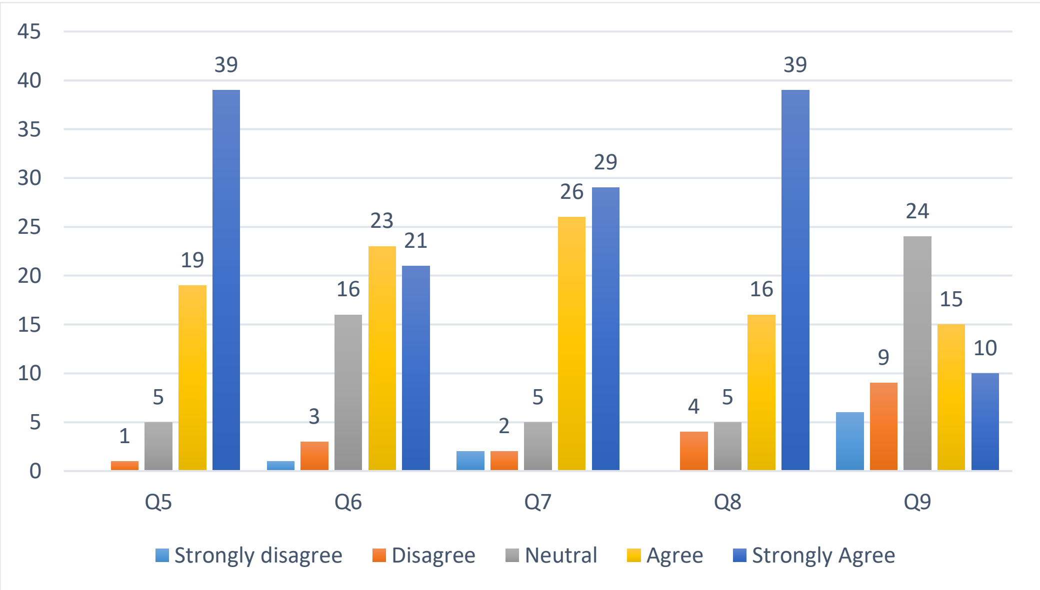


Figure 7.26: Summary of participants' responses

Figure 7.26 and Table 7.7 show that openness to the problem statement, experience, awareness, and clarity are important factors in achieving positive results when using spikes. However, project metrics, such as budget and quality, have little bearing on the success of spike application, according to the opinion of those surveyed.

Correlations

On further analysis of the five statements, some correlations among the independent factors were established, as presented in Table 7.8. In particular, openness to problem statements and solutions (Q5) shows a significant positive relationship with experience in software development (Q6) (r=0.362, 99% confidence interval [CI]). Similarly, openness to problem statement and solution (Q5) shows a significant positive relationship with clarity of issues (Q8) (r=0.277, 95% CI). These relationships imply that whenever openness to the problem statement increases, clarity of the issues to be solved is also enhanced.

Table 7.8: Correlation analysis

		Q5	Q6	Q7	Q8	Q9
Q5	Pearson Correlation	1	0.326**	0.141	0.277*	0.038
	Sig. (2-tailed)		0.009	0.268	0.027	0.763
Q6	Pearson Correlation	0.326**	1	0.277*	0.161	0.312*
	Sig. (2-tailed)	0.009		0.027	0.203	0.012
Q7	Pearson Correlation	0.141	0.277*	1	0.100	0.215
	Sig. (2-tailed)	0.268	0.027		0.431	0.088
Q8	Pearson Correlation	0.277*	0.161	0.100	1	0.283*
	Sig. (2-tailed)	0.027	0.203	0.431		0.024
Q9	Pearson Correlation	0.038	0.312*	0.215	0.283*	1
	Sig. (2-tailed)	0.763	0.012	0.088	0.024	
	N	64	64	64	64	64

** Correlation significant at the 0.01 level (2-tailed). * Correlation significant at the 0.05 level (2-tailed).

In a similar positive relationship, the experience of software development (Q6) was found to have a significant relationship with awareness of time constraints (Q7) (r=0.277, 95% CI). Q6 also demonstrated a significant positive relationship with project metrics (Q9) (r=0.312, $p<0.05$, 95% CI). Finally, Q8 demonstrated a significant positive relationship with Q8 (r=0.283, 95% CI). These findings imply that when one factor is improved, it has a positive impact on the other factor with which it has a significant relationship. That is, when a team's experience in software development is high, the team's awareness of time constraints is also high. Therefore, when one factor is considered, it is possible to consider other factors related to it or to be influenced by the factor considered. In general, the findings imply that while various factors can be considered, some should be considered concurrently for the successful application of spikes.

One-sample t-test

A one-sample t-test was used to compare the mean differences for the individual questions to the total mean score for the five questions. The mean response to the five questions was determined to be 4.0 (corresponding to 'Agree' on the five-point Likert scale). Testing the difference between each question and the mean value reveals that the majority of the participants agreed with most statements. For the same reasons discussed in section 5.5.4, Bonferroni correction is not incorporated in this test.

Table 7.9: One-sample t-test

	Test value = 4.0					
	t	df	Sig. (2-tailed)	Mean Difference	95% CI Lower	95% CI Upper
Q5	5.612	63	0.000	0.500	0.32	0.68
Q6	-0.522	63	0.603	-0.062	-0.30	0.18
Q7	1.841	63	0.070	0.219	-0.02	0.46
Q8	3.669	63	0.001	0.406	0.18	0.63
Q9	-5.383	63	0.000	-0.781	-1.07	-0.49

Table 7.9 shows that Q5 (M=4.50, SD=0.713) differs from the mean response in a statistically significant way (t=5.612, p=0.00). With a p-value (sig.) less than 0.05, it can be concluded that the majority of participants agreed that openness to problem-solving is an important factor to consider when using spikes. However, Q6 (M=3.9, SD=0.957) showed statistically insignificant results (t=-0,522, p=0.603). Since the p-value is greater than 0.05, it can be concluded that there is no significant difference between the mean response for Q6 and the total mean response for the five items. Thus, most of the participants also agreed with the statement that the experience of the software development team is important in spike applications. Similar to Q6, Q7 (M=4.22, SD=0.951) showed a statistically insignificant difference when comparing the means (t=1.841, p=0.07). The p-value is greater than 0.05, indicating that the mean difference from the overall mean is not significant, and a mean difference of 0.219 from the overall mean indicates that a substantial number of participants agreed that awareness of time constraints and investment are important factors in spike applications.

The results for Q8 (M=4.41, SD=0.886) were significant, with a p-value less than 0.05. This indicates that there is no significant difference from the overall mean response. The mean difference is positive (0.406). This value shows that the response for Q8 corresponds with "agree" on the Likert scale. Thus, most of the respondents agreed that clarity of the issue to be solved by the development team is a critical factor in applying spikes effectively. Similarly, Q9 (M=3.22, SD=1.161) yielded statistically significant results. However, the mean difference is -0.781, indicating that the mean response for Q9 is less than 3.5, i.e., lower than the overall mean. This means that the majority of participants were either neutral or disagreed with the proposition of the importance of project metrics, such as budget and quality, for the effective application of spikes. Thus, it is difficult to infer with certainty that the project budget and quality are important factors in the success of spike applications.

7.4. Discussion

The results from both the interviews and questionnaire provide significant information concerning the application of spikes in ASD. Notably, they focused on examining the success factors that are applicable when using spikes in ASD, as well as the categorisation of the factors. This section discusses the findings in detail in relation to existing research.

The success of a spike in ASD is measured by how well the spike achieved its goals. This can be assessed by the amount of learning accomplished, the amount of time saved, the accuracy of the estimation, the quality of the work produced, and the satisfaction of the team members and stakeholders. This was mentioned by the participants in different places throughout this thesis, including sections 5.6.3 and 8.7. Furthermore, it was demonstrated earlier in section 7.2.

A group of 16 individual practitioners and 3 focus groups with a total of 26 participants were interviewed, it was established that there are several common factors that agile teams consider when they are applying spikes. However, these factors vary based on the type of project, approaches used, and other organisational or team-based decisions. The client's needs are also a key determinant of how success factors are taken to account during spike applications. With the spikes intended to provide information for architectural decisions or point to which solution is better for a specific problem (Wirfs-Brock et al., 2015), agile teams are increasingly reinventing how they are applied by considering certain success factors that apply to their organisations or teams.

The findings showed that most participants considered some factors when they used spikes (see section 7.2.1). Among those factors are timeboxing and sufficient time. The term "timebox" refers to establishing a specific timeframe, whereas "sufficient time" denotes allowing the team enough time to complete the task. Constraining the team to a limited timeframe to spike many stories may not be feasible and may potentially lead to incomplete or undesirable outcomes (Arcos-Medina and Mauricio, 2020). Furthermore, clear objectives and expectations are essential for successful spikes. The objectives determine the direction of specific efforts or actions, whereas the expectations are the anticipated outcomes of the actions or efforts. In ASD, objectives are guided by the project's complexity and the client's needs while the expectations guided by the role spikes are supposed to fulfil. In addition, the findings showed the importance of documentation in using spikes effectively. Although “working software over comprehensive documentation” is one of the agile manifesto’s core values, it does not imply that there is no place for documentation in agile development, but rather that the documentation should assist the software. Therefore, architecture documents that are lengthy and hefty do not fit agile development’s flexible, lean, and basic approach (Hadar et al., 2013). According to Selic (2009), documentation in ASD should be devoid of unnecessary technological details and tightly linked to application ideas and specifications, including design reasoning and selected design alternatives. Moreover, the findings revealed communication is essential in bringing all stakeholders on board and ensuring unanimity in making critical decisions that may disrupt the product development cycle and potentially the product outcomes. When the product development schedule is being prepared, the application of spikes is considered one of the unplanned activities (Ogle, 2019). Everyone on the team will be aware of the expectations and the objectives of the spikes if the intentions and decisions to use spikes are clearly communicated. However, it is not necessary to include all team members in the spike’s application process, and it should be approved by the PO (Hunt, 2018). The complexity of communication increases when the ASD team is large. This impacts the dissemination and sharing of information between the team members. Thus, smaller groups are preferable since fewer channels are involved in passing the necessary information (Lalsing et al., 2012). Some practitioners in the data obtained believe that running safe-to-fail experiments during spike applications is a good way of ensuring enough room for adjustment if the results are not satisfactory or applicable to the problems being addressed. According to the UK National Health Service (NHS, 2021), a safe-to-fail experiment is a small experiment that approaches a problem from various perspectives/angles while allowing any emerging possibilities to be observed. Such findings highlight the importance of these factors when spikes are supposed to be used effectively.

The participants pointed out CI/CD and unit testing as success factors for agile spikes because they provide a way to test the new code and provide feedback on the code's quality. This feedback can be used to improve the code and ensure that the spike successfully meets its intended goal. Unit testing helps to ensure that the code is doing what it is supposed to do and that it is working as expected, while CI/CD automates the process of software integration and deployment, which helps teams build and deploy code faster (Arachchi and Perera, 2018). This makes it easier to find and fix any issues that may arise and helps to ensure that the spike is successful, as participants opine.

On the other hand, the factor of creating a motivated or passionate team seems to argue that organisations or team leaders ought to motivate the team to go the extra mile of using spikes for investigative purposes to achieve better results. Therefore, the factor is likely to promote team morale when designing and executing a spike, increasing the team's chances of success.

Among the common factors discussed above, the study findings identified five broad categories based on the aspect being considered. These categories include organisational, people-related, process-related, technical, and project-related factors. However, some factors can be classified under more than one category based on the given team’s opinions. For example, team expertise can be both a technical and a people-related factor. The factor is classified as people-related when only skills are considered in terms of the team's expertise. However, when practitioner of spiking is considered, the factor is classified as technical. As Conboy et al. (2011) noted, the people- and process-related factors are inextricably linked and sometimes confused. This limitation makes it essential to clearly understand what factors come under each of the five categories of success factors in spike applications.

Finally, the researcher should accept participants' subjective opinions about what they perceive as success factors. This is certainly a valid approach, as it allows the researcher to see what participants perceive as important in achieving the objectives of spikes successfully since these factors may vary from one practitioner to another and everyone has his/her own idea of what contributes to success.

7.5. Summary

This chapter has explored the common success factors impacting the application of spikes. This was achieved through conducting semi-structured interviews and focus groups with practitioners in software development. Also, the study employed a questionnaire to collect quantitative data from practitioners. The practitioners involved had considerable knowledge of ASD and the application of spikes in different countries worldwide.

The interviews and focus group meetings were conducted online due to distance and COVID-19 containment measures. There was one session with each focus group and individual participants involved during the interviews to determine the CSSFs. The process used semi-structured questions that provided the scope to ask more questions based on the interview context and to obtain more information from the participants. All the interview proceedings were transcribed to facilitate qualitative data analysis. The interviews involved 16 individuals, and there were 3 focus groups, each with 3–4 members. A list of the CSSFs discussed in this chapter was identified from the responses.

A survey was also undertaken online to collect information from a wider sample, comprising 64 participants in total. The questions were both closed-ended and open-ended. Based on the questionnaire responses, the study demonstrated that the effective application of spikes might be influenced by various factors related to the project, process, organisation, people, and the technical aspects of the software being developed. The participants further affirmed that these categories are applicable in most cases, although others argued it is not necessary to consider these factors. Nonetheless, the study established that the CSSFs influence the application of spikes in an ASD project.

Chapter 8: Validation of the Roles, Efficiency, Efficacy, and Common Success Factors of Spikes using Case Studies

This chapter discusses the findings of case studies conducted with participants from all over the world. There were three focus groups with a total of nine agile practitioners. The three focus groups included development teams from organisation A, organisation B, and organisation C. All of them were engaged in identifying CSSFs in Chapter 7, except the participants from organisation C (Indian team). Also, the individual practitioners are the same who were involved in identifying CSSFs except for CS_P5, and CS_P15. The case studies aimed to validate the previous findings related to RQ1, RQ2, RQ3, and RQ4 by seeking agile practitioners' opinions about the different agile spike issues covered in this research. By validating the results from previous chapters, the researcher can ensure that the conclusions of this study are accurate and reliable. The objectives of case studies were achieved through direct and indirect interaction with development teams participating in this study, as stated in chapter 6. Focus group interviews were conducted in two separate sessions, while the individual participants were in one session, as described in section 8.1. This was after information was ascertained to ensure eligibility, including demographic details showing the agile and spike experience of the participants. The sessions started when the level of experience of the participants with agile spikes was confirmed to be satisfactory. Figure 8.1 summarizes the case study process.

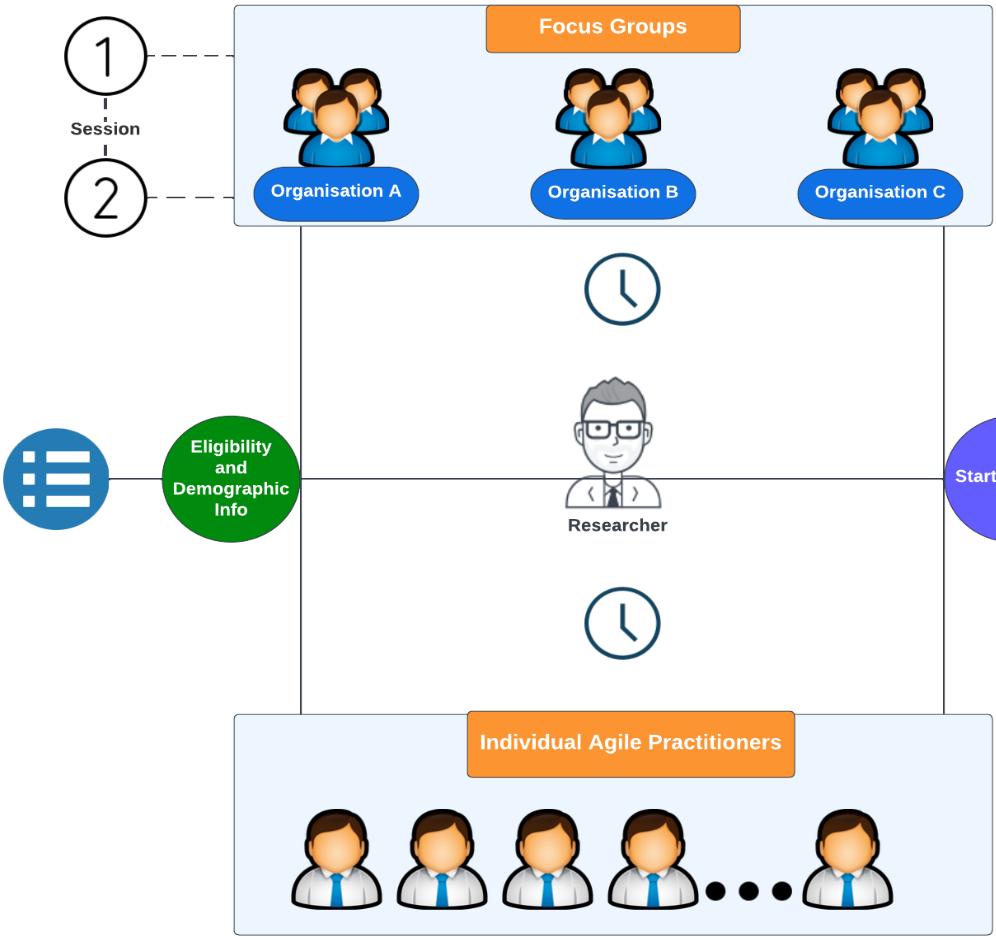


Figure 8.1: An overview of the case study

8.1. Organisation and Participant Selection

This case study aimed to validate the findings obtained during the study by seeking practitioners' opinions and perspectives on various concerns regarding agile spikes covered in this research. As stated previously, the three focus groups were from various organisations in different countries, and the members of each focus group worked at the same organisation.

Since this study was conducted virtually due to COVID-19 restrictions, it was not possible to gain consent from the organisation to participate in any of the stages of development under the pretext of protecting their privacy. The three organisations that opted to participate were asked to give written informed consent based on guaranteeing the confidentiality of the information they provided. The following is an overview of these organisations:

- Organisation A is a cloud-based PBX service provider for business communication. The head office is in Munich. The organisation was established in 2006, and it has subsidiaries and a partner network in Europe. The Portuguese branch is in Lisbon and employs approximately 435 people.

- Organisation B is a software development company with 60 employees based in Leamington Spa, UK. Since its inception in 1997, it has supplied custom software development, IT strategy consulting, and UX/UI design solutions to small, midmarket, and large enterprises.
- Organisation C is an Indian digital solutions firm founded in 1994. It is headquartered in New Delhi and has more than 1,200 employees. It offers digital engineering and technology solutions in the United States, Central Europe, the rest of Europe, and beyond. The organisation provides product architecture design, user experience, agile development, software development, IT operations, rapid prototyping, and cloud migration.

The remaining participants came from various sectors and regions, but they all worked in software development and had varying levels of experience, as shown in Tables 8.1 and 8.2.

Table 8.1: Summary of focus groups who took part in the case studies

Organisation code (Focus groups)	Country	Practitioner code	Focus group	Job role(s)	Years of experience in agile	Years of experience in spikes
Organisation A	Portugal	FG1-1	1	Developer	2	1
		FG1-2		Developer	5	5
		FG1-3		Engineering manager (Team leader)	11–15	10
		FG1-4		Tester	5	4
Organisation B	UK	FG2-1	2	Scrum Master and developer	7	5
		FG2-2		PO, developer (Team leader)	9	7
Organisation C	India	FG3-1	3	Agile coach (Team leader)	9	9
		FG3-2		Developer	4	2
		FG3-3		Developer	3	2

Table 8.2: Summary of individuals involved in the case studies

Practitioner code	Country	Job role (s)	Years of experience in agile	Years of experience in spikes	Interview duration (mins)
CS_P1	US	Agile coach	7	7	91
CS_P2	UK	Agile coach and developer	16–20	16–20	76
CS_P3	US	Scrum Master	3	3	71
CS_P4	Spain	Scrum Master	6	3	62
CS_P5	Switzerland	Scrum Master	11–15	7	58
CS_P6	India	Scrum Master	5	3	134
CS_P7	UK	Engineering manager	6	4	95
CS_P8	Canada	Agile coach	10	5	85
CS_P9	US	Agile coach	11–15	11–15	82
CS_P10	UK	Scrum Master	3	2	88
CS_P11	US	Agile coach	16–20	16–20	92
CS_P12	Australia	Developer, Scrum Master, agile coach	11–15	11–15	86
CS_P13	Germany	Agile coach	5	3	93
CS_P14	US	Scrum Master	11–15	5	89
CS_P15	Italy	PO	11–15	11–15	141
CS_P16	Germany	PO	7	5	82
CS_P17	Belgium	Scrum Master	5	5	64

The duration of the interviews and focus groups varied based on the follow-up questions and participation in some development sessions, as presented in Table 8.3.

Table 8.3: Duration of each focus group session

Organisation code (Focus groups)	Country	Session	Duration (mins)
Organisation A	Portugal	1	88
		2	67
Organisation B	UK	1	65
		2	62
Organisation C	India	1	84
		2	77

The mean duration of the focus groups was 147.66 minutes (SD=18.14 mins). The mean duration of the interviews with the individual participants was 87.58 minutes (SD=21.92 mins), as shown in Tables 8.4 and 8.5.

Table 8.4: Duration of individual interviews

Interview duration	Mins
Mean	87.58
Standard deviation	21.92
Minimum	58
Maximum	141

Table 8.5: Statistical measures of focus group durations

FG duration	Mins
Mean	147.66
Standard deviation	18.14
Minimum	127
Maximum	161

Some participants take on various roles in their organisations, as can be seen in the case of CS_P2, CS_P12, and some participants in the focus groups. Only two individuals employed the SAFe methodology, while the rest were using Scrum, as shown in Figure 8.2.

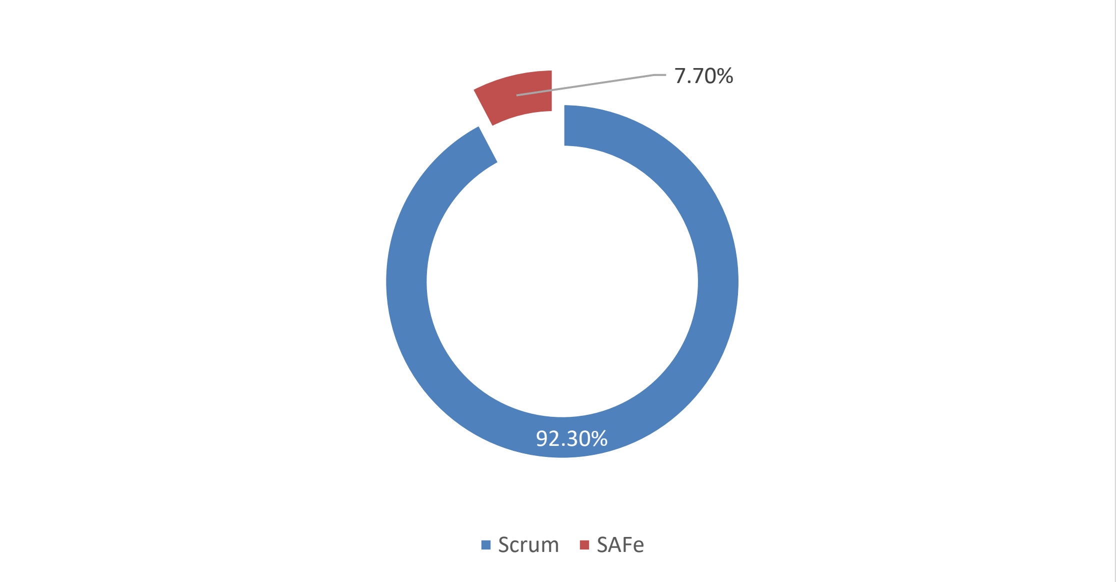


Figure 8.2: The agile methodology used by the participants in the case studies

8.2. Use of Architectural, Design Spikes and Spike Solutions in Resolving Technical and Functional Issues

Architectural and design spikes, as well as spike solutions, are forms of technical and functional spikes, as discussed in Chapter 1. The participants were asked whether these spikes could encompass technical and functional spikes and the 18 participants affirmed that indeed they could. The following are some of their most interesting responses. CS_P10 stated, “Yes. Architectural and design spikes, and spike solutions can resolve technical and functional issues and can contain technical and functional spikes. Design spikes can start off as looking at the design but become more functional as the point of the spike is to gain knowledge, so as knowledge is gained these could lead to more functional and technical spikes being created/investigated”. FG1_4 supported this claim, “Yes. They can resolve some technical and functional issues. Or they can solve all the pre-existing issues. But I’m sure new issues will come around in the future as the years pass by”. CS_P8 has the same opinion, “Architecture spikes, design spikes, and spike solutions could resolve technical and functional issues and can contain technical and functional spikes. However, I believe every spike should be small and specific enough to deal with more granular unknowns. A spike could trigger a different set of spikes, though”. CS_P13 affirmed what stated by CS_P8, CS_P10, and FG1_4, “If defined properly, then yes. Architecture spikes, design spikes, and spike solutions can resolve technical and functional issues, and can contain technical and functional spikes.”

In contrast, six participants disagreed that these spikes could be used to resolve technical or functional issues. Ideally, they considered that these issues should be solved by technical or functional spikes. Notably, some objected based on the notion that spikes cannot contain other spikes. Some of their responses included the following. CS_P9 mentioned “No, spikes cannot contain other spikes. A spike is a discrete type of work item, a peer of a story (backlog item that can be implemented within a single sprint). If I were to assign a hierarchy, a spike would belong to a feature”. (CS_P15) supported this claim, “A spike can’t be recursive (i.e., contain another spike) also because this will create unnecessary dependencies. Don’t confuse a spike with technical debt handling.”

Most responses showed consensus on whether architectural spikes, design spikes, and spike solutions could be considered subtypes of technical and functional spikes. Although some practitioners did not differentiate between these categories, they acknowledged the roles of each type in practice. Therefore, it can be concluded that these are forms of technical and functional spikes, and hence they can be used to intervene in resolving technical and functional issues. The following are some participants’ opinions on the classification of spikes. CS_P1 stated, “I think that any of those could be spikes. I don’t tend to distinguish between types of spikes”. FG2_2 supported this claim, “We would use all of these spikes in different circumstances, regardless of their classification”. CS_P6 emphasized that spikes can be categorised, “At a high level, spikes can be categorised. For instance, the functional spike – is used when there is a lot of uncertainty about the impact on the end-user”.

8.3. Roles of Spikes in Previous Software Projects

To authenticate the roles of spikes determined in Chapter 5, the participants were asked to identify the specific roles spikes have played in their previous and ongoing projects. Among the roles identified by the participants, there were seven key aspects, each of which is addressed in turn in the following paragraphs.

8.3.1. Risk management

Risk management involves the process of identifying, assessing, and controlling threats to a software development project. The participants reported that this task is part of the role of spikes fulfilled in an ASD process. Notably, two of the participants stated that spikes have been used to carry out this task in the development process from their practice. Their responses included the following. CS_P12 said, “Spike performed a risk management role in my last project. Risk management — remove unknowns early, giving time to adjust to unfavourable outcomes, and avoid surprises at the end”. Also, CS_P17 stated, “The purpose of spikes in my previous projects was primarily to reduce uncertainty/risk and allow the team to be more confident in planning work for the sprint”.

Notable responses from the participants regarding the role of understanding unknowns showed that spikes are widely used to explore unknown concepts or processes in the development process. These are some of the most relevant responses. FG1_1 affirmed that spikes could use for new technology, “The spikes were mainly used to understand/select new technologies that might be helpful for the project, or to get used to a specific technology already known by the company (but not yet applied in the project”. Whereas CS_P8, emphasized that spikes are to understand unknowns, “The role spike played in my previous project was helping us to understand the small unknowns, risks, and tech applications”. CS_P9 also supported those claims “I have used spikes in the past, with my own teams, and with teams that I’ve consulted and coached. Spikes have been used to answer a question, to resolve an ambiguity. As you know, spikes come from the XP world”.

The role of spikes in understanding unknowns was also noted by CS_P10, CS_P13, and FG1_2. They all confirmed that spikes are used in understanding unknown details, technology, or information about an ASD project.

De-risking new dependencies in software development is the process of ensuring that any new software dependencies are thoroughly investigated and evaluated before they are implemented. This process involves identifying any risks associated with the new dependencies, understanding the impact of these risks, and mitigating them. This can include conducting security scans and penetration tests, auditing source code, and implementing security controls. By de-risking new dependencies, organizations can ensure that their applications are secure and reliable (Cox, 2019). Regarding this, the participants interviewed pointed out that spikes are used to accomplish this role. For instance, CS_P16 said, "The role of spikes in my last project was to ensure that vendor-provided services/API worked before developing stories that relied on them". In addition, CS_P14 stated, "Spikes were responsible for validating technological solutions in my recent projects".

These responses illustrate that spikes are not only used for POC and investigation, but also for risk management. By understanding unknowns, it becomes easier for the development team to control the impending risks.

8.3.2. Ensuring smooth and sustainable running of the project

Another role of spikes found to be practised in contemporary software development projects was to ensure the smooth running of the project. In the earlier chapters, it was established that spikes are useful in estimating user stories and researching. This function allows the software development project to run smoothly since decisions are made easily and seamlessly. In this case study, many participants gave their perspectives in this regard. CS_P1 stated, "Usually, spikes are used to give additional focus to backlog refinement. They identify specific questions or concerns that need to be addressed to continue with refinement". CS_P12 stated another role of spikes, "Productivity and quality gains – here's a better way of doing something that will save us time or deliver improvements on non-functional areas". In addition, CS_P15 remarked that the spikes are also used in enhancing compatibility, failure recovery, and load balancing to aid the smooth running of the project. FG1_2 commented that spikes are useful in helping to resolve integration issues that may arise. Thus, the case study confirms that spikes play a role in ensuring the smooth running of a software development project through their various uses in the process.

8.3.3. Proof of Concept (POC)

The role of spike in relation to POC was mentioned by several participants during the case study interviews. For instance, FG1_3 stated that "The role spike played in my previous project was creating POC and MVP to explore new solutions or tech solutions difficult or impossible to estimate". CS_P11, CS_P7, and CS_P16 made the same point. CS_P7 stated, "The role spike played in my previous project was prototyping and POC". Whereas CS_P16 said, "The role spike played in my previous project was prototyping and POC". CS_P16 confirmed these claims, "In my last project, spikes were responsible for reviewing and evaluating the new technology as a proof of concept".

8.3.4. Investigations

Investigations in ASD involve checking the feasibility of the solution in terms of the scope and objectives of the software being developed. Some of the participants in the case study noted that spikes are used to carry out this process. Their responses pointed out that procedures related to prototyping and proof of concept are part of the investigations that spikes can undertake. For instance, CS_P13 said, "Spikes assessed if my previous project was doable to identify potential blockers, the timeline, and skill set required". Also, CS_P4 stated that they were using spikes to investigate a legacy project or collect required new features. CS_P11 commented that spike played essential role in experiments, research, prototyping, proof of concept in their previous projects.

8.3.5. Decision-making

Spikes play a significant role in decision-making by providing information to stakeholders and the development team. Teams use the knowledge gleaned from the spiking process to make informed judgments in areas where there is no consensus due to lack of information. Some participants stated that this role has a significant impact on the activities of software development projects. For instance, FG1_4 said, "The main role was to help kick off our project. Whenever we are re-designing or building completely new software or changing a critical implementation, spike findings are important to reach a consensus between the team members". FG2_2 stated, "Usually, exploring technical uncertainty gives greater insight into the scope of work or the possible solutions. Suggested direction is important when the outputs present multiple options".

8.4. Effectiveness of Spikes in Fulfilling their Roles

As noted in the previous section, the participants identified several roles of spikes in the ASD process. This gives rise to consideration of how effective spikes are in satisfactorily completing these roles. The participants were asked this question, and the majority noted that spikes are very effective. Figure 8.3 presents a summary of the responses.

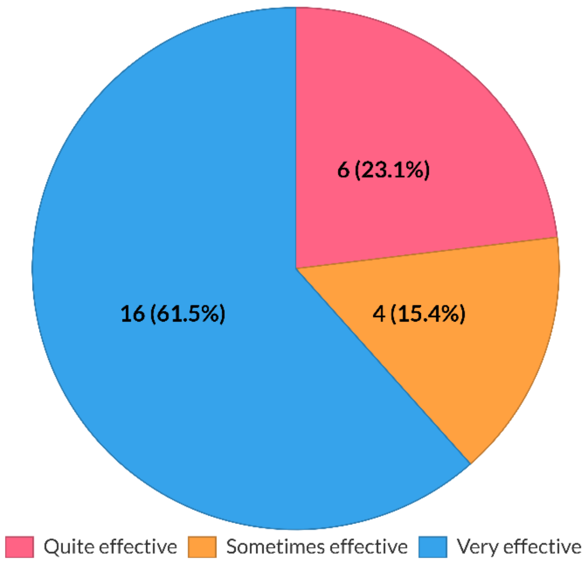


Figure 8.3: Participants' opinions concerning the effectiveness of spikes in fulfilling their roles

The majority of respondents, who confirmed that spikes are very effective in the roles they play in the development process, had the following to say, "It took a few tries to get it right, but once we did, it proved to be very useful, especially because it gave the developers a chance to think about the project holistically to provide data-driven decisions." (CS_P13). CS_P15 stated that spikes always clarify doubts, highlighting the path to follow. Even when a spike fails, it's successful learning as it prevents costly mistakes. Also, CS_P6 said, "We would have failed miserably without the spike, which is an essential part of the whole plan. In my view, if a team is succeeding without a spike, I think they aren't developing anything challenging, or they may have a bunch of people who are highly experienced and are capable of handling any uncertainties (very few teams are like these)". In addition, FG2_1 emphasized that spikes are effective in their previous projects by saying, "Spikes were very effective. They gave a clear view of the work that needed to be achieved with better confidence in estimating the work, which resulted in better predictability of work and forecasting".

Six participants were of the opinion that spikes are quite effective when applied correctly, two of whom had the following to say, FG1_1 "The spikes were pretty effective. With them, our team was able to acquire knowledge for those technologies, and some technical decisions were made according to the output of the spikes". FG3_1, commented on the effectiveness of spikes by saying, "Spikes were quite effective because they could clearly indicate if something would work or not". CS_P1, CS_P3, CS_P4, and CS_P16 expressed the same.

Four participants were not entirely satisfied with the effectiveness of spikes in the various roles, saying that they are only effective sometimes. CS_P3 stated that' "Spike varies in the effectiveness, depends on who performs the spike". CS_P12 commented on the effectiveness of spikes by saying, "spikes are effective when well-defined and appropriately resourced; combined with good visibility of the activity throughout the team including the project owner".

These responses suggest that although spikes are essential in fulfilling the roles they are designed for, they are not effective in all roles. Sometimes, their effectiveness is dependent on who is applying the spikes and whether appropriate resources are available.

8.5. Effectiveness of Spikes in Risk Management and Estimation

The participants in this case study noted that spikes are useful in risk management and the estimation of user stories and other project facets. Again, the issue is their effectiveness. In the interviews, the majority affirmed that spikes are effective in risk management and the estimation of user stories. The summary responses are shown in Figure 8.4.

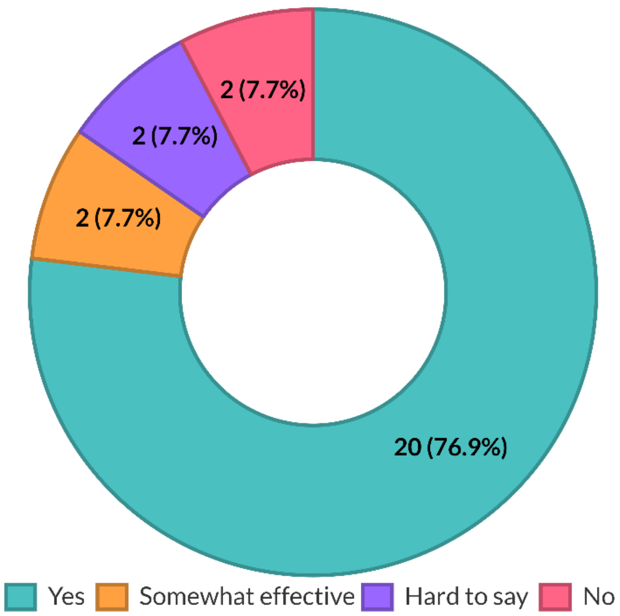


Figure 8.4: Participants' opinions concerning the effectiveness of spikes in risk management and estimation

As can be seen, the majority agreed that spikes are effective in risk management and estimation. Their responses were based on past experience and knowledge of using spikes to perform these two functions. CS_P10 commented that the spikes are very effective in risk management and estimation, but their effectiveness is based on different factors. Risk management was not a desired direct outcome of the spike; however, it was a by-product as better understanding leads to reduced risk. CS_P14 stated, "Yes. Spike worked very well in risk management and estimation. Spike allowed us to verify things before writing user stories that depended on them, so we were able to solve dependencies before pulling work into a sprint". CS_P4 affirmed that spikes are effective by saying, "Yes. Spikes are very effective in risk management and estimation. We avoided writing code without knowing what to do. Without a proper spike, you can't properly estimate and reduce the risk". These claims were supported by CS P7 and CS FG1 2 as well. However, a few participants considered that the effectiveness of spikes is difficult to quantify, and it is hard to say whether they are effective or not. For instance, CS_P17 said, "Hard to say. I think most times a spike gives the team more confidence in their estimation, but I have no data on whether it actually has made a difference". Also, CS_P11 stated, "Spikes might reveal unforeseen risks or might be used to determine if particular risk mitigation would help. However, quantifying this is impossible". Furthermore, two other participants viewed spikes as not effective in risk management and estimation. CS_P9 pointed out that spikes are not effective in risk management and estimation. They are wasteful but less wasteful than the mistakes that would occur if upstream work was undone and spikes weren't done. Also, FG1_4 said, "Depends on the context. For instance, working with the same tech stack definitely is a plus for the developer/tester when estimating stories. But spikes themselves aren't enough to be effective in risk management".

The responses show that most developers consider spikes effective in risk management and estimation. Although the experience with spikes is not the same for every developer, their varied views still point to the potential effectiveness of spikes in risk management, whether directly or indirectly.

8.6. Skills Required for Development Teams to Apply Spikes Effectively

Although the skills factor was identified and classified as among those influencing the successful application of spikes, the participants' perspectives on this factor differed. The majority believed that specific skills are required to utilise spikes effectively, while some felt that no such skills are needed. The following paragraphs detail some of the skills highlighted by participants.

8.6.1. Technical skills and maturity

The first skill set was related to the team's ability to handle technical issues and adapt to new technologies in line with the software project being developed. Some of the notable responses included the following. CS_P8 stated that the development teams need specific skills related to the context. If it's an API integration, the team needs to build that skill. CS_P10 affirmed that technical skills would help the development team use spikes properly.

8.6.2. Being open-minded

Accepting others' opinions and learning from them is essential to accommodate potential risks. In addition, constructive criticism can refine the developer's personality, and learning from mistakes is required to complete the spiking process successfully. Some participants supported this skill. For instance, FG1_1 emphasized that the development team needs to be open-minded and willing to learn. In addition, FG1_4 affirmed that the development teams need to be humble to accept opinions and positive criticism.

8.6.3. Communication skills

As stated in Chapter 7, communication is fundamental in bringing all stakeholders on board and establishing consensus in making critical decisions that may influence product development and outcomes. Therefore, all development team members should be good at communicating with others in a friendly and smooth manner to facilitate sharing of information and experience among them, thus achieving the objectives of the spikes that have been set. Most participants agreed that this skill is critical for effectively using spikes. For instance, FG1_4 said, "Communication plays a big role here to persuade and convince team members that the solutions proposed are the best. Not only the investigation findings are important, but also the communication between all team members". FG2_2 pointed out that the development teams should be able to articulate output clearly.

8.6.4. Understanding the meaning and objectives of spikes

The agile team should be aware of the reason for employing spikes, the purposes for which they are appropriate, and the feasibility of the outcomes to decide whether to begin work on a particular feature. Once the team has a grasp of these aspects, they will be able to outline clearly the objectives of spikes with high precision, which leads to effective application. Most participants agreed that this skill is essential for using spikes effectively. FG3_3 stated, "Spikes should only be used when we are unsure how to implement a feature. Also, we must limit ourselves to research and POC rather than using spikes as a medium to build the actual feature". FG3_1 pointed out that the development teams should understand what the outcomes of the spike should be. There should be a clear distinction between what is a spike and what is a code. CS_P6 commented on this skill by saying, "The development teams should understand what a spike means, the difference between XP and SAFe (Spike and Enablers), when to apply spikes, and how to run a spike and its outcomes".

8.6.5. Time discipline

Timeboxing is the process of allocating a specified time in which to carry out the planned activities. This was discussed in Chapter 7 as one of the success factors of spikes, namely that development teams must be disciplined, especially with regard to time. Most participants emphasised the need for this in the case study sessions. For instance, CS_P12 said, "Having discipline around time spent on the spike would help the development team apply the spike effectively". FG2_2 supported this option to stick to the scope and time boxes.

8.7. Objectives of Spikes in Recent Software Projects

As explored in this study, spikes are intended to perform various roles to mitigate risks and aid discovery during software development. Therefore, realising the objectives of spikes can enhance the application to ultimately obtain valuable outcomes. During the case study sessions, several issues and questions were raised with the participants, whether in focus groups or individual interviews. Among these was the following:

Did the spikes achieve their objectives?

All members of the focus groups believed that the spikes had met their objectives in recent projects. Individual participants, with the exception of CS_P9 and CS_P17, agreed with the three groups, as shown in Figure 8.5.

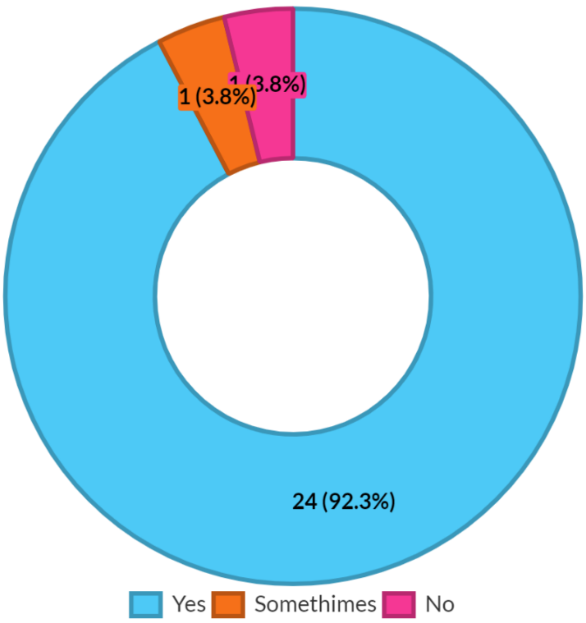


Figure 8.5: The participants' perspectives on spikes achieving their objectives

When asked if the spikes achieved their objectives, CS_P9 differed from the other case study participants, saying "No, because people don't understand what a spike is and how to use them properly. They become catch-all (buckets). A spike, properly used, is named with the question that is to be answered (e.g., 'Understand the steps needed to connect to a Jira project database in order to retrieve and update individual work items'), is timeboxed (e.g., 'Up to 4 days'), and is resource-constrained (e.g., 'One developer'). The spike ends when either the question is answered, or the time expires".

It was possible to attend some sprint planning and retrospective sessions during the case studies and observe some user stories and PBIs to which the spikes were applied. These sessions were with focus groups and some individuals involved, as detailed in the following subsections.

8.7.1. Organisation A (FG1)

Organisation A was the first in which the case study approach was applied through focus group interviews. As shown in Table 8.1, the group comprised four members, and their project was a development of a portal (back office) for the clients to configure and handle their cloud PBX. The objective of the spikes in this project was discussed with the team, creating POC and MVP to explore new solutions or tech solutions difficult or impossible to estimate. Observing some user stories and spike use during sprint planning and retrospective sessions when allowed to attend was possible. Figure 8.6 presents an example user story discussed in those sessions.

As a user, I want to be able to configure the PBX system so that I may make phone calls without any assistance from support.

Context:
Configuring a PBX system is very complex and technical. Users must be able to configure their extensions, call forwards, sites, and IVR easily and without external help.

Acceptance criteria:
Customers should access a web portal where they can configure everything.

Figure 8.6: Example of a user story – FG1

This user story was discussed during the sprint planning session, through which the work team was determined to carry out the spikes during the next sprint (the team typically allocated extra time for a spike in case they needed to apply it). During the retrospective session, the spiked user story illustrated in Figure 8.7 was observed.

Spike:
Use a visual representation of the system to configure settings.

Context:
Customers continue to request a visual representation of the system's set up to make configurations easy and ensure a pleasant user experience.

Timebox: 3 days

To do:
Stage 1: Display a visual representation of the current configuration.
Stage 2: Visual configuration editing.

Figure 8.7: Example of spike – FG1

8.7.2. Organisation B (FG2)

The second focus group was initially reticent about providing information regarding their software projects. However, it was possible to attend one of their development sessions to gain some information. The team was working on establishing new billing and payment services for insurance and service products. The objective of spikes in this project was to evaluate the potential of various third-party providers in an ID&V integration. The team needed to determine how they might assist in meeting the business objectives of the project, considering the practicalities of technical integration. The user story that was observed during the case study to achieve the spike objectives is illustrated in Figure 8.8.

As a customer, I want to request a refund.
An agent needs to be able to raise a refund against a collected payment to pay the money back to the customer via Vendor.

Context:

- A customer is raising a complaint and as a part of the mitigation process, an agent is initiating the payment refund process.
- Given that the customer has raised a complaint, the agent is creating a payment refund request for collected payment.
- When the agent is initiating the payment refund, the collection date for settlement is earlier than today (this is driven by the collected settlement, not the pending settlement).
- The refund payment authorisation record is created in the payment authorisation service.
- The payment identifier is required to show this is a refund.

Acceptance criteria:

- The payment method is set to the payment method of the parent payment.
- The amount is set to the amount of the parent payment.
- The refund payment authorisation record is linked to the parent payment and to the complaint.
- The refund reason is captured and added to the payment authorisation record.

Figure 8.8: Example of a user story – FG2

In a further discussion with the PO (team leader) about the spike objectives, he stated that clear objectives are essential for spikes. Otherwise, the spikes will be ineffective. FG2 added the spike in one of their sprints to understand the potential options for refunds based on user stories that need to be spiked, as shown in Figure 8.9.

Spikes:
Present options and recommendations on how refunds can be made to customers.

Context:

- Considerations as to how cancelation fees (as part of overall ad hoc payments) should work.
- Considerations as to how refunds (as part of overall ad hoc payments) should work.

Timebox: 4 days

To do:
As developers, we want to understand the potential options for how refunds (with overall ad hoc payments being paid out to the customer in mind) should work in the future to develop/deliver the best solution possible for the customer and business needs (ad hoc payments include refunds and goodwill).

Figure 8.9: Example of spike – FG2

8.7.3. Organisation C (FG3)

FG3 was working on migrating from an existing identifying tool in a banking sector to a new adaptable solution. This was a compliance project with a huge user base. They were utilising Scrum methodology to manage their development process. There was an informative discussion with this team concerning their previous and ongoing project to review the spike objectives and how they could lead to positive outcomes. It was possible to attend two sessions (sprint planning and retrospective) to make some observations and highlight the information obtained. Among the objectives of spikes in this project was to choose between two open-source technologies for fulfilling some functions in software. Some examples of user stories and spikes were observed in this project, as shown in Figure 8.10.

As a user, I want to get the OTP on my mobile so that I can paste it directly into the portal.

As a system administrator, I want to confirm the user's identity before migrating via OTP so that fake users cannot migrate on anyone else's behalf.

Context:

- This application is highly secure, and it is really important to identify the person and authenticate his/her identity.
- The user will be sent an OTP which will populate in the top bar of the phone from where he/she can copy and paste it directly with one click.

Acceptance criteria:

- The user must get an OTP on the phone.
- The OTP should populate in the top bar in the centre.
- The OTP should have a hint that this OTP is retrieved from messages.
- On clicking this OTP, it should be pasted in the OTP text box.
- The feature must be available on Android and IOS.

Note:

- The look and feel should be as per the wireframes given by UX.

Figure 8.10: Example of a user story – FG3

A discussion that took place with the team leader in the presence of two members of the development team was observed that covered certain user stories, such as the one seen in Figure 8.10, as well as the potential outcomes of spikes. This user story was added to the sprint backlog by the development team in preparation for inclusion in the next sprint, which could include spikes if necessary. The spikes to be examined in the next sprint process are presented in Figure 8.11.

Spikes:
OTP auto-retrieval and population.
Context:
After completing the user experience test, we got feedback that the auto-retrieval feature is something that customers really wish to have. We had never done it in the past and thought of delivering a feature to our customers that they would like.

Timebox: 3 days

To do:

- Explore the possibility of implementing this feature on both platforms - Android & IOS.
- Explore if we have a plug-and-play kind of feature that is readily available.
    - Does this plug and play support all our devices at minimum up to iPhone 5?
    - If this plug-and-play support does not work, how long will it take to create this custom feature?
- How will we ensure that the custom feature is thoroughly tested?
- Present the findings to the team.
- Based on the option that the team chooses, create follow-up tasks.

Figure 8.11: Example of spike – FG3

In a discussion of the outcomes of the spikes with the development team in the retrospective session, they confirmed that the objectives were achieved successfully.

8.7.4. Individuals' experiences of spikes to achieve objectives

Some individuals in this study were permitted access to some sessions during their team development processes. However, this access was limited to communicating with those specific participants since the other team members did not take part in the case study discussions.

Practitioner CS_P15

CS_P15 was one of the participants who gave access to attend a sprint planning session with his team. CS_P15 works in a large software development company in Milan, Italy. As indicated in Table 8.2, the participant, working as a PO, has a wealth of experience, which is reflected in the study's information enrichment. This participant believed that spikes are not associated with user stories, stating, “In my approach, there is no correlation between a user story and a spike. It is meant to acquire clarity and know-how before putting a story in the sprint backlog. If a story is not ready you can't estimate it and therefore you can't commit to it”.

Among the perspectives expressed by CS_P15 was that a spike is an item on its own, not strictly bound to a story. While a story contributes to the product increment, the spike does not because it contributes to an increase in knowledge within the team. It is important to define what spikes are and why they are carried out (objectives). They should always provide measurable facts, not biased opinions. Moreover, each PBI must pass the definition of ready (INVEST) check. This also means that dependencies should be removed beforehand. It is unwise to estimate and commit to a story if the team still has some doubts (either technical or functional) to be clarified through a spike.

It was possible to attend a sprint planning session after completing the case study questions. CS_P15 and his team were developing an integrated process control and maintenance system (IPCMS), and the scrum methodology was employed to carry out the project. This involved an integrated, multi-site maintenance service in which both the customer and the service provider would share a huge quantity of data about every piece of equipment under contract (pumps, motors, valves, etc.). The data would be used both for process monitoring and predictive maintenance. During the sprint planning session CS_P15 and his team wrote the following objectives on a whiteboard without setting up another project/control process:

1. Minimising the codebase differences (possibly to a single ‘connector’ class).
2. Realise the POC on the hosted stack and use it as reference implementation (RI).
3. Export the DB on AWS-RDS and check its consistency with the RI-DB.
4. Realise the new ‘connector’ class for AWS-RDS.
5. Realise the 2nd POC pointing to AWS-RDS.
6. Showcase the results with a particular focus on performance, auditability, and total cost of ownership.

I discussed these objectives with the PO and noted the following details:

- Number 1 was a non-functional requirement.
- Numbers 2, 3, and 5 were chores.
- Number 4 was a technical spike.
- Number 6 was a functional spike.

It was observed that CS_P15's team marked items 4 and 6 in the main backlog as ‘not examined’ in preparation for being examined in the next sprint. The other items were to be absorbed in the slack allocated to chores. These activities had to be completed within the sprints without affecting the velocity.

From the standpoint of CS_P15, an AWS RDS solution looked interesting for many reasons: scalability, cost predictability/transparency, reliability, security, etc., making it worth exploring. The mission spikes were to indicate how much it would take to build a POC showing two back-end services with 100% equivalence (same API, same business logic) pointing to two similar datasets, one Hosted-MySQL and one AWS RDS.

Practitioner CS_P6

CS_P6 collaborated by giving permission to attend a planning session after completing the case study questions. CS_P6 worked in a branch of a well-known software company located in Hyderabad, India. The participants' team was using the SAFe methodology to develop a BLE-enabled Android and IOS mobile application for a health care product. During the planning session, the team used technical and functional spikes. Before starting using spikes CS_P6's team undertook an analysis/feasibility study to see if the spike would be feasible or whether there might be a better alternative available: If the proposed alternative were straightforward to implement, it would not spike; however, if the suggested alternative was also something needing a POC, then a new spike story would be created. The team had a discussion with solution architects (SAs) to decide on the final approach (if a better alternative was available). The SAs paired with the developers and drove the execution completing the analysis. At this stage, the focus was more on getting the solution. The coding standards/quality guidelines were not strict as this was mostly a POC. When the POC was successfully completed, a demonstration of the operational POC was delivered to the PO to gain an initial opinion. CS_P6 and his team made a few changes based on the PO's comments. Finally, a demonstration was done with all architects and stakeholders, and the following integration steps were decided based on further comments. CS_P6 noted that the POC outcome is usually treated as an input for the spike. However, they combined the spike and POC to bring more value as they provided working software for visualisation and the feasibility study or analysis. Commenting on combining spikes and POC, CS_P6 remarked "Although many will argue that not all spike stories require POC, it made sense in our case because it was a product, and we needed more than a theory to make better decisions". Figure 8.12 illustrates an example of a POC-based spike observed while attending a planning session.

Functional spike:
A custom vertical slider for controlling the volume tuning. (A slider the user interacts with manually by moving vertically up and down.)

Context:
As a user, I wish to control the volume settings from my smartphone.

Timebox: 2 days

Acceptance criteria:

- The slider should be vertical in alignment.
- The slider should have a total of 10 steps
- The slider should be interactive on both Android and IOS platforms.
- The slider should be programmatically configurable to a specific value without any user interaction.
- The slider should also show a tooltip displaying the value at the centre.
- The slider should follow the design provided by the UX team (attached in the user story).

Figure 8.12: Example of functional spike – CS_P6 project

Of particular note is that the timebox for the spike shown in Figure 8.12 should not exceed more than 20% of the team's capacity (10 days sprint, so 20% equals two days for one developer). Also, if the technical feasibility fails due to technical challenges, it is highly recommended that the development team propose an alternative approach (mindful that the sprint goal is not at risk).

The functional spike exemplified here was less complicated than the technical spike presented in Figure 8.13, which used 20% of the team's capacity (8 story points) since the team velocity was 40 story points. If the estimation of the spike exceeds 20% capacity, the team should discuss it with the PO to help prioritise the spikes over other user stories in the following sprint.

Technical spike:
Enabler as using SAFe

Context:
Update the PCL to .NET Standard for wider support and integration of better performing libraries.

Timebox: 4 weeks (2 sprint)

Acceptance criteria:
- The project should be converted to .NET Standard
- All the NuGet packages should be updated to support .NET Standard 2.0.
- All the existing functionality of the application should work as expected.
- Minor UI issues are acceptable within the scope of a primary fix or workaround that can be implemented in a short period of time (not more than 2 weeks).
- The plugin component should also be converted to .NET Standard or made compatible for consumption.
- A detailed document on the PCL to .NET Standard should be available (reviewed by architects).
- The Software of Unknown Provenance (SOUP) list should be updated with all the latest library details, brought up to the present to support .NET Standard 2.0.

To do:
- Analyse and convert the existing PCL to .NET Standard.
- Resolve all the errors that may appear post-conversion.
- Perform a sanity, smoke test to validate if any existing feature is breaking or if everything is working as expected.
- Upon successful conversion, an in-house UAT will be scheduled to get feedback on the overall functionality.
- Team X (confidential) will perform regression to ensure the overall quality and functionality are the same as exists in PRODUCTION.
- Finally, prepare a document about the overall procedure and steps taken to convert to .NET standard.

Figure 8.13: Example of technical spike – CS_P6 project

The technical spike was used to evaluate the impact of new technology on the existing implementation since the project was based on a portable class library (PCL). As PCLs were to be phased out, .NET Standard was preferred because it provides better support. All .NET Standard APIs are supported on every platform, and with each version there is more API coverage and stability.

CS_P6's team had PI planning with a projection of the plan for the next three months with the milestones/releases planned since the technical spike was complex in nature and the target release was post six months; a timebox of 2 sprints was set with the consent of the PO.

According to CS_P6, the timebox for a spike depends on many factors:

- Complexity, the amount of work, risk, and uncertainty.
- Milestones the spike will be part of.
- Priority in the product backlog.
- Team's capacity, external dependency, and technical capability.

Although the experiences of the development teams interviewed in the case study sessions differed, it was observed that the spikes are essential for reconnaissance solutions and providing alternatives when the team encounters technical or functional issues in all software project facets. However, the observation sessions were not analysed because they did not adequately represent the population of interest and the sample size was small. Therefore, the results of this data would not be sufficiently general.

According to Hunt (2018), the development team responsible for carrying out the spike should decide what the spike should be after approving it by PO. The success of the spike should be measured by how well it achieves the objectives set out in the spike. For example, the objectives may involve researching a particular technology or identifying a solution to a problem. The success of the spike should be judged based on how well these objectives are met, as demonstrated in chapter 7.

8.8 Total Number of Spikes Used in Sprint/Iteration and the Project

A matter of specific interest lay in knowing the total number of spikes used in each sprint/iteration or the entire software project. It was not possible to be involved in all the development processes since the case studies were conducted online. The number of spikes was extracted directly from the question asked to the participants and indirectly through the observation. The responses varied among the participants due to several factors, the most important of which was the complexity of the project being developed and the technology used. In contrast, a few of the individual participants did not explicitly state the actual number. Tables 8.6 and 8.7 present the number of spikes used in the recent project in which the participants were involved.

Table 8.6: Number of spikes used by the focus groups

Participant code	Number of spikes in sprint/iteration	Number of spikes in the entire project	Number of sprints/iterations for each spike	Comment
FG1 (Organisation A)	1	4–5	Not applicable	
FG2 (Organisation B)	1	2–3	Not applicable	
FG3 (Organisation C)	0	4-5	6–10	Rarely used

Table 8.7: Number of spikes used by the individual participants

Number of spikes	Sprint/iteration			In the entire project		
	0–1	**1–2**	**2–3**	**1–5**	**10–15**	**15–30**
Practitioner code	CS_P5 CS_P17	CS_P3 CS_P7 CS_P12 CS_P5 CS_P8	CS_P1	CS_P6 CS_P8 CS_P12 CS_P13 CS_P16 CS_P17	CS_P3 CS_P5	CS_P2 CS_P15 CS_P7
Number of sprints/iterations for each spike						
Practitioner code	**2**	**3**	**4**	**5**	**6**	**7**
	CS_P6 CS_P12	CS_P14	CS_P11	CS_P4	CS_P10	CS_P16

CS_P9 was dissatisfied with the use of spikes and thus did not declare the numbers that were used in his recent software development, justifying this by saying, “Spikes were useful in resolving ambiguity. The reason I stopped using them is that, too often, spikes are used as crutches, as a way for a team to work on a backlog item without having to fully account for the additional effort. The desire was to have a spike for every backlog item. Spikes were disguising the fact that these backlog items weren’t effectively understood and refined… they became a lazy way out for the PO and the team to not do their upfront work, and they degraded throughput. So, spikes were an imperfect solution to the problem of ambiguity… they were a way of managing the problem instead of eliminating it by having a more robust refinement process at the project and team levels. When we disallowed spikes, it forced teams to work on the elaboration and refinement of requirements… and significantly increased both quality and throughput”.

8.9. Use of Spikes during the Sprint/Iteration Retrospective

The demographic information gathered during the case studies revealed that the participants used either the Scrum or SAFe methodology. Both have sprint/iteration retrospectives that aim to review and discuss the results and practices carried out in the previous sprint/Iteration and identify ways to make improvements in the following sprint/iteration. According to Hunt (2018), a sprint retrospective involves evaluating what happened throughout the development and release process and discussing ways to improve productivity in the following sprints/iteration. For this reason, the participants were asked whether they ever used spikes during retrospective sessions. They all agreed that they did not do so, except CS_P5, which confirms the information presented in Chapter 5 concerning the appropriate time of spike usage. FG1_3 said, "No, we did not carry out a spike during the sprint retrospective. However, we discussed issues that arose when carrying out spikes in order to become more effective at using them". Also, CS_P11 emphasized that spikes cannot be run during retrospective sessions, "That's not how a retrospective works. We might review the way we did the spike to see what we learned about our approach. But a retrospective is designed to continually improve the process, not to run the process". FG3_1 supported this claim by saying, "We did not carry out a spike during the sprint retrospective. We create cards on the backlog for spikes and timebox the amount of time we have. The team carries out this work like business as usual, although a team member is usually allocated in advance, which is not our usual practice".

8.10. Defining the spike objectives in the project planning stage

Spikes are used for different purposes during the software development process. In the case studies, the participants were asked whether spikes were used to carry out roles specified in the project planning. In response, most confirmed that spikes' roles or objectives are not planned before their need arises. Thus, in project planning the roles of spikes are not specified. The opinions of the participants are summarised in Figure 8.14.

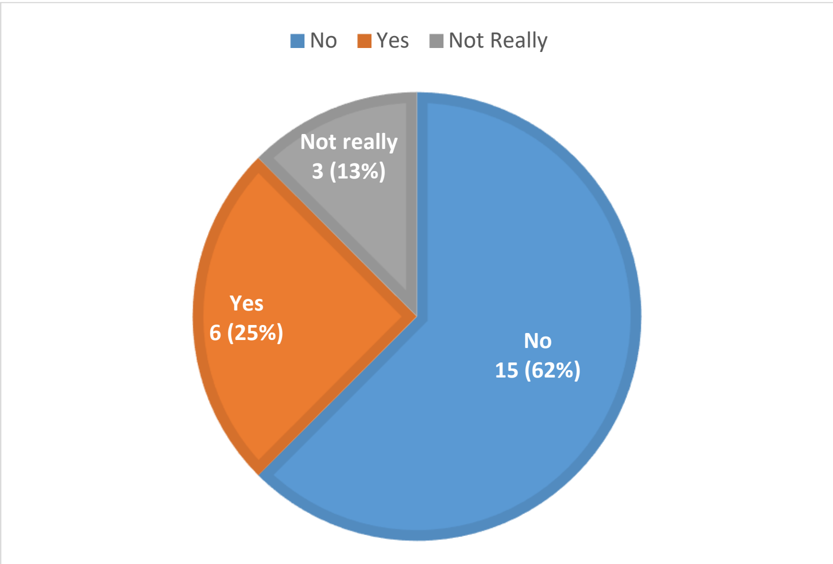


Figure 8.14: Participants' perspectives on the use of spikes in project planning

Based on the responses, most spiking processes are not planned beforehand. As some of the participants noted, spikes are only used to identify unknowns when they arise. There is no specific role in the plan that needs the use of spikes. However, spikes can be applied when more information is needed, or a decision needs to be made. Most of the participants, including CS_P2, CS_P3, CS_P4, CS_P8, CS_P13, CS_P14, CS_P16, FG3_2, and FG2_2, responded by saying that "No, spikes are not allocated to carry out the tasks that were created during project planning.". These responses contrast with the opinions of six other participants who mentioned that spikes are used to carry out specified tasks that have been created in the project planning session. Notable responses that are contrary to the views of the majority included the following. FG1_2 said, "Spikes are used for carrying out roles specified in project planning. Usually, spikes are identified during the previous sprint, and in the planning of the upcoming sprint, we specify the need for it and who should tackle it (developer, PM, PO, Scrum Master)". Also, FG2_1 stated, "Sometimes yes, spikes are used for carrying out roles specified in project planning. Spikes are used at the start of the kick-off of new demand and projects".

Others were not sure whether spikes were used to carry out roles specified in the project plan. Notably, CS_P11 stated that "Spikes are not part of any roles specified in the project planning as there can be other methodologies used. However, I can see the application of spikes in risk management, Implementation, and many other areas, but it depends on how much time is really allocated to each activity". This response illustrates that although spikes are necessary, it is not entirely the case that they carry out (or not) roles specified in the project plan. For instance, spikes can be needed in prototyping, but there may be no specific plan for their use.

8.11. Common Spike Success Factors (CSSFs) for Effective Application

The common success factors for the application of spikes identified by participants were discussed in Chapter 7. However, it was necessary to confirm the suitability of these factors during the case study sessions by presenting them to the participants for examination and thus validate the most common factors that influence the application of spikes.

The mechanism for validating the factors was by delineating each category with its factors as previously identified for discussion separately with each participant to ensure that they were not inclined toward certain factors. As shown in the results presented in Chapter 7, five categories were discussed with the focus groups and individuals during the case study sessions. Measurements were undertaken to reflect the compatibility rate for each factor, represented by the number (percentage) of participants who agreed on the relevance of a specific factor. Each participant was asked whether a factor influenced the application of spikes and could be considered a factor in their success in terms of effective application. The participants' responses varied, with some supportive (Yes) and others not supportive (No). Table 8.8 summarises the analysis of common success factors, along with the scores assigned to each factor based on participant consensus.

Table 8.8: Validation of common spike success factors (CSSFs)

Factor category	Common Spike Success Factors (CSSF)	Approved (No.)	Overall Score (%)
People level	Capability of the development team	23	88
	Clear communication	25	96
	Collaboration	26	100
	Team skills, knowledge, and expertise	19	73
	Team morale and motivation	20	77
	Cross-functional team	18	69
	Team autonomy	21	80
	Team inclusiveness	16	61
	Willingness to ask for help	17	65
	Being open-minded	22	84
Organisational level	Acceptance of agile methodology	25	96
	Access to resources	17	65
	Willingness to invest time	24	92
	Stakeholder expectations and management	20	77
	Organisational diversity	14	54
	Organisational support & politics	21	80
	Running safe-to-fail experiments	19	73
Development process level	Following agile-oriented requirements	26	100
	Timeboxing	23	88
	Focusing on specific questions	18	69
	Setting clear goals	23	88
	Clear definition of 'done'	20	77
	Right use of spikes	17	65
Project level	Setting realistic goals and timetable	23	88
	Sufficient budget	16	61
	Effective training	24	92
	Access to resources	17	65
	Use spikes as an investment	24	92
Technical level	Team's technical ability and expertise	21	80
	Access to appropriate tools and resources	17	65
	Team's knowledge	15	58
	Technical training	24	92
	Understanding architecture limitations and guidelines	20	77

8.12. Tools for Mitigating and Managing Risk in Agile Software Development (ASD)

Among the questions raised with case study participants to corroborate some observations was whether their organisation used other risk management techniques. Figure 8.15 depicts the difference between "Yes" and "No" responses and demonstrates that 19 participants reported using other risk management techniques to deal with hazards in agile development.

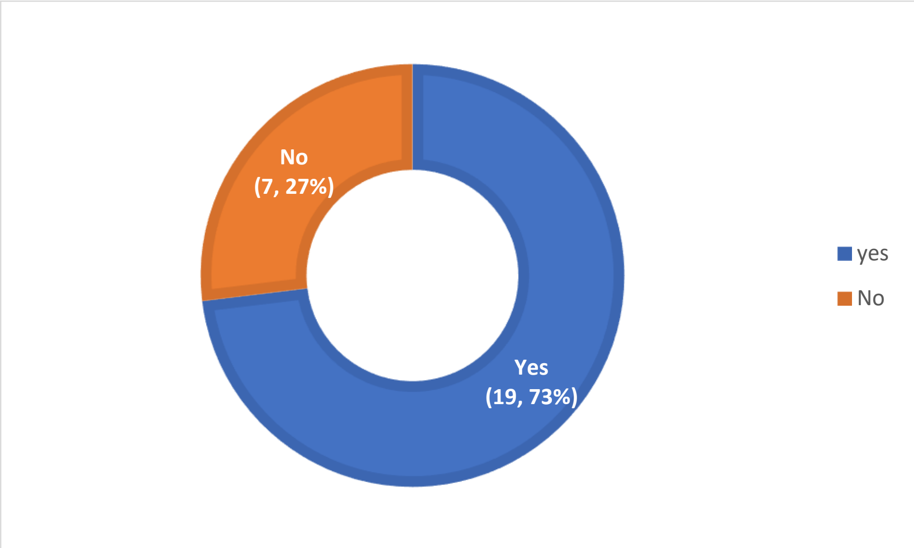


Figure 8.15: Use of other risk management techniques by participants

Table 8.9 depicts the tools/techniques mentioned by the participants.

Table 8.9: Other risk management tools/techniques used by the participants

	Risk management tools used in agile development
Participant code	**Tool/technique**
Organisation A (FG1)	Sprint retros
Organisation B (FG2)	Break work into independent features and PMO criteria
Organisation C (FG3)	RAID and FMEA analysis
CS_P1	Maintenance of a risk register, with red/amber/green tracking for likelihood
CS_P2	Early and regular delivery
CS_P5	Risk burndown chart
CS_P6	Qualitative and quantitative analysis
CS_P7	Design and architecture reviews, and workshops with businesses and SMEs
CS_P8	Pre-mortems and POCs
CS_P9	Project- and team-level refinement and MVPs
CS_P14	Statistical forecasting (based on Monte Carlo)
CS_P15	Automated tool as per PMI practice
CS_P17	Dependency mapping during PI planning

Some participants shared some approaches and mechanisms they employed in their organisations to mitigate risks during software development. For instance, FG1 said, "In our company, we use the sprint retros to find solutions for the problems identified during the sprint". In addition, FG2 stated, "The other techniques that we follow are: addressing technical uncertainty early, breaking work into independent features of about a week's work, and features into independent stories of about a day's work, considering security from design onwards, and applying PMO criteria against deliverables". FG3 added, "Our company uses RAID analysis, FMEA analysis, and release forecasting to kill the risk of not meeting the deadline as it was a compliance project". CS_P9 said, "Project- and team-level refinement is key to reducing the risk of uncertainty derailing a project". Among other risk management techniques/tools mentioned by the participants are qualitative and quantitative analysis as CS_P6 stated, "Our organisation has many risk management techniques including qualitative and quantitative analysis. The five methods are most used: Avoidance, retention, sharing risk, transferring risk, and loss prevention". CS_P14 added, "We use statistical forecasting (based on Monte Carlo)". Furthermore, CS_P8 stated, "We utilise pre-mortems, but usually uncover the unknowns, and usually trigger the creation of new spikes and/or PoCs". Also, CS_P15 pointed out that there is an automated tool in place to assess and monitor assumptions, risks, issues, and dependencies according to PMI practice. CS_P2 indicated that early and regular delivery is essential for risk management.

It can be observed that the majority of the participants use other risk management techniques to identify, prioritise, and refine the risks in the sprint backlog before the development teams put the items into a sprint. In other words, these techniques can be used during sprint planning to make PBIs ready and eligible for the sprint backlog. However, these practices and techniques do not provide practical solutions for discovering risks and new technologies during the sprint as spikes do.

8.13. Most Significant Challenges Impeding the Effective Use of Spikes

During the interviews, the participants were asked about their challenges while using spikes during their software development projects. Several factors emerged from the responses, detailed as follows.

8.13.1. Lack of clarity in goals

For spikes to be applied effectively, there should be clear goals that will guide their design and time of application. Based on the participants' responses, unclear objectives are a hindrance to the effective application of spikes. In response to the question, CS_P1 mentioned that "Not knowing what question to ask or what is necessary to learn, not being clear on what the spike should do leads to challenges". This view was shared by CS_P10, CS_P12, CS_P14, FG2_2, and FG3_1. Moreover, CS_P17 noted that "Not framing the question correctly is a challenge to the effective application of spikes". Thus, without clear goals, it is challenging to apply spikes successfully.

8.13.2. Changes in requirements

Some changes can occur in the product design during the development process based on the client's or PO's decision. The participants highlighted such changes as a potential challenge in applying spikes. CS_P15 said that "Unforeseen releases owing to strategy alterations (urgent vs important) is one of the hurdles to the correct application of spikes". In the same line of thought, FG1_4 noted that "…product changes during an undergoing project can compromise the technical implementation of spikes and require additional investigation. This can hinder the effective application of spikes".

8.13.3. Inadequate comprehension of spikes

As mentioned in Chapter 7, the development team's understanding and skills are essential in effectively applying spikes. Thus, lacking an extensive understanding of spikes is challenging for their application. According to the participants, failing to understand why spikes are used is a precursor of incorrect application in the development process. For instance, CS_P12 said, "For spikes that develop code, there is a temptation to morph that code into a prototype, a partial implementation, or even to get reuse as production code. This is rarely a good idea. (1) It generally leads to slowing down the spike, as production concerns such as test coverage or particular libraries come into play. (2) It often gives poor quality results when code that was rushed into existence to answer a spike question has insufficient quality or performance to be effective as production code. (3) It distracts attention from answering the question, as the developer concentrates on delivering something useful, not on being certain the spike question has been rigorously addressed". CS_P2 stated, "Ignorance is a challenge to the correct application of spikes. Spikes are a simple idea. Despite that, some people are unable to grasp the idea. If you understand why we might choose to do spikes, it is pretty hard to fail and misuse them. So, the only hindrance to the correct application is not knowing what the correct application is". FG3_3 supported these claims by saying, "A lack of knowledge about the feature for which we are performing the spike poses a difficulty to the correct use of the spike".

8.13.4. Organisational culture

Organisational culture refers to the regulations and procedures that a team/organisation follows in all its actions. For spike applications, the participants have pointed out that this can be a challenge in applying spikes correctly. For instance, CS_P8 stated, "The cultural aspect – when people feel uncomfortable estimating something wrong because they may feel they will be punished for it, they call for multiple spikes. This can hinder the correct application of spikes". In an autonomous team, there is the freedom to innovate ideas to gain more information about the process, which might support the spike process. However, there might be strict adherence to protocol and procedure in a conservative team, and the sense of being punished may lead to the team not using spikes when needed.

8.13.5. Lack of a cost-benefit analysis of spike application (no metrics to assess)

A cost-benefit analysis (CBA) is a systematic approach to evaluating the advantages and disadvantages of various solutions. It is used to identify which solutions provide the best way to benefit while saving money (Mantei and Teorey, 1988). In this context, CS_P6 emphasised the significance of CBA, which may pose a challenge to the effective application of spikes, "Sometimes it's hard to provide data on the cost saved or the benefit spikes implementation. For example, using a new library version with better support enables the team to implement faster and easy maintenance. So, to prove this, the team has to collect details of old library implementation, wait time, and all the details, which is not as easy as fetching details from SONARQUBE to display code quality".

The following are the remaining factors highlighted in the previous chapter that may be a challenge to the effective application of spikes:

- Lack of discipline
- Poor communication among the members of the team
- Stakeholder expectation
- Lack of consensus among team members.

8.14. Summary

This chapter has presented the validation of findings for RQ1, RQ2, RQ3, and RQ4 through discussions with 26 participants in software development from various sectors and countries. Around 75% of those who took part were from the IT sector, while the others came from finance and banking, energy, consulting, and government sectors. The case studies were conducted virtually with three focus groups from three organisations and 17 individuals from various countries across the world, verifying the roles of spikes, effectiveness, and success factors.

Since the case studies were carried out virtually due to COVID-19 restrictions, the organisations that were invited to participate did not permit involvement in all stages of the development process. However, it was possible to take part in some developmental stages with the three organisations and with some individual participants to gain more insight into the application of spikes and their success factors.

To achieve the objectives of the case studies, were divided into five stages: preliminary evaluation of past software projects, interviews and focus group meetings, case study planning, sprint planning and application of spikes, and sprint review and retrospective. It was possible to examine the use of spikes during these stages, and validate the roles, efficacy, and common success factors. Furthermore, it has been observed that most of the risk management techniques and practices stated in this study did not fulfil the roles that the spikes do.

Chapter 9: Conclusions and Future Work

This chapter presents the overall conclusions of the study based on the research questions in 9.1. It then further discusses the contributions in 9.2 and implications in 9.3. It outlines the challenges and limitations in 9.4, which are in turn used to suggest proposed future work in 9.5.

9.1. Conclusions

The application of spikes in agile software development (ASD) projects has become necessary for most developers. In exploring the use of spikes in ASD, the study conducted a series of interviews and questionnaires with software development practitioners to provide information shedding light on the four research questions. In each stage of the research, a mixed-methods approach, including both qualitative and quantitative data, was employed. According to Mitchell (2018), mixed methods are of use to researchers who wish to explore research from a broad perspective and obtain first-hand information from participants. Primarily, the focus of the study was to establish the different roles spikes play in ASD, their efficiency and effectiveness in managing risk, and common success factors for their implementation. Following that, the findings were validated using case studies, as discussed in Chapter 8. The study first investigated three research questions:

RQ1: What roles do spikes play in agile software development (ASD)?
RQ2: What are the agile teams' opinions on the efficiency of spikes in ASD?
RQ3: What are agile teams' perspectives on the effectiveness of spikes in managing risk?

The second phase of this study focused solely on RQ4 and sought to identify common spike success factors (CSSFs) driving their application in ASD. Finally, the third phase of this study comprised case studies undertaken to validate the findings of the previous two studies. Table 9.1 summarises the methods used in this study.

Table 9.1: Summary of methods used in the study

Research Question	Method Used	Purpose	No. of Participants
RQ1: What roles do spikes play in ASD? **RQ2: How can agile spikes be used efficiently?** **RQ3: How can agile spikes be used to manage risk effectively?**	Semi-structured interviews (16 questions)	To identify the roles that spikes are used to perform in ASD To establish how efficient spikes are in ASD and how effective they are in mitigating risks	22 participants
	Questionnaire distributed online to selected participants (30 questions, closed- and open-ended)	To establish the efficiency and effectiveness scores based on participants opinions	72 participants
RQ4: What are the common success factors for spike applications?	Semi-structured interviews + focus groups	Identifying the factors that lead to the successful application of spikes in ASD	3 focus groups (3–4 participants) 16 individual participants
	Questionnaire distributed online to recruited participants	To quantitatively establish the factors that lead to the successful application of spikes, and then compare them with results from the interviews	64 participants
Case study (Validation of findings)	Interviews and focus group discussions	To validate the findings for RQ1, RQ2, RQ3, and RQ4	3 focus groups (2–4 participants) 17 individual participants

The participants in the interviews, focus groups, and surveys were from different countries and performed various roles in ASD. Notably, the studies included developers, Scrum Masters, SMEs, testers, POs, agile coaches, and PMs, among several others. Significant information on the roles, efficiency, effectiveness, and common success factors of spikes were identified and confirmed based on their responses.

The study findings confirmed that spikes help estimate user stories, manage risk investigations, and solve technical deficits. Key among the findings obtained for RQ1 was that spikes are necessary for solving knowns, known unknowns, and unknown unknowns. For knowns, other than estimating user stories, spikes are employed to gain a better understanding of user requirements. Furthermore, the case study confirmed that spikes are helpful when exploring new tasks and technologies, reducing uncertainty and complexity during ASD. By allowing developers to deploy spikes for research purposes, the entire development team can identify and understand unknown details of a technology or project requirement. Ultimately, spikes can be useful in improving the quality of the end product.

Aside from the roles spikes play, the participants also rated their effectiveness. However, some of the participants were of the view that the effectiveness of spikes would in practical terms depend on several factors, as mentioned in Chapter 7, the most important of which was the complexity of the product being developed. Judging from the responses obtained, clearly spikes are indeed effective in several domains in which they are used, including UX design, cloud computing, data warehousing, and information security domains. This effectiveness might be the reason spikes are widely used and recognised by developers in ASD. Although some noted that spikes are sometimes not effective, the case studies validated their effectiveness, establishing that more than 80% of participants found them to be effective in the roles they were deployed to fulfil (see Chapter 8).

As one of their roles, the study established that spikes are used to mitigate risk. Moreover, the study confirmed that spikes are effective in risk mitigation in ASD projects. About 90% of the participants scored the effectiveness of spikes in risk mitigation 4 or 5 (maximum 5), as shown in Chapter 5. This implies that the efficacy of spike application is widely visible to most agile teams who have dealt with them closely to mitigate risks. However, it is worth noting that it is possible to run an entire project without using any spikes. Thus, we can conclude that despite the different roles that spikes play, their effectiveness, and their importance in mitigating risk, it is still an optional technique that development teams can use for various purposes when needed.

The study further concludes that several factors influence the successful or effective application of spikes. For instance, timeboxing, support from the management, team expertise, access to resources, and technical training of the development team, among other factors, may influence how successful spikes are in executing their intended roles. However, it should be noted that these factors vary across development teams, and some may not consider any factors when applying spikes. Ideally, some developers rely on the product design to guide their development process and only use spikes when necessary. With these findings, it can be said that people, procedural, organisational, and technical factors influence the extent to which the application of spikes in ASD is successful.

However, as the case studies highlighted, the common success factors are not universal and thus not applicable to all teams. Each development team is bound to consider different factors depending on the project they are undertaking.

In summary, the responses to RQ1, RQ2, RQ3, and RQ4 identified through the first two studies were substantiated and validated by the results of the case studies. This was made possible by combining both qualitative and quantitative approaches to collect the empirical data needed to draw conclusions concerning the four research questions. The qualitative approach, in particular, was helpful in gathering specific information regarding the opinions of the participants based on their experience. These provided in-depth data in response to the four research questions. Notwithstanding, the quantitative methods used were also important in quantifying some of the findings to give an idea of the proportion of participants who supported a specific idea. Figure 9.1 presents a summary of the research methods and processes leading to the findings and conclusions of the whole study.

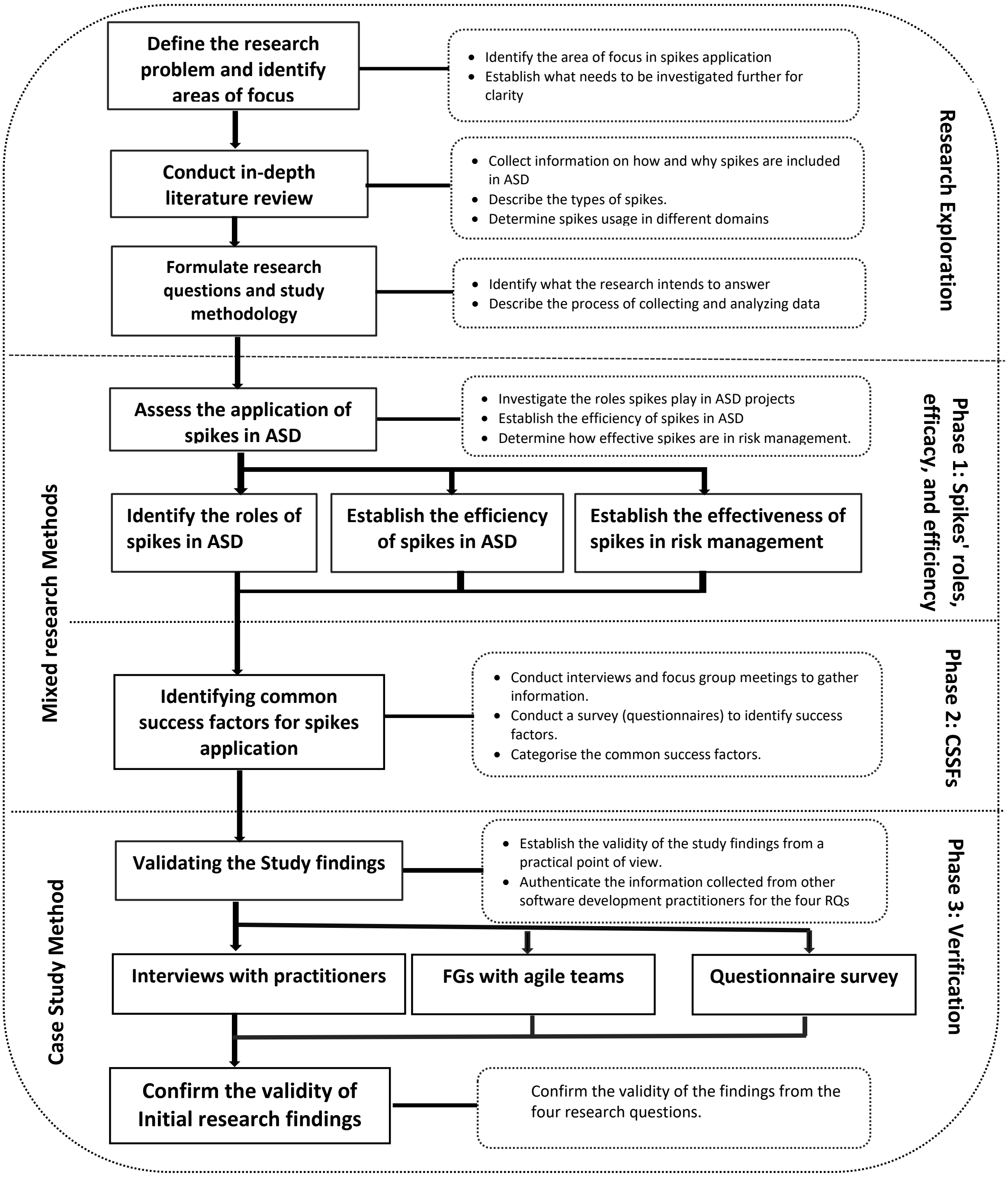


Figure 9.1: Overview of the study process and methods

9.2. Research Contributions

Research contributions aim to fill the gaps discovered during the research process. In this regard, appropriate research methods must be used to respond to the questions that have been developed to address these gaps.

The inclusion of a diverse range of agile practitioners, in terms of regions, roles, and experience, provided an opportunity to obtain solid contributions enhancing the reliability of the research. The sample was as representative as possible of the global population of ASD practitioners.

The lack of studies on the use of spikes in ASD, and the lack of information in existing studies, exemplify the need for concrete research contributions to inform this field. This study contributes to the existing knowledge of spike applications through four main contributions based on the research gaps identified in Chapter 2, each of which is addressed in turn in the following subsections.

9.2.1. Identifying the roles of spikes in agile software development (ASD)

The roles of spikes were determined during the first phase of this study to answer the first research question, which resulted in a list of roles based on practitioners' opinions supported by the literature review. The roles are not well-defined in the literature, which has led to much confusion among researchers concerning the uses of spikes and proof of concept (POC). The main roles identified in this study are the following: providing a better understanding of user stories, allowing a better estimation, reducing uncertainty, exploring new/unfamiliar technologies, and understanding the business necessity (see Chapter 5). This contribution will help enrich the knowledge content of the spikes in the agile world.

9.2.2. Efficiency and effectiveness of spikes in ASD

Although spikes have previously been researched, a literature search did not identify any empirical studies quantifying the efficiency and effectiveness of spikes. Through the interviews, questionnaires, and later case studies, this research has demonstrated that spikes are efficient and effective in risk management and estimation. However, it should be noted that this does not entirely imply that spikes will be effective or efficient all the time. In some instances, the desired results may not be obtained, which explains why agile teams need to run safe-to-fail experiments with spikes (as a success factor). Thus, this study provides additional evidence that spikes can be efficient and effective when carefully deployed with the organisation's support, team expertise, and other required technical skills.

9.2.3. Identifying common spike success factors (CSSFs)

The list of factors identified by participants as influential in the successful application of spikes was the most notable contribution determined during the second phase of this research. Various methods – interviews, focus groups, and questionnaires – were adopted to identify and develop the CSSFs. The findings bring to light how the factors identified can determine the success of spikes. For instance, the study identified that organisational factors, such as allowing access to resources, could profoundly impact how successful spikes can be in fulfilling their roles. This contribution goes a long way to illustrating that despite spikes being widely known to most agile teams, they are impacted by several factors that may influence their effectiveness.

9.2.4. Validating and evaluating the CSSFs

The third phase aimed to validate the findings obtained in the prior two studies addressing the four research questions by seeking participant opinions and perspectives on various issues concerning agile spikes covered in this research. Three focus groups from various IT organisations across the world and 17 individual agile practitioners were recruited to verify the CSSFs.

The factors were evaluated by examining each category with its factors defined in Chapter 7 independently with each participant to guarantee that they were not biased toward particular factors. The measure calculated the number of participants who agreed on the inclusion of a specific factor and then obtaining the percentage agreement. The CSSFs identified will greatly help practitioners, especially those with little experience of handling spikes appropriately, enabling them to apply them effectively.

9.3. Research Implications

The findings from the first phase present extensive information on the application of spikes in ASD. Addressing the dearth of research on the use of spikes, especially in ASD processes, this research delves deep into establishing the possible roles spikes can play and how effective they are in these roles once they are deployed. Furthermore, phase 3 provides evidence that spikes are effective in managing risk during the software development process by enhancing research and experimentation to explore different solutions to the problem identified. The information contained herein presents significant opportunities to advance the application of spikes among developers, POs, organisations, and all practitioners in general. Since the information was obtained from participants with varying levels of experience in ASD projects and the application of spikes, it reflects what agile teams do in their day-to-day activities.

9.3.1. Implications for organisations

The CSSFs identified and validated in this study give organisations a new way of looking at how spikes can be used effectively based on the factors that influence their implementation. For instance, by enabling teams to access the resources needed and training practitioners, organisations can build a resilient team capable of designing and executing any type of spike needed to resolve issues in the development process, or to identify and mitigate unknown risks. Furthermore, the findings of this study elaborate in detail, based on participants' experience, the specific roles that spikes can be used to accomplish in the development process. Finally, the study provides information on how organisations can remain flexible in addressing changes in the design and process of software development, incorporating spikes in sprints/iterations as needed. Phase 2 thus provides significant information that can guide an organisation's decision to use spikes.

9.3.2. Implications for practitioners

The implications of this research for agile practitioners are significant because phases 2 and 3 gathered data from many of them. The study presents evidence of the efficiency and effectiveness of spikes in playing various roles. Further, the CSSFs offer practitioners a much more detailed perspective of what might influence success or failure in applying spikes than previously available. Although not all factors may be applicable for all agile teams on all occasions, they nonetheless offer a possible explanation of what might improve or reduce the chances of successfully applying spikes to resolve a problem during the software development process.

9.3.3. Implications for researchers

The conceptualisation of spikes and their application in ASD has not been sufficiently covered in the literature, as it became evident during the review phase (research exploration). There are still limited empirical studies that cover this topic. This research thus adds to the existing knowledge in this area and provides a foundation for further empirical studies to be undertaken. In this regard, researchers can reference this research and possibly build upon it to explore further how spikes are integrated into ASD processes.

9.4. Research Limitations and Challenges

The successful completion of this research came with some challenges and limitations. In particular, the study faced three key challenges and had two limitations. First, data collection occurred when COVID-19 was declared, and restrictions were imposed. Therefore, it became impossible to meet the participants recruited physically, and most of the interviews and all the questionnaires were conducted online. Furthermore, during phase 3 it was not possible to access most software development activities because of this challenge.

Second, there was a challenge in finding qualified and experienced software development participants to interview. Some companies and agile teams were unwilling to participate in phases 1, 2, and 3 of this study. However, this challenge did not discourage the search for participants with practical experience of using spikes. Third, in scrutinising the responses from the participants in the interview transcripts and questionnaires to come up with the most relevant information, some responses had to be removed since they were random, did not respond to critical questions, or were from participants who had no experience with spikes. Furthermore, the time constraints meant that there would be no opportunity for follow-up or clarification from some participants. Thus, the information was excluded if a response was ambiguous or incomplete.

This research had two significant limitations. First, the questionnaires and the interviews were based on self-report data. This was more of an issue in the questionnaire since there was no way of verifying the authenticity of the responses provided by the participants. This might have influenced the findings in all phases in this study. Second, there was a lack of studies concerning the application of spikes available for review and building the statement of the problem in the existing literature. Due to this limitation, most of the findings in this study do not reflect any previous research. The findings are thus novel, but they have been validated by the case studies conducted. However, even the case studies were not done through physical interaction with the participants. Thus, this might have influenced the results of the entire study.

9.5. Future Work

As discussed in 9.4, this research has faced challenges and limitations that might have impacted the results obtained and perhaps the conclusions. It is not sufficient to conclude that spikes are efficient in ASD and effective in risk management. Future research can concentrate on determining the precise accuracy of spikes in estimating user stories and risk reduction. It would be much more feasible to conclude whether spikes are effective when clearly determining their accuracy level.

Future studies concerning spikes should be conducted in person to avoid the limitations brought by using self-report data. This would be a significant advance in ensuring the integrity of the information provided by participants. Furthermore, it would allow the researcher to incorporate observation as a data collection method and obtain first-hand information during the software development project. In this case, the results of the spike application could be reviewed in the presence of the researcher. This would provide realistic and honest data concerning whether the CSSFs were influential and if the spikes were efficient and effective in executing the intended roles.

The five categories identified in this study may not be an exhaustive list of CSSFs. Based on the responses of some participants, there might be other applicable categories. Thus, future research can also focus on uncovering additional categories to accommodate all the relevant factors and further streamline the existing five to clarify which ones belong to which category and avoid any overlap. Such information could be crucial for developers and organisations that use spikes for the various roles discussed in this study.

A significant advance would be to perform exploratory factor analysis (EFA) and confirmatory factor analysis (CFA) of the common success factors in spike applications. CFA would allow the researcher to test the hypothesis that a relationship exists between the observed variables and their underlying latent construct(s) (Thompson, 2004). As a result, the classification of the factors would be more accurate. EFA will help identify all interrelated items (in this case, the factors) without imposing any preconceived structure (Child, 1990). Such analytic methods need to be conducted to ensure the accuracy of the categories developed. With such a study in the future, a much more precise categorisation of the CSSFs will be available for researchers, organisations, and practitioners to guide their work.

In addition to what was mentioned above, there are significant opportunities to expand future work based on insights made in this thesis, which are as follows:

The decision between using A/B testing and agile spikes when introducing changes or experiments into a production environment is critical to software organisations. They both involve experiments to determine the most effective approach to a given problem. A/B testing is a method of testing in which a control group (A) and a modified group (B) are used to measure the effects of a change or experiment. By running A/B tests, organisations can identify the most successful product or service variant and make changes to ensure their product and services are optimised for the most successful user experience (Kohavi and Longbotham, 2017). Investigating A/B testing and agile spikes is important to understand the advantages and disadvantages of each approach and how they can be used together to maximise the effectiveness of experiments, as some practitioners in section 5.6.2 suggested A/B testing as an alternative to using spikes for functional risks. Furthermore, it can help organisations make informed decisions about how best to deliver experiments into production.

Conducting a Grey Literature analysis can offer a great opportunity for further research, as it can provide access to information that is not typically attainable or available through traditional sources. This is an area that warrants further exploration and could extend the literature review reported in Chapter 2. Research can expand to include reports, industry studies, and other documents that may not be found in traditional academic sources. Garousi et al. (2019) provide helpful guidelines for performing Grey and multivocal literature reviews in software engineering. These guidelines offer a systematic approach for planning, executing, and reporting Grey and multivocal literature reviews. The Garousi-style Grey Literature analysis can be used to analyse various media outlets to gain insights into how agile spikes are being discussed and perceived. This type of analysis can identify key topics, trends, and themes in the discussion of spikes, as well as any potential areas of improvement in the use of spikes, and the public's perception of them. Additionally, it can be used to identify any potential opportunities to leverage spikes to improve the development process.

Future research could explore the use of different communication methods and strategies to convey the outputs of spikes since communication is essential in bringing all stakeholders on board to ensure unanimity in making decisions concerning product outcomes, as demonstrated in sections 7.2.1, 7.3.3, and 8.6.3. (Which the researcher might have been able to investigate via case studies if it had been possible to conduct these in person). Such research could focus on the types of communication channels that are most effective for transferring information between team members, how to ensure that all members understand the results, and which methods can be used to ensure that the results are documented and shared accurately. Research into the most effective way of communicating the outputs from spikes could provide valuable insights into how the development teams can better collaborate and work together to achieve the desired outcomes. Additionally, the research could analyse how using different communication methods affects the speed and accuracy of decision-making. Finally, it would also be beneficial to have a designated person or team to be responsible for capturing the results of the spikes and compiling them into a summary report to be shared with the entire development team.

9.6. Closing Remarks

This thesis has presented the background to the research, including a review of some relevant literature, and identifying research gaps. The research methodology was explained, and the results of interviews and questionnaires have been presented and analysed in Chapters 4 and 5. Similarly, Chapters 6, 7 and 8 cover the focus group and case studies research undertaken, including the data collected and its analysis. Furthermore, the final chapter summarises the research contributions generated by the investigation. The chapter also covers limitations and suggests some possible future work.

In conclusion, my PhD experience was an invaluable opportunity to develop a broad range of research methods and technical skills. I have gained experience in designing and executing research projects, writing and submitting manuscripts to prestigious journals and conferences, and developing technical skills such as coding, data analysis, and visualisation. Also, I have gained a deep understanding of research methods, learning to apply them in multiple contexts, including the development of research instruments, data collection and analysis, the interpretation of findings, and the significance of critical and creative thinking in the development of research projects. In addition, I have acquired a comprehensive understanding of the principles of agile software development and associated methodologies. Altogether, my PhD journey not only improved my research capabilities but also provided me with practical skills that will be beneficial to me in my future career.

References

Abbas, N., Gravell, A.M. and Wills, G.B. 2008, June. Historical roots of agile methods: Where did 'Agile thinking' come from? In Abrahamsson, P., Baskerville, R., Conboy, K., Fitzgerald, B., Morgan, L. and Wang, X. (Eds). *Agile Processes in Software Engineering and Extreme Programming. XP 2008. Lecture Notes in Business Information Processing* (Vol. 9) (pp. 94–103). Berlin, Heidelberg: Springer.

Adams, K. and Lawrence, E. 2019. *Research methods, statistics and application* (2nd ed.). Los Angeles, CA: SAGE Publications.

Agarwal, N. and Rathod, U. 2006. Defining 'success' for software projects: An exploratory revelation. *International Journal of Project Management*, 24(4), 358–370.

Akem, M. 2018. *Agile and the Internet of Things (IOT)*. [Online] Available at: https://www.projecttimes.com/articles/agile-and-the-internet-of-things-iot.html

Albadarneh, A.A.I.a.Q. 2015. Risk management in agile software development: A comparative study. *IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies*.

Ambler, S. 2002. *Agile modeling: Effective practices for extreme programming and the unified process*. NJ: John Wiley & Sons.

Ambler, S. 2012. *The architecture owner role: How architects fit in on agile teams*. [Online] Available at: http://agilemodeling.com/essays/architectureOwner.htm.

Anderson, D.J. 2010. *Kanban: Successful evolutionary change for your technology business* (1st ed.). Washington: Blue Hole Press.

Appelo, J., 2016. Create a safe-to-fail environment. [online] Available at: https://www.forbes.com/sites/jurgenappelo/2016/04/03/create-a-safe-to-fail-environment/?sh=66668bbf10d8

Arachchi, S.A.I.B.S. and Perera, I., 2018, May. Continuous integration and continuous delivery pipeline automation for agile software project management. In 2018 Moratuwa Engineering Research Conference (MERCon) (pp. 156-161). IEEE.

Archibald, M.M., Ambagtsheer, R.C., Casey, M.G. and Lawless, M. 2019. Using Zoom videoconferencing for qualitative data collection: Perceptions and experiences of researchers and participants. *International Journal of Qualitative Methods*, 18, 1–8.

Arcos-Medina, G. and Mauricio, D. 2020. Identifying factors influencing on agile practices for software development. *Journal of Information and Organisational Sciences*, 44(1), 1–31.

Armstrong, R. 2014. When to use the Bonferroni correction. *Ophthalmic and Physiological Optics*, 34(5), 502-508. https://doi.org/10.1111/opo.12131

Babbie, E.R., 2021. The practice of social research. 15th Edition. Cengage learning.

Barquero, B., Bosch, M. and Gascón, J. 2019. The unit of analysis in the formulation of research problems: The case of mathematical modelling at university level. *Research in Mathematics Education*, 21(3), 314–330.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R. and Kern, J. 2001. *The agile manifesto*.

Beck, K. and Gamma, E. 2000. Extreme programming explained: Embrace change: Addison-Wesley Professional.

Becker, S., Bryman, A. and Ferguson, H. eds., 2012. Understanding research for social policy and social work: themes, methods and approaches. policy press.

Bergin, J., Kussmaul, C., Reichlmayr, T., Caristi, J. and Pollice, G. 2005, February. Agile development in computer science education: Practices and prognosis. ACM SIGCSE Bulletin, 37(1), 130–131.

Beznosov, K. and Kruchten, P. 2004, September. Towards agile security assurance. In *Proceedings of the 2004 Workshop on New Security Paradigms* (pp. 47–54). ACM.

BIGDSE. 2018. *Proceedings of the 2nd International Workshop on BIG Data Software Engineering*. [Online] Available at: https://sse.uni-due.de/bigdse16/

Blankenship, J., Bussa, M., Millett, S., Lewis, R. and Foggon, D. 2011. *Pro Agile. NET Development with Scrum. Berkeley: Apress*, p. 17.

Boar, B.H., 1984. Application prototyping: a requirements definition strategy for the 80s. John Wiley & Sons, Inc.

Boehm, B.W. 1991. Software risk management: Principles and practices. *IEEE Software*, 8(1), 32–41.

Boynton, P. and Greenhalgh, T. 2004. Selecting, designing, and developing your questionnaire. *British Medical Journal*, 328(7451), 1312–1315.

Brown, D. 2012. *Agile user experience design: A practitioner's guide to making it work*. Waltham: Newnes.

Canty, D. 2015. *Agile for project managers*. NW: Auerbach Publications.

Carter, N., Bryant-Lukosius, D., DiCenso, A., Blythe, J. and Neville, A. 2014. The use of triangulation in qualitative research. *Oncology Nursing Forum*, 41(5), 545-547.

Cheng, B., Zhao, S., Niu, M. and Chen, J. 2018, April. Agile IoT service communication and orchestration platform using event driven SOA paradigm. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 1–2). IEEE.

Chen, H. M., Kazman, R. and Haziyev, S. 2016. Agile big data analytics development: An architecture-centric approach. Proceedings of the *Annual Hawaii International Conference on System Sciences*. IEEE, 2016–March, pp. 5378–5387. doi: 10.1109/HICSS.2016.665.

Child, D. 1990. *The essentials of factor analysis* (2nd ed.). London: Cassel Educational Limited.

Chow, T. and Cao, D.B., 2008. A survey study of critical success factors in agile software projects. Journal of systems and software, 81(6), pp.961-971.

Chyung, S.Y., Barkin, J.R. and Shamsy, J.A., 2018. Evidence-based survey design: The use of negatively worded items in surveys. Performance Improvement, 57(3), pp.16-25.

Cochran, W.G. 1963. *Sampling techniques*. New York, NY: Wiley.

Cockburn, A. 2002. *Agile software development*. Reading, MA: Addison Wesley Longman, Inc.

Cohen, J. 1992. Quantitative methods in psychology: A power primer. *Psychological Bulletin*, 112, 1155–1159.

Cohen, L., Manion, L. and Morrison, K. (2007) Research methods in education. 6th edition. London: Routledge.

Cohn, M. 2005. *Agile estimating and planning*. NJ: Pearson Education.

Conboy, K., Coyle, S. and Wang, X. 2011. People over process: Key challenges in agile development. *IEEE Software*, 28(4), 48–57.

Conboy, K., Morgan, L., Power, K., Stol, K., Valerdi, R. and Fitzgerald, B. 2013. *Lean Enterprise Software and Systems*. Berlin, Heidelberg: Springer, p. 195.

Coram, M. and Bohner, S. 2005, April. The impact of agile methods on software project management. In *12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05)* (pp. 363–370). IEEE.

Corden, A. and Sainsbury, R. 2006. *Using verbatim quotations in reporting qualitative social research: Researchers' views*. York: University of York.

Cox, R., 2019. Surviving Software Dependencies. [Online] Available at: https://queue.acm.org/detail.cfm?id=3344149

Cronbach, L. and Shavelson, R. 2004. My current thoughts on coefficient alpha and successor procedures. *Educational and Psychological Measurement*, 64(3), 391–418.

Cruzes, D., Dybå, T., Runeson, P. and Höst, M. 2015. Case studies synthesis: A thematic, cross-case, and narrative synthesis worked example. *Empirical Software Engineering*, 20(6), 1634–1665.

Da Silva, T.S., Martin, A., Maurer, F. and Silveira, M. 2011, August. User-centered design and agile methods: A systematic review. In *2011 AGILE Conference* (pp. 77–86). IEEE.

Da Silva, T.S., Silveira, M.S., Maurer, F. and Hellmann, T. 2012. User experience design and agile development: From theory to practice. *Journal of Software Engineering and Applications*, 5(10), 743.

Dimmick, D., 2012. Fitting Big-Picture UX into Agile Development. [Online] Available at: https://www.smashingmagazine.com/2012/11/design-spikes-fit-big-picture-ux-agile-development

Douglass, B.P. 2016. *Agile systems engineering*. Waltham: Morgan Kaufmann.

Donovan Wells, J. (2000) "Extreme Programming: A gentle introduction", www.extremeprogramming.org.

Drury, M., Conboy, K. and Power, K. 2012. Obstacles to decision making in agile software development teams. *Journal of Systems and Software*, 85(6), 1239–1254.

Dudovskiy, J. 2018. *The ultimate guide to writing a dissertation in business studies: A step-by-step assistance.* Pittsburgh, PA: Jonathan.

El-Khawaga, G.H. 2013. Architecting in the context of agile software development: Fragility versus flexibility. *International Journal of Computer Science, Engineering and Applications* (IJCSEA), 3.

Fahmideh, M., Grundy, J., Ahmed, A., Shen, J., Yan, J., Mougouei, D., Wang, P., Ghose, A., Gunawardana, A., Aickelin, U. and Abedin, B., 2021. Software engineering for blockchain based software systems: Foundations, survey, and future directions. arXiv preprint arXiv:2105.01881.

Fowler, M. and Highsmith, J. 2001. The agile manifesto. *Software Development*, 9(8), 28–35.

Fuqua, A. 2016. *What's a spike, who should enter it, and how to word it?* [Online] Available at: https://www.leadingagile.com/2016/09/whats-a-spike-who-should-enter-it-how-to-word-it/

Garousi, V., Felderer, M. and Mäntylä, M.V., 2019. Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. Information and Software Technology, 106, pp.101-121.

Grady, N.W., Payne, J.A. and Parker, H. 2017, December. Agile big data analytics: Analytics Ops for data science. In *2017 IEEE International Conference on Big Data* (Big Data) (pp. 2331–2339). IEEE.

Hadar, I., Sherman, S., Hadar, E. and Harrison, J.J. 2013. Less is more: Architecture documentation for agile development. In *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering* (CHASE) (pp. 121–124). IEEE.

Hancock, D. and Algozzine, R. (2016) *Doing case study research: A practical guide for beginning researchers*. New York, NY: Teachers College Press.

Hellmann, T., Sharma, A., Ferreira, J. and Maurer, F. 2012. Agile testing: Past, present, and future. In *Agile Conference* (AGILE).

Hennink, M., Hutter, I. and Bailey, A. 2010. *Qualitative research methods*. London: SAGE Publication.

Highsmith, J.A. and Highsmith, J. 2002. *Agile software development ecosystems*. IN: Addison-Wesley Professional.

Hoda, R., Noble, J. and Marshall, S., 2012. Documentation strategies on agile software development projects. *International Journal of Agile and Extreme Software Development*, 1(1), pp.23-37.

Hoda, R., Noble, J. and Marshall, S. 2013. Self-organizing roles on agile software development teams. *IEEE Transactions on Software Engineering*, 39(3), 422–444.

Hossain, A., Kashem, M.A. and Sultana, S., 2013. Enhancing software quality using agile techniques. IOSR Journal of Computer Engineering, 10(2), pp.87-93.

Hughes, R., 2012. Agile data warehousing project management: business intelligence systems using Scrum. Newnes.

Hunt, J. 2018. *PMI-ACP Project Management Institute Agile Certified Practitioner Exam Study Guide*. Indianapolis, IN: John Wiley & Sons, Inc.

Huo, M., Verner, J., Zhu, L. and Babar, M.A. 2004, September. Software quality and agile methods. In *Proceedings of the 28th Annual International Computer Software and Applications Conference*, 2004 (COMPSAC 2004) (pp. 520–525). IEEE.

Isaacs, M. 2016. *What is agile testing? Process, strategy, test plan, life cycle example*. [Online] Available at: https://www.guru99.com/agile-testing-a-beginner-s-guide.html

Jalote, P., Palit, A., Kurien, P. and Peethamber, V.T. 2004. Timeboxing: A process model for iterative software development. *Journal of Systems and Software*, 70(1–2), 117–127.

Kalem, S., Donko, D. and Boskovic, D. 2013, May. Agile methods for cloud computing. In *2013 36th International Convention on Information and Communication Technology, Electronics and Microelectronics* (MIPRO) (pp. 1079–1083). IEEE.

Karlsson, J. 2020. *Agile product backlogs basics*. [Online] Available at: https://www.perforce.com/resources/hns/agile-product-backlog-basics

Kimmond, R.M., 1995. Survey into the acceptance of prototyping in software development. In Proceedings Sixth IEEE International Workshop on Rapid System Prototyping. Shortening the Path from Specification to Prototype (pp. 147-152). IEEE.

Kitchenham, B., Pickard, L. and Pfleeger, S. 1995. Case studies for method and tool evaluation. *IEEE Software*, 12(4), 52–62.

Kitchenham, B.A., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., El Emam, K. and Rosenberg, J., 2002. Preliminary guidelines for empirical research in software engineering. IEEE Transactions on software engineering, 28(8), pp.721-734.

Kitchenham, B., & Charters, S. (2007). Guidelines for performing Systematic Literature reviews in Software Engineering Version 2.3. Software Engineering Group, School of Computer Science and Mathematics, Keele University, 45(4ve), 1051. https://doi.org/10.1145/1134285.1134500

Knaster, R. and Leffingwell, D. 2018. *SAFe 4.5 distilled: Applying the scaled agile framework for Lean software and systems engineering*. NY: Addison-Wesley Professional.

Kohavi, R. and Longbotham, R., 2017. Online Controlled Experiments and A/B Testing. Encyclopedia of machine learning and data mining, 7(8), pp.922-929.

Lalsing, V., Kishnah, S. and Pudaruth, S. 2012. People factors in agile software development and project management. *International Journal of Software Engineering & Applications*, 3(1), 117–137.

Landsheer, J. and Boeije, H. 2008. In search of content validity: Facet analysis as a qualitative method to improve questionnaire design. *Quality & Quantity*, 44 (1), 59–69.

Layton, M. 2019. What is a spike and why does it confuse so many people?. [Online] Available at: https://platinumedge.com/ask-mr-agile-what-is-a-spike-and-why-does-it-confuse-so-many-people

Leffingwell, D., Martens, R. and Zamora, M. 2008. *Principles of agile architecture*. Leffingwell, LLC. & Rally Software Development Corp.

Leffingwell, D. 2009. The Big Picture of Enterprise Agility.[Online] Available at: https://scalingsoftwareagility.files.wordpress.com/2009/08/the-agile-enterprise-big-picture.pdf

Leffingwell, D. 2010. *Agile software requirements: Lean requirements practices for teams, programs, and the enterprise*. Boston: Addison-Wesley Professional.

Lenarduzzi, V., Lunesu, M.I., Marchesi, M. and Tonelli, R. 2018. Blockchain applications for agile methodologies. In *19th International Conference on Agile Processes in Software Engineering and Extreme Programming*.

Lethbridge, T., Sim, S. and Singer, J. 2005. Studying software engineers: Data collection techniques for software field studies. *Empirical Software Engineering*, 10(3), 311–341.

Lucassen, G., Dalpiaz, F., Werf, J.M.E.M.v.d., Brinkkemper, S. 2016. The use and effectiveness of user stories in practice. In Daneva, M. and Pastor, O. (Eds.), *Requirements* *engineering: Foundation for software quality*. REFSQ 2016. Lecture Notes in Computer Science (vol. 9619) (pp. 205–222). Cham: Springer.

Majid, M.A., Othman, M., Mohamad, S.F., Lim, S.A. and Yusof, A. 2017. Piloting for interviews in qualitative research: Operationalization and lessons learnt. *International Journal of Academic Research in Business and Social Sciences*, 7(4), 1073–1080.

Mantei, M.M. and Teorey, T.J. 1988. Cost/benefit analysis for incorporating human factors in the software lifecycle. *Communications of the ACM*, 31(4), 428–439.

Marchesi, M., Marchesi, L. and Tonelli, R., 2018, October. An agile software engineering method to design blockchain applications. In Proceedings of the 14th Central and Eastern European Software Engineering Conference Russia (pp. 1-8).

Matharu, G.S., Mishra, A., Singh, H. and Upadhyay, P. 2015. Empirical study of agile software development methodologies: A comparative analysis. *ACM SIGSOFT Software Engineering Notes*, 40(1), 1–6.

Maximini, D., Maximini, and Rauscher, 2018. *Scrum culture*. NY: Springer International Publishing AG.

Mayr, S., Erdfelder, E., Buchner, A. and Faul, F. 2007. A short tutorial of GPower. *Tutorials in Quantitative Methods for Psychology*, 3(2), 51–59.

Meline, T. 2006. Selecting studies for systematic review: Inclusion and exclusion criteria. *Contemporary Issues in Communication Science and Disorders*. ASHA, 33(21-27).

Mitchell, A. 2018. A review of mixed methods, pragmatism and abduction. *The Electronic Journal of Business Research Methods*, 16(3).

Moniruzzaman, A.B.M. and Hossain, D.S.A. 2013. *Comparative study on agile software development methodologies*. arXiv preprint arXiv:1307.3356.

Moran, A. 2014. Agile risk management. In *Agile risk management* (pp. 33–60). Cham: Springer.

Morris, D. 2018. *What is a Spike?* [Online] Available at: https://beardedeagle.com/what-is-a-spike/

Nemati, H. (Ed.) 2007. *Information security and ethics: Concepts, methodologies, tools, and applications*. NY: IGI Global.

Nicolette, D. 2014, Is a spike a type of story?. [Online] Available at: https://davenicolette.wordpress.com/2014/09/25/is-a-spike-a-type-of-story/

NHS. 2021. *Safe to fail experiments*. Online library of Quality, Service Improvement and Redesign tools. [Online] Available at: https://www.england.nhs.uk/wp-content/uploads/2021/12/qsir-safe-to-fail-experiments.pdf

Nørskov, S. and Rask, M. 2011. Observation of online communities: A discussion of online and offline observer roles in studying development, cooperation and coordination in an open-source software environment. *Forum: Qualitative Social Research*, 12(3).

Proestakis, A. 2018. *Get value from technical spikes*. [Online] Available at: https://www.ibm.com/garage/method/practices/discover/value-from-technical-spikes

Pomberger, G. and Weinreich, R., 1994. The Role of Prototyping in Software Development. In TOOLS (13) (p. 525).

Ogle, J. 2019. *thoughtbot: Design spiking*. [Online] Available at: https://thoughtbot.com/blog/design-spiking

Oliver, P. 2010. *The student's guide to research ethics*. Maidenhead, UK: Open University Press.

Rahman, N., Rutz, D. and Akhter, S. 2013. Agile development in data warehousing. In *Principles and applications of business intelligence research* (pp. 286–300). IGI Global.

Rindell, K., Hyrynsalmi, S. and Leppänen, V., 2017. Case study of agile security engineering: Building identity management for a government agency. *International Journal of Secure Software Engineering* (IJSSE), 8(1), 43–57.

Runeson, P. and Höst, M. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2), 131–164.

Runeson, P., Höst, M., Rainer, A. and Regnell, B. 2012. *Case study research in software engineering: Guidelines and examples*. Hoboken, NJ: Wiley.

Ryan, F., Coughlan, M. and Cronin, P. 2009. Interviewing in qualitative research: The one-to-one interview. *International Journal of Therapy and Rehabilitation*, 16(6), 309–314.

Salmons, J. 2015. *Qualitative online interviews*. Thousand Oaks, CA: SAGE Publications.

Saunders, M., Lewis, P. and Thornhill, A. 2016. *Research methods for business students* (7th ed.). Italy: Pearson Education.

Selic, B. 2009. Agile documentation, anyone? *IEEE Software*, 26(6), 11–12.

Singh, Y.K., 2006. Fundamental of research methodology and statistics. New Age International.

Shankarmani, R., Pawar, R., Mantha, S.S. and Babu, V. 2012. Agile methodology adoption: Benefits and constraints. *International Journal of Computer Applications*, 58(15).

Sharifloo, A.A., Saffarian, A.S. and Shams, F. 2008, March. Embedding architectural practices into extreme programming. In *19th Australian Conference on Software Engineering* (ASWEC 2008) (pp. 310-319). IEEE.

Schneider, J.-G. and Vasa, R. 2006. Agile practices in software development – Experiences from student projects. In *2006 Australian Software Engineering Conference* (ASWEC 2006), 18–21 April 2006. doi: 10.1109/ASWEC.2006.9.

Scholz, R. and Tietje, O. 2002. *Embedded case study methods: Integrating quantitative and qualitative knowledge*. Thousand Oaks, CA: Sage Publications.

Schwaber, K. 2004. Agile project management with Scrum. Microsoft Press.

Schwaber, K. and Sutherland, J. 2011. The scrum guide. Scrum Alliance 21, no. 19.

Shaffer, J. 1995. Multiple hypothesis testing. *Annual Review of Psychology*, 46(1), 561–584.

Shore, J. and Warden, S. 2008. *The Art of AgileSM*. Available at: https://www.jamesshore.com/v2/books/aoad1/spike_solutions

Siponen, M., Baskerville, R. and Kuivalainen, T. 2005, January. Integrating security into agile development methods. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences* (pp. 185a-185a). IEEE.

Snyder, H., 2019. Literature review as a research methodology: An overview and guidelines. Journal of business research, 104, pp.333-339.

Stol, K.J. and Fitzgerald, B. 2015, May. A holistic overview of software engineering research strategies. In *Proceedings of the Third International Workshop on Conducting Empirical Studies in Industry* (pp. 47–54). IEEE Press.

Thompson, B. 2004. *Exploratory and confirmatory factor analysis: Understanding concepts and applications*. Washington, DC, 10694.

van Drongelen, M., Dennis, A., Garabedian, R., Gonzalez, A. and Krishnaswamy, A. 2017. *Lean mobile app development: Apply Lean startup methodologies to develop successful iOS and Android apps*. Birmingham: Packt Publishing Ltd.

Vinodh, S., Devadasan, S.R., Maheshkumar, S., Aravindakshan, M., Arumugam, M. and Balakrishnan, K. 2010. Agile product development through CAD and rapid prototyping technologies: An examination in a traditional pump-manufacturing company. *The International Journal of Advanced Manufacturing Technology*, 46(5–8), 663–679.

Warfa, A. (2016). Mixed Methods Research in Biology Education: Approach and Uses. CBE Life Science Educ., Dec 1, volume 15 (4).

Wang, A., Sørensen, C.F. and Conradi, R., 2004. A Three Level Framework for Process Support: The MOWAHS Approach. In Software Engineering Research and Practice (pp. 10-16).

Wirfs-Brock, R.Y.J.a.G.E. 2015. Patterns to develop and evolve architecture during an agile software project. *Proceedings of the 22nd Conference on Pattern Languages of Programs* (pp. 1–18). ACM.

Wohlin, C., 2014. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In Proceedings of the 18th international conference on evaluation and assessment in software engineering (pp. 1-10).

Woodward, C.J., Montgomery, J., Vasa, R. and Cain, A. 2013, August. Agile development spikes applied to computer science education. In *Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering* (TALE) (pp. 699–704).

Yadav, A., Lundeberg, M., DeSchryver, M., Dirkin, K., Schiller, N., Maier, K. and Herreid, C. 2007. Teaching science with case studies: A national survey of faculty perceptions of the benefits and challenges of using cases. *The Journal of College Science Teaching*, 34–38.

Yin, R. 2014. *Case study research design and methods* (5th ed.). Thousand Oaks, CA: Sage Publications.

Yli-Huumo, J., Maglyas, A. and Smolander, K. 2016. How do software development teams manage technical debt? – An empirical study. *Journal of Systems and Software*, 120, 195–218.

Younas, M., Jawawi, D.N., Ghani, I., Fries, T. and Kazmi, R. 2018. Agile development in the cloud computing environment: A systematic review. *Information and Software Technology*.

Appendix A: Research Ethics Forms

Participant Information Sheet

Study Title: Usage and perceptions of spikes in agile software development: An exploratory study in an industrial context

Researcher: Hussein Al Hashimi **ERGO number:** 62395

Please read the information below carefully and ask questions if anything is not clear or you would like more information before you decide to take part in this research.

What is the research about?
The primary goal of this research is to investigate the various aspects of spikes in agile, including how they are used and how their behaviour is different from traditional agile user stories. This research aims to investigate the role and use of spikes in agile and to highlight this role to measure the effectiveness of agile spikes in reducing risk in project activities. In addition, this study investigates how the use of agile spikes can reduce technical and functional risks. This study will help to understand the relationship between agile spikes and software projects, while answering research questions and results based on the collected data.

In this context, the following areas will be explored by this study:

- Effective use of spikes in various domains
- Role of spikes in managing risk
- Role of spikes in Agile Software Development.
- Causes of uncertainty in agile software projects
- Role of spikes in estimating user stories in agile development
- Success factors of spikes that enable user stories to be completed correctly.

Why have I been asked to participate?
The ideal participant for this study will be persons with a broad knowledge of software development using different agile methodologies as well as spikes technique. In this way, the collective responses to the focus group interviews and survey questions will represent a personal opinion on each question based on the professional knowledge they possess. Your participation in this study is very important to clear the misunderstanding associated with the use of spikes in agile software development. Therefore, you are being invited because you passed the above description.

Why have I been asked to participate?
The ideal participant for this study will be persons with a broad knowledge of software development using different agile methodologies as well as spikes technique. In this way, the collective responses to the focus group interviews and survey questions will represent a personal opinion on each question based on the professional knowledge they possess. Your participation in this study is very important to clear the misunderstanding associated with the use of spikes in agile software development. Therefore, you are being invited because you passed the above description.

What data will be collected?
The study seeks to collect specific data that answers the research questions only. No personal data will be collected that can identify the participants. Specifically, on the demographics data, only years of experience in agile developments and agile role will be collected. The rest will be your personal opinions based on the questions seeking to uncover how efficient agile spikes are in risk management during software development. Although no personal details are collected, all the information collected will be kept confidential and compliant with Data Protection Laws.

Will my participation be confidential?
Your participation and the information we collect about you during the course of the research will be kept strictly confidential.

Only members of the research team and responsible members of the University of Southampton may be given access to data about you for monitoring purposes and/or to carry out an audit of the study to ensure that the research is complying with applicable regulations. Individuals from regulatory authorities (people who check that we are carrying out the study correctly) may require access to your data. All of these people have a duty to keep your information, as a research participant, strictly confidential.

The data collected will be held electronically under a password-secured and encrypted database. For any audio records during interviews, the information will be transcribed, and the audio destroyed thereafter. The destruction is intended to prevent unauthorised use of the audio files by anyone. Furthermore, the transcribed information will be stored electronically in the same database as the recorded data from the focus group discussion. The only persons who will have access to the data are the researcher, the researcher supervisor, and authorised faculty staff for the assessment of quality.

Do I have to take part?
No, it is entirely up to you to decide whether or not to take part. If you decide you want to take part, you will need to sign a consent form and other required forms to show you have agreed to take part. All the required documents will be sent to your email, or they will be handed to you.

What happens if I change my mind?
You have the right to change your mind and withdraw at any time without giving a reason and without your participant rights being affected. Also, you can withdraw later within one month from the group discussion date by contacting me on my email: hah2n17@soton.ac.uk. Your data will be deleted directly if you decide to withdraw. If you withdraw from the study, we will keep the information about you that we have already obtained for the purposes of achieving the objectives of the study only.

What will happen to the results of the research?
Your personal details will remain strictly confidential. Research findings made available in any reports or publications will not include information that can directly identify you or your Organisation without your specific consent. The analysis of data will be published in conferences or journals.

Where can I get more information?
For further details, please contact me or my supervisor, Dr.Andy Gravell.
Investigator: Hussein Al Hashimi, hah2n17@soton.ac.uk
Dr.Andy Gravell, amg@ecs.soton.ac.uk

If you have any ethical concerns or practical issues with the survey, please contact the University's Research Ethics Committee at **(**risethic@soton.ac.uk**).** Every concern or question will be treated with the utmost urgency, confidentiality, and seriousness it deserves. You will be informed of the outcome or response as soon as it is addressed.

What happens if there is a problem?
If you have a concern about any aspect of this study, you should speak to the researchers who will do their best to answer your questions.

If you remain unhappy or have a complaint about any aspect of this study, please contact the University of Southampton Research Integrity and Governance Manager (023 8059 5058, rgoinfo@soton.ac.uk).

Appendix B: Interview Consent Form

Study title: Usage and perceptions of spikes in agile software development: An exploratory study in an industrial context

Researcher name: Hussein Al Hashimi
ERGO number: 62395

Please initial the box(es) if you agree with the statement(s):

I have read and understood the information sheet (02/12/2020 _ version no. 1 of participant information sheet) and have had the opportunity to ask questions about the study.	
I agree to take part in this research project and agree for my data to be used for the purpose of this study.	
I understand my participation is voluntary and I may withdraw at any time for any reason without my participation rights being affected.	
I understand that taking part in the study involves audio/video recording which will be used for data analysis purpose and then will be destroyed for the purposes set out in the participation information sheet.	

Name of participant (print name) ....

Signature of participant...

Date..…

Name of researcher (print name) Hussein Al Hashimi

Signature of researcher ....

Date..….

☐ Please tick (check) this box to indicate that you consent to taking part in this survey.

Appendix C: Study Details

What are the aims and objectives of this study?
The primary goal of this research is to investigate the various aspects of spikes in agile, including how they are used and how their behaviour is different from traditional agile user stories. This research aims to investigate the role and use of spikes in agile and to highlight this role to measure the effectiveness of agile spikes in reducing risk in project activities. In addition, this study investigates how the use of agile spikes can reduce technical and functional risks. This study will help to understand the relationship between agile spikes and software projects while answering research questions and results based on the collected data. In this context, the following areas will be explored by this study: • Effective use of spikes in various domains • Role of spikes in managing risk • Role of spikes in ASD • Causes of uncertainty in agile software projects • Role of spikes in estimating user stories in agile development • Success factors of spikes that enable user stories to be completed correctly.

Background of the study (*a brief rationale for conducting the study)*
In any software development project, risk is a factor that defines success to a significant extent. Through agile spikes technique, the risks can be managed effectively through rapid prototyping, investigation, and research. With the efficiency of spikes in agile software development having limited research, empirical evidence is therefore required to ascertain the role spikes play during the development process. Subsequently, the efficiency of the technique in risk management is determined. Although there is existing research on agile spikes, the question of how they can be used to improve the quality of software products has not been covered in detail. This gap forms the basis for this study, which seeks to uncover these details by collecting primary data from agile software development professionals.

Key research question *(Specify hypothesis if applicable*)
The research study has four key questions, with two subdivided into one and two for purposes of achieving the research objectives. They included: • RQ1. What roles do spikes play in different agile methods? • RQ2. How can agile spikes be used efficiently? How do spikes estimate user stories, effort and delivery time in agile software development? What is the appropriate time for spikes to be applied? • RQ3. How can agile spikes be used to manage risk effectively? How is project uncertainty defined using spikes in agile software development? • RQ4. What are the factors that help to apply spikes successfully?

Study design *(Give a **brief** outline of the study design and why it is being used)*
This study will utilise an exploratory and empirical research design. Both qualitative and quantitative data will be used in the study. Data will be collected through direct and indirect methods. The direct method involves contact with practitioners in the case study, such as interviews and focus groups. The indirect method obtains raw data without contact with practitioners, such as using tools to mentor a practitioner's performance and productivity. The direct method of data collection through a focus group will be used in the case study, allowing the researcher to maintain full control over the collected data and obtain them in the required context and form. A focus group is an interview with a group of participants, asking questions to stimulate discussion and obtain responses. In addition, a discussion and interview will be conducted with the product owner on the behaviour and performance of the development team, which represent indirect methods. Moreover, there will be observations of the development team's behaviour and approach to managing risk in the sprint and sprint planning event to understand how they can deal with risk and uncertainty. The study of spikes in agile software development will also adopt an empirical research design to enable better access to data and information, both directly and indirectly. For access to data and information, direct observation will be employed in one-to-one interviews, focus group interviews, and surveys. Indirectly, data and information will be accessed easily through the experience of collecting the data. Empirical research designs accommodate both quantitative and qualitative methods of design while collecting data directly, especially through the observation of skills. Usage of empirical research design and techniques has increased in various software engineering research studies, due to their efficacy

PRE-STUDY

Characterise the proposed participants
It will be 3 focus group discussions in the case study, and in each discussion meeting, there will be around 2-3 participants who achieve the sampling criteria. The participant could be a project manager, software programmer, analyst, designer or tester working on the project. There are some criteria, stated below, that should exist in the participant.

Describe how participants will be approached
In the group discussion, participants will be approached with an email containing an introduction about the researcher, the research problem, and an overview of the required time and information from them. Microsoft Teams call is another option to describe the whole information to the participants in case the participant prefers this way. The expert's contact information will be obtained from their Organisations' websites. Some experts are known to the author from his social network and relation with others.

Describe how inclusion/exclusion criteria will be applied (if any)
There are certain criteria that should be fulfilled in all participants including: • They are all the legal age i.e., above 18 years • They are experts in software development • Holds at least one agile role (agile team or expert) • Have a good background in agile methodologies

- They have experience in agile software development and have used the spikes technique before
- They are still actively involved or were actively involved in agile software development

Describe how participants will decide whether or not to take part
According to the participants' reply to the invitation email or Microsoft Teams call and based on their confirmation to agree to participate in the group discussion. They should mention in that email the details (time, date, and location) of the meeting. Usually, the meeting would be held in their offices, or any preferable place offered by the participants, or even virtually due to COVID-19. At the group discussion meeting, the participant should sign the participant information form and consent form.

DURING THE STUDY

Describe the study procedures as they will be experienced by the participants
After the participants signed the required documents, it will be explained to them a brief about the case study. Since the method of the case study is a combination of the focus group method and observation method, the participants will be asked some background questions, and then move forward to the case study questions regards the spikes in agile software development. In the focus group discussion, there are some structured and semi-structured questions that will be asked to participants regards the spikes success factors and some risk management techniques and processes that have been used such as POC and MVP.

Identify how, when, where, and what kind of data will be recorded (not just the formal research data, but including all other study data such as e-mail addresses and signed consent forms)
How: The focus group discussion will be conducted by meeting the participants face to face or virtually. This kind of discussion can take an advantage of the social cues such as voice intention and body language, which give the discussion chance to gain more information that can be added to the verbal answers from the participants. **When:** the meeting will be held after we agree on the specific time. **Where:** at the participants' company's office, any preferable place they recommend, or can conduct a virtual meeting. **What kind of data:** answer the group discussion questions that are related to the spikes in agile software development.

Appendix D: Practitioner Interviews for RQ1 to RQ3

Interview Questions (Semi-Structured)

#Participant No. Choose an item. **Date:**

Section 1: Demographic Details

i. Experience in agile Choose an item.

ii. Experience in spikes Choose an item.

iii. Agile role Choose an item.

iv. Your country Click here to enter text.

Section 2: Roles of spikes in Agile Software Development

1. What agile methodologies is your team or company using in software development?
2. Based on your experience, what can you say about using spikes in agile software development?
3. How can spike help you in agile software development?

Section 3: Efficiency of Spikes in ASD

1. What roles would you say spikes have in estimating user stories in agile development? How do spikes estimate user stories in agile?
2. What is the appropriate time to use spikes in agile software development?
3. According to your understanding, do you believe agile spikes are influencing project activities? If yes, how?
4. On a scale of 1-5 where 1 being the lowest, how would you rate the efficiency of spikes in agile software development? Why this score?

Section 4: Effectiveness of Spikes in Risk Management and Software Domains

1. Spikes are known to be used in different domains during agile software development. What are some of these domains that you have utilised or have witnessed spikes being used?
2. What do you believe in the influence of agile spikes on the quality of software products?
3. In your opinion, what are the most appropriate agile methods that can use spikes more effectively?
4. How is project uncertainty defined in agile software development? can you list some causes of uncertainty in agile software projects?
5. How useful do you find spikes to be in reducing technical and functional risks?

Figure AP. D1: Interview questions for RQ1 to RQ3

Interview Questions (cont. 2)

Section 4: Effectiveness of Spikes in Risk Management and Software Domains (cont. 2)

6. How did you find the use of spikes in risk management?
7. On a scale of 1-5 where 1 being the lowest, how will you rate the effectiveness of spikes in reducing risk? Why this score?

Figure AP. D2: Interview questions for RQ1 to RQ3

Appendix E: Practitioner Interviews for RQ4

Interview Questions for RQ4 (Semi-Structured)

#Participant No. Choose an item. **Date:**

Section 1: Demographic Details

i.	Experience in agile	Choose an item.
ii.	Experience in spikes	Choose an item.
iii.	Agile role	Choose an item.
iv.	Your agile method	Choose an item.
v.	Organization sector	Choose an item.
vi.	Organization size	Choose an item.
vii.	Your country	Click here to enter text.

Success Factors

1. From your experience in using spikes, have you considered some factors that help the successful application of spikes?
   If yes, can you provide some of them?

2. In your opinion, what should agile teams particularly project/product manager need to realize to enhance the spikes' outcomes?

3. The success factors for correct application of spikes may take different dimensions. Based on your experience and your opinion, can you provide some of the factors based on the following categories?
   a) Technical factors
   b) Organizational factors
   c) Process/procedural factors
   d) People factors
   e) Project-related factors

4. Do you think the categorization of the success factors above fits the actual factors that should be taken into account?

   Yes ☐
   No ☐

Figure AP. E1: Interview questions for RQ4

Appendix F: Codes and Themes

Nodes

Name	Files	References	Created On
Others	0	0	29/04/2020 22:21
Effect of spikes on project activities	0	0	28/04/2020 16:33
Effect on business logic	0	0	28/04/2020 16:21
Scrum's Product Backlog Refinement vs Spike	0	0	28/04/2020 16:11
Theme 1_Roles of spikes in agile softwae development	0	0	28/04/2020 16:01
Provides better understanding of user stories	6	6	28/04/2020 16:10
Uncertainty reduection	5	5	28/04/2020 22:15
New technology exploration	3	3	28/04/2020 22:15
Understanding business necessity	1	1	28/04/2020 22:15
Clarity on the needs of customers	2	2	29/04/2020 07:15
Risk reduction	3	3	29/04/2020 08:43
For decision making	1	1	29/04/2020 10:47
Gives detailed product information	2	2	29/04/2020 11:46
Timebox research or prototype	1	1	29/04/2020 12:04
Used when information on a project is unkonown	2	2	29/04/2020 14:08
Allows for a the provison of a better estimation	3	3	29/04/2020 15:00
Saves time	1	1	29/04/2020 15:01
Provides time for research	1	1	29/04/2020 18:19
Provides information that are not normally available	1	1	29/04/2020 18:52
Used for researching better ways to accomplish a task	1	1	29/04/2020 21:13
Improves story clarity	1	1	29/04/2020 21:38
Theme 2_Efficiency of Agile Spikes	0	0	29/04/2020 22:04
Theme 3_Impact of Spike in quality software products	0	0	28/04/2020 16:18
Theme 4_ Spikes applications in different domains	0	0	28/04/2020 21:47
Theme 6_Managing Risk with Spikes	0	0	29/04/2020 22:12

Figure AP. F1: NVivo node classification for RQ1 to RQ3

Nodes

Name	Files	References	Created On	Created By	Modified On
New Node	0	0	27/06/2021 06:09	HH	27/06/2021 06:09
Organisational Factor	26	45	18/06/2021 14:27	HH	18/06/2021 15:08
Participants'opinion on listed Factors	25	32	18/06/2021 14:38	HH	18/06/2021 15:08
People's Factor	24	52	18/06/2021 14:24	HH	18/06/2021 15:08
Process or Procedural Factors	23	36	18/06/2021 14:31	HH	18/06/2021 15:08
Project Related Factors	23	46	18/06/2021 14:34	HH	18/06/2021 15:08
Realizations that enhance spike outcome	25	47	18/06/2021 14:10	HH	18/06/2021 15:08
Success Factors based on experience	25	64	18/06/2021 14:03	HH	18/06/2021 15:09
Technical Factor	26	47	18/06/2021 14:23	HH	18/06/2021 15:08

Figure AP. F2: NVivo node classification for RQ4

Nodes

Name	Files	References	Created On	Created By
Are the spikes being used to carry out roles specified in project planning	0	0	06/09/2021 21:08	HH
Avoiding Challenges	0	0	06/09/2021 21:08	HH
Can ADSL Spikes resolve Technical and Functional issues	0	0	06/09/2021 21:08	HH
Challenges of Correct Application of Spike	0	0	06/09/2021 21:08	HH
Did Spike achieve its objective	0	0	06/09/2021 21:08	HH
Does your organization need to improve the existing risk management technique or replace it	1	1	06/09/2021 21:08	HH
Does your organization used other risk management techniques	0	0	06/09/2021 21:08	HH
Factors for Successful Spike	0	0	06/09/2021 21:08	HH
How to apply spike correctly	0	0	06/09/2021 21:08	HH
Number of Spike for whole project	0	0	06/09/2021 21:08	HH
Number of Spike used for each iteration	0	0	06/09/2021 21:08	HH
Objective of Spike in your recent software project	0	0	06/09/2021 21:08	HH
Recent Software Project	0	0	06/09/2021 21:08	HH
Roles Spike has played in previous project	0	0	06/09/2021 21:08	HH
Skills needed by development team	0	0	06/09/2021 21:08	HH
Spike Efficiency	1	1	06/09/2021 21:08	HH
Spike efficiency in risk management and estimation	0	0	06/09/2021 21:08	HH
Sprint Retrospective	0	0	06/09/2021 21:08	HH
Types of Spike	0	0	06/09/2021 21:08	HH
What Agile Teams can do when Spike didn't lead to the required result	0	0	06/09/2021 21:08	HH

Figure AP. F3: NVivo nodes classification for case studies

Item	Participants	Participant ID
Theme 1: Efficiency, experience, and appropriateness of agile spikes		
Methods in use		
LESS	1	P4
SAFe	1	P4
TDD	1	P6
ATDD	1	P6
RAD	1	P7
Lean	1	P7
Scrumban	1	P12
XP	2	P5, P7
Kanban	7	P10, P12, P16, P15, P18, P5, P9
Scrum	22	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P19, P20, P21, P22
Experience on spike usage		
Provide more understanding of the project scope	5	P4, P13, P14, P15, P16
Background task in Scrum is called product backlog refinement	1	P1
Measurable work in XP and SAFe is called spike	1	P1
Clarify and reduce uncertainty and ambiguity in new Technology	7	P4, P5, P8, P14, P20, P21, P22
Investigation and Better estimate delivery	9	P3, P4, P5, P9, P16, P18, P20, P21, P22
Provides prior knowledge on the event of a project	9	P1, P3, P9, P12, P13, P15, P20, P21, P22
Provide more insight through exploratory work	7	P1, P2, P9, P12, P17, P18, P20
Used for learning	3	P1, P2, P5, P6, P16, P17
Using in UI to develop a first POC	2	P2, P12
Spike usage time appropriateness		
Before sprint planning	12	P2, P5, P6, P7, P8, P10, P13, P14, P15, P16, P18, P20
Between backlog grooming and sprint planning	2	P13, P19
During Sprint planning	7	P4, P6, P7, P8, P9, P10, P16, P22
During sprint	18	P1, P2, P3, P4, P5, P6, P7, P8, P9, P11, P12, P14, P15, P17, P19, P20, P21, P22
When faced with uncertainty	9	P11, P12, P15, P17, P18, P20, P22, P6, P8
Rating of Effectiveness of spike in agile software development		
3	2	P10, P16
4	11	P1, P2, P3, P4, P5, P13, P14, P17, P19, P21, P22
5	9	P6, P8, P9, P11, P12, P15, P17, P18, P20

Table AP. F1: Study themes for RQ1 to RQ3

Item	Participants	Participant ID
Domains where spike is applicable		
New visual Interface and Business analysis	1	P2
Cloud applications	1	P3
UX design	1	P4
Insurance	1	P5
Air Traffic Control	1	P5
Antivirus Software	1	P5
Online marketing	1	P7
Supply chain software	1	P7
Finance software	1	P9
Retail software	1	P9
Automotive	1	P10
Medical and health care	1	P1, P10
Server or database migrations, development and warehousing	1	P17, P3, P7
Communications	3	P5, P6, P10
HR software	2	P5, P7
SaaS	3	P1, P13, P21
Data analysis and integration	4	P2, P22, P3, P7
API development	6	P12, P14, P16, P17, P18, P19
Web development	7	P12, P15, P18, P20, P19, P21, P22
Mobile Apps	6	P12, P15, P17, P18, P21, P22
Embedded Software	1	P12
IT and software development	8	P11, P8
Most Appropriate Agile methods for spike usage		
XP	2	P17, P8
Kanban	3	P19, P2, P8
Scrum	10	P11, P14, P15, P16, P19, P2, P20, P21, P3, P8
All agile methods	12	P1, P10, P12, P13, P16, P18, P21, P22, P4, P6, P7, P9
Theme 2: Role of spikes in software development		
Understanding business necessity	14	P1, P2, P3, P4, P9, P10, P11, P12, P13, P16, P17, P19, P20, P21, P22
Help in decision making	6	P1, P5, P7, P12, P13, P15
Provides information that is not normally available	9	P3, P5, P7, P11, P12 P13, P19, P20, P21
Provides a better understanding of user stories	16	P1, P2, P4, P5, P6, P7, P8, P9 P10, P14, P15, P17, P19 P20, P21, P22
Researching better ways to accomplish a task	6	P9, P11, P12, P13, P16, P17
Improves story clarity	9	P1, P3, P9, P12, P13, P15, P20, P21, P22
Explore new/unfamiliar technologies and tasks	13	P1, P2, P4, P5, P6, P9, P10, P14, P17, P19 P20 P21 P22
Allows for better estimation	17	P1, P2, P3, P4, P8, P9, P10 P11, P13, P14 P15, P16, P17, P19, P20, P21, P22
Estimating user stories	10	P4, P8, P10, P13, P14, P15, P17, P19, P20, P22
Estimate the work in the Product Backlog	5	P1, P3, P11, P16, P21

Table AP. F2: Study themes for RQ1 to RQ3

Item	Participants	Participant ID
Estimate the time required	4	P2, P9, P10, P15
Estimate the effort required to complete the story	1	P15
Sizing the user story more precisely	3	P6, P7, P12
Uncertainty reduction	16	P1, P2, P4, P5, P6, P7, P8, P9 P10, P14, P15, P17, P19 P20, P21, P22
Uncertainty can be minimised by short iterations	3	P11, P12, P18
Theme 3: Impact of Spike in quality software products		
Improve software quality	19	P1, P2, P3, P4, P5 P6, P7, P8, P9, P11, P12, P13, P14, P16, P18, P19, P20, P21, P22
No improvement in quality	3	P10, P15, P17
Developing and experimenting would be a better approach.	1	P10
The quality of the software will depend much more on the talent of the product team, developers, and QA specialists	1	P15
I believe the agile process is great to improve the quality of software however it sometimes favours speed of feature rollout over quality of software produced.	1	P17
Theme 4: Spike role in estimating user stories in agile development		
Not effective on velocity estimation	1	P8
Allows for better estimation	17	P1, P2, P3, P4, P8, P9, P10 P11, P13, P14 P15, P16, P17, P19, P20, P21, P22
Estimating user stories	10	P4, P8, P10, P13, P14, P15, P17, P19, P20, P22
Estimate the work in the Product Backlog	5	P1, P3, P11, P16, P21
Estimate the time required	4	P2, P9, P10, P15
Estimate the effort required to complete the story	1	P15
Sizing the user story more precisely	3	P6, P7, P12
Theme 5: Usefulness of spike in reducing technological and functional issues		
Spike offers quick learning	1	P6
Investigate new complex functionality	7	P1, P8, P11, P12 P13, P14, P18
Decrease unknowns	10	P1, P2, P4, P5, P11, P12, P13, P14, P18, P20
Not useful	2	P10, P15
Useful in reducing functional issues	14	P1, P2, P3, P4, P5, P6, P8, P13, P14, P16, P17, P18, P20, P22
Useful in reducing technical issues	20	P1, P2, P3, P4, P5, P6, P7, P8, P9, P11, P12, P13, P14, P16, P17, P18, P19, P20, P21, P22
The technical issues that are related to missing knowledge can be reduced through spikes.	1	P2
The technical debt could be researched during Spike, and the outcome of the spike could be the form of the best way of approaching or moving ahead.	1	P9
Functional uncertainty is addressed through the PO and refinement	1	P11
Technical spikes are very useful for teams to investigate technical issues before they estimate work	1	P11
Most teams don't have the time due to product teams pushing requirement fast and always trying to get features out. Using Spikes properly, will help the team to make technical decisions on the product.	1	P12
Spikes are most useful in scrum where work is broken down into sprint units. They are not as useful in Kanban but still somewhat useful especially for estimating the overall time needed to complete a project.	1	P15
Technical and functional issues will be resolved as long as the development team is competent regardless of whether a spike was used to investigate the story.	1	P15

Table AP. F3: Study themes for RQ1 to RQ3

Item	Participants	Participant ID
Theme 6: Managing Risk with Spikes		
Causes of uncertainty		
New/unfamiliar technologies or tasks	7	P1, P3, P8, P11, P13, P20, P21
Lack of clear stakeholder requirements	5	P1, P8, P13, P17, P20
Lack of experience for the agile team	7	P1, P2, P6, P8, P11, P20, P22
Compliance	1	P3
Project complexity	1	P3
Changing of development team members	1	P15
Poorly written stories	6	P4, P8, P19, P20, P21, P22
Misuse of agile processes	2	P17, P21
Inability to forecast unknown work	3	P4, P6, P9
Undefined project scope	6	P1, P4, P8, P15, P16 P19
Cross functions among team members	1	P21
Dependencies	4	P9, P10, P21, P22
Using incorrect language in development	1	P10
Project uncertainty, in my opinion, is why we do agile software development. I define it as three areas: The knowns, the known unknowns, and the unknown unknowns.	1	P12
The knowns are items (client/customer/stakeholders) know they want in the software and what is initially planned.	1	P12
The known unknowns are items they do not usually think of that the team will have to conduct spikes or document.	1	P12
Lastly, the unknown unknowns are what come up throughout development that needs attention from the team and will utilise spikes to make clear.	1	P12
Definition of uncertainty		
Defined requirements	1	P4
New features	1	P4
Any story with inadequate information	3	P17, P18, P21
Lack of certainty	4	P10, P14, P22, P7
Spike and Risk Management		
Spikes are Multiple forms of risk management	1	P2
Technical Risk	1	P1
Business risk (Schedule risk and budget risks)	1	P1
Reduces risk of maldevelopment	1	P10
Reduces the risk of technological debt	2	P10, P22
Estimating the risk associated with a new feature	4	P2, P7, P6, P9
Investigation of third-party integration	1	P15
Spike for visibility tests	2	P1, P8
Ability to forecast or plan work early	3	P1, P7, P21
Rating of Effectiveness of spike in risk reduction		
3	5	P10, P13, P16, P17, P3
4	9	P1, P2, P4, P19, P12, P22, P5, P8, P14
5	8	P11, P18, P15, P20, P21, P6, P7, P9

Table AP. F4: Study themes for RQ1 to RQ3

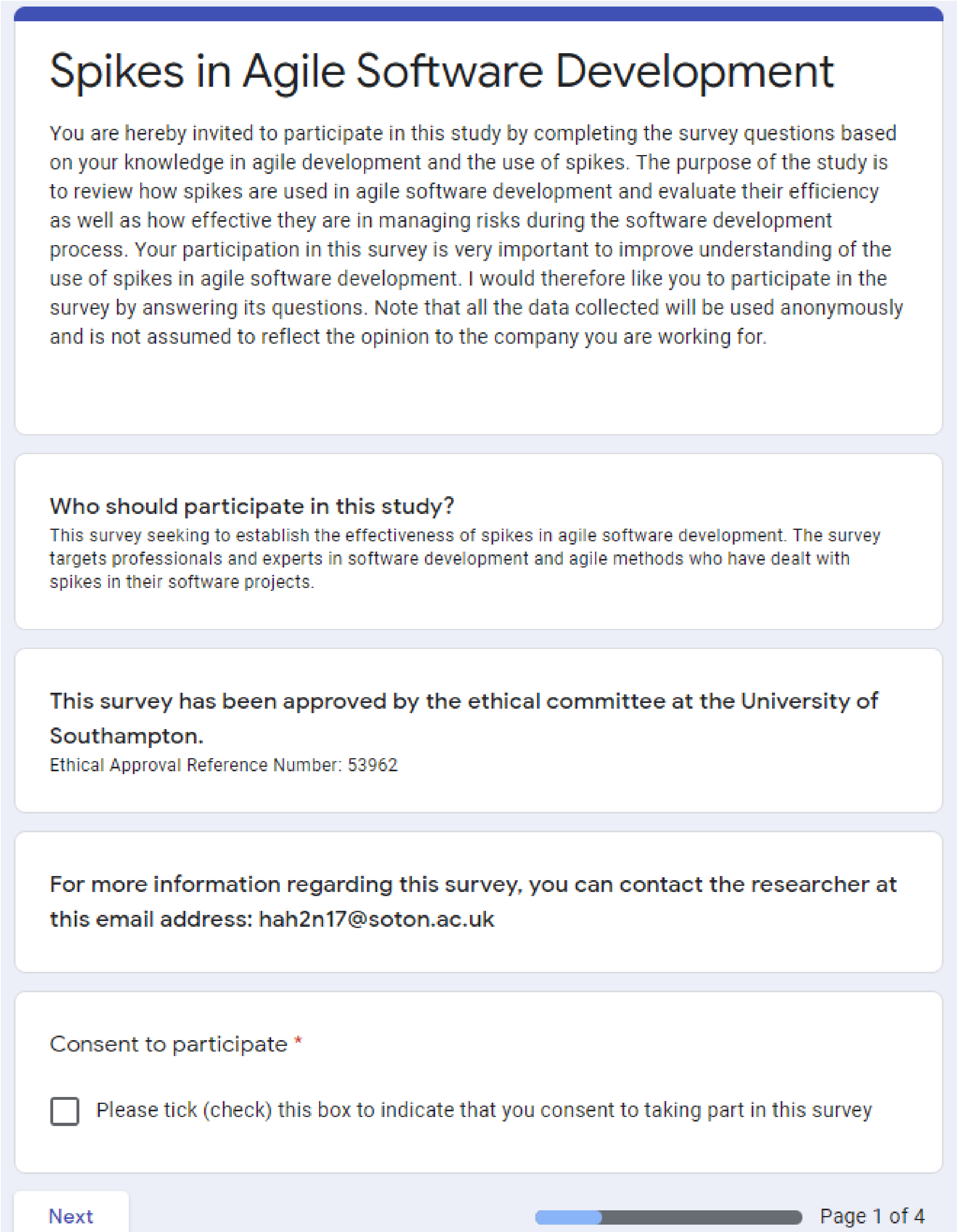
Name	Participants	Participant ID
Theme 1: Experience in Agile Spike, Agile role and Participants' country		
Experience in Agile		
6-10 yrs	12	P1,P4,P5,P8,P10,P12,P13,P18,P19,FG1-2,FG2-1,FG2-3
1-5 yrs	7	P3, P7,P11, P16, FG3-1, FG1-1,FG1-4
16 – 20 yrs	5	P2, P9, P14, P15, FG2-2
11 – 15 yrs	2	P6, FG1-3
Experience in Spike		
1-5yrs	13	P3,P5, P7, P8, P10, P11,P13, P16, FG1-1, FG1-4, FG1-2, FG3-1, FG3-3
6-10yrs	8	P1,P4, P6, P12, FG1-3, FG2-1, FG2-3, FG3-2,
16-20yrs	5	P2, P9, P14, P15,FG2-2
Agile role		
Developer	10	P2, P3, P6,P7, P15, P16, FG3-1, FG1-1,FG1-2, FG2-1
Agile Coach	9	P1, P2, P4, P6, P9,P10, P12, P15, FG3-3
Scrum Master	5	P5, P6, P7,P11,P14,FG3-2, FG2-1
Product Owner	4	P13, P14, FG3-3, FG2-3
Engineering Manager	2	P8, FG1-3
Tester	2	FG2-2, FG1-4
Product manager	1	FG1-3
Agile method		
Scrum	23	P1,P2,P3,P4,P5,P6,P7,P9,P10,P11,P13,P14,P15,P16, FG1-1, FG1-2,FG1-3, FG1-4, FG2-2, FG2-3, FG3-1, FG3-2, FG3-3
Kanban	1	P8
Organisation Sector		
IT & Software Development	21	P1,P2,P3,P4,P5,P6,P7, P10,P11,P13,P16, FG1-1, FG1-2,FG1-3, FG1-4, FG2-1, FG2-2, FG2-3, FG3-1, FG3-2, FG3-3
Consultancy	3	P9, P14, P15
Government	2	P6, P11
Energy	1	P8,
Finance	1	P6
Organisation Size		
Large	17	P1,P3,P4,P7, P8, P10,P11,P12, P13,P16, FG1-1, FG1-2 ,FG1-3, FG1-4, FG3-1, FG3-2, FG3-3
Medium	4	P5, FG2-1, FG2-2, FG2-3
Micro	3	P2, P15
Small	2	P6,P9
Country		
UK	6	P2,P8, FG2-1, FG2-2, FG2-3
Portugal	5	P16, FG1-1, FG1-2,FG1-3, FG1-4
US	4	P1, P3, P9, P15
Germany	3	FG3-1, FG3-2, FG3-3
India	2	P7, P12
Australia	1	P6
Canada	1	P10
New Zealand	1	P4
Spain	1	P5
Sweden	1	P14
Success Factor based on participant's experience		
Timeboxing	11	P4, P6,P7 ,P9 , P10 , P11, P16, FG3-3, FG1-2, FG2-1, FG2-2
Clear and specific goals and expectation	8	P2, P6, P8, P9, FG3-3, FG1-2, FG2-1, FG2-2
Documentation	4	P5, FG3-1, FG3-3, FG2-1
Clarity and spike creation for the unknown	3	P12, P16, FG2-2
Clear communication	3	P3, P5, P10
Right use of Spike	3	P2, P15, FG1-1
Motivated or Passionate team	2	P7, FG2-1
Organisational or Product Owner Support	2	P8, FG1-3
Shared technical understanding of issue to be addressed	2	P4, P7

Table AP. F5: Sample of the Study themes for the case study

Name	Participant ID	Frequency
Theme 1: Experience in Agile Spike, Agile role and Participants' country		
Experience in Agile		
1-5 yrs	P3, P6, P10, P13, P17, FG1_1, FG1_2, FG1_4, FG3_2, FG3_3	10
6-10 yrs	P1, P4, P7, P8, P16, FG2_1, FG2_2, FG3_1	8
11 – 15 yrs	P5, P9, P12, P14, P15, FG1_3	6
16 – 20 yrs	P2, P11	2
	Experience in Spike	
1-5yrs	P3, P4, P6, P7, P8, P10, P13, P14, P16, P17, FG1_1, FG1_2, FG1_4, FG2_1, FG3_2, FG3_3	16
6-10yrs	P1, P5, FG1_3, FG2_2, FG3_1,	5
11 – 15 yrs	P9, P12, P15,	3
16-20yrs	P2, P11	2
	Agile role	
Developer	P2, P12, FG1_1, FG1_2, FG1_3, FG2_1, FG2_2, FG3_2, FG3_3	9
Agile Coach	P1, P2, P8, P9, P11, P12, P13, FG1_4, FG2_2, FG3_1,	10
Trainer	P11	1
Scrum Master	P3, P4, P5, P6, P10, P12, P14, P17, FG2_1, FG2_2,	10
Product Owner	P15, P16, FG2_2	3
Engineering Manager	P7, FG1_3	2
Consultant	P11	1
	Agile method	
Scrum	P1, P2, P3, P4, P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, FG1_1, FG1_2, FG1_3, FG1_4, FG2_1, FG2_2, FG3_1, FG3_2, FG3_3	24
SAFe	P5, P6,	2
	Organisation Sector	
IT & Software Development	P1, P2, P3, P4, P6, P8, P10 P12, P13, P14, P15, P16, FG1_1, FG1_2, FG1_3, FG1_4, FG2_1, FG2_2, FG3_1, FG3_2, FG3_3	15
Consultancy	P9, P11	2
Government	P10	1
Energy	P7	1
Finance and Banking	P5	1
Education	P17	1

Table AP. F6: Sample of the Study themes for case study

Appendix G: Study questionnaires for RQ1 to RQ3



Spikes in Agile Software Development

You are hereby invited to participate in this study by completing the survey questions based on your knowledge in agile development and the use of spikes. The purpose of the study is to review how spikes are used in agile software development and evaluate their efficiency as well as how effective they are in managing risks during the software development process. Your participation in this survey is very important to improve understanding of the use of spikes in agile software development. I would therefore like you to participate in the survey by answering its questions. Note that all the data collected will be used anonymously and is not assumed to reflect the opinion to the company you are working for.

Who should participate in this study?

This survey seeking to establish the effectiveness of spikes in agile software development. The survey targets professionals and experts in software development and agile methods who have dealt with spikes in their software projects.

This survey has been approved by the ethical committee at the University of Southampton.

Ethical Approval Reference Number: 53962

For more information regarding this survey, you can contact the researcher at this email address: hah2n17@soton.ac.uk

Consent to participate *

☐ Please tick (check) this box to indicate that you consent to taking part in this survey

Next

Page 1 of 4

Figure AP. G1: Questionnaire interface for RQ1 to RQ3

Section A: Demographic details

Experience in Agile? *

Choose ▾

Experience of using Spikes? *

Choose ▾

Your agile role/roles? *

- ☐ Product Owner
- ☐ Scrum Master
- ☐ Agile Coach
- ☐ Developer
- ☐ Tester (QA)
- ☐ Other: ____________

Which agile role do you have most experience of? *

- ○ Product Owner
- ○ Scrum Master
- ○ Agile Coach
- ○ Developer
- ○ Tester (QA)
- ○ Other: ____________

Back | Next | Page 2 of 4

Figure AP. G2: Demographic details for the questionnaire (RQ1 to RQ3)

Section B: Effectiveness and efficiently of spikes in agile software development

1. In your experience with agile development, which is the agile method you have used most? *

- ◯ Scrum
- ◯ Lean
- ◯ Kanban
- ◯ Extreme Programming (XP)
- ◯ Crystal
- ◯ Dynamic Systems Development Method (DSDM)
- ◯ Feature Driven Development (FDD)
- ◯ Other: ____________

2. How often have you used spikes, or seen them used, in agile software development? *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

3. Based on the agile role you have most experience of, to what extent do you agree with the statement that agile spikes are effective in risk management? *

- ◯ Strongly agree
- ◯ Agree
- ◯ Neutral
- ◯ Disagree
- ◯ Strongly Disagree

Figure AP. G3: Questionnaire questions for RQ1 to RQ3

4. In your opinion, how can agile spikes be used efficiently in software development? *

Your answer

5. In your experience with agile software development, do you think that spikes can increase the quality of the product? *

- ◯ Yes
- ◯ No
- ◯ Not sure

6. Which of the following domains have you applied or seen spikes being applied more frequently? Please mark all the applicable fields. *

- ☐ Big data
- ☐ Internet of Things (IoT)
- ☐ Cloud computing
- ☐ User Experience Design (UX)
- ☐ Computer Science Education
- ☐ Data Warehousing
- ☐ Information Security
- ☐ Other: ________

7. Out of the domains you selected or stated above, kindly rate out of 5 points the efficiency of applying spikes in these domains. Where 1 being low and 5 high *

1	2	3	4	5
◯	◯	◯	◯	◯

Figure AP. G4: Questionnaire questions for RQ1 to RQ3

8. Is it likely that software development experts apply spikes to estimate user stories? *

- ◯ Very likely
- ◯ Somewhat Likely
- ◯ Neutral
- ◯ Somewhat Unlikely
- ◯ Very Unlikely

9. Which types of spike do you use mostly in your agile software development? *

- ◯ Technical spikes
- ◯ Functional spikes

10. Which of the two types of spike do you find more effective when used? *

- ◯ Technical spikes
- ◯ Functional spikes
- ◯ Both of them
- ◯ Not sure

What is the rationale for your selection in question 10?

Your answer

11. On a scale of 1-5, where 1 being low and 5 high, please rate the efficiency of spikes in agile software development. *

1	2	3	4	5
◯	◯	◯	◯	◯

12. Based on your agile role, would you advise other experts with a similar role to utilise spikes in their agile software development? *

- ◯ Definitely
- ◯ Probably
- ◯ Possibly
- ◯ Probably Not
- ◯ Definitely Not

Figure AP. G5: Questionnaire questions for RQ1 to RQ3

For questions, 13-17, based on the experience and knowledge you have on your role, do you agree with the following statements? *

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
13. The efficiency of spikes in agile software development depends on the type of applied spikes	○	○	○	○	○
14. The efficiency of spikes in agile software development depends on the team who applied the spikes.	○	○	○	○	○
15. In some instances, spikes do not lead to the desired solution when applied	○	○	○	○	○
16. In most cases, spikes are used in prototyping, exploration, investigation, design and research activities in agile development	○	○	○	○	○
17. Spikes are the best approach for risk management in agile software development	○	○	○	○	○

Back Next Page 3 of 4

Figure AP. G6: Likert-scale questionnaire items for RQ1 to RQ3

Section C: Risk Management through agile spikes

For questions 19-26 please mark ONLY ONE appropriate answer. It is normal for every software development project to have some risks. Agile spikes are considered one of the solutions to reducing the uncertainty that might be present. Based on your experience, to what extent you agree with the following statements. *

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
19. Spikes are effective in reducing uncertainty in agile software development	○	○	○	○	○
20. Spikes can estimate user stories more precisely during the software development process	○	○	○	○	○
21. Spikes should be sparingly used as solutions to problems since they do not yield direct value to the customers	○	○	○	○	○
22. Spikes can be used when uncertainty about a process, system or operation exists	○	○	○	○	○

Figure AP. G7: Likert-scale questionnaire items for RQ1 to RQ3

23. Spikes can potentially increase risk in a project when wrongly applied	○	○	○	○	○
24. I find spikes to be effective in addressing software development risks and uncertainties in new systems	○	○	○	○	○
25. I find spikes more convenient than any other approach when estimating user stories	○	○	○	○	○
26. I believe that risks in agile software development projects can be managed without any application of spikes.	○	○	○	○	○

27. What reason do you have for the answer in question 25 above? *

Your answer

28. Why do you think so in question 26 above? *

Your answer

Figure AP. G8: Likert-scale questionnaire items for RQ1 to RQ3

18. In your opinion, how frequently has your team or organisation utilised agile spikes to minimise risks in your software development projects? *

- ○ Always
- ○ Often
- ○ Sometimes
- ○ Rarely
- ○ Never

29. On a scale of 1-5, where 1 being low and 5 high, how do you rate the effectiveness of spikes in risk management with regards to agile software development? *

1	2	3	4	5
○	○	○	○	○

30. Apart from earning some knowledge about a specific topic, what other roles do spikes play in agile software development? *

Your answer

Back | Submit | Page 4 of 4

Figure AP. G9: Questionnaire questions for RQ1 to RQ3

Appendix H: Study questionnaire for RQ4

Spikes' Success Factors in Agile Software Development

You are hereby invited to participate in this study by completing a few questions based on your knowledge and experience in agile software development and the use of spikes. The purpose of the study is to establish the factors that help to complete the spikes successfully. The outcomes of the survey will be used to come up with a checklist of factors that need to be considered when using spikes in agile software development. I would therefore, like you to consent participating in the survey by answering the questions below. Note that all the data collected will be dealt with securely and confidentiality and does not reflect the opinion of the company you are working for, but instead it represents a personal stance about the subject. Furthermore, you are allowed to decline participating in the study at any time.

Who should participate in this study?

The survey targets professionals and experts in software development and agile methods who have dealt with spikes in their software projects.

This survey has been approved by the ethical committee at the University of Southampton.

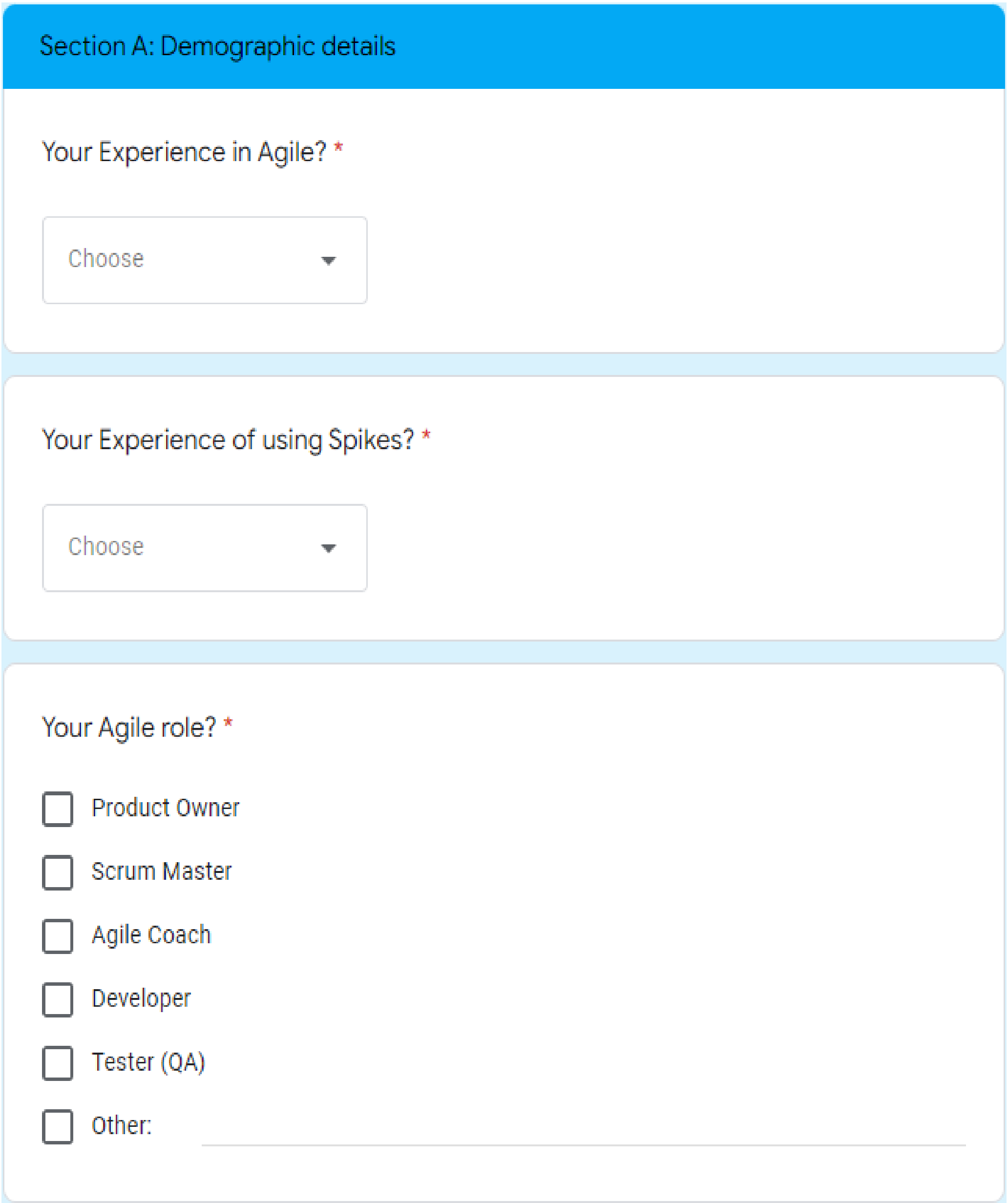
Ethical Approval Reference Number: 62395

For more information regarding this survey, you can contact the researcher at this email address: hah2n17@soton.ac.uk

Consent to participate *

☐ Please tick (check) this box to indicate that you consent to taking part in this survey

Figure AP. H1: Questionnaire interface for RQ4



Section A: Demographic details

Your Experience in Agile? *

Choose

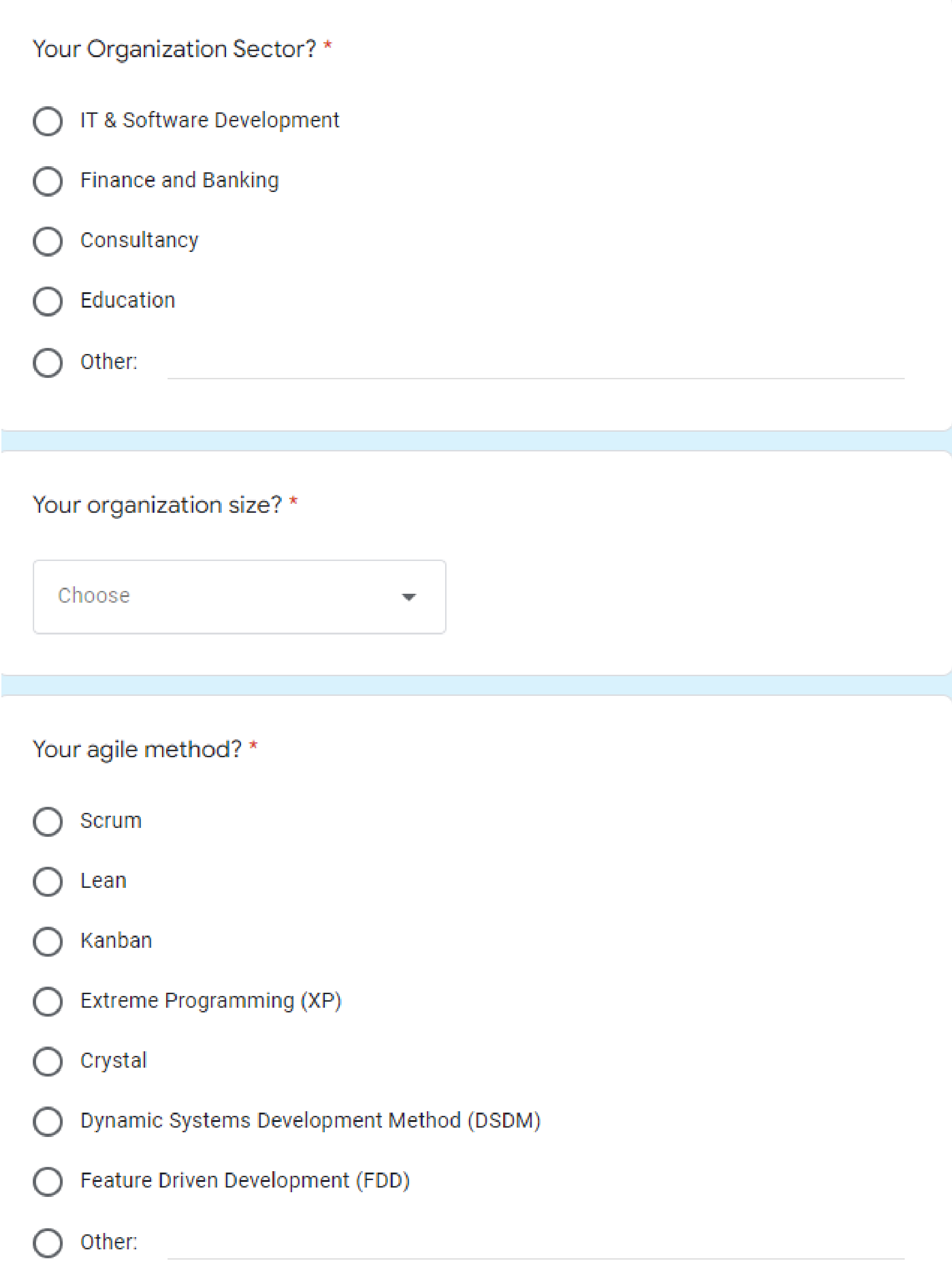
Your Experience of using Spikes? *

Choose

Your Agile role? *

- [ ] Product Owner
- [ ] Scrum Master
- [ ] Agile Coach
- [ ] Developer
- [ ] Tester (QA)
- [ ] Other: ____

Figure AP. H1: Demographic details for the questionnaire (RQ4)



Your Organization Sector? *

- ◯ IT & Software Development
- ◯ Finance and Banking
- ◯ Consultancy
- ◯ Education
- ◯ Other: ______

Your organization size? *

Choose ▾

Your agile method? *

- ◯ Scrum
- ◯ Lean
- ◯ Kanban
- ◯ Extreme Programming (XP)
- ◯ Crystal
- ◯ Dynamic Systems Development Method (DSDM)
- ◯ Feature Driven Development (FDD)
- ◯ Other: ______

Figure AP. H1: Demographic details for the questionnaire (RQ4)

Section B: Spikes' Success Factors

The factors influencing the application of spikes

1. Based on your experience with agile software development and spikes, do you think that there are factors affecting the application of spikes? *

- ◯ Yes
- ◯ No
- ◯ Not sure

2. In most agile software development processes, success factors are categorized into Organisational, People, Process, Technical and Project factors. Which among the following factors do you or your organisation consider when utilizing spikes? Please select all applicable

Organizational *

- ☐ Management commitment
- ☐ Organizational environment
- ☐ Team environment
- ☐ Other: ____________

Figure AP. H1: Questionnaire items for RQ4

People *

- ☐ Team Capability
- ☐ Customer Involvement
- ☐ Motivation
- ☐ Expertise
- ☐ Team size
- ☐ Other: __________

Process *

- ☐ Project management process
- ☐ Project definition process
- ☐ Work schedule
- ☐ Configuration management
- ☐ Other: __________

Technical *

- ☐ Agile software technique
- ☐ Delivery strategy
- ☐ Technical training of the team
- ☐ Other: __________

Figure AP. H1: Questionnaire items for RQ4

Project

- [ ] Project nature
- [ ] Project schedule
- [ ] Project size
- [ ] Other: ____________________

3. Do you think the categorization of the success factors above is fit the actual factors that should be taken into account when applying spikes? *

- ○ Yes
- ○ No

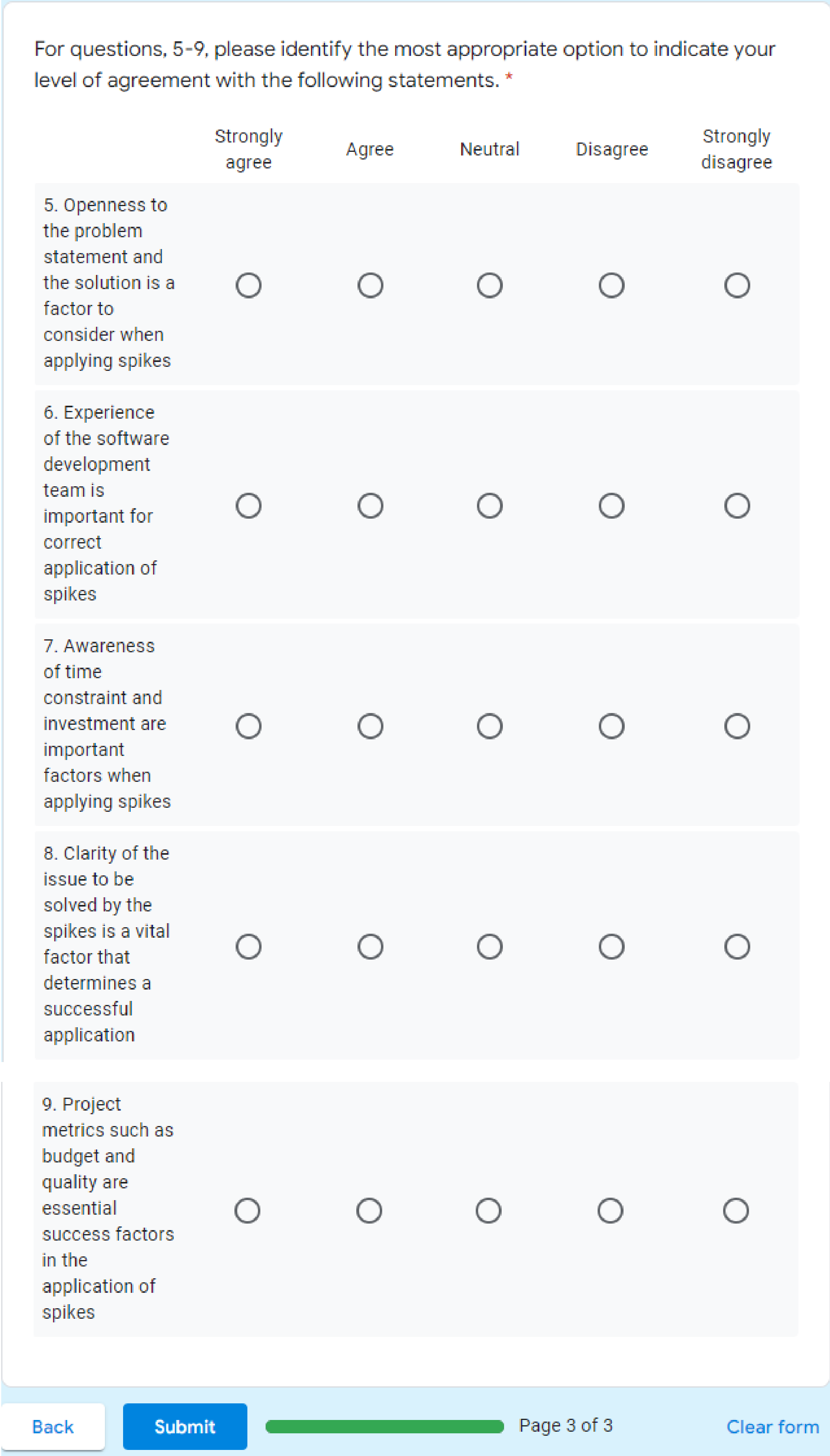
If No, please justify your answer below.

Your answer ____________________

4. Based on your experience, should the product owner or development team take into account some factors that help to apply and complete spikes successfully?

- ○ Yes
- ○ No

Figure AP. H1: Questionnaire items for RQ4



For questions, 5-9, please identify the most appropriate option to indicate your level of agreement with the following statements. *

	Strongly agree	Agree	Neutral	Disagree	Strongly disagree
5. Openness to the problem statement and the solution is a factor to consider when applying spikes	○	○	○	○	○
6. Experience of the software development team is important for correct application of spikes	○	○	○	○	○
7. Awareness of time constraint and investment are important factors when applying spikes	○	○	○	○	○
8. Clarity of the issue to be solved by the spikes is a vital factor that determines a successful application	○	○	○	○	○
9. Project metrics such as budget and quality are essential success factors in the application of spikes	○	○	○	○	○

Back Submit Page 3 of 3 Clear form

Figure AP. H1: Likert-scale questionnaire items for RQ4

Appendix I: Outputs of Questionnaires Analysis for RQ1, RQ2, and RQ3

Descriptive Statistics

	N	Mean	Std. Dev.
Agile Experience	72	6.69	3.143
Spike Experience	72	5.19	3.156
Valid N (listwise)	72		

Frequencies

		Agile Role	Agile Role Experience	Agile Method	Frequency of Using Spikes	Spikes are Efficient in Agile Software Development
N	Valid	72	72	72	72	72
	Missing	0	0	0	0	0

Frequencies

		Can Spikes Improve Product Quality?
N	Valid	72
	Missing	0

Agile role

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Product owner	18	25.0	25.0	25.0
	Scrum Master	50	69.4	69.4	94.4
	Other	4	5.6	5.6	100.0
	Total	72	100.0	100.0	

Agile role experience

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Product owner	12	16.7	16.7	16.7
	Scrum Master	40	55.6	55.6	72.2
	Developer	2	2.8	2.8	75.0
	Tester	1	1.4	1.4	76.4
	Other	17	23.6	23.6	100.0
	Total	72	100.0	100.0	

Agile method

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Scrum	65	90.3	90.3	90.3
	Lean	2	2.8	2.8	93.1
	Kanban	3	4.2	4.2	97.2
	DSDM	1	1.4	1.4	98.6
	Other	1	1.4	1.4	100.0
	Total	72	100.0	100.0	

Frequency of use of spikes					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Rarely	4	5.6	5.6	5.6
	Sometimes	17	23.6	23.6	29.2
	Often	35	48.6	48.6	77.8
	Always	16	22.2	22.2	100.0
	Total	72	100.0	100.0	

Efficiency of spikes in agile software development					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Disagree	2	2.8	2.8	2.8
	Neutral	9	12.5	12.5	15.3
	Agree	39	54.2	54.2	69.4
	Strongly Agree	22	30.6	30.6	100.0
	Total	72	100.0	100.0	

Spikes improve product quality					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Yes	66	91.7	91.7	91.7
	No	5	6.9	6.9	98.6
	Not sure	1	1.4	1.4	100.0
	Total	72	100.0	100.0	

Likelihood of spikes estimating user stories					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Very Unlikely	8	11.1	11.1	11.1
	Somewhat Unlikely	7	9.7	9.7	20.8
	Neutral	18	25.0	25.0	45.8
	Somewhat Likely	25	34.7	34.7	80.6
	Very likely	14	19.4	19.4	100.0
	Total	72	100.0	100.0	

Advising others to use spikes					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Possibly	8	11.1	11.1	11.1
	Probably	20	27.8	27.8	38.9
	Definitely	44	61.1	61.1	100.0
	Total	72	100.0	100.0	

Efficiency of spikes in ASD depends on the type of spike applied					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	4	5.6	5.6	5.6
	Disagree	13	18.1	18.1	23.6
	Neutral	7	9.7	9.7	33.3
	Agree	30	41.7	41.7	75.0
	Strongly Agree	18	25.0	25.0	100.0
	Total	72	100.0	100.0	

Efficiency of spikes in ASD depends on the team applying the spike					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	3	4.2	4.2	4.2
	Disagree	6	8.3	8.3	12.5
	Neutral	8	11.1	11.1	23.6
	Agree	29	40.3	40.3	63.9
	Strongly Agree	26	36.1	36.1	100.0
	Total	72	100.0	100.0	

In some instances, spikes do not lead to the desired solution when applied					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	3	4.2	4.2	4.2
	Disagree	10	13.9	13.9	18.1
	Neutral	9	12.5	12.5	30.6
	Agree	32	44.4	44.4	75.0
	Strongly Agree	18	25.0	25.0	100.0
	Total	72	100.0	100.0	

In most cases, spikes are used in prototyping, exploration, investigation, design, and research activities in agile development					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	1	1.4	1.4	1.4
	Disagree	3	4.2	4.2	5.6
	Neutral	2	2.8	2.8	8.3
	Agree	30	41.7	41.7	50.0
	Strongly Agree	36	50.0	50.0	100.0
	Total	72	100.0	100.0	

Spikes are the best approach for risk management in agile software development					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	2	2.8	2.8	2.8
	Disagree	8	11.1	11.1	13.9
	Neutral	29	40.3	40.3	54.2
	Agree	25	34.7	34.7	88.9
	Strongly Agree	8	11.1	11.1	100.0
	Total	72	100.0	100.0	

Frequency of team or Organisation utilizing agile spikes to minimise risks in software development projects? *					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Rarely	5	6.9	6.9	6.9
	Sometimes	22	30.6	30.6	37.5
	Often	36	50.0	50.0	87.5
	Always	9	12.5	12.5	100.0
	Total	72	100.0	100.0	

Spikes are effective in reducing uncertainty in agile software development					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Neutral	4	5.6	5.6	5.6
	Agree	29	40.3	40.3	45.8
	Strongly Agree	39	54.2	54.2	100.0
	Total	72	100.0	100.0	

Spikes can estimate user stories more precisely during the software development process					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	3	4.2	4.2	4.2
	Disagree	4	5.6	5.6	9.7
	Neutral	18	25.0	25.0	34.7
	Agree	30	41.7	41.7	76.4
	Strongly Agree	17	23.6	23.6	100.0
	Total	72	100.0	100.0	

Spikes should be sparingly used as solutions to problems since they do not yield direct value to the customers					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	10	13.9	13.9	13.9
	Disagree	15	20.8	20.8	34.7
	Neutral	11	15.3	15.3	50.0
	Agree	19	26.4	26.4	76.4
	Strongly Agree	17	23.6	23.6	100.0
	Total	72	100.0	100.0	

Spikes can be used when uncertainty about a process, system, or operation exists					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	2	2.8	2.8	2.8
	Disagree	3	4.2	4.2	6.9
	Neutral	6	8.3	8.3	15.3
	Agree	30	41.7	41.7	56.9
	Strongly Agree	31	43.1	43.1	100.0
	Total	72	100.0	100.0	

Spikes can potentially increase risk in a project when wrongly applied					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	8	11.1	11.1	11.1
	Disagree	9	12.5	12.5	23.6
	Neutral	19	26.4	26.4	50.0
	Agree	19	26.4	26.4	76.4
	Strongly Agree	17	23.6	23.6	100.0
	Total	72	100.0	100.0	

Spikes are efficient in addressing software development risks and uncertainties in new systems					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Neutral	16	22.2	22.2	22.2
	Agree	38	52.8	52.8	75.0
	Strongly Agree	18	25.0	25.0	100.0
	Total	72	100.0	100.0	

Spikes are more convenient than any other approach when estimating user stories					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	8	11.1	11.1	11.1
	Disagree	24	33.3	33.3	44.4
	Neutral	26	36.1	36.1	80.6
	Agree	11	15.3	15.3	95.8
	Strongly Agree	3	4.2	4.2	100.0
	Total	72	100.0	100.0	

Risks in agile software development projects can be managed without any application of spikes					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly Disagree	3	4.2	4.2	4.2
	Disagree	21	29.2	29.2	33.3
	Neutral	14	19.4	19.4	52.8
	Agree	29	40.3	40.3	93.1
	Strongly Agree	5	6.9	6.9	100.0
	Total	72	100.0	100.0	

Types of spikes commonly used					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Technical	57	79.2	79.2	79.2
	Functional	15	20.8	20.8	100.0
	Total	72	100.0	100.0	

Most efficient type of spike					
		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Technical	31	43.1	43.1	43.1
	Functional	2	2.8	2.8	45.8
	Both	36	50.0	50.0	95.8
	Not sure	3	4.2	4.2	100.0
	Total	72	100.0	100.0	

Descriptive statistics			
	N	Mean	Std. Dev.
Spikes Efficiency in Domains	72	4.06	.690
Spike Efficiency in Agile Development	72	4.13	.711
Spike Effectiveness in Risk Management	72	3.92	.707
Valid N (listwise)	72		

Statistical Tests

Correlations

Descriptive Statistics

	Mean	Std. Dev.	N
Spike Efficiency in Domains	4.06	.690	72
Spike Effectiveness in Risk Management	3.92	.707	72

		Spike Efficiency in Domains	Spike Effectiveness in Risk Management
Spike Efficiency in Domains	Pearson Correlation	1	.356**
	Sig. (2-tailed)		.002
	N	72	72
Spike Effectiveness in Risk Management	Pearson Correlation	.356**	1
	Sig. (2-tailed)	.002	
	N	72	72

** Correlation significant at the 0.01 level (2-tailed).

Descriptive Statistics

	Mean	Std. Dev.	N
Spike Effectiveness in Risk Management	3.92	.707	72
Spike Efficiency in Agile Development	4.13	.711	72

		Spike Effectiveness in Risk Management	Spike Efficiency in Agile Development
Spike Effectiveness in Risk Management	Pearson Correlation	1	.413**
	Sig. (2-tailed)		.000
	N	72	72
Spike Efficiency in Agile Development	Pearson Correlation	.413**	1
	Sig. (2-tailed)	.000	
	N	72	72

** Correlation significant at the 0.01 level (2-tailed).

Regression

Variables Entered/Removed[a]

Model	Variables Entered	Variables Removed	Method
1	Spike Efficiency in Agile Development, Spike Efficiency in Domains[b]	–	Enter

[a] Dependent variable: Spike effectiveness in risk management
[b] All requested variables entered

Coefficients[a]

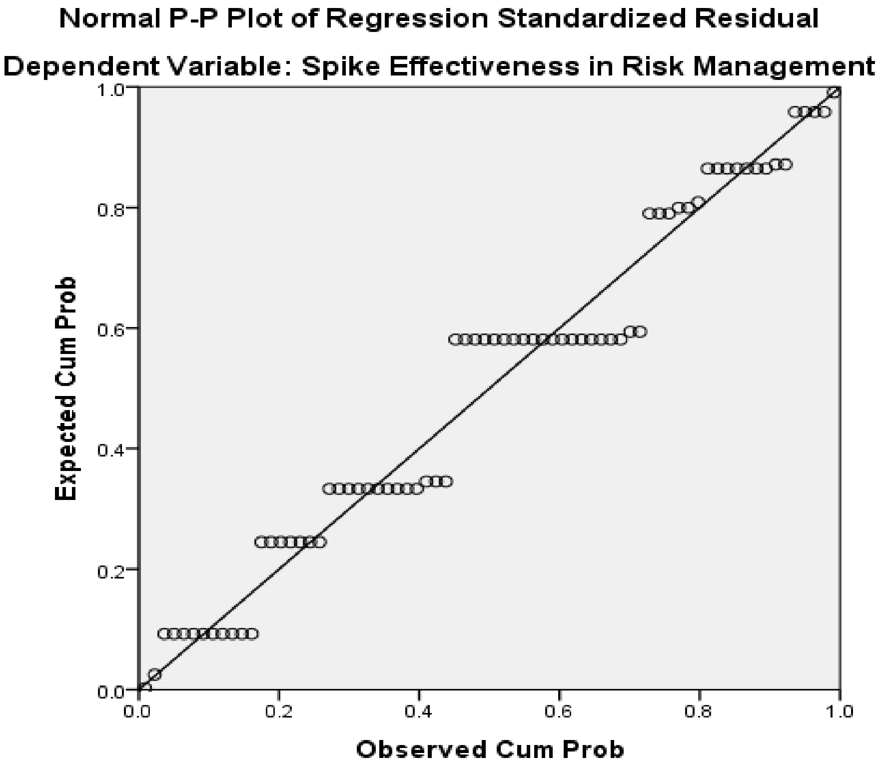
Model		Unstandardised Coefficients		Standardised Coefficients	t	Sig.
		β	Std. Error	Beta		
1	(Constant)	2.206	.477		4.621	.000
	Spike Efficiency in Domains	.021	.212	.021	.101	.919
	Spike Efficiency in Agile Development	.393	.205	.396	1.916	.060

[a] Dependent variable: Spike effectiveness in risk management

Residuals[a]

	Minimum	Maximum	Mean	Std. Dev.	N
Predicted Value	3.43	4.28	3.92	.292	72
Residual	-1.866	1.549	.000	.644	72
Std. Predicted Value	-1.665	1.247	.000	1.000	72
Std. Residual	-2.858	2.371	.000	.986	72

[a] Dependent variable: Spike effectiveness in risk management



One-Sample t-Test

	Test Value = 3				
	t	df	Sig. (2-tailed)	Mean Difference	95% CI Lower
Q13) The efficiency of spikes in agile software development depends on the type of spike applied	4.406	71	.000	.625	.34
Q14) The efficiency of spikes in agile software development depends on the team applying the spike	7.437	71	.000	.958	.70
Q15) In some instances, spikes do not lead to the desired solution when applied	5.491	71	.000	.722	.46
Q16) In most cases, spikes are used in prototyping, exploration, investigation, design, and research activities in agile development	13.579	71	.000	1.347	1.15
Q17) Spikes are the best approach for risk management in agile software development	3.678	71	.000	.403	.18

One-Sample t-Test

	Test Value = 3
	95% CI Upper
Q13) The efficiency of spikes in agile software development depends on the type of spike applied	.91
Q14) The efficiency of spikes in agile software development depends on the team applying the spike	1.22
Q15) In some instances, spikes do not lead to the desired solution when applied	.98
Q16) In most cases, spikes are used in prototyping, exploration, investigation, design, and research activities in agile development	1.55
Q17) Spikes are the best approach for risk management in agile software development	.62

One-Sample t-Test

	Test Value = 3				
	t	df	Sig. (2-tailed)	Mean Difference	95% CI Lower
Q19) Spikes are effective in reducing uncertainty in agile software development	20.844	71	.000	1.486	1.34
Q20) Spikes can estimate user stories more precisely during the software development process	6.255	71	.000	.750	.51
Q21) Spikes should be used sparingly as solutions to problems since they do not yield direct value to the customers	1.524	71	.132	.250	-.08
Q22) Spikes can be used when uncertainty about a process, system or operation exists	10.498	71	.000	1.181	.96
Q23) Spikes can potentially increase risk in a project when wrongly applied	2.569	71	.012	.389	.09
Q24) I find spikes to be efficient in addressing software development risks and uncertainties in new systems	12.613	71	.000	1.028	.87
Q25) I find spikes more convenient than any other approach when estimating user stories	-2.698	71	.009	-.319	-.56
Q26) I believe that risks in agile software development projects can be managed without any application of spikes.	1.332	71	.187	.167	-.08

One-Sample t-Test

	Test Value = 3
	95% CI Upper
Q19) Spikes are effective in reducing uncertainty in agile software development	1.63
Q20) Spikes can estimate user stories more precisely during the software development process	.99
Q21) Spikes should be used sparingly as solutions to problems since they do not yield direct value to the customers	.58
Q22) Spikes can be used when uncertainty about a process, system or operation exists	1.40
Q23) Spikes can potentially increase risk in a project when wrongly applied	.69
Q24) I find spikes to be efficient in addressing software development risks and uncertainties in new systems	1.19
Q25) I find spikes more convenient than any other approach when estimating user stories	-.08
Q26) I believe that risks in agile software development projects can be managed without any application of spikes	.42

Reliability

Scale: ALL VARIABLES

Case Processing Summary

		N	%
Cases	Valid	72	100.0
	Excluded[a]	0	.0
	Total	72	100.0

[a] Listwise deletion based on all variables in the procedure.

Summary Item Statistics

	Mean	Minimum	Maximum	Range	Max/Min	Variance	N
Item Means	3.719	2.681	4.486	1.806	1.674	.238	15

Reliability Statistics

Cronbach's alpha	Cronbach's alpha based on standardised items	N
.525	.568	15

Reliability Statistics

Cronbach's alpha	No. of Items
.778	3

Univariate Analysis of Variance

Between-Subjects Factors

		Value Label	N
Spikes can estimate user stories more precisely during the software development process	1	Strongly Disagree	3
	2	Disagree	4
	3	Neutral	18
	4	Agree	30
	5	Strongly Agree	17

Descriptive Statistics

Dependent Variable: Spike Effectiveness in Risk Management

Spikes can estimate user stories more precisely during the software development process	Mean	Std. Dev.	N
Strongly Disagree	3.33	.577	3
Disagree	3.75	1.500	4
Neutral	3.78	.647	18
Agree	3.93	.640	30
Strongly Agree	4.18	.636	17
Total	3.92	.707	72

Tests of Between-Subjects Effects

Dependent Variable: Spike Effectiveness in Risk Management

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Corrected Model	2.635[a]	4	.659	1.343	.263	.074
Intercept	492.302	1	492.302	1003.628	.000	.937
Q20	2.635	4	.659	1.343	.263	.074
Error	32.865	67	.491			
Total	1140.000	72				
Corrected Total	35.500	71				

[a] R Squared = .074 (Adjusted R Squared = .019)

Univariate Tests

Dependent Variable: Spike Effectiveness in Risk Management

	Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Contrast	2.635	4	.659	1.343	.263	.074
Error	32.865	67	.491			

The F test analyses the effect spikes can have in estimating user stories more precisely during the software development process. This test is based on linearly independent pairwise comparisons among the estimated marginal means.

Post-Hoc Tests

Spikes can estimate user stories more precisely during the software development process

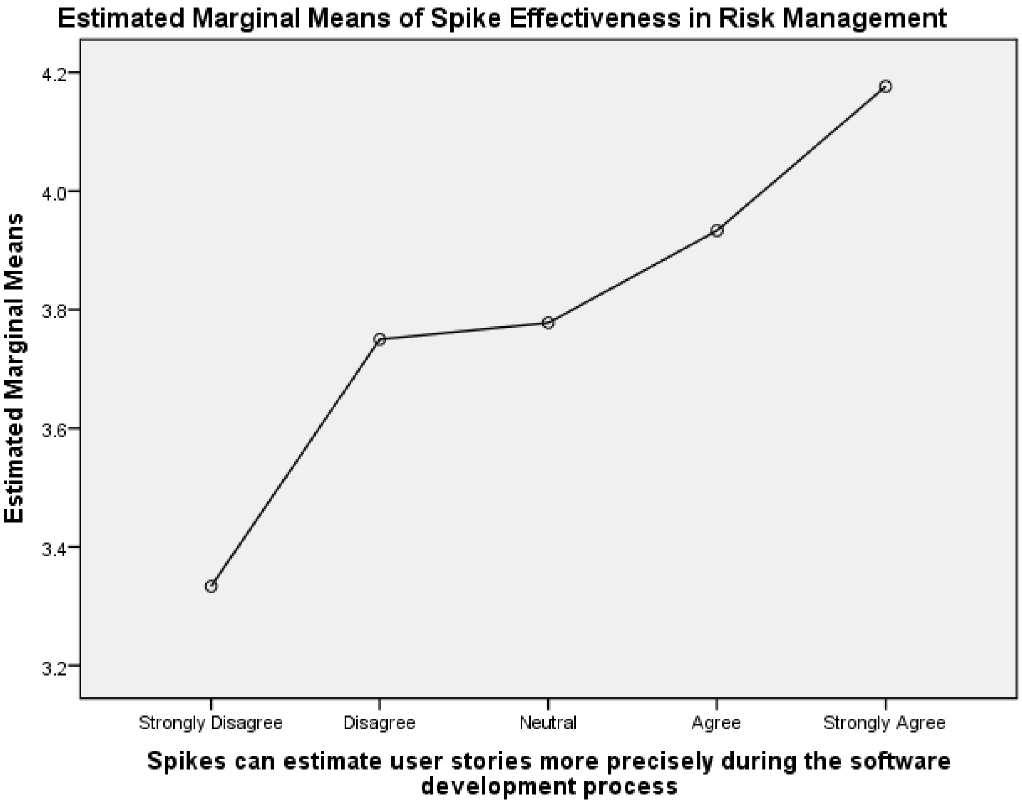
Multiple Comparisons
Dependent Variable: Spike Effectiveness in Risk Management
Bonferroni

(I) Spikes can estimate user stories more precisely during the software development process	(J) Spikes can estimate user stories more precisely during the software development process	Mean Difference (I-J)	Std. Error	Sig.
Strongly Disagree	Disagree	-.42	.535	1.000
	Neutral	-.44	.437	1.000
	Agree	-.60	.424	1.000
	Strongly Agree	-.84	.439	.588
Disagree	Strongly Disagree	.42	.535	1.000
	Neutral	-.03	.387	1.000
	Agree	-.18	.373	1.000
	Strongly Agree	-.43	.389	1.000
Neutral	Strongly Disagree	.44	.437	1.000
	Disagree	.03	.387	1.000
	Agree	-.16	.209	1.000
	Strongly Agree	-.40	.237	.970
Agree	Strongly Disagree	.60	.424	1.000
	Disagree	.18	.373	1.000
	Neutral	.16	.209	1.000
	Strongly Agree	-.24	.213	1.000
Strongly Agree	Strongly Disagree	.84	.439	.588
	Disagree	.43	.389	1.000
	Neutral	.40	.237	.970
	Agree	.24	.213	1.000

Multiple Comparisons
Dependent Variable: Spike Effectiveness in Risk Management
Bonferroni

(I) Spikes can estimate user stories more precisely during the software development process	(J) Spikes can estimate user stories more precisely during the software development process	95% CI	
		Lower Bound	Upper Bound
Strongly Disagree	Disagree	-1.97	1.14
	Neutral	-1.71	.82
	Agree	-1.83	.63
	Strongly Agree	-2.12	.43
Disagree	Strongly Disagree	-1.14	1.97
	Neutral	-1.15	1.10
	Agree	-1.27	.90
	Strongly Agree	-1.56	.70
Neutral	Strongly Disagree	-.82	1.71
	Disagree	-1.10	1.15
	Agree	-.76	.45
	Strongly Agree	-1.09	.29
Agree	Strongly Disagree	-.63	1.83
	Disagree	-.90	1.27
	Neutral	-.45	.76
	Strongly Agree	-.86	.37
Strongly Agree	Strongly Disagree	-.43	2.12
	Disagree	-.70	1.56
	Neutral	-.29	1.09
	Agree	-.37	.86

Profile Plots



Descriptive Statistics

Dependent Variable: Spike Effectiveness in Risk Management

Spikes are the best approach for risk management in agile software development	Mean	Std. Dev.	N
Strongly Disagree	3.50	.707	2
Disagree	3.50	.926	8
Neutral	3.79	.726	29
Agree	4.04	.539	25
Strongly Agree	4.50	.535	8
otal	3.92	.707	72

Tests of Between-Subjects Effects

Dependent Variable: Spike Effectiveness in Risk Management

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Corrected Model	5.281[a]	4	1.320	2.927	.027	.149
Intercept	453.337	1	453.337	1005.129	.000	.938
Q17	5.281	4	1.320	2.927	.027	.149
Error	30.219	67	.451			
Total	1140.000	72				
Corrected Total	35.500	71				

[a] R Squared = .149 (Adjusted R Squared = .098)

Estimated Marginal Means

Spikes are the best approach for risk management in agile software development
Dependent Variable: Spike Effectiveness in Risk Management

Spikes are the best approach for risk management in agile software development	Mean	Std. Error	95% CI	
			Lower Bound	Upper Bound
Strongly Disagree	3.500	.475	2.552	4.448
Disagree	3.500	.237	3.026	3.974
Neutral	3.793	.125	3.544	4.042
Agree	4.040	.134	3.772	4.308
Strongly Agree	4.500	.237	4.026	4.974

Post Hoc Tests

Spikes are the best approach for risk management in agile software development

Multiple Comparisons
Dependent Variable: Spike Effectiveness in Risk Management
Bonferroni

(I) Spikes are the best approach for risk management in agile software development	(J) Spikes are the best approach for risk management in agile software development	Mean Difference (I-J)	Std. Error	Sig.
Strongly Disagree	Disagree	.00	.531	1.000
	Neutral	-.29	.491	1.000
	Agree	-.54	.494	1.000
	Strongly Agree	-1.00	.531	.640
Disagree	Strongly Disagree	.00	.531	1.000
	Neutral	-.29	.268	1.000
	Agree	-.54	.273	.519
	Strongly Agree	-1.00*	.336	.040
Neutral	Strongly Disagree	.29	.491	1.000
	Disagree	.29	.268	1.000
	Agree	-.25	.183	1.000
	Strongly Agree	-.71	.268	.104
Agree	Strongly Disagree	.54	.494	1.000
	Disagree	.54	.273	.519
	Neutral	.25	.183	1.000
	Strongly Agree	-.46	.273	.964
Strongly Agree	Strongly Disagree	1.00	.531	.640
	Disagree	1.00*	.336	.040
	Neutral	.71	.268	.104
	Agree	.46	.273	.964

Multiple Comparisons

Dependent Variable: Spike Effectiveness in Risk Management

Bonferroni

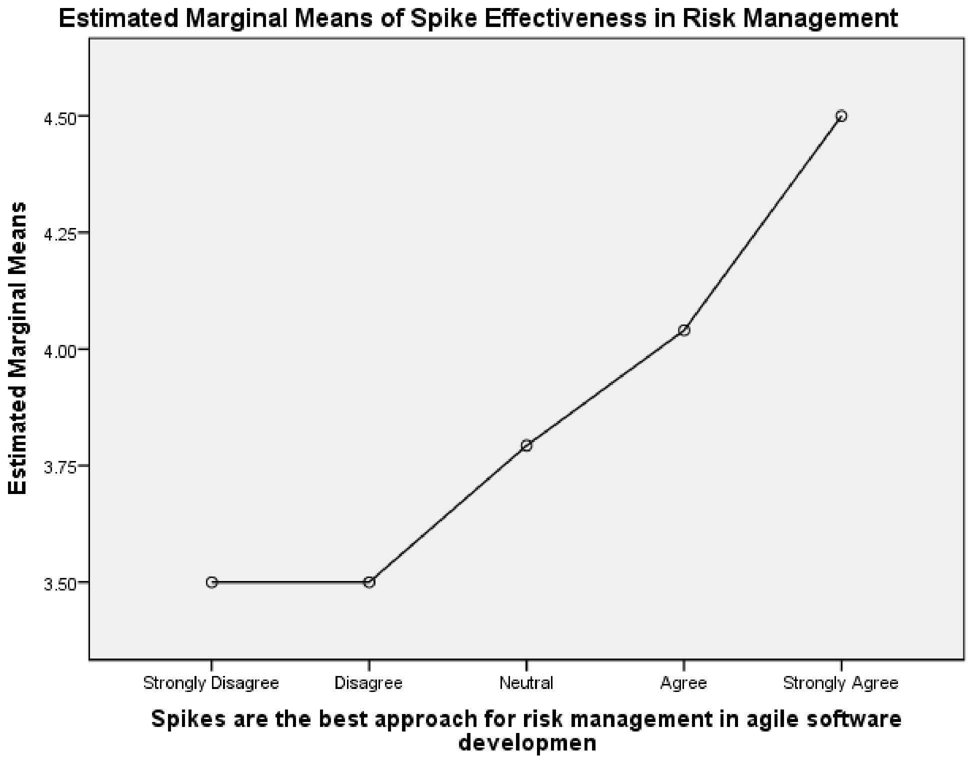
(I) Spikes are the best approach for risk management in agile software development	(J) Spikes are the best approach for risk management in agile software development	95% CI	
		Lower Bound	Upper Bound
Strongly Disagree	Disagree	-1.54	1.54
	Neutral	-1.72	1.13
	Agree	-1.97	.89
	Strongly Agree	-2.54	.54
Disagree	Strongly Disagree	-1.54	1.54
	Neutral	-1.07	.49
	Agree	-1.33	.25
	Strongly Agree	-1.97	-.03
Neutral	Strongly Disagree	-1.13	1.72
	Disagree	-.49	1.07
	Agree	-.78	.29
	Strongly Agree	-1.49	.07
Agree	Strongly Disagree	-.89	1.97
	Disagree	-.25	1.33
	Neutral	-.29	.78
	Strongly Agree	-1.25	.33
Strongly Agree	Strongly Disagree	-.54	2.54
	Disagree	.03	1.97
	Neutral	-.07	1.49
	Agree	-.33	1.25

Based on observed means.

The error term is mean square (error) = 0.451.

* Mean difference significant at the 0.05 level.

Profile Plots



Appendix J: Outputs of Questionnaires Analysis for RQ4

Statistics

		AgileExp	SpikeExp	OrgSect	OrgSize	AgileMethod	FASA	Categorisation	Factors Consideration
N	Valid	64	64	64	64	64	64	64	64
	Missing	0	0	0	0	0	0	0	0

Agile Exp

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Up to 1 year	2	3.1	3.1	3.1
	2 years	1	1.6	1.6	4.7
	3 yaers	6	9.4	9.4	14.1
	4 years	5	7.8	7.8	21.9
	5 years	6	9.4	9.4	31.3
	6 years	4	6.3	6.3	37.5
	7 years	4	6.3	6.3	43.8
	8 years	5	7.8	7.8	51.6
	9 years	3	4.7	4.7	56.3
	10 years	5	7.8	7.8	64.1
	Between 11 and 15 years	15	23.4	23.4	87.5
	Between 16 and 20 years	8	12.5	12.5	100.0
	Total	64	100.0	100.0	

Spike Exp

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Up to 1 year	10	15.6	15.6	15.6
	2 years	7	10.9	10.9	26.6
	3 yaers	6	9.4	9.4	35.9
	4 years	3	4.7	4.7	40.6
	5 years	9	14.1	14.1	54.7
	6 years	9	14.1	14.1	68.8
	7 years	2	3.1	3.1	71.9
	8 years	2	3.1	3.1	75.0
	9 years	1	1.6	1.6	76.6
	Between 11 and 15 years	11	17.2	17.2	93.8
	Between 16 and 20 years	4	6.3	6.3	100.0
	Total	64	100.0	100.0	

Org Sect

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	IT & Software Development	41	64.1	64.1	64.1
	Government	2	3.1	3.1	67.2
	Consultancy	9	14.1	14.1	81.3
	Aviation	1	1.6	1.6	82.8
	Finance & Banking	7	10.9	10.9	93.8
	Logistics	1	1.6	1.6	95.3
	Healthcare	1	1.6	1.6	96.9
	Education	1	1.6	1.6	98.4
	Product Management	1	1.6	1.6	100.0
	Total	64	100.0	100.0	

Org Size

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Micro	6	9.4	9.4	9.4
	Small	6	9.4	9.4	18.8
	Medium	12	18.8	18.8	37.5
	Large	40	62.5	62.5	100.0
	Total	64	100.0	100.0	

Agile Method

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Scrum	49	76.6	76.6	76.6
	Kanban	3	4.7	4.7	81.3
	XP	2	3.1	3.1	84.4
	SAFe	3	4.7	4.7	89.1
	More than one	7	10.9	10.9	100.0
	Total	64	100.0	100.0	

FASA

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Yes	59	92.2	92.2	92.2
	No	2	3.1	3.1	95.3
	3	3	4.7	4.7	100.0
	Total	64	100.0	100.0	

Categorisation

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Yes	46	71.9	71.9	71.9
	No	18	28.1	28.1	100.0
	Total	64	100.0	100.0	

Factors Consideration

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Yes	59	92.2	92.2	92.2
	No	5	7.8	7.8	100.0
	Total	64	100.0	100.0	

Descriptive Statistics

	N	Mean	Std. Deviation
Q5	64	4.50	.713
Q6	64	3.94	.957
Q7	64	4.22	.951
Q8	64	4.41	.886
Q9	64	3.22	1.161
Valid N (listwise)	64		

Q5

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Disagree	1	1.6	1.6	1.6
	Neutral	5	7.8	7.8	9.4
	Agree	19	29.7	29.7	39.1
	Strongly Agree	39	60.9	60.9	100.0
	Total	64	100.0	100.0	

Q6

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly DIsagree	1	1.6	1.6	1.6
	Disagree	3	4.7	4.7	6.3
	Neutral	16	25.0	25.0	31.3
	Agree	23	35.9	35.9	67.2
	Strongly Agree	21	32.8	32.8	100.0
	Total	64	100.0	100.0	

Q7

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly DIsagree	2	3.1	3.1	3.1
	Disagree	2	3.1	3.1	6.3
	Neutral	5	7.8	7.8	14.1
	Agree	26	40.6	40.6	54.7
	Strongly Agree	29	45.3	45.3	100.0
	Total	64	100.0	100.0	

Q8

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Disagree	4	6.3	6.3	6.3
	Neutral	5	7.8	7.8	14.1
	Agree	16	25.0	25.0	39.1
	Strongly Agree	39	60.9	60.9	100.0
	Total	64	100.0	100.0	

Q9

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Strongly DIsagree	6	9.4	9.4	9.4
	Disagree	9	14.1	14.1	23.4
	Neutral	24	37.5	37.5	60.9
	Agree	15	23.4	23.4	84.4
	Strongly Agree	10	15.6	15.6	100.0
	Total	64	100.0	100.0	

Case Processing Summary

		N	%
Cases	Valid	64	100.0
	Excluded[a]	0	.0
	Total	64	100.0

a. Listwise deletion based on all variables in the procedure.

Reliability Statistics

Cronbach's Alpha	Cronbach's Alpha Based on Standardized Items	N of Items
.570	.575	5

Correlations

		Q5	Q6	Q7	Q8	Q9
Q5	Pearson Correlation	1	.326**	.141	.277*	.038
	Sig. (2-tailed)		.009	.268	.027	.763
	N	64	64	64	64	64
Q6	Pearson Correlation	.326**	1	.277*	.161	.312*
	Sig. (2-tailed)	.009		.027	.203	.012
	N	64	64	64	64	64
Q7	Pearson Correlation	.141	.277*	1	.100	.215
	Sig. (2-tailed)	.268	.027		.431	.088
	N	64	64	64	64	64
Q8	Pearson Correlation	.277*	.161	.100	1	.283*
	Sig. (2-tailed)	.027	.203	.431		.024
	N	64	64	64	64	64
Q9	Pearson Correlation	.038	.312*	.215	.283*	1
	Sig. (2-tailed)	.763	.012	.088	.024	
	N	64	64	64	64	64

**. Correlation is significant at the 0.01 level (2-tailed).

*. Correlation is significant at the 0.05 level (2-tailed).

T-Test

One-Sample Statistics

	N	Mean	Std. Deviation	Std. Error Mean
Q5	64	4.50	.713	.089
Q6	64	3.94	.957	.120
Q7	64	4.22	.951	.119
Q8	64	4.41	.886	.111
Q9	64	3.22	1.161	.145

One-Sample Test

	Test Value = 4.0					
					95% Confidence Interval of the Difference	
	t	df	Sig. (2-tailed)	Mean Difference	Lower	Upper
Q5	5.612	63	.000	.500	.32	.68
Q6	-.522	63	.603	-.062	-.30	.18
Q7	1.841	63	.070	.219	-.02	.46
Q8	3.669	63	.001	.406	.18	.63
Q9	-5.383	63	.000	-.781	-1.07	-.49