FDRD: Feature Driven Reuse Development Process Model

Sonia Thakur[1], Harshpreet Singh[2],
[1,2]Department of Computer science and Engineering,Lovely Professional University,Jalandhar, India
[1]soniathakur.2401@ymail.com,[2]harshpreet.17478@lpu.co.in

***Abstract*— As fast the business requirements changes, the need of rapid of development and economical feasible software also increases. The new software development techniques and models are coming to picture to solve the problems of rapid changing requirements. Agile methodology is one of the approaches to fulfill the current business requirements, which is flexible to adapt the change at any phase of development. Feature driven development (FDD) is an agile based process model based on feature development, adapted by many organizations. The limitation of agile process is its incapability to reuse components those are developed through agile processes. Adopting reuse is a challenging task but it can be used at an initial level by integrating with various development processes. Reuse oriented development of software is considered to be one of the most efficient techniques to improve software quality as it increases the productivity and reduces the development effort and cost. This paper purposes a reengineered Feature driven reuse development (FDRD) process model which integrate reuse concept with feature driven development process model. The model improves the productivity of organization and quality of the produced product.**

Keywords*— *Software Reuse, Feature Driven Development (FDD)

I. INTRODUCTION

Changes are required by the customers and are frequent to any software product or module which is under development, due to market competitions. The priority of requirement changes frequently and only specific development is done which is urgently required. Feature Driven Development model is one of the agile methodology approach adapted by medium sized organization to support software development and is an evolution to traditional process model approach.

Organization faces many problems in software development including change in customer requirements, change in technology, increased cost, delayed schedules, unsatisfied requirements and lack of software professional. This type of situation often referred to software crisis. Billions of code is developed, here question arises that what we are seeking to achieve? Here the concept of reuse comes into picture. Reuse is promoted as one of the effective weapon against never ending software crisis. Effective reuse of software artifacts increases productivity, reduce time to develop and to market, and reduce the cost of the development. These are the major requirements demanded by the customers and the organizations.

A. Software Reuse

Software reuse is a process of creating new software from the existing software artifacts rather than building from scratch. Software reuse catalyzes improvements in quality by incorporating components whose reliability has already been established [1]. Reuse not only consider just lines of code, in a general way reuse can be a design pattern, code, test cases, documentation, knowledge, framework etc. To achieve reuse as a business objective many obstacle block the progress of organizations to adopt reuse such as: technology engineering, process models, organization environment and business perspective. The work done here is providing solutions to all these obstacles by a systematic approach.

1) Types of Reuse

a) Opportunistic Reuse: While getting ready to begin a project, the team realizes that there are existing components that they can reuse [2].

*b) Planned Reuse:*The business start with initial concern and investment of reuse. This type of reuse is concerned with company employee efforts and company conformance to build software products with concern of reuse. It is a strategically approach so that the developed assets can be reused in future.

B. Feature Driven Development Model

In this modern competitive software environment the changes are very frequent to any software product [2]. Agile methodology provides solution to this problem. Feature driven development is one of the agile methods. FDD is a process for helping teams produce frequent, tangible working results. It is a highly iterative and collaborative agile development i.e. composed of five processes, these are described using the ETVX (Entry-Task-Verify-Exit) based.

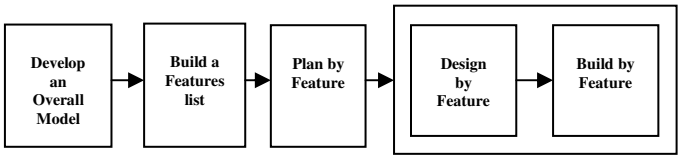

Fig. 1. Feature driven development model [3]

II. RELATED WORK

Software reuse concept exist form 1969, basic concept behind reuse is economical benefits to organization. Active areas of reuse research in the past twenty years include libraries, domain engineering methods and tools, reuse design,

ISBN No. 978-1-4799-3914-5/14/$31.00 ©2014 IEEE

design pattern, generator and business and finance [2]. Different technical approaches are defined to implement reuse. Majority of work is based on line of code, maintain component repository, classification of components and search optimization techniques [2],[4]. Many different effort and cost estimation models [5],[6] are also defined to give support to reuse. Some work is also done to implement agile methodology and reuse, but still the reuse is come into the picture in a role of lines of code, again the techniques developed to implement reuse are database repository maintenance [7], and selecting the component. This paper proposes a model which integrates reuse and FDD with a systematic approach, which results as reuse as a business. It begins from code snippet and end at early level of decision to reuse the existing reusable feature/feature set.

III. INTEGRATING REUSE AND FEATURE DRIVEN DEVELOPMENT

In every development organization reuse exists whether it is planned reuse [2] that is known as Reuse Engineering Software Business (RESB) or *active reuse,* because the organization having knowledge of reuse can conforming it in every best possible way. Another approach is *passive reuse*,[8] when a developer reuses code from his own desk drawer it is not under concern of organization or project manager. This passive reuse neither helps in growth of organization nor in developer growth. By integrating reuse with FDD, it result in high productivity and less cost which directly increase the profits of organization and improved status in market in terms of quality product and less time to market. According to proposed model, it follows bottom up approach according to FDD model i.e. Build by feature and with constraint of not to disturb the existing project development.

A. *Domain Analysis and Engineering*

The process of identifying related software system in a domain and discovering commonality and variability among the system, is known as domain analysis. Information composed helps in identify the models that are used to create artifacts for reuse. After some experience domain engineer find that the product developed are not new systems but variant of other system products. So the domain engineering is an activity to improve the quality of system through reuse of software artifacts. So, domain analysis and engineering plays a vital role in software reuse as a systematic approach.

B. *Bottom-up Approach*

To integrate reuse we applied bottom-up approach. According to that firstly work at build by feature phase, which contain coding, testing and implementation. After reusable features are defined, now move to design by feature which walkthrough the domain, design and inspect the design. By working on these two stages a reusable feature will be extracted with maximum design documentation information. Now, when reusable feature set is created and maintained, it is ready to reuse at initial level of any project overall design as deign of any project. It shows an idea to start from small investment and to be ready for big gains .By integrating reuse with FDD the work of architect and domain engineer take a new way, which is very effective and efficient. With the advent of time this bottom up approach increase the benefits of company and it results in an incremental adoption because it starts from no reuse to informal code reuse and so on.(see Fig. 2)

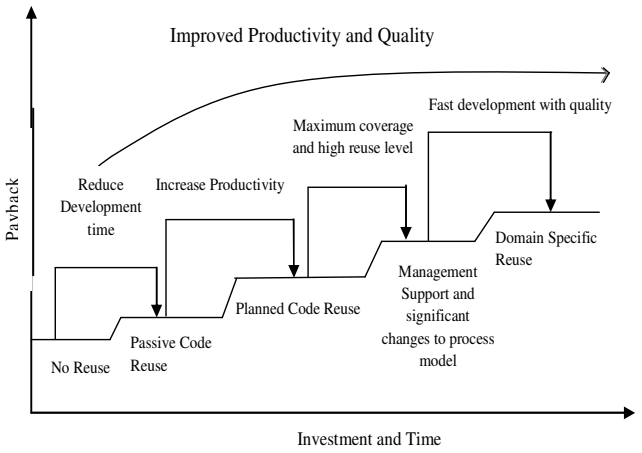

Fig. 2. Time graph of reuse

C. *Reusable Feature Set*

As we know that reuse provides a generalized solution whereas agile method provides specific solution. To integrate both the opposite behavioral concepts need reusable artifacts to attain reusability. Reusable artifacts are code and other components that can be reused from one project to another[2]. To create a reusable assets in FDD model following steps are taken:

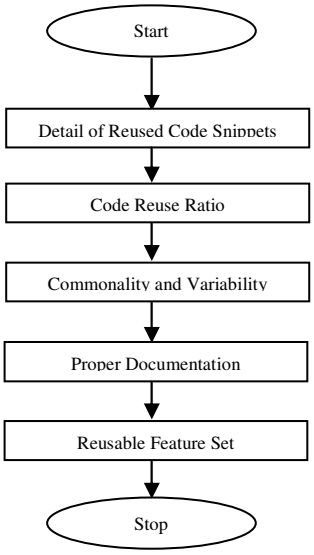

Fig. 3. Flow chart to create a reusable Feature Set

- As the code is at low level reuse artifact, but to introduce code reusability as a helpful entity to support reuse within a working organization which already follow a process model and projects work.
- Calculate the code reuse ratio by using the formula:

CRR=Number of project reuse a specific code snippets ÷ Total number of project developed

(If CRR value greater than 0.5 than that code snippet is proposed to be reusable feature)

- Now, find the commonality and variability points which helps to define that code under a specific feature and repacking it separately. Commonality is an assumption held uniformly across given set of objects(S) where as variability is an attribute with different values for any element of S. For example, commonality is login interface to access system and variability is login location of particular user.
- Now, a proper documentation is done to make it reusable to provide ease of use in future. Technical writers are the main lead role in that process. It contains architecture, uml models, data dictionary and description document etc.

D. Redefined Roles and Responsibilities

In reuse driven software engineering three key roles exist to implement reuse in organization i.e. creator, supporter and reusers. When project requirements comes reusers forward those requirements to creator, then creator find out the existing components and pass it to the supporter, then supporter certify and advise for components and forward that report to reuser. Now reuser integrates the component with newly developed components and releases the required product. In FDD six key roles involved to design an overall model, here the chief architect and domain Administrator play the role of creator it involves reuse of knowledge and experience, project Administrator and system Administrator plays the role of supporter and feature owner, class owner and supporting team member play the role of reusers, those working with development phase.

E. Reengineerd Process Model

After creating reusable feature set, now it's time to reengineer the process model which describes how and where to integrate reuse with FDD. It is like rethinking to existing aspect for better results and which enhances the productivity and improve the working environment of organization (See Fig 4). Effective reuse is not simple addition to existing software development process. The systematic application of software reuse will dramatically change the software process [11].

IV. CASE STUDY

A survey was conducted and various problems regarding the delay in delivery of software, effort estimation and cost estimation, which directly affect the productivity, financial gains and quality of product, were outlined. Aspect that makes agile methodology adopt rapidly by development companies is,

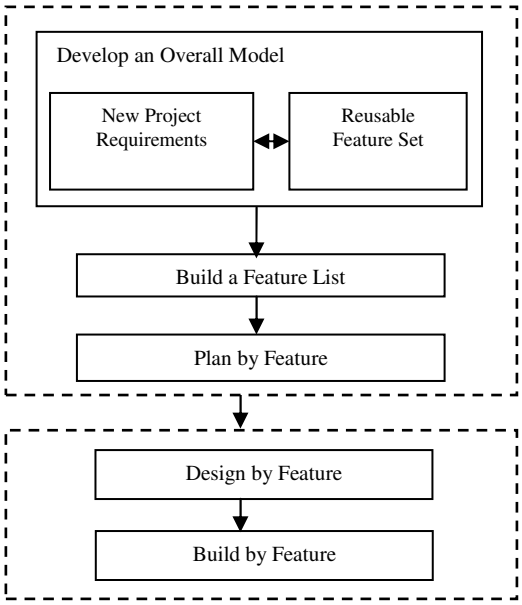

Fig. 4. Proposed Feature Driven Reuse Development Model

flexible enough to adapt change whenever required by customer at any time of software development, which sometimes result in dirty coding and low concern of variation points.

Microchip Pvt. Ltd is a medium sized organization, which provides web solution. Company deals in Web services. They works with feature driven development process model one of the agile methodology. They face problems delay in delivery, partial testing, bad coding habit, and problem in maintenance later on. Best part of company working environment is flexibility. After studying there working criteria and customer behavior, I proposed my model to test in their environment for better results. To start implement proposed model there are some prerequisites, these are, working habit of employees, experience of employees and communication session with project managers, which helps to find out the week point and facts for failure to achieve their goals. After that check the status of past projects by communicating with developers, find out the mostly used code snippets and create reusable feature which can be used in future.

The result from first project is not that much good, because of first experience to find and take decisions regarding reuse, but it was quite better than the past experience of employees.

In second project (Table I) the decision regarding reuse is taken easily and in early stage which results in reduces effort and timely deliver to market. When we know that how much work is already done and how much is left, it also encourage the team to do work fast and accurately and free employee can be assigned to any other project where deadlines are not meeting or any other issue, which directly increase the productivity of company quantitatively and qualitatively.

TABLE I. PROJECT DESCRIPTION

Entity	Value
Project Name	Information Management system
Duration	50 Days
Front-End	PHP and HTML4
Back-End	MYSQL
Description	To manage Industrial training institute

A. *Develop an Overall Model*

1) Study the Requirement documents, decision about team and rough estimation of features. Table I. gives details of project what to develop and constraints like time, fornt-end and backend required.

TABLE II. REQUIRED INFORMATION

Initial Participant to Study of Requirements	Chief Architect, Domain Administrator and Manager
Team Formation	10 Team members
Initial level Informal Features Count	10 Features approximately

TABLE II. gives detail of who are the participants and what are decisions taken by them like size of team and what kind of features projects demand.

TABLE III. TEAM DESCRIPTION

Roles	No. of Assigned Members
Project Manager	01
Chief Programmer	01
Domain Administrator	01
Database Designer	01
Supporting Developer and Designers	04
Tester	02
Total	10

TABLE III. Describes the team member involves in project.

TABLE IV. INFORMAL FEATURE LIST

Sr. No	Informal Feature List
1	Login Feature
2	Attendance Feature
3	Course Detail
4	Fee Structure and Payment Feature
5	Salary Feature
6	Online Test Feature
7	Extra Billing
8	Company Portfolio
9	Message System

TABLE IV. Represent the informal list proposed by chief architect, domain administrator and manager, this informal list helps to take decisions for reuse.

TABLE V. FEATURES LIST WITH DECISION OF EXISTING FEATURES

Sr. No	Informal Feature List	Status
1	Login Feature	Exist
2	Attendance Feature	Exist
3	Course Detail	New
4	Fee Structure and Payment Feature	New
5	Employee Management	Exist
6	Online Test Feature	New
7	Extra Billing	New
8	Company Portfolio	New
9	Message System	New

B. *Build a Feature List*

1) Assign a priority value to each feature set which helps to take decisions for development.

TABLE VI. FORMAL LIST OF FEATURE SET WITH PRIORITY RATE

Sr. No	Feature List	Priority
1	Student Management system	High
2	Employee Management System	Medium
3	Company Portfolio	High
4	Course Management System	High
5	Online Test system	Medium
6	Expense Management	Low
7	Message System	Low

2) Divide complex feature-set into features. (See Figure)

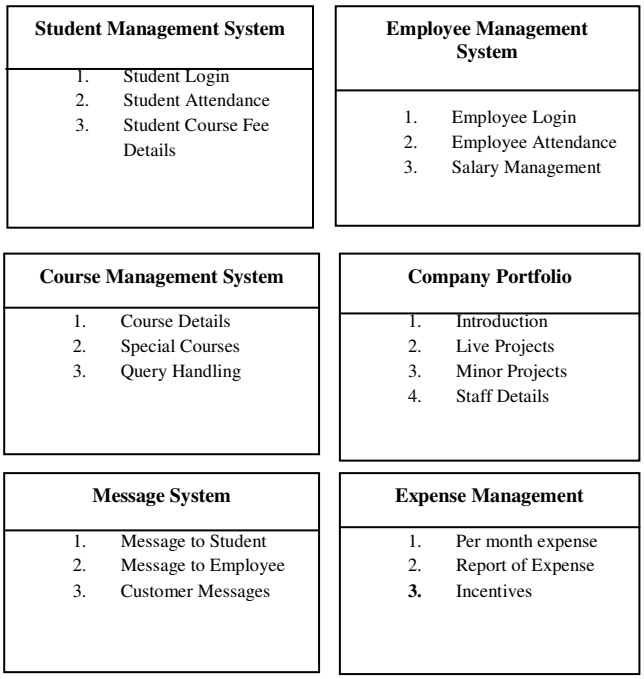

Fig. 5. Divide feature sets into sub feature

C. *Plan by Feature*

1) Feature set having high priority will be taken in first iteration (see TABLE VII.). If more feature-set having high

priority than customer give some weight-age value, which helps to calculate priority value.

TABLE VII. PRIORITY LIST

Priority Based Sequence	Feature-Set
1	Company Portfolio
2	Course Management System
3	Student Management System
4	Employee Management System
5	Online Test System
6	Message System
7	Expense Management system

2) Define the completion dates for each feature-set. (see Fig 6)

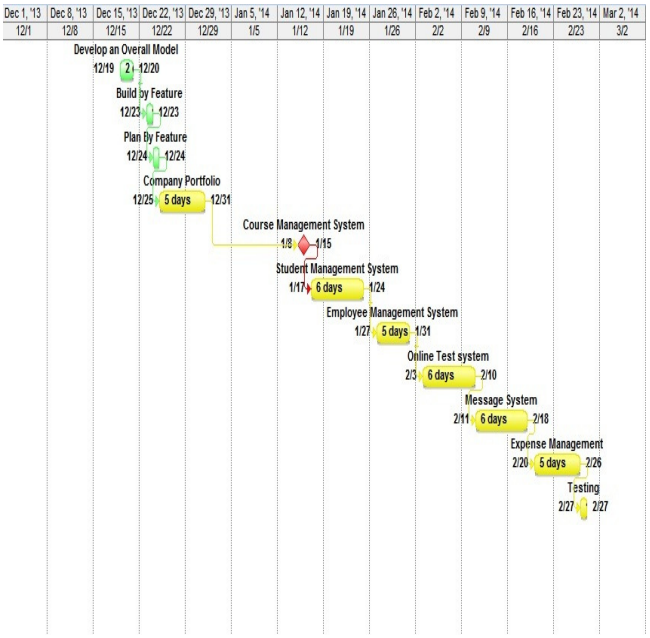

Fig. 6. Gantt chart of project

D. *Design By Feature*

Design for particular feature-set is develop in this phase. The object model designed in phase1 is overall model representing whole software system; where as design model for feature-set is problem specific according to the requirements. It is like small project development to create a large project, similar to prototype process model.

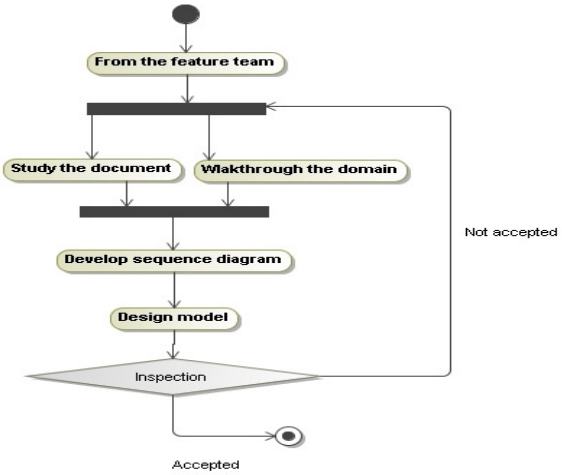

Fig. 7. Design Model for particular feature-set

E. *Build by Feature*

According to design model, implementation of classes and methods are started. New features and modification to reusable feature is done in this phase and integrated (see Fig. 7.). In testing defects in reusable feature are less than new develop feature, which increase the reliability and accuracy.

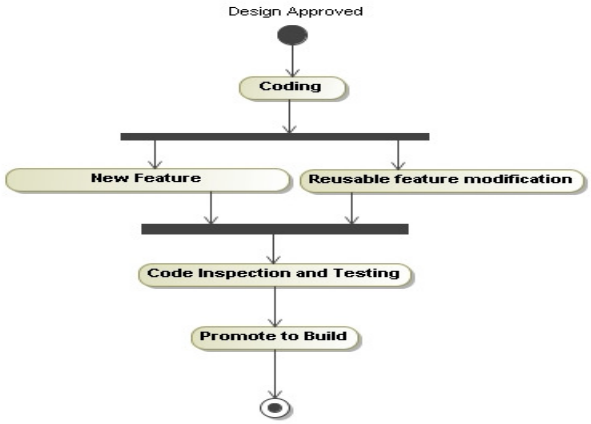

Fig. 8. Decisions and Implementation flow of designed feature-set

V. ADVANTAGES OF PROPOSED MODEL

By integrating reuse and FDD enhance not only the productivity but also the quality of product. The proposed model increases the profits and reduces the development cost for a new project with reusable feature set. It also gives new responsibility to the various team members, which enhance their knowledge and ability to work efficiently. A person is not restricted to single role, so it helps to get experience from one role to improve the predictability and the ability to handle risks in another projects. So, this proposed model aims in reusing the code snippet and it also reuses knowledge and experience of employee.

VI. CONCLUSION

The paper purposed a reengineered process model. Till now agile method and software reuse are discussed separately, but this model combines the strengths of both and the opposite behavior of reuse and feature driven development helps to give new shape to feature-set, which is used as and when required to develop a project. This model increases the productivity, quality of developed products and uses the past experience to improves the predictability of the process risks.

VII. FUTURE WORK

The future scope of this work is to evaluate the effort and cost involved in project development with various risk factors to implement and find the critical success factors of reengineered process model.

ACKNOWLEDGMENT

I would like to thanks to all those who provide me the possibility to complete this research. A special gratitude I give to my guide Assistant Professor Harshpreet Singh, whose contribution in stimulating suggestions and encouragement.

Furthermore I would like to acknowledge with much appreciation the crucial role of the staff of Microchip Pvt. Ltd, who gave me the permission to work with their resources, which really helps to complete my tasks.

REFERENCES

[1] Richard W.Selby, (2005)," Enabling Reuse-Based Software Development of Large-Scale Systems", Software Engineering, IEEE Transaction on, Vol 31, Issue 6.

[2] Neha Budhija and Satinder Pal Ahuja, (2011),"Review of Software Reusability", International Journal of Advance Research in Computer Science and Software Engineering, Vol 3,Issue 1

[3] Marek Rychly and Pavlina Ticha, (2007) ," A Tool for Supporting Feature Driven Development", International Fedration For Information Processing.

[4] B. Jalender, Dr. A Goverdhan and Dr. P Premchand, (2010), "A Programmatic Approach to Software Reuse", Journal of Theoretical and Applied Information Technology, Islamabad

[5] Mayank Mandloi, Prof.Sachin Patel and Prof. Rakesh Pandit, (2013), "Cost Estimation Model for Software Reuse", International Journal of Advanced Research in Computer Science and Software Engineering, Vol-3, Issue 6

[6] Ralph M. DeFrangesco,"A Cost Model for Software Reuse".

[7] Sukhpal Singh and Inderveer Chana, (2012), " Enabiling Reusabilty in Agile Software Development", International Journal of Computer Application, Vol-50, No.13.

[8] Ivar Jacobson, Martin Griss and Patrick Johnson Book, "Software Reuse:Architecture, Process and Organization for Business Success", 2007.

[9] G.N.K. Suresh Babu and Dr. S.K. Srivatsa,(2009),"Analysis and Measures of Software Reusability", International Journal of Reviews in Computing.

[10] Naresh Kumar Nagwani,Pradeep Singh, (2009), "An Agile Based Model for Change-Oriented Software Engineering," IEEE International Journal of Recent Trends in Engineering, vol. 1, no. 1.

[11] Nasib S. Gill and Sunil Sikka, (2011),"Inheritance Hierachy Based Reuse and Reusability Metrics in OOSD", International Jouranl on Computer Science and Engineering, Vol 3, No 6.

[12] Leilei Kong and Tao Yuan, (2009),"Extension Feature-Driven Use Case Model for Requirement Tracibility",International Conference on Computer Science and Education.

[13] Mili H., Mili F. and Mili A., (2002), "Reusing Software: issues and research directions",Software Engineering, IEEE Transaction on, Vol 21, Issue 6.

[14] Hasan Kahtan, Nordin Abu Bakar, Rosmawati Nordin, (2012),"Reviewing the Challenges of Security Features in Component Based Software Development Models", E-Learning, E-Management and E-Services (IS3e), IEEE Symposium on.

[15] Richard Mordinyi, Eva K and Alexander Schatten (2010), " Towards an Architectural Framework for Agile Software Development", IEEE International Conference and Workshops on Engineering of Computer-Based Systems.

[16] Livemore and J.A., (2007),"Factors that Impact Implementing an Agile Software Development Methodology", Southeast Conference, Proceedings, IEEE.

[17] Naresh K. Nagwni and Pardeep Singh, (2009), "An Agile Methodology Based Model for Change Oriented Software Engineering", International Journal of Recent Trends in Engineering, Vol 1, No 1 .

[18] Jianxiong Pang and Lynne Blair, (2004),"Refinement Feature Driven Development- a methodology for early aspects", Lancaster University, Bailrigg .

[19] Martin L. Griss, (1998), "Software Reuse: Architecture, Process and Organization for Business Success", Addison-Wesley-Longman 1997, ISBN 978-0-201-92476-3, pp. I-XXVIII, 1-497.

[20] Kevin D. Wentzel, (1994), "Software Reuse- Facts and Myths", International Conference of Software Engineering.

[21] Kaushal Pathak and Anju Saha, (2013)," Review of Agile Software Development Methodologies", International Journal of Advanced Research in Computer Science and software Engineering, Vol 3.

[22] William B. Frakes and Kyo Kang,"Software Reuse Research: Status and Future", IEEE Transactions on Software Engineering, vol. 31, no. 7, pp. 529-536, July 2005.

[23] Palmer, S. & Felsing, J., (2002)," A practical guide to feature-driven development", Prentice Hall. Upper Saddle Hill River, NJ.

[24] Sarbjeet Singh, Sukhwinder Singh and Gurpreet Singh,(2010), "Reusability of Software", International Journal of Computer Applications, Vol 7, No 14.

[25] Korkala, Abrahamsson and Kylloen, (2006), "A Case Study on the Impact of Customer Communiucation on Defects in Agile Software Development", Agile Conference, 11pp-88.

[26] Bukhardat J.M. and Detienne(1995), "An Empirical Study of Software Reuse by Experts in Object-Oriented Design",INTERACT'95, Norway.

[27] Seungwon Lee and Ho-Jin Choi, (2009), "Software Component Reusability Measure in Component Grid", International Conference on Advanced Communication Technology, Vol 1.