Improving diffusion of Information Technology in communities in a developing world context

Ronell Alberts[1] and Vreda Pieterse[2]

[1]Meraka Institute, Council for Scientific and Industrial Research, Pretoria, ZA,
ralberts@csir.co.za

[2]University of Pretoria, Department of Computer Science, Pretoria, ZA,
vpieterse@cs.up.ac.za

Abstract: Diffusion of information technology in a developing world context is difficult due to the fact that most of the targeted communities are in market neglect environments. Market neglect environments are characterised by a failure of the market to make an impact. In these environments the client base is marginalised, small or with low economic power. Consequently, the prospect for immediate return on investment or profit in the short or medium term is low in these environments. Software development for market neglect areas faces a number of unique challenges while still needing to produce products of high quality, on budget and on time. Traditional software methodologies have been applied in these areas with limited success, but due to the unique challenges within these areas, it has become apparent that a new or adapted software methodology is needed to ensure the effective diffusion of technology in market neglect communities. In this paper, we aim to pin down the unique problems experienced when developing for market neglect areas and to identify tools and methods required in a software development methodology to address these problems in order to improve the diffusion of information technology in market neglect areas.

1. Introduction

Software development methodologies share the common goals to accurately define requirements, structure the development effort, and deliver systems of high quality within a specific time frame and budget [1, 14, 5, 4]. Market neglect environments (MNEs) are areas where industry is not active due to the uncertain prospect of return on investment such as in developing world countries. Developing software and diffusion information technology interventions in MNEs pose unique challenges that differ from those encountered when developing and deploying software in a corporate setting. The quality of products developed in a MNE is dependent on the processes for knowledge acquisition and internal processes for the assimilation and application of technology and knowledge [22]. This implies that the application of an effective software development methodology, when developing software for the application in a MNE, is an essential factor to ensure that the

information technology interventions developed have a positive impact on the community in the MNE, maintain the highest level of quality and are effectively diffused in the community.

Existing software development methodologies have been applied in software development projects for MNEs with limited success. However, a number of essential unique needs of MNEs are not being addressed. As a consequence, design, development and diffusion of information technology in communities of MNEs in the developing world context often fail. Therefore, there is a gap in the literature describing a software development methodology that effectively addresses the unique needs encountered in software development projects for MNEs. The definition of methodologies is done by specifying discrete methodology elements that are instantiated during the use of the methodology on specific projects as needed [11].

Methodologies such as the Crystal family [8] and Multiview [4] have been described to allow a development team to adapt the methodology for each project, based on the problem area and the team composition. Various authors have proposed the combination of techniques from different methodologies, for example Avison and Taylor [3] state that many situations will "require parts of several methodologies to be used", and McCormick [19] pleads for a Reform Party between the extreme beliefs held by Extreme Programming (XP) supporters on the one hand and those who endorse Capability Maturity Model (CMM) assessment on the other hand. Similarly, we aim to combine appropriate components of other methodologies in order to compile a software development methodology for the effective application in MNEs.

The objectives of our research are: to pin down the unique problems experienced when developing for MNEs and to identify tools and methods needed in a software development methodology that addresses the problems in order to improve the diffusion of information technology in MNEs. This work is of particular interest to organisations developing software for MNEs looking for a more effective software methodology to follow as well as institutions interested in adapting existing software methodologies to include support for software development for MNEs and diffusion of information technology interventions in MNE communities.

The structure of the paper is as follows: Market neglect environments are briefly discussed in Section 2 and the theoretical foundation and research design is described in Section 3. This is followed by a description of the case study and the findings in Section 4. The requirements for a software development methodology for application in MNEs, extracted from the findings, are described in Section 5. The paper is concluded in Section 6.

2. Market Neglect Environments

A market neglect environment (MNE) is defined as an environment where the market has failed to make an impact because the prospect for return on investment

or profit is low in the short or medium term. The phrase *market neglect* originates from the concept of *market failure* and includes activities in a *pre-competitive* space. Market failure occurs when, without intervention, a market does not allocate resources efficiently [9]. Martin and Scott [18] argue that in the case of market failure, government funded intervention is needed to stimulate innovative activities to enable new companies and Small and Medium Enterprises (SMEs) to bring socially useful products to the market. Pre-competitive activities refer to research and development (R&D) which is distanced from the market and cover a spectrum from applied research to near-market development [22]. Pre-competitive R&D projects generate real economic effects in terms of new products, processes, innovative capabilities and knowledge [13].

Market neglect areas may occur when the client base is marginalised, small or with low economic power. This is particularly applicable to developing countries where the average economic power of potential users is low or where only a small section of the population will benefit, such as women, youth, disability groups and other marginalised communities. Effective diffusion of information technology is needed in MNEs for the upliftment of marginalised communities, empowerment through technology, and addressing problems related to the digital divide.

3. Theoretical Foundation

This research was conducted within an interpretive qualitative research tradition. Our ontological perspective is relativist and subjective. We therefore presuppose that there are multiple realities [6] and that what is taken as reality is an output of human cognitive process [15]. We also assume a subjectivist epistemology as identified by Lincoln and Guba [17]. In essence, it means that researchers and participants co-create understanding. This model provides a theoretic basis and analytic lens with which we investigated and established an understanding of the experiences of teams who are involved in software development for MNEs.

The research method used was that of a case study [23]. The method of interpretive case study was used to enable the exploration of the contemporary adaptation of a software engineering methodology application due to the change of the target market from industry to a MNE. The principles suggested by Klein and Myers [16] were followed. The principle of contextualization was particularly appropriate since the aim was to reflect on the historic background of the development team and its application of traditional software methodologies as well as the reaction of the team to the emerging need for a new software development methodology for a MNE.

Data was obtained through semi-structured interviews conducted with a range of role players. The interviewees included project managers, business analysts, a systems architect, software developers and researchers working on projects aimed at MNEs. Their responses were recorded through note taking during the interviews. The responses were analysed and coded using qualitative content analysis to solicit latent content as described by Graneheim and Lundman [12]. Concepts were

extracted through comparison analysis by distinguishing between similarities and differences of stated experiences [21]. 105 distinct concepts were identified. The identified concepts were grouped together into six higher, more abstract categories [10] describing the limitations experienced in applying the traditional software development methodology in a MNE. The defined categories, i.e. the limitations experienced, are described in the findings in Section 4.

4. Case Study and Findings

The case study is based on software development teams in the Meraka Institute, part of the Council for Scientific and Industrial Research (CSIR) in South Africa. The Meraka Institute was established in 2005. One of the major objectives of the Meraka Institute is to facilitate research, development and implementation of information and communication technology (ICT) applications to address problems related to the digital divide in developing countries. The institute focuses on the pre-competitive space in areas of market neglect. The software development teams in the institute were originally part of a division of the CSIR involved in software development for government and industry. The software development teams are well established and have extensive experience in developing software products using established software development methodologies based on the principles of the Unified Process (UP) [14].

Through projects conducted in MNEs, it became apparent that although the use case driven, architecture centric, iterative and incremental characteristics of the UP still proved effective, unique challenges encountered in MNEs were not supported by the traditional software development methodology. The limitations of the current methodology experienced by the team are described in the following subsections.

4.1 Needs Assessment and Requirements Elicitation

The biggest flaw in the current methodology application is the methods and processes used for needs assessment and requirements elicitation. This limitation was mentioned by all interviewees in the case study. Needs assessment is ineffective and too technology focused for an area where users cannot articulate their needs effectively in technical terms. One of the analysts commented:

> Currently we may want to 'force fit' their needs into something that is familiar for us. We need a structured process where the user is involved, feels empowered and is able to participate.

The requirements are mostly defined by analysts, based on the knowledge gained from the domain and needs assessment. Opportunity for user centric design or interaction design is low due to the inaccessibility of representation of actual users and this has a negative effect on the confidence in the accuracy of the defined re-

quirements. Due to the fact that there is no concrete client that has the final say on the requirements, there is a lack of trust in the requirements produced by the analysts. This leads to conflict between analysts and development teams. As one developer commented:

> We don't always trust the requirements from the analysts because we also have our own ideas.

Developers will challenge the requirements, or simply implement it the way they think it should be. This adds to the pressure on the analysis team to constantly reassess and defend the requirements after it has been defined.

4.2 Design and Implementation

Due to the unknown domain, frequent changes and tight deadlines, design is done informally and is mostly communicated verbally between team members. Limited or no technical documentation is produced and the resulting technical design is not visible. This leads to situations where it only becomes apparent that the design does not support the requirements when it has already been implemented, leading to conflict in the team and unnecessary rework. In addition, the lack of technical documentation makes it difficult for industry partners to come on-board during or after development and makes outsourcing of maintenance of completed products virtually impossible. As one manager remarked:

> There is always a scuffle when a prospective industry partner is identified to develop the technical documentation after the fact.

The current development methodology is experienced by the development team as being too rigid and does not allow for creativeness in the implementation phase. Research into emerging technology enables the technical team to identify opportunities for the application these technologies to the benefit of the market. Due to the requirement driven characteristic of the current methodology, these opportunities are not addressed and the creativeness of the technical team is not supported. A developer remarked:

> The most innovative ideas are ignored because it does not fit in the current plan or deliverable.

Requirements change frequently due to continuous research input, feedback from communities and the changing nature of the MNE. The continuous changes imply that the system must be regularly changed. This has a negative impact on the morale of the development team. In the words of a developer:

> Please just confirm that we did achieve what we set out to do before you change the goal posts again!

4.3 Evaluation and Deployment

Due to the low confidence in the requirements, it is not clear if the developed software addresses the real needs of the community and will make an impact in the market. There is no formal method for the evaluation of the effectiveness of the products in the community. User evaluation of completed products is difficult due to the inaccessibility of representative members of the user community.

Deployment of products to the community in the MNEs is ineffective. The community does not understand the benefits of the technology or products and therefore does not use it. Ineffective communication, involvement and feedback from the ultimate beneficiaries remain big obstacles. An analyst commented:

> It is very difficult to get user buy-in and participation after deployment.

4.4 Research Activities

Due to the unstructured environment and unknown domain, research is needed to ensure impact in the communities. The activities for development in MNEs are a fusion of structured software development and research. The emphasis of the current methodology is on the software development activities alone while the team finds it difficult to include the research activities in the process. The unstructured nature of research activities makes it difficult to manage. This leads to two separate processes: one for software development and one for research. It is important to have synergy between research and development since sometimes research is needed to enrich the development activities, while other times development is needed to evaluate or enable research activities. The current methodology does not support this need. A researcher commented:

> The timelines of research does not always correlate to that of a development project. It often takes longer to do research than to build something that is defined.

4.5 Team Composition

Due to the unique and sometimes unknown nature of MNEs, skills from disciplines other than those traditionally used in software development projects are needed. This includes disciplines such as social researchers, psychologists, domain experts, and other non-technical role players active in the domain. The coordination and integration of efforts over such a diverse group is a challenge that is not usually experienced in software projects. A technical resource remarked:

> The social researchers and domain experts are working in such a fuzzy, non definable space. I really struggle sometimes to understand where they are coming from and what their findings have to do with the real world with real deliverables.

On the other hand, a social researcher commented:

> The technical team does not always understand the deeper needs of the community, and I struggle to articulate it in such a way that they do.

4.6 Management and Sustainability

Developing software for a MNE requires activities on various levels, including research, development, partner management, funding and political issues. Effective management of these activities is not addressed in the current methodology. This is verbalised by one manager noting:

> You have this idea in your head and what emerges is totally different. You plan everything carefully, and then it becomes a bit chaotic. However, chaos is not necessarily a bad thing because that is where creativity lives. It is just, in the end you need to produce something with proven quality which makes management important.

The current methodology does not support effective management of role players in the market, collaboration, follow-ups and client relationships. Since the funding bodies are not the ultimate beneficiaries, expectation management and appreciation of benefits are experienced as challenging. As a project manager indicated:

> The person paying the money is not the person benefiting from the system. So, the guy with the cheque book wants to see results, but the results can only be reported by the community who did not request the intervention and may be hesitant to use it.

All products and services developed for MNEs must be sustainable to ensure an extended lifetime after the initial development and deployment. Currently, sustainability issues such as the inclusion of technology partners and SMEs in the design and development of products, securing of funding, business plans etc., are not effectively addressed.

All of the above limitations highlight requirements of a methodology for application in MNEs. These requirements are discussed in the next section.

5. Identified Requirements

The limitations experienced in applying a traditional software development methodology as described in Section 4 lead to the definition of a number of requirements for a software development methodology to make it suitable for application in a MNE. The defined requirements and the effectiveness of some existing popular methodologies in addressing these requirements are discussed in more detail in the following subsections.

5.1 Needs Assessment and Requirements Definition

The requirement for effective needs assessment and requirements definition is de-
fined from the limitations described in Section 4.1. This is due to the nature of
MNEs where there is usually no articulated preconceived need and the user base is
not clearly defined, geographically distributed and possibly technologically unso-
phisticated. Needs assessment techniques must not be technology focused and can
benefit by including techniques from social research disciplines. The techniques
must be effective to capture the diverse needs within a MNE user base and ulti-
mately ensure a positive impact on the lives of the community in the market. In
the absence of a client that can affirm or negate the validity of the defined re-
quirements, it is crucial that the requirements definition techniques are sufficiently
rigorous and transparent to ensure trust in the validity of the requirements.
Needs assessment and requirements definition techniques of most methodologies
are not sophisticated enough since they assume an organisational context for needs
assessment. In a MNE this is not necessarily the case where you most often deal
with a community rather than an organisation. In addition, most methodologies
rely on client confirmation of the defined requirements which is not always possi-
ble in MNEs. Per contra, the Soft Systems Methodology (SSM) [7] has a holistic
approach to needs assessment which is very appropriate for MNEs. Unfortunately,
SSM does not address requirements definition. Multiview [2] includes the needs
assessment techniques from SSM and also involves computer specialists and the
users in the requirements definition process which may render Multiview as more
applicable for MNEs.

5.2 Technical Design

The limitations described in Section 4.2 highlight the need for appropriate techni-
cal design and sufficient technical documentation for communication within the
development team and ultimate hand-over to other technical teams for further de-
velopment and maintenance. This is a universal requirement. However, some agile
methodologies, such as eXtreme Programming (XP) [5], neglect its importance.
ETHICS [1, 20] also fails since it only supports design in terms of data flow and
human-computer interaction design which we deem inadequate for MNEs. Fortu-
nately many methodologies such as UP, Rapid Application Development (RAD)
[1] and Multiview include formal documented technical design.

5.3 Introduction of New Technology

The requirement for mechanisms to allow the development team to introduce new
technology in the form of features in existing products, where emerging technol-
ogy will benefit the community, is defined from the limitations described in Sec-

tion 4.2. It appears to be the case that most software development methodologies fall short in supporting this requirement.

5.4 Include Research Activities and Methods

The limitations described in Section 4.4 indicate that the inclusion of research activities and methods is required. Development of information technology interventions for MNEs usually implies research. Coordination between development and research activities is problematic due to the differences in focus and pace. Research results must be fed into software development activities while research needs, identified during development, should inform the research. Both social and technical research must be supported. Due to the combination of research and development activities, success cannot be measured on successful implementation of information technology in communities alone. Success should also be measured in terms of impact on the community, as well as contribution to the research body of knowledge. Most existing methodologies concentrate only on the software development process and do not include phases for research as required for MNEs. Multiview seems to be the most applicable methodology since it is based on the action research process and includes investigations into socio-technical aspects through methods such as ethnographic approaches and analysis of political aspects.

5.5 Coordination of Multi-disciplinary Teams

The requirement for coordination of multi-disciplinary teams is defined from the limitations described in Section 4.5. For the successful diffusion of technology in MNEs, it is important to employ experts from different diverse disciplines such as social research, psychology, domain experts, human-computer interaction, development, architecture, and analysis. The methodology should provide processes and methods for experts from different disciplines to communicate and coordinate their activities toward a common goal. Existing popular methodologies address the integration of disciplines usually associated with software development, but do not specifically allow for inclusion of disciplines that are not normally associated with software development. The analysis phase in Multiview and SSM is aimed at different aspects such as social, political and technical views. This leads us to believe that multi-disciplinary teams are better supported by these methodologies than some of the other methodologies.

5.6 Involvement of a Loosely Defined User Base

The limitations described in Sections 4.1 and 4.3 indicate that techniques to effectively involve a loosely defined user base are required. In MNEs, the user base is not clearly defined and may be technologically unsophisticated. However, it remains essential to effectively reach and include members of the user base in the software development activities and to empower them to contribute. Users should ideally be involved in activities such as requirement definition, interface design, usability evaluation, and quality assurance. The identification of the correct users from an undefined user base, definition of user evaluation test cases, effective user feedback mechanisms and utilisation of community champions must be addressed. In addition, techniques for the effective communication of the benefits to the community are essential to ensure user involvement throughout the process, effective diffusion and ultimate buy-in and adoption of the resulting systems and products. Most popular software development methodologies fully acknowledge the need for user involvement. However, the difficulty experienced in MNEs (where the users are not clearly defined, are technologically unsophisticated, and may be geographically distributed) is not explicitly addressed by any of the methodologies we are aware of.

5.7 Support Clients who are not the Beneficiaries

The requirement to support clients who are no the ultimate beneficiaries are defined from the limitations described in Sections 4.3 and 4.6. In MNEs, clients or funding bodies provide funding as part of their social responsibility for the benefit of a community or group other than themselves. The client, therefore, is not the ultimate beneficiary. The methodology should define mechanisms to quantify results, such as proven impact in the market, new knowledge generation and research outputs, to be presented to funding parties. In addition, it must also define techniques to ensure that the beneficiaries buy into the project and value the resulting products even if they did not pay for it. None of the methodologies we investigated seem to support this requirement.

5.8 Sustainability

The limitation described in Section 4.6 indicates that mechanisms to ensure sustainability are required. After the successful completion and deployment of information technology interventions in a MNE, the funding body and the initial development agency may withdraw from the process. The developed products must therefore be self-sustained through partnerships with SMEs and champions in the community, or ongoing funding streams must be secured to ensure the continuous life of the intervention. The methodology should address issues of sustainability

such as business models, partnerships, funding, hand-over to SMEs and industry. None of the methodologies we investigated seem to support this requirement.

6. Conclusion

Software development for MNEs has similar challenges to traditional software development projects, such as the need for accurate requirements; applicable architecture decisions; iterative and incremental development; effective evaluation; and delivery of quality products on time. However, from the case study it seems that the nature of a MNE also creates unique challenges that are not addressed by traditional software development methodologies. These challenges include understanding unknown domains; effective needs assessment and requirements elicitation techniques; management of multi-disciplinary teams; and the coordination and support of software development and research activities. The UP has been applied in projects for MNEs, but with limited success. There are needs that are not addressed which cause difficulties in the projects. These include a need for a structured process that: (1) allows for technical creativity, multi-disciplinary activities and support for research activities; (2) work for a user base that is not defined, not accessible and may be technologically unsophisticated; (3) embraces cultural, social and language differences; and (4) enables development teams to develop quality products that address the real needs of a community.

It is apparent from the study that existing methodologies support some of the requirements for application in MNEs, but not all. Multiview seems to be the closest fit, but it still does not fully address all the requirements defined, and it will be necessary to adapt it to address the requirements such as (1) the need for controlled introduction of new technology; (2) involvement of a loosely defined user base; (3) support of clients who are not the beneficiaries; and (4) sustainability in the face of the possible withdrawal of the funding body and/or the initial development agency. It is proposed to adapt existing methodologies and combine techniques from different methodologies to support the needs of a specific application area such as MNEs. In addition, methods and techniques from other disciplines, such as human-computer interaction, interaction design, social research, psychology, action research and design research, should also be included in the methodology since it is essential that the development process for MNEs include both development and research activities. The application of such an adapted methodology will improve the diffusion of information technology interventions in MNEs.

Further research will entail the definition, implementation and evaluation of an adapted software development methodology that supports the defined requirements for application in MNEs.

References

1. D. Avison and G. Fitzgerald. Information Systems Development: Methodologies, Techniques and Tools. McGraw-Hill Education, 3rd edition, 2003.

2. D. Avison, A. T. Wood-Harper, A. T. Vidgen, and J. R. G. Wood. A further exploration into information systems development: The evolution of multiview2. Information Technology & People, 11(2):124–139, 1998.

3. D. E. Avison and V. Taylor. Information systems development methodologies: A classification according to problem situation. Journal of Information Technology, 12(1):73–81, 1997.

4. D. E. Avison and A. T. Wood-Harper. Multiview - an exploration in information systems development. Australian Computer Journal, 18(4), 1986.

5. K. Beck. Extreme programming explained. Addison-Wesley, 2001.

6. D. Y. Borochowitz. Teaching a qualitative research seminar on sensitive issues an autoethnography. Qualitative Social Work, 4(3):347–362, 2005.

7. P. Checkland and J. Scholes. Soft Systems Methodology in action. John Wiley & Sons, 1990.

8. A. Cockburn. Crystal Clear: A Human Powered Methodology for Small Teams. Addison-Wesley, 2004.

9. Economist. Market failure. http://www.economist.com, Last accessed: 2007/04/30.

10. B. Glaser and A. Strauss. Discovery of Grounded Theory. Strategies for Qualitative Research. Sociology Press, 1967.

11. C. Gonzalez-Perez and B. Henderson-Sellers. Templates and resources in software development methodologies. Journal of Object Technology, 4(4):173–190, 2005.

12. U. H. Graneheim and B. Lundman. Qualitative content analysis in nursing research: concepts, procedures and measures to achieve trustworthiness. Nurse Education Today, 24(2):105–112, 2004.

13. R. Grimaldi and N. von Tunzelmann. Assessing collaborative, pre-competitive R&D projects: the case of the UK link scheme. R&D Management, 32(2):165–173, 2002.

14. I. Jacobson, G. Booch, and J. Rumbaugh. The unified software development process. Addison-Wesley, 1999.

15. P. Johnson and J. Duberley. Understanding management research. London: Sage, 2000.

16. H. K. Klein and M. D. Myers. A set of principles for conducting and evaluating interpretive field studies in information systems. MIS Quarterly, pages 67–93, 1999.

17. Y. Lincoln and E. Guba. Paradigmatic controversies, contradictions, and emerging confluences In Handbook of Qualitative Research, 2nd edn. Sage, 2000.

18. S. Martin and J. T. Scott. The nature of innovation market failure and the design of public support for private innovation. Research Policy, 29(4-5):437–447, 2000.

19. M. McCormick. Technical opinion: Programming extremism. Commun. ACM, 44(6):109–119, 2001.

20. E. Mumford. Designing Human Systems: The ETHICS Method. Manchester Business School, 1983.

21. P. L. Munhall. Nursing research: A qualitative perspective. Jones and Bartlett Publishers, 2001.

22. P. Quintas and K. Guy. Collaborative, pre-competitive R&D and the firm. Research Policy, 24(3):325–348, 1995.

23. G. Walsham. Interpretive case studies in is research: Nature and method. European Journal of Information Systems, 4(2):74–81, 1995.