Tanzania Journal of Engineering and Technology 2022, **41(3)**:1-13
OPEN ACCESS articles distributed under Creative Commons Attribution Licence [CC BY-ND]
Websites: https://ajol.org/tjet; https://tjet.udsm.ac.tz

**Copyright** © 2022 College of Engineering and Technology, University of Dar es Salaam
ISSN 1821-536X (**print**); ISSN 2619-8789 (**electronic**)
https://doi.org/10.52339/tjet.v41i3.839

*Special Issue – Sixth Tanzania Annual ICT (TAIC) Conference, 26 – 28, October 2022, Zanzibar, TANZANIA*

# Investigating the Support for Agility in Developing Government Software Systems: A Case of Three East African Countries

**Leonard Peter Binamungu**† **and Masoud Mahundi**

Department of Computer Science and Engineering, University of Dar es Salaam, P.O. Box 33335, Dar es Salaam,

†Corresponding author: lepebina@udsm.ac.tz, https://orcid.org/0000-0001-6385-1353

## ABSTRACT

**Agile Software Development Methods support an iterative and incremental way of developing software systems while responding to change by prioritising various aspects atdifferent times. This differs from traditional sequential methods like Waterfall, in which one software development stage has tobe completed before starting the next stage. To produce software systems that meet the requirements of their institutions, several governments in Africa have issued standards and guidelines to be followed during the development of government software systems. Such standards and guidelines specify the specific activities and deliverables for each stage of software development. While well-intentioned, such guidelines and standards can also hinder creativity and innovation which could be key to producing good quality and sustainable software systems. Given the degree of leniency that they offer, Agile methods could give room for such creativity and innovation among team members. However, despitesuch good potential in Agile methods, the literature lacks evidence regarding if and how the software development guidelines and standards issued by several African governments support agility. Various documents from three East African countries were reviewed, to determine if and how they support for agility during the development of government software systems. Guidelines and standards were reviewed using the lens of four Agile Values stated in the Agile manifesto. Results show the following: there is a marked lack of support for agility during the development of government software systems; the standards and guidelines are generally characterised by excessive micromanagement of the development process, leaving little or no room for innovation and creativity amongst members of the development teams; and the guidelines seem to assume uniformity across development projects, irrespective of the fact that software development projects can vary depending specific contextual dictates. Furthermore, recommendations on how governments can adopt and support agility during software development are provided.**

ARTICLE INFO

**First Submitted**: June 28, 2022

**Revised**: Sep. 30, 2022

**Presented**: Oct. 26-28, 2022

**Accepted:** Nov. 14, 2022

**Published:** Nov. 30, 2022

**Keywords:** *Agile software development, government software systems, support for agility, agile project management, agile implementation*

## INTRODUCTION

To increase the efficacy of the delivery of public services to citizens, governments across the world have been adopting the use of Information and Communication Technologies (ICTs). This is also true of African governments. The use of ICTs in African governments ranges from basic communication through emails to the use of various software systems that attend to core business functions of the government, such as planning and budgeting, financial services, payrolls, management information systems, and decision support systems, among others. With an increase in the use of software to support key functions and business processes in governments, several governments in Africa have instituted mechanisms to guide how the software systems for use in support of government services should be developed and maintained. More specifically, some governments have developed and instituted guidelines and standards for developing and maintaining government software systems. Irrespective of how well-intentioned these standards and guidelines are, there are still chances that they could either facilitate or hinder the process of obtaining good quality and sustainable government software systems. Guidelines and standards can be especially facilitative if they enable governments to get software systems that meet the requirements, but they can also be a hindrance particularly if they restrict flexibility and creativity during software development.

To provide a level of flexibility in developing government software systems amid higher levels of bureaucracy, some governments have adopted Agile Software Development Methods (ASDMs) (Patanakul & Rufo-McCarron, 2018). Grounded in the Agile Manifesto (Beck et al., 2001), ASDMs embrace an iterative way of developing software systems, breaking the rigidity found in sequential methods like waterfall. Thus, ASDMs provide adequate room for flexibility and accommodation of changes during software development, while focusing on delivering valuable working software (Patanakul & Rufo-McCarron, 2018; Abrahamsson, Salo, Ronkainen, & Warsta, 2017; Dingsøyr, Nerur, Balijepally, & Moe, 2012). Studies have indicated that Agile methods can work in bureaucratic environments typical of many government institutions, providing several benefits such as improving software quality and customer satisfaction, among others (Pinheiro, Maurer, & Sillito, 2008, 2009, 2010; Patanakul & Rufo-McCarron, 2018). Strategies for adopting Agile methods in developing government software systems have also been proposed in the literature (Vacari & Prikladnicki, 2015; Fruhling, McDonald, & Dunbar, 2008; Pinheiro et al., 2008, 2009, 2010).

Furthermore, several African governments have issued guidelines and standards for the development of software systems for government institutions. However, as far as this subject matter is concerned, no previous studies have investigated the extent to which these guidelines and standards support agility during the development of government software systems. Among other things, such a study could help governments to position themselves in a way of reaping the benefits of using ASDMs. To fill this gap, software development guidelines and standards issued by three countries in East Africa–Tanzania, Kenya, and Rwanda–are analysed to determine how they support or hinder agility during the development of government software systems. The following four agile values were used as theoretical lenses during the documents review process: the value of individuals and interactions over processes and tools, the value of working software over comprehensive documentation, the value of customer collaboration over contract negotiation, and the value of responding to change over following a plan.

The results of this study revealed a marked lack of support for agility during the development of government software systems. This can be evidenced by excessive micromanagement of the development process, leaving little or no room for

innovation and creativity amongst members of the development teams; and the assumption of uniformity across development projects, when, in practice, software development projects can vary a great deal. In the end, recommendations on how government can accommodate agility in developing software systems are provided.
The rest of this paper is structured as follows: the related work on supporting agility in developing government software systems is presented, followed by the description of the approach used to collect and analyse data for this research. Results in their raw form are then presented, followed by analysis and discussion. Finally, the conclusion, recommendations on how governments can embrace agility in software development, as well as future research directions are unveiled at the end of the paper.

## RELATED WORK

This section presents work related to supporting agility in the development of government software systems. In particular, to demonstrate the importance of agile methods indeveloping government software systems, this paper starts by presenting the literature showing the upsides of using agile methodsin developing software systems for public sector contexts. Afterwards, the challenges that face the adoption and use of agile practices in developing government software systems are presented. Moreover, to show how agile methods can be putto use in public sectors, the strategies that have been used to adopt agile methods in developing government software systems are presented. Because specific criteria are required to determine whether or not the software development guidelines and standards issued by certain governments supportagility, the paper also discusses how to measure the agility of an organisation/business process. The review of the literature is concluded by stating the gap that this research aims to fill.

**Benefits and challenges of using agile methods in developing government software systems:** (Pinheiro et al., 2008, 2009) studied the challenges posed by inflexible waterfall processes in a highly bureaucratic large government agency, as well as the impact of adopting agility in the same organisation.Among other things, rigid waterfall processes were found to produce unstable software, characterised by frequent bugs and failures. On the contrary, the adoption of an agile process was found to increase software quality and stability, as well as enhanced client satisfaction. In another study conducted in a highly bureaucratic government agency, (Pinheiro et al., 2010)made the comparison between projects that followed agileprocesses from the beginning and projects that had migrated to agile processes after starting with non-agile processes. Their comparison focused on two main project aspects: the times taken to fix bugs, as well as the ability to prevent budget and schedule overruns. Not only that following an agile process from project inception was found to detect bugs early in time and to reduce the time taken to fix bugs, but also it was found to prevent both cost and schedule overruns.
Some researchers investigated how a software company contracted by the US government agency transitioned to agile methodologies from traditional waterfall methodologies. As summarised in the following quote,the study uncovered several benefits of using agile methodologies on a government project: *"ability to change, flexibility, customer engagement, velocity, immediate customerfeedback, team members knowing and understanding their tasks/priorities, faster delivery time, more cohesive relationships between the product owners and project team, morebuy-in from the customer, increased quality of the product being delivered.*" (Patanakul & Rufo-McCarron, 2018, p. 185). They also reported the following six challenges: unclear understanding of the concept of agility in software development, partly due to lack of proper training and coaching; resistance to

change among project stakeholders; loose commitment from the government agency (product owner), which delayed some important decisions; lack of clarity between project stakeholders regarding when and what to document; hardship in integrating agile practices with existing processes, techniques, and tools–for example, making testing an integral part of the ongoing development process; and perceived hardships in automating tasks (e.g., test automation), when automation is regarded to be an integral part of a typical agile process. Similar problems regarding the use of agile methods were reported in other studies in the literature e.g., (Dikert, Paasivaara, & Lassenius, 2016; Ghobadi & Mathiassen, 2016). Moreover, a study by (Vacari & Prikladnicki, 2015) found that while agile processes can be adopted when developing government software systems, the adoption process can equally be particularly challenging and complex due to lack of experience, management support, and stakeholders' resistance, among others.

**Strategies for adopting agile methods in developing government software systems:** Previous research has also revealed a number of strategies that can be used to adopt agile methods in bureaucratic public organisations (Vacari & Prikladnicki, 2015; Fruhling et al., 2008; Pinheiro et al., 2008, 2009, 2010). Among them are earning top-level management support (Pinheiro et al., 2008, 2009; Tureček, Šmiřák, Malík, & Boháček, 2010; Vacari & Prikladnicki, 2015); collaboration, as well as timely communication and sharing among project stakeholders (Fruhling et al., 2008; Planning, 2006); starting with a few willing people who might in turn influence others (Vacari & Prikladnicki, 2015); and gradual introduction of agility with demonstrable results, while balancing schedule, budget, and project scope (Alleman, Henderson, & Seggelke, 2003).

**Measuring the support for agility in an organisation:** Measuring how agile an organisation/business process is, is generally hard and context-sensitive (Gong & Janssen, 2010). Nevertheless, studies have proposed various proxy metrics for measuring the degrees of agility in organisations/business processes, mostly focusing on the costs of change, in terms of time, money, and human resources (Gong & Janssen, 2010; Gebauer & Schober, 2006; Jansen-Vullers, Kleingeld, Loosschilder, Netjes, & Reijers, 2007). (Gong & Janssen, 2010), for example, proposed several quantitative and qualitative metrics for measuring the agility of a business process in a particular organisation. Among them are the following five metrics:

- *Throughput:* The number of processes/operations executed per unit of time. The higher the throughput the higher the agility of a business process/organisation.
- *Response time:* Time is taken for an interaction between components/stakeholders to complete. The lower the response time the higher the agility of a business process/organisation.
- *Case handling time:* Time is taken for a business process operation to complete. The lower the case handling time the higher the agility of a business process/organisation.
- *Operational cost:* The number of resources (human/financial) spent on a business process. The lower the operational cost the higher the agility of a business process/organisation.
- *Quality:* The level of customer satisfaction, as measured through such things as the number of complaints/appeals in a given time. In the software development/maintenance context, customer satisfaction can be indicated through such things as the number of software bugs reported by users. The higher the quality the higher the agility of a business process/organisation.

**Research Gap:** To the best of our knowledge, no previous study has investigated how formal software development guidelines and standards issued by various governments support or hinder agility. *Inter alia*, such studies could offer insights regarding the extent to which governments' guidelines and standards provide room to reap the benefits brought by embracing agile software development methods.

## METHODS AND MATERIALS

This study used document review as a method to gather and analyse data. The details of data collection and analysis are presented in the next subsections.

### *A. Data Collection*

The East African Community (EAC) has six countries: Tanzania, Kenya, Uganda, Rwanda, Burundi, and South Sudan. Put together, the six EAC countries have a population of about 177 million (East African Community, 2020). Such a large population could benefit from effective ICT-enabled government services, and effective government software systems could be one way to dispense services of good quality to citizens. Since the quality of the software development process can impact the quality of the resulting software product, authors particularly wanted to study the agility of the process used to develop government software systems for countries in the East African region, in order to generate lessons that might inform the improvements of both the software development processes and products in governments.

To that end, for each of the six countries, the study first wanted to get official documents in which guidelines and standards for the development of government software systems are stipulated. The preliminary search conducted revealed that in four of the six countries–Tanzania, Kenya, Uganda, and Rwanda–the standards and guidelines for the development of software systems for government ministries, departments, institutions, or agencies are overseen by a country-specific authority for e-government: *e-Government Authority* for Tanzania, *ICT Authority* for Kenya, *National Information Technology Authority-Uganda (NITA-U)* for Uganda, and *Rwanda Information Society Authority* (RISA)[4] for Rwanda. For Burundi and South Sudan, however, authors were unable to get either evidence of the existence of such authorities or documents stipulating guidelines and standards for the development of government software systems. Burundi and South Sudan were thus excluded in our further search and analysis of documents. Moreover, for each of the four remaining countries, the websites of the respective e-government authorities for documents detailing guidelines and standards for the development of government software systems were searched. After that, snowballing approach was used to obtain from other related government ministries, departments, institutions and agencies, additional documents with guidelines and standards for the development of government software systems.

All retrieved documents were screened for relevance. A document was considered relevant for our analysis if the whole or part of it was about guidelines and standards for the development of software systems in government institutions. More specifically, authors restricted the analysis to guidelines and standards for the process starting from gathering software requirements and converting requirements into working software, to the commissioning of a software system. Thus, the agility of processes after government software systems are in operation are beyond the scope of the present study. Table 1 summarises both the number of documents that were obtained for each country and those that passed our relevance criteria.

Based on Table 1, none of the documents obtained from Uganda passed the relevance criteria, so Uganda was also excluded from further analysis. Thus, what is discussed hereafter are the analysis and results based on documents from the following three East African countries: Tanzania, Kenya, and Rwanda.

### *B. Data Analysis*

To extract information on agility support in the development of government software systems, each section in the eleven relevant documents was critically analysed to discover if and how it supported or hindered agility during the development of government software systems.

**Table 1: Distribution of documents for the four countries**

| S/N | Country | Total number of retrieved documents | Total number of relevant documents |
|---|---|---|---|
| 1 | Tanzania | 35 | 7 |
| 2 | Kenya | 11 | 2 |
| 3 | Uganda | 8 | 0 |
| 4 | Rwanda | 6 | 2 |
| | Total | 60 | 11 |

In particular, the following four Agile Values (Beck et al., 2001) were used as a theoretical lens against which various sections in the relevant documents were examined for support or hindrance of agility:

- The value of individuals and interactions *over* processes and tools.
- The value of working software *over* comprehensive documentation.
- The value of customer collaboration *over* contract negotiation.
- The value of responding to change *over* following a plan.

For each relevant document, authors recorded how each of thefour Agile Values was supported or hindered by the various sections/parts of the document. To ensure uniformity during document analysis, authors worked together during the review of the first two documents to determine how the Agile Values were supported or hindered. Thereafter, there was branching, and each author focused on a subset of documents. The authors then worked together to cross-check the evidence gathered by individual authors from subsets of the relevant documentsthey were working on.
Going together through the evidence collected by different authors helped to identify parts where both authors agreed, as well as parts on which they disagreed. A section/part in a particular document was admitted as support/hindrance of a specific Agile Value if both authors agreed.

## RESULTS AND DISCUSSIONS

### Results

In this section, the results of this study are presented based on the agile values. Because of the long official names of the documents that actually produced the results that are about to be presented, names of the documents are shortened to simplify references in the article. Table 2 lists these documents and their short names. On a general note, this study found three aspects characterising the policy frameworks for developing government software systems in the analysed countries.

**Table 2: Short names for documents that produced results after a detailed review**

| **Short name** | **Actual name** |
|---|---|
| *Tanzania 1* | Standards for Development, Acquisition, Operation and Maintenance of e-Government Applications |
| *Tanzania 3* | Government Software Applications Quality Assurance Checklist |
| *Tanzania 4* | Quality Assurance Compliance Guidelines for e-Government Applications |
| *Tanzania 5* | National Information and Communication Technology Policy, May 2016 |
| *Kenya 1* | The strategic framework, administrative structure, training requirements and standardisation framework |
| *Kenya 3* | Systems and Application standards |
| *Rwanda 1* | ICT Implementation guidelines in Government Institutions |
| *Rwanda 2* | Enterprise Architecture Blueprint Development Guidelines for GoR |

While they manifest differently across countries, they remain notable across all of them, only different in extents. These characterising aspects are:

(1) *Excessive micromanagement of the software development process:* The level of detail in the guidelines does not give much freedom of choices to development teams. The list of mandatory documents developers has to produce as required in Rwanda, Kenya and Tanzania, the mandatory steps to follow as documents in Tanzania require, and the quality checks put in place by governments for all the three countries leave the development processes dependent on high level authorities. In Rwanda, through *Rwanda 1* for example, developers are required to submit to RISA their choices for development platforms and programming languages for approval. This micromanagement seriously suffocates the flexibility to adopt to changes as agile methods advocate.

(2) *Lack of recognition of agile methods:* While there are occasional mentions of agile methods as valid methods for software development in public institutions, most guiding documents are not in support of it. This is clearly notable through the different guides provided in each phase of software development, from requirements elicitation to as far as implementation of the developed software. Most of the analysed documents include agile methods in the iterative methods but the guidelines often state against agile values.

(3) *Assumption of uniformity across projects:* Most of the decrees in the analysed guidelines tend to assume similarities in the software developed in terms of size, complexity, and operationalisation. They mostly assume large-scale development. Public institutions have too many differences between them to be closely guided by common standards. In this regard, Kenya stands out as an exception where *Kenya 3* advises flexibility in consideration of the size and complexity of the application. They do, however, insist on the same set of documentation that developers have to produce.

The subsequent subsections give a detailed account of how the dictates in the policy framework support or hinder the agile values, and that is done by focusing on different phases of software development.

### A. *Individuals and interactions over processes and tools*

With this agile value, the development initiatives are to value people for whom the system is developed and the information they possess instead of strictly abiding by sets of pre-described processes and tools. It, thus, puts more emphasis on collaboration between customers and members of the development team. The policy frameworks in the analysed documents, largely guides otherwise. There is an excessive emphasis on processes and tools during software development. As it is illustrated through Table 3, these shortcomings have been noticeable in all the phases of software development. Data is presented in relation to three basic stages of software development: *requirements elicitation, design* and *development.*

During requirements elicitation, the analysis shows that there are specific mentions of interviews as a method for requirements gathering. This promotes interaction with the customer, congruent to the agile values requirements. However, there are two sets of constraining guidelines: the need to know the entire set of requirements beforehand, and the requirements for following strict processes. In Tanzania, for example, the standards for developing government software require that both business and system requirements are all gathered before programming starts.

**Table 3: Assessment of the agile value on individuals and interactions over processes and tools**

| Development stage | Favourable guides | Constraining guides |
|---|---|---|
| Requirements Elicitation | The mention of interviews as a method for data collection | (1) Insist on understanding an entire set of requirements at the beginning<br>(2) They describe the details of processes to be followed |
| System Design | | (1) Specification of the architecture<br>(2) Specification of Security Design |
| Development | An emphasis on testing the different units | (1) Specification of tools to be used<br>(2) Specification of styles |

Section 2.1.2.3 of this document lists nine (9) items which must be attended to in the process. Some of them describe processes which “should be followed” while others describe the kind of information to be collected. This does not leave much room for the developer and customer to interact and explore other options as the project context may dictate and, in particular, as much as advocated by the agile methods. The Tanzanian document goes further to specify that “the specific intended use of the system to be developed must be analysed to specify systems requirements”. With this, it goes on to list four components to be part of the requirement analysis.

The same situation befalls the design phases of software development. Main documents from both Kenya and Tanzania, for example, specifically require that the entire architecture of the system be established beforehand, and they go ahead to even specify the contents of the architecture. It goes further by listing five (5) instructions on the formats of user passwords, even though different systems have different security requirements. There have also been specific instructions to development teams with regards to the programming activities. Documents 1 and 3 from Tanzania, for example, specify the coding style to be used. Rwanda 1, on the other hand, requires that the platforms and programming languages for use in developing government software systems need to be approved by one particular agency. *Kenya 1* has also gone further to specify the tools. Annex five of *Kenya 1* emphasises that XML should serve as the universal and primary standard for the exchange of data between all the information systems relevant for administrative purposes.

### *B. Working software over comprehensive documentation*

The agile coalition intended for this value to remove the overheads of producing volumes of documents denying the development phase of time and manpower resources. Much of the analysed documents, though, were very much insistent on producing more than enough documents in the whole process. Document 1 from Tanzania, for example, requires developers to come up with eleven (11) documents in the process of software development. There are three (3) heavy documents to be produced before the start of development and three (3) more during software development. They are heavy because each of these documents had an elaborate list of contents to be part in it. In Kenya, Document 3 requires developers to produce eight (8) documents. Table 4 provides examples of documents required during software development in public institutions, examples from Kenya and Tanzania.

**Table 4: Required documents in Tanzania and Kenya**

| | **Tanzania** | **Kenya** |
|---|---|---|
| *Requirements Elicitation* | 2 | 3 |
| *System Design* | 1 | 1 |
| *Development* | 5 | 1 |
| *Post Implementation* | 2 | 3 |
| **Total** | **10** | **8** |

This number only includes those compulsory documents following the language used in the guiding documents.
In Tanzania, those among the "minimum set of documentation" were included, while in Kenya, Document 3 states that the public institutions "shall ensure that all systems have the following documentation". Table 4 merely provide examples but there are many other requirements for documentation. *Rwanda 1* in section 5.1 clearly states that "all systems should be documented in five viewpoints including the enterprise viewpoint (describe purpose, scope and processes), the information viewpoint (determines the structure and semantics of the system's information), the computational viewpoint, the engineering viewpoint, and the technology viewpoint."

### *C. Customer collaboration over contract negotiation*

This value is in favour of working closely with the customer and, in the process, discovering the details of requirements as opposed to having formal and binding agreements before actual work is experienced. The guiding documents in the three countries included in the analysis seem to very much favour agreements between users and the developers before development. In *Tanzania 1*, for example, section 2.1.2.2 (ix) requires the developer to "Obtain sign-off and approval" after requirements have been gathered. Even after requirement analysis, section 2.1.3.6 of the guidelines instructs that there is also a requirement for signing off. The section clearly states; "Upon successful completion of the review(s), a baseline for requirements of the application must be established and formal sign off must be obtained." The guidelines also emphasise on these agreements in the design phase mandatory documentation. There is a need to document the architecture of the system as well as "the detailed design of the system must be documented." In *Kenya 3,* there is also a strong emphasis on the functional department to approve the requirements.

### *D. Responding to change over following a plan*

This value aims at discouraging over reliance on long established plan in the course of software development. The value, instead, emphasises interactive collaborations between software developers and their clients which would gradually reveal what the client truly wants. For development teams, this would also mean having flexible and self-organising teams. Generally, the detail with which the guidelines instruct does not leave much room for flexibility to accommodate changes in the process of software development. The level of details in *Tanzania 1*, for example, does not stop at the architectures but to the details of requirements which are to be documented.

## Discussion

The study aimed at investigating how the policy frameworks for software development, in the selected countries, influence the use of agile methods in public institutions. To that end, the study uncovered a few characteristic features of the policy frameworks which influence agility in developing government software systems. The first aspect is the micromanagement of the software development processes by higher levels, beside the functional unit that is going to use the system. In light of the five metrics for measuring agility in public institutions as (Gong & Janssen, 2010) propose, this micromanagement is detrimental to the use of agility in public institutions. While it might seem to improve on the quality of the resulting application, it leads to poor throughput, poor response time and case handling time. This is because of the need to involve higher authorities in much of the processes. Where the decision for a programming language and platform has to be approved by a national body, like in Rwanda, the development activities are likely to be slowly attempted meaning low throughput and response time.

The second aspect is that of not entirely recognising agile methods in the guidelines. There are specific mentions of agile methods in the guidelines, as in Kenya, or lumped in the "iterative methods", as in Tanzania but, at large, the guidelines seem to suggest the use of waterfall-based methods. This lack of recognition of agile methods is likely related to the limited use of agile methods. This is likely why several researchers propose to promote the use of agile methods by starting with seeking top management support (Pinheiro et al., 2008, 2009; Turecˇek et al., 2010; Vacari & Prikladnicki, 2015). There is significant evidence showing the advantages of using agile methods for software development, some of which were discussed in section II. Efforts need to be put to assist public institutions adopt them for their software needs. The third characteristic is that of assuming uniform development environment. Software in public institutions have a wide array of differences between them. From their sizes, complexity to security sensitivity. It is not a feasible thing to have all the software development be met through common detailed guidelines.

Generally, examining agile specific values, the policy frameworks have not been much in support of agility. On the first agile value where interactions are encouraged, as opposed to processes and tools, the guiding instructions in the policy frameworks emphasise on the processes and tools. They emphasise on the detailed knowledge of the requirements, including the description of the system to be developed. Agile methods are built on the principle of flexibility and ability to manage change. (Conboy, 2009) describes flexibility as the ability to create change, react to change and "proaction" in advance of change. This ability to manage change is what determines agility in software development. With the strictness in defining all the requirements in advance and the detailed specifications of system functionalities, developers can hardly manage to learn through changes. This is also true for the rest of the values. For example, wanting developers to produce eleven documents before starting to program does not leave them with much flexibility. In general, governments in the studied countries run the risk of facing the challenges caused by inflexible software development processes; suchchallenges include producing unstable software characterised by frequent bugs and failures, as observed by (Pinheiro etal., 2008, 2009). In order to produce stable and good quality software systems that are likely to increase the satisfactionof citizens when accessing government services, government organisations should institute deliberate mechanisms to use agile methods during software development (Pinheiro et al., 2008, 2009, 2010; Patanakul & Rufo-McCarron, 2018).

## CONCLUSIONS AND RECOMMENDATIONS

To fulfil country-specific needs, enable compliance across government institutions, as well as maximising value for money, some African governments have issued guidelines andstandards for development of government software systems. However, despite the good intentions behind these guidelines and standards, still, they can sometimes hinder creativity and innovation among team members, producing poor quality products that can be hardly sustained. At the same time, ASDMs enable creativity and innovation during software development by allowing teamsto change according to project-specific dictates. Despite such benefits offered by agile methods, the literature lacks evidence regarding if and how agility is supported by software development guidelines and standards issued by various African governments.

Through the review of documents conducted to determine if and how agility is supported, a marked lack of support for agility in the guidelines and standards for developing government software systems was found across the three East African countries. As well, it was learned that the standards and guidelines proposed by the three countries embrace excessive micromanagement during software development, leaving little or no room for creativity and innovationamong members of software development teams. Besides, it was found that the guidelines and standards assume uniformity across projects, when there can be context-specific issues across different projects.

Based on the findings of this study, it is recommended to review government software development guidelines and standards to accommodate the following, among other things:

- Strengthen aspects that encourage continuous collaborations between clients (government institutions) and development team members. In particular, it should be possible for development teams to embrace agile approaches and techniques that maximize customer collaboration and hence developing software systems that actually take careof clients' requirements. For requirements engineering, for example, behaviour driven development (North, 2006) is an agile technique that can help both developers, testers, and customers to collaboratively specify correct requirements for software systems.
- Enable development teams to produce just enough documentation so that they can spend more time on developing software systems that actually add value to government institutions.
- While agile development does not mean complete absence of documentation,

attention should be given to only extremely necessary documentation, which can vary depending on project-specific dictates. Development teams for the specific software projects should be able to collaboratively determine what needs to be documented.
- Institute and practice agile techniques for project planning.

Additionally, in pursing the above recommendations, government authorities in charge of software guidelines and standards can draw lessons from successful implementations of agile methods in other government institutions. Nonetheless, this study has only focused on the analysis of documents stipulating the guidelines and standards fordeveloping government software systems; it has not uncovered empirical evidence regarding how these guidelines and standards are used in practice by development teams. Thus, part of the future work is to study how guidelines and standards are actually used by teams, to uncover lessons that could inform policies for software development in African government institutions.

## REFERENCES

Abrahamsson, P., Salo, O., Ronkainen, J., and Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439*.

Alleman, G. B., Henderson, M., and Seggelke, R. (2003). Making agile development work in a government contracting environment-measuring velocity with earned value. In *Proceedings of the agile development conference, 2003.adc 2003* (pp. 114–119). https://doi.org/10.1109/ADC.2003.1231460

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). Manifesto for agile software development. *Software development* **9**(8): 28-35.

Conboy, K. (2009). Agility from first principles: Reconstructing the concept of agility in information systems development. *Information systems research*, **20**(3): 329–354. https://doi.org/10.1287/isre.1090.0236

Dikert, K., Paasivaara, M., and Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, **119:** 87–108. https://doi.org/10.1016/j.jss.2016.06.013

Dingsøyr, T., Nerur, S., Balijepally, V., and Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of systems and software, 85.6 (2012): 1213-1221*. https://doi.org/10.1016/j.jss.2012.02.033

Fruhling, A., McDonald, P., and Dunbar, C. (2008). A case study: introducing extreme programming in a US government system development project. In *Proceedings of the 41st annual Hawaii international conference on system sciences (hicss 2008)* (pp. 464–464). https://doi.org/10.1109/HICSS.2008.4

Gebauer, J., and Schober, F. (2006). Information system flexibility and the cost efficiency of business processes. *Journal of the Association for Information Systems*, **7**(3):8. https://doi.org/10.17705/1jais.00084

Ghobadi, S., and Mathiassen, L. (2016). Perceived barriers to effective knowledge sharing in agile software teams. *Information systems journal*, **26**(2): 95–125. https://doi.org/10.1111/isj.12053

Gong, Y., and Janssen, M. (2010). Measuring process flexibility and agility. In *Proceedings of the 4th international conference on theory and practice of electronic governance* (pp. 173–182). https://doi.org/10.1145/1930321.1930358

Jansen-Vullers, M. H., Kleingeld, P., Loosschilder, M., Netjes, M., and Reijers, H. A. (2007). Trade-offs in the performance of workflows–quantifying the impact of best practices. In *International conference on business process management* (pp. 108–119). https://doi.org/10.1007/978-3-540-78238-4_13

McMahon, P. E. (2006). Lessons learned using agile methods on large defense contracts. *Journal of Defense Software Engineering, 19(5)*, 25-30.

North, D. (2006). Introducing BDD. *Better Software Magazine*

Patanakul, P., and Rufo-McCarron, R. (2018). Transitioning to agile software development: Lessons learned from a government-contracted program. *The Journal of High Technology Management Research*, **29**(2): 181–192. https://doi.org/10.1016/j.hitech.2018.10.002

Pinheiro, C., Maurer, F., and Sillito, J. (2008). Moving towards agility in a bureaucratic environment–RUP as a bridge between waterfall and agile processes. *AGILE 2008*.

Pinheiro, C., Maurer, F., and Sillito, J. (2009). Improving quality, one process change at a time. *In 2009 31st international conference on software engineering companion volume* (pp. 81–90). https://doi.org/10.1109/ICSE-COMPANION.2009.5070966

Pinheiro, C., Maurer, F., and Sillito, J. (2010). Improving responsiveness, bug detection, and delays in a bureaucratic setting: A longitudinal empirical iid adoption case study. In *International conference on agile software development* (pp. 214–219). https://doi.org/10.1007/978-3-642-13054-0_21

Tureček, T., Šmiřák, R., Malík, T., and Boháček, P. (2010, June). Energy project story: from waterfall to distributed agile. In *International Conference on Agile Software Development* (pp. 362-371). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-13054-0_39

Vacari, I., and Prikladnicki, R. (2015). Adopting agile methods in the public sector: a systematic literature review. In *International Conference on Software Engineering and Knowledge* https://doi.org/10.18293/SEKE2015-159