Software Testing: An Analysis of the Impacts of Test Automation on Software's Cost, Quality and Time

George V Murazvu

Submitted in fulfilment of the requirements of University of Huddersfield

For the Degree of:

Computer science and Informatics (MSc by Research)

School of Computing and Engineering

University of Huddersfield

April 2020

Supervised by Dr Simon Parkinson

LEFT BLANK INTENTIONALLY

Abstract

Software testing is an essential yet time consuming and tedious task in the software development cycle despite the accessibility of most capable quality assurance teams and tools. Test automation is widely being utilised within the software industry to provide increased testing capacity to ensure high product quality and reliability. This thesis will specifically be addressing automated testing whereby test cases are manually written and executed automated. Test automation has its benefits, drawbacks, and impacts on different stages of development. Furthermore, there is often a disconnect between non-technical and technical roles, where non-technical roles (e.g., management) predominantly strive to reduce costs and delivery time whereas technical roles are often driven by quality and completeness. Although it is widely understood that there are challenges with adopting and using automated testing, there is a lack of evidence to understand the different attitudes toward automated testing, focusing specifically on why it is not adopted. In this thesis, the author has surveyed practitioners within the software industry from different roles to determine common trends and draw conclusions. A two-stage approach is presented, comprising of a comprehensive descriptive analysis and the use of Principle Component Analysis (PCA). In total, 81 participants were provided with a series of 22 questions and their responses were compared against job role types and experience levels. In summary, 6 key findings are presented covering expertise, time, cost, tools and techniques, utilisation, organisation and capacity.

Keywords: Software testing, Manual testing, Automated testing, Attitudes, Principle Component Analysis

Candidate's Declaration

I, George Murazvu confirm that, the research work done, which shall be presented in this thesis is my own achievement.

Where I have consulted the published work of others, this is always clearly attributed. Where I have quoted from the work of others, the source is always given. Except for such quotations, this dissertation is entirely my own work;

I have acknowledged all main sources of help. I have read and understand the penalties associated with academic misconduct.

George Murazvu

Date 15/04/2020

Student ID: 1766085318

Contents

List of Figures

List of Tables

Acknowledgement

It would have been a challenging and difficult path, from the start to the end of this Dissertation without close support and support of my supervisor Dr Simon Parkinson, and friends at University of Huddersfield.

My deepest appreciation goes to Dr Simon Parkinson, as my principal supervisor. Has shown close cooperation from the start of the project. I would like to appreciate his effort, patience and the contribution he has given and quick response to queries concerning the project. The response was outstanding, giving me treasured suggestions and remarks that was a positive contribution, which gave me motivation and aided to the outcome and improved quality of my work. Dr Simon Parkinson's firm suggestion that the project might be difficult to accomplish with the limited time available, leading to the change of project scope.

My last point of thanks goes to my wife, children and brother who were with me alongside in moral boosting and support.

Chapter 1

1.0 Introduction

The development of computer science, software engineering, and the increasing use of artificial intelligence and data mining technologies has led to the development of a wide range of applications that are critical to operations in business, health care, and education. Unfortunately, the development of software is a complex and expensive process, prone to errors and subsequent failure to meet user requirements [7]. Organisations therefore invest significant resources into ensuring that software products are tested against set criteria, ensuring they are of the best quality before being released to their clients and users [3]. Traditionally testing has been a manual process, involving humans executing applications and comparing their behaviour against certain benchmarks. However, advances in technology and the constant desire to improve quality have introduced and increased the use in automated testing, which uses computer algorithms to detect and highlight bugs in software applications [9]. Automated testing can generally be categorised into two types: that where manual unit test cases are written and used by automated testing tools, or a framework whereby the testing tools automatically generate unit test cases. In this study, the focus is on the first type where unit test cases are manually created. The phrase 'automated testing' is used throughout the rest of the paper and is referring to instances of automated testing involving the manual creation and automated use of unit testing.

1.1 Problem Statement

A key aspect to software development frameworks is that they all have distinct testing phases. For example, the Waterfall model has a distinct test phase after development has taken place [11]. Although there are frameworks involving iterative and concurrent testing, many development frameworks assume users can specify a finished set of requirements in advance, ignoring the fact that they develop as the project progresses and changes depending on the client's circumstances. Manual testing and the correction of errors, as well as the integration of changes, is feasible in small projects as the code size is small and easy to manage. However, as client requirements change or more requirements are added, the projects grow in complexity, yielding more lines of code and a higher probability of software faults occurring (commonly named bugs). This results in the need for an increased frequency of manual software testing. Consequently, there has been a shift to more flexible methodologies like Agile model that combine testing with the completion of each phase to identify software problems before progressing to the next phase.

1.2 Project Goal

At the end of this research, it is foreseen that the following question will be answered: do common themes emerge when investigating opinions as to why automated testing is not used, with the focus being on job role and level of experience? To answer this research question, a twenty-two-question survey has been created to collect attitudes toward automated testing from employees working in the software testing industry. The data is then thoroughly analysed by using quantitative techniques to determine key patterns and themes.

1.2.1 Objectives

- Review research literature to gain an understanding of research in the area;
- Establish key automation software testing benefits versus manual testing;
- Design and circulate questionnaire-based questions to sample of software testing professionals; and
- Analyse the data to identify and discuss key findings regarding test automation.

1.2.2 Justification

Automated software testing has many well-established known benefits; however, several organisations are still not using automation techniques. The results from the 2018 State of Testing Report survey on test automation highlights that automation is not yet as common as organisations desire. There are still many factors hindering update and use, such as challenges in acquiring and maintaining expertise, cost, and the utilisation of the correct testing tools and frameworks. Although previous studies present the reasons as to why automated testing might not be used, there is an absence of literature focusing on different job roles and experiences. There is also debate amongst academics and professionals as to the merits of automated testing over traditional testing methods [32]. This research thesis presents an empirical study to gain an understanding of the different attitudes of employees working within the software industry. The particular focus of this research is to understand whether there are common patterns surrounding different roles and level of experience. Furthermore, this research aims to identify common reasons as to why automation is not being used.

1.3 Project Structure

This paper is structured as follows: Chapter 2 presents and discusses existing work, grounding this study in relevant literature. Chapter 3 describes and justifies the process adopted in this thesis, which includes using a two-stage analysis approach. This section also presents and discusses the results of the study in detail, identifying common themes pertinent to the aim of this study. Chapter 4 provides a summary of key findings, discussing how these findings motivate future work. Finally, in Chapter 5 a conclusion of the work is provided. The full set of participant responses are available in Appendix 6.

Chapter 2

2.0 Literature Review

2.1 Introduction

The purpose of this thesis is to identify the mindsets of those working in the software testing industry. This section surveys academic works which tackle this question, comparing any existing approaches and methodologies. In one recent study, the author defines manual testing as a procedure to test the product for discovering software bugs [3]. Software is erroneous if it deviates from the system requirements and/or implements any requirement incorrectly. Taipale et al. agree by stating that manual software testing is the procedure of physically testing software for defects, and it requires a tester to assume the job of an end-user whereby they utilise the application's features to ensure correct functionality [32]. Also known as functional testing. There are several types of software testing that target different objectives, such as effectiveness, efficiency, user satisfaction, completeness, defect types, etc. A methodological framework has been developed for this purpose that outputs a set of guidelines and checklists on what type of testing should be applied to achieve a certain objective based on a given case study [35].

2.2 Definition of terms

2.2.1 Software testing

Software testing is characterised as an action to check whether the actual outcomes tie the expected outcomes and to guarantee that the software system is free form defects. It includes execution of a system part or system segment to assess at least one or more properties of interest and can be done either manually or by automated tools (Wikipedia). In a nutshell software testing means verification of system or application under test

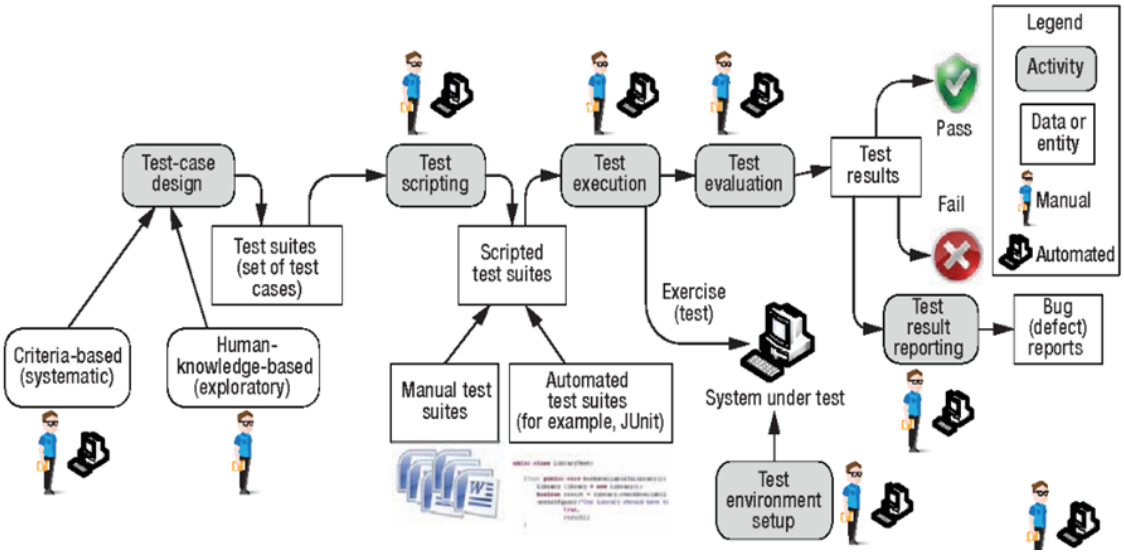

Figure 1: Manual testing & automated testing (www.softwaretestinghelp.com)

2.2.2 Manual Software testing

Manual testing is defined as a procedure to test the product physically to discover the bugs [3]. Taipale et al. agrees by stating manual software testing is the procedure of physically testing software for imperfections, and it requires a tester to assume the job of an end-user whereby they utilise the greater part of the application's features to ensure correct functionality [31].

2.2.3 Automated Software testing

Automated software testing is defined as a process where software testing tools like Selenium are utilised to conducts pre-scripted tests on software, to confirm whether all the usefulness are working appropriately [34]. Another report submits that Selenium WebDriver is a group of open source Application Programmable Interfaces (APIs) which are utilised to automate and execute the testing of a web application and it supports various browsers like Google chrome, Internet Explorer (IE), Safari and Firefox which is an added advantage when testing across multiple browsers [23].

2.3 Why Using Automated Software testing

Whilst software testing usefully identifies errors and hence reduces associated costs, evidence suggests that its proportion to the accumulated costs of total development is high. A research study has determined that it contributes between 40% and 80% of the total development costs [10]. This could be regarded as contrary to business strategy for profit maximisation, and hence software manufacturers are increasingly looking for ways to reduce their development costs. In one recent report, process efficiency is described as the ability of a process to produce desired outcome with the optimum number of resources [1]. Whilst it is

commonly agreed that automated testing helps to identify software faults quicker when compared to manual testing, literature questions whether it is able to significantly reduces the overall costs of a project [21]. The automated software testing is defined as a process where automated software testing frameworks (like the Selenium web-testing suite [24]) are utilised to conducts pre-scripted tests on software, to confirm whether all the usefulness are working appropriately [34] and it supports various browsers like Google Chrome, Internet Explorer (IE), Safari and Firefox which is an added advantage when testing across multiple browsers.

2.4 Need for Automated Software Testing

There is a strong evidence of reduced expenditure by using test automation. A report by Infosys [1] states that the manual testing of product features and performance is expensive, lengthy and tedious task. A recent survey [37] claims that cost of software testing is between 30% and 50% of the entire budget, and there is an undeniable requirement for testing methods that can decrease the duration required to guarantee software quality and reliability. In other work, the author discovered a set of factors that influences the cost of test automation, which all provide positive outcomes on cost, quality and release time to market [21]. Another research study presented an experiment on an automated test generation tools and proposed a methodology named 'TestDescriber', which creates comprehensive documentation for each individual test, thereby improving and aiding the reduction of expert knowledge required to perform the tests [27]. This is an extension of previous work [34] that developed a toolkit to facilitate the automatic generation of test data for structural testing cases.

In relation to financial impacts of automated testing, a study discovered that the cost increases from 1:5 (from requirements to after release) for simple systems to as high as 1:100 for complex systems [14]. This statement confirms that once the bug is found in production, it will cost more to rectify as the system might need to be taken out of operation in order for the bug to be fixed, which will result in the company losing revenue or even customers migrating to competitors because of lack of confidence with their software systems. A similar study confirms that the longer a software fault is left undetected, the more expensive it will be to fix once discovered [16].

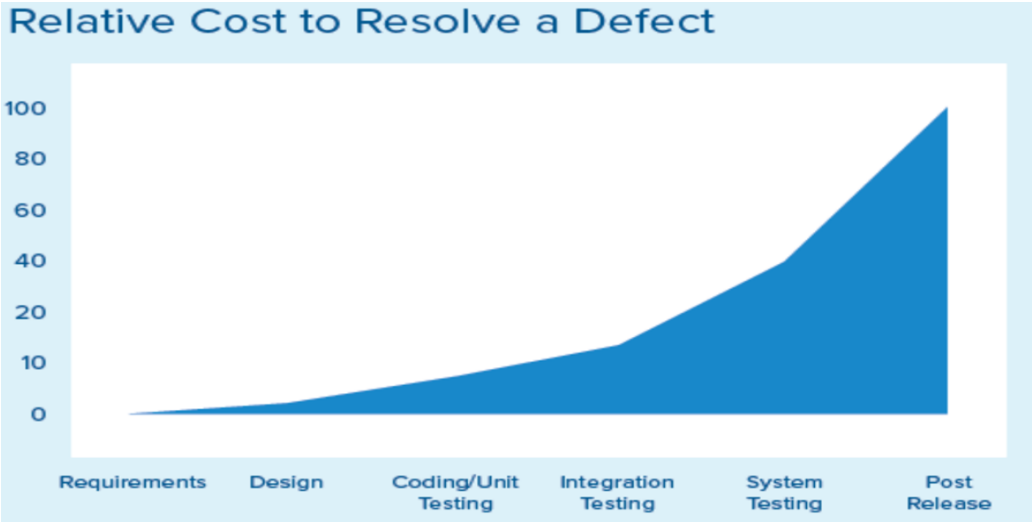

Figure 2: Stage by Stage Relative cost to Fix Defects (https://ecs.co.uk)

A recent research work examined the relative proficiencies of both random and organised methods to automated software testing and identified that proficiency is an imperative property of software testing; conceivably significantly more essential than adequacy [5]. The test automation can provide benefit in many ways, such as test reusability, repeatability, test inclusion and exertion spared in test executions. Another work added that since complex software faults exist even in basic software systems, engineers are searching for automated systems to identify software faults, resulting in an increased trust and accuracy [29]. A similar study states that automated testing is a productive method to gain trust in the software's accuracy [15]. This observation is well argued and is based on the premise that automated software testing removes the element of human error and is faster to run regression tests, which can take days if they are to be performed manually. Furthermore, another paper claimed that when comparing automated software testing versus manual software testing, the impacts and advantages of automated testing are provided in long-term when compared to manual testing [30]. This is due to the fact that an automated testing tool can consider and process all factors holistically, in an efficient manner, as compared to manual testing.

A research study examined different methods of software testing and concluded that performing manual testing is wasteful and error prone; using automated tests is efficient in reducing the release time of software [4]. The experiment was based on a mathematical procedure with the intention of increasing the chances of having a resource-effective test automation process. Another paper investigated the techniques of enhancing the effectiveness of software test automation. This point is supported by stating that automated testing liberates testing staff to accomplish other testing duties [17]. This paper also explored

the challenges and the best practices related to quality within software development and determined that completing software testing can reduce financial expenditure through by catching issues before they make it far through the product development process.

Another paper reports that the primary issue of a tester and or organisation that desire to automate their software testing process is how much the testing tools cost [8]. Furthermore, the concern is whether it will satisfy the testing requirements. Open-source testing tools are available as well and free to use, which is seen as a positive aspect and does help organisations to automate software testing. Another research study conducted an experiment to investigate the benefits of automated testing techniques by using the open-source Ball Aviation Universe testing framework. It concluded that automated testing yields numerous advantages, such as mitigation against client input errors, quicker execution times, and decreased client oversight amid execution [23].

2.5 Agile Software Testing

There is strong evidence that agile testing fits well with automated software testing. Agile testing methodology is flexible, and it combines testing with the completion of each phase of software development to identify software problems before progressing to the next phase. There is a strong cooperation between the tests author and the developers to ensure test scripts can be swiftly generated and are robust to warrants elasticity and flexibility. If Agile is to succeed, testing must be a central pillar of the development process.

2.6 Why Automated Software Testing Is Not Fully Utilised

A study states that during the research into the current situation and potential improvements in software test automation, it was observed that the principle advantages of test automation were quality improvement, the likelihood to execute more tests in less time, and familiar reuse of testware [19]. However, another work identified that when investigating the present condition of test automation in software testing companies by concentrating on the perspectives and perceptions of supervisors, testers and developers in every company, it was concluded that the biggest burdens were the expenses related with implementing test automation, particularly in unique altered conditions [32]. Another paper performed an experiment using the AutoTest tool, which is a fully automated testing framework running on the Linux system. After combining automated and manual testing, it was realised that software can be tested either physically or automatically, and these two methodologies are able to complement each other [22].

Similarly, another experimental framework to compare testing procedures based on efficiency, effectiveness and applicability [10]. It employed 70 distinct test design techniques and

concluded that automated testing cannot be applied in every case due to lack of ability to determine issues and/or increased difficulty in the implementation. This agrees with the observations and lessons learned from automated testing [28] that the utilisation of an automated test tools do not improve fault detection compared to manual testing. Moreover, it was discovered that 80% of professionals disagreed that the automation testing would serve as a complete replacement to manual testing. This issue seems to be well known as another paper determined that automated tests found only 26% (on-average) of the faults [20]. They further state that when an automated test suite has been configured and integrated, it is usually reused in future tests. This makes the testing substantially less likely to uncover defects in the product during the next iteration. Regarding the open-source software testing tools, a paper investigated a number of such tools and concluded that they are not regularly maintained and are difficult to use [25]. Also, organisations are still likely to use commercial tools due to the level of support available, which can help them fully utilise the technology. Hence, due to the aforementioned reasons, automated software testing is not used in some organisations.

2.7 Summary

Existing literature highlights that there is a known gap between academic and practitioner opinions on automated software testing, and there is a need to close the gap by exploring attitudes concerning the benefits and restrictions of test automation. However, the appreciation for test automation is unbalanced as the achievement rate is low and the impediments are always high at the beginning for acquiring the resources to setup automation testing and training tools. Moreover, the automated tests are not well-suited for every organisation, and are varied in terms of accuracy, applicability and usefulness factors.

In terms of reliability, manual testing is not as precise because of the likelihood of human errors. Nevertheless, automated testing is a consistent method, as it is done by tools and scripts. Also, there is less or no testing tiredness especially when doing regression testing, which is the re-running of experiments in request to guarantee that changes made to the product do not present new blunders and to ensure that the framework's functionalities have not been influenced by those changes [8]. Dustin, et al added that to execute regression testing in manual mode it includes more time and cash [9].

Automated testing does not include human perception. So, it can never give confirmation of ease of use and positive client experience. However, manual testing process permits human

perception, which might be valuable to offer easy to use system. The author also agrees that manual software cannot be replaced completely by automated software testing, because automated test cannot discover defects than what an accomplished tester can manually do. Usually the manual testing cases are the one which are used or converted to automated scripts.

There is a connection between cost and efficiency which is very critical and significant to organisations. Cost in production, is the value of money that has been utilised up to yield something, and later is not available for use anymore (Wikipedia). However, efficiency is the scope to which time, effort, or cost is well-used for the planned task or purpose. It often includes specifically the ability of a specific application of determination to produce a specific result effectively with a least amount or quantity of waste, expense, or unnecessary effort (Wikipedia).

Chapter 3

3.0 Methodology

3.1 Introduction

This chapter seeks to present quantitative research to evaluate and analyse with a view to achieve the aim and objectives set for this research. This chapter will highlight the common reasons as to why or why not testing automation is being used to improve software testing in the software development cycle.

3.2 Quantitative and Qualitative Data Analysis

Quantitative research is ideally overseen through surveys that have close-finished inquiries. While building a questionnaire, it is essential that the questions utilise a phrasing that the respondents are used to.

Vaismoradi el at, submitted that Qualitative research is basically exploratory research. It is utilised to pick up a comprehension of essential reasons, feelings, and inspirations [33]. It gives bits of knowledge into the issue or creates ideas or speculations for potential quantitative research. Another paper added that Qualitative research is likewise used to reveal trends in thought and opinions and dive further into the issue. Qualitative data gathering strategies differ utilising unstructured or semi-organized systems [33]. Some regular techniques incorporate focus groups, singular meetings, and observations. The sample data size is commonly small, and respondents are chosen to satisfy a given portion.

3.3 Research Methodology

The author of this thesis used a two stage analyses to address the research question as illustrated by the following diagram.

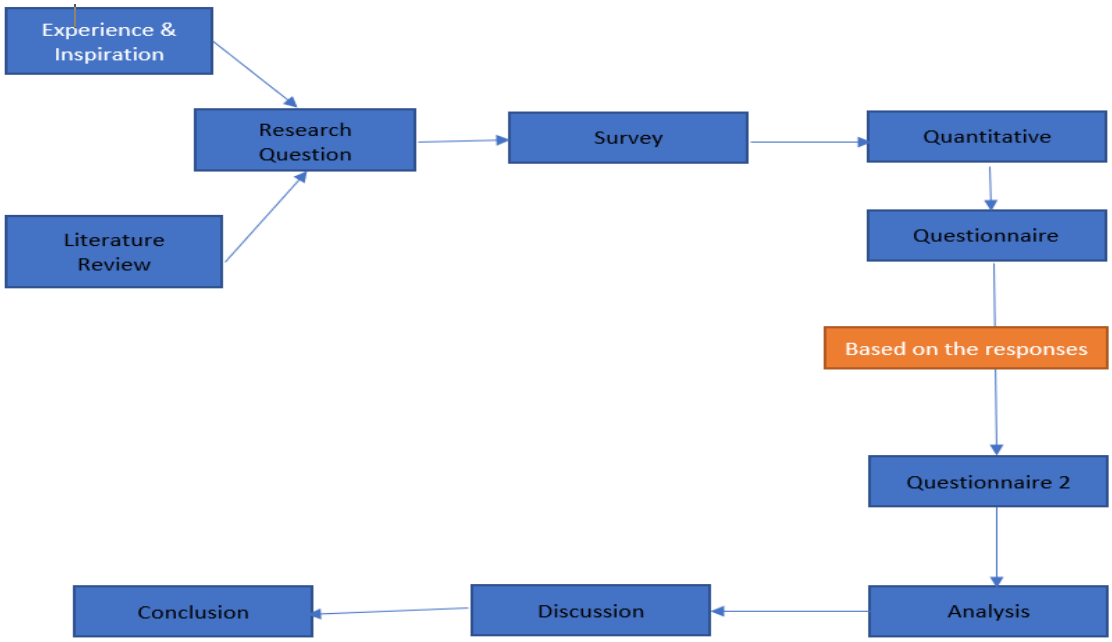

Figure 3: Research methodology

In the first phase, a widespread questionnaire was designed, however essential analysis was performed to provide the groundwork for understanding how individuals' thoughts towards the reasons as to why automated testing is not used, and also to enlighten patterns specific to individuals performing different roles and of different experiences. The generated link from Google forms was posted on LinkedIn. LinkedIn was used by the author to target fellow professionals in the software testing industry, which includes groups such as: Quality Assurance (QA) testers, Software Developers in Test (SDIT), software testing managers and Automation Engineers.

In the second phase another questionnaire was designed again using Google forms and circulated via LinkedIn. At this stage, the author conducts a principal component factor analysis (principal components extraction). This is a standardised and widely used approach, which provides the opportunity to further examine the relationships between participant opinions on automated testing as a whole, while looking for clustering of certain variables [2]. In particular, the author examined the dimensionality of individuals' responses gathered from phase one to examine whether or not automated testing attitudes comprise a distinct attitudinal dimension.

3.4 Questionnaire

A questionnaire is a set of questions that can be utilised to do overviews or meetings. The questions can be created dependent on various investigations and reports. It ought to contain various types of questions to think about the more extensive part of any subject and to wipe out the monotonous pattern. There likewise ought to be some statistic addresses where clients can pass on some data about themselves. It should have a period limit and not be excessively extensive, with the goal that the client would not feel exhausted and can focus while replying. It tends to be conveyed as written word and can be distributed via internet means to gain a wide range of responses. It is always better to get more response from the viewers so that analysis of that questionnaire can be done thoroughly with more data. Questions in a survey can be open finished or close finished or the blend of both. It is anyway clear that often, close - finished inquiries are more replied by the respondents.

3.5 Questionnaire Design

To measure attitudes toward automated testing, a scale was constructed based on twenty items asking respondents about their broad feelings about automated software functionality as well as about the adoption. The questionnaire was created in a way that develops a comprehensive analysis as to the common reasons as to why automated software testing is not being used. To understand this, and what facilitates the development of technological mechanism, practitioner's attitudes and concerns was investigated first. The questionnaire is mostly derived from the help of existing frameworks and methodologies. The survey was circulated through professional and social media channels to acquire participants. Groups such as the following was targeted: Quality Assurance (QA) testers, Software Developers in Test (SDIT), software testing managers and Automation Engineers. A total of 22 questions were in the designed questionnaire.

The questions were classified into themes. The author selected these themes as they repeatedly were presented in related research and they represent a natural divide of the individual, the technology, and the environment within which both operate.

Theme	Question Numbers
biographic	Q1, Q2
Time	Q4, Q15
Cost	Q11, Q19, Q20
Tools and Techniques	Q6, Q16, Q17, Q18, Q21
Utilisation	Q5, Q7, Q8, Q12, Q22
Organisation and Capability	Q3, Q9, Q10, Q13, Q14

Table 1: Respondent experience in the IT sector

3.5.1 Biographic

In section one asked demographic questions, to find more details about the respondents. As an example, the author asked the following question: What is your job title? - The motive for asking this question was to validate that the questionnaire was hitting the right audience and in this case the author is targeting those which are in software testing roles. When the data is collected it will be easy to clean any anomalies based on the job roles, and those deemed irrelevant will be removed. This will result in reliable data being collected and analysed in the next stage.

How many years' experience in the IT sector do you have? – This was another question which was asked under demographic section. The reason for asking this question was to try and gather data with the view that those with high experience for example 15 years will still support the old way of testing and those with 5 years' experience will embrace new automated software testing technologies than manual testing.

3.5.2 Time

One of the critical elements in software development is time. In the questionnaire the author asked questions to do with time, and the motive was to measure the impact of this element. As an example, the following question: They are time-consuming to learn was asked in phase one questionnaire, under the section titled challenges /problems faced with the existing test automation tools in your projects / organisations. If high percentage of respondents choose strongly to agree or agree, this will indicate that automated software testing tools takes time to learn and this will have an impact on resource allocation in an organisation which might results in deadlines being not met, because staff will spend time learning the tools. Otherwise the opposite is true.

3.5.3 Cost

The author also asked questions to do with costs. As an example, the following question was asked: Commercial tools are too expensive. The motive for asking this question is to try and identify how much organisations are willing to invest. If high percentage of respondents chooses strongly to agree or agree, the results will mean that the implementation of automated software testing might be expensive. And if the respondents choose the opposite then it will imply that the implementation is not expensive

3.5.4 Quality

Some of the questions asked by the author were around quality. The motive for asking this question is to identify the skills needed for automated software testing. If one does not have the right skills, then quality will down. Below is one example of question asked: They require strong programming skills. If high percentage of the respondents choose to agree then the results will mean in order for a quality framework to be put in place the expert would need strong programming skills.

3.5.5 Tools and techniques

Tools and techniques questions were also presented to the respondents by the author. The motive for asking this question is to try and identify if there are right tools and techniques to perform automated software testing in organisations. If high percentage of respondents chooses strongly to agree or agree, the results will mean that there are right tools and techniques in organisations. And if the respondents choose the opposite then it will imply that there are no correct tools and techniques.

3.5.6 Utilisation

The author when designing the questionnaires asked some questions to do with automated software utilisation in organisations. The motive for asking this question was to try and identify if automated software tools are utilised in organisations. If high percentage of respondents chooses strongly to agree or agree, the results will mean that automated tools are being fully utilised in organisations. And if the respondents choose the opposite then it will imply that tools are not being utilised.

3.5.7 Organisation and capability

The author when designing the questionnaires asked some questions to do with automated software utilisation in organisations. The motive for asking this question was to try and identify how much organisations are willing to invest. If high percentage of respondents chooses strongly to agree or agree, the results will mean that the implementation of automated software

testing might be expensive. And if the respondents choose the opposite then it will imply that the implementation is not expensive

The author chose to use quantitative data analyses because though it does not always shed light on the full density of human knowledge or perception, quantitative data analyses provides a larger sample sizes which make the assumptions from quantitative research generalisable as compared to the qualitative method.

3.6 Questions Asked

The questions asked to the participant in this study are:

1. What is your job title?
2. How many years of experience in the IT sector do you have?
3. Lack of skilled resources prevents automated testing from being used.
4. Individuals not having enough time prevents the use of test automation.
5. Difficulties in preparing test data and environments prevents their use.
6. Not have the right automation tools and frameworks is preventing use.
7. Difficult to integrate different automation tools/frameworks together is pre- venting their use.
8. Requirements changing too often are preventing their use.
9. Not realising and understanding the benefits of test automation is preventing their use.
10. Lack of support from senior management is preventing their use
11. Commercial tools are too expensive, which prevents their use.
12. Open-source tools are hard to use.
13. Test automation tools require a high-level of expertise, which is often not available.
14. Automated testing requires strong programming skills.
15. Automated testing techniques are time-consuming to learn.
16. Automated testing tools and techniques lack necessary functionality.
17. They are not reliable enough to make them a suitable for use.
18. They lack support for testing non-functional requirements (usability, safety, security, etc.).

19. Expensive to generate test cases/test scripts.
20. They require high maintenance costs for test cases, test scripts and test data.
21. Automated testing tools and techniques change too often, introducing problems that need fixing.
22. Difficult to reuse test scripts and data across stages of testing.

Table [1] provides the mapping of the questions to each theme. As identifiable from cross-referencing the question with the theme allocation presented in Table [1], it is noticeable that multiple questions are asked in each theme. The purpose of this was to extract more information from the participants on their automation to enable stronger analysis. It is also to note that all items are negatively worded. There is also an open-ended section for the participant to provide further comments. Note that the questions are not asked in a grouped order to try to introduce variation within the questions been asked, making the participant revisit the theme after changing to a different theme. For each question, the participant was presented with a statement which they can either agree or disagree. The participant was provided with responses based on the Likert scale [6] which are either strongly disagree, disagree, neutral, agree, and strongly agree. Furthermore, free text input is made possible at two points in the survey to acquire any additional information. The purpose of these inputs is to acquire comments from the participant that might rationalise their answer or provide further information. The first at approximately halfway through the survey at question 10 and the second at the end of the survey at question 22.

As all questions were multiple choice, the responses provided in Section 6 are their numerical versions (strongly agree = 5, agree = 4, neutral = 3, disagree = 2, strongly disagree = 1). Furthermore, the graphs provided in Figure 3 and Figure 4 use a character abbreviation (strongly agree = SA, agree = A, neutral = N, disagree = D, strongly disagree = SD). Because all items are negatively worded, so score reverse were not needed.

The survey was created as a digital survey and distributed through special interest groups. More specifically, we created used Google Forms to create and host the survey and software engineering and testing special interest groups on LinkedIn.

3.7 Limitations

One of the limitations faced by the author is the fact that the essentially restricted intricacy and length of questionnaires keep them from being utilised to clarify activity (since this expects us to comprehend individuals' expectations), the hugeness of activity, and the associations between acts.

The author of this report is also of the view that the quantity of details that are regularly gathered while performing subjective research are frequently overwhelming. Dealing with that information to pull out the key focuses can be a time-consuming effort. Another limitation is that the nature of the information that is gathered through quantitative research is profoundly reliant on the abilities and perception of the professionals. If the professional has a one-sided perspective, at that point their viewpoint will be incorporated with the information gathered and impact the result.

3.8 Research Ethics

According to Israel and Hay (2006), there are many reports encouraging that, research must be directed morally. Miller et al (2008) also submit that the accentuation is that, research about ethics involves following great practice and keep up of honesty of research. In this research an undertaking is classed as less hazard basically because it does not really include direct contact with members, for example those who participated in the survey.

The author of this report also wishes to submit that he did not copy any work during literature review. Where others work was used in this report the author gave credit to the author and referenced it.

Regarding data collection, the author ensured that no identifiable information was captured from the users, ensuring that a dataset cannot be tracked back to a specific individual. However, as the respondents are providing information on their job role and potentially employer, all necessary steps are ensured to keep the data safe. These are:

1. The data was stored on European Google servers and password only accessible by members of the project team.
2. The data was analysed using Google Drive cloud-based software (Spreadsheet etc.), ensuring that the data did not need to be copied or downloaded for analysis, thus minimising that chance of someone gaining an unauthorised copy.

Chapter 4

4.0 Results and Analysis

4.1 Introduction

This segment presents review survey questions and analysis is made to the outcomes. In total the author received 84 respondents in phase 1 and in phase 2 the survey managed to get 25 respondents worldwide. The web study was live from 9th of January to 15th January 2019 for phase 1 and from 25th of March to the 29th of March 2019. After analysis of the 84 reactions in phase 1, the author finished with 82. Again, responses in phase 2 were analysed and all 25 reactions were deemed legitimate. After adding responses from phase 1 and phase 2, 107 was the total number of responses collected.

4.2 Participants

Experience (years)	Volume
0.6	1
1.5	1
1.9	1
2	3
3	5
4	2
5	12
6	6
7	3
8	4
8.5	1
9	3
10	9
11	2
12	5
13	3
15	5
16	1
17	2
18	1
20	5
21	1
25	3
30	1
31	1
33	1

Table 1: Respondent experience in the IT sector

Figure [1] illustrates the experience of the participants. Years of experience is used to measure how long a participant has been involved with automated testing, which is a measure also utilised in other academic work [12]. It is important to make the distinction that the authors are not assuming that years of experience relates to an individual's skill level, more that an assumption is made that they will have had more interaction with automated testing tools and techniques, therefore forming more strong attitudes. Experience duration range from 0.5 to 33 years, and as evident in the table, a good variation was surveyed, but the majority of participants are in the ranges between 1 and 20 years. This is of significant importance as it demonstrates that the survey will not be overly biased to IT professionals with either short or long experience duration. Table [2] illustrates the variation of roles and the number of respondents. Note that the role title was entered by the user and resulted in a wide variation of roles. It is worth noting that the roles have been placed in themes for ease of comparison. The themes adopted are the same as those in the State of Testing Report (2018) as discussed in Section [1]. In the table it is evident that the majority of job role themes are in Quality Assurance, Software Testing, and then senior versions of each role. Outside of technical roles, there are 3 Chief Executive Officers, 4 consultants, 7 managers, and 1 student. Although it can be seen that in general the majority of respondents are undertaking more technical roles, the 15 non-technical responses account for 18% of the total reposes and is not insignificant.

Job role	# participants
CEO	3
Consultant	3
Senior Consultant	1
Manager	7
Student	1
QA	7
Senior QA	8
Tester/Engineer/Analyst/Architect	22
Senior Tester/Engineer/Analyst/Architect	15
Test Automation	5
Senior Automation	9
Total	81

Table 2: Participant roles in the IT sector

4.3 Results: Stage 1

In this section, the responses from the questionnaire are analysed in detail. Figure 2 provide the numbers of responses for each available response (strongly disagree, disagree, neutral, agree, strongly agree) and Table 3 present the actual numbers. Figure 3 provides bar charts for each response in relation to the response choices, whilst also showing the response split between different job roles, as provided in Table 2. Furthermore, Figure 4 provides information on how many years of experience the participants have against the responses.

Question #	SD	D	N	A	SA	SD%	D%	N%	A%	SA%
Q3	2	16	17	31	15	2	20	21	38	19
Q4	10	21	15	25	10	12	26	19	31	12
Q5	3	18	13	25	22	4	22	16	31	27
Q6	12	34	15	17	3	15	42	19	21	4
Q7	9	24	17	25	6	11	30	21	31	7
Q8	2	21	21	24	13	2	26	26	30	16
Q9	15	25	12	18	11	19	31	15	22	14
Q10	10	20	18	22	11	12	25	22	27	14
Q11	7	8	14	35	17	9	10	17	43	21
Q12	19	32	19	10	1	23	40	23	12	1
Q13	11	27	25	17	1	14	33	31	21	1
Q14	3	13	16	40	9	4	16	20	49	11
Q15	10	20	25	26	0	12	25	31	32	0
Q16	15	31	25	9	1	19	38	31	11	1
Q17	18	35	19	8	1	22	43	23	10	1
Q18	8	17	24	26	6	10	21	30	32	7
Q19	7	24	22	26	2	9	30	27	32	2
Q20	3	23	15	34	6	4	28	19	42	7
Q21	2	12	25	38	4	2	15	31	47	5
Q22	6	32	15	26	2	7	40	19	32	2

Table 2: Response to questions. SD = strongly disagree, D = disagree, n = neutral, A = agree, and SA = Strongly Agree. Percentages are also provided to show the distribution amongst the total responses.

Question 3 asked the participants whether they agreed with the statement that the lack of skilled resources is preventing automated testing from being adopted within an organisation. Overall, 57% of the responses agree (38% agree and 19% strongly agree) with the statement, 21% are neutral, and 22% disagree (20% disagree and 2% strongly disagree). This demonstrates that the majority of the responses agree that a lack of skilled resource is a problem. Furthermore, as demonstrated in Figure 3a, the majority of people agreeing with this statement are performing non-technical roles, whereas the majority of the people disagreeing are performing more technical roles. This is of significance as it highlights the different viewpoints when considering whether there is a resourcing issue. In addition, Figure [4a] highlights that the majority of responses provided by participants with in-excess of 20 years' experience are either agree or strongly agree. However, it is also worth noting that participants that strongly disagree are only in the 15-20 years of experience category.

Question 4 asks the participants if they believe individual's not having enough time prevents the use of automated testing. The response to the question is well balanced with only a small majority stating that they agree. More specifically, 43% of the participants state that they agree (31% agree and 12% strongly agree), 19% remain neutral, and 38% state that they disagree (26% disagree and 12% strongly disagree). As demonstrated in Figure [3b], the distribution of job roles amongst the responses are balanced, with both technical and non-technical roles agreeing and disagreeing. It is evident that CEOs are either neutral or agreeing, but as only 3 participants identified as being CEOs, then it can be stated that there are insufficient to be statistically relevant. One identified trend is that participants are strongly disagreeing have identified as performing technical roles and those, with only 2 of the 10 responses being of a senior role. This indicates that more junior roles more strongly disagree with the presented statement. It can also be established from Figure [4b] that there is an even distribution of years of experience amongst the answers.

Question 5 asks the participants whether they believe that difficulties in preparing test data and environments is responsible for preventing the use of automated testing. Overall, the majority of the responses agree with this statement. More specifically, 58% agree (31% agree and 16% agree), 16% are neutral, and 26% disagree (22% disagree and 4% agree). Figure 3c presents the breakdown of responses versus job role. Interestingly, the results demonstrate that non-technical roles (CEO, Consultant, Management) are mostly agreeing with this statement and there is only 1 response from a Manager that disagrees. Furthermore, if considering responses from technical roles alone, they are mostly balanced with a slight emphasis on disagreement with the statement. Figure [4c] also demonstrates that there is an even distribution of years of experience amongst the answers.

Question 6 asks the participants whether they believe not having the right automation tools and frameworks are preventing use. The majority theme here is that 57% disagree (42% disagree and 15% strongly disagree), 19% are neutral, and 25% agree (21% agree and 4% strongly agree). Figure [3d] illustrates the responses in relation to job role, and it is evident that there are submissions from each role in each response type, apart from CEO who are only agreeing with the statement. It is evident that overall the majority of participants do not believe the use of automated testing is prohibited by the inability to identify and use the correct tools. Interestingly, Figure [4d] illustrates that participants that are strongly agreeing have 10 to 15 and 30 to 35 years of experience. However, overall there is an even distribution of years of experiences amongst the responses.

Question 7 asked the participant whether they agreed with the statement that difficulties in integrating different tools/frameworks together is preventing their use. A small majority were in favour of disagreeing with the statement. More specifically 41% disagree (30% disagree and 11% strongly disagree), 21% are neutral, and 38% agree (31% strongly agree and 7% agree). Figure [3e] illustrates that the majority of non-technical roles agree with this statement and the balance based on technical roles is almost even, with a slight emphasis in disagreeing with the statement. Figure [4e] demonstrates an even distribution of years of experience amongst the responses, although the responses with a greater number of years of experience are on balance more in agreement than disagreement.

Question 8 asks the participant as to whether they agree or not with the statement that frequent requirement change is often preventing the use of automated testing. The provided responses are overall in agreement with the statement. More specifically, 46% agree (30% agree and 16% strongly agree), 26% are neutral, and 28% are disagree (26% disagree and 2% strongly dis- agree). In Figure [3f] it is evident that the job role distribution is mostly even with the majority of respondents operating in Software Tester, Engineering, Analysts and Test Architects, whereas respondents with Quality Assurance roles are majority agreeing. Interestingly non-technical positions, such as CEOs, are either neutral or disagreeing with this statement, which could indicate a misalignment between both non-technical and technical employee experiences with automated testing when it comes to the impact on changing software requirements. Figure [4f] demonstrates an even distribution of years of experience amongst the responses.

Question 9 asked whether people believed that automated testing is often not used due to people not realising and understanding the potential benefits. The overall trend is that a majority disagree with this statement. More specifically, 49% of the participants disagree (31% disagree and 19% strongly disagree) with this statement, 15% are neutral and 36% agree (22% agree and 14% strongly agree) with the statement. In Figure [3g], it is evident that the distribution of roles is evenly spread as is the distribution of years of experience amongst the responses, as demonstrated in Figure [4g].

Question 10 asked the participant as to whether they believe a lack of support from senior management is preventing their use. The results from this question are well-balanced with the number of participants agreeing with the statement being slightly higher than those disagreeing. More specifically, 41% agree (27% agree and 14% strongly agree), 22% are neutral, and 37% disagree (25% disagree and 12% strongly disagree). In Figure [3h], it is evident that there is an even distribution of job roles amongst the responses; however, the role of consultant only appears in the neutral, agree, and strongly agree responses, whereas managers and CEOs are on average disagreeing with the statement. The difference with consultants could be due to the fact that they are not directly employed by an organisation and provide independent observation. Figure [4h], illustrates the age range of the participants, which is on average are evenly distributed, with a slight emphasis on participants with greater experience providing a neutral, agree, or strongly agree response. In literature review Rafi (2012) highlighted that advantages of test automation were identified with test re-usability, repeatability, test inclusion and exertion spared in test executions, however the impediments were high at the beginning for resources to setup automation testing tool and training. Maybe the results are so tight because the management might have chosen to be neutral or they chose to disagree because of the high costs of setting up the tool and training.

Question 11 asked whether the participant agreed or disagreed that commercial tools are too expensive, thus preventing their use. The majority of the participants agreed with this statement. In total, 64% agreed (43% agree and 21% strongly agree), 17% neutral, and 19% disagree (10% disagree and 9% strongly disagree). Figure [3i] illustrates the number of responses in relation to each job role. It is observed that the majority of management, senior consultants, and QA and senior QA roles agree with the statement, whereas CEOs are neutral or disagree. Other roles are well represented across all response options and therefore it is not possible to identify a common pattern. A similar conclusion is deduced from Figure [4i] where the years of experience is evenly distributed amongst the available responses. One of participants mentioned in the comment section that a tool is only expensive when not used, or used in a wrong way, or used but with no benefits in return.

Question 12 asked the participant whether they think that open-source automation tools are difficult to use. The majority of participants disagree. In total, 63% disagree (40% disagree and 23% strongly disagree), 23% are neutral, and 14% agree (12% agree and 1% strongly agree). As illustrated in Figure [3j], the number of non-senior and technical roles is low for both agree and strongly agree. In relation to years of experience, Figure [4j] illustrates that a higher number of individuals with a lower number of years of experience disagree with the statement, which could be down to the fact that those with fewer years of experience received dedicated

training on the tools that they are using, i.e., they might be recent graduated having been training specifically on the used technology.

Question 13 asked the participant whether they agree or disagree with the statement that test automation requires a high-level of expertise, which is of- ten not available. Overall the trend is that the respondents disagree with a majority. More specifically, 47% disagree (33% disagree and 14% strongly dis- agree), 31% are neutral, and 22% agree (21% agree and 1% strongly agree). Figure [3k] illustrates how different roles selected their answers, It is evident that there is an even distribution of roles; however, only one technical employee strongly agrees, and only those strongly disagreeing and made up on solely technical roles. It is therefore a fair assumption to state that only those with a good technical understanding disagree with the statement. Figure [4k] illustrates that the number of years of experience within each reply category is well distributed; however, strongly disagree has the highest average years of experience when compared to the other categorise.

Question 14 asked whether the participants believe that automated testing requires strong programming skills. The responses overall strongly agreed with this statement. More specifically, 60% agreed (49% agree and 11% strongly agree), 20% are neutral, and 20% strongly disagree (15% disagree and 4% strongly disagree). Figure [3l] demonstrates that there is an even distribution of roles providing responses within each response category, and Figure [4l] illustrates that there is an even distribution of years of experience within each response category.

Question 15 asked whether the participant believe that automated testing techniques are time-consuming to learn. The responses to this question are quite evenly distributed, with 37% disagreeing (25% disagree and 12% agree), 31% neutral, and 32% agreeing (32% agree and 0% strongly agree). The low percentage of participants strongly agreeing with this statement results in an average of between neutral and disagreement. Figure [3m] demonstrates that the distribution of job roles amongst response categories. However, it is worth noting that in general non-technical roles are responding more closely with agree and neutral replies, which could perhaps be down to their lack of experience with the technology. Figure [4m] demonstrates the years of experience for each response category. For example, all responses from CEOs are in the agree category. Interestingly, there is an even distribution apart from agree whereby there is the highest quantity of participants with the lowest number years of experience. This could demonstrate that those with a lower amount of experience

could believe that automated testing takes more time to learn, which would most likely originate from the fact that they will have more to learn during earlier years of employment.

Question 16 asked participants whether they believed that automated testing tools and techniques lack necessary functionality. Overall, the majority of participants disagree with this statement. In total, 57% disagree (38% disagree, 19% agree), 31% are neutral, and 12% agree (11% agree and 1% strongly agree). Figure [3n] demonstrates an even distribution of job roles amongst the response categories with no identifiable pattern. Figure [4n] demonstrates that there is a slight increase in the portion of responses from participants with an increased number of years of experience in the agree and strongly agree category. This could perhaps indicate that more experienced employees have a stronger belief that current techniques and tools lack functionality, which could be down to the fact that they have in-depth experience and knowledge of missing functionality.

Question 17 asked the participants whether they believe that automated testing tools and techniques are not reliable enough, making them unsuitable for use. The responses overwhelmingly disagreed with this statement. More specifically, 65% disagree (43% disagree and 22% strongly disagree), 23 are neutral, and 11 agree (10% agree and 1% strongly agree). Figure [3o] illustrates that there is a diverse distribution of job role amongst each response category. It is worth noting that the only one participant strongly agreed, and they are performing a technical role, which could indicate that their dissatisfaction originates form working closely with automated testing tools and techniques. Furthermore, as evident in Figure [4o], there is no relationship between years of experience and response, apart from the observation that there is a higher proportion of participants with a lower number of years of experience either agreeing or strongly agreeing. This could indicate that they have not yet mastered their craft and utilise the full potential of automated tools, or even their dissatisfaction with their chosen career.

Question 18 asked the participants whether they agree that automated testing lacks support for testing non-functional requirements (usability, safety, security.) The responses to this statement are close, but the majority is in agreement with this statement. More specifically, 40% agree (32% agree and 7% strongly agree), 30% and neutral, and 31% disagree (21% disagree and 10% strongly agree). Figure [3p] and Figure [4p] presents that there is an even distribution amongst roles and years' experience within the response categories.

Question 19 asked the participant whether they believe automated test scripts and cases are expensive to generate. The responses to this question are balanced with only a slight emphasis on disagreement. In total, 38% disagree (30% disagree and 8% strongly disagree), 27% and neutral and 35% agree (32% agree and 3% strongly agree). Figure [3q] illustrates that the general trend is that it is more likely for managerial roles to agree with this statement. It is also worth noting from Figure [4q] that the small number of responses that both strongly agree and disagree have greater than 5 years of experience, whereas the other categorise have an even distribution. This could indicate that the views of experienced employees are on average neutral, with a minority having polarised views.

Question 20 asked the participants whether they agree that automated testing requires high maintenance costs. Overall, the participants agreed with this statement, with 49% agreeing (42% agree and 7% strongly agree), 19% submitting neutral, and 32% disagreeing (28% disagree and 4% strongly disagree). Figure [3r] illustrates that in general non-technical roles are more likely to agree with this statement, which is perhaps to be expected considering their daily interaction with financial operations. There is a slight emphasis on technical staff to not agree with this statement, which is perhaps down to their lack of involvement with the financial side of their employers' activities. Furthermore, Figure [4r] identifies that those strongly agreeing or disagreeing have a higher number of years of experience.

Question 21 asked the participants whether or not they agree that auto- mated testing tools and techniques change too often, introducing problems that need fixing. A majority of the responses agree with this statement. More specifically, 52% agree (47% agree and 5% strongly agree), 31 are neutral, and 17% disagree (15% disagree and 2% strongly disagree). Figure [3s] illustrates that non-technical employees are more likely to agree with the statement, with only management roles submitting as strong accept. Interestingly, it also illustrates that QA and senior QA roles only responded as agree. The majority of technical testing, engineering, and automation roles are either neutral or dis- agreeing, and they are also the only roles to strongly disagree. Furthermore, Figure [4s] also displays that in general the participants with a higher number of years of experience are more likely to respond with a neutral or disagreeing response. This indicates a different point-of-view between non-technical and technical roles, as well as number of years of experience that an individual has. Experienced individuals may have gained sufficient expertise in how to maintain their scripts and keep them updated with new versions of testing tools.

Question 22 is the final question and asked the participants whether they agree that it is difficult to reuse test scripts and data across different stages of testing. The responses are in general aligning with disagreeing with this statement. More specifically, 47% disagree (40% disagree and 7% strongly disagree), 19% are neutral, and 35% agree (32% agree and 3% strongly agree). The lower number of neutral responses indicates polarised views on this statement. From analysing the different roles that are presented in Figure [3t], it is evident that non-technical employees are more likely to agree with statement; however, this is a weak correlation as some non-technical staff do disagree. Furthermore, technical staff are distributed across all categorise. However, only those undertaking QA and technical roles strongly disagree. Figure [4t] illustrates that of users who strongly agree, they all have a high number of years of experience. The different categories of experience and then evenly distributed among the different response categorise, apart from those that strongly dis- agree that have between 5- and 10-years' experience only.

4.4 Results: Stage 2

In this stage, Principle Component Analysis (PCA) is performed using SPSS (version 24). Principle Component Analysis (PCA) is a statistical analysis technique that uses linear algebra techniques (specifically orthogonal transformation) to convert a data set believed to contain correlations into a subset of correlated data, known as principle components. In this process, the twenty items (questions) were used and these comprised the final attitude scale. In performing Principle Component Analysis (PCA), the level of variance (known as the Cronbach's alpha) is calculated and is used as a measure of how suitable the data is for identifying principle components. Nunnaly and Bernstein [26] state that .70 is an acceptable minimum for a scale that is newly developed. In our results, reliability for these 20 items of the sample produced a Cronbach's alpha of .86. It is important to note that the alpha co- efficient was not increased by eliminating items. Ferketich [13] recommended that corrected item-total correlations should range between .30 and .70 for a good scale. In our result, all 20 questions had significant item-total correlations and were retained (ranging from .24 to .60).

Automated software testing with oblique (nonorthogonal) rotation was used to investigate the components that stop people adopting automated testing. Analysis of the scree plot and eigenvalues led to the extraction of two components, which together accounted for 39% of variance in the data (see Table 4). We termed component one *non-software factors*, which comprises items relating to the finance, expertise, and time. The second component we

termed *software factors*, which comprises of ten items loading on this component related to the effectiveness, efficiency, completeness, and adaptability.

Our analyses of the twenty items reveals a two-component structure (Table 4). The non-software component consists of ten items explaining 29% of the variance and yields an eigenvalue of 5.8. Eigenvalue values are a measure of a component's magnitude. The non-software factors component is highly correlated with a high internal consistency (Cronbach's alpha=.813). The software factors consist of ten items explaining 10% of the variance and yields an eigenvalue of 2.0. The software factor also highly correlates (Cronbach's alpha=.790). The results present the commonality scores, indicating how well each item fits to the components.

Table [5] presents the average percentage response grouped by role type and also by identified components from the principle component factor analysis. It is evident that based on the identified factor, participants undertaking a technical role are more strongly agree that non-software reasons are preventing their use and they more strongly disagree that software reasons are preventing their use. It is also evident that they more strongly agree with the non-software factor being responsible for not adopting automated testing. Participants undertaking a non-technical role are more strongly agreeing that both non-software and software factors are preventing the use of automated testing.

Chapter 5

5.0 Discussion and Findings

5.1 Introduction

The purpose of this study was to test the nature of the relationship between a set of predictors including software characteristics, non-software issues and those reasons relating to practitioner support and opposition for automated software testing adoption. In this spirit, scholars have found that automated software testing characteristics, e.g. functionality and usability and adaptability, can have a strong effect on practitioner's support or opposition. In particular, the author sought to test these predictors across different scenarios in order to gain an understanding of how the perceptions of individuals operating in different roles and with different levels of experience differ. To that end, it has been established that there are key identifiable patterns surrounding the attitudes towards automated testing from employees undertaking different roles and having different levels of experience. These key findings can be used by employers within the software industry to better understand the viewpoints of their employees.

5.2 Research Findings

Based on the values in Table [5], the responses for technical roles are asymmetric as technical roles believe that reasons for not adopting automated software testing is due to the non-software factor. However, the responses for non-technical roles are symmetric and, agreeing with both non-software and software reasons are the factors preventing adoption. The author deduce that this could be down to the following reasons: (1) questions in non-software factor related to cost that all i.e. not just non-technical employees agree with; (2) Based on common practice in the IT sector, technical employees are often promoted to non-technical (managerial) roles, meaning that they have both technical and non-technical attitudes; and, (3) Non-technical might have less understanding on how capable technical people are. I.e., management lack of understanding of their employees' skill.

Based on the combination of the comprehensive basic analysis and principle component analysis, the author draws the key findings presented in the remainder of this section. Throughout this section the original questions and their responses are cross-referenced by adding the question number in parenthesise (e.q., q3 for question 3). In this section, free text

optional responses provided by the user are analysed alongside the previously discussed quantitative information. The full responses provided by 19 of the participants can be seen in Table [6], and as this section is trying to establish key findings from the data, they are used to substantiate quantitative patters. A summary of the key theses in the free text submissions can be seen in Table [7].

When asking participants about whether they believe a lack of skilled resources are preventing automated testing from being used, it is evident that managerial staff believe this to be true, whereas those with more technical expertise do not (q3). It is also evident that people do not believe that automated testing is not fully utilised due to people not realising its benefits (q9). Furthermore, technical roles do not believe there is an issue with open-source tools; however, less technical roles are more likely to support this argument (q12). In addition, there is a weak indication that those with technical expertise believe a high-level of expertise is required (q13). It is however evident that the majority of the participants believe that strong programming skills are required to undertake automated testing (q14). However, when relating this to the results of the principle component analysis, it is evident that technical employees do not believe that technical reasons are preventing the use of automated testing.

It is perhaps not too surprising that technical roles are more likely to believe that a high-level of expertise are required. This is because they are working closely with the technology and will have a comprehensive understanding of what knowledge is required. However, as demonstrated, technical roles are less likely to believe that skilled resources are preventing the use of automated testing as they have already gone through the learning process, becoming a competent automated tester. On the contrary, management are more likely to be viewing the capability within their organisation versus what is to be delivered, and therefore, a lack of skilled resource might refer to there being insufficient resource available to deliver a project on time, rather than the absence of expertise from preventing thorough software testing.

In terms of comments provided by the participants, 7 of the 19 responses were directed at the necessity and lack of expertise. All of the 7 responses provided in Table [7] are provided by individuals performing technical roles (cross reference participant number with Table [6]). Interestingly, all the responses do agree that technical knowledge is important, but one fascinating observation is that some responses draw attention to the fact that there is a lack

of training and mentorship within testing roles. One response even highlights the importance of individuals to be able to learn necessary skills independently. It is also interesting that a couple of responses directly state that the management of people is extremely important to help remove any skill and expertise gap, resulting in a more thorough and robust testing process.

Question	Non-software factors	Software factors	Commonalities
Q20	.786		.606
Q19	.708		.568
Q13	.688		.499
Q14	.623		.384
Q21	.608		.375
Q8	.572		.294
Q15	.515		.347
Q11	.492		.217
Q18	.483		.397
Q4	.447		.278
Q22	.419	.362	.404
Q6		.741	.496
Q3		.662	.401
Q9		.618	.371
Q7		.617	.452
Q10		.504	.243
Q5		.500	.386
Q16		.487	.419
Q17		.469	.303
Q12	.343	.380	.346
Eigenvalues	5.815	1.971	
Percent variance explained	29.076	9.857	

Table 4: Pattern Matrix. Rotation Method: Oblimin with Kaiser Normalisation. Rotation converged in 8 iterations.

	Technical role			Non-Technical role		
Component	% Disagree	% Neutral	% Agree	% Disagree	% Neutral	% Agree
Non-software	35	25	41	22	23	55
Software	50	21	29	31	24	45

Table 5: Average % responses to questions, grouped by role type and factor

Finding 1:

Although technical employees are more likely to believe that testers need a high-level of expertise and that open-source tools are challenging; this is not identified as a factor preventing their adoption. However, on the contrary non-technical roles do agree that an absence of expertise is preventing the use of automated testing.

Whether individuals have enough time to perform automated testing is polarised, with an even split agreeing and disagreeing. However, it has been identified that those with more junior roles are more likely to agree with this statement (q4). Furthermore, when considering how

difficult they are to learn, majority of the people disagree that they are time-consuming to learn. However, in general, the least experienced employees tend to agree, and so do managers and CEOs (q15). This is in agreeing with the results of the principle component analysis whereby technical staff are identified to agree that non-technical reasons are behind not adopting automated testing.

This finding agrees with the fact that the work levels and deadline pressures will be different in different organisations, and furthermore, people will respond and handle these pressures differently. The fact that junior employees are more likely to state that they do not have sufficient time to perform automated testing duties is explainable by the fact that junior employees might take longer to perform testing duties. This might also be due to a lack of experience due to the employee learning new expertise necessary for their role, which could be slowing down the testing.

Finding 2:

Those with less experience are more likely to agree that individuals do not have enough time to engage in automated testing. Furthermore, employees with less technical experience with automated testing and increased management responsibilities disagree that they are time-consuming to learn.

The majority of participants agree that commercial tools are expensive, but there is no distinct pattern (q11). However, there is a weak correlation that managerial roles are more likely to agree with the statement that test scripts are more expensive to generate (q19). This is further compounded whereby non-technical roles agree that there are high maintenance costs for test cases and scripts (q20). This agrees with the presented principle component analysis as both technical employees agree with non-technical reasons being responsible for not adopting automated testing. Furthermore, non-technical roles are split between believing that software and non-software factors are responsible for not adopting automated testing.

It is not surprising that majority of users agree that the costs of automated testing are expensive. Furthermore, the pattern that managerial staff more strongly agree with this statement is explainable through their closeness with the financial operations of the business. It is however quite surprising that managerial staff believe that automated testing has high maintenance costs. A fundamental aspect of automated testing is its reuse and ease of

maintenance. This difference in perspective is likely to originate from management's lack of understanding when it comes to fundamental aspects of automated testing.

Comments provided by the participants also mirror the fact that automated testing is expensive to perform and maintain, which is largely down to the cost of the testing team. One participant (#75) states that management do not see the wasted amount of time in automated software testing, and this could provide justification as to why non-technical roles agree that they are expensive to maintain. If they saw the amount of wasted time, they might have a better understanding of the true cost.

Finding 3:

The majority of participants agree that automated testing is expensive, with non-technical roles more likely to agree that they are expensive to use and maintain.

When considering whether automated testing tools and techniques lack functionality, in general the more experienced employees are likely to agree, but overall the majority disagree (q16). When asked whether people believe that automated tool are reliable enough, there was a very strong tendency to disagree (q17). There is a slight agreement in that people believe that automated testing tools lack support for testing non-functional requirements (q18). When asking about whether automated testing tools and techniques change too often, introducing problems that need fixing, the general trend is that a higher number of years of experience leads to an increased chance of disagreement. Furthermore, of the response categorises, non-technical roles agree/strongly agree (q21). This aligns with the findings from performing principle component analysis where non-technical roles more strongly believe that software reasons are preventing the use of automated testing, whereas those undertaking technical roles believe it is non-software issues.

The reason behind more experienced employees disagreeing that automated testing tools and techniques lack functionality is most likely down to the fact that more experienced employees either have fully mastered the tools, or they have developed sufficient workaround techniques. Furthermore, experienced staff do not believe that updates cause significant problems, which could be put down to the fact that they are experienced in how to handle revisions within the automated testing frameworks. Non-functional requirements are a secondary feature set of automated testing tools and techniques, and as such, are not the primary feature set integral

to their core use. This is most likely the reason behind why the majority of participants do not see an issue with their lack of support for non-functional requirements.

Many comments were received in regard to the capabilities of tools and techniques, and in general they state that the tools, techniques and frameworks do not lack functionality. Rather, they justify the complexity with tightly integrating the functionality within a project and how this can make it hard to reuse and fix revisions. Furthermore, it is evident that technical employees also believe that those in managerial roles do not understand what is involved in the implementation of automated testing. It is also interesting that one response from an individual performing a technical role (#64) even states that test scripts breaking is a good sign as it clearly demonstrates that they are working. A comment from an individual in management (#81) states that product deliver is more important that testing, demonstrating that for management their emphasis is on project completion rather than testing.

Finding 4:

All but the more experienced employees disagree that automated testing tools and techniques lack functionality. Furthermore, experienced employees are more likely to disagree that problems are introduced due to fast revisions, whereas those with managerial roles agree.

In terms of utilisation, when considering whether difficulties in preparing test data and scripts inhibits their use, only non-technical staff agree and there is a balanced response from technical roles (q5). Furthermore, the majority of participants do not believe that not have the right automation tools and the available frameworks are preventing use (q6). When asking staff specifically about the difficult to integrate tools being a problem, non-technical roles agree, technical roles are balanced with a slight emphasis in disagreement (q7). The majority of participants also agreed that requirements changing too frequently are impacting on their use (q8); however, it is also the case that non-technical roles do not agree. There is also a strong disagreement from technical staff that test scripts are difficult to reuse across different testing stages (q22). This finding also agrees with the performed principle component analysis where it was identified that non-technical staff more strongly believe the reasons for not adopting automated testing to be technical.

The fact that non-technical employees believe that there are difficulties, both in setting-up and maintaining automated tests, are prohibiting the use of automated testing tools is most likely down to the disconnect between non- technical and technical staff when it comes to

understanding limitations with software testing. All participants believe that there are sufficient frameworks to meet their individual testing requirements. Interestingly, only management believe that changing requirements do not impact on automated testing techniques. This difference could most likely originate due to a managerial misunderstanding of the impact on changing requirements throughout the software development cycle.

Comments provided by the participants do support the argument that those in testing roles understand the technical complexities involved and why automated testing might not be fully utilised. However, there are a lack of responses from managerial staff to justify that this is only a viewpoint from technical employees. There are many reasons specified for poor utilisation, from formal training and guidance, a preference to view automated testing second to manual, and that automation might be used for the wrong reasons i.e., to replace manual rather than complement.

Finding 5:

Only managerial staff believe that test preparation and integration inhibit their use. Furthermore, only managerial staff do not believe that software requirements change too frequently, having negative impacts on automated testing.

There is neither agreement nor disagreement that a lack of support is pre- venting the use of automated testing. There is however an observation that consultants tend to agree with this statement (q10). This is interesting as it demonstrates that there is no majority, either in terms of role or experience, that are stating a lack of support is preventing them adopting and using automated testing within their organisation. However, it is also worth noting that the responses to this question are rather polarised with people agreeing and disagreeing, but overall there are few holding strong views on this. This is consistent with the performed principle component analysis, which deter- mined that non-technical roles and technical roles both agree (technical more strongly) that non-software factors, such as finance, expertise, and time are preventing the adoption of automated testing.

Comments received from participants detail that training is a common limiting factor to their update, but the biggest theme is that non-technical either do not understand nor value test automation. This means that automation is seen as an afterthought from manual testing and thus will not be well supported by their employer.

Finding 6:

Whether a lack of support is preventing automated testing use is polarised.

Chapter 6

6.0 Conclusion

There seems to be a big gap in automation testing due to lack of skills. From experience the author think the main cause for this gap is that there is no common entry to automated software testing. The author thinks currently there are two career paths into test automation which are either people that do computer science at university and become QA engineers, but clear majority of those doing computer science become developers. The other path is from manual testing, where manual testers will learn to code and get into automation. This area needs more research to find a common entry into software automation career path.

In total ten key findings have been established, demonstrating key differences in perceptions of both technical and managerial employees, as well as employees of different experience levels. The two-stage analysis approach presented in this thesis demonstrated that an overarching two-factor split can be established when considering the attitudes towards automated testing of both technical and non-technical staff. It has been established that technical employees strongly believe that preventing factors to automated testing use are those of a non-software nature, whereas non-technical roles believe it is both the reasons of software and non-software challenges. These attitudes have been further analysed and explained through considering different roles and years of experience.

Although the study is based on the responses from 81 different users, future work should focus on gaining a larger number of samples with a more even distribution across the different roles' types.

One of the main limitations of this survey is that it is performed on a relatively small (81) dataset, which makes it difficult to form a generalised view and opinions. Loadings with a high *Cronbach's alpha* that have several high loading marker variables (> .80) do not require such large sample sizes as solutions with lower loadings [31]. The results produced a *Cronbach alpha* of .86, justifying the reliability of the survey. Another limitation is that the questionnaire does not cover all factors that are involved in the process of automated software testing, and therefore, the key findings might not be true or applicable in every case. However, this

research has achieved its aim in developing an understanding as to why people are not adopting automated planning, which establishes a suitable position for further research

Another limitation to our study is that we consider on a large-scale automation software in a general sense rather than focus on any specific automation software. Research finds that public opposition tends to be highest when projects are proposed and then disappears once construction is completed [36]. However, the author believe this limitation to be fairly minor because he was trying to understand practitioner's attitude about automation generally rather than any relation to any testing software adoption. The selection of questionnaire items always restricts the potential structure that can emerge from innovation adoption studies. The author designed the questionnaire to include items relating to a broad range of potential experiences, motivated both theoretically and by prior qualitative research.

7.0 References

1. (2019). URL https://www.infosys.com/IT-services/validation-solutions/Documents/infosys-test-automation-accelerator.pdf
2. Abdi, H., Williams, L.J.: Principal component analysis. Wiley interdisciplinary reviews: computational statistics **2**(4), 433–459 (2010)
3. Ammann, P., Offutt, J.: Introduction to software testing. Cambridge University Press (2016)
4. Berner, S., Weber, R., Keller, R.K.: Observations and lessons learned from automated testing. In: Proceedings of the 27th international conference on Software engineering, pp. 571–579. ACM (2005)
5. B¨ohme, M., Paul, S.: A probabilistic analysis of the efficiency of automated software testing. IEEE Transactions on Software Engineering **42**(4), 345–360 (2015)
6. Boone, H.N., Boone, D.A.: Analyzing Likert data. Journal of extension **50**(2), 1–5 (2012)
7. Charette, R.N.: Why software fails [software failure]. IEEE spectrum **42**(9), 42–49 (2005)
8. Dustin, E., Garrett, T., Gauf, B.: Implementing automated software testing: How to save time and lower costs while raising quality. Pearson Education (2009)
9. Dustin, E., Rashka, J., Paul, J.: Automated software testing: introduction, management, and performance. Addison-Wesley Professional (1999)
10. Eldh, S., Hansson, H., Punnekkat, S., Pettersson, A., Sundmark, D.: A framework for comparing efficiency, effectiveness and applicability of software testing techniques. In: Testing: Academic & Industrial Conference-Practice and Research Techniques (TAIC PART'06), pp. 159–170. IEEE (2006)
11. Elghondakly, R., Moussa, S., Badr, N.: Waterfall and agile requirements-based model for automated test cases generation. In: 2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS), pp. 607–612. IEEE (2015)
12. Faraj, S., Sproull, L.: Coordinating expertise in software development teams. Management science **46**(12), 1554–1568 (2000)
13. Ferketich, S.: Focus on psychometrics. aspects of item analysis. Research in nursing & health **14**(2), 165–168 (1991)
14. Fewster, M., Graham, D.: Software test automation: effective use of test execution tools. ACM Press/Addison-Wesley Publishing Co. (1999)
15. Garrett, T.: Useful automated software testing metrics. Software Testing Geek (2011)

16. Graham, D., Fewster, M.: Experiences of test automation: case studies of software test automation. Addison-Wesley Professional (2012)
17. Jansing, D., Novillo, J., Cavallo, R., Spetka, S., et al.: Enhancing the effectiveness of software test automation. Ph.D. thesis (2015)
18. Jolliffe, I.T., Cadima, J.: Principal component analysis: a review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 374(2065), 20150202 (2016)
19. Kasurinen, J., Taipale, O., Smolander, K.: Software test automation in practice: empirical observations. Advances in Software Engineering 2010 (2010)
20. Kumar, D., Mishra, K.: The impacts of test automation on software's cost, quality and time to market. Procedia Computer Science 79, 8–15 (2016)
21. Leitner, A., Ciupa, I., Meyer, B., Howard, M.: Reconciling manual and automated testing: The AutoTest experience. In: 2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07), pp. 261a–261a. IEEE (2007)
22. Melton, J.R.: The hidden benefits of automated testing. In: Proceedings of the 2015 Aerospace Testing Seniar. CVENTS (2015)
23. Mittal, V., Garg, N.: Test automation using selenium webdriver 3.0 with c (2018)
24. Monier, M., Elmahdy, M.M.: Evaluation of automated web testing tools. International Journal of Computer Applications Technology and Research 4(5), 405–408 (2015)
25. Nunnally, J.C., Bernstein, I.H., Berge, J.M.t.: Psychometric theory, vol. 226. McGraw-hill New York (1967)
26. Panichella, S., Di Sorbo, A., Guzman, E., Visaggio, C.A., Canfora, G., Gall, H.C.: How can I improve my app? classifying user reviews for software maintenance and evolution. In: 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 281–290. IEEE (2015)
27. Rafi, D.M., Moses, K.R.K., Petersen, K., M¨antyl¨a, M.V.: Benefits and limitations of automated software testing: Systematic literature review and practitioner survey. In: Proceedings of the 7th International Workshop on Automation of Software Test, pp. 36–42. IEEE Press (2012)
28. Rahman, A.A., Hasim, N.: Defect management life cycle process for software quality improvement. In: 2015 3rd International Conference on Artificial Intelligence, Modelling and Simulation (AIMS), pp. 241–244 (2015). DOI 10.1109/AIMS.2015.47
29. Rex, B.: Managing the testing process: Practical tools and techniques for managing hardware and software testing (2002)
30. Tabachnick, B.G., Fidell, L.S., Ullman, J.B.: Using multivariate statistics, vol. 5. Pearson Boston, MA (2007)

31. Taipale, O., Kasurinen, J., Karhu, K., Smolander, K.: Trade-off between automated and manual software testing. International Journal of System Assurance Engineering and Management 2(2), 114–125 (2011)
32. Tracey, N., Clark, J., Mander, K., McDermid, J.: An automated framework for structural test-data generation. In: Proceedings 13th IEEE International Conference on Automated Software Engineering (Cat. No. 98EX239), pp. 285–288. IEEE (1998)
33. Vaismoradi, M. and Snelgrove, S., 2019, September. Theme in qualitative content analysis and thematic analysis. In Forum Qualitative Sozialforschung/Forum: Qualitative Social Research (Vol. 20, No. 3).
34. Vogel-Heuser, B., Fay, A., Schaefer, I., Tichy, M.: Evolution of software in automated production systems: Challenges and research directions. Journal of Systems and Software 110, 54–84 (2015)
35. Vos, T.E., Marin, B., Escalona, M.J., Marchetto, A.: A methodological framework for evaluating software testing techniques and tools. In: 2012 12th international conference on quality software, pp. 230–239. IEEE (2012)
36. Warren, C.R., Lumsden, C., O'Dowd, S., Birnie, R.V.: green on green: public perceptions of wind power in Scotland and Ireland. Journal of environmental planning and management 48(6), 853–875 (2005)
37. Zhou, Z.Q., Sinaga, A., Susilo, W., Zhao, L., Cai, K.Y.: A cost-effective software testing strategy employing online feedback information. Information Sciences 422, 318–335 (2018)

8.0 Appendix

Response No	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22
1	Tester/Engineer/Analyst/Architect	6	5	3	3	4	4	5	5	5	4	3	1	2	1	2	1	3	2	2	2	2
2	Tester/Engineer/Analyst/Architect	17	4	1	5	2	2	2	2	4	3	2	3	4	3	2	2	2	4	5	3	4
3	Senior QA	20	5	4	2	2	2	2	4	4	4	2	2	4	2	2	2	2	2	4	4	2
4	Tester/Engineer/Analyst/Architect	5	4	1	3	4	3	4	5	1	3	3	3	4	4	3	2	3	3	3	2	3
5	Consultant	31	4	4	5	3	3	3	4	5	1	3	2	4	3	3	3	3	2	3	3	3
6	Test Automation	10	3	4	5	4	3	2	4	3	5	4	3	4	2	3	4	4	3	4	4	2
7	Consultant	12	4	2	4	1	4	3	4	4	4	4	2	4	2	1	1	1	1	2	3	3
8	CEO	10	5	3	5	5	4	2	4	4	2	3	2	4	4	3	3	2	4	4	3	4
9	Manager	21	5	4	4	3	4	3	3	3	5	4	3	4	4	5	4	4	4	4	5	5
10	QA	5	3	2	2	1	1	4	2	3	5	1	1	2	1	3	2	1	1	3	4	1
11	Tester/Engineer/Analyst/Architect	6	2	4	5	3	2	4	5	4	5	2	5	5	3	3	1	5	2	5	4	1
12	Tester/Engineer/Analyst/Architect	3	2	2	2	4	4	2	4	4	4	1	3	3	4	2	2	4	3	3	3	4
13	Senior Automation	10	2	2	3	1	2	3	2	4	4	2	3	4	3	2	2	2	2	3	3	2
14	Tester/Engineer/Analyst/Architect	5	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2	3	1
15	Senior Tester/Engineer/Analyst/Architect	4	2	3	3	1	2	4	2	2	5	1	3	4	3	1	1	3	3	4	3	3
16	Senior Consultant	15	4	5	4	2	4	5	2	2	4	4	4	5	4	2	2	2	4	4	4	2
17	Tester/Engineer/Analyst/Architect	3	4	4	3	2	4	4	2	2	5	3	2	3	4	2	3	4	2	4	4	4
18	Tester/Engineer/Analyst/Architect	5	3	1	2	2	2	3	1	2	2	2	1	4	2	2	2	2	2	2	2	2
19	Senior Tester/Engineer/Analyst/Architect	17	4	5	5	3	3	4	5	4	5	2	4	4	4	3	3	5	4	4	4	3
20	Senior Tester/Engineer/Analyst/Architect	6	3	1	3	2	3	4	3	4	5	2	3	3	2	3	2	3	3	3	3	3
21	Senior QA	6	4	3	4	2	3	4	3	5	4	3	3	4	3	2	3	4	3	4	4	3
22	Senior Automation	10	4	2	4	2	4	2	2	3	4	2	2	3	3	2	2	4	2	2	2	2
23	Test Automation	5	2	3	2	2	1	2	1	1	4	4	2	2	1	1	4	2	3	4	4	2
24	Senior Automation	10	3	4	4	4	4	4	5	5	5	3	3	4	2	3	2	3	2	2	3	4
25	Senior Automation	11	4	4	5	2	3	5	1	2	5	1	4	4	2	1	1	4	1	2	3	2
26	Consultant	13	4	4	4	1	2	2	3	3	2	2	3	4	3	2	1	3	3	2	4	3
27	Senior Tester/Engineer/Analyst/Architect	9	4	3	4	2	2	3	4	3	4	2	3	4	3	2	2	4	3	4	3	4
28	Senior Tester/Engineer/Analyst/Architect	10	3	4	1	1	1	2	5	5	1	3	1	1	1	1	3	3	1	4	4	3
29	QA	3	3	4	3	2	4	4	5	1	4	3	3	4	3	2	3	3	4	4	4	4
30	CEO	5	5	3	4	4	3	5	1	1	3	3	4	5	4	3	3	2	5	4	4	4
31	QA	3	3	2	5	2	4	5	2	2	4	1	2	4	2	3	2	4	2	2	4	2
32	Senior Tester/Engineer/Analyst/Architect	7	4	5	5	2	5	5	3	5	5	4	3	5	4	2	1	5	4	5	4	2
33	Senior QA	8	2	4	4	2	4	3	2	4	4	3	1	4	2	3	2	4	4	4	4	4
34	Senior Tester/Engineer/Analyst/Architect	2	4	2	3	3	3	2	1	2	3	1	1	2	1	1	2	2	2	2	3	3
35	Test Automation	5	5	3	5	4	4	5	2	5	5	4	3	4	3	4	3	3	5	4	4	4
36	Student	9	5	3	5	4	4	4	5	4	5	3	4	4	2	3	3	4	4	4	4	4
37	Tester/Engineer/Analyst/Architect	3	3	4	5	4	5	2	4	4	3	2	2	2	2	4	5	5	3	3	3	4
38	Senior Tester/Engineer/Analyst/Architect	20	4	4	5	3	3	3	2	3	4	2	4	3	4	2	3	4	4	4	4	4
39	Senior Tester/Engineer/Analyst/Architect	5	4	5	5	1	1	3	1	2	4	1	2	2	4	2	1	3	2	2	1	1
40	Tester/Engineer/Analyst/Architect	2	2	2	4	2	2	4	3	4	2	1	2	2	3	2	2	4	4	4	3	3
41	Senior Tester/Engineer/Analyst/Architect	15	4	2	5	2	5	2	2	2	4	2	2	2	4	2	2	2	2	3	3	2
42	Senior Tester/Engineer/Analyst/Architect	5	2	1	4	3	4	4	2	2	3	2	2	2	2	3	3	2	2	2	3	2
43	Tester/Engineer/Analyst/Architect	7	4	1	2	2	2	2	4	4	4	2	4	4	4	2	2	2	2	4	2	2
44	QA	9	3	1	2	1	1	4	1	3	5	1	2	4	3	1	1	1	2	2	4	2
45	Tester/Engineer/Analyst/Architect	1.5	3	2	3	2	3	3	4	4	3	2	2	3	3	2	2	3	2	3	3	4
46	Tester/Engineer/Analyst/Architect	18	1	4	2	4	3	3	1	3	4	3	3	3	3	1	1	2	2	2	3	2
47	Senior QA	8.5	5	5	5	2	4	2	5	5	1	1	3	3	4	3	3	3	4	4	4	2
48	Tester/Engineer/Analyst/Architect	2	3	4	2	2	4	3	3	1	4	2	2	4	2	3	2	2	2	4	2	2
49	Manager	10	4	4	5	3	4	5	2	2	4	2	4	5	3	2	2	4	4	5	5	4
50	Senior Automation	0.6	5	3	3	2	2	3	3	5	1	1	4	4	4	3	4	4	4	4	4	4
51	Senior Automation	12	3	4	2	4	1	1	3	4	4	2	2	4	3	2	2	4	3	3	4	3
52	Senior Automation	5	4	3	2	4	5	3	1	2	1	2	1	3	4	1	1	3	1	2	2	1
53	Senior Tester/Engineer/Analyst/Architect	5	2	2	4	1	3	2	2	1	5	1	2	5	3	4	2	2	3	2	2	2
54	Test Automation	25	4	4	5	3	4	3	4	4	3	3	2	4	3	4	2	4	3	4	4	4
55	Senior QA	15	3	4	3	4	4	4	3	2	3	2	2	4	3	2	4	3	4	4	4	3
56	Senior Tester/Engineer/Analyst/Architect	15	2	2	2	2	2	2	2	2	4	2	3	4	3	3	3	4	3	4	4	4
57	QA	2	4	5	3	2	2	4	4	4	1	4	5	4	4	4	1	5	4	4	4	4
58	Senior QA	8	4	3	4	3	4	4	2	3	4	4	4	4	4	3	3	3	4	4	3	2
59	Senior QA	12	3	2	4	2	4	5	2	4	4	2	4	4	4	3	2	4	4	4	4	2
60	Manager	13	4	5	5	3	2	3	4	2	4	1	2	3	2	1	1	1	1	1	3	2
61	CEO	25	5	4	4	4	3	4	2	2	2	4	4	3	4	4	4	4	3	4	4	4
62	Senior QA	10	4	4	3	2	2	5	2	2	4	2	3	2	1	2	1	4	4	4	4	4
63	Tester/Engineer/Analyst/Architect	6	4	5	4	3	4	3	3	3	3	3	4	3	4	2	3	4	3	4	4	2
64	Tester/Engineer/Analyst/Architect	11	4	1	2	2	2	2	4	1	4	3	4	5	2	4	4	2	2	2	2	2
65	QA	16	4	3	5	3	2	4	3	2	4	2	2	3	1	1	1	3	4	2	4	2
66	Tester/Engineer/Analyst/Architect	13	2	2	2	3	3	3	1	3	5	2	1	4	2	3	2	3	2	2	4	2
67	Test Automation	8	2	3	4	2	2	4	4	4	2	2	3	4	3	3	3	3	3	2	3	3
68	Senior Automation	20	4	2	4	3	3	4	4	4	4	3	2	2	4	3	2	3	4	3	4	4
69	Tester/Engineer/Analyst/Architect	15	1	2	1	1	1	5	1	1	3	2	2	2	2	2	2	3	3	3	1	3
70	Tester/Engineer/Analyst/Architect	8	5	4	4	2	4	4	5	3	4	4	3	4	4	3	4	4	4	3	4	4
71	Tester/Engineer/Analyst/Architect	5	4	2	2	2	2	2	4	3	2	1	1	1	2	1	1	1	2	1	2	1
72	Senior Tester/Engineer/Analyst/Architect	30	2	5	2	2	3	2	2	4	4	2	4	4	4	2	2	3	4	4	4	4
73	Manager	20	5	4	5	4	5	5	5	3	5	5	4	4	3	3	3	5	4	5	5	5
74	Manager	12	3	2	4	2	2	4	2	3	4	2	3	3	2	2	2	3	4	3	3	2
75	Tester/Engineer/Analyst/Architect	12	5	1	2	4	1	2	1	1	1	1	3	3	1	1	2	1	1	2	2	2
76	QA	7	5	2	4	4	2	2	1	5	3	1	1	2	1	1	1	4	2	1	4	2
77	Senior Automation	33	5	4	5	5	4	3	4	5	3	2	3	5	3	4	2	2	3	2	3	2
78	Senior Tester/Engineer/Analyst/Architect	25	3	2	4	2	2	3	2	2	3	1	2	4	2	2	2	4	3	2	4	2
79	Manager	4	2	5	2	1	2	5	1	2	2	3	2	4	4	2	2	1	4	5	5	2
80	Tester/Engineer/Analyst/Architect	10	2	3	4	2	2	3	2	3	4	2	2	4	3	2	2	4	3	4	3	4
81	Manager	20	4	2	4	5	5	4	2	3	4	3	3	3	4	4	3	3	3	3	2	4

Table 6: Questionnaire responses

Table [6] provides the full breakdown of the individual questionnaire responses. Table [7] provides comments provided at the two designated points when performing the test.

Response #	Summary points
2	– Problems are with people and not technology.
3	– Skilled team will prevent issues.
5	– False expectations from management. – Lack of consideration to test data and scripts. – People are the cost and over expectation of automation. – Setting and using testing tools requires knowledge.
15	– Company recognise value of automation.
20	– High maintenance due to when testing is performed.
24	– Management do not understand automation.
33	– People management is poor.
35	– Lack of mentorship and guidance. – Self-teaching is important and widely performed.
39	– Management do not understand time/effort required. – Automation is always seen as a nice to have after manual is performed. – Benefits of automated testing are significant.
43	– Implementation of open-source frameworks is key to their value.
49	– Training is the most challenging aspect.
51	– Management see value in automated testing for the stability of the end product.
55	– Training is the most challenging point.
56	– Once automated testing is used, it may become tricky to understand its value.
64	– Automation still viewed as secondary. – The need to maintain scripts as new tools and techniques develop is a good sign.
66	– Automated testing is more of a process than a skill. – Test automation need to be pre-planned to consider software development factors.
75	– New experienced testers are likely to make mistakes. – Automation engineers lack product knowledge and are disconnected from the project they are working on. – Those involved in automation spend lots of time making scripts and maintaining them. – Managers often do not see the level of waste in automated testing. – Managers invest heavily without seeing or understanding the benefit. – Management pushing advice on automation without knowledge is a bad thing. – People often build their own frameworks, but spending too much time here can be disadvantageous for the project. – Translation of testing output to management is currently under performed. – Changing requirements is normal and a necessary part of a product?s life-cycle. – Automation used for the wrong reasons. – Automated testing can be hard to see the true benefits, and therefore management are likely to want it until they do not see any positive impact. – Commercial tools are too expensive. – Open-source tools are not hard to utilise, unless the person is unfamiliar with the area. – Knowing when and how to use automated testing is important. – Level of programming ability is less important than the ability to learn when needed. – A wrongly utilised tool is expensive. – Challenges with maintenance stem from a lack of understanding. – Bespoke and low-level test scripts can be translate to new projects, but this is necessary as they encode application-specific behaviour.
77	– A deep understanding is required.
81	– Product delivery is more important than testing.

Table 7: Summary of free-text responses

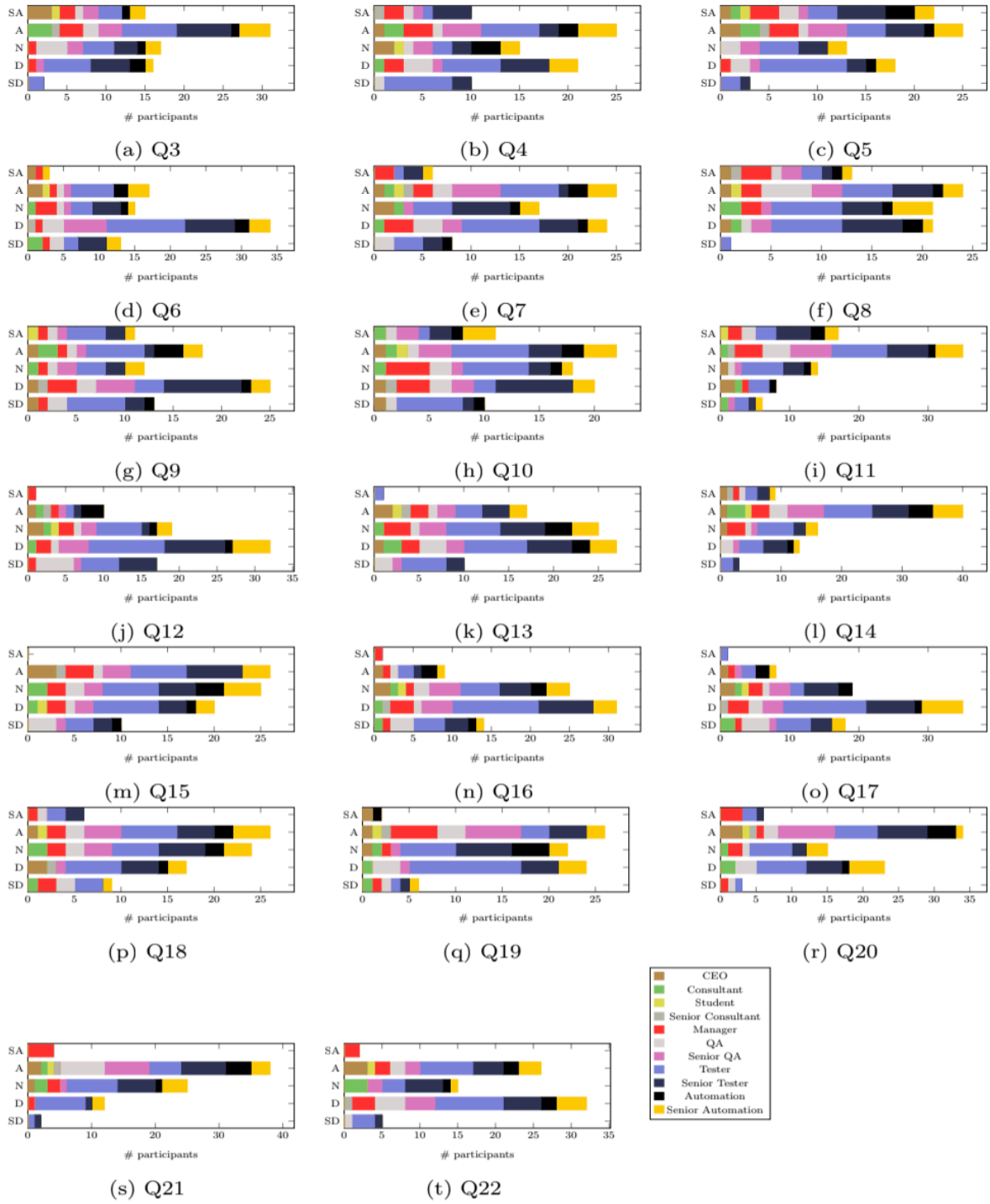

Fig. 3: Responses for each survey by question, illustrating the distribution of answers based on job role

Fig. 4: Responses for each survey by question, illustrating the distribution of answers based on number of years' experience

Software Test Metrics

Metric Name	Description	Category
Test Coverage	Total number of test procedures/total number of test requirements. The *Test Coverage* metric will indicate planned test coverage.	Coverage
System Coverage Analysis	The *System Coverage Analysis* measures the amount of coverage at the system interface level.	Coverage
Test Procedure Execution Status	Executed number of test procedures/total number of test procedures This *Test Procedure Execution* metric will indicate the extent of the testing effort still outstanding.	Progress
Error Discovery Rate	Number total defects found/number of test procedures executed. The *Error Discovery Rate* metric uses the same calculation as the defect density metric. Metric used to analyze and support a rational product release decision	Progress
Defect Aging	Date Defect was opened versus date defect was fixed *Defect Aging* metric provides an indication of turnaround of the defect.	Progress
Defect Fix Retest	Date defect was fixed & released in new build versus date defect was re-tested. The *Defect Fix Retest* metric provides an idea if the testing team is re-testing the fixes fast enough, in order to get an accurate progress metric	Progress
Current Quality Ratio	Number of test procedures successfully executed (without defects) versus the number of test procedures. *Current Quality Ratio* metric provides indications about the amount of functionality that has successfully been demonstrated.	Quality
Quality of Fixes	Number total defects reopened/total number of defects fixed This *Quality of Fixes* metric will provide indications of development issues.	Quality
	Ratio of previously working functionality versus new errors introduced The *Quality of Fixes* metric will keep track of how often previously working functionality was adversarial affected by software fixes.	Quality
Problem Reports	Number of Software Problem Reports broken down by priority. The *Problem Reports Resolved* measure counts the number of software problems reported, listed by priority.	Quality
Test Effectiveness	Test effectiveness needs to be assessed statistically to determine how well the test data has exposed defects contained in the product.	Quality
Test Efficiency	Number of test required / the number of system errors	Quality

Common Software Test Metrics[5]