

BAM Submission

A J Bridger

User Acceptance Testing and Training– an unrecognised contributor to SaaS ERP technology adoption.

Abstract

The Software as a Service (SaaS) technology revolution is now well underway. An increasing number of organisations are now moving to vendor fully hosted and supported ES and ERP (Enterprise Resource Planning) software SaaS applications to potentially help reduce software costs. These platforms offer considerable advantages to organisations through improved upgrade and version management, reliability and reduced inhouse management costs - and the potential to customise if required. However, the deployment of these platforms often invokes a major shift in organisational technical and commercial management capability – and subsequent issues with adoption. This study addresses a specific question of whether structured UAT or User Acceptance Testing, given the changes in technological delivery, has the potential to be a key element in improved user adoption. The results of an interpretive semi-structured qualitative analysis study suggests that a key organisational training, technical and commercial skill is in the management and execution of user acceptance testing (UAT) of these applications. This study reveals that many organisations have failed to grasp the contractual, technical and employee skillset risks needed to both accept and continually support third party SaaS software applications into operational use. Lack of precise testing, regression testing and validation of changes or customisation into the application may often induce errors, rework, project delays and cost overruns. This study concludes that organisational commercial SaaS application management requires User Acceptance Testing as a key organisational skill that is now the interface point to vendor software product acceptance and contract compliance – and should be identified as a key pre-project variable.

Introduction

Statista (2021) estimated that the SaaS software market, by 2024 will be valued at some \$462bn USD. However, The Standish Group, in their often-cited Standish Group Chaos Report (2018), suggest that, in general, around 16% of software projects and implementations will deliver their forecast benefits, 30% of projects will be abandoned and the remainder, 54%, will experience a range of difficulties that will impede the quality of delivery and value of benefits or are “challenged”. This study aims to contribute to research that supports organisations and helps improve the testing and evaluation of software into operational environments and improve project delivery performance.

The recognition of the need for software testing, and specifically User Acceptance Testing, is not a new concept. In many respects, all software development from the early 1990’s onwards will have allocated a project phase to some form of acceptance testing and is therefore not software specific. The early evolution of a range of software development methodologies like SDLC or Software Development Lifecycle, V-Test methodology and Agile all support testing regime rigour. However, testing covers a wide range of test types; software developers will often conduct testing of their own code via unit and functional tests; software teams will also execute integration testing and end to end tests. Mitigation of both software bug risk and elimination has also been enhanced using automated test tools (Sommerville, 2016).

However, once functional code has been lodged into the test environment, software vendors will often signal that user acceptance testing can commence. User acceptance testing is defined broadly as “The stage in which users or customers provide input on system testing...alternatively it may be an informal process where users experiment with a new product to see if they like it and to check that it does what they need....User testing is essential even when comprehensive system and release testing have been carried out” (Sommerville, 2016, p.249).

Acceptance testing may also be combined with what is termed as regression testing or re-testing of additional code functions that may have been directly impacted by changes i.e., from upgrades or custom code. One key element of acceptance testing is in the use of destructive testing. Destructive testing is designed to allow users to “break” the application and usually involves inserting erroneous data into system fields in attempt to trigger errors and subsequently gives developers an understanding how the application manages the resulting problem or whether a fix is required.

Historically, particularly with inhouse managed technologies and systems, it may be posited that the divisions between accountabilities for acceptance testing may have become blurred and associated skills lost through staff turnover. As Cooke-Davis (2002) posits, previous project experiences are a key contributing factor to overall robust project delivery. However, the advent of SaaS applications has re-positioned the commercial boundary between organisational business areas and software vendors; as inhouse information technology departments may have limited roles in SaaS contracts, business areas become accountable and responsible for both the integrity, functioning and contractual management of the technology. The advent of smaller “pay as you go” SaaS technologies where customisation is not available means that software vendors take total responsibility for both test and function validation. This study suggest that the role, skills and techniques of user acceptance testing are now a key capability that organisations need to both acquire and develop.

Previous Research

Historical Research Perspective

The role of user acceptance in the implementation of information systems is long established as part of the SDLC or Systems Development Life Cycle at Figure A. UAT has also been described as a process “used to ensure the fitness of the system for use, which is conducted by the business users” (Padmini et al. 2016, p.1). Lobkov (2019) in a test focused study concludes that poor testing can negatively impact user perceptions once the system is live – and therefore this study suggests that, inductively, it may subsequently affect both user perceptions and adoption elements which are critical in delivering robust organisational outcomes and benefits. It is further suggested that “Software testing is the most important and the most critical step in the entire software development process, which at the same time consumes the most resources and requires huge time investments” (Lobkov, 2019, p.11).

This has been supplemented by a range of differing test models, for example, Vee Test, and in recent years, the development of Agile test approaches. The execution of user acceptance testing is normally conducted late in the project lifecycle and may be iterative if customisation or software change is involved. The test phase may also be iterative and

involve both destructive testing “is to ensure a software product exhibits proper behavior when subject to improper usage or improper input” (Akingbehin, 2006, p.3). Regression testing is defined as “the verification that previously functioning software remains after a change” (Agarwal, 2019, p.38). To that end, user acceptance testing, by default, if changes are made to the software during the project, requires both regression and destructive testing (Memon and Soffa, 2003). Sommerville (2016) described the process of User Acceptance Testing as part of the SDLC process at Figure A, and then in more detail at Figure B. Sommerville (2016) text on Software Engineering originated in the 1990s and has also focused the steps of Acceptance Testing at Figure B.

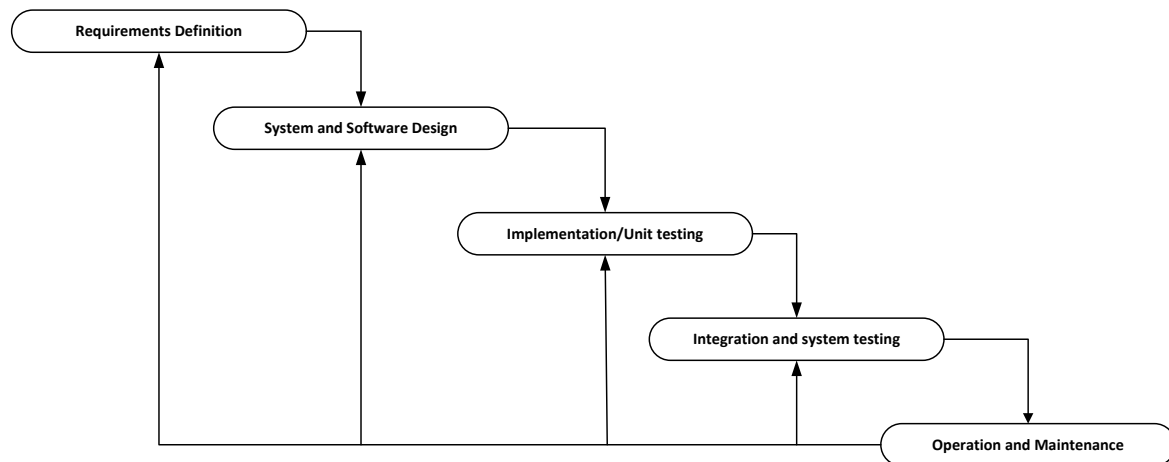


Figure A - Software Development Life Cycle (SDLC) or Waterfall approach
(adapted from Sommerville, 2016, p.47)

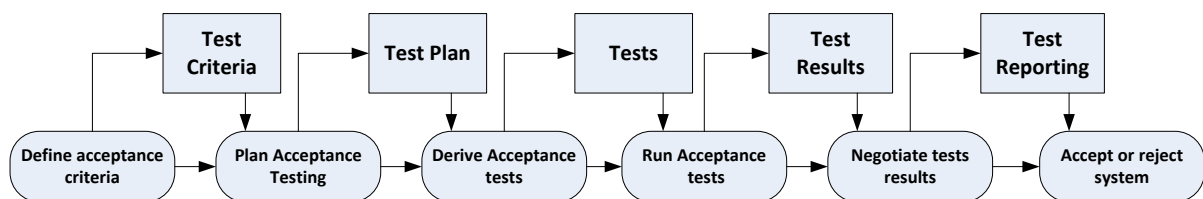


Figure B – User Acceptance Test process
(adapted from Somerville, 2016, p.250)

Melnik and Maurer (2005) suggest that documenting acceptance tests initially may help users define more accurate requirements. In effect, users should define what needs to be tested and therefore subsequently aids requirements development. However, like requirements definition, this may involve users replicating current function tests and may be the precursor to extensive customisation as opposed to utilisation of new package functionality (Bridger, 2021). Ricca et al. (2009) also takes the perspective that suggests acceptance tests may also be a means of evaluating or reaffirming requirements and aligns with one of the key criticisms of the SDLC, for example, that development is iterative and not linear in nature

(Figure A). Dos Santos et al. (2018) also conducted research on the use of user interaction diagrams to assist non-technical users create potentially more accurate requirements scenarios; however, their study sample was small and Agile development focused. Yamaura (1998), though an ageing paper, developed a practical approach to test case development; to that extent, considerable theoretical, practical and practitioner research and education exists for organisations to structure user acceptance testing.

SaaS or Software as a Service – a different paradigm

With the advent of Software as a Service technologies or SaaS (Kumar, 2017), where the vendor supplies both hardware and software support services, the shift to functional business areas deploying systems without internal IT support often moves accountability to business units to ensure effective testing. The development of SaaS packages may, to some extent, have alleviated the need for extensive testing, particularly if the application provider allows little or no customisation. Two technical terms are important in this study, firstly, software package configuration “does not involve source code change of the SaaS application. It usually support variance through setting pre-defined parameters, or leveraging tools to change application functions within pre-defined scope, e.g., adding data fields, changing field names, modifying drop-down lists, adding buttons, and changing business rules, etc. Configuration can support tailoring requirements within pre-defined configurable limit” (Sun et al., 2008, p.19). Whereas customisation involves “source code changes to create functionality that is beyond the configurable limit... is a much more costly approach for both SaaS vendors and clients” (Sun et al. 2008, p.19).

Given the importance of acceptance testing to the operational integrity and often fiscal management of organisations, only limited research attention appears to have been paid to this project element. However, SaaS applications may incur considerable test overheads; Naganathan (2011) suggests “Therefore a comprehensive testing effort is required so that organizations can reap all the benefits of SaaS such as higher system availability, greater reliability, higher flexibility, scalability and enhanced levels of security. Organizations need to understand and appreciate the challenges associated in enabling this end-to-end testing that SaaS applications demand” (Naganathan, 2011, p.4). As an example, one of the most complex SaaS applications involves integrated manufacturing and accounting technologies and are therefore mission critical applications. Testing with these technologies is therefore a key risk reduction activity. The key difference between older “behind the firewall” and SaaS applications is that the key contractual touchpoint is that organisational acceptance is contractually formalised – and, subsequently, additional rectification work may be both chargeable and costly.

Acceptance testing of financial systems is a critical task in the delivery of an ERP (Enterprise Resource Planning) package particularly where customisation may be extensive on what are often the core accounting modules. However, in a review of Critical Success Factor or CSF research, focus on this element appears limited. In a review of fourteen papers on CSFs testing was listed in two papers as key CSF (Desalegn and Pettersson, 2018, Seres et. al, 2020). In a review of ERP CSF papers pre-2013, Shaul and Tauber (2013), testing is only mentioned briefly.

The importance of this activity is recognised (Dos Santos et al., 2018, Lobkov, 2019). However, Dos Santos et al. (2018) list some twenty-two techniques that may be used in the User Acceptance Test environment. For packaged applications in particular, techniques such

as Use Cases, Test Scenarios, Spreadsheets, Natural Language and User Scenarios may deployed (Dos Santos, 2018). Scripts are essentially the step-by-step actions that a tester will execute.

UAT effectiveness

Little detailed research is identifiable on the efficacy of organisational UAT. However, Padmini et al. (2016) summarises the key issues in UAT from their empirical research case study. These are at Figure C.

No.	Observations
1	UAT team does not have an understanding about testing levels and types, defect identification, defect logging, defect lifecycle, severity, and priority of use case scenarios and defects.
2	UAT Team is not clear about their roles and responsibilities.
3	Lack of knowledge in writing scenarios for UAT.
4	Conflict between project leadership and UAT team's objectives.
5	Lack of communication between project leadership and UAT team.
6	Fear for vendor and project leadership.
7	Some UAT entry criteria were not passed, though UAT started.
8	No proper communication between vendor and UAT team.
9	Necessary information not shared with the UAT team.
10	Limited presence of UAT team members at UAT premises.
11	UAT test scenarios are not get signed off by responsible body.
12	No proper documentation and naming convention practiced.
13	No proper configuration management used.
14	Lack of perception and attitude.

Figure C – Key issues in UAT Study (adapted from Padmini et al., 2016, p.5).

Early attempts at test automation research have been undertaken; Hanssen and Haugset (2009) conducted research into the use of FIT or Framework for Integrated Tests. However, their study identified that "Tests were written by the developers, no person had a dedicated responsibility for writing the acceptance tests and most noteworthy – no customers participated in expressing tests" (Hanssen and Haugset, 2009, p.7). However, their study reported fewer errors in the applications and suggested that their results demonstrated automated testing may hold significant future promise.

Given the advances in information technology since 2000, considerable effort has been placed on research into automation of User Acceptance Testing. Dos Santos et al. (2018) list these as Declarative Domain System Test Language, Test Programming Language and Domain Specific Language. Geetha et al (2021), in recognising the effort and overhead involved in UAT and any resulting Regression Testing suggest that "regression testing is a very expensive form of testing that includes re-execution of all test cases thus making the overall test suite more cumbersome and leading to an increase of test cases." (Geetha et al. 2021, p.20) and propose a combination of what are termed as multilevel walks and an optimisation algorithm. A study by Dalal et. Al, (2018) also recognises the overheads with regression testing.

There is also only a limited range of texts that may help organisations develop the skills to effectively conduct acceptance and subtest types. A search on the web for commercial UAT training programmes revealed a variety of available courses throughout the UK for organisations.

This study examines how effective participating organisations were in managing and testing new software - and includes both ERP and a range of other enterprise level applications.

Research Methodology

This research study adopted a qualitative, interpretivist methodology based with an emphasis on interpretative analysis focused on Heidegger interpretive phenomenological approach (Reiners, 2012). As the researcher has experience in this domain and therefore has prior knowledge, this philosophical approach was chosen for the study. A key driver for this approach involved evaluating how participants may be potentially engaged. At the pre-project stage, organisational participants may be limited in terms of their range of experience, role and potential number of participants. At the post project stage, there are likely to be more participants, though commercial project team members may have left post project closure. Users that support projects often return to their role within the organisation post go-live. The often-wide impact and resourcing required to initiate and deliver projects of this type may involve individuals of both varying experience and skills, therefore their understanding of both technical and project terms may also vary considerably. A qualitative approach therefore allowed a more tailored approach to participants experiences whilst using a core suite of interview questions. Potential participating sites were part of a maximum variation purposive sampling technique (Sharma, 2017).

This study collected primary semi-structured interview data from eight participating sites and 36 individual interview respondents though with nine projects (one site had two investment projects). As this study as cross sectional, data was gathered from both sites prior to, or early in project commencement, as well as sites having completed an implementation. Respondents were employees from participating sites - but often with widely varying functional backgrounds, a key component of the decision to use a qualitative approach.

The objective was to identify and engage with sites from a cross section of sectors that may have deployed or about to deploy enterprise technologies. This included commercial or manufacturing, government and academic or educational establishments. All sites were UK based. A wide core set of interview questions were developed and adapted to meet interviewee experience with their respective projects. Interview and survey questions were developed and tested via a single site pilot project. Philosophically, this research adopted an inductive, qualitative approach as a means of identification of additional quantitative research Bryman (2006), Cresswell, (1994), Denzin and Lincoln (2006).

Participant site coordinators for the study identified and provided initial engagement with respondents. Acquiring participating sites was complex; several potential survey sites withdrew as early engagement suggested that there was a reluctance to discuss project details, particularly if the project had failed to be fully implemented or abandoned. Participating sites were provided with guidelines on the spectrum of respondents that the study would seek to engage to collect the data. Each site identified staff willing to participate that met the criteria.

These target respondents could include:

- a) Governance staff – executives and steering group members.
- b) IT Managers.
- c) CEOs.
- d) Site seconded Project staff – either early project initiation stage or post project participants.
- e) Operations staff – managers and system users.
- f) System Support staff.
- g) Project manager – if in place or closing a project.

Participants were provided with the title and objectives of the study in conjunction with a consent form. These were collated prior to the site visit. A profile of the respondents and their employment categories is at Table 1. Several respondents had also been engaged in multiple roles with differing implementations on the same site.

The approach to qualitative data analysis was via a thematic approach (Braun and Clarke, 2019, Vaismoradi et al., 2013). A key benefit of this approach is that it provides for a comprehensive methodology that analyses communicated textual information with the intent of identifying evolving themes from the raw data. Interviews were digitally recorded if the respondent approve.

Similarly, thematic analysis also provides the capacity to develop more structured relationships within and between themes in the textual data. The application can, if the data evidence warrants it, provide the basis for model development from text data. To enable this analysis, given the volume of textual data, NVIVO software was deployed to evaluate if thematic patterns existed within the data; this was the most suitable option given both volume and complexity (Braun and Clarke, 2019; Cresswell, 1994). This ensured that data encompassed a range of business sectors including government. Data saturation (Fusch and Ness, 2015) became evident as many sites appeared to share a commonality of issues from initial text mapping as the data collection stage progressed.

To help reduce variation in the data analysis, two approaches were used to analyse the data – firstly, the use of study initial overall findings are encapsulated in an Ishikawa Fishbone diagram at Figure 2 (Ilie and Ciocoiu, 2010 citing Ishikawa, 1968). This diagrammatic model is used for representation of diagnostic cause and effect. This study has used the terms antecedent and outcome as findings are not suggested as causally related variables. The second approach involved the use of Gioia Methodology (Gioia et al., 2013) for identification and cross analysis of themes. This is at Figure D.

Data Site Analysis

Each site was coded for anonymity and a brief description of their organisational activity and associated enterprise project status (pre or post project); the customisation theme emerged as a subset of the research analysis. A summary table of sites is at Table 1.

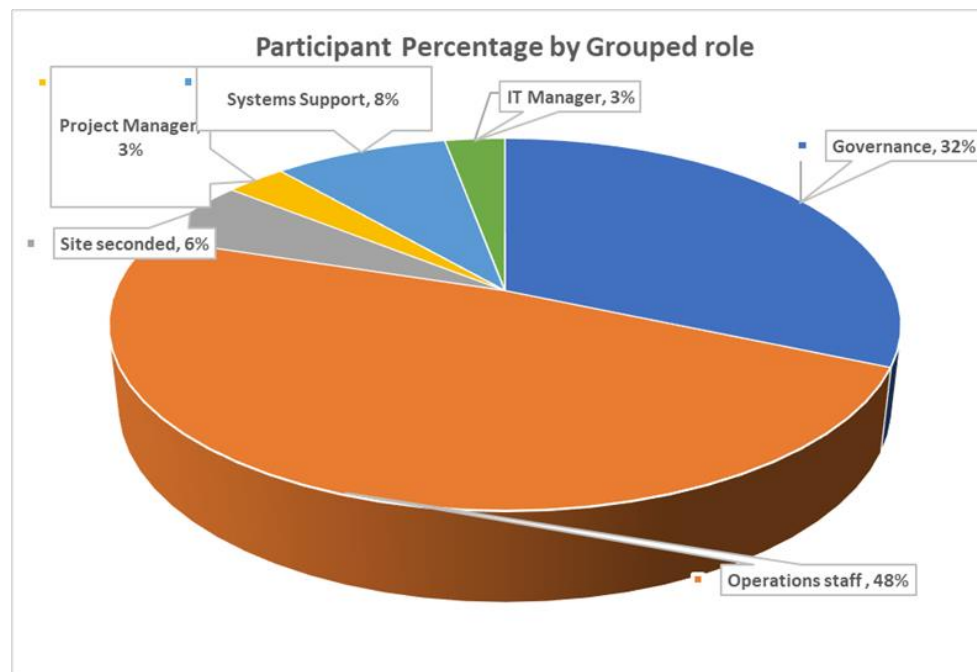
A more detailed analysis of participant organisational activity is at Table 2 and Graph 1.

Site Code name	Stage	Organisation Type	Description	Interview count	Notes
CSERVProj1	Complete	Commercial	Field service application	2	Interviewees covered both systems
CSERVProj2	Initiated	Commercial	Customer Contact Application	4	Interviewees covered both systems
PUBERP1	Initiating	Governmental	Full ERP	1	Conceptual phase initiated
EDMPUB	Complete	Governmental	Full enterprise non ERP application	6	Re-run implementation effort after several attempts
EDMMGT	Initiating	Educational	Enterprise Educational admin	6	Front end closing and initiation commenced
EDRPA1	Complete	Educational	Full ERP	7	Full ERP implementation
EDRPA2	Complete	Educational	Full ERP	8	Full ERP implementation
MAN01	Complete	Manufacturing	Full ERP	1	Perceptions from merger/multi site implementation.
WFL001	Complete	Commercial	Full integrated Enterprise workflow application	1	PM interview - company now closed, 2.5 years after implementation.
Total				36	Interviews c. 1hr duration.

Table 1 – Site and interview summary

Sector	Stage	Site	Position Group	Percentage of total participants
Government	Post	EDMPUB	Operations staff	11.1%
			Governance	2.8
	Pre	PUBERP1	IT Manager/CIO	2.8
Commercial	Post	CServProj1	Governance	5.5
			Governance	5.5
	Pre	CservProj2	Operations staff	11.1
			Governance	2.8
			Governance	2.8
Academic/Educational	Post	MAN01	Governance	2.8
			Governance	2.8
			Governance	2.8
	Pre	EDMMGT	Governance	2.8
			Site seconded	2.8
			Operations Staff	11.1
			Project Manager	2.8
	Post	EDRPA1	Governance	2.8
			Site seconded	2.8
			Operations staff	11.1
			Systems Support	2.8
			Governance	8.3
			System Support	5.5
	Post	EDRPA2	Operations staff	2.8
			Operations staff	2.8
Total				100%

Table 2 – Demographic Breakdown of sites by role.



Graph 1 – Participants by percentage grouped work category.

Antecedent

Outcome

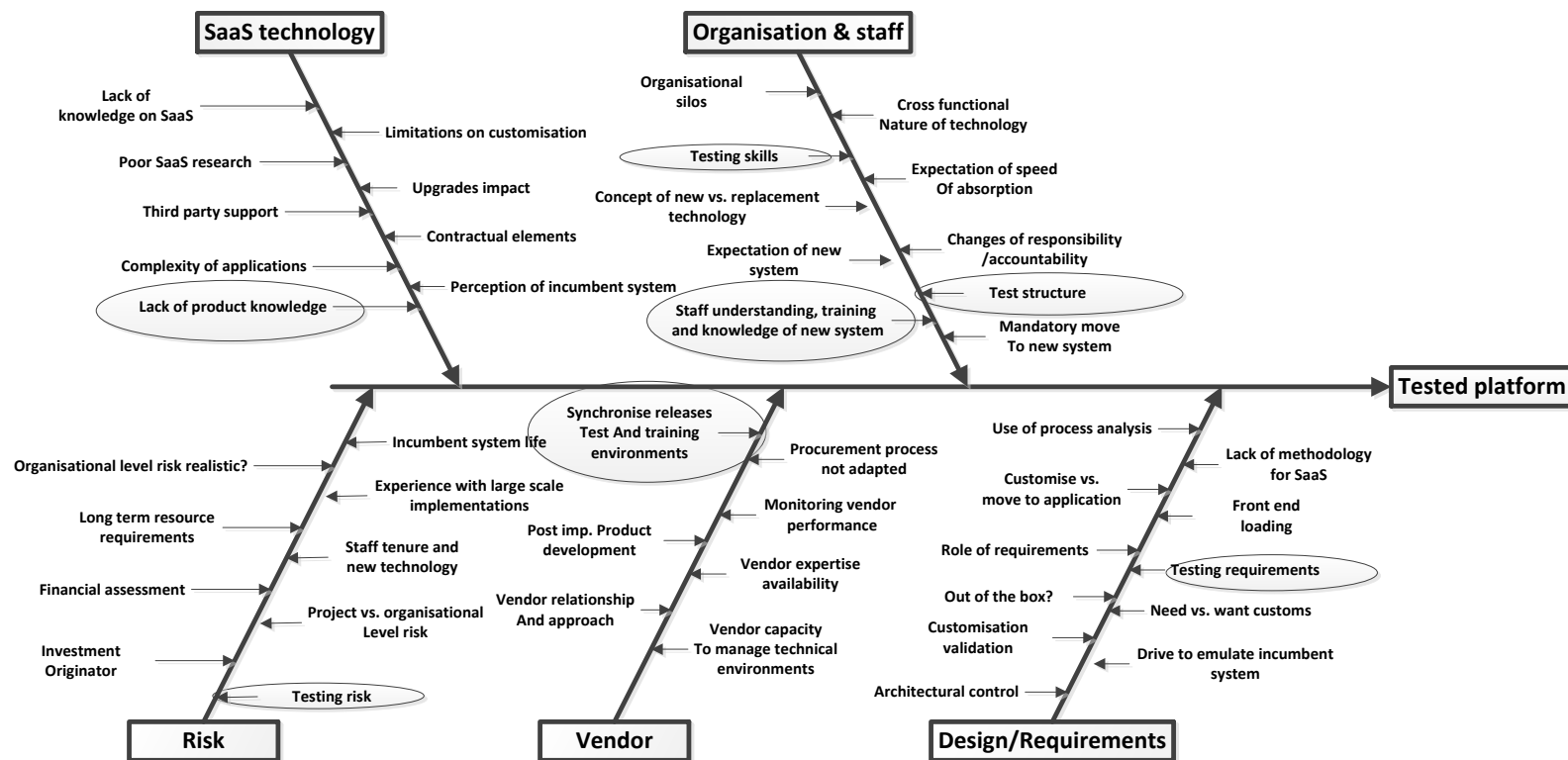


Figure 2 - Ilie and Ciocoiu(2010) citing Ishikawa (1968) Fishbone -antecedents and outcomes – paper area of focus.

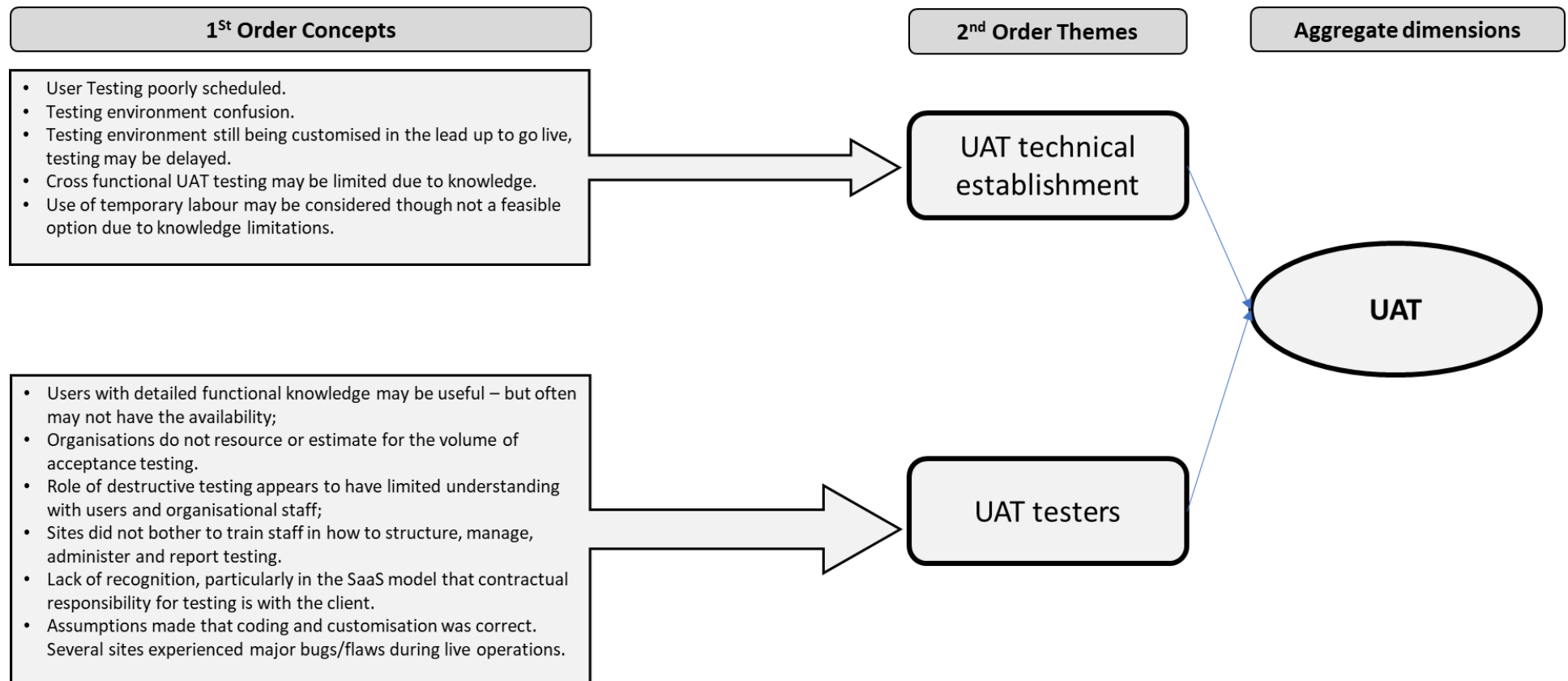


Figure D - Thematic analysis - Using Gioia Methodology (Gioia et al., 2013).

Top level theme	Description
Knowledge theme	Technical Product knowledge, project skills, testing of SaaS applications.
Testing	The efficacy of preparation, resourcing, management, structure, and delivery of system testing.

Table 3 – Top level themes and categories

The list of themes identified for this study are at Table 3.

Analysis and interpretation of the data used a latent approach and focused on the meaning of the dialogue as opposed to implicit or sub text patterns (Maguire and Delahunt, 2017). This study deployed data triangulation (Flick, 2004) using several site participants to cross validate more complex statements and findings on sites. This approach was designed to ensure that Lincoln and Guba (1985) suggestion concerning trustworthiness in qualitative research with its four sub-elements of credibility, confirmability, dependability, and transferability could be effectively evaluated.

The findings of this study were also reviewed and validated by a senior IT executive participant. This analysis will deploy a variant of Yourdon (1989) context diagrams. These diagrams align the comments as inputs to the process and are participant data responses.

Analysis

Analysis of the textual data at Figure 4 suggests that at both the pre and post project stages, a lack of test planning and skills in execution had significant impacts on the organisations capacity to deliver the right outcomes. It is important to stress that none of the sites had what may best be termed as a full failure i.e. project terminated and not implemented in to a live environment. Their key issues revolved around a range of often significant implementation issues that had testing been effective, would have likely to have been identified from testing.

Three of the sites (one at the pre-project stage and two at the post-project stage) were focused on major ERP implementations and were decommissioning accounting systems that were over twenty years old. Both participating sites had their previous system acceptance testing managed by their internal IT department but had opted to implement SaaS applications. This approach was triangulated, to some extent, by MANO1, a manufacturing facility where IT had retained the test execution function and released the working application to functional areas on completion of tests. Figure 4, comment 1 reflects the often-long term accumulated knowledge and experience of core ERP technology though was site that had seen a suite of upgrades as opposed to a new technology. However, this data is important as it reinforces the efficacy of testing if well documented designed and updated.

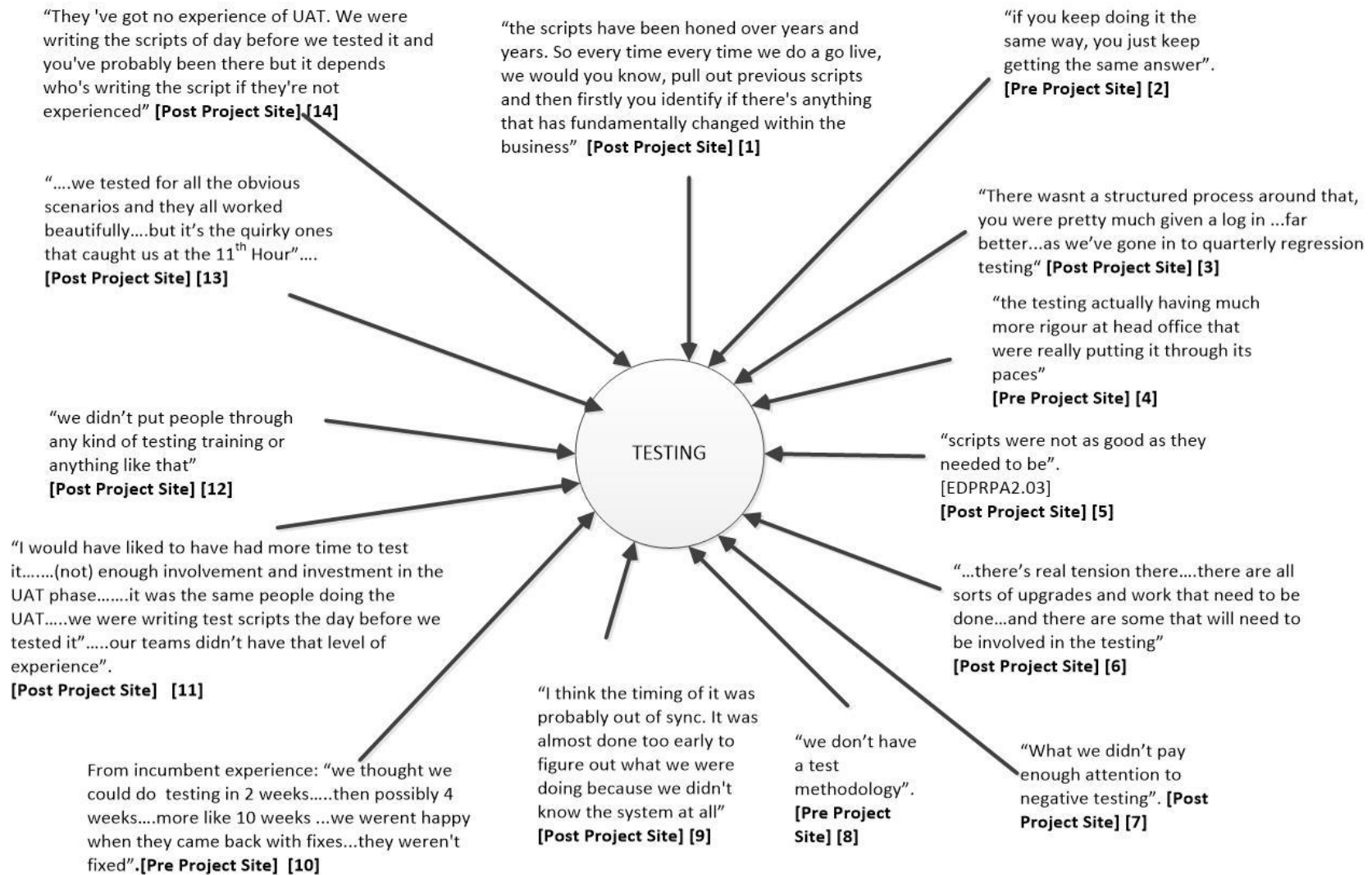


Figure E - Context diagram of Test Data

There is considerable evidence for the impact for lack of robust testing in the post-project study sites. These can be grouped in Figure 4 in to testing experience [1, 3, 6, 8, 9, 11,14], the failure to address both stress and destructive testing [7, 13] and a failure to develop staff skills and effort duration estimates [2, 11, 12, 14].

A table that summarises test approaches and issues by site is at Figure F below.

Site	UAT Test Risks	Key Issues
CSERVProj1	Limited experience.	Not tested to any great depth.
CSERVProj2	New CTO, seeking suite of SaaS test environments. Old Incumbent technologies and no experience with major system UAT.	New executive keen to see destructive testing and improved test discipline.
EDMPUB	Limited testing, few records kept.	Package was not customised to any extent.
EDMMGT	Pre Project phase, not mentioned by many participants as an issue. Several users managed incumbent UAT within their function.	One user had experience of UAT and patch release testing.
PUBERP1	Project at conceptual phase. No UAT test regime or methodology in place for organisation currently.	Large number of feeder integrated systems will be removed by new ERP. This could magnify test knowledge, experience and script development issues.
WFLO1	Tested by project/IT team.	Limited user internal experience with UAT (User Acceptance Testing).
MANO1	Release/Upgrades managed by core internal development team.	Users conducted only limited testing.
EDPRPA1	Limited, late-stage planning and not destructively tested.	Post implementation errors had highlighted the risks of test capability. Major error with payroll subsystem.
EDRPA2	Limited, late-stage UAT planning and not destructively tested.	Tested, but not necessarily within a structured environment.

Figure F - Site Testing approach

The impact of the risk associated with ineffective UAT was highlighted in one of the post-project

sites:

“We have had particular problems around payroll because of the complexity of what we've built in and designed...again, misunderstanding of people using it – and trying to use it the way they used it before”... [Post Project Site]

However, system vendors and integrators appeared to have contributed to user test effort by losing control over system instances. Instances are versions of the software installation for differing project team groups to access. One site had over thirteen of these instance versions running for a variety of technical reasons. Study data suggests that it had become apparent, as the project proceeded, that both control and configuration of these environments had been lost by the vendors. This had created considerable confusion and damaged the integrity of technology project management – and testing. As one senior user suggested:

“We had 13 environments and we found issues with the partner not keeping the config up to date in environments.” [Post Project Site].

However, for testing to be effective, many commercial project management methodologies focus on entry criteria. Entry criteria are conditions that need to be met for the activity to commence. There is considerable evidence in the study data that several key entry criteria to the acceptance testing phase had not been clearly defined. Subsequently, testing had become ad-hoc and inconsistent across a number of site implementations.

Knowledgeable Users

Experientially, one of the key entry criteria to the UAT phase is the need for knowledgeable, experienced and formally trained internal users. However, late in the project lifecycle, and close to go-live, one senior user suggested that:

“they still didn't know how the system was going to work...and this came up time and time again....we don't know how the system works”.
[Post Project Site]

Exposure to the new capability was also limited; all ERP sites in particular had opted for often extensive customisation and integration which had a direct impact on project cost and schedule. Given the extent of customisation on two sites in particular, failure on a UAT test may often lead to code repair and an associated regression test when fixed and made available. However, completely new functionality that is not delivered as part of the original package may not only require UAT testing but also both regression and destructive testing of any associated modules or code associated with the new function.

Some attempt had been made to train users; however:

“The training process was theatrical – timing was poor, there was some generic systems training but not on the final version shipped as the

implementation versionthis created a gap between the training and the final access...we then had to patch the training” [Post project Site]

Training had not identified the importance of systems access and well-designed programmes:

“training was complex and difficult ... it was not good at the time....we didn’t use the system for a while after”. [Post Project Site]

This data suggests that without a robust training system and strategy – testing may start to move back towards implementation and go live – increasing delivery risks.

Trained Testers and defined accountabilities

The data at Figure E also suggests that none of the sites had defined, early in the project lifecycle, the need for trained staff and effective test regimes, methods and scripting. However, identifying testers with the prerequisite levels of skill and knowledge had proved difficult. Execution of test scripts may in some cases be also simplistic – for example, as simple as - enter data, press enter. However, from personal experience, some test scripts may become require several extensive scenarios to be evaluated – and then errors captured, repaired and regression tested again until a pass is achieved.

It may be suggested that UAT tests may need a level of knowledge, skill and experience that may be limited organisationally. One senior user commented that users may also not understand the basics of their roles:

“I noticed it was interesting because it unnerved a number of people...they didn’t actually know what they were doing... when I was sitting with them and saying by okay, we want to look at this process. Let’s change it ...why? We do it this way... and they didn’t know why they were doing it. They results is because “I’ve just been told it’ll push that button and do that” when I was sitting with them and saying ... “okay, we want to look at this process, let’s change it” Why? we have been doing it this way...and they didn’t know why they were doing it... “I’ve just been told I’ll push that button... and do that”. So, I did have a number of individuals without really insecure about it all because they realised ...they didn’t understand” [Post project Site]

A key discovery in the data is that all sites appeared to delegate testing to each specific functional area - which is a logical decision. However, on most post project sites, triangulation of data suggested that this functional autonomy appeared to lack a single central focus for testing – in effect organisations appeared to lack a dedicated project test manager. This is an area that would warrant further research as it is difficult to envisage how coordination of a wide range of tests could be effectively managed without this focus. One participant had mentioned that in a previous role contract testers had been deployed to accelerate testing. However, the resulting outcomes often reflected that fact that these acceptance testers also did not understand the implications or ramifications of issues and associated functions.

However, Padmini et al. (2016) study appears to omit one key area in the acceptance testing domain – that of the level of package customisation. All post project sites, except one in this study, had applied often extensive customisation to the application. One post project site in particular, EDPRPA2, had also required that a number of onsite applications be developed outside of the SaaS application in an effort to overcome perceived shortcomings of the vendor software package. This created a range of integration touch points that also needed extensive testing – and therefore increased the test workload.

This would have extended the scope of testing to not only the package itself but a range of integrated functional or data modules developed to work alongside the core application. Issues with management of ERP customisation in particular have been analysed by Bridger (2021) and suggests that the drive to often replicate incumbent system functionality may induce assumptions in users that the new system will work in the same way as the legacy system - a factor that proved to be incorrect from data. It may be hypothesised that if users emulate existing functionality, there may be an assumption that robust, detailed testing is therefore not required. As a number of sites found, assumptions around software accuracy and robustness proved to be a major failing at point of implementation.

Padmini et al., (2016) analysis of key acceptance testing observations has been empirically aligned to each participating study site approach at Figure G. There are several common threads of data; sites appeared to have little detailed knowledge of a formal approach to testing, there was no single UAT team - and UAT appeared more functionally based and undertaken by secondees to the project where available.

This study also identified that scripting was often late, rushed and lacked a logical coherence mainly as staff had not been trained to develop test scripts. It may be inductively suggested that the late-stage development and poor coordination of scripts may be attributable to pressure on the project schedule. UAT appeared to have been stop-start or simply not conducted as one site openly admitted to.

No.	Observations – Padmini et al. (2018)	Study observations
1	UAT team does not have an understanding about testing levels and types, defect identification, defect logging, defect lifecycle, severity, and priority of use case scenarios and defects.	Little apparent understanding at both the pre project and pre-implementation phase of the skill, resource and coordination required.
2.	UAT Team is not clear about their roles and responsibilities.	Identified, UAT improved on participant sites as projects progressed.
3	Lack of knowledge in writing scenarios for UAT.	Identified, tests appeared to be simplistic from interviews.
4	Conflict between project leadership and UAT team's objectives.	Not reported.
5	Lack of communication between project leadership and UAT team.	Not reported.
6	Fear for vendor and project leadership.	Not reported.
7	Some UAT entry criteria were not passed, though UAT started.	Identified, little or no pre-test coordination mentioned in interviews, testing was often seen as necessary evil.
8	No proper communication between vendor and UAT team.	Not reported.
9	Necessary information not shared with the UAT team.	Not reported.
10	Limited presence of UAT team members at UAT premises.	Not reported.
11	UAT test scenarios are not get signed off by responsible body.	Indications in data that tests were completed though lack of clarity on process.
12	No proper documentation and naming convention practiced.	Identified.
13	No proper configuration management used.	Identified, training efforts Often hampered by poor configuration management on sites.
14	Lack of perception and attitude.	Identified, importance and impacts of poor testing experienced.

Figure G – Empirically amended UAT challenges (adapted from Padmini et al., 2016).

Discussion

The data in this study suggests that for many organisations, acceptance testing is often a late-stage deliverable, under resourced, poorly structured and not managed into the post project environment as part of an ongoing system management requirement. Only one site recognised the close relationship between UAT and destructive testing. A small number of sites did not test their applications and rolled them in to production. Subsequently, they had suffered a number of largely avoidable system failures that had undermined organisational confidence in the implementation. The role and impact of reliability, system use and a range of the factors has been identified in TAM or Technology Acceptance Models (Davis, 1989, Venkatesh and Bala, 2003), ISSM or Information Systems Success Model (Delone and Maclean, 1993) and the T-O-E or Technology-Organisation-Environment model (Tornatzky and Fleischer, 1990).

Participating sites in this study had highlighted that the new system was crucial but was one component of a continually changing technical landscape within the organisation. This implies that testing techniques may have been learned or acquired as previous organisational knowledge. However, this study suggests that this testing experience is rarely well developed, reused or documented. It may also imply that acceptance testing is point solution based i.e. managed haphazardly for each new technology acquisition or solution. Cooke-Davis (2002) research suggests that organisations who have the capacity to learn from experience and effectively re-apply knowledge have a considerably enhanced chance of project success.

One of the key elements in organisation of UAT is in the availability of a system that is up to date and contains the relevant code. In effect, this is not too dissimilar to the availability of a viable, consistent training platform. This study suggests that these two requirements are tightly coupled and one activity precedes the other; training then testing. The lack of a correctly configured, available application instance for training was also identified by Chadhar and Daneshgar (2018) in a study on situated learning on an ERP site. Similarly, their research suggested that a lack of broader training and knowledge of the new technology and the level of integration had hampered adoption and integration of the capability into operations.

Dos Santos et al. (2018) identified some 21 techniques that are used across technology projects that may be used to deliver user acceptance testing. These varied from automated testing to the use and development of natural language skills test scripts. However, it is suggested that “it is not possible to define “what is the best one” empirically.” (Dos Santos et al., 2018, p.419). Identifying employees who can deliver robust testing and system bug tracing may not be that easy to achieve; this study does not support proposition that “the most technologically-ready employees should be selected in order to test the implemented system” (Chatzoglou et al., 2016, p.1250). This study suggests that participating organisations in this study did not formally identify and train users to conduct effective user acceptance testing. The study data also suggests that coordination between the project, vendor and user acceptance testing participants appeared confused and poorly coordinated. Some participating sites also minimised or did not test their applications; when this is combined with poor business requirements engagement, implementations struggled to develop or delivery their perceived investment benefits. Norton (2021) suggests that user acceptance testing is a key part of the change management process; though in a study focused on the implementation of a medical imaging system, the study

concluded that “The acceptance test of a PACS is an important evaluation of the PACS components, modality connections, network communications, and system performance. In addition to a system inventory, certifying contract compliance, and identifying deficiencies, an AT is a technical evaluation of a system that demonstrates readiness for clinical use and a baseline for a facility’s quality control program that will monitor the PACS in the future.” (Norton, 2021, p.442). Padmini et al., (2016) findings are fully supported.

There are several elements to the testing phase; firstly, there is no assumption that the entire application requires testing. As packaged SaaS applications may be used simultaneously by many other organisations, effective debugging and testing is likely to be conducted by the vendor; this does not automatically eliminate a test phase of some kind - though bug risks are considerably reduced. It may also be suggested by this study that where little or no custom code is added to the application, and configuration has been deployed, this too may significantly reduce test overhead as configuration only changes the “look and feel” of the application and not the processing of data. However, where customisation is used, particularly on accounting systems, testing may become increasingly more difficult – and with a higher associated risk if the application is not thoroughly tested.

For organisations looking to acquire an ERP in particular, there is some merit in evaluating the test requirements at the pre project stage. Research in to front-end or pre-project activity suggests that “empirical evidence of project performance has consistently demonstrated that ultimate project success and failure can often be traced back to what happens at the early part of the project life cycle but that our knowledge of how to manage this stage of a project’s development cycle is often poor” (Edkins et al., 2013, p.81). It has also been proposed that organisational pre-project activity “distinguishes between those that see the front-end as the vital ‘shaping’ part of a potential project, and those that see the project only starting once the ‘front-end’ is completed” (Williams et al., 2019, p. 1138 citing, Morris, 2016). Turner (2022), though a construction related perspective on mega-projects, suggests that many later stage issues or project challenges emerge due to a lack of scenario planning at the pre-project stage. This study suggests that many organisations do not consider, at the pre-project stage, a range of scenarios that are acceptance testing risk oriented, for example, system testing, training, resourcing and long-term system management/release testing. SaaS application deployments are starting to drive this requirement and this study’s value lies in focusing on a novel scenario based approach to long terms system integrity and testing.

However, a focus on training and building skilled user acceptance test teams, though an upfront cost where little or no budget exists until the project is formally initiated, may not be feasible. However, recognition of the relationship between customisation, packaged software and testing is well understood; most organisations will have conducted a range of previous implementations. This would be either from system upgrades or new or replacement technology investments. This study suggests that there is likely to be a growing importance on user acceptance testing skills as more vendors move their applications to the SaaS environment. It is suggested that the SaaS market, by 2030 will reach some \$10trn globally (McKinsey, 2022). Data from one site in this study suggests

that the development of comprehensive scripts within projects makes them eminently reusable for upgrades and software patch management.

It has also been suggested that acceptance testing effort is directly related to software requirements definition; “The consequences of unmanaged or improperly managed requirement changes can spell disaster for system development. These negative consequences can result in software cost and schedule overrun, unstable requirements, endless testing and can eventually cause project failure and business loss” (Jayatilleke and Lai, 2017). This study does not fully support these findings on the basis that customisation requests, implementation of the change and associated testing may occur at any time in the project lifecycle.

Conclusion

This study concludes that, based on the gathered data, that User Acceptance testing of new technology is still poorly understood and executed. The study demonstrates that poor acceptance testing on one participating site, with one specific module, had been inconsistent and lead to a major employee impact - and considerable associated rework to remedy the issues. Organisational emphasis, either pre project or early in the project lifecycle on training staff to conduct UAT and formalising a management method could yield significant dividends later in the project lifecycle when both schedule and budget pressures often become more intense.

There are several limitations in the development of this study. The most significant is that the data collection focused on pre-project perceptions of test effort – and post project experiences of the test phase and the subsequent impact on delivery of the new capability. At the post project stage, many staff who had worked on the project team had left and therefore users who had conducted UAT often came from varying backgrounds and differing levels of experience.

This study would have benefitted from both a longitudinal and detailed study examination of project artefacts, like test scripts and associated issue registers to help build a timeline of decisions and activity that organisations undertake, during projects, to both mobilise and deliver the test phase. However, this also creates a focus for further research on what is a key project deliverable in the pursuit of effective Enterprise Systems implementations.

References

- Akingbehin, K., 2006, July. Towards destructive software testing. In *5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse (ICIS-COMSAR'06)* (pp. 374-377). IEEE.
- Braun, V. & Clarke, V. 2019, "Reflecting on reflexive thematic analysis", *Qualitative Research in Sport, Exercise and Health*, vol. 11, no. 4, pp. 589-597.
- Bridger, A.J., Enterprise System Training – a whole of system life commitment? 31st Aug – 2nd September 2022, *BAM 2022 Conference: Reimagining business and management as a force for good* British Academy of Management (BAM)
- Bridger, A.J., *ES (Enterprise System) Project Customisation – start as you mean to go on*, 31st Aug – 3rd September 2021, “*BAM2021 Conference in The Cloud*”. British Academy of Management (BAM)
- Bryman, A., 2006. Integrating quantitative and qualitative research: how is it done?. *Qualitative research*, 6(1), pp.97-113.
- Chadhar, M.A. & Daneshgar, F. 2018, "Organizational Learning and ERP Post-implementation Phase: A Situated Learning Perspective.", *J.Inf.Technol.Theory Appl.*, vol. 19, no. 2, pp. 7.
- Chatzoglou, P., Fragidis, L., Chatzoudes, D. & Symeonidis, S. 2016, "Critical success factors for ERP implementation in SMEs", *2016 Federated Conference on Computer Science and Information Systems (Fed CSIS)*IEEE, , pp. 1243.
- Cooke-Davies, T.J. 2002, "Consistently doing the right projects and doing them right–What metrics do you need", *The measured*, vol. 4, no. 2, pp. 44-52.
- Cresswell, J.W., 1994. Research design.
- Desalegn, J. & Pettersson, A. 2018, "Investigation of Critical success factors for ERP implementation: A user perspective".
- Dos Santos, E.C., Vilain, P. and Longo, D.H., 2018, May. Poster: a systematic literature review to support the selection of user acceptance testing techniques. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)* (pp. 418-419). IEEE.
- Edkins, A., Geraldini, J., Morris, P. & Smith, A. 2013, "Exploring the front-end of project management", *Engineering Project Organization Journal*, vol. 3, no. 2, pp. 71-85.
- Flick, U. 2007, "Concepts of triangulation", *Managing quality in qualitative research*, , pp. 38-54.
- Flick, U. 2004, "Triangulation in qualitative research", *A companion to qualitative research*, vol. 3, pp. 178-183.

- Fusch Ph D, P.I. and Ness, L.R., 2015. Are we there yet? Data saturation in qualitative research.
- Geetha, U., Sankar, S. and Sandhya, M., 2021. Acceptance testing-based test case prioritization. *Cogent Engineering*, 8(1), p.1907013.
- Gioia, D.A., Corley, K.G. and Hamilton, A.L., 2013. Seeking qualitative rigor in inductive research: Notes on the Gioia methodology. *Organizational research methods*, 16(1), pp.15-31.
- Guba, E. G., & Lincoln, Y. S. (1994). Competing paradigms in qualitative research. In N. K. Denzin & Y. S. Lincoln (Eds.), *Handbook of qualitative research* (pp. 105–117). Sage Publications, Inc.).
- Hanssen, G.K. and Haugset, B., 2009, January. Automated acceptance testing using fit. In *2009 42nd Hawaii international conference on system sciences* (pp. 1-8). IEEE.
- Ilie, G. & Ciocoiu, C.N. 2010, "Application of fishbone diagram to determine the risk of an event with multiple causes", *Management research and practice*, vol. 2, no. 1, pp. 1-20.
- Jayatilleke, S. and Lai, R., 2018. A systematic review of requirements change management. *Information and Software Technology*, 93, pp.163-185.
- Kumar, R. 2017, "Software as a Service (SAAS), Future of Information Technology & Business", *International Journal*, vol. 5, no. 2, pp. 199-203.
- Lincoln, Y.S. & Guba, E.G. 1985, *Naturalistic inquiry*, Sage.
- Lobkov, K., 2020. Acceptance Testing for Software as a Service (SaaS). *Software Quality Professional*, 23(1).
- Maguire, M. and Delahunt, B., 2017. Doing a thematic analysis: A practical, step-by-step guide for learning and teaching scholars. *All Ireland Journal of Higher Education*, 9(3).
- McKinsey, 2022, Title: <https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/the-saas-factor-six-ways-to-drive-growth-by-building-new-saas-businesses> (Accessed: 3 August 2022)
- Melnik, G. and Maurer, F., 2005, October. The practice of specifying requirements using executable acceptance tests in computer science courses. In *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications* (pp. 365-370).
- Memon, A.M. and Soffa, M.L., 2003. Regression testing of GUIs. *ACM SIGSOFT software engineering notes*, 28(5), pp.118-127.
- Vijayanathan Naganathan, S.B., 2011. The challenges associated with SaaS testing.
- Norton, G.S., 2021. Change Management and Acceptance Testing. In *1Practical Imaging Informatics* (pp. 425-444). Springer, New York, NY.

- Padmini K. V. Jeeva, I. Perera and H. M. N. Dilum Bandara, "Applying agile practices to avoid chaos in User Acceptance Testing: A case study," *2016 Moratuwa Engineering Research Conference (MERCon)*, 2016, pp. 96-101, doi: 10.1109/MERCon.2016.7480122.
- Reiners, G.M. (2012), "Understanding the Differences between Husserl's (Descriptive) and Heidegger's (Interpretive) Phenomenological Research". *J Nursing Care* 1:119.
- Ricca, F., Torchiano, M., Di Penta, M., Ceccato, M. and Tonella, P., 2009. Using acceptance tests as a support for clarifying requirements: A series of experiments. *Information and Software Technology*, 51(2), pp.270-283.
- Seres, L., Tumbas, P., Matkovic, P. & Sakal, M. 2019, "Critical success factors in ERP system adoption: comparative analysis of the private and the public sector".
- Sharma, G. 2017, "Pros and cons of different sampling techniques", *International journal of applied research*, vol. 3, no. 7, pp. 749-752.
- Shaul, L. and Tauber, D., 2013. Critical success factors in enterprise resource planning systems: Review of the last decade. *ACM Computing Surveys (CSUR)*, 45(4), pp.1-39.
- Sommerville, I., 2016. Software Engineering 10th Edition (International Computer Science). *Essex, UK: Pearson Education*, pp.1-808.
- Standish Group, T., 2018. CHAOS Report: Decision Latency Theory: It Is All About the Interval.
- Statista 2022, Enterprise Software, <https://www.statista.com/outlook/tmo/software/enterprise-software/worldwide> (Re-Accessed 20 June 2022)
- Sun, W., Zhang, X., Guo, C.J., Sun, P. & Su, H. 2008, "Software as a service: Configuration and customization perspectives", *2008 ieee congress on services part ii (services-2 2008)*IEEE, , pp. 18.
- Turner, R., 2022. Using Principal–Steward Contracting and Scenario Planning to Manage Megaprojects. *Project Management Journal*, 53(1), pp.8-16.
- Vaismoradi, M., Turunen, H. & Bondas, T. 2013, "Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study", *Nursing & health sciences*, vol. 15, no. 3, pp. 398-405.
- Williams, T., Vo, H., Samset, K. & Edkins, A. 2019, "The front-end of projects: a systematic literature review and structuring", *Production Planning & Control*, vol. 30, no. 14, pp. 1137-1169.
- Yamaura, T., 1998. How to design practical test cases. *IEEE software*, 15(6), pp.30-36.
- Yourdon, E. 1989, *Modern structured analysis*, Yourdon Press.