17

Using JAD to Bridge the Design-Reality Gaps; a Major Cause of IS Projects' Failures in the Developing Countries

Euphraith Muthoni Masinde, School of Computing and Informatics, University of Nairobi – Kenya, muthoni@uionbi.ac.ke/muthonimasinde@yahoo.com, P.O. Box 15697 Nairobi, 00100 GPO Kenya

Information Systems (IS) projects failure is 'a gap between what the users expect from an IS and how well these expectations are met by the perceived performance of the delivered system'. IS projects fail more than they succeed. IS failure rates in the Developing Countries (DCs) are much higher than those in the Industrialised Countries (ICs) because among other reasons, the gaps tend to be exaggerated by the huge difference between the ideas/IS projects and the political /behavioural realities in the DCs. These chronic failure rates have continued to place the DCs on the wrong side of the digital divide, turning IS projects and ICTs in general into a technology of inequality. Solution: employment of Joint Application Development (JAD); a software development methodology that will involve the stakeholders in the entire process of IS implementation. This paper explains how JAD can be used to eradicate most of the causes of IS projects' failures in the DCs using the University of Nairobi case study. The CHAOS Ten Success factors have been employed to analyze data for nine IS projects.

Introduction

According to research by the Standish Group, on average, only 16% percent of all IS projects in the world 'succeed'. Despite this lame-duck status, IS permeate just about every aspect of life in the ICs and their failure cause havoc everywhere (Donaldson and Jenkins, 2000). Incomplete requirements specifications and lack of user involvement are the two most common factors that cause these projects to fail or be cancelled. Not involving the users will mean that the final product is what the developer(s) thought the users needed rather than what the users actually needed. Several authors have referred to this situation as 'gaps'. Heeks(2002) called them design-reality gaps while Linda (2000) called them expectation-perception gaps. Lyytinen and Hirschheim (1987) referred to this situation as Expectation Failure. One way of bridging or at least reducing these gaps is to employ a software development methodology that actively involves users in the entire process; this is where JAD comes in.

According to Heeks (2002), most Information Systems projects in DCs fail totally or partially. IS failure of course is not a DCs' malaise, for instance, in the USA, in 1995, the cost effort ploughed into computer projects that were subsequently cancelled plus the cost of project overruns were estimated by the Standish Group at a spectacular $140 billions! No one can accurately give the figures that measure the software failure rates in the DCs; this is not even possible in the ICs. Some approximates in the latter are that ¼ projects totally fail, 1/3 to 3/5 partially fail while the rest succeed (The Economist 2000). There is no known proof that the figures in the DCs are higher or lower than these but practical reasons such as lack of technical and human infrastructure may push the failure rates in the DCs upwards. The reality of the failure rates is more mundane and lack of literature in this area and lack of IS projects' evaluations (the little found is on individual projects' case studies) makes the study of failure/success rates of IS projects in the DCs more complicated. Sahay and Walsham (1995) summarises the situation of IT in the developing countries as follows: 'The process of IT use in the Developing Countries is a complex phenomenon and it typically involves actors at various levels. It is important to study the interaction of these actors on the process of IT implementation and use" (p. 118)

JAD has been in use for close to 30 years now and it is a methodology aimed at involving all key stakeholders [1] in the entire development process. JAD has been refined over the years incorporating features such as electronic meeting systems by Carmel et al (1992); that differentiates Electronic-JAD (E-JAD) from Traditional-JAD(T-JAD). Various software tools available today can be used in automating most of the tasks [2] that are carried out during JAD sessions. The studies of the uses of JAD have also been carried out, such as in the 'an exploratory study of JAD in information systems delivery' by Davidson (1992)

Measuring success of a IS project is a very difficult task and whatever parameters one uses; no absolutely satisfactory results may be attained. Authors of various pieces of literature in this area have proposed different ways of measuring success. One way is by looking at the technical properties of the system, the fit between organization needs and system capabilities and also customers' satisfaction proposed by Christine and Paul (1999). Whatever the criterion used, in this paper, systems will be classified into three success scales: Successful, Challenged/Partial Failure or Failed (Standish Group).

The University of Nairobi adopted the JAD as a methodology in 1999 and has since then reversed the trend of massive IS projects' failures that had existed since early 70s. The success is quantified by the very high successful rate of IS projects that have been put in place so far.

JAD – An Overview

Definition

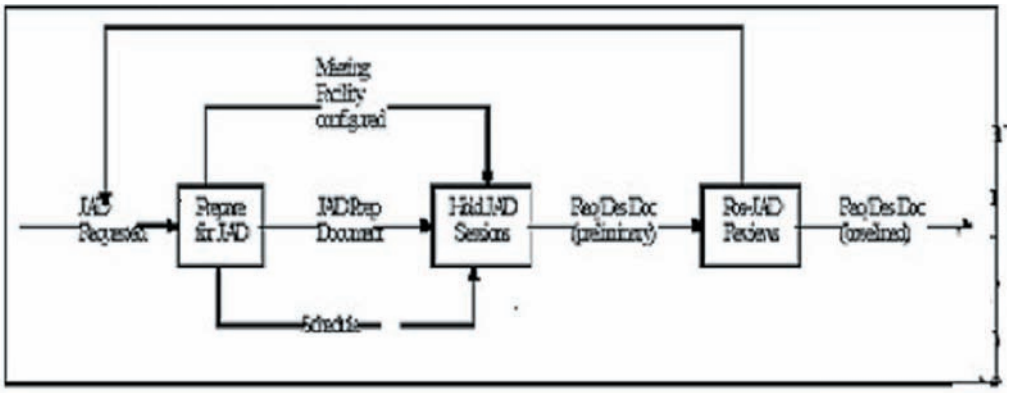
Joint Application Design (JAD) is a structured process in which users, managers, and analysts work together for several days in a series of intensive meetings to specify or review systems requirements. The systems development personnel at IBM developed it in the late 1970s. JAD has evolved over time to include other phases (design, coding,

etc) of software development, hence acquiring the name Join Application Development. It is the latter definition that is used in this paper. There is a close correlation between JAD and Rapid Application Development (RAD); to some extent, JAD is a tool for RAD success (Hoffer et al). Despite the different definitions and forms that JAD has acquired, the key characteristic of JAD is the facilitated sessions. Each JAD session has well-defined objectives, detailed agenda and guidelines, visual aids and final documents containing all the decisions made by the group. JAD should be used after the high-level requirements have been developed and should consist of 3 major phases as shown in Figure 1 proposed by Allan and Mary , 2000. Each session is supported by a session leader, a facilitator, a scribe, technical specialists (e.g. in Database Design and user interface design) and domain expert

Prepare for JAD Hold JAD Sessions Pos-JAD Reviews JAD Prep Document Req/ Des Doc (preliminary) Req/Des Doc (baselined) JAD Requested Meeting Facility configured Schedule Figure 1:

High-level process diagram of the JAD process

Figure 1: High-level process diagram of the JAD process



Advantages/Disadvantages of JAD

Numerous articles, case studies, and other related studies have shown a number benefits of using JAD. These have been summarised by Alan on the website (http://www.carolla.com/wp-jad.htm) as: saves time, eliminates process delays and misunderstandings and improves system quality; It is one of the best ways to reduce function creep, most of which results from poor initial requirements. By properly using transition managers, and the appropriate users, the typical cultural risk is mitigated while cutting implementation time; It also avoids bloated functionality, gold-plating, and helps designer's delay their typical "solution fixation" until they understand the requirements better; Lays the foundation for a framework of mutual education, separate brainstorming, binding negotiation, and progress tracking; Finally, JAD helps avoid the requirements from being too specific and too vague, both of which cause trouble during implementation and acceptance.

JAD has its own share of disadvantages; top on the list being cost. JAD can push the cost of the entire project upwards in terms of people's time and money. Bringing many people from different levels together in a room may mean that some may not give their ideas; cannot challenge their bosses and the same people may later turn around and reject the system (Hoffer et al).

JAD Principals

Authors suggest general principles of JAD such as to involve all the major stakeholders or stakeholders' representatives, ensuring that JAD teams have support from upper management, involving a technical facilitator with skills in both systems analysis and group dynamics and ensure that each stakeholder has a representative empowered with decision-making. Further, each session should be short (2 – 4 hours), each session must produce JAD minutes, which contains attendees' resolutions, action items, and open issues. The facilitator sends copies to all team members and their managers. Defined in most of the literature also are the JAD tasks such as identifying all stakeholders and clarifying executive goal. (Hoffer et al).

Is Success/Failure

Categories of IS Projects' Success/Failure

Success/failure of software projects has been classified by authors Standish Group, Puri et al,2000, Kitiyadisai, 2000, Benjamin, 2001 into three categories::1) Successful – in which most stakeholder groups attain their major goals and do not experience undesired outcomes, 2)Challenged/partial failure – where major goals are unattained or significant undesirable outcomes such as over-budgets and over the time estimates are experienced. This category of projects is difficult to be assessed because the failure/success may subjective. 3)Failed/total failure – is where the project is never implemented, cancelled or implemented and immediately abandoned.

Failure Factors

May (2000) gave some of the major specific causes of software failures as: Poor user input leading to systems that do not meet their needs, Stake holder conflicts; either the 'stake holders' of the system are not well defined or they are not willing to work together, Vague requirements, Hidden cost of going "lean and mean" which means over-reducing the number of employees while maintaining the same deadlines, Failure to plan – 'failure to plan is planning for failure', Communication breakdown between people in the various levels in the software project. Architecture that may not be flexible, late failure warning signals. Lientz and Rea, 1999 analyses the cause so IS projects failures in form a list with 25 reasons (p. 12-14) which all seem to fit in the above categories.

Success Factors

The factors that determine whether a particular project gets into either of the above categories are many (addressing all the factors that lead to software failures). Research as shown that the nature of the project matters too. For example, the size of the project, (smaller projects tend to have a higher success rate than larger ones) and company size (bigger companies have better chances to succeed). All authors in this area agree that the factors that make IS projects successful are not especially technical, e.g. McConnell, 1998. McConnell argues that user involvement is a critical survival skill because it ensures that the users will use/like the final product. He summarises it in the statement; "Ask users what they want, show them what you intend to build and ask them how the like it." CHAOS study identified ten weighted successful factors for IS project (Standish Group):

Table 1 – The CHAOS Ten Success factors

Success Criteria	Points
1. User Involvement	20
2. Executive Support	15
3. Clear Business Objectives	15
4. Experienced Project Manager	15
5. Small Milestones	10
6. Firm Basic Requirements	5
7. Competent Staff	5
8. Proper Planning	5
9. Ownership	5
10. Others	5
Total	100

Information Systems Failure - A Crisis

The Oxford English Dictionary defines crisis as 'a decisive moment, a time of danger or great difficulty, a turning point'. In this paper, IS crisis will be used to refer to all the problems that are encountered in the IS projects. For software industry, IS crisis has been with us since the birth of the industry (software) itself. Pressman, 1994 proposes the use of the term 'chronic affliction' instead of 'IS crisis' because of its longevity and reoccurrence. IS crisis has led to devastating losses that are documented in literature.

There is a large body of literature addressing IS projects failures in the ICs. Some of the case studies include: The London Ambulance Service Computer-Aided Despatch (LASCAD) System Project which is one of the most frequently quoted UK examples of IS failures in recent times (Dalcher and Tully, 2002, Christian Lundestad, 2003, Paul, 1999,Paul, 1995). "On the 27th October 1992, an IS made the lead story on the BBC's nine o'clock news; the new computerised system established at the headquarters of the London Ambulance Service (LAS); LASCAD, failed and that as a result, the lives of 20-30 people may have been lost." (Paul, 1991), (p. 1). The system was introduced in an atmosphere of mistrusts by staff and there was incomplete 'ownership' of the system by the majority of its users. There was disorganisation, low staff morale, friction between

management and the workforce and an atmosphere of hostility towards computing systems (Paul, 1995). Other cases documented by Flowers, 1996 and Glass, 1998 are: the FBI Virtual Case File that was delivered one year late with only 1/10 of functionalities met and with $170 million wasted; Licence Registration System for Washington State Department that was delivered late, $40 million wasted and was never used. Closer to our lives, the Microsoft's new Sender ID technology quarantines unwanted email but it cannot tell the healthy from the sick (IEEE Spectrum January 2006)

Contrary to this, there is no adequate literature (known to the author) on IS projects failures in the DCs. Some of the projects documented include: 1) the Accounts and Personnel Computerisation Project of Ghana's Volta River; the project partially failed because of the lack of involvement of some lower-level staff (Tetty 2000). 2) The Touch Screen Kiosks for remote rural communities in South Africa's North-West province failed soon after implementation. The kiosks were not useful to the local users because the content therein was either not updated or did not make sense to them (Benjamin 2001). 3) Workflow System for a South African tyre manufacturing firm that was never used (Calitz 2000)

Walsham and Sahay (2006) analysed most recent (2000 onwards) literature on IS in the DCs under the topic '*Research on Information systems in Developing Countries: Current Landscape and Future Prospects.*' They classified the literature into 3 categories: 1) Those that addressed how to deal with the challenges facing IS practitioners 2) The ones dealing with the role of technology and 3) Those that proposed suitable theories and methodologies. In the first category, one of the challenges is 'local adaptation and cultivation'. Bringing IS to a new local context involves some implicit elements of cultural transfers and mutual learning. Proposals by authors in this area (Bada, 2000, Ehikhamenor 2003, Makome 2003, and D'Melllo 2003) converge to the point that understanding the local context is crucial in IS implementation. In the second category, the issue of 'standardisation versus localization' was analysed. The authors (Braa and Hedberg 2002 and Thompson 2002) propose localization rather than standardisation. Recommended also is the issue of evaluating the applicability of each technology (e.g. GIS) to make sure that it does not conflict with the local way of perceiving and knowing. In the third category of literature evaluated (Bada 2002, Heeks, 2002 , Braa and Hedberg 2002 and Madon and Sahay 2002), suggested that new IS theories for the DCs were required.

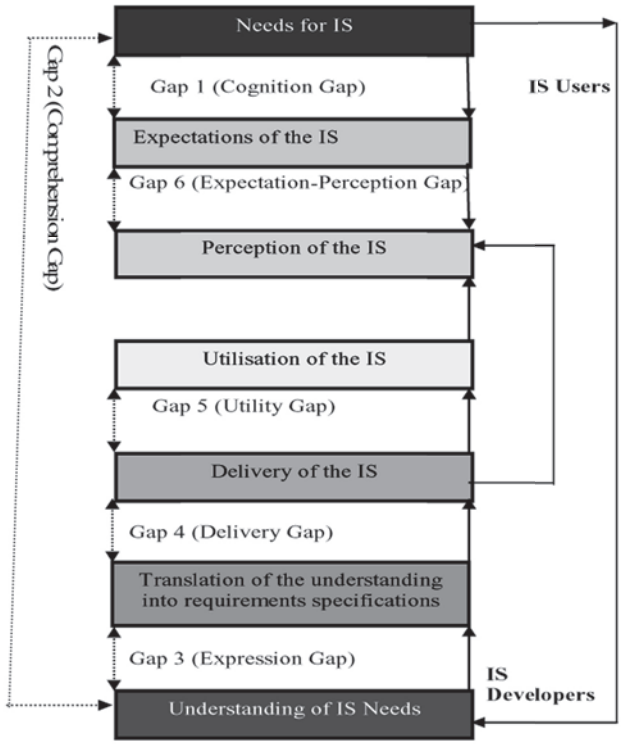
Solution to IS Gaps

In all the above, lack of cohesion between the IS project developers and the stakeholders was the major contributor to the failures. Consequently, what was delivered was not what the stakeholders were all along expecting.

Expectation-Perception Gaps

This expectation-perception gap can be understood and analysed by a gap model that was proposed by Linda, 2000 as shown in figure 2.

Figure 2 A gap analysis model of IS Development



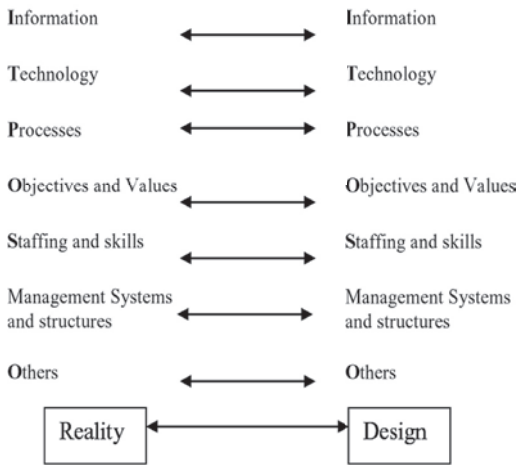
Linda analysed 6 gaps that arise as a result of developers working in isolation from the users/stakeholders: Cognition Gap (Gap 1) – the difference between 'what the users needs' and 'what they think they needs'. It is as a result of users' inability to cognate upon their information needs; Comprehension Gap (Gap 2) – the difference between 'what the user need' and 'what the developers think the user need'; it is the developers' inability to comprehend users' information needs; Expression Gap (Gap 3) – the difference between 'the developers' understanding of users' needs' and 'the translation of developers' understanding into requirements specifications'. This is affected by the developers' mental constructs (e.g. perceptive process, values, ethics, motives, prejudices, intellectual ability, experience, etc); it is caused by developers' inability to translate the perceived information needs of users into requirements specifications; Delivery Gap (gap 4) – difference between 'a system as specified' and 'a system as delivered'; this is the developers' inability to transform specified needs for information provision into systems deliverables; Utility Gap (Gap 5) – difference between 'a system delivered' and 'a system in use'; it is usually as a result of users' inability to utilize the delivered systems to satisfy their information needs; Expectation-Perception gap (Gap 6) – a function of gaps 1, 2, 3 , 4 and 5. Solution: keep Gap 6 closed by closing gaps 1 through 5 by having IS professionals becoming service providers to users rather than the becoming proprietors of the IS.

A similar model was proposed by Lyytinen and Hirschheim (1987). They categorised IS failures into: 1)Correspondence Failure – a management perspective of IS failure where there is lack of correspondence between the objectives set out and the results of evaluating the resultant IS; 2)Process Failure – is the unsatisfactory development performance; the development process cannot produce a workable system or the system is produced but the project runs over budgets in terms cost, time and other resources; 3)Interaction Failure – is the mismatch of the requirements and the resulting IS, a situation that leads to a system that is hardly used; 4)Expectation Failure is a superset of Correspondence, Process and Interaction Failures. Lyytinen, 1998 broadened the notion of Expectation Failure to generalise failures into two: *Development Failure* and *User Failure*.

Design-Reality Gaps

Heeks, 2002 presents these gaps (in the case of DCs) using seven dimensions: Information, Technology, Process, Objectives and values, Staffing and skills, Management and structures and Others; yielding the ITPOSMO mnemonic (figure 3). These factors lead to very huge gaps essentially between the rationality of the software projects design (hard) and the political/behavioural realities (soft) of the DCs' organisations.

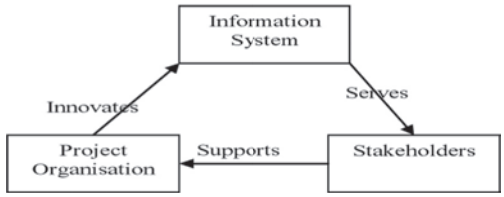
Figure 3. Design-Reality Gaps



Termination Failure

Sauer, 1998 'lightens' the definition of IS failure to only when the development or operation ceases; called Termination Failure He proposed a triangular model with three interacting components: Information System, Supporters and Project Organisation. Beynon-Paul, 1999]suggested the replacement of Supporters with Stakeholders yielding the model in figure 4

Figure 4 – Sauer's Model of IS Development

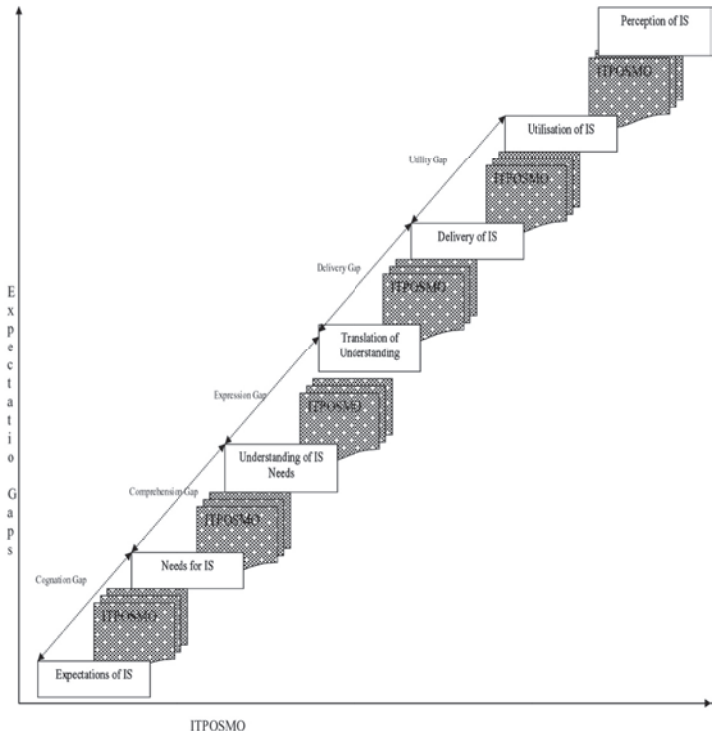


Sauer introduced the concept of flaws that may be corrected within any innovation process at a cost. E.g. program bugs, hardware performance, organisational changes etc. a build up of uncorrected flaws may lead to termination failure.

It is clear from above that most of the IS projects failures are not caused by the technical shortcomings but rather by the social issues around the IS projects. Huy et al, 2006 argues that, to some extent, there might be need to reform IS education to make the learning process effective for IS professionals. The IS professionals may be lacking on the area of 'System Thinking' hence being unable to deal with organizational and social issues of IS projects that they handle.

Putting Linda's and Heeks models together results in the situation depicted in figure 5. In the model, it is clear that the 'expectation gaps' for IS projects in the DCs are so huge because of the 'design-reality gaps'

Figure 5:



Using Jad To Improve Is Projects' Success Rates In The Dcs

Not much literature exists on the assessment of IS projects success/failure in the DCs. The little that exists brings out the fact that failure rates are higher than those in the industrialized countries. Even rarer, is literature on how to deal with the high failure rates. The few approaches to addressing this include theoretical ones (Baruah, 2000 and Barrett, 2001) as well as those based on soft systems ideas. The latter recognizes that social and organizational factors are more likely to determine the success/failure of a software project in the DCs more than elsewhere in the world. An IS project that tries to match values, perceptions and assumptions of the key stakeholders has higher chance of succeeding (Heeks 2002). A balance however need to be reached to avoid a situation where the projects do not change the environment at all in which case it would beat the logic of having the system in the first place.

Local Improvisations

Just like everywhere else in the world, IS projects in the DCs fail more than they succeed because of the gaps between what is developed and what dominant stakeholders wanted. What is unique about the DCs is the fact that the ideas/software systems come from the industrialized countries (totally different environment) making the gaps even wider (figure 5) hence, increasing the failure rates. The solution to averting the trend of software failures in the DCs therefore lies in bridging the above gaps. One proposal by Heeks, 2002 is the use of local improvisations using bi-directional approach; changing local realities to make them closer to the IS projects design; and/or change the IS projects' design to make them closer to the DCs organisational realities. In doing so, the success will still depend on how well the initial requirements are acquired and that the direction of gap-bridging will vary from one project to another. Some proposal of supporting this gap-bridging include: exposing organisational realities, improving local IS capabilities, educating the 'carriers' of the industrialised innovations (such as donors, consultants, ICT vendors, DC personnel trained according to traditional industrialised curricula) on the realities of the DCs and analysing both the 'how' and the 'what' in relation to the software project.

Why JAD?

There are still some loose ends here; how do we formalise the whole process of bringing the stakeholders together? JAD solves this via JAD sessions because they provide both qualitative and quantitative consultation with all the stakeholders of the project in a formal set up.

Based on the CHAOS Ten Success Factors, JAD can be used to achieve a very big percentage of success as follows: In JAD, users (or user representatives) are involved in the software development (20%). JAD sessions involve managers, hence assured of executive support (15%). The representation of most stakeholders during sessions ensures that the business objectives are clearly stated (15%) and that all the stakeholders will feel that they own the software (5%). Important too is the fact that once the requirements are stipulated, they are bound to be firm (not changing always) since most

of the important decision makers are represented (5%). This is already 60% success chance by just adopting JAD!

UoN – CASE STUDY

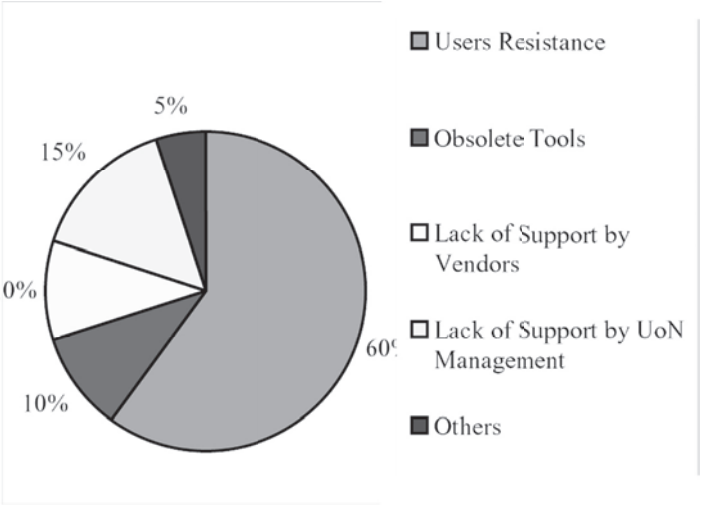
Background Information

The University of Nairobi (UoN) is the oldest, largest and best-established public university in Kenya and the larger Eastern Africa region. Its origin can be traced back to 1951 when its precursor, the Royal Technical College of East Africa was established (www.uonbi.ac.ke). The University of Nairobi is located at the heart of the capital city, Nairobi, with an enrollment of over 30,000 students pursuing diploma, and degree courses in most areas of study. The University's Information and Communications (ICT) Center is in charge of Management Information Systems' (MIS) support among its responsibilities. The University has always operated on some kind of 'semicomputerized' systems until early 90s when these 'systems' could no longer support the University's operations. This was triggered by a revolution in the University, aimed at expanding the University in terms of students' population, increasing the number of programs and relaxing most of the programs regulations to accommodate Module II students (Kiamba, 2003). Most of the software that had been put in place (both in-house developed and off-the-shelf) had just remained unutilized and later discarded. The factors that contributed to this are: - resistance by the end-users, lack of support by the University's top management, lack of support by vendors of these systems and use of obsolete hardware and operating systems. With the pressure from the dynamism in the new way the University was being run, there was urgency to acquire the essential information systems and ensure very high level of success in the implementation of these systems. Hence, the choice of the JAD methodology by the Director of the University's ICT Center. Today, this decision has turned out to be a success story for the University.

IS Failure at the Uon

Among the many causes of software failures mentioned in this paper, user resistance contributed to 60% of reasons as to why software projects at the UoN failed. Other reasons that contributed to the failures are: incompetent staff, because most of the users were not computer literate, lack of support from the University's top managers who did not see the need to fund such systems, use of inappropriate tools such as obsolete operating systems, computer hardware and programming tools and lack of experienced project managers. From document reviews and informal interviews conducted, the failure causes were distributed as follows:

Chart 1: Causes of IS



New Software Development Approach At The UoN

In the early 2000, having realized the critical need of a functioning MIS department at the UoN, the top management put in place a well structured computing unit (the ICT Center) for the University. MIS was then re-established under this center, staffed with very competent project managers and supplied with excellent software development tools. Modular approach to software development was used with Students Management Information System (SMIS) and Human Resources Management Information System (HRMIS) pioneering. Halls Management Information System (HMIS) followed two years later. For each of the above projects, the end users (at all levels) were identified and together with the technical people(systems analysts/programmers, network engineers, database administrators among others), formed a team that worked together on the projects from the projects' initialization to commissioning. To ensure that the top management of the University supported the projects, the relevant managers/ representatives were on several occasions invited to attend some of the teams' meetings. Also, the minutes of the teams' sessions were always copied to these managers to update them on the progress of the projects. Further, out-station seminars were held at intervals of 6 months to update stakeholders on the progress of the projects.

However, developing these systems has not been a smooth ride all through. Most of the systems were completed long after the deadlines and went way beyond their budgets. Quite a bit of user resistance was experienced especially because most of the University employees were still computer illiterate at the time of commissioning the systems. The overwhelming support from the top management especially the Vice Chancellor helped address most of the hiccups. In summary, the systems are a big success and have made a significant mark in the history of the University of Nairobi.

Data collection and Analysis

In determining the success/failure of the IS projects, Sauer's definition of IS project success/failure was used. A sample of 30 users of all the new systems was requested to

fill questionnaires regarding their view of the success/failure of the systems. Informal interviews were conducted for 10 stakeholders of the systems; these were mainly middle level managers (such as the Examination Officers, Finance Officers, etc) and the systems analysts. The questionnaires/interviews used a framework of the CHAOS 10 success factors of measuring success. On average, the following were the results.

Chart 2: UoN IS Projects Success Levels

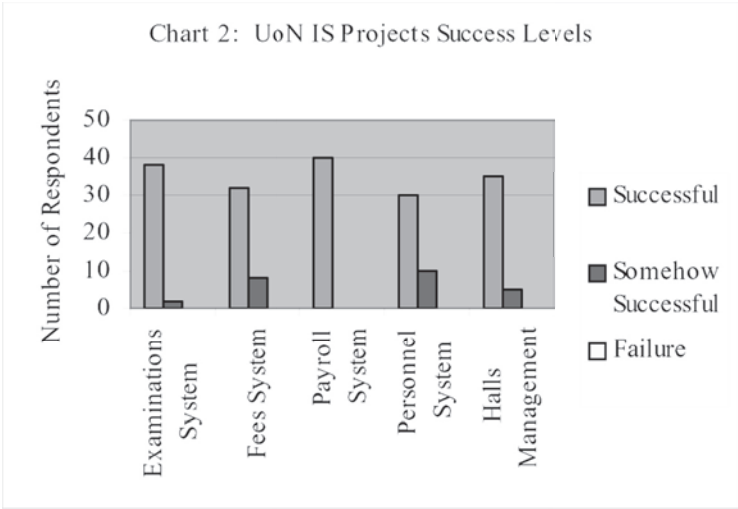
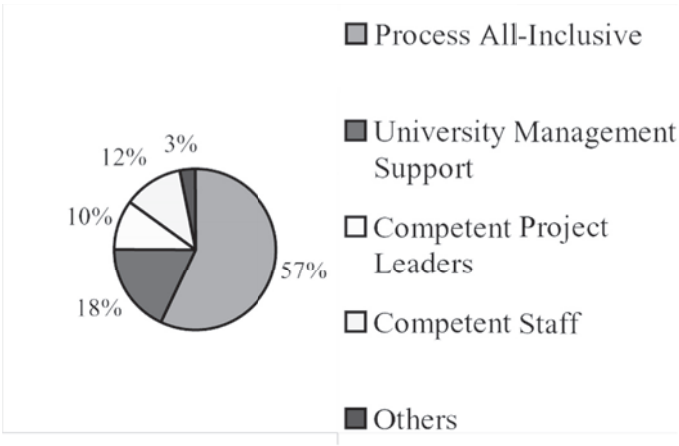


Chart 3: UoN IS Projects



Conclusion

From this paper, it has been shown that more than 60% success of IS projects can be achieved by employing the use of JAD methodology. During JAD sessions, stakeholders are actively involved and their ideas/views incorporated. This is a critical success factor in the DCs. Potential IS 'gaps' are bridged continuously.

Having worked for the UoN, JAD will work for most organisations in the DCs and this may reverse the current trend in the IS industry in these countries. Further quantitative as well qualitative research is required into actual situation of IS failure/ success in the DCs. Also, more 'experiments' to prove that the model proposed in this paper need to be carried out using data from other institutions within the DCs

Footnotes

[1] Key stakeholders in relation to an information system in a given organization will include the top management, the sponsor, the end-users and IT technical people.

[2] Some of the JAD tasks that are automated includes brainstorming, outlining, matrix analysis, voting and prioritizing, strategic plan development, business process re-engineering, requirements definition, prototype evaluation, implementation plan development and system migration assessments.

References

Alan M. Christie and Mary Jo Staley, 2000l; Organizational and Social Simulation of a Software Requirements Development Process; Software Process Improvement and Practice; Softw. Process Improve. Pract. 2000; 5: 103–110 (2000)

Bada, A.O. (2002). Local adaptation to global trends: A study of an IT-based organizational change program in a Nigerian bank. The Information Society, 18(2), 77–86.

Barrett, M., S. Sahay and G. Walsham, 2001. Information technology and social transformation. The Information Society 17(1):5-20.

Baruah, N., 2000. Computerizing tax administration in India. In Information Flows, Local Improvisations and Work Practices, Proceedings of the IFIP WG9.4 Conference 2000, Cape Town, South Africa.

Benjamin, P., 2001. Community development and democratisation through information technology. In Reinventing Government in the Information Age, ed R. Heeks. London: Routledge

Braa, J., & Hedberg, C. (2002). The struggle for districtbased health information systems in South Africa. The Information Society, 18(2), 113–127.

Calitz, A.P., 2000. Transforming work practices with IT. In Information Flows, Local Improvisations and Work Practices, Proceedings of the IFIP WG9.4 Conference 2000, Cape Town, South Africa

Carmel Erran, Joey F. George, Jay F. Nunamaker 1992: Supporting Joint Application Development (JAD) with Electronic Meeting Systems: A Field Study. ICIS 1992: 223-232 [DBLP:conf/icis/CarmelGN92]

Christian V. Lundestad 2003; Pleasure and Pain: Drift and Vulnerability in Software Systems. PhD Thesis in the University of Oslo / Universiteit Maastricht; 2003

Christine Alexander and Paul J. Ambrose 1999. The Impact of Software Development Process on Information Systems Success: A Theoretical Perspective Integrating Customer Satisfaction, Technical and Organizational Issues, 1999 ACIS:723-725

D'Mello, M. (2003). Thinking local, acting global: An analysis of identity related issues in a GSO in India. In M. Korpela, R. Montealegre, & A. Poulymenakou (Eds.), Proceedings of the International Federation of Information Processing, IFIP 9.4 and 8.2 Joint Conference on Organizational Information Systems in the Context of Globalization (pp. 175–186). Dordrecht, The Netherlands: Kluwer.

Dalcher Darren and Colin Tully 2002; Learning from Failures; in the Software Process Management and Practice. 2002; 7: 71-89(DOI:10.1002/spip.156) John Wiley and Sons

Davidson Elizabeth J 1992. An Exploratory Study of Joint Application Design (JAD) in Information Systems Delivery. ICIS 1992: 271-283;

Donaldson John and John Jenkins 2000; Systems Failures: An Approach to understanding what can go wrong. Published in Proceedings of the European Software Day of EuroMicro'00 Maastricht, NL, September 2000

Ehikhamenor Fabian A (2003). Information Technology in Nigerian Banks: the Limits of Expectations. Information Technology for Development 10(2003) 13- 24. IOS Press

Flowers, S. 1996. Software Failure! Management Failure: Amazing Stories and Cautionary Tales. Reading, MA: Addison Wesley.

Glass, R. L 1998. Software Runaways: Lessons Learned from Massive Software Project Failures: Englewood Cliffs, NJ: Prentice-Hall

Heeks Rechard 2002 "Failure, Success and Improvisation of Information Systems Projects in Developing Countries". The Development Informatics working paper series; Published by: Institute for Development Policy and Management, University of Manchester, Precinct Centre, Manchester, http://www.man.ac.uk/idpm/idpm_dp.htm#devinf_wp

Hoffer Jeffrey A., Joey F. George and Joseph S. Valacich; Modern Systems Analysis and Design; 3r Ed. Pearson Education

Huy V. Vo, Bongsug Chae and David L. Olson 2006; "Integrating Systems Thinking into IS Education"; Notes and Insights in the Systems Research and Behavioural Science; Syst. Res.23,107/121 (2006) Published online in Wiley InterScience (www.interscience.wiley.com) DOI:10.1002/sres.720 IEEE; Spectrum, January 2006

Kiamba Crispus., 2003; The Experience of The Privately Sponsored Studentship and Other Income Generating Activities at The University of Nairobi A Case Study Prepared For A Regional Training Conference On Improving Tertiary Education In Sub-Saharan Africa: Things That Work!; Accra, September 23-25, 2003

Kitiyadisai, K., 2000. The implementation of IT in reengineering the Thai Revenue Department. In Information Flows, Local Improvisations and Work Practices, Proceedings of the IFIP WG9.4 Conference 2000, Cape Town, South Africa.

Lientz Bannet P. and Kathyrn R. Rea 1999; Breakthrough Technology Project Management. Academic Press

Linda Sau-ling Lai 2000; "An Integration of Systems Science Methods and Object-oriented Analysis for Determining Organizational Information Requirements" ;A research paper published by Systems Research and Behavioral Science, Syst. Res. 17, 205– 228 (2000)

Lyytine K., R. Hirschheim 1987; Information Systems Failures: a Survey and Classification of Empirical Literature; Oxford Surveys.

Lyytinen K. 1998; The Expectation Failure Concept and System Analysis View of Information System Failures: Results of an Exploratory Study; Information and Management 14 (1998) 45 -55.

Macome, E. (2003). On the implementation of an information system in the Mozambican context: The EDM case. In M. Korpela, R. Montealegre, & A. Poulymenakou (Eds.), Proceedings of the International Federation of Information Processing, IFIP 9.4 and 8.2 Joint Conference on Organizational Information Systems in the Context of Globalization (pp. 169–184). Dordrecht, The Netherlands: Kluwer.

Madon, S.,&Sahay, S. (2002). An information based model of NGO-mediation for the empowerment of slum dwellers in India. The Information Society, 18(1), 13– 19

May, Lorin J. 2000 Crosstalk Associate Editor "Major Causes of Software Projects Failures". © 1998-2000 by John Suzki, last Revised:21st April 2000

McConnell Steve, 1998; Software Project: Survival Guide. Microsoft Press.

Paul Beynon-Davies 1999; Human Error and Information Systems Failure: The case of the London Ambulance Service Computer-Aided Despatch System Project; ELSEVIER; Interacting with Computers 11(1999) 699-720

Paul. Beynon-Davies 1995; Information Systems 'Failure': The case of the London Ambulance Service Computer-Aided Despatch System Project; European Journal of Information Systems 4(1995) 171 – 184

Pressman, R., 1994; Software Engineering : A Practitioners Approach, European Edition, McGraw Hill, 1994

Puri, S.K., K.P.S. Chauhan and M. Admedullah, 2000. Prospects of biological diversity information management. In Information Flows, Local Improvisations and Work Practices, Proceedings of the IFIP WG9.4 Conference 2000, Cape Town, South Africa.

Sahay, S., &Walsham, G. (1995). Information technology in developing countries: A need for theory building. Information Technology for Development, 6(3/4), 111– 124.

Sauer C 1993; Why Information Systems Fail: A Case Study Approach; Alfred Waller, Henley-On-Thames, 1993

Standish Group International, Inc., "CHAOS:A Recipe for Success" http://www.standishgroup.com/sample_research/ Tettey, W.J., 2000. Computerization, institutional maturation and qualitative change. Information Technology for Development 9(2):59-76.

The Economist, 2000. No gain without pain, Government and the Internet Survey, 24

Thompson, M.P.A. (2002). Cultivating meaning: Interpretive fine-tuning of a South African healthinformation system. Information and Organization, 12(3), 183–211 June:7-10.