Agile vs. UML Software Development Methodologies for Dynamic Market (A Comparative Study)

Mubarak R. Al Rashoud

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science of Philosophy

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY
KING ABDULAZIZ UNIVERSITY, JEDDAH – SAUDIA ARABIA

Rabei II, YEAR - April, YEAR
Agile vs. UML Software Development Methodologies for Dynamic Market (A Comparative Study)

Mubarak R. Al Rashoud

We certify that we have read this thesis and that in my opinion is fully adequate in scope and quality as a thesis for the degree of Master of Science of Philosophy.

Thesis Supervisors:

Dr. Mahmoud Kamel (Advisor)  
Signature  Date

Dr. Ibrahim Al-Bidewi (Co. Advisor)  
Signature  Date
Agile vs. UML Software Development Methodologies for Dynamic Market (A Comparative Study)

Mubarak R. Al Rashoud

This thesis has been approved and accepted in partial fulfillment of the requirements of the degree of Master of Science of Philosophy.

Examiners:

Dr. Ahmed Balamesh
(External Examiner)
Signature: ___________ Date: _______

Dr. Osama Abulnja
(Internal Examiner)
Signature: ___________ Date: _______

Dr. Ibrahim Al-Bidewi
(Advisor)
Signature: ___________ Date: _______
Chapter 1

Introduction

The Standish group reports illustrated in 2003 that 10\% of the software engineering projects had been cancelled before they ever get completed. Further results indicate 51\% of projects had been implemented with a cost overrun of 28\%, with time overrun of 82\%, and with only 25\% of the required features. On the success side, the average is 44\% for software projects that had been completed on time and on-budget. The Standish group estimated in 2003 that $55 billion was the lost dollar value for the failed and challenged US projects. (A challenged project is a delivered and operational project, but over-budget, over the time estimate, and offers fewer features and functions than originally specified). The rates of the succeeded, failed and challenged projects, from 1994 until 2003, in United State, are shown in table 1.1 [72].

<table>
<thead>
<tr>
<th>Project year</th>
<th>Succeeded</th>
<th>Failed</th>
<th>Challenged</th>
</tr>
</thead>
<tbody>
<tr>
<td>1994</td>
<td>16%</td>
<td>31%</td>
<td>53%</td>
</tr>
<tr>
<td>1996</td>
<td>27%</td>
<td>40%</td>
<td>33%</td>
</tr>
<tr>
<td>1998</td>
<td>28%</td>
<td>38%</td>
<td>34%</td>
</tr>
<tr>
<td>2000</td>
<td>28%</td>
<td>23%</td>
<td>49%</td>
</tr>
<tr>
<td>2003</td>
<td>23%</td>
<td>10%</td>
<td>51%</td>
</tr>
</tbody>
</table>

Table 1.1: Succeeded, Failed and Challenged Projects Rates [70] [72] [77]
The most important aspect of the Standish researches is discovering why projects fail. To do this, the Standish group surveyed many IT executive managers for their opinions, about why projects are challenged and why they are cancelled. The results of the survey concluded that the three major reasons that cause projects to be challenged are: lack of user inputs, incomplete requirements, and changing requirements. While the three major factors that cause projects to be cancelled are: incomplete requirements, lack of user involvement and lack of resources [7]. These reasons are extremely clear and appeared in the dynamic market environment. In this environment Time To Market (TTM) pressure increases, and at the same time, market changes fast. This causes high requirements volatility with taken into consideration to fully satisfy the customers' requirements [7].

Working under such circumstances is the most challenging task for the development team. Instable requirements can break the software architecture, especially if the used software methodology is not able to cope and response to the dynamic changes. Software projects under time criticality may collapse if the product is not delivered in the due time, or if the delivered product dose not satisfies all customer needs. One of the major project management decisions, to control these problems, is the "selection of the project’s software development methodology that helps coping with the challenges, and prevents many potential dynamic project problems" [7]. Many software development methodologies are found in the software world. Table 2.1 shows the most well-known development methodologies.

<table>
<thead>
<tr>
<th>Planned methodologies</th>
<th>Evolutionary models</th>
<th>Agile methodologies</th>
<th>Ad hoc</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waterfall (serial development)</td>
<td>Spiral model (risk driven iteration)</td>
<td>Feature-driven development (FDD)</td>
<td>No discipline (chaotic ‘hacking’)</td>
</tr>
<tr>
<td>Incremental development models</td>
<td>Rational unified process (RUP)</td>
<td>Adaptive software development (ASD)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Extreme programming (XP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scrum</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Dynamic Systems Development Method (DSDM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Crystal</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2.1: The Most Famous Development Methodologies [7], [6]**

A development methodology that is suitable for one project may not be appropriate for another. This depends on the project conditions and the development
environment. In [10], a process model selection frame has been proposed, which the project manager can use as a systematic guide for choosing the project’s process model. The authors have illustrated how the different software development methodologies react to the main software development failure factors.

This thesis presents in chapter 2, a comparative study between two specific development approaches. It compares between the planned methodologies, which have a coupling relation with UML analysis and design techniques, and agile methodologies. This study presents the advantages and disadvantages of using the two approaches in the dynamic market environment. As it had been stated in [4], the agile methodologies differ from planned methodologies in that agile methodologies are more adaptive to the external market factors than planned methodologies, which assume precise prediction of the project behavior. Another difference is that agile methodologies are more people oriented, whilst planned methodologies are process oriented. Authors in [7] and [10] have shown that planned methodologies employ a lot of heavy and complex design techniques using UML. All of design activities are totally separated from the implementation activity. This makes planned methodologies too heavy to keep up with the pace of dynamic software development projects. Developing in the planned approaches is like a black box to the customer. This means validation is done when the customer starts working with the product during acceptance phases. The results of that is the increasing of the cost of changes of requirements during development process.

Agile methodologies control this problem by simplifying the design process and almost couple it with the coding effort. They allow delivering the product in an incremental manner. The customer is involved in the development effort. This gives agile methodologies the ability to change quickly according to the external factors without any obvious cost. Therefore "when the project under development is extremely changeable, because of the markets factor, and continuously changes its requirements, then agile methodologies are suitable and better to be used over planned methodologies" [10].

As it is stated in [10], in addition to their high response to the dynamic factors, agile methodologies increase the product quality and team productivity. This fact has been gained by comparing the results of developing two versions of Sabre Airline Solutions. First version had been implemented using a traditional method, while the
other version had been implemented using XP (one of the agile methods). As in [6], XP methodology makes awareness development and maintenance less effortful on a software development team. The author gained this knowledge, by studying two development teams within the same organization, the first team utilized the Extreme Programming (XP) methodology, and the other used the traditional methodologies. In spite of this, some studies indicate that agile and traditional approaches provide roughly the same results in terms of product quality, as it is stated in [6].

The proposed agile practices in software development vary, but they share common characteristics such as iterative development, working in frequent consultation with the customer, and developing the project in a set of frequent releases. eXtreme Programming (XP) is one of the widely used agile methodologies. XP is a software development approach that advocates rapid iterations, rigorously tested code, working closely with end users and applies the simplicity and lightness in its planning and design efforts. XP conducts release planning by performing what is called planning game. "The customer presents the desired requirements to the developer with the knowledge of the importance of the requirements. Then the developer estimates the requirements."

This approach of XP planning may be efficient when there is only one customer, and when there are no constraints that control the prioritization of requirements. "But the situation is different when there are competing stakeholder interests" [8]. In this case many point of views and opinions are raised, which lead to conflicts among different stakeholders. On the other hand, technical constraints have clear effects in the planning process. For example, it is not possible to start implementing a requirement unless the prerequisite requirements are already finished, even though they are less valuable to the customer. These different variables present at the beginning of each release when release plan is produced. Therefore, a technique has to be found to optimize the selection of the requirements that would be implemented in each release.

The release planning problem is addressed and illustrated by many authors. In [41], a cost-value approach for prioritizing requirements has been developed. The Analytic Hierarchy Process (AHP) has been used to compare the customer alternatives in a stepwise fashion, and measure their contribution to the customer objectives. In [8], a
A hybrid approach called e-release planning has been proposed, which combines the strengths of computational intelligence with human intelligence supported. In [\textsuperscript{21}], a method called EVOLVE has been presented. This method is aimed at the continuous planning of incremental software development based on genetic algorithms. It has been stated that, in the release planning process, three main considerations should be taken into account: the technical precedence inherent in the requirements, the typically conflicting priorities as determined by the representative stakeholders, as well as the balance between required and available effort. In [\textsuperscript{21}], an evaluation of methods for prioritizing software requirements has been illustrated. Six different methods for prioritizing software requirements have been evaluated (AHP, hierarchy AHP, Minimal spanning tree, Bubble sort, Binary search tree and Priority groups). The evaluation was based on the quality requirements for a telephony system. All six methods have been used individually on separate occasions to prioritize the requirements. The analytic hierarchy process has been found to be the most promising method, although it may be problematic to scale-up. In [\textsuperscript{21}], an approach for improving existing methods for release planning has been presented. This has been achieved by handling the uncertainty of data using fuzzy logic. The fuzziness with respect to the effort estimates, effort capacity constraints and the different objectives related to cost, benefit and quality, has been considered. The satisfaction of traditional constraints on effort has been performed using a fuzzy system, to obtain an overall satisfaction level of a solution.

XP process is discussed generally in chapter \textsuperscript{3} of this thesis, with more concentration on planning activity. A framework for XP is provided. This framework is composed of a set of algorithms that represent the different XP processes. The release planning problem is discussed, and illustrated formally.

This thesis proposes a fuzzy decision maker (FDM) based model to solve the release planning problem. The main function of the proposed FDM is to optimize the decision of which requirements (user stories in XP) should be delivered in each release. The proposed FDM takes into consideration many factors to achieve the optimal planning process; i) stakeholders individual importance, ii) the values and the priorities of different stories, for the different stakeholders, iii) precedence and coupling constrains between the different user stories, iv) risk that could be faced to deliver each story, and v) maximum effort for each release.
An engineering configuration management (ECM) system has been taken as a case study, to clear out all discussed concepts. The ECM has been implemented using XP, taking into consideration the prioritization for user stories using the proposed FDM. This case study is discussed in details in chapter 4 of this thesis. The conclusions and the further studies are shown in chapter 5.
المستخلص

من أكبر التحديات التي تواجه تطوير البرامج، هو التطور تحت تأثير الأسواس الديناميكية التي تتميز بالسرعة والتصاق في طرح المنتج إلى الأسواق مما يؤدي إلى عدم استقرار وعدم ثبوت المتطلبات. هذه العوامل قد تؤدي إلى فشل عملية التطوير، وخصوصا إذا لم يتم اختيار طريقة التطور المناسبة لعوامل السوق الديناميكية. لذا فإنه من أهم القرارات التي تتخذها مدير المشروع هو اختيار طريقة التطوير المناسبة. المطورون الذين يستخدمون الطرق السريعة يدعون أن هذه الطرق هي الطرق المعجزة التي تستطيع التعامل مع السوق الديناميكية. ومن جهة أخرى فإن الفريق من المبرمجين يدعى أن الطرق التقليدية، المعتمدة على التخطيط طويل الأجل، وعلى لغة النمذجة الموحدة (UML) - غير قادرة على التعامل مع سرعات الأسواق الديناميكية. هذه الأطروحة تقدم مقترحات بين هاتين الطرقتين من عدة زوايا: التحليل، التصميم، كلفة التغيير وكيفية اعتبار الموارد البشرية. الدراسة توضح كيف أن الطرق السريعة لها سرعة في الاستجابة لمتغيرات الأسواق. وكما توضح الدراسة، رد فعل كل من الطرقتين، لعوامل الفشل الموجودة في الأسواق الديناميكية. بالإضافة إلى ذلك فإن هذه الأطروحة تسلط الضوء على سرعة استجابة الطرق السريعة عن طريق مناقشة المعالجات المختلفة التي عن طريق يتم التطور بطريقة البرمجية الفائقة السرعة (XP) التي تعد من أشهر طرق البرمجية السريعة حيث يتم تقديم إطار عمل متكامل لهذه الطرقية. توجد بعض نقاط ضعف متعلقة بطريقة البرمجية الفائقة السرعة وذلك من ناحية إمكانية استخدامها على مشايع متعددة الزبائن ومن ناحية عدم تأسيب عملية التخطيط، لذلك يتم رفض فكرة التخطيط في البرمجية الفائقة السرعة عوضاً، ويتم تقديم نموذج يعتمد بالدرجة الأولى على مصادر البيانات باستخدام المنطق البياني، وذلك بهدف تحقيق التخطيط الوظيفي وخصوصا عندما تكون هناك ألوان مختلفة لعدد من الزبائن. هذا النموذج أثبت نتائج جيدة عندما تم استخدامه في تطوير نظام إدارة التشريفات، حيث وجد أن الزبائن حصلوا على أهم المتطلبات من النسخة الأولى للبرنامج وتم إضافة وتعديل بعض المتطلبات لاحقا من دون أي كلفة.

تذكر...
ABSTRACT

In the software development, the most challenging task is to develop projects under the pressure of the dynamic market - where (Time To Market (TTM) and requirements instability) - could fail the development process. Therefore, the project management should choose the development methodology that can control the problems associated with the dynamic market. The enthusiastic programmers in agile methodologies argue that planned methodologies are heavy to cope with the rapid changes of the dynamic market, because they strongly emphasize on the planning process by incorporating a lot of detailed design techniques like UML. The enthusiastic programmers in agile methods claim that agile is a miracle approach that has solutions for all problems related to the dynamic market. They also claim that agile achieves higher flexibility and better to satisfy actual customer requirements by developing and delivering the software product in an incremental fashion, as well as, it avoids any development overheads. This thesis presents a comparative study to compare between planned methodologies _ which have strong coupling relationship with UML analysis and design techniques _ and agile methodologies. The comparison contains many issues such as analysis, design, human resource, cost of the changes of the requirements and communication. The comparison also shows how the lightness of the agile methodologies gives better responses to the different problems related to the dynamic market. Also the study illustrates that the agile minimizes the cost of the changes of the requirements during the development process. Furthermore, the thesis focuses on the lightness of the agile methodologies planning by discussing eXtreme Programming (XP) process. First, a framework of the traditional XP is presented. This framework presents a sequence of processes that build the XP project. A set of algorithms are presented to describe the different XP processes. There are many limitations of the traditional XP release planning, such as its limited scale and its informality. In this thesis, the release planning problem is presented formally. The most factors that have clear affects in the release planning are considered, which are the priority of the requirements, their values, their risks, the effort needed to implement them and the precedence and coupling among them. If there are many stakeholders, the priorities and values of the requirements should be taken from the perspective of each stakeholder. To solve the release planning problem, a Fuzzy decision maker (FDM) is designed and
implemented. The main function of the proposed FDM is to optimize the decision of which requirements (user stories in XP) should be delivered in each release. The FDM takes into consideration the opinions and point-views of the different stakeholders.

The suggested FDM is tested in the generation of the XP release planning for the Engineering Configuration Management (ECM) system which is taken as a case study. It is found that customers have had the most important features of the system in the early releases. Many requirements have been added in the middle of the project, others have been removed, without any additional cost.
TABLE OF CONTENTS

ABSTRACT i
ACKNOWLEDGEMENT ii
TABLE OF CONTENTS iii
LIST OF TABLES V
LIST OF SYMBOVS Vi

I Introduction
\subsection{Planned Methodologies vs. Agile Methodologies}
\subsection{Introduction} 9
\subsection{Agile and Planned Methodologies Requirements Analysis} 10
\subsection{Nature of Requirements} 16
\subsection{Planned Methodologies and Requirements Analysis} 19
\subsection{Agile Methodologies and Requirements Analysis} 20
\subsection{Designs in Agile and Planned Methodologies} 21
\subsection{Planned Methodologies and Design} 26
\subsection{Agile Methodologies and Design} 27
\subsection{Agile / Planned Methodologies and Human Resources} 29
\subsection{People Orientation or Process Orientation} 29
\subsection{Communications Among the Members of the Development Team} 30
\subsection{Development under dynamic market} 37

\section{Optimizing eXtreme Programming Release Planning}
\subsection{Introduction} 36
\subsection{XP Common Values} 37
\subsection{XP Practices} 38
\subsection{XP Process Framework} 39
\subsection{XP Planning} 40
\subsection{Release Planning} 41
\subsection{Release Planning: Exploration Phase} 42
\subsection{Release Planning: Planning phase} 45
\subsection{Iteration Planning} 50
\subsection{Tasks Brainstorming} 51
\subsection{Tasks Accepting} 52
\subsection{XP Construction Process} 54
\subsection{Limitations of Traditional XP Process} 55

\section{Formulation Problem of XP Release Planning}
\subsection{Release Planning Variables} 56
\subsection{Problem Statement for Software Release Planning} 58
\subsection{XP Release Planning Using Fuzzy theory} 59
\subsection{Basic Concepts of Fuzzy System} 60
\subsection{Fuzzy Logic and Fuzzy Sets} 61
\subsection{Fuzzy numbers} 61
\subsection{IF –THEN Rules} 61