ENHANCEMENT TECHNIQUE OF IMPROVING SOFTWARE QUALITY: - REFACTORING

Ramesh Kumar*

Dr. R. K. Verma**

ABSTRACT

Refactoring is an important part of the evolution of reusable software and frameworks. Its uses range from the seemingly trivial, such as renaming program elements, to the profound, such as retrofitting design patterns into an existing system. Despite its importance, lack of tool support forces programmers to refactor programs by hand, which can be tedious and error prone. The Smalltalk Refactoring Browser is a tool that carries out many refactorings automatically, and provides an environment for improving the structure of Smalltalk programs. It makes refactoring safe and simple, and so reduces the cost of making reusable software.

*Research Scholar Singhania university, singhania, Pacheri Bari, Jhunjhunu (Rajashtan)

**Research Guide (Department of Computer Sc. & Application, KITM, Kurukshetra)

INTRODUCTION:

Definition of software quality by Pressman –Software quality is the conformance to explicitly stated functional and performance requirement, explicitly documented development standard, and implicitly characteristics that are expected of all professionally development software. Software quality is the degree to which a system, component, or process meets specified requirement. Another word software quality is the degree to which a system component or process meets specified requirement.

Software quality is importance field in software engineering. Software quality is most widely talked issue in the software industry in these days. A software quality measure how good software is designed and how the software will confirm to the design. In other word we can say that when we design of software it is important that we design good quality software. A quality of design measure how valid the design and the requirements are in a creating a worthwhile product of the challenges of software quality is that "everyone feels they understand it".

The methodologies currently employed to produce "software of quality" are based on the notion that quality is strongly related to rigor in the specifications and texts that appear throughout the software design process. Readability, modularity, modifiability, style, adequacy of comments, and good tests are considered to be important guidelines. Failure to achieve software of quality is attributed not to possible limitations in the process itself, but to inadequate knowledge. Once the process is completed, the designer delivers the software to the customer and declares the job done.

WHAT DOES REAL MEANING OF QUALITY?

The quality of the software we create most are affected company/organization's visibility depends on it. Most software related method including those described in IEEE software claim to assess or improve software quality in some way. So we must question what we understand about of software quality, and what our customer understand about the meaning of software quality. Measurement let us know if our technique really improves our software as well as how process quality affects product quality. We understand to know about the software quality we build in can affects the product's use after delivery to the customer and if the investment of time and resources assure high quality, high profit in other words , we want know that if we create a good software if provide a good business. Mostly people believe that quality is very important and that it can be improved. Organization and countries continue invest a great deal of time, cost and effort in improving software quality. If we want to

improve the software quality it depend on your depend on quality approach improvement. Some organization take a process based approach while other use product based approach. So we want say that the quality depend on approach they are used. Software quality improvement is necessary day by day to provide better software for the customer and get their satisfaction. In the software higher quality reduced development times go hand in hand.

IMPORTANCE OF SOFTWARE QUALITY:

Now a day the quality of software becomes a most importance issue. The importance of software quality increase day by day. A good quality of software gives us a better result. The importance of software quality according to Bill Venners: - "the going has been so good that the software profession has been able to treat quality as one issue among many. Increasingly it will become the dominant issue." Why?

The improvement of quality

Standards in software development are essential to efforts toward improving communication between customer and contractor, reducing software costs throughout the entire life cycle, and improving overall software quality. Organizations have a number of viable choices to consider when it comes to applying software quality process improvement methodologies. Quality improvement is critical to the success of software development organizations.

The following are technique that incorporate and derive the quality improvement.

QFD- Quality function deployment (QFD) is a "method to transform user demands into design quality, to deploy the functions forming quality, and to deploy methods for achieving the design quality into subsystems and component parts. QFD links the needs of the customer (end user) with design, development, engineering, manufacturing, and service functions. Quality Function Deployment (QFD) is the systematic translation of the "voice of the customer" to actions of the supplier required to meet the customers' desires, based on a matrix comparing what the customer wants to how the supplier plans to provide it.

TQM- TQM is an integrative philosophy of management for continuously improving the quality of products and processes. TQM capitalizes on the involvement of management, workforce, suppliers, and even customers, in order to meet or exceed customer expectations. TQM takes into account all quality measures taken at all levels and involving all company employees Total Quality is a description of the culture, attitude and organization of a company that aims to provide, and continue to provide, its customers with products and services that satisfy their needs.

METHOD TO IMPROVE THE QUALITY:

There are many methods are used to improve the software quality:

1. Refactoring
2. Code inspection or software review
3. Documenting code

Refactoring:

Refactoring means improving the design of existing code. In the context of test driven development, code is refactored to removed duplication and express intent as soon as it is written and passes the tests. Refactoring is a disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior. Its heart is a series of small behavior preserving transformations. Refactoring is the process of changing the design of exiting software modifying the structure of its code without affecting its external behavior. A variety of specific refactoring might be employed in that process. Each refactoring describe discipline way to modify code order

to achieve a specific design change without introducing bugs, leaving the code in a less complete state or otherwise tempering with the way the software behave. The concept of refactoring covers practically any revision or cleaning up of source code, but Fowler consolidated many best practices from across the software development industry into a specific list of "refactorings" and described methods to implement them in his book, Refactoring: Improving the Design of Existing Code. While refactoring can be applied to any programming language, the majority of refactoring current tools have been developed for the Java language.

One approach to refactoring is to improve the structure of source code at one point and then extend the same changes systematically to all applicable references throughout the program. The result is to make the code more efficient, scalable, maintainable or reusable, without actually changing any functions of the program itself. In his book, Fowler describes a methodology for cleaning up code while minimizing the chance of introducing new bugs. To rewrite existing source code in order to improve its readability, reusability or structure without affecting its meaning or behavior

Refactoring is typically done in small steps. After each small step, you're left with a working system that's functionally unchanged. Practitioners typically interleave bug fixes and feature additions between these steps. So refactoring doesn't preclude changing functionality, it just says that it's a different activity from rearranging code.

The key insight is that it's easier to rearrange the code correctly if you don't simultaneously try to change its functionality. The secondary insight is that it's easier to change functionality when you have clean (refactored) code.

REFACTORING TOOLS

Refactoring means modifying existing source code in a structured and incremental way. In refactoring, the behavior of the code is not radically changed. Refactoring promotes reuse as the refactored code is potentially easier to use in another context or for another purpose. Refactoring is a generic programming technique that is not tied to a specific implementation language. We refer to the Refactoring web site of Martin Fowler for the various refactoring techniques.

Use of Project Analyzer to refactor VB code

Project Analyzer is a Visual basic source code analyzer, optimizer and documenter. In other words, it's a static code analyzer for VB, VB.NET, ASP.NET and Office VBA that detect quality control issues like dead code, un-optimal variable declarations, un-optimized syntax, memory leaks and functionality issues such as missing event handlers or questionable tab order from your project and instantly remove it. Project Analyzer is a tool that helps refactor existing Visual Basic code. It is useful in two ways.

Firstly, Project Analyzer helps to identify code that would potentially benefit from the refactoring techniques. Hiding is also possible for variables that are not required outside of their class/module. It may also possible to make a variable local to a procedure. In most cases, refactoring is best done manually, because code changes need human consideration. Automated changes are meant for routine tasks only.

Visustin - Refactoring with flow charts

Another tool that is beneficial in the refactoring process is Visustin. Logic errors easily appear in code that contains nested conditionals, loops and jumps. A flow chart displays program logic in a more comprehensible manner, not tied to the order that the underlying code is written in.

A long procedure may contain the same line of code in several locations. It is probable that you can change the logic to eliminate the duplicate lines.

A logical structure may be duplicated in two or more procedures (possibly via copy & paste coding). When detected, this logic is best moved to a new function and called from the other functions.

A complex algorithm is usually best to split in several functions. A flow chart is useful for finding the blocks from which to form new functions.

References:

1. Pressman
2. http:// en.wikipedia.org/wiki/software quality
3. Steve McConnell
4. Peter J.Denning
5. Barbara Kitchen ham
6. Mike Bria
7. Bill Venners
8. Bertrand Meyer
9. Martin Murray
10. http://en.wikipedia.org/wiki/Toyota_Production_System
11. Lockheed Martin Corporation
12. Martin Fowler
13. Kent Beck; John Brant
14. http://www.aivosto.com