Risk analysis for software Reengineering process transformation with XP

Joshphin JasalineAnithauthor.X
Assistant professor, MCA Department
NMSS. Vellaichamy Nadar College
Madurai,TamilNadu,India.

M.Karthika
Assistant professor, MCA Department
NMSS. Vellaichamy Nadar College
Madurai,TamilNadu,India.

Dr.K.Alagarsamy
Associate Professor, Computer Centre,
Madurai Kamaraj University, Madurai
TamilNadu, India.

Abstract

The people are passion with new trend and technology. Any software can survive only when it meets the customer need with current trend and technology. Otherwise the software has to undergo many modifications or it should switch to new one. The reengineering approach is a gift to the software engineers, to make the project with reduced cost and reduced risk. This paper deals how the software engineering process moves to reengineering with the many different types and its need. In reengineering process also, there are many risks can arrive as software engineering. The light weight process Extreme programming methodology is suggested for the software reengineering process transformation. This paper deals with the expected risks of process transformation and a quantitative method for risk assessment. This assessment helps to minimize the effort during transformation, indirectly cost benefit and increases the customer satisfaction.

1.Introduction

Legacy transformation is about maintaining and extending the value of this legacy investment through migration to new platforms. Re-implementing applications on new platforms can have benefits through reduced operational costs, and through the additional capabilities of new technologies it provides access to valuable functions through more economical means. Migration to a new platform also provides an opportunity to align applications with current and future business needs through the addition of business functionality and through application restructuring [3]. In this paper how the reengineering phases are combined with XP is explained. A light weight process XP is an iterative methodology well suited for medium and small level projects. The reengineering process with light weight process will be easy to maintain, cost benefited, with its simple rules. Any software project might expect risks in their development phases. Same as the Reengineering process transformation with XP might face risks. The risk management is a good practice to manage the risks. There are four steps of risk management are, to identify them, categorize them, mitigate them, and manage them. This paper deals with the risk management of the reengineering process transformation with XP methodology.

2. Reengineering

2.1 Definition: Software reengineering also known as both renovation and reclamation is the examination and alteration of a new form and the subsequent implementation of the new form.[1]

The need of reengineering is to transform an existing system into a new system to support the environment with new trend and technology.

The existing software is called legacy system. The new system can be arrived from the legacy in which, many modern techniques are used or adding subsystem, adding extra resources to face the new trend and technology with user satisfaction.

2.2 Objective of reengineering: While applying new technologies generating a new target system that maintains the required functionality from the existing systems it takes and in stills a good software development method and properties. There are four general reengineering objectives.

Preparation for functional enhancements

- Improve maintainability
- Migration
- Improve reliability

The phases of reengineering process are

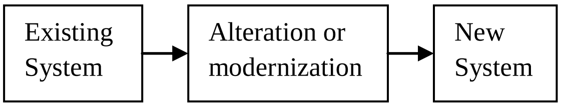

Fig.1 Reengineering phases

Reengineering projects are not always a simple process. Sometimes it is very complicated and to satisfy the customer satisfaction. Most software development projects will not be completed with complete customer satisfaction. Developer will say that the customer is not explained the project very clearly. To overcome this complaint the software industries hit upon a methodology known as Extreme programming (XP).

3. EXTREME PROGRAMMING

Extreme programming is a prominent and popular agile methodology. It is very success in many companies and industries of different sizes in world wide. It is based around the development and delivery of small increments of functionality, customer involvement in the process, constant code improvement and egoless programming.[2] .Extreme programming mainly focuses the customer satisfaction. Any software project using XP improves in five essential ways. They are communication, simplicity, feedback, respect and courage.

Extreme programming empowers the developers to confidently respond to changing customer requirements, even tardy in the life cycle[3].Extreme programming empowers the developers to confidently respond to changing customer requirements, even tardy in the life cycle. Xp emphasizes teamwork. The team consists of managers, customers, and developers all are equal partners in a collaborative team. XP implements a simple, yet effective environment enabling teams to become highly productive. The team self-organizes around the problem to solve it as efficiently as possible. The most prominent and surprising aspect of Xp is its simple rules. The rules may seem to be embarrassed but are based on sound values and principles. The goal is to realize the rules to define an environment that promotes team collaboration and empowerment. There are three agile levels of planning.

i. Release planning ii. Iteration iii. Daily plan or stand up meeting.

An agile process makes an honest plans based on feedback. A responsive project heart beat creates a i. *new* plan in each ii. *iteration* based on feedback from the previous iteration and changing customer needs, not what was left unfinished within an ii iteration make each day a heartbeat all its own. Start each day with a iii. *Standup meeting.*

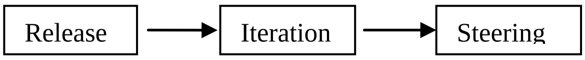

Fig.2 XP Phases

Integrate code at least once a day if not every couple of hours. The most essential ingredient that changes repetitive development into iterative development is feedback. A shorter iteration gives the team more frequent feedback and more opportunities to learn and steer the project. XP recommends iteration duration as one week.

The shorter iterations make the customers, managers, developers to feel optimistic about progress. It will lead to a steadiness throughout the project and it will avoid the rushing of finished product at the end. A measurable, predictable, sustainable pace helps the team to plan and meet them commitments. With these steady easy to predict pace will make plans that are more likely to come true. At the end of the iteration conduct a demonstration for every delivery of working software.

Every meeting will start with a planning meeting. All the iterations, the team should feel like fresh start, a blank canvas full of new opportunities for an empowered team. An agile project has all activities going on all the time as needed. No documents with signature to guard the progress are maintained. Instead it relay on automated tests and working software to our customer often and accept changes to ensure our efforts continue in the right direction. Because working software is an agile theme that affects everything do.

The agile process focused on to set a project heartbeat, to produce working software in all iteration. It is possible only with honest plan. The entire team is focused on working incremental versions of a system. When the team produces working software in small increments are no longer locked into a minimum cost to get return on investment. The team can stop when ever they think that they have done enough or a higher priority projects comes along. Working software ready for production release is easy when they have automated unit tests, automated acceptance/QA tests, and a one button build that runs them in a reasonable time. The customers don’t always know the best way how to solve their problem, help them by demonstrating working software frequently.

4. Reengineering process transformation phases with XP

Reengineering process transformation using Xp phases are depicted in fig.2.

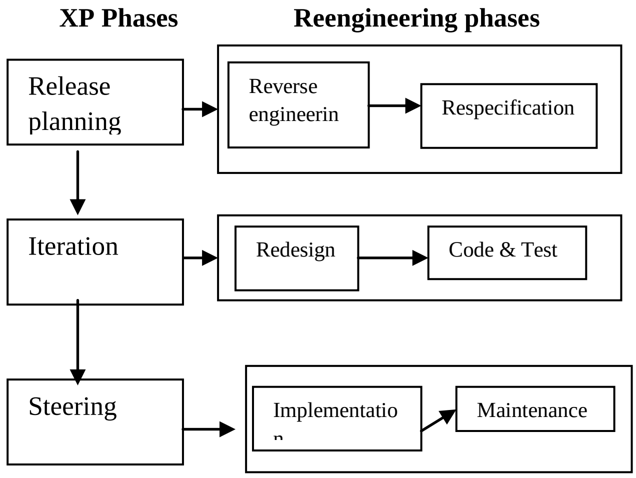

Fig.2 Reengineering process transformation phases with XP

4.1 Release planning

The most essential task of extreme programming is release planning. The two kinds of release planning method are predictive and adaptive [6]. Predictive planning means that developers make a detailed plan covering the whole software life cycle. But in the adaptive planning method developers only make the detailed plan for short time and keep a rough long-time plan. The relative reengineering phase, it has reverse engineering and respecification phases. From these two phases the requirements and design specifications are prepared by minimum number of iterations with customer feedback.

4.2 Iteration

This phase of XP is a subset of he release plan stories that will be done in next iteration. Only one iteration plan exists at a time. With the chosen collection of important features the team can estimate the effort to implement it. The developers have the authority to set the estimates. The manager will set total amount of work that the next iteration can have planned. In reengineering phase the works design, code and test the design with no of iterations. Each iteration, it will release working software.

4.3 Steering

The small release might undergo a demonstration in this phase for customer satisfaction and get acceptance for next iteration plan. In reengineering it is implementation and maintenance of the newly developed software.

5. Risk Analysis

5.1 Risk Definition: A risk is a potential problem, a situation that, if it materializes, may adversely affect the project. Risks that are materialized are no longer risks, they are problems.

All projects have risks, and all risks are ultimately handled.1) some disappear 2) some develop into problems that demand attention, and 3) a few escalate into crises that destroy projects. The goal of risk management is to ensure that risks never fall into the third category. There are four steps to identify them, categorize them, mitigate them, and manage them.

5.2 Identification and Analysis: Risk analysis begins with the identification of risks. Then the probability and loss of each risk are estimated. At last, all kinds of risks are combined to show the whole risks for process transformation. Probability and loss of a risk can be estimated quantitatively or qualitatively. Quantitative estimation requires a lot of time and cost, and sometimes it is difficult for developers to collect enough data for quantitative analysis [2]. Thus qualitative risk analysis is more suitable for reengineering process transformation using XP. Risks may be cause losses in multiple aspects of a software reengineering process. The requirement analysis is the primary step for any software reengineering process. If the requirement is very clean and clear the remaining part of the process will become risk less. Thus, the requirements in reengineering process transformation risks are taken in to a serious consideration and risk taxonomy is prepared.

Table.1 Requirements Risk taxonomy

Risk type	Risk item
Goal oriented	Undefined
	Vision mismatch
Document	Not Available
	Partial /incomplete
People	Less customer involvement
	Legacy system developers not available
information	Unstable, vague story
	Defective
Impact	Unrealistic
	Mission mismatch

The degrees of “Low”, “Medium”, and “high” are used as the description for probability and loss of risks and then the Risk Exposure (RE) is defined according to the Table.2. Reengineering process transformation using Xp may have multiple risks. Risk exposure is used to accumulate according to the type of loss. The taxonomy uses a method to combine multiple risks by scores.

Table. 2 Score for Risk Exposure

Risk Exposure	Score
Deplorable	5
Major	4
Critical	3
Important	2
Minor	1

Developers may use the score table to accumulate RE and this will help to mitigate the risks. The scores give a straight way for comparing multiple procedures. It will help the people to make decisions.

Table.3 Probability of risk

Probability	Description

5	91-100% or very likely to occur
4	61- 90% or likely to occur
3	41-60% or may occur about half of the time
2	11-40% unlikely to occur
1	0-10% or very unlikely to occur

Table.4 Risk Impact

Probability	Risk Impact				
	Minor	Important	Critical	Major	Deplorable
0-10%	1	1	1	2	2
11-40%	1	1	2	2	3
41-60%	1	2	2	2	3
61-90%	2	2	2	2	3
91-100%	2	3	3	3	3

The Table.3 describes the probability and loss with risk exposure.

6. Risk Matrix

Multiply the two risk vectors probability and impact together to obtain the simple product which is the risk value. Using the threshold value colour the matrix.

7. Mitigation plan

7.1 Risk rating plan

The important risks that threaten the success of the project are identified. Next step is to prepare a risk mitigation plan. According to the risk rating a mitigation action will be taken. So, prioritize the risk according to the threshold value and comprehends it as risk plan. Risk rating is done through the customer story and the defects. These are written in the cards directly to help the developers for this a highlighter and a color code is used. Here some suggestive wordings are depicted in the risk plan. If the risks changes or no of risks increased, then the wording also will change.

Table.5 Risk Rating

25	1. urgent action required 2.immediate notification to senior people	**Critical**
14-20	1. weekly review requires 2. notification to seniors	**Major**
6-13	1. requires monthly review 2. must monitor on monthly basis 3. notification to senior people	**Moderate**
1-5	1. No impact on any aspect 2. requires quarterly review 3. No explicit action required	**Minor**

8. Risk management

Risk management is to track and manage risks on a project. For this a risk register is suggested. It helps the team to minimize and mitigate the risks.

Risk Register

A risk register can be useful to track and manage the risks on a project. For this a spreadsheet can be used for notification. In our reengineering process transformation with XP the iterations must undergo a risk assessment for each iteration, add a new page to the spread sheet and its corresponding risk assessment. In this way, the risks can track and how a risk has changed over the course of a project. It helps to monitor how risks are added or deleted from register. As the iterations near the end of completion stage, all the risks gradually should move into the minimal range shaded in green color. Otherwise, the project is not in a correct path. The team should take care of the risks. All of the time, effort and money invested up to those points is at risk of being lost.

Conclusion

The risks in reengineering process transformation with light weight process XP can be minimized with the risk management and make the process more cost benefited. This paper deals with the risk management concepts and the risks are analyzed with the qualitative method. A risk exposure, and a probability of risk, risk rating, risk register are discussed.

REFERENCES

1. Chikofsky and cross-ii, “Reverse engineering and design recovery”, IEEE, 1990.
2. B.Boehm, tutorial: software risk management ,IEEE computer society press,1989.
3. Beck,k., Fowler,M., Planning Extreme programming, Addision –wsley, 1st edition, 2000.
4. Mingshu li, meng huang, fengdi shu., A risk driven method for extreme programming Release planning, ICSE’ O6, shanghai, china.
5. **Chikofsky.E.J; Cross.J.H.,” Reverse Engineering and Design Recovery : A Taxonomy”, IEEE Software, Vol.23, 1990,pp.13-17.**
6. Larman.c Agile iterative development A management’s Guide, pearson Education, 2004.
7. K.Beck ,Extreme programming Explained: Embrace Change, Addison Wesley, reading, mass., 1999.
8. Ian Somerville, Reengineering, “Software engineering Eighth edition.
9. Legacy Transformation , Declan Good ,August 2002 COPYRIGHT CLUB DE INVESTIGACIÓN TECNOLÓGICA .
10. Byrne, Eric J., Software reverse engineering: A case study”, software-practice and experience, 12/91.
11. Sneed, Harry M, “planning the reengineering of legacy systems”. IEEE 95.
12. what are the basic steps in risk management , www.adeak.com
13. Risk Analysis of global software development and proposed solutions, Liguo yu, Alok Mishra, AUTOMATIKA 2010.
14. Design paradigm and risk assessment of hybrid reengineering with an approach for development of reengineering metrics, IJSE&A, vol.3, 2012.