International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:1, No:12, 2007

Lessons from Applying XP Methodology to Business Requirements Engineering in Developing Countries Context

Olugbara O.O., and Adebiyi A.A.

Abstract**—**Most standard software development methodologies are often not applied to software projects in many developing countries of the world. The approach generally practice is close to what eXtreme Programming (XP) is likely promoting, just keep coding and testing as the requirement evolves. XP is an agile software process development methodology that has inherent capability for improving efficiency of Business Software Development (BSD). XP can facilitate Business-to-Development (B2D) relationship due to its customer-oriented advocate. From practitioner point of view, we applied XP to BSD and result shows that customer involvement has positive impact on productivity, but can as well frustrate the success of the project. In an effort to promote software engineering practice in developing countries of Africa, we present the experiment performed, lessons learned, problems encountered and solution adopted in applying XP methodology to BSD.

Keywords**—**Requirements engineering, Requirements elicitation, Extreme programming, Mobile Work force

I. Introduction

MOST standard software development methodologies are often not applied to software projects in many developing countries of the world. The primary reason is the inherent difficulty in conformance to such standards that probably do not address in details the management of requirements unpredictability. Software requirements can change at will and many standards only enforce rules that tend of overlook the important aspect of requirements unpredictability. Studies have shown that deficient requirement is the main cause of software development failure and incorrect understanding of requirements is one of the most common causes of errors in computer systems [1]. The problems that result from inept, inadequate or inefficient requirements engineering are expensive and plague most software systems and software development organizations. How to generate, represent and aggregate the right requirements, therefore, is the most important and difficult part of software development [2]. Requirements elicitation is an important tool for requirements generation and documentation. Software requirements elicitation is a process by which all parties involved in a software project discover, review and understand user needs and the limitations of the development activity of the software [3]. But, there is always the problem of managing people as well as the project itself for large software development.

There are two main categories of requirements, namely business and invented. Specifically, Business Requirements Engineering (BRE) is the requirements engineering for contracted applications, while Invented Requirements Engineering (IRE) [4] is the requirements engineering for off-the-shelf applications. The focus of this paper is on BRE for the following important reasons. (a) Requirements for business applications exhibit high degree of unpredictability, (b) BRE directly involves many stakeholders with diverse opinions and (c) the user or customer of a business application is the primary source of requirements acquisition.

The objective of this work is to investigate whether BRE can generally be effective through the adoption of XP methodology. The investigation becomes essential due to the claim in literature that XP addresses the problem of unpredictable requirements. The rest of the paper is fleetingly sum-up as follows. Section 2 overviews the XP methodology. Section 3 describes our experience of XP practice at Kreative Resources Consultant (KRC). Section 4 details the approach followed to address the challenges. Section 5 concludes the work by summarizing the experience gained.

II. The Extreme Programming Methodology

The XP methodology [5] is an agile software development process that has reputation for building applications with unpredictable requirements. XP delivers excellent promises like strong customer orientation, support for elicitation and ease of learning, overhead reduction and productivity enhancement [6]. The aim is to incrementally deliver a useful minimal system as early as possible. Thus, as a methodology, it builds upon the incremental software development model and object-oriented techniques. XP can guarantee quality software production through code refactoring and it is suitable for refinement of unpredictable requirements. But, XP comes with inherent deficiencies, for example, it is limited in terms of team size and its inability to scale [7]. It is best used with small to medium sized teams of no more than 12 people

Manuscript received October 25, 2007

O.O. Olugbara is a Senior Lecturer in Computer and Information Sciences Department, Covenant University, Ota, Nigeria (corresponding author phone: +27783461919; e-mail: oluolugbara@gmail.com).

A.A. Adebiyi is a lecturer in Computer and Information Sciences Department, Covenant University, Ota., Nigeria (e-mail: ariyo_adebiyi@yahoo.com).

highly skilled and motivated individuals [8]. XP cannot handle large system development projects where a lot of time is spent building something that often does not deliver value to the user [9].

Paulk reviewed XP from the perspective of Capability Maturity Model (CMM), a five-level model that prescribed process improvement priority for software organizations [10]. It was concluded that lightweight methodologies, such as XP advocate many good engineering practices and XP fits more concretely into levels 2 and 3 of CMM. Nawrocki et al., from a software engineering point of view, proposed the following three modifications to XP. (a) Written documentation of requirements managed by tester/analyst, (b) modified planning game to accommodate multiple customer representatives and (c) the requirements engineering phase at the beginning of the project that provide a wider system perspective [11]. They evaluated both XP and their Modified XP (MXP) using Summerville-Sawyer maturity model [12]. XP scored low and was classified as initial, but, MXP considerably improved on XP and was classified repeatable. Generally, XP Maturity Model (XPMM) [13] is very suitable for assessing XP maturity. This model is a lighter four-level maturity model specifically proposed for discriminating between real XP and pseudo XP projects, which have a common characteristic of not having written documentation.

The Nawrocki et al. model completely transferred the responsibility of an analyst, who is in charge of requirements documentation and management to a tester, who assists the customer to choose and write functional tests and running them on regular basis [11]. The extent to which requirements should be documented was addressed by Document Capability Maturity Model (DCMM) [14]. The documentation can be conducted by both the development and business teams.

III. The XP Methodology Experiment

KRC is a software consultancy and training firm resident in Nigeria, where the first author works as a part-time principal consultant. The firm has members with experiences ranging from less than one year to over eighteen years. We conducted a case study at KRC to examine the effectiveness of XP on a real-life software project. Initially, we constituted a group of nine full-time and highly experienced staff of four programmers, two testers/analysts, two document writers and one project manager, who acted as a senior instructor. The team was given organic software to develop, two computers were provided and two subgroups were formed, each with two programmers, one computer, one tester and one technical writer, all working on aspects of the same project. The testers and writers acted as customer representatives, although they were KRC staff. The two subgroups recorded tremendous success and the qualitative feedback was very encouraging. The project duration was short, cost was low, most errors discovered were corrected very quickly by pair programming and code refactoring and this gave members full confidence.

In the second stage of the experiment, the same team was asked to work on a completely new semidetached software project contracted to KRC by a government ministry. The size of the project team was increased by addition of four customer representatives to provide support. The testers and writers were asked to constantly liaise with the customer representatives, while the programmers settled for the real coding business. The initial result was impressive, but after two weeks, KRC was faced with a terrible situation. The customer representatives stopped showing up, hence, KRC representatives (mainly testers and writers) decided to visit the customers, but they had a pungent experience. For instance, there was a demonstration of a nonchalant attitude and regular grouchiness by the customer representatives. Most times, those that supposed to provide support were either coming late to office or not coming at all.

The result of the experiment indicates that (a) productivity can be very low in XP projects due to the inability of a developer to rapidly learn new requirements in the business domain and (b) the expressive power of the customer to effectively communicate requirement needs is a serious problem that can be encountered.

IV. The Solution Methodology

The experience of customer problems was the basis for the adoption of MXP methodology. Several complicated issues were fairly managed as follows. First, incentives were approved for the customer representatives by the management of KRC for them to show cooperation. KRC employed some of the representatives on a part-time basis to achieve maximum productivity, but more cost was incurred than anticipated. There is the need therefore, for an adaptive cost model that can account for the extra effort/time expended in XP project.

Next, the planning game session was used to train customer representatives on the practice of XP and requirements engineering principles. The training did not take more than one week, but very intensive. The customer representatives were actively engaged in the development process, for example, they prepared their requirements document, wrote their stories and test cases and we recorded very successive results. Thus, XP is highly effective when customers have adequate knowledge of what to program and the experience to drive the process on their own.

A. Tasks Partitioning

The concept of Tasks Partitioning (TP) was introduced to improve the quality of MXP practice in KRC. TG is defined as the process of grouping tasks or subtasks resulting from business stories into partitions. Each partition is then assigned to a group of members, which for instance, consists of two programmers, two testers/analysts and a leader. The number of groups depends on the available resources and the philosophy is maximum capacity utilization. A tasks partition was assigned to a group depending on the complexity of the tasks in the partition and the ingenuity of the group members. The partition containing most difficult tasks was assigned to a group with more experienced members, while less experience group was assigned to less complex tasks partition. As the development progresses group members were moved around to work on different tasks. Thus, the process of knowledge sharing was facilitated. Figure 1 illustrates the use case

diagram for tasks assignment. From the figure, both the customer representative and the developer can be assigned or reassigned to a new group.

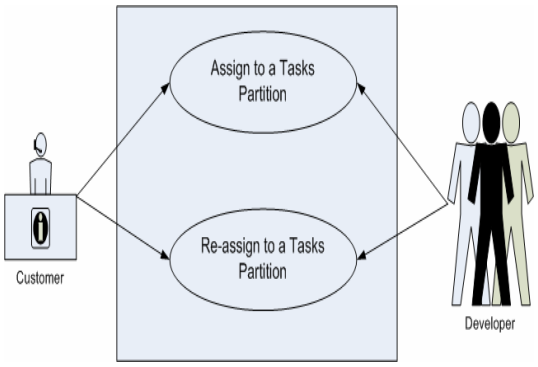

Fig. 1 Tasks Assignment

An important group in the model is a strategic group, which consists of top management representatives from the development team. The group is responsible for planning, policy formulation and strategic decision-making. The group also controls and coordinates the overall activities, including customer training, provision of basic incentives, integration testing and validation. The model is a network of information flow and a node component, in this case, represents a group. This comes with the attendant benefit of support for multiple participants through tasks grouping, support for a wider system perspective through training and minimization of customer problems through on-site availability of experienced customer representatives. In addition, the methodology allows for considerable interaction flexibility and free flow of information. It encourages much lighter documentation and requirements management, increases project velocity, eliminates start-up and time to market pressures and creates a relaxed atmosphere for workers.

B. Mobile Work force

The concept considered for realizing full customer participation and satisfaction is the "Mobile Work Force" (MWF), which is defined as a group of mobile or adaptive developers, prepared to carry development activities to the customer's organization. In several instances, we asked the customer to provide a makeshift office and all the necessary infrastructures to enable our MWF work in the customer's organization. In this particular scenario, we had on-site developers instead of on-site customers. The experience was exciting and the model perfectly worked. The exciting thing is that we saw what customers were doing in reality and helped them to achieve better results. In comparison, this method evolves adaptive developers, while the original XP evolves adaptive customers. The customers were very satisfied with our model of XP, we enjoyed their maximum cooperation and in many situations, they converted the makeshift offices to IT project offices after the completion of the project to express their satisfaction. Figure 2 illustrates the summary of Developer-Activity-Customer circle of the MXP practice. Finally, we identify as very crucial to the success of XP, the following key factors. (a) IT awareness must be well promoted in many organizations, (b) customer training and motivation usually provided by the development should be encouraged, (c) adaptive cost estimation model should be adopted and (d) embracing MWF for full customer participation is a necessity.

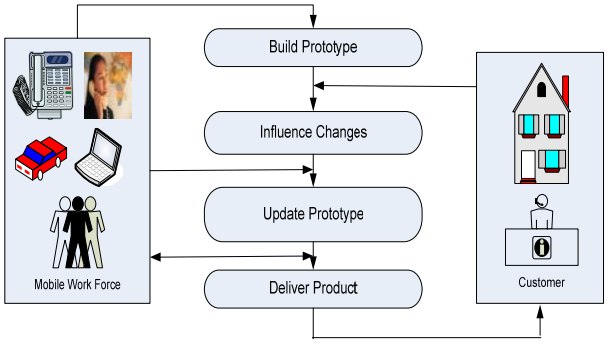

Fig. 2 Developer-Activity-Customer Cycle

V. Conclusion

Conclusively, XP is a highly organized development process, which emphasizes direct communication with customers and nominates express coding. The methodology is code-oriented, people-oriented and adaptive in nature. Direct communication is essential in order to encourage new ideas, create and reuse knowledge and to promote skill in a dynamic world of unpredictability. High quality end-product certification of business requirements is guaranteed through direct feedback, code refactoring and functional tests execution.

XP proposes effective coding and this can be achieved through improved requirements acquisition, enhanced acceptance test production and swift feedback from customers. Customer representatives must be endowed with requisite knowledge and dexterity in order to advance a case for XP. Adequate planning and customer training can accomplish this basic requirement. However, we believe generally, that XP adoption without significant investment in developing customer representatives will almost certainly not produce expected results for applications with unpredictable requirements. Development firms must ensure that for effectiveness, customer representatives on their teams receive adequate incentive, training and skill development. These inescapable obligations should be well promoted in XP project.

We are currently using XP in a number of project teams, both within and outside the university system and we are getting excellent results in terms of work contentment, cost reduction, customer satisfaction, meeting schedule and business requirements, team productivity and empowerment. In particular, the KRC team was able to start over on a difficult problem and deliver a high-quality product on time and within budget using the XP method. The use of XP complemented by MWF technique made us achieved better

results in terms of productivity, cost reduction and product quality. We are now able to manage complexity and save more time than ever before. This comes with the attendant advantage of improved B2D interactivity that is central to productivity.

REFERENCES

[1] R.G. Wolak, DISS 725-System Development: Research Paper 2, Software Requirements Engineering Best Practices, School of Computer and Information Sciences, Nova Southeastern University, 2001. http://www.itstudyguide.com/papers/rwDISS725researchpaper2.htm
[2] P. Sawyer, I. Sommerville and S. Viler, "Capturing the Benefit of Requirements Engineering", IEEE Software 16(2), 1999, pp 78-85.
[3] R. Thayer and M. Dorfman, Software Requirements Engineering, 2nd Ed. IEEE Computer Society Press, 1999.
[4] C. Potts, "Invented Requirements and Imagined Customers: Requirements Engineering for Off-the-Shelf Software", 1995, http://www.doi.ieeecomputer society.org/10.1109/ISRE.1995.5125 53
[5] K. Beck and M. Fowler, Planning Extreme Programming (2000), Addison Wesley.
[6] K. Beck, "Embracing change with extreme programming", IEEE Computer, 32, 10 1999, pp. 70-77.
[7] B. Bahli, and E.S.A. Zeid, "The role of knowledge creation in adapting extreme programming model: an empirical study", Proceeding of ICICT-IEEE conference on information and communications technology 2005, pp. 75-87.
[8] M. Aoyama, "Web-based agile software development", IEEE Software 1998, 15(6), pp. 56-65.
[9] F. Maurer and S. Martel, "Extreme programming: rapid development for web-based applications". IEEE Internet Computing 2002, pp. 86-90.
[10] M.C. Paulk, "Extreme Programming from a CMM Perspective", 2001, http://www. hristov.com/andrey/fht-stuggart/xp-cmm-paper.pdf.
[11] J. Nawrocki, M. Jasinski, B. Walter, and A. Wojciechowski, "Extreme Programming Modified: Embrace Requirements Engineering Practices" Proceedings of the IEEE Joint International Conference on Requirements Engineering, 2002.
[12] I. Sommerville and P. Sawyer, Requirements Engineering: A Good Practice Guide. John Willey and Sons, 1997.
[13] J. Nawrocki, B. Walter and A. Wojciechowski, 'Toward Maturity Model for Extreme Programming", 2001, http://www.dsv.su.se/~mira/Agile%208.pdf.
[14] S. Huang and S. Tilley, "Towards a Document Maturity Model", ACM-SIGDOC' 2003, pp 93-99

O.O. Olugbara is a senior lecturer in the department of Computer and Information Sciences, Covenant University, Nigeria. He holds PhD in Computer Science. His research interests include database compression and mining, artificial intelligent techniques, mobile computing, software engineering methodologies, object-oriented software development and mathematical modeling and simulation.

A.A. Adebiyi is a Ph.D student in Department of Computer and Information Sciences Covenant University, Nigeria. He holds B.Sc (Computer Science), M.Sc (Management Information System) and MBA. His current research interests are on data mining, requirement engineering and e-commerce technology. He is a member of the Nigerian Computer Society (NCS), and Computer Professional Registration Council of Nigeria (CPN).