The African Journal of Information Systems

Volume 6 | Issue 1 Article 2

January 2014

How FOSS Replaced Proprietary Software at a University: An Improvisation Perspective in a Low-income Country

John Effah
University of Ghana Business School, jeffah@ug.edu.gh

Gideon Abbeyquaye
University of Cape Coast, gabbeyquaye@ucc.edu.gh

How FOSS Replaced Proprietary Software at a University: An Improvisation Perspective in a Low-income Country

Research Paper

Volume 6, Issue 1, January 2014, ISSN 1936-0282

John Effah
University of Ghana Business School
jeffah@ug.edu.gh

Gideon Abbeyquaye
University of Cape Coast
gabbeyquaye@ucc.edu.gh

(*Received August 2013, accepted November 2013*)

ABSTRACT

The purpose of this study is to understand the rationale for and the process of replacing an imported proprietary higher education management software with a locally developed free and open source software (FOSS). Information Systems (IS) research on FOSS and higher education in low-income countries has focused more on teaching and learning. Less attention has thus been paid to the area of management and administration. Also, low-income country IS research on technology transfer has focused more on applications from the high-income world. Less research therefore exists on transfers between low-income countries. To address these research gaps, this study employs improvisation theory and interpretive case study methodology to investigate why and how a low-income country university replaced a proprietary higher education management software from another low-income country with a locally developed FOSS. The findings show that the university did so through improvisation to overcome the rigidity of the proprietary software and benefit from the flexibility of the FOSS. The study offers rich insight into how low-income country universities can deploy FOSS through improvisation to address design-actuality gap with imported proprietary software and also presents implications for research and practice.

Keywords

Free and open source software, proprietary software, imported software, higher education management, low-income country, improvisation, Ghana.

INTRODUCTION

The purpose of this study is to understand the rationale for and the process of developing a free and open source software (FOSS) to replace an imported proprietary higher education management software in a low-income country university. Proprietary software is developed and owned by an individual or organization that exercises control over it and restricts users from copying, modifying, and redistributing to others (Lungo and Kaasbøl, 2007). Conversely, FOSS is a free software with user accessible source code and a license that allows modifications, copying, and re-distribution to others (von Hippel and von Krogh, 2003, Khelifi et al., 2009). Unlike proprietary software, FOSS does not restrict users from changing the source code or further transfer (Aksulu and Wade, 2010, Perens, 2005). The higher education sector is important for every country to participate in and benefit from the growing knowledge society (Ismail, 2008). To be effective and efficient, higher education institutions require reliable information systems (IS) to support management and administrative functions. The significance of IS for higher education management is signaled by the limited but growing literature on the subject (Tatnall et al., 2009). One strand of this research that is increasingly attracting the attention of researchers, practitioners and governments in low-income countries is FOSS. Contrary to proprietary software, FOSS offers a license that grants users the right to copy, modify, and redistribute without commercial, legal, and technical restrictions (Khelifi et al., 2009).

FOSS offers opportunities for low-income country higher education institutions to address financial, legal, and technical restrictions with proprietary software, especially those imported from the high-income world (Khelifi et al., 2009). Nevertheless, FOSS research on low-income country higher education thus far has focused more on teaching and learning (e.g. Mavengere and Ruohonen, 2010, Mengesha, 2010a, 2010b). Less empirical research exists on higher education management and administration which significantly support teaching and learning functions. In general, IS research on higher education management in low-income countries remains limited (Semeon et al., 2010). Moreover, higher education IS research on technology transfer in low-income countries has focused mainly on systems from the high-income world (e.g. Pscheidt, 2011). As a result, less is known from empirical studies on transfers between low-income countries, especially in Africa.

In view of these research gaps, this study intends to extend FOSS research on higher education in low-income countries to the area of management and technology transfer between low-income countries. The key research questions that motivated the study therefore concern why and how a low-income country university would replace a higher education management proprietary software with a locally developed FOSS. Addressing these questions is significant, especially to higher education management in their quest for knowledge on how to choose between proprietary and FOSS applications. The findings can also offer insight to researchers and practitioners on issues concerning technology transfer between low-income countries. This study therefore addresses the research questions by employing interpretive case-study (Walsham, 1995, 2006) as a methodology and improvisation theory as analytical lens (Weick, 2001) to understand how an African university in the low-income country context of Ghana developed and deployed a higher education management FOSS application to replace an imported proprietary higher education management software from South Africa, another low-income country.

The rest of the paper is organized as follows. The next section reviews the literature on free and open source software, low-income countries, and higher education. The following section introduces and discusses improvisation theory as the analytical lens for the study. The section after that presents the research methodology. The case study report is then presented as the research findings. The discussion follows the findings and the paper concludes with its contribution, implications for research and practice, as well as recommendations for future work.

FREE AND OPEN SOURCE SOFTWARE, LOW-INCOME COUNTRIES, AND HIGHER EDUCATION

Free and open source software (FOSS) is offered for use at no cost with user rights to copy, modify, and redistribute (von Hippel and von Krogh, 2003). Its free license agreement allows users to change the source code to meet their requirements (Neumann, 2005, von Hippel and von Krogh, 2003). Conversely, proprietary software is under a commercial license agreement that technically and legally restricts users from copying and source code modifications (Lungo and Kaasbøl, 2007). As Lungo and Kaasbøl (2007) point out, owners of proprietary software exercise strict control over its use and modifications. FOSS therefore offers opportunity for users to reduce the total cost of software ownership and to modify the source code to suit their situated and custom information needs. On the contrary, proprietary software is associated with higher total cost of ownership and with legal as well as technical restrictions on modification. Although proprietary software can be customized, vendors prefer to maintain generic source code across their user groups in order to facilitate mass support and upgrades. Proprietary software therefore seldom gets customized during or after implementation. Yet, FOSS provides opportunities for users with the necessary skills and knowledge to customize the software to meet their custom information needs whenever necessary (Câmara and Fonseca, 2007, Janamanchi et al., 2009).

FOSS promises some benefits to low-income countries, including commercial, technical, and legal freedom, especially from foreign vendor lock-ins (Khelifi et al., 2009, Mengesha, 2010b). With FOSS, low-income countries have opportunities to reduce total cost of ownership including licensing fees (May, 2006, Waring and Maddocks, 2005). With no purchasing and licensing costs, low-income countries can freely try FOSS before deciding whether to use it or not (Gallego et al., 2008, von Hippel and von Krogh, 2003). For example, higher education management IS's require large investments from low-income countries (Semeon et al., 2010). FOSS can help low-income countries to reduce the huge financial requirements provided they have the necessary knowledge and skills to customize the software to meet their local needs (Câmara and Fonseca, 2007).

FOSS also provides opportunities for low-income countries to address process incompatibility with imported proprietary software from the high-income world. Most low-income countries depend on software designed in and imported from the high-income world. However, due to contextual differences, it is difficult to get such software to work properly in the low-income country context (Avgerou, 2000, 2008, Braa et al., 2004, Heeks, 2002, Nhampossa, 2005). Heeks (2002) describes the incompatibility problem as design-actuality gap, which portrays a mismatch between foreign-based designs and actual local use context in the low-income world. Proposed options to address the design-actuality gap through improvisation are changing the local actuality to fit the software design or customizing the design to match the actuality (Heeks, 2002, 2005). The customization option calls for changes to source code, which is often not accessible to users of proprietary software. However with FOSS, low-income country users have access to the source code to address design-actuality gap challenges.

Nevertheless, without the required knowledge and skills, low-income countries may not benefit much from the opportunities FOSS promises over proprietary software (Câmara and Fonseca, 2007). Unfortunately, most low-income countries lack the requisite programming skills (Bakar et al., 2012, Piotti and Macome, 2007) to customize FOSS to meet local needs. As Câmara and Fonseca (2007) point out, without the necessary skills and knowledge, access to free software alone is not enough for low-income countries to benefit from FOSS. They also need to develop local knowledge and skills. Low-income countries do, however, have the opportunity to acquire the necessary knowledge on FOSS by joining online communities with global memberships (Khelifi et al., 2009).

FOSS has increasingly become popular among low-income country governments (Khelifi et al., 2009). So far, one sector that has benefited much is health (e.g. Bernardi, 2009, Câmara and Fonseca, 2007, Lungo, 2006, Lungo and Kaasbøl, 2007). Empirical research on FOSS in the higher education sector in low-income countries remains limited. Notable exceptions are Mengesha (2010a, 2010b), Khelifi et al. (2009), and Sanga (2010). These studies focus on teaching and learning technologies. For example, Mengesha (2010a, 2010b) focuses on FOSS implementation in a library context and calls for the need to pay attention to frame congruence without which low-income countries may not realize the potential benefits of FOSS. Khelifi et al. (2009) focuses on operating systems of FOSS with emphasis on learning and knowledge transfer. Their study presents FOSS as an opportunity for low-income country higher education institutions to reduce cost and address performance concerns with proprietary software (Khelifi et al., 2009). In short, the literature on FOSS for teaching and learning in low-income countries presents FOSS as a more cost-effective and flexible alternative than proprietary software. However, research on FOSS and higher education management in low-income countries remains limited and thus forms the focus of this study.

THEORETICAL FOUNDATION: IMPROVISATION

This study draws on improvisation (Weick, 2001) as the theoretical lens (Gregor 2006) to explain the development of a FOSS application to replace a proprietary higher education management software. Improvisation as a construct has received varied definitions (Tan et al., 2010). One common interpretation is the overlapping of planning, design, and development activities to address emergent problems (Cunha et al., 1999, Moorman and Miner, 1998, Weick, 2001). Having originated from organization studies, the concept of improvisation can be likened to emergent techniqnues in works of art like music and drama in which composition and peformance are made to converge for novel outcomes to emerge (Baker et al., 2003). Improvisation theory therefore helps to explain how innovative solutions are made to emerge from concurrent design and execution processes (Baker et al., 2003, Miner et al., 2001). It is considered useful for explaining how organizations respond to changes by iterating through concurrent planning, design, and execution activities (Weick, 2001) .

In practice, improvisation involves simultaneous diagnosing, planning, and execution activities to solve problems (Fisher and Amabile, 2008). It has been described as “a situated action of performance where thinking and action emerge simultaneously at the spur of the moment” (Ciborra, 1999: 78). As a result, improvisation contrasts with conventional planning and execution which are considered as sequential and separate activities (Bergh and Lim, 2008). Conventional planning and execution follow standard plans and designs (Bergh and Lim, 2008); but improvisation is driven by spontaneous, free-form enactment of emergent and overlapping planning, design, and execution (Crossan, 1998). Improvisation therefore accommodates simultaneous planning, exploring, experimenting, and tinkering to allow tailored and situated solutions to emerge (Baker and Nelson, 2005, Barrett, 1988). Adopting improvisation approach means enacting a loose combination of planning, design, and development to create innovations (Cunha et al., 2009). The IS development life-cycle based on improvisation is therefore not determinate and sequential but iterative and overlapping (Orlikowski, 1996). In this study, the development of FOSS to replace imported proprietary software is investigated from an improvisation perspective.

A related concept of improvisation is bricolage (Baker and Nelson, 2005). Bricolage refers to making do with just the resources at hand, which are not necessarily the best known, to solve situated problems or to seize opportunities (Baker and Nelson, 2005). It therefore involves creating solutions from available rather than optimal resources (Cornford et al., 2007). Innovators adopting this approach are called bricoleurs (Weick, 2001). As improvisers, bricoleurs do not follow determinate plans and procedures nor do they use superior resources; they scan their environment for whatever resources are available for use (Garud and Karnoe, 2003). Thus improvisers do not wait for optimal resources before taking initiatives (Cunha et al., 1999, Weick, 1999); they make do with whatever resources they can find within their environment. As a result, bricolage occurs alongside improvisation (Bansler and Havn, 2003).

Some authors view improvisation and bricolage as two separate and unrelated concepts (Bansler and Havn, 2004). However, Cunha et al. (1999) argue that the two concepts are complementary and can therefore be combined to investigate a single phenomenon. They define improvisation as “the conception of action as it unfolds, drawing on available material, cognitive, affective and social resources” (1999 p. 308). Their definition therefore calls for the combination of improvisation and bricolage as a complementary lens (Bansler and Havn, 2004). In sum, Cunha et al. (1999) posit that people tend to improvise through bricolage by drawing on resources at hand (Baker and Nelson, 2005). In line with their position, this study employs improvisation with bricolage as a combined analytical perspective.

Improvisation theory has increasingly become popular in IS research (Walsham and Sahay, 2006, Silva, 2002). It has been used by several researchers (e.g. Orlikowski, 2000, 1996, Ciborra, 1996, Elbanna, 2006, Silva, 2002) to investigate introduction and workarounds of new technologies in organizational contexts. The theory is considered useful for explaining dynamics and responsiveness of system development to continuous changes in organizational environments. From improvisation perspectives, IS development and implementation are viewed as situated activities with flexible trajectories (Elbanna, 2006). As noted in the organizational research literature, people do not approach technology with fixed interpretation; they rather appropriate and make sense of technologies as they unfold (Weick, 2001). Improvisation perspective therefore contrasts with the deterministic view that technology development follows determinate and efficient design and development process without responsive changes. This study considers improvisation as consistent with the FOSS development process that followed flexible and responsive process.

The rationale for choosing improvisation as the theoretical foundation for this study stems from the two constructs of improvisation and bricolage, which we consider as appropriate for explaining the emergent design and flexible process of FOSS development. We therefore consider the theory useful for explaining how open-source tools formed the basis of bricolage for overlapping design and development process to address the design-actuality gap of the imported proprietary higher education management software. This rationale is in line with the view that improvisation is appropriate for addressing and investigating the design-actuality gap of imported software in low-income country contexts (Heeks, 2002). We also consider improvisation as a useful perspective to understand how innovations are tailored to meet custom information needs in a particular context (Baker and Nelson, 2005), as in the example of the case university which was confronted with a design-actuality gap that required emergent and dynamic system development response.

RESEARCH SETTING AND METHODOLOGY

This study forms part of a larger IS research project in higher education management in low-income countries. The current study focuses on the development of a FOSS for student records management in the University of Cape Coast (UCC) in Ghana to replace a proprietary software from South Africa, called Integrated Tertiary Software (ITS). The case is considered significant for research because it offers an opportunity to investigate how a locally developed FOSS replaced a proprietary software from another low-income country. Given the dominant focus of low-income country IS research on technology transfers from the high-income world, this study presents an opportunity to shed light on technology transfer from one low-income country to another.

Research Setting

The study was conducted at the University of Cape Coast (UCC). Established in 1962, UCC is one of the older public universities in Ghana. Its original mandate was to produce highly qualified teachers for secondary, technical and teacher training educational institutions in the country. Over the years, it has expanded into professional programs including business management, medicine, and law. UCC currently offers various degrees programs across bachelors, masters, and doctorates. The current student population stands at 47,000, comprising 17,000 full-time and 30,000 distance education students dispersed across the country.

In 1997, UCC implemented the ITS through the support of the Ministry of Education. After the implementation, in the attempt to get the software to work according to local requirements, the university faced several challenges including design incompatibility. Maintenance and support also became expensive as such activities often required the presence of ITS consultants from South Africa. These challenges triggered the need for a locally developed FOSS to replace the imported proprietary software.

Research Methodology

The study followed qualitative, interpretive case study (Walsham, 1995, 2006, Klein and Myers, 1999) methodology. Interpretive case study research in IS seeks to understand information system phenomena from how participants make sense of their interaction with information technology in real-life organizational and social contexts (Walsham, 1995). The ontological and epistemological position of interpretive research is based on subjectivity (Myers, 2009, Orlikowski and Baroudi, 1991). Interpretive research therefore posits that the reality of a research phenomenon and the knowledge thereof are both socially constructed between researchers and their participants (Walsham, 2006). Consequently, interpretive research does not seek or claim objectivity for its focus, process and output but follows emergent and subjective methodology informed by research participants and their contexts.

This study therefore adopts a subjective view for the FOSS development and sought to understand why and how the university developed the FOSS application to replace the imported proprietary software within a low-income country context. The rationale for choosing qualitative, interpretive case study approach was based on the understanding that it offers an opportunity for in-depth understanding into the design-actuality gap of the ITS proprietary software and how the FOSS came to replace it. This rationale is in line with the established position in the IS literature that interpretive case study promotes rich insight into the research phenomena and participants' interactions with real-life contexts (Myers, 2009, Walsham, 2006, 1995, Klein and Myers, 1999).

Data Gathering

The fieldwork for data gathering was conducted by the second author. The fieldwork occurred over a six-month period from April 2012 to September 2012. Access to the research setting was gained through the second author, who works with the university as a senior systems analyst in the student records and management information section. He also doubles as a part-time lecturer in the department of computer science and information technology. The second author has been involved in the open source project from its inception, through development to implementation as well as maintenance and support. He also worked with the FOSS development team as the leader of the student records and information systems, playing various roles including contributing to systems analysis and design, database design, workflow design, and testing. With this background, the second author gathered data in remote collaboration with the first author.

Data was gathered from multiple sources: semi-structured interviews, informal discussions, observations, documents, and artifact analysis. The second author conducted 23 face-to-face semi-structured interviews. Participants for these interviews included 5 IT staff of the university, 3 members of the consulting firm, and 15 participants from various user groups, including management, administrators, faculty, and students. The interviewees were selected through purposeful sampling (Patton, 1990) based on the researcher's knowledge of those knowledgeable about the phenomenon as well as snowball sampling (Richards and Morse, 2013, Patton, 1990) based on referrals from previous interviewees. The interviews focused on challenges with the imported proprietary software as well as the development and use of the FOSS. Each interview session lasted between 1 and 2 hours. In view of his familiarity with the people and the subject, interviews were not tape recorded; rather, the second author took notes which were written up immediately after each discussion.

Additional data came from his recollection on the implementation and use of the proprietary software and his participant observation of the development and use of the FOSS. He also gathered documentary data from project documents, reports and minutes of project meetings, and websites of the university and the consulting firm. After the fieldwork, the second author conducted continuous follow-up interviews with participants via e-mail and telephone conversation to seek clarification for emerging findings from the analysis as noted below.

Data Analysis

We followed an interpretive approach to data analysis (Walsham, 1995, 2006, Myers, 2009). In view of the understanding that in interpretive research, data gathering and data analysis occur hand-in-hand without clear demarcation (Myers and Avison, 2002), initial data analysis occurred in conjunction with data collection. Detailed analysis, however, followed data gathering. The process involved the two researchers' continuous and intensive reading and reflecting on the gathered data iteratively in light of the research questions and the improvisation and bricolage analytical lenses. Detailed analysis involved identifying concepts related to problems with the proprietary software that triggered the need for the FOSS and the significant and emergent events that occurred during the FOSS development. We also looked for concepts regarding the tools used for developing the FOSS and the benefits it provided for use over the proprietary software. The emerged concepts were used to develop themes which formed the basis for structuring the research findings as presented in the next section.

CASE STUDY FINDINGS

In the late 1990s, Ghana Government institutions responsible for higher education including the Ministry of Education, the Ministry of Finance, and the National Council for Tertiary Education encountered challenges whenever they requested information on student records from the universities. After some unsuccessful attempts to address the problem, the Government identified a higher education management software in South Africa called Integrated Tertiary Software (ITS) as a possible solution. Subsequently, ITS was imported and implemented in the four public universities at the time, namely University of Ghana, Kwame Nkrumah University of Science and Technology, University of Education in Winneba, and the University of Cape Coast (UCC). ITS is a proprietary higher education management software developed in South Africa for the higher education market. The aim of the Government was to promote the use of ICT in the country's higher education management sector to enable generation and dissemination of information to meet internal and external use. ITS therefore became a mandatory software for the public Universities. The rest of this section presents the case study findings on the challenges UCC encountered with ITS and why and how it replaced it with a locally developed FOSS.

Challenges with the Proprietary ITS

UCC implemented ITS in 1997. The initial implementation phase focused on student information systems and was led by the vendor from South Africa as the ITS consultant. The post-implementation use of the software created several challenges largely due to incompatibility between the software design and UCC's practices and processes. One of the key challenges was incompatibility between the ITS student-ID format and that of the university. Whereas the ITS student-ID was based on sequential numbering without categorization for faculty or year of admission, UCC's student-ID numbering required such categorization. For example, ID number ED/1992/001 represents a student of the faculty of education admitted in 1992 and being the first to have reported. SS/1995/003 refers to a student of the faculty of social science admitted in 1995 being the third in the order of reporting.

According to the academic and administrative staff of the university, the original UCC student-ID format was very useful for identifying students by faculty, year of admission and order of reporting. The users were therefore not ready to compromise on changing the format. However, this format was inconsistent with the serial numbering system within the ITS. Yet, when users proposed to the vendor to customize the ITS Student-ID format, the vendor was not prepared to do so with the explanation that:

> "ITS is a standardized software being used in several institutions in several countries, we cannot do this [customization]. You can present a proposal to the User Group Conference for it to be voted on. If it is accepted, then the changes can be done. Otherwise, the status quo remains."

Another problem of incompatibility was the gap in reporting formats and paper sizes. As a result, the standard content, format and paper sizes in the ITS did not fit the reporting requirements of UCC for documents such as transcripts, admission letters, and certificates. The user community expected the IT staff of the university to be capable of changing the software to meet their reporting needs. However, the IT staff had no access to the source code of the ITS, much less the necessary programming language skills to do so. Nevertheless, the purchasing agreement did not include source code and the vendor was not also ready to undertake source code modifications to meet the requirements of the users. The ITS consultant therefore focused solely on configuration issues. Besides this, any post-implementation technical problem required the university to bring in the ITS consultant from South Africa to solve it.

The situation generated some misunderstanding between the user community and the university IT staff. The administrative staff wondered why the IT staff could not change the software to meet their needs. Students complained about delays in getting documents such as introductory and admission letters as well as academic transcripts. The university management was unhappy that any small technical problem required bringing the ITS consultant from South Africa to Ghana to solve it. They considered the overreliance on the ITS consultant for routine support to be overly expensive in terms of international calls (which were then very expensive), airfares, hotel accommodations, and service fees. Nonetheless, the ITS consultant could not address all the issues raised by the users including the student-ID problem. Yet, the user community kept blaming the issue on the incompetence of the university IT staff and their failure to solve the problems.

The IT staff found it difficult to convince the user community and management of the university that their inability to get the software to address their needs was due to inaccessibility to source code. Regardless of how the IT staff explained the problem, the users could not understand and rather viewed the IT staff as incompetent. For some of the problems such as report formats and paper size, the IT staff improvised by exporting data from the ITS into productivity software such as spreadsheets, word processing, and desktop database management systems for workaround solutions. Although this was not the best for the user community, as there were still delays in generating and issuing documents, it reduced the tension between them and the IT staff.

In 2001, some members of the IT staff contacted a local FOSS consultant called IT Consortium (ITC), who approached the university and proposed to develop an open source application to address the design-actuality gap of the ITS for managing student information. The FOSS consultant succeeded in convincing the IT staff that open source application is free in terms of licensing fees and that the IT staff would have unrestrictive access to the source code to solve all technical problems anytime users would request for changes. The university would only pay the FOSS consultant for the initial development service. In view of the restrictions with the ITS and the opportunity for FOSS to address the incompatibilities, the IT staff bought into the FOSS idea and decided to convince management and the users to do so.

Improvisation through FOSS

Together with the FOSS consultant, the IT staff of the university managed to convince the management that the problems with the ITS could be addressed through FOSS development. Once the IT staff highlighted that the software would be free, except for development expenses, and the source code would be accessible to the IT staff to respond to changing information needs of users, management agreed to the FOSS development. Following the agreement, the IT staff and the FOSS consultant formed an improvising project team to develop a FOSS-based students information system to address the problems of the ITS.

Based on bricolage and improvisation approach, the team employed open source development tools, including MySQL database system, PHP development tools, CSS, and HTML, Apache Web server, and Linux operating system to develop an online student information system (OSIS). Linux was used as the operating system platform; MySQL was used to create and manage the database; Apache was installed and configured as the Web server; and PHP, CSS and HTML were used to develop scripts, functions, procedures, and web forms. The FOSS consultant was familiar with the use of such open source tools and had used them to develop software for other clients. Throughout the development and beyond, members of the development team benefited immensely from various open source online communities (for MySQL, PHP, Apache and Linux). Various team members joined and posted emerging issues and problems to such community forums for solutions and they often got useful feedback. As the leader of the university's IT staff on the project, the second author's role during the project included requirement analysis and design. In this capacity, he provided context-based knowledge on user requirements and served as the liaison between the development team on one side and the users and management on the other side.

The project team adopted prototyping as the development approach to ensure that the views of various user groups were captured alongside the FOSS development. By this approach, they kept showing prototypes to the various relevant user groups including management, faculty, students, and administrative staff. As a result, the development team went through several iterations to ensure that varied views from users had been captured. Although they found this process to be tedious, their experience with the rigid nature of the ITS motivated them to do so. After several cycles of prototyping, the team completed the development and proceeded to test the software against user and performance requirements. In situations where some modules were not working to the expectations of a particular group, the development team made the necessary changes to get it to meet such expectations.

When the development team was convinced through the testing that the software was working as expected, they proceeded to the implementation. The implementation included data migration from the ITS. Data migration covered all student and examination records. Once the migration was completed, the users began to use the system on a pilot basis and began appreciating its usefulness. The new system had addressed their problems of incompatibility with the ITS such as the student-ID and reporting formats. For example, the director of academic affairs was very impressed because he could access and monitor students' records in real-time during registration. He expressed his excitement as follows: "this [FOSS] is a marked improvement over the previous system [ITS] where registration was done in batches." Such positive comments from various user groups including the management and administrative staff motivated the development team to quickly move beyond the piloting phase and deploy the software for use by all relevant user groups.

FOSS Application Replaced ITS

After the successful development and testing of the FOSS, the IT staff and the FOSS consultant trained the users to use the new FOSS. All the necessary records were migrated from the ITS to the FOSS. As the FOSS application was Web-based, it was named OSIS (online student information systems). However, the use of the FOSS resulted in some initial resistance from some of the lecturers. With the old system, lecturers recorded student marks on paper for administrators to key into the software.

However, OSIS had online capability for lecturers to enter examination marks themselves. The IT staff had to organize seminars to convince the lectures about the benefits of entering and securing their own marks. The seminar helped most of the lecturers to better appreciate the FOSS and its benefits over the old proprietary software. One influential professor, who later became the vice-Chancellor, became convinced about the quality of the new FOSS over the old proprietary. This lecturer had opposed the FOSS initially and the idea of lecturers entering records directly but later came to support it after realizing that the system could track all marks entered in a secured way. Also, with full backing from management, the lecturers had no option but to use the new software to enter marks online. Currently, lecturers continue to use the system to enter and submit their marks online.

After using the FOSS application for some time, management and users came to understand why with the ITS everything depended on the ITS consultant from South Africa and the IT staff had no control over technical changes. For example, if the registrar wanted an ad hoc report that did not meet the standard form of the ITS, the university had to bring down the ITS consultant from South Africa to do that. However, with the FOSS, the IT staff had become capable to use open source development tools (MySQL, PHP, Apache and Linux) to meet custom and ad hoc information demands. Not too long after the implementation, the IT staff had developed enough experience and capacity to provide full support and continuous development of the FOSS without recourse to the FOSS consultant. They have developed the competence to continuously adapt the FOSS to dynamic user information needs and the university’s policies and procedures. One such example was the change in policy on the number of failed courses required for students to be dismissed. This policy change required redesign of the data structures in the MySQL and the PHP source codes. However, with sufficient competence and knowledge gained from relevant open source forums, the IT staff were able to handle the situation themselves.

For over a decade now, the IT staff have gained enough experience to resolve problems at the source code level and have been participating in online open source forums, logging problems and receiving feedback for resolution. This situation has increased user confidence and trust in them. They consider the FOSS as a far better option than the proprietary ITS, which restricted them from working as improvisers and bricoleurs to address ad hoc technical issues and requests from users. Beyond the initial modules and without the support of the FOSS consultant, the IT staff have continued developing additional modules.. Some of the added modules include online admission letter generation, introductory letter generation, certificate generation, and request tracking system. The IT staff have also developed several ad hoc reports which were not part of the original FOSS and are currently in the process of developing a module for online examination time-tables. A member of the IT staff remarked that:

> “With the previous software, it was impossible for us to develop new reports which were not part of the original [ITS] design... with this software we able to create all new reports requested by users.”

With the above positive developments, the FOSS came to replace the ITS completely and has been nicknamed ‘Agyenkwa,’ meaning saviour. The student ID problem has been solved. Issues with transcripts and admission letters have also been resolved without recourse to any external assistance.

DISCUSSION

This study sought to find out why and how a low-income country higher education institution developed FOSS to replace imported higher education management proprietary software from another low-income country. The findings show that the university did so because it found the proprietary software to be too rigid to meet the custom and changing information needs of users. One key disadvantage of proprietary software is that its source code is closed and therefore inaccessible to users. Consequently, proprietary software is characterized by rigidity during use (Lungo and Kaasbøl, 2007) and limited capability to meet custom and changing needs of users. By exercising control over the source code of proprietary software, vendors can customize the source code to meet local user needs. However, vendors are unlikely to do so in order to avoid complicating future updates and upgrades across their user community. The rigid nature of the proprietary software therefore reduced the capability of the university IT staff to meet emergent and ad hoc user information needs, especially where the modifications required the need to change source codes.

The difficulty of getting proprietary software transferred from one context to work in another is well discussed in the low-income country information systems literature (Heeks, 2002, Avgerou, 2008, 2000, Nhampossa, 2005, Braa et al., 2004). The problem has been attributed to differences between the context of design in the high-income world and the context of use in the low-income world. Heeks (2002) conceptualizes this as design-actuality gap between the high-income and the low-income worlds respectively. However, in the case of the findings of this study, the transfer of the proprietary software was not between a high-income and a low-income country, but between two low-income countries (South Africa and Ghana).

These findings demonstrate that the design-actuality gap is not peculiar to software transferred from high-income to low-income as presumed in the extant technology transfer literature (Heeks, 2002, Avgerou, 2000, Avgerou, 2008, Nhampossa, 2005, Braa et al., 2004). Thus a design-actuality gap can also exist between low-income countries. As in this case, the proprietary software in question was even transferred from one African country to another but yet resulted in a design-actuality gap. Improvisation by changing proprietary software design to fit use context actuality has been identified as one way to overcome the incompatibility (Heeks, 2002). However, given that vendors often want to maintain standard source code across their clientele in order to reduce support and maintenance cost, the consulting firm would hesitate to customize the source code to meet the university's custom requirements. In contrast, FOSS provides flexibility and cost advantage to respond to context-based and changing users requirements (Mengesha, 2010b, Khelifi et al., 2009).

The potential benefits of FOSS over proprietary software can only be harnessed if low-income country organizations possess the required skills and knowledge to customize the FOSS to meet local needs (Câmara and Fonseca, 2007). In the case of the university, the FOSS skills were not initially available. However, the university succeeded in developing such skills through knowledge transfer by working with a FOSS consultant to develop the initial application. Over time, the internal IT staff developed the required knowledge, competence, and skills to cultivate the FOSS application to meet the custom information needs of the university. Lack of technical skills has been identified as a key constraint for low-income country organizations to benefit from FOSS (Piotti and Macome, 2007, Bakar et al., 2012). The findings from this case suggest that low-income country organizations, including higher education institutions, can develop FOSS capability through project-based knowledge transfer from external consultants.

On how the university replaced the proprietary software and thus avoided its design-actuality gap, the findings show that the university employed improvisation and bricolage approaches to develop the FOSS application. However, rather than improvising on the proprietary software by configuring or customizing it in the absence of access to source code, the development team as improvisers and bricoleurs rather adopted new software improvisation. The improvisation was achieved through prototyping and bricolage with freely available FOSS development tools including PHP for interface and functionality design, MySQL for database design, and Linux for operating system. Improvisation has been described as an appropriate approach for dealing with unstructured problems that require immediate solutions (Augier et al., 2001). Heeks (2002, 2005) offers two improvisation alternatives for addressing a design-actuality gap in low-income countries: (1) by modifying the software design to suit business process actuality or (2) by changing the business process actuality to suit the software design. While both are conditioned by the existing software, the findings from this study extend the solution for a design-actuality gap to a third alternative of employing bricolage by using open source development tools and improvisation to develop a FOSS application that can address the problems of proprietary software.

One consequence of the successful FOSS improvisation was that the internal IT staff became empowered to meet the custom information needs of the user community. By having access to the source code, they were able to respond to changing and custom user demands as well as cultivate and extend the software to provide additional functionalities beyond what were initially offered. In sum, the findings demonstrate how closed source, proprietary software can disempower internal IT staff and make them appear incompetent to solve technical problems and how FOSS can re-empower them to be in control and regain their confidence and trust among users. As demonstrated in the findings, the internal IT staff were often helpless in addressing the information needs of various user groups due to the closed nature of the proprietary software. However, by gaining technical control over the software and competence to address the emerging problems through FOSS improvisation, the IT team became empowered and regained the necessary technical control.

CONCLUSION

This study aimed to understand the rationale for and the process of replacing a higher education management proprietary software transferred from one low-income country with a FOSS in another. The findings show the proprietary software was replaced due to its design incompatibility and inflexibility with the use context. In contrast, the FOSS application successfully replaced the proprietary software because of its flexibility and responsiveness to context-based and changing user information needs through improvisation development.

The study adds to the low-income country information systems research in two ways. It is the first to provide rich insight into the development and use of FOSS from improvisation perspective to address the design-actuality gap in higher education management proprietary software. Second, it provides insight into the instance of design-actuality gap of technology transfer from one low-income country to another. From this insight, the study extends the issue of design-actuality beyond the high-income and low-income country domain to the low-income and low-income country domain.

The study offers significant implications for low-income country IS research. First, design actuality-gap is not limited to software transfer between high-income and low-income world only. It can exist with software transferred from one low-income country to another. Second, by the two solutions proposed in the literature for addressing design actuality gap (changing the actuality to suit the proprietary software or changing the design to suit the actuality), improvisation offers a third option to overcome a design-actuality gap by bypassing the proprietary software. Third, the study uncovers an interesting relationship between internal IT staff empowerment and the type of software used (proprietary or open source) in relation to technical support. This interesting relationship calls for further research. Future research can also benefit from investigating institutional impact on FOSS development in low-income country higher education environments.

In terms of practical implication, the study shows that institutions that are stacked with proprietary software problems can improvise through FOSS to avoid incompatibility. Even where the organization does not possess the required technical expertise, they can develop internal FOSS development capability through third party consultancy arrangements and relevant online communities. In view of this implication, the study recommends FOSS for low-income country higher education institutions that have challenges with proprietary software. With FOSS, they can meet their custom information needs and overcome software lock-in and design-actuality gap problems.

The limitations of this study stems from its single case approach and limited focus on student information systems without connection with teaching and learning systems. Future research can benefit from FOSS studies that account for system integration between higher education management systems and teaching and learning systems as well as issues with migration from proprietary to FOSS platforms.

REFERENCES

Aksulu, A. & Wade, M. (2010), "A comprehensive review and synthesis of opensource research", *Journal of the Association for Information Systems*, 11, 576-656.

Augier, M., Shariq, S. Z. & Vendelo, M. T. (2001), "Understanding the context: its emergence, transformation and role in tacit knowledge sharing", *Journal of Knowledge Management*, 5(2)**,** 125-136.

Avgerou, C. (2000), "Recognizing Alternative Rationalities in the Deployment of Information Systems", *Electronic Journal of information Systems in Developing Countries*, 3(7)**,** 1-15.

Avgerou, C. (2008), "Information systems in developing countries: a critical research review", *Journal of Information Technology*, 23(3)**,** 133–146.

Bakar, A., Sheikh, Y. & Sultan, A. B. (2012), "Opportunities and Challenges of Open Source Software Integration in Developing Countries: Case of Zanzibar Health Sector", *Journal of Health Informatics in Developing Countries* 6(2)**,** 443 - 453

Baker, T., Miner, A. & Eesley, D. (2003), "Improvising firms: bricolage, account giving and improvisational competencies in the founding process", *Research Policy*, 32, 255-276.

Baker, T. & Nelson, R. E. (2005), "Creating Something from Nothing: Resource Construction through Entrepreneurial Bricolage", *Administrative Science Quarterly*, 50, 329–366.

Bansler, J. & Havn, E. (2003), Improvisation in Action: Making Sense of IS Development in Organizations, *Proceedings of Action in Language, Organisations and Information Systems (ALOIS).* Linköping, Sweden.

Bansler, J. & Havn, E. (2004), "Improvisation in Information Systems Development", IN Al., B. K. E. (Ed.) *Information Systems Research : Relevant Theory and Informed Practice: Proceedings of IFIP TC8/WG8.2 20th Year Retrospective*. Kluwer Academic Publishers Group.

Barrett, F. J. (1988), "Creativity and improvisation in jazz and organizations: Implications for organizational learning", *Organisational Science*, 9(5)**,** 605–622.

Bergh, D. D. & Lim, E. N. (2008), "Learning how to restructure: absorptive capacity and improvisational views of structuring actions and performance", *Strategic Management Journal*, 29(5)**,** 593-616.

Bernardi, R. (2009), IT Innovation In a Health Information System in Kenya: Implications for a Sustainable Open-Source Software Model in Developing Countries, *10th International Conference on Social Implications of Computers in Developing Countries.* Dubai School of Government, Dubai.

Braa, J., Monteiro, E. & Sahay, S. (2004), "Networks of Action: Sustainable Health Information Systems Across Developing Countries", *MIS Quarterly 28(3),* . 28(3)**,** 337-362.

Câmara, G. & Fonseca, F. (2007), "Information policies and Open Source Software in Developiong Countries", *Journal of the American Society for Information Science and Technology*, 58(1)**,** 121-132.

Ciborra, C. U. (1996), "The platform organization: recombining strategies, structures and surprises," *Organization Science*, 7(2)**,** 103–18.

Ciborra, C. U. (1999), "Notes on improvisation and time in organizations", *Accounting, Management and Information Technologies*, 9(2)**,** 77–94.

Cornford, T., Venters, W. & Zheng, Y. (2007), Agility, Improvisation, or Enacted Emergence. ICIS 2007 Proceedings. Paper 8. http://aisel.aisnet.org/icis2007/8

Crossan, M. (1998), "Improvisation in action", *Organization Science*, 9(5)**,** 593-599.

Cunha, M. P., Cunha, J. V. & Clegg, S. R. (2009), " Improvisation bricolage: a practice-based approach to strategy and foresight", IN Costanzo, L. A. & Mackay, R. B. (Eds.) *Handbook of research on strategy and foresight.* Edward Elgar Publishing.

Cunha, M. P., Cunha, J. V. & Kamoche, K. (1999), "Organizational Improvisation: What, When, How and Why", *International Journal of Management Reviews* 1(3)**,** 299-341.

Elbanna, A. R. (2006), "The validity of the improvisation argument in the implementation of rigid technology: the case of ERP systems", *Journal of Information Technology*, 21(3)**,** 165–175.

Fisher, C. M. & Amabile, T. (2008), "Creativity, improvisation and organizations", IN Rickards, T., Runco, M. & Monger, S. (Eds.) *The Routledge Companion to Creativity*. Routledge.

Gallego, M. D., Luna, P. & Bueno, S. (2008), "User acceptance model of open source software", *Computers in Human Behavior*, 24(5)**,** 2199-2216.

Garud, R. & Karnoe, P. (2003), "Bricolage versus breakthrough: distributed and embedded agency in technology entrepreneurship", *Research Policy*, 32, 277-300.

Gregor , S. (2006), "The Nature of Theory in Information Systems", *MIS Quarterly*, 30(3)**,** 611–642.

Heeks, R. (2002), "Information systems and developing countries: failure, success, and local improvisations", *The inforrmation Society*, 18, 101-112.

Heeks, R. (2005), "e-Government as a Carrier of Context", *Journal of Public Policy*, 25(1)**,** 51-74.

Ismail, N. A. (2008), "Information technology governance, funding and structure: A case analysis of a public university in Malaysia", *Campus-Wide Information Systems*, 25(3)**,** 145-160.

Janamanchi, B., Katsamakas, E., Raghupathi, W. & Gao, W. (2009), "The State and Profile of Open Source Software Projects in health and medical informatics", *International Journal of Medical Informatics*, 78(7)**,** 457–472.

Khelifi, A., Talib, M. A., Farouk, M. & Hamam, H. (2009), "Developing an Initial Open-Source Platform for the Higher Education Sector—A Case Study: Alhosn University", *IEEE Transaction on Learning Technologies*, 2(3)**,** 239-248.

Klein, H. K. & Myers, M. D. (1999), "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems", *MIS Quarterly*, 23(1)**,** 67-93.

Lungo, J. (2006), Critical issues associated with adoption and use of open source software in public sector: insights from Tanzania, *ECIS 2006 Proceedings.* http://aisel.aisnet.org/ecis2006/167

Lungo, J. & Kaasbøl (2007), Experiences of open source software in institutions: cases from Tanzania and Norway *Proceedings of the 9th International Conference on Social Implications of Computers in Developing Countries,* . São Paulo, Brazil.

Mavengere, N. B. & Ruohonen, M. J. (2010), "Using Open Source Software for Improving Dialog in Computer Science Education - Case Mozambique University", IN Tatnall, A., Kereteletswe, O.C. & Visscher, A (Ed.) *Information Technology and Managing Quality Education. 9th IFIP WG 3.7 Conference on Information Technology in Educational Management*.

May, C. (2006), "The FLOSS alternative: TRIPs, non-proprietary software and development", *Journal Knowledge, Technology & Policy*, 18(4), 142-163

Mengesha, N. T. (2010a), "The Role of Technological Frames of Key Groups' in Open Source Software Implementation in a Developing Country Context," ", *Electronic Journal of Information System in Developing Countries*, 43(1)**,** 1-19.

Mengesha, N. T. (2010b), "Technology Capacity Development through OSS Implementation: The Case of Public Higher Education Institutions in Ethiopia", *The African Journal Of Information Systems*, 2(1)**,** 1-24.

Miner, A. S., Bassoff, P. & Moorman, C. (2001), "Organizational Improvisation and Learning: A field Study," *Administrative Science Quarterly*, 46(2)**,** 304-337.

Moorman, C. & Miner, A. S. (1998), "The convergence of planning and execution: improvisation in new product development.", *Journal of Marketing*, 62(1)**,** 1-20.

Myers, M. D. (2009), *Qualitative Research in Business & Management*, London, Sage.

Myers, M. D. & Avison, D. (Eds.) (2002) *Qualitative Research in Information Systems: A Reader,* London, Sage Publications.

Neumann, P. G. (2005), "Attaining Robust Open Source Software", IN Feller, J., Fitzgerald, B., Hissam, S. A. & Lakhani, K. R. (Eds.) *Perspectives on Free and Open Source Software. Cambridge:*.

Nhampossa, J. L. (2005), Re-Thinking Technology Transfer as Technology Translation: A Case Study of Health Information System in Mozambique *PhD Thesis.* University of Oslo, Oslo.

Orlikowski, W. (1996), "Improvising Organisational Transformation Over Time: A situated change perspective", *Information Systems Research* 7(1)**,** 63–92.

Orlikowski, W. (2000), "Using technology and constituting structures: A practice lens for studying technology in organizations", *Organization Science*,, 11(4)**,** 404–428.

Orlikowski, W. & Baroudi, J. J. (1991), "Studying Information Technology in Organizations: Research Approaches and Assumptions", *Information Systems Research*, 2, 1-28.

Patton, M. Q. (1990), *Qualitative evaluation and research methods*, Beverly Hills, CA:, Sage.

Perens, B. (2005), "The emerging economic paradigm of Open Source", *First Monday*, Special Issue 2.

Piotti, B. & Macome, E. (2007), "Public healthcare in Mozambique: Strategic issues in the ICT development during managerial changes and public reforms", *International Journal of Medical Informatics*, 76(1)**,** 184 – 195.

Pscheidt (2011), "Structurational analysis of cross-cultural development of an academic registry information system in Mozambique", *Information Technology for Development*, 17(3)**,** 168-186.

Richards, L. & Morse, J. (2013), *README FIRST for a User's Guide to Qualitative Methods*, Sage.

Sanga, C. (2010), A technique for the evaluation of free and open source e-learning systems, *PhD Thesis.* University of the Western Cape.

Semeon, G., Negash, S. & Musa, P. (2010), The Success of Student Information Management System: The Case of Higher Education Institution in Ethiopia, *AMCIS 2010 Proceedings. Paper 278.* http://aisel.aisnet.org/amcis2010/278

Silva, L. (2002), "Outsourcing as an improvisation: A case study in Latin America", *The Information Society*, 18(2)**,** 129–138.

Tan, B., Pan, S. L., Chou, T.-C. & Huang, J.-Y. (2010), Enabling Agility through Routinized Improvisation in IT Deployment: The Case of Chang Chun Petrochemicals, *ICIS 2010 Proceedings. Paper 225.* http://aisel.aisnet.org/icis2010_submissions/225

Tatnall, A., Visscher, A., Finegan, A. & O'mahony, C. (Eds.) (2009) *IFIP International Federation for Information Processing: Evolution of Information Technology in Educational Management*, Boston, Springer.

Von Hippel, E. & Von Krogh, G. (2003), "Open source software and the "private-collective" innovation model: Issues for organization science," *Organization Science*, 14(2)**,** 209-223. .

Walsham, G. (1995), "Interpretive case studies in IS research: nature and method", *European Journal of Information Systems*, 4, (74-81).

Walsham, G. (2006), "Doing interpretive research", *European Journal of Information Systems*, 15(3)**,** 320-330.

Walsham, G. & Sahay, S. (2006), "Research on information systems in developing countries: Current landscape and future prospects", *Information Technology for Development*, 12(7-24.

Waring, T. & Maddocks, P. (2005), "Open Source Software implementation in the UK public sector: Evidence from the field and implications for the future", *International Journal of Information Management*,, 25(5)**,** 411-428.

Weick, K. (1999), "Sensemaking as an organizational dimension of global change", IN Dutton, J. & Cooperrider, D. (Eds.) *The Human Dimensions of Global Change*. Sage, Thousand Oaks, CA.

Weick, K. (2001), "Organizational redesign as improvisation", IN Weick, K. (Ed.) *Making Sense of the Organization*. Oxford, Blackwell Publishers.