ISSN No. 0976-5697

Volume 3, No. 1, Jan-Feb 2012

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

A Framework to Introduce Component Based Methodology in AGILE Software Development

Puneet Kansal*and Sachin Gupta
(Software Engineering)
University Institute of Engineering & Technology,
KUK, Kurukshetra, India
puneet.coe@gmail.com*[1]
sachin.gupta_2006@yahoo.co.in[2]

Abstract: Software industry is discovering new software development models. Various models has been discovered in last two decades such as stage wise model, water fall model, transform model, evolutionary model, spiral model, and new one the agile software development model. The basic need of software industry is to get quality product, reducing development time, and reducing product cost and productivity improvement. No single model is capable of providing all these basic needs of the industry. So this paper combines two most popular technique of software development i.e. Component Based Development Technique with Agile Software Development Model, which depicts great results in the form of a better quality and reusability.

Keywords: Agile software development, Component based method, Reusability, Quality.

I. INTRODUCTION

Agile software development methodology is currently widely in use, due to its characteristics of rapid software development and accommodation to changing requirement at any level of development [1].

Agile was a significant departure from the other software development methodologies such as waterfall model & evolutionary model. Evolution of agile & other software development methodologies is shown in figure:

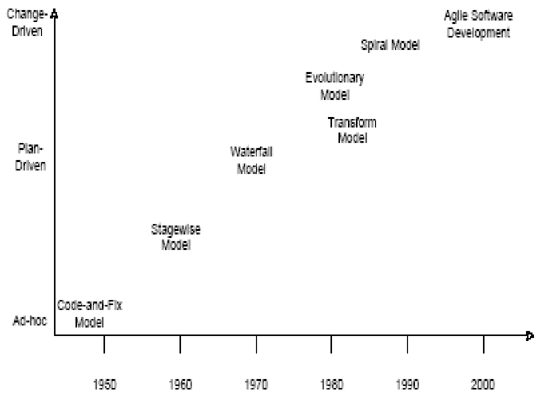

Figure 1: The Evolution of Software Process Models [2]

A. Agile Values:

Manifesto for agile software development state that “we are uncovering better ways of developing software by doing it and helping other do it. Through this work we have come to values:

a. Individuals and interactions over processes and tools
b. Working software over comprehensive documentation
c. Customer collaboration over contract negotiation
d. Responding to change over following a plan

That is, while there is value in the items on the right, we have the items on the left more.” [2, 3]

B. Agile SDLC:

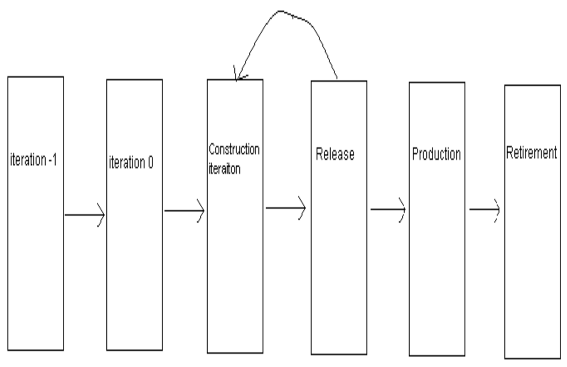

Figure 2: Agile SDLC [4]

The agile SDLC looks very much similar to traditional SDLC, but its not the case. Because agile SDLC is “collaborative”, “iterative” and “incremental”. The agile project developer’s work closely with their stakeholder to properly understanding their needs, they work as a team to implement and test their solutions, also quick feedback is get from holders. Instead of specialists handling artifacts to one another, and thereby injecting defects at various steps, the agile developers provide “generalization specialists”, with full life cycle skills.

a. Iteration 1(Select the project): Iteration 1 is the pre project planning phase. This phase includes various tasks:
   a) Define the business opportunity.
   b) Identify a viable for the project.
   c) Asses the feasibility.

b. Iteration 0: This is project initiation phase. The responsibilities of this phase are:
   a) Garnering initial support and funding for the project.
   b) Actively working with stakeholders
   c) Starting to build the team
   d) Modeling an initial architecture for the system

e) Setting up the environment
f) Estimating the project

c. ***Construction Iteration:*** During construction iterations agilists incrementally deliver high quality working software which meets the changing needs of our stakeholders. We achieve this by:
a) Collaborating closely with both our stakeholders and with other developers
b) Implementing functionality in priority order
c) Analyzing and designing
d) Ensuring quality
e) Regularly delivering working software
f) Testing, testing and yes testing.

d. ***Release Iterations (the End Game):*** During the release iterations (also known as the "End Game"), system is transitioned into production.

e. ***Production:*** The goal of the production phase is to keep systems useful and productive after they have been deployed to user community.

f. ***Retirement:*** The goal of the retirement phase is the removal of a system release from production, and occasionally even the complete system itself, an activity also known as system decommissioning or system sun setting.

C. ***Reusability:***

Now days, reusability has become an important part of software engineering. Reusability helps in "fast development ", "code reduction", "cost reduction" and "improve quality" of the project.

II. PREVIOUS WORK

There are various reuse approaches currently in use and are explained as follows:

a. ***Component Based Development:*** The CBSE is concerned with developing standardized components. A component is an independent executable entity that can be made up of one or more executable objects. When reusing components, it is essential to make trade off between ideal requirements and the services actually provided by available components.[5]

b. ***Patterns:*** Patterns are "Design & interaction" of objects. Patterns are building blocks for design and construction. With each pattern, small piecework is standardized into larger unit i.e. in architecture; a pattern is an architectural design or style.

c. ***Architectural patterns:*** It is basically an integration of components and are at higher level than design patterns. Actually, Architectural design describe overall pattern followed by the entire system. [6]

d. ***Design patterns:*** It shows the interaction between different components. These are descriptions of communicating objects and classes.[7]

III. PROBLEM DOMAIN & MOTIVATION

Agile software development is most popular model getting used in software industry these days. Agile software development model helps in rapid software development and less costly projects. [8]

But this model compromise with quality and reusability features. So a new concept is required to overcome this problem. This paper depicts a new concept which helps in introducing reusability concept in agile software development methodology.

IV. PROPOSED MODEL

The proposed model named "component repository based agile software development" is incorporating reusability in ASD. This model consists of three layers:
a. Agile software development life cycle
b. Software processes for adding & retrieve design patterns in repository.
c. Software design repository

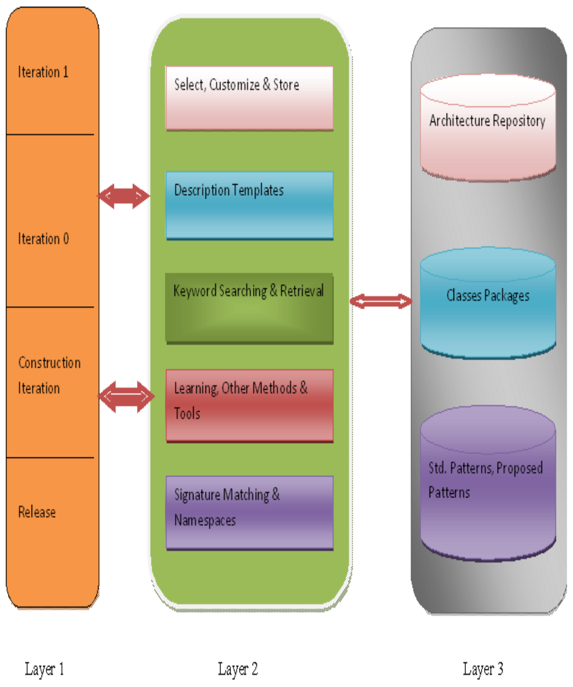
Iteration 1
Iteration 0
Construction Iteration
Release
Select, Customize & Store
Description Templates
Keyword Searching & Retrieval
Learning, Other Methods & Tools
Signature Matching & Namespaces
Architecture Repository
Classes Packages
Std. Patterns, Proposed Patterns
Layer 1
Layer 2
Layer 3

Figure 3: Component repository based agile software development Model

a. The proposed framework gathers information from agile SDLC stages. "Standard design of project" can be taken from the iteration 0 and construction iteration phases to store in repository. These standard designs can be used later by other project. While storing design patterns in repository proper naming is compulsory.
b. The secondary layer define the processes as follow:
   a) Selection customization and storage: This module is used for selecting, modifying and restoring artifacts from the repository.
   b) Keyword searching and retrieval: Keywords are names of pattern, object and architecture. These keywords are used for matching needed artifacts from the repository.
   c) Learning, other approaches and tools: This module represents the algorithms, which can be used for learning more design patterns & architecture.
   d) Signature matching and namespaces: In this module, signature matching helps in retrieval of methods, while namespaces retrieve the packages from the repository.
   e) Describe templates: Description templates are the basic structure of the components. This basic structure of the components helps in storage and retrieval of components from repository.

a. The third layer is "software design repository". Mainly three type of reusable data is store into repository given by:
   a) The reference architecture is stored & retrieved into the repository on the basis of technology used in the project such as .net, java, c#.
   b) Repository contains the classes and packages.
   c) Design pattern repository contains the standard design patterns & proposed design patterns.

A. Proposed Model Working:

The whole working of proposed model is divided into two major parts on the basis of the agile SDLC phases. In the 0 iteration phase, we gather initial requirement of the project. Then a logical view is prepared on the basis of initial requirement. After that a architecture design of the project is retrieve from the repository on the basis of technology used. If no architecture is found, then new architecture is prepare which can also be store in repository for future reuse. The next phase is construction phase. In this phase iteration planning is done on the basis of new requirements. While planning, design patterns and components are store and retrieve from the repository. After reuse of design patterns from repository, coding is done, and in end, refactoring is done for changing requirements. After that, technical experts apply test first driven approach to make project bug free. Reviews are make again and again, suggest to go with refactoring or stop after some iteration. The refactoring code is store as component into repository for future reuse.

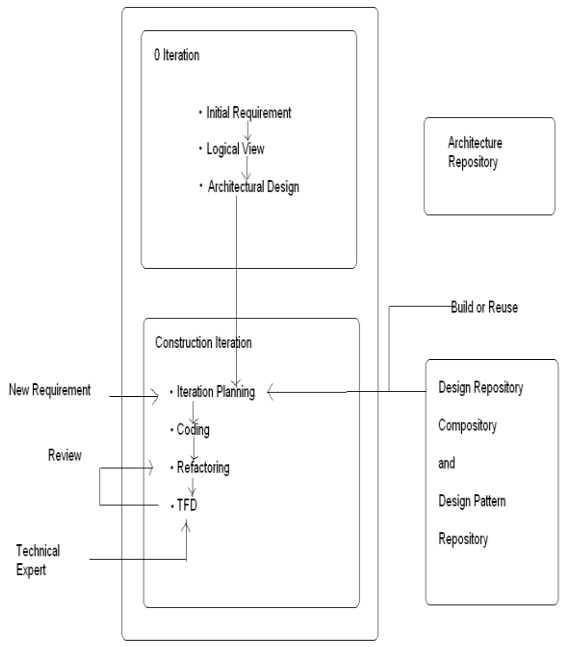

Figure 4: Software processes for adding & retrieve design patterns in repository

V. IMPLEMENTATION

To implement this system, a case study is used shown as follow:

Hotel reservation system is considered for implementation of proposed model. Each step of proposed model will be applied on hotel reservation system as explained below:

a. First of all, get all the initial requirement from the iteration 0 phase. Use this information to make a logical view of the project. Here, customer wants an online hotel reservation system, so that user has option of many hotels. From these hotels, customer must be able to select a hotel and reserve a room.
b. Prepare a logical view based on initial requirement. This logical view will be match with design patterns available in repository. As shown below:-

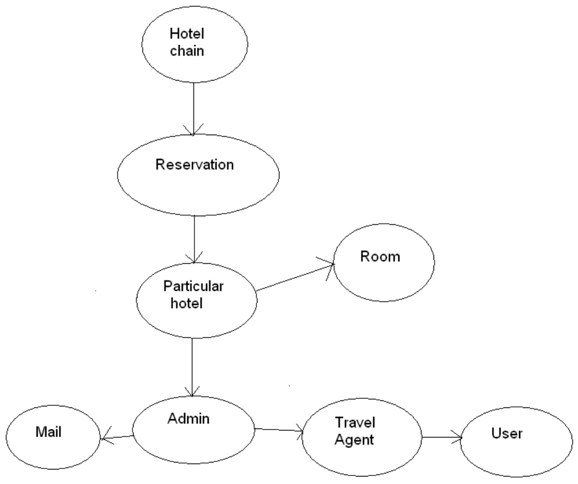

Figure 5: Logical view

c. Now based upon client's choice technology and reference architecture available in repository, we choose here Microsoft technology based architecture.
d. In the next iteration phase (construction iteration), based upon logical view & reference architecture a design pattern will be referred from repository. As repository is searched, the reservation pattern is found.
e. After getting "design Pattern" of related system, coding can also be undertaken.
f. Now refactoring is done according to expert judgment.
g. Next, test first design must be applied for the review.

At last, we get a refined design of the reservation system. As shown in figure:-

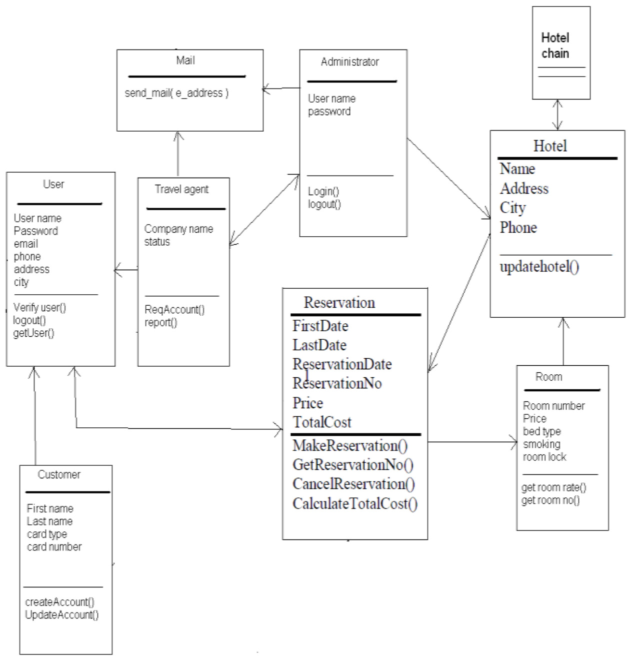

Figure 7: Refined Design of Hotel Reservation

VI. CONCLUSION

Agile software development is most popular model getting used in software industry these days. Agile software development model helps in rapid software development and less costly projects. But this model compromise with quality and reusability features. This dissertation depicts a new concept which helps in introducing reusability concept in agile software development methodology.

Basically this dissertation depicts a new framework to add reusability concept into agile software development model. The proposed framework consists of three layers. The first layer represent agile software development life cycle. The second layer represents software processes to introduce reusability. The third layer represents the reusable component repository.

It will enhance the reusability in agile software development. It will also reduce the developer's time and will improve the quality.

VII. FUTURE SCOPE

The work proposed in this paper is not still automated. A lot of work can be performed on it in future. A tool can be designed that would be capable of learning from components and design patterns stored in agile repository using machine learning algorithms or another approaches. And it can be further developed to make the whole or some part of the process automated.

VIII. REFERENCES

[1]. Dyba T., and Dingsoyr T., "Empirical Studies and Agile Software Development: A Systematic Review", Information and Software Technology, 2008, vol. 50, pp. 833-859.

[2]. Salo O., "Enabling Software Process Improvement in Agile Software Development Teams and Organizations", ESPOO 2006, VTT Publications 618, pp149 +app.96 pp.

[3]. www. Agilealliance.org.

[4]. Scott W. Ambler "The Agile software development life cycle" Ambysoft, 2005-2010, pp.3-7.

[5]. Ian Sommerville "Software Engineering" 2004, pp.27-28.

[6]. Gamma E., Helm R., Johnson R., and Vissides J., "Design Patterns: Elements of Reusable Object Oriented Software", Addison Wesely,1994.

[7]. Salvio J.D., StAphane J., and Faculdade C.D., "Object Oriented Software Architecture Design based on UML/PeriNet Approach for Deadlock Prevention and Real Time Systems", Journal of Computational Methods in Science and Engineering Sept. 09, 2005, vol. 5, pp. 67-83.

[8]. Turk D., France R., and Rumpe B., "Limitations of Agile Software Processes", 3rd International Conference on XP and Agile Processes in Software Engineering (XP 2002), May 2002.