Journal of Contemporary Management
Volume 17 Issue 1

Software testing: some influencing factors in a South African organisation

DOI nr: https://doi.org/10.35683/jcm19076.57

T SEKGWELEO
Cape Peninsula University of Technology
Cts330ci@gmail.com

T IYAMU*
Cape Peninsula University of Technology
IYAMUT@cput.ac.za
***** corresponding author
ORCID NR: https://orcid.org/0000-0002-4949-094X

ABSTRACT

Organisations rely on software in providing services to its clients and business partners. Due to this reliance, the quality of software contributes to dictating the type of services that some organisations provide, which in return, influence their relationship with clients and partners. For this reason many organisations focus on the quality of software, which makes testing critical. The challenge is that various factors influence the software testing in an organisation. The objective of this study was to examine the factors that influence the software testing and evaluation in an organisation in the automotive industry. The qualitative methods and case study approach were applied in the case. The data were analysed by following the interpretivist approach, which was guided by two theories: actor network theory (ANT) and diffusion of innovation (DOI). Based on the findings from the analysis, a framework was developed, which can guide an understanding of the factors that influence software testing and evaluation in this specific organisation as well as a guide for other organisations.

Key phrases

ANT; case study; DOI; software testing and qualitative methods

1. INTRODUCTION

Organisations either develop or purchase software for business purposes (Iyamu, Sekgweleo & Mkhomazi 2013:607). Such software requires testing prior to its implementation to ensure that it functions as expected. Software testing is performed by following software quality standards to identify defects and flaws in the software and ensures they are removed (Sowunmi, Misra, Fernandez-Sanz, Crawford & Soto 2016:2). It is for this reason that organisations require skilled personnel, software testers, who understand various types of testing as well as the right tools that enable them to properly conduct software testing and evaluation. Through the use of testing tools, software testers, generate, manage and execute the testing of software in a specific environment maintained for a particular type of test (Bhardwaj 2015). According to Abbas, Sultan and Bhatti (2017:40), with the automation of software testing tools, software testers can reliably test the software in less time and repeatedly reuse those tools to retest the software.

The software testing process is required early in the life cycle, prior to any system coding and during each of the stages preceding implementation (Munassar & Govardhan 2010:96). According to Skidmore (2006:51), methodologies such as V-Model provide a relationship between development and testing in ensuring proper testing and quality assurance throughout the entire project life cycle. Kasurinen (2012:18) suggests that a test process encompasses test planning & control, test analysis & design, test implementation & execution, evaluating exit criteria, reporting and test closure activities.

Software testing can be conducted manually and also through automation testing tools (Mishra, Ostrovska & Hacaloglu 2017:931). Irrespective of the tools or approaches that are employed in the testing of software, there are challenges (Gautam & Nagpal 2016:10291). Bamotra and Randhawa (2017:123) argue that manual testing is time consuming, resource intensive and allows some defects to remain uncovered. Automation tools, then, enable automation testers to record and replay those test cases or user actions which could be performed manually by a software tester (Singh & Tarika 2014:1510).

The objective of this study was to examine the factors that influence the testing and evaluation of software in a South African organisation. This was done through the lenses of ANT and Roger's DOI, because of the theories' focus. In ANT, human and non-human are actors, and of equal capability (Callon 1986:197). The theory focuses on formulation of socio-technical networks with aligned interests, including the relationship and interaction between the actors within the networks (Effah 2012:5). DOI focuses on the innovation of technologies that is communicated to the members of social system through certain communication channels to masses over time (Chang 2010:1).

2. PROBLEMATISING SOFTWARE TESTING CHALLENGES

The testing of software is purely based on the requirement specification knowledge (Nidhra & Dondeti 2012:35). This is arguably correct in that despite availability of many testing tools, software testing continue to encounter challenges in organisations. According to Farooq and Quadri (2013:43), potential failures of software can be avoided by performing exhaustive system testing, allowing possible permutations of inputs which can be either valid or invalid. This is often done through automated testing tools, to effortlessly rerun as many times as possible, in alleviating human error during software testing (Mishra & Pradhan 2012:291). Despite the numerous efforts and emphases, the problem still remains with testing of software in many South African organisations. This can be attributed to the fact that there is no framework to guide how the tools can be used. The lack of framework resulted in employees attempting their individual ways and approaches, which result in human errors. Some of the errors have had severe financial consequence for many South African organisations in recent years (Sekgweleo 2015:52).

3. LITERATURE REVIEW

Software development is a process of developing a software system following a particular system development methodology (Bassil 2012:745). The systems development life cycle (SDLC) is a practical and systematic process adopted by software developers in developing software, enabling the software development team to plan, develop and control the way in which software is developed (Amlani 2012:9). According to Bukhari, Faisal and Hira (2014:2), SDLC consists of various stages including the collection of requirements, design, development, testing and implementation. SDLC methodologies can be either traditional or agile.

Software testing occurs throughout the development or enhancement of existing software. Hooda and Chhillar (2015:11) describe software testing as a process in which both software requirements and components are tested manually or through the use of software automation tools to determine if the software meets the specified user requirements. Software is evaluated with the intention of producing quality software (Saleh 2011:98). Organisations rely on software for competitiveness and sustainability as it enables business to smoothly execute its functions without any disruptions.

It is essential to have clear business requirements before developing and testing the software in order to deliver quality software. Problems relating to user requirements, when determined later in the software development process, often negatively impact the software cost (Sener & Karsak 2012:21). These problems might be realised as early as the planning stage if the software testing is involved early in the software development life cycle. Software inspection enables the software testing team to systematically detect defects in all stages of software development (Qazi, Shahzadi & Humayun 2016:11). This practice assists in exposing defects early so that they can be fixed in time. It is easier to detect and correct defects sooner rather than later in the software development life cycle. There is various software testing methods that can be employed to test the software.

There are two common types of testing methods, which are black box (functional) and white box (structural) testing (Mishra *et al*. 2017:934). In black box testing, the software is tested upon expected output; the tester does not need to know the internal workings condition of the software (Dhiman & Sharma 2016:508). In white box testing the software tester tests both the functionality as well as the internal workings of the software (Jamil, Arif, Abubakar & Ahmad 2016:178). In the use of any of the methods, the tools and testers make a difference in the testing and evaluation of software in many instances (Nidhra & Dondeti 2012:30).

4. RESEARCH METHODOLOGY

The qualitative methods and the case study approach were applied in this study. This is primarily because the qualitative methods focus on assisting empirical answers with regards to people's behaviour, opinions and cultures (Linsley, Kane & Barker 2019:93). The case study method was employed because the approach is considered most suitable in an environment where there are large numbers of variables in a small number of applied units of analysis when the context is of great importance, as in this study (Henderson 2016:8). Hollweck (2015:108) argues that a high-quality case study puts emphasis on rigour, validity and reliability. A South African organisation was selected for the study. The organisation was given a pseudo name, Mootledi Logistics, because the organisation wanted its identity preserved. Based on research code of code, the organisation’s request had to be adhered to.

The semi-structured interview technique was used to collect data. A total of 14 employees in both business and IT units, at various levels were interviewed. The researcher ended the interview process at the total of 14, because there was no new information that was forthcoming. The selection of the technique was influenced by its characteristics, as asserted by Alshenqeeti (2014:40) “a semi-structured interview has a more flexible version of the structured interview as it allows depth to be achieved by providing the opportunity on the part of the interviewer to probe and expand the interviewee's responses”. This allows for the opportunity to record conversations, take notes, refine questions and clarify things that were unclear during the interviews process, which enriched the data. In the process, the interviewees were assured of anonymity, and their right to privacy was upheld. This formed part of the ethical considerations for this research. In addition, some documents that relate to the study in the areas of organisational structure, standards and policies were granted access by respondents.

The interpretivist approach was employed in the analysis of the data. The approach was guided by ANT (Callon 1986:226) and DOI (Rogers 1995:17). ANT was applied first, followed by DOI, primarily because it was critical to first establish the formulation and existence of networks so as to know how the technology can be diffused in the environment. It was also necessary to first understand the tools, methods, and relationships between the actors involved in software testing, prior to assessment of innovative and diffusion.

5. DATA ANALYSIS AND DISCUSSION

As discussed earlier, both ANT and DOI were applied as lenses in the data analysis. As explained in the overview section concerning order-of-use, ANT was applied first, followed by DOI. The focus of the two theories is quite different. The ANT focuses on actors (human and non-human), heterogeneity of networks, and actors interactions and relationships within networks (Callon 1986; Iyamu *et al*. 2013). The ANT was selected to help examine the interactions and relationship among actors within networks (software development teams, project managers' team, and business units) in the testing of software. This was to gain an understanding of the factors that influence processes and activities in testing software, through the interaction and relationship among the actors. The ANT, however does not explicitly focus on how technology solutions (such as software) are implemented or diffused, which is a gap that needed to cover. From that angle, the DOI was employed.

The DOI focuses on innovation, and how they are diffused within an environment (Rogers 1995). The theory has been used for many years to examine the processes by which innovations are diffused and communicated within environments (Chang 2010). According to Overhage and Schlauderer (2012), DOI theory clarifies the *why,* and *at what rate* innovations gets diffused to a social system over a period of time. The theory is not concerned with the respective actors and their interactions, which ANT covers.

A total of 14 employees participated in the study. For the purpose of this study, and analysis of the data, the participants and the organisation were assigned code-name: ML for the organisation (Mootledi Logistics) and 01 to 14 for the participants, which can be read as ML01 to ML14. The code name makes referencing easier during analysis. For example, ML01, 5:1-3 refers to: the organisation ML, first participant, page number 5, and line numbers 1 to 3.

5.1 Actors

At Mootledi Logistics, both human and non-human actors were involved in the testing and evaluation of software. The human actors were from both the information technology (IT) and business departments. The actors had a common interest, to achieve the objectives of the organisation through software testing, evaluation and use. The actors from the IT department included project managers, business analysts, software developers and software application manager. From the business department perspective, the actors were business end-users, including business managers, and employees from finance, sales, operations, fleet services, human resources and risk management units. The non-human actors in the testing of software within the organisation fell into two categories: technical and non-technical (i.e. business managers, business analysts, and end users), with technical including hardware, software and network. Some hardware for enabling and supporting software testing and evaluation included personal computers, servers and scanners.

Operating systems and software testing tools such as Mentis and Selenium were involved in the testing of software in the organisation. The network consists of a local area network (LAN), wide area network (WAN) and Wi-Fi. The non-human actors involved in the testing and evaluation of software in the organisation included documentation, process and methodology. The process that was followed was defined by the organisation. The methodology was agile method (i.e. Scrum). It is a transparency enabler, used to deliver a framework that leads to sustainable and continuous improvement. One of the participants explained: "*We make use of N-Unit for integration testing; we use Selenium for automated web testing. Part of the agile methodology was followed in the delivering of software in an iterative and efficient manner*" (ML01, 08:0283-0284).

5.2 Networks

In Mootledi Logistics, groups and units (networks) were involved in testing and evaluation of software. These networks were divided into two main groups: the IT and business departments. Within the main networks, there were sub networks. Some networks also replicated themselves within other networks, which ANT refers to as heterogeneity. The networks were consciously and unconsciously formed. From the IT department perspective, the networks involved in the testing and evaluation of software were divided into two categories: technical and non-technical factors. The networks that focused on technical factors were the IT steering committee, a team of project stakeholders, a project management office (PMO), and a software development team.

The IT steering committee consisted of head of departments. The stakeholders were employees, which include product owners, project managers, IT managers and business managers. The PMO consisted of project managers within the organisation. Software developers, software project managers and software testers formed the software development team. Each of these teams or committees had roles, tasks assigned to them by the focal actors (software development manager) for reaching a common goal, which was to test and evaluate software in the organisation.

5.3 Moments of translation: Problematisation

Mootledi Logistics relies on software for competitiveness and sustainability as software assists the employees to do their day-to-day activities as well as serving the organisations' customers. For this reason, any software that was developed within the organisation needed to be tested. Whenever a need arose to develop the new software, a certain process was followed within Mootledi Logistics: to log the request with the IT department through the change management process. The change management process was a platform used to decide whether the request made business sense. The project management team was responsible for evaluating the request and reported the outcome to the IT steering committee. After this the IT steering committee's responsibility was to approve the requests based on whether it made business sense.

Those who were responsible for conducting software testing and evaluation within Mootledi Logistics included software developers, business analysts and business users. It was vital for them to properly and rigorously test the software before it could be deployed to production for use. It is always better and less expensive to discover faults while the software is still in the testing environment (because they could be fixed) rather than when it is in the production environment. Faults discovered by business users in the production environment can have dire consequences on the image of the organisation. One participant stated that: "*With proper software testing we minimise faults coming back and forth which saves a lot of time and it enables us to deliver quality software*" (ML09, 46:1665-1666).

Discussions regarding software testing and evaluation took place at the redline meetings that occurred within the organisation. Redline meetings focuses on taking critical decision such as adoption of solution. The meetings are attended only by managers of teams within the organisation. The meeting is on regular basis, which also helps to improve the relationship between managers and their subordinates including business and IT departments. It was an ongoing process of improving how things were done within the organisation, enabling all parties involved in the testing and evaluation of software to be on the same page (to have the same understanding). It also assisted managers in providing more information about the requests they have logged to avoid rolling out software that remained within the business without being used. The IT operations manager further stated that: "*I think the link between business and IT department is being defined much better than what it was before, and it would encourage the better output within the IT department*" (ML02, 19:0699-0701).

Lack of processes within the organisation made it impossible to deliver the requested software. Previously the organisation had no proper quality assurance processes in place and the employees did not conduct thorough testing of the software developed. Poor analysis as well as no proper functional requirements specifications negatively affected the testing and evaluation of software. The software development and support manager highlighted that: "*We had no process previously and it was a question of let's test this software a little bit*" (ML01, 13:0454-0455).

5.4 Moments of translation: Interessement

Software testing and evaluation attracted the attention of various actors within the organisation whose interest was either of a voluntary or obligatory nature or both. The actors included individuals and groups from management to operational level such as the chief executive officer (CEO), chief information officer (CIO), IT steering committee, project management team, software development team, business analysis team as well as business users. These actors had various roles and responsibilities in the testing and evaluation of software. Some were indirectly linked, and others directly linked to the software testing activities. The CIO, for example, was indirectly linked to software testing activities because he only sponsored the approved requests: "*A sponsor being someone at the executive level who values the project and agrees to provide resources to execute the project*" (ML01, 07:0257-0258).

The actors had diverse interests in the testing and evaluation of software within the organisation. Their interests were influenced by various factors. From a management perspective, they were expected to ensure that software was in place to enable business users from various departments to perform their day-to-day duties, to serve customers and to provide customers with self-service software. Those departments included finance, risk and fleet department even to the point of Mootledi Logistics' customers. "*The end users can be as broad as the public (customers), anyone who can go to our web site and reserve a vehicle from the web site*" (ML01, 03:0079-0080).

The interest of teams and individuals was triggered by lack of quality assurance processes and standards especially for conducting software testing and evaluation within the organisation. Other teams, such as software development and business analysis, had their own standards which they followed when doing their tasks. When software had to be tested and evaluated, there was no software testing standard that was followed. The software developers, business analysts and business users shared the responsibility of testing and evaluating software. The various team standards were followed to conduct software testing: "*The software developer's standard may not be exactly the same as the tester standard or* *the business analyst standard because they may not know what exactly the expectations are*" (ML07, 45:1395-1396).

Some employees were interested in the software testing and evaluation, because they were bound by the performance contract signed with their line managers on behalf of the organisation. These performance contracts were used to assess the performance of individual employees and teams concerning effectiveness and productivity, assisting managers to identify employee strengths and weakness. As a result, relevant training was provided to develop those who lacked in skill. Performance contracts were also used to determine whether actors were eligible for salary increases or bonuses. Actors were expected to perform their duties as agreed with their managers: "*It is part of our annual employee performance*" (ML01, 11:0381).

While several efforts were made by managers to stir interest for the proposed solution to the problematisation of software testing and evaluation, the building of interest among the employees could not be regarded a success. Not all actors who were interested participated in the testing and evaluation of software. It is for this reason that the enrolment of employees was not completely successful as some employees were not willing to take part in the testing of software and evaluation.

5.5 Moments of translation: Enrolment

Participation of actors was pursued through different means in the organisation. The managers of various teams engaged with their subordinates and negotiated their participation in the testing and evaluation of software. Redline meetings were held as the negotiation platform to get those interested to enrol in the network that was going to test the requested software. Software testing couldn't exist in isolation; various teams, including project management, business analysis, software developers as well as business departments combined their skills to play a vital role. The managers of these teams used the power bestowed on them by the organisation to entice employees to participate and accept the roles and responsibilities allocated to them during the testing and evaluation of software: "*The scrum master should be someone who can liaise with business, product owner, software developers and business analysts*" (ML14, 68:2523-2524).

Even though there was no dedicated software testing team within Mootledi Logistics, some kind of software testing was conducted. Software developers conducted unit testing and end-to-end testing, business analysts conducted functional testing and business users conducted user acceptance testing (UAT). This was to ensure that when the software was handed over to the business department for use, it functioned as expected. The responsibility of the IT department was to enable the business department to execute its day-to-day functions of serving customers: "*Currently we don't have software testing teams, so testing is conducted by software developers, business analysts, business users and even the production support personnel*" (ML13, 63:2302-2303).

As stated earlier in the previous stage, interest from employees was either of a voluntary or obligatory nature, or both. The roles and responsibilities allocated to them during the development, testing and evaluation of software. These managers used their managerial discretion to enrol the employees in the network responsible for testing and evaluating software: "*The managers from different teams got to decide who they wanted to place on the project based on their skills*" (ML10, 47:1719-1720).

The IT managers ensured that they enrol competent, skilled and dedicated employees who would fulfil their roles in the testing and evaluation of software. Mootledi Logistics relied on software for competitiveness and sustainability. Therefore, new software was developed, and existing software upgraded when a need arose. It was important for managers to have strong teams to fulfil these needs: "*There is diverse software that requires testing within our organisation, therefore, we conduct different types of testing in order to test and evaluate them*" (ML01, 01:0010-0011).

Without quality assurance processes in place, it was difficult for those who were testing and evaluating software to deliver quality software. Previously, the organisation was following the traditional methodologies, such as waterfall, to develop, test and evaluate software. They used to encounter challenges such as changes in requirements, business users not knowing exactly what they want, software taking too long to be completed and late inclusion of business users in the software development life cycle. Re-examining user requirements and re-developing the software cost the organisation both time and money. Now the organisation has moved into agile software development such as scrum to improve on challenges which they previously faced: "*Traditional methodologies are not as flexible as agile because with waterfall if a requirement is missed or has changed you have to wait until the software is deployed to production and then log a change request*" (ML10, 50:1832-1835).

It is through functional software that Mootledi Logistics is able to function and serve its customers. The IT department's responsibility is first and foremost to deliver quality functional software. Mootledi Logistics seems not to be concerned much about performance testing. Performance testing is another important aspect of software testing and evaluation for determining how the software performs in terms of responsiveness and stability under a particular workload. There were no tools within the organisation that enabled the software developers to check the performance. Performance testing provides confidence that when the software is deployed to production it would be able to respond well under strenuous load on the network. The software developers were improvising the performance element within the code they were writing when developing the software: "*I take much time to actually make sure that the efficiency in the processing is there so that the code I am writing will not slow the software*" (ML12, 58:2098-2100).

5.6 Moments of translation: Mobilisation

Pursuance of employees was carried out along the structure and channels as defined by the organisation. In accordance with the organisational structure the product owner requested for the software, the IT steering committee decided on whether the software made business sense, and when approved, it was handed over to the IT department to develop, test and evaluate the software: "*Business owner is someone who has requested for the software*" (ML14, 69:2524-2525).

The CEO sponsored the entire initiative of developing, testing and evaluating the software. This software was going to be used by business users to serve customers and perform their day-to-day duties. The managers of various teams persuaded their subordinates to be part of the network to deliver quality software. The scrum master mobilised teams by ensuring that they delivered accordingly and also reported progress to the business owner: "*The scrum master and the team held sprint meetings to plan and communicate about the activities that needs to be done*" (ML13, 64:2330-2331).

Mobilisation occurred through various platforms, including redline meetings as well as scrum meetings. Redline meetings used to be attended only by managers to discuss progress and issues within the project. Business users were formerly excluded but now they are included in those meetings because managers were unable to provide some crucial information regarding projects. Due to agile adoption, now there are scrum meetings attended by the members of the same team to discuss challenges and progress: "*During the scrum meeting everyone has to report about what they have done and what they are going to do*" (ML14, 69:2518-2519).

It was always helpful to provide feedback to the product owner in order to know the progress about the software requested. Such feedback was also critical for project sponsors because no one wanted to waste the organisation's time and money on something that was not achievable or something that was failing. Therefore, feedback persuaded all the actors to do their best to deliver the tested and evaluated software. At the end of the day, the requested software had to be signed off and accepted by business owners and business users: "*Then there was UAT whereby the business user runs through the testing in order to accept the softwar*e" (ML01, 06:0211-0212).

Therefore, product owners had to mobilise their teams to make use of the software that had been thoroughly tested and evaluated. The business team became the focal actor because the testing and evaluation of the software was initially instigated by them. The management was encouraged by the task of mobilisation which was linked to their performance appraisals. Mobilisation was successful, and employees were excited about the contribution in the testing and evaluation of software.

6. DIFFUSION OF INNOVATION: INNOVATION DECISION PROCESS

The innovation decision includes knowledge, persuasion, decision, implementation and confirmation (Sang &Tsai 2009:1877).

6.1 Innovation decision process: knowledge

The IT department had the enormous responsibility of delivering the software requested by the business department. The delivery of this initiative (software) was new within the organisation; therefore, the IT department, which included project managers, business analysts, software developers and business intelligence personnel, needed to fully understand the business requirements to best deliver the tested, evaluated and functional software. The business analysts had the enormous responsibility of gathering the business requirements which would establish the development, testing and evaluation parameters for the requested software: "*The business analyst is the person who documented the business requirements the individual who have a clearer idea of how the software should function*" (ML10, 49:1786-1787).

The business requirement specification, or functional requirement specification, provided absolute knowledge about the type of business that was required. This does not necessarily mean that the business requirements would be accurate at the first attempt. Sometimes businesses were uncertain as to what exactly they wanted. In such cases, the business analyst was able to assist as this person has both business and technical knowledge. There were also possibilities that some business requirements could be missed during the business requirement gathering process which would hinder the development and testing as well as the evaluation of software: "*If the software developer does not understand the business requirements they would automatically build incorrect software*" (ML14, 68:2490-2491).

The comprehension of the requested software was very important to the software development team as it enabled the team to develop, test and evaluate the software. It was important for everyone involved in the software development life cycle to have a clear and well-defined understanding of what a specific business wanted. The software development life cycle process is a collaborated initiative that requires combined skills from various fields of specialisation such as business analysis, software development and software testing. For example, software testing cannot occur if the software does not exist; without business requirements software developers could not develop the software. Knowledge gained from business activities assisted business analysts to gather and document business requirements. As a result of the compiled skills and knowledge, the software development team was able to develop, test and evaluate the requested software: "*Business knowledge was also important to decide whether the test case was sufficient to actually test against the functional requirement*" (ML01, 13:0445-0446).

It was the responsibility of the software development team to seek clarity on vague business requirements in order to develop the correct software, as it is a challenge to develop and test software with ambiguous business requirements. Some of the software developers and software testers had to be innovative in clarifying vague business requirements in order to do precisely what was expected from them. There were instances, for example, whereby software developers were issued three pages of business requirement specifications which were not clear. As a result, they had to liaise with business analysts as well as business users to find the necessary clarity. One of the software developers stated that: "*I would rather sit with the business users to get clarity, liaise with the business analysts and then start developing the software because jumping straight into software development without understanding what was required leads to back and forth which wastes time*" (ML14, 69:2502-2504).

6.2 Innovation decision process: Persuasion

The IT department was somewhat unfamiliar with the software that the business department had requested. The IT department created team of software developers who were primarily responsible for developing, testing and evaluating software on behalf of the organisation. This team included a project manager, business analyst, software developers and business intelligence personnel: "*Since our organisation has decided to go the agile route, the project manager is assigned the dedicated team of five individuals that includes a business analyst, two Microsoft developers, the business intelligent personnel and the K2 developer*" (ML10, 47:1721-1723).

This team was interested in the innovation (requested software) and it actively looked for related information in order to develop, test and evaluate this software. Each individual had to actively play his role so that the team could succeed in delivering quality software. The business analysts liaised with business users in order to find out what they really wanted in order to document the business requirements. The entire software development team (scrum) was dependent on the business requirement specification to perform their duties.

Failure in working as a team would negatively impact the output as well as delay the delivery timelines of the IT department. Consequently, the product owner as well product sponsors would be dissatisfied with the IT department delivering the software later than the agreed timelines. The business department only expected what they have requested on the agreed timelines. They wanted software that would make their lives easier in terms of performing their duties and servicing customers: "*With proper software testing we minimise faults coming back and forth which save a lot of time, money and it enables the IT department to deliver quality software*" (ML09, 46:1665-1666).

6.3 Innovation decision process: Decision

The business users were required to conduct UAT before the software could be promoted or deployed to the production environment. The UAT is the final stage of the software testing process, the process whereby business users test the software to ensure that it can handle required tasks in real-world scenarios, according to specifications: "*After the actual software testing the business analyst goes through the whole process with the business user to the point whereby the business user accepts the solution, which we call UAT*" (ML05, 28:1026-1028).

The UAT is preceded by system integration testing (SIT) which helps the software development team to verify whether the subsystems constituting the software solution work as expected and cooperate in a streamlined manner. A variety of software testing needed to be conducted for delivering software that met business requirements. During software development, software developers conducted unit testing which was termed developer testing at Mootledi Logistics. Unit testing is a software development process in which the smallest testable parts of the software, called units, are individually and independently tested for proper operation. This type of testing increased the confidence of software developers that when handing over the software to the business analysts and business users to test, it was fully functional. This was unfortunately not always the case at Mootledi Logistics: "*The biggest challenge is that the software developers don't test the software properly the first time because they are expected to test it before deploying it to the testing environment but when the software is handed over to the business analysts to test it just keeps on failing*" (ML07, 37:1354-1356).

The organisation had a definite need to improve how software was developed, tested and evaluated. Previously, the organisation followed traditional methodologies such as waterfall to develop software. Some participants stated that the waterfall methodology was too rigid, which caused delay in the gathering and compilation of the requirements specification. This is was because business users were involved too late in the software development process. As a result, business requirements changed while the software was being developed. Also software developers, software testers, business intelligence personnel and support personnel had to wait for the requirements specification to be completed before performing their duties. As a result, by the time the software development was completed, the technology had already changed or advanced. The one software developer asserted that: "*The biggest issues in technology is that by the time you are done with the waterfall project the business rule has changed so what you are delivering doesn't apply anymore*" (ML12, 59:2168-2170).

Due to the above waterfall weaknesses, the organisation decided to adopt scrum agile methodology. Risks associated with scrum include losing a team member without transferring knowledge, reprioritization of product backlog, which may result in termination of one or more sprints. This sometimes result in loss of time or a particular technology not working sufficiently which may force the team to consider alternative plan. Scrum is designed for teams of three to nine software developers who divide their work into actions that can be completed within sprints. The scrum team holds daily scrums (stand-up meetings) to report on progress, issues and the way forward. With agile methodology, business users were involved from the beginning of the project because they needed to verify all the functionalities they had requested. This was to avoid late requirements changes. The changing of software development methodologies as well as the use of software testing tools was aimed at improving software development, software testing and its evaluation. This was the step in the right direction in terms of delivering quality software.

6.4 Innovation decision process: Implementation

The software gets deployed to production once the business users and the product owner had accepted and signed it off. The software developers prepared the installation manual guide to assist the production support team in deploying the software. The production support team configured the production environment to accommodate the new software. Once the software was deployed, it was ready for use.

The process was that after deployment, the software is tested to ensure it fulfils the business users' requirements. The responsibility of the production support team was to ensure that the new software was always available for use and functional at all times. They also maintained the existing and new software, assisted business users with technical production matters, carried out necessary research and provide in-depth analysis for resolving production issues. They were also expected to know and understand how all the software they maintained functioned. One business analyst stated that: "*At the end of the day the production support needs to support the software*" (ML07, 39:1418).

The manager who requested software for their department had to use the power bestowed on him to encourage staff to actually use the new software. Staff needed to understand, that the software was meant to simplify the day-to-day duties as well render services more efficiently to Mootledi Logistics customers. At Mootledi Logistics, any problems arising within the department needed to be attended to as soon as possible to ensure the efficient running of the department. Unresolved problems within the department would certainly have a negative impact on the organisation itself. Every department within the organisation has to function effectively for the entire organisation to be competitive.

6.5 Innovation decision process: Confirmation

At this stage, it is critical to recognise and appreciate the benefits of using the innovation (new software), making the software part of the ongoing routine and promoting it to others within the department. This gives the department the opportunity to function more effectively. Business users are able to perform their daily duties using the new software as well as rendering the service to customers. One participant highlighted that: "*Some of the software at our rental counters are used by our staff to serve the customers*" (ML01, 03:0077-0078).

Some of this software, such as the organisations' web site, enables customers to serve themselves. Customers are able to make bookings and reserve cars through the website, at their personal convenience, without having to be physically present at the Mootledi Logistics premises. The organisation had to develop efficient software that would enable it to maximise profits. Another participant asserted that: "*The organisation must ensure that the software it implements saves time and make the work environment friendlier*" (ML06, 34:1247-1248).

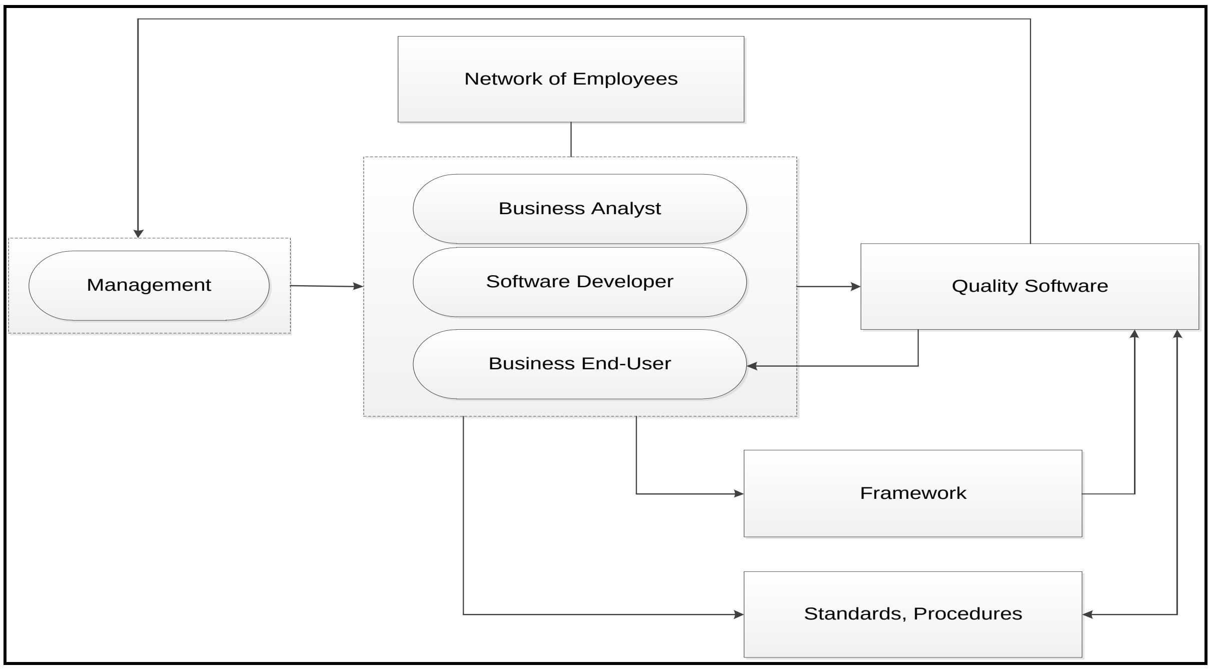
Business user output indicated that the organisation made the right decision by implementing this new software. The organisation was heading in the right direction by introducing this new software which enhanced efficiency of services. The IT operations manager confessed that the organisation is not there yet but: "*I think we are heading in the right direction hopefully we would get there*" (ML02, 19:0682-0683).

7. FINDINGS AND DISCUSSION

Based on the analysis of the empirical data from Mootledi Logistics, five factors were found to have a critical influence on the software testing and evaluation within the organisation: 1) lack of testing framework; 2) lack of management buy-in; 3) network of employees; 4) quality of software; and 5) lack of standards and procedures. Figure 1 depicts these factors and their relation to each other.

These factors are empirically revealed to influence testing and evaluation of software in Mootledi Logistics. The factors can also be used by organisation similar settings or challenges to guide testing and evaluation of software in their environments. These factors are discussed below.

Figure 1: Factors influencing software testing and evaluation



Source: Developed by the authors

6.1 Lack of testing framework

A framework can be regarded as a helpful structural tool that is often used to guide scope, boundaries and procedural activities. At Mootledi Logistics, however, there was no framework adopted and used for testing and evaluation of software. As a result, software was tested by different employees, such as a software developer, business analyst, software tester or end user, at various stages of testing. This was as a result of two factors: (1) individuals were selected to carry out testing based on favouritism, who the managers were more comfortable with, rather than on merit; and (2) employees were selected or rotated based on their availability.

Some of the implications arising from the lack of testing framework were as follows: (1) there was no handover from one employee to another for smooth continuance of the process until completion; (2) there were inconsistencies in how testing was conducted, which detrimentally impacted the quality of some software in the organisation; (3) software testing and evaluation sometimes took longer to complete because employees might have been testing the same scenarios over and over without realising that these were tested before; (5) defects in software were hardly traced or tracked; and finally, (5) some crucial scenarios were missed as a result of poor structure, improper processes and many employees testing the same software.

6.2 Lack of management involvement

Even though software was heavily relied upon at Mootledi Logistics for business processes and services, management buy-in was a challenge. There was no investment made in the testing and evaluation of software from the management perspective. For example, instead of purchasing proprietary software, they resorted to adopting open source software. This type of software does not require licensing for use and can be accessed for free. While the software development team trained themselves on how to use those tools, this sometimes had a negative impact on software testing timelines. The time spent, learning how to use the tools could have been utilised for actual software testing and evaluating activities.

Also, the organisation did not have a dedicated software testing team to perform testing and evaluation activities of software that were developed within the organisation. Software testing is a specialised field and requires a dedicated team. Software testers have to be trained and equipped with the necessary skills for testing various types of software, and at any given time. The lack of management involvement and weak commitment did not entice many of the employees to commit to the concept of testing either. This frequently impacted the quality of software that was developed and managed in the organisation.

6.3 Network of employees

As in many environments, networks were consciously and sometimes unconsciously formed within Mootledi Logistics. During testing and evaluation of software, many of the employees worked along the organisational structure, an indication of consciousness. Some of the employees, specifically software developers, were uneasy with the formal approach, and rather preferred their individual approaches. The uneasiness influenced some of the employees to form a network in carrying out testing and evaluation of software in the environment.

Rather than following the organisational structure, some employees identified themselves through interaction during the testing and evaluation of software. As result, some employees felt comfortable only working with certain other employees. After development, the software is hand-over to the business analyst or business end-user with whom they felt comfortable working with, for testing and evaluating.

That became the normative practice by some employees to determine who would test which software. The implication of performing testing and evaluation of software in that fashion compromised the quality of software. If the business analyst or business end user was a friend to a software developer, they would be lenient in testing and evaluating the software, thereby imploring favouritism. As a result, software was not always properly or thoroughly tested. Those who were strict and thorough in testing and evaluating software were often bypassed or avoided because they were capable of detecting defects, which some software developers were uncomfortable with, in that it exposed their inability to produce quality work.

6.4 Quality of software

While the organisation advocated quality at all times, this did not necessarily and often reflect on the software produced in the environment. Quality of software was challenged in that there were no real or formal criteria or requirements that could be used to guide the testing. The lack of framework and weak commitment from management negatively affected the quality of software. Employees needed to be guided by some kind of framework to deliver quality software. Without such a framework, employees wouldn’t be aware of what to cover when testing and evaluating software. The management support regarding software testing and evaluation was critical in the delivery of quality software. Management needed to invest money in software testing by establishing a dedicated software testing team, training software testers and purchasing proper software testing tools. The dedicated software testing team would have assisted the software development team in delivering quality software. The lack of commitment from management impacted the software quality.

Poor quality software stirred dissatisfaction from the customers. Customer couldn’t receive the products and services rendered to them, services included booking of vehicles online. As a result, business end users as well as customers couldn’t perform their duties because of dysfunctional software. While organisations believe in keeping their customers happy at all times, with poor quality software, it is impossible to achieve that.

6.5 Lack of standards and procedures

Standards and procedures set the criteria that can be used in the selection of software testing methods. There were various types of software testing methods that can be followed in conducting software testing and evaluation, including black box and white box testing. Black box testing is conducted purely based on the requirement specification knowledge. White box testing is conducted when the software tester has exceptional knowledge about the software as opposed to its functionality. Software testers are provided access to the code because they could perform grey box testing, which is conducted when the software tester has the limited knowledge of the software.

Standards and procedures also assist in selecting software testing tools and how to utilise those tools. It is a wasteful expenditure to purchase software testing tools and not know how to use them.

8. CONCLUSION

Software testing is the most essential part of systems development life cycle as it helps determine whether the software is of quality or not. Quality software is reasonably bug or defect free, delivered on time and within budget, meets requirements and expectations and is maintainable. ISO 8402-1986 standard defines quality as "the totality of features and characteristics of a product or service that bears its ability to satisfy stated or implied needs". With quality software, organisations are able to function and provide necessary services to its clients. Many organisations as well as Mootledi Logistics are faced with various challenges including lack of testing framework, management not showing interest in the testing and evaluation of software, lack of software testing standards and procedures which results in delivering poor quality software. As a result, the organisation is not able to provide necessary services to clients. For this reason, it was important to understand the factors influencing the testing and evaluation of software in order to deliver quality software. There was a need for a framework to guide the organisation in making informed decisions about testing the software as well as eradicating challenges that are encountered during software testing. Failure in recognising that, could likely result in organisations implementing poor quality software at all times. This article will add value to software developers, software testers, software managers and academics.

Poor quality of software has implications for management of the organisation. Some of the implications are as follows: (1) business is interrupted unnecessarily, which negatively affects the image of the organisation; (2) it is the responsibility of the management team to ensure that employees are trained and equipped with software testing skills as well as knowledge for using those tools; and (3) although software testing tools can be expensive but they can as well save excessive time and money in the long run. The influencing factors guide and enable employees to create test requirements, test cases, and execute test cases appropriately in increasing quality.

REFERENCES

ABBAS R, SULTAN Z & BHATTI SN. 2017. Comparative analysis of automated load testing tools: Apache JMeter, Microsoft Visual Studio (TFS), LoadRunner, Siege. *Communication Technologies (ComTech)* 39-44. (DOI:10.1109/COMTECH.2017.8065747.)

ALSHENQEETI H. 2014. Interviewing as a data collection method: A critical review. *English Linguistics Research* 3(1):39-45. (DOI:https://doi.org/10.5430/elr.v3n1p39.)

AMLANI RD. 2012. Advantages and Limitations of Different SDLC Models. *International Journal of Computer Applications & Information Technology* 1(3):6-11. [Internet:http://www.ijcait.com/IJCAIT/13/1334.pdf; downloaded on 26 May 2015.]

BAMOTRA A & RANDHAWA AK. 2017. Software Testing Techniques. *International Journal of Innovative Computer Science & Engineering* 4(3):122-126. [Internet:https://www.citefactor.org/search/keywords/articles/Software+Testing+Techniques; downloaded on 07 November 2017.]

BASSIL Y. 2012. A Simulation Model for the Waterfall Software Development Life Cycle. *International Journal of Engineering & Technology* 2(5):742-749. [Internet:https://arxiv.org/pdf/1205.6904.pdf; downloaded on 26 May 2015.]

BHARDWAJ S. 2015. Performance Testing Tools: A Comparative Analysis. *International Journal of Engineering Technology Management and Applied Sciences* 3(4):100-105. [Internet:https://pdfs.semanticscholar.org/aca4/b3bf1f9213a65e5c4f23637df5de495d478d.pdf; downloaded on 20 February 2016.]

BUKHARI A, FAISAL S & HIRA K. 2014. A comparative study on usage of traditional and agile software development methodologies in software industry of Asia. Athens. (International Conference on Software Engineering Research and Practice (SERP); 1-8.)

CALLON M. 1986. Some Elements of a Sociology of Translation: Domestication of the Scallops and the Fishermen of St Brieuc Bay. In Law J. Ed. Power, Action and Belief: A New Sociology of Knowledge. London: Routledge. (pp 196-232.)

CHANG H. 2010. A New Perspective on Twitter Hashtag Use: Diffusion of Innovation Theory. *Proceedings of the American Society for Information Science and Technology* 47(1):1-4. (DOI:10.1002/meet.14504701295.)

DHIMAN S & SHARMA P. 2016. Performance Testing: A Comparative Study and Analysis of Web Service Testing Tools. *International Journal of Computer Science and Mobile Computing* 5(6):507-512. [Internet:https://pdfs.semanticscholar.org/aca4/b3bf1f9213a65e5c4f23637df5de495d478d.pdf; downloaded on 07 November 2017.]

EFFAH J. 2012. Mobilizing Culture for E-Business in Developing Countries: An Actor Network Theory Account. *The Electronic Journal on Information Systems in Developing Countries* 52(5):1-18. (DOI:https://doi.org/10.1002/j.1681-4835.2012.tb00370.x.)

FAROOQ SK & QUADRI SMK. 2013. Empirical Evaluation of Software Testing Techniques - Need, Issues and Mitigation. *Software Engineering: An International Journal (SEIJ)* 3(1):41-51. [Internet:http://seij.dtu.ac.in/vol-3/issue-1/paper4.pdf; downloaded on 20 February 2016.]

GAUTAM S & NAGPAL B. 2016. Descriptive Study of Software Testing & Testing Tools. *International Journal of Innovative Research in Computer and Communication Engineering* 4(6):10288-10295. (DOI:10.15680/IJIRCCE.2016. 0406006.)

HENDERSON S. 2016. Research Methodology. *International Journal of Sales, Retailing and Marketing* 4(9):1-97. [Internet:http://www.ijsrm.com/ijsrm/Current_&_Past_Issues_files/IJSRM4-9.pdf; downloaded on 07 November 2017.]

HOLLWECK T. 2015. In Robert K. Yin. (2014). Case Study Research Design and Methods . Thousand Oaks, CA: Sage. 282 pages. *Canadian Journal of Program Evaluation* 30(1):108-110. (DOI:10.3138/cjpe.30.1.108.)

HOODA I & CHHILLAR RS. 2015. Software Test Process, Testing Types and Techniques. *International Journal of Computer Applications* 111(13):10-14. [Internet:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.695.1299&rep=rep1&type=pdf; downloaded on 20 February 2016.]

IYAMU T, SEKGWELEO T & MKHOMAZI SS. 2013. Actor Network Theory in Interpretative Research Approach. *IFIP International Federation for Information Processing* 605-610. (DOI:https://doi.org/10.1007/978-3-642-38862-0_41.)

JAMIL MA, ARIF M, ABUBAKAR NSA & AHMAD A. 2016. Software Testing Techniques: A Literature Review. (6th International Conference on Information and Communication Technology for the Muslim World). (pp 177-182). (DOI:https://doi.org/10.1109/ICT4M.2016.045.)

KASURINEN J. 2012. Software Organizations and Test Process Development. *Advances in Computers* 85:1-63. (DOI:https://doi.org/10.1016/B978-0-12-396526-4.00001-1.)

LINSLEY P, KANE R & BARKER JH. 2019. Evidence-based practice for nurses and healthcare professionals. Thousand Oaks, California: Sage Publications Inc. (pp 91-154.)

MISHRA D, OSTROVSKA S & HACALOGLU T. 2017. Exploring and expanding students' success in software testing. *Information Technology & People* 30(4):927-945. (DOI:https://www.researchgate.net/deref/http%3A%2F%2Fdx.doi.org%2F10.1108%2FITP-06-2016-0129.)

MISHRA S & PRADHAN A. 2012. Software Testing Techniques Adopted in Corporate Sectors: A Precise Study. *International Journal of Emerging Technology and Advanced Engineering* 2(9):290-295. [Internet:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.413.5837&rep=rep1&type=pdf; downloaded on 20 February 2016.]

MUNASSAR NM & GOVARDHAN A. 2010. A Comparison between Five Models of Software Engineering. *IJCSI International Journal of Computer Science Issues* 7(5) 94-101. [Internet:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.402.8120&rep=rep1&type=pdf#page=115; downloaded on 20 February 2016.]

NIDHRA S & DONDETI J. 2012. Black box and white box techniques - A literature review. *International Journal of Embedded Systems and Applications* 2(2):29-50. (DOI:https://www.researchgate.net/deref/http%3A%2F%2Fdx.doi.org%2F10.5121%2Fijesa.2012.2204.)

OVERHAGE S & SCHLAUDERER S. 2012. Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects. (45th Hawaii International Conference on System Sciences). (5452-5461.) (DOI:https://doi.org/10.1109/HICSS.2012.387.)

QAZI AM, SHAHZADI S & HUMAYUN M. 2016. A Comparative Study of Software Inspection Techniques for Quality Perspective. *International Journal of Modern Education and Computer Science* 8(10):9-16. (DOI:10.5815/ijmecs.2016.10.02.)

ROGERS EM. 1995. Diffusion of innovations. 4th ed. New York: Free Press. (pp 15-23.)

SALEH MF. 2011. An Agile Software Development Framework. *International Journal of Software Engineering (IJSE)* 2(5):97-106. [Internet:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.740.6970&rep=rep1&type=pdf; downloaded on 20 February 2016.]

SANG HA & TSAI DR. 2009. Analyzing Strategies of Integrating ICT into Teaching Activities Using Innovation Diffusion Theory. (5th International Joint Conference on INC, IMS and IDC. (pp 1876 -1878.) (DOI:https://doi.org/10.1109/NCM.2009.323.)

SEKGWELEO T. 2015. Understanding Traditional Systems Development Methodologies. *International Journal of Advances in Management and Economics* 4(3):51-58. [Internet:https://www.managementjournal.info/index.php/IJAME/article/view/370; downloaded on 20 February 2016.]

SENER Z & KARSAK EE. 2012. A decision model for setting target levels in software quality function deployment to respond to rapidly changing customer needs. *Concurrent Engineering Research and Applications* 20(1):19-29. (DOI:https://doi.org/10.1177%2F1063293X11435344.)

SINGH I & TARIKA B. 2014. Comparative Analysis of Open Source Automated Software Testing Tools: Selenium, Sikuli and Watir. *International Journal of Information & Computation Technology* 4(15):1507-1518. (DOI:10.13140/2.1.1418.4324.)

SKIDMORE S. 2006. The V-model. *Professional Scheme Paper* 2:48-49. [Internet:https://pdfslide.net/documents/automatisert-testing-av-dynamisk-html-idintnuno-automatisert-testing-av.html; downloaded on 26 May 2015.]

SOWUNMI OY, MISRA S, FERNANDEZ-SANZ L, CRAWFORD B & SOTO R. 2016. An empirical evaluation of software quality assurance practices and challenges in a developing country: a comparison of Nigeria and Turkey. *SpringerPlus* 5(1):1-13. (DOI:https://doi.org/10.1186/s40064-016-3575-5.)