ISSN (e): 2250 – 3005 || Volume, 08 || Issue, 8|| August – 2018 ||
International Journal of Computational Engineering Research (IJCER)

A Framework for Architecture Refinement into Extreme Programming

Mrs. Nagalambika Swamy1, Dr. L. Manjunath Rao2,Mr. Praveen K S3
1Associate Professor, Department of MCA, East West Institute of Technology, Karnataka, India
2Professor and Head, Department of MCA, Dr. Ambedkar Institute of Technology, Karnataka, India
3Associate Professor, Department of MCA,East West Institute of Technology, Karnataka, India
Corresponding Author:Mrs. Nagalambika Swamy

ABSTRACT: This paper introduces about a new framework aim to make an Extreme programming (XP) as a design centric along with code centric. This framework uses an Architecture refinement approach which uses a simple abstract design of XP is converted into more elaborated concrete components and provides details of each component such as classes with attributes, properties, operations and associations and this retains all essential properties of its abstract design, By using this proposed refinement approach, the design becomes simple and it closer to the implementation , reduce the "refactoring" principle of XP and improves the total execution time and code quality.
KEYWORDS: Agile Software Development, Architecture Refinement, Architecture space, Design space, Extreme Programming, Pair programming, Refactoring.

Date of Submission: 17-08-2018 Date of acceptance: 31-08-2018

I. INTRODUCTION

Since recent year havewitnessed the use of agile software development has been increasing, especially for medium-to-large scale projects. Extreme Programming(XP) is one of the lightweight agile development methodologies. XP focus on close relation between development team and customer through face to face communication, frequent delivery and rapid response to changes in requirements[1].

Extreme Programming (XP) avoids up-front design by concentrating only on current needs in environment of frequent changes in the requirements. Software architecture plays an important role as a bridge between softwarerequirements andits implementation. It provides an abstract description of a system.The architecture exposes specific properties,while hiding others. A good architecture ensures that a system will satisfy key requirements in the areas as interoperability, portability, performance, scalability and reliability. A horrific architecture can be terrible.Software architecture design approaches produce architectures to match functional and quality requirements before detailed design and implementation [2].

Software Architecture has a major impact on system efficiency andmaintainability. Architecture refinement allows us to more elaborated architecture design and keepsall the important properties of its abstract architecture.

XP provides new skills and challenges as designers need to learn how to do a simple design. By using a proposed refinement approach the design become clean and it takes close to implementation and reduce the "refactoring" principle of XP.

II. PROBLEM DEFINITIONS

Extreme programming (XP) is one of the agile development methodologies that is intended for adaptive projects where requirements are vague and not clear, one of the Key features of XP is the fast receptiveness to varying customer requirements [4].

- As Extreme Programming (XP) is focused on the code rather than on design,it may because problemas good design is extremely important for software applications. Also XP requires too much refactoring and time consuming [6].
- If programmers are located geographically, XP is not the best option [6].

III. PROPOSED FRAMEWORK

The framework aim to make extreme programming as a design centric along with code centric. Figure 1 shows the framework of proposed approach. In this paper, focus on refinement of architecture design is converted into more elaborated concrete design and gives details of each component such as classes its attributes, operations

and associations. Using refinement, system organizes the logic in the system so too many dependencies in the system can be avoided. Improve the total execution time and code quality.
Extreme Programming is one of the popular agile development approach. Figure 1 shows the new flow of software development cycle in XP. The cycle consists of the following stages: "planning," "architecture-space," "Design-space," "Coding," and "Testing". This cycle is used for implementing current requirements from the desired system, and it is repeated until the system development is completed.
a). Planning: the planning phase is used to collect customer requirement in the form of story cards, new stories were added continuously as requirements are not known in advanced. This meant that functionality was continuously added during the entire project.
Once the story cards have been prepared, the development team breaks these down into tasks and estimates the resources,effort and time required for implementation.

b). Architecture space: the Architecture space provides an abstract description of a system in terms of details database design and functional architecture. For the developersit's very important to understand the flow of the system with database and functional design. In this phase the customers, stakeholders, design team and testers are plays a major role. It should also have projects standards.
Architecture space serves as a blueprint for a system. It provides an abstraction to manage the system complexity and establish a coordination and communication mechanism among components. It provides the structured solution to meet all the technical and operational requirement, while optimizing the common quality attributes like security and performance.
No extra requirement is added early with the guess that it might be used later on and keep design simple as it saves time.

c). Design space: During the Design phase, the view of the application developed during the Architectural space is broken down into components and modules. Design space is the stage where business needs, customer requirements and all technical considerations come simultaneously in the formulation of a system.
The design space provides detail about the software class design, data structure, components and interfaces:

- The class and data design transforms analysis classes into design classes along with data structured to implement the software
- The class design helps to add and analysis of data structuresand minimizationof memory, execution speed and other measures.
- The interface design describes how the softwareinteroperates and communicates with systems and the user that use it.
- The component-level design transforms structural elements of the software architecture into a procedural description of software components.

The design space can be assessed for quality and be improved before code is generated and tests are conducted.
Architecture space provide a blueprint for system construction and composition where as design space provides a partial blueprint for development by elaborating each component details

d). Coding: Developers should write all the code in accordance with rules emphasizing communication through the code. Coding standards are to be followed and all code must have unit tests.
By using proposed framework it is not necessary to do pair programming, a single programmer independently can write efficient code by understanding the architecture and design space. It is clear that, in terms of resources consumed, pair programming generates an increase in the project budget. Two programmer's quite expensive resources sitting together all day working on the same code. Even though the experience has shown that the increase in development effort is compensated by improving design quality.
e). Testing: The rules of testing are:

- Unit tests must be done for all code before it's been released.
- When a defect is found, tests are to be written.
- Often acceptance tests are to be run.

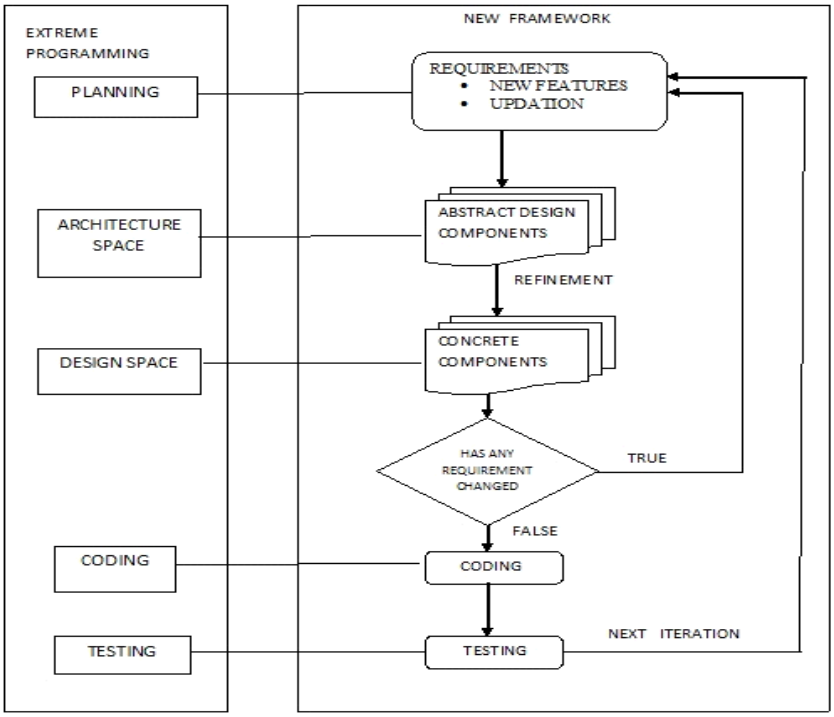

Figure 1. The Framework of our Approach

Figure 2 describes how architecture refinement converts architecture space to design space, a design space provides a partial blueprint for development by indicating the major dependencies and components between them, the major internal system interfaces and constraining what parts of a system may rely on services provided by other parts.

The rectangle box with ellipse symbol indicated the components and its functionalities. How functionalities are refined in term of class diagrams, which describe very detail the data structures, properties and operations with parameter used in each functionality and making XP the design centric.

The proposed solution describes how architectural refinement leads to a much clearer understanding of requirements, implementation strategies, and potential risks. Usually in XP people not process are supported, by using this framework the normal skilled person can develop the software just by seeing and understanding the framework.

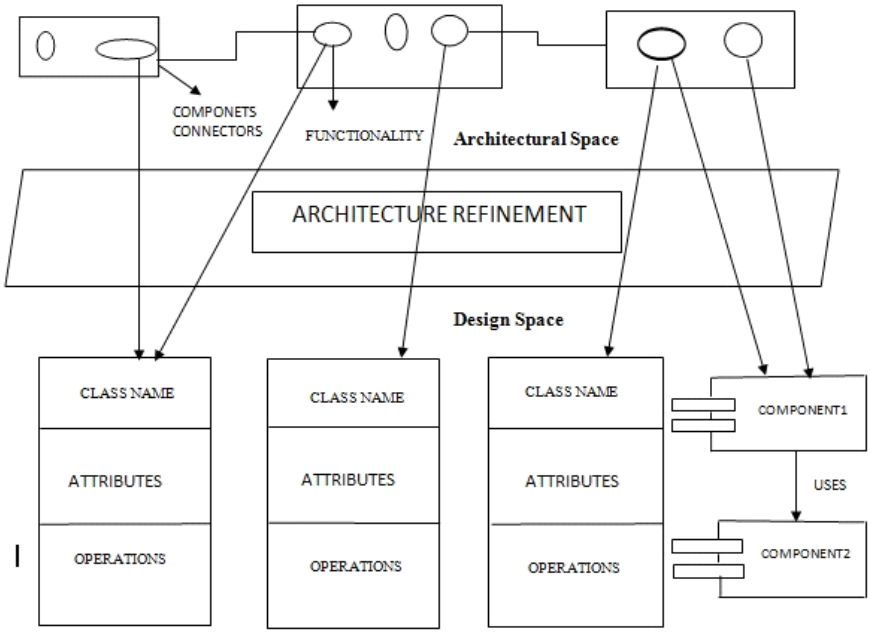

Figure 2. Architecture Space Refinements to Design Space

IV. ADVANTAGES

- Proposed framework can be used as code centric as well as design centric development.
- No need of pair programming, with the proposed framework single developer independently can do a efficient coding.
- A Normal skilled developer can do a coding.
- It minimize the code refactoring effort
- With current framework of XP if programmers are separated geographically can work.

V. CONCLUSION

This research paper gives a clear vision about a current XP framework, is used to make XP both design centric along with code centric and constantly redesigning through architecture refinement. The design is loosely coupled and simple as possible, thus making future modifications easier and achieving the XP values of simplicity and feedback. It helps to reduce the development time, effort and provide quality software.

ACKNOWLEDGEMENT

I am extremely thankful to express my sincere gratitude to my Husband, Parents, Guide, Colleagues and Friends for their kind co-operation and for providing valuable suggestion and constant encouragement for the improvement and successful completion.

REFERENCES

[1]. Rolf Njor Jensen et.al., Architecture and Design in Extreme Programming; Introducing "Developer Stories" Spinger-verlab Berlin Heidelberg 2006, pp. 133-142.
[2]. https://www.researchgate.net/punlication/233792525.
[3]. Marwan Abi-Antoun and NenadMedvidovic, "Enabling the Refinement of a software Architecture into a Design" Spinger-verlag Berlin Heidelberg 1996, pp.17-31.
[4]. Mrs. Nagalambika swamyet. Al., "Component Based Software Architecture Refinement and Refactoring Method into Extreme Programming". International Journal of Advanced Research in Computer and CommunicationEngineering Vol. 5,Issue 12, December 2016.
[5]. Bingzhi Gao; Xiaojuan Ban; QiangLv; XiaoliLi,A component-based method for software architecture refinementInternational conference on Intelligent Control and Information Processing, pp. 574-578
[6]. https://hygger.io/blog/disadvantages-and-advantages-of-extreme-programming.
[7]. Sukhpal Singh et.al.,"Enabling Reusability in Agile Software Development" International Journal of ComputerApplications (0975-8887) Volume 50-No.13, July 2012

Mrs. Nagalambika Swamy "A Framework for Architecture Refinement into Extreme Programming "International Journal of Computational Engineering Research (IJCER), vol. 08, no. 08, 2018, pp. 84-87