The Design Quality Comparison and Inspection Efficiency for Hardware and Software*

Zhao Fengyu[1†] and Ma Yizhong[2]

[1] College of Computer Engineering
University of Shanghai for Science and Technology
Shanghai, 200093, China

[2] School of Economic and Management
Nanjing University of Science and Technology
Nanjing, 210019, China
E-mails: [1]zhaofengyv@usst.edu.cn, [2]yzma-2004@163.com

Abstract

The process of producing software differs in many aspects from that of traditional manufacturing. Software is not manufactured in the classical sense. Development of software more closely resembles the development effort that goes into design new product [1-3]. In this article, we first describe the foundations of process improvement, which all processes can share. The process improvement differences between software and manufacturing process are then discussed, and a defect driven process inspection and improvement is introduced. Based on the discussion, two experiments were designed and the results of the results were collected. Through the comparison, we found that some efficient quality improvement approaches can be easily adapted in the software improvement and that the inspection efficiency is also significant.

Key Words: Software Quality, Human Error Inspection, Inspection Comparison, Hardware Design, Software Design

1. Introduction

In the past decades, the scale of software is increasing. Initially, a few individuals could handcraft small programs; the work soon grows beyond them. Teams of one or two dozen professionals were then used, but success was mixed. Today as the scale of our work continues to grow, the large projects require the coordinated work of many teams, and the complexity is outpacing our ability to solve problems intuitively as they appear. For example,

† Corresponding Author

* This paper is supported by The National Natural Science Foundation of China (No.70372010) and Development Foundation of Shanghai Education Committee (No. 04EB14).

the Windows 2000 has over 50 million line of code, over 3000 engineers were participated the development and the engineers were divided into over 500 small teams [5]. So more discipline, systematic and engineering approaches to software process management are required. Whenever the human being faces a big problem, they tried to search for similar scenarios that they are familiar with. By comparing the similarities and differences between the problems and similar scenarios, a process to solutions of the problem can be originated from the knowledge that they learned from the scenarios. Among the methods, one structure approach is software engineering, which is the establishment and use of sound principles in order to obtain software economically that is reliable and works efficiently on real machine. In this area, a lot of researches have been deployed and many research results were published to deal with different problems occurred in software development. To name a few, there are risk analysis, project management, requirement engineering, configuration management, software process improvement, such as CMM, ISO/IEC 15504 (also called SPICE), BOOTSTRAP [6] etc.

Based on a project for U.S. air force, which is to provide a guide to the military service in selecting capable software contractors, a five-level process maturity assessment framework (capability maturity model, CMM in brief) was given. The basic principle of the CMM framework is process continuous improvement advocated by quality gurus, which is widely practiced in manufacturing industries.

Though the basic principle of software process is same as process continuous improvement practiced in manufacturing industries, but the software process has its unique characteristics. In this article, we first describe the process improvement foundations, which can be shared by the two processes. Some differences between software process and manufacturing process are then discussed. Based on the discussion, two experiments were designed and the results of the results were collected. Through the data analysis and comparison, we found that some efficient quality improvement approaches can be easily adapted in the software improvement.

2. The Differences of Hardware and Software Design

The process of producing software differs in many aspects from traditional production process. Software is not manufactured in the classical sense. Development of software more closely resembles the development effort that goes into a new product design [4]. Both software and manufactured products can have high quality through good design and developing process, but the quality problem during production is only existed in manufactured products. Watts S. Humphrey listed 8 factors that make software different from other engineering fields. The following three items are the most significant factors excerpted from them.

- The insignificant cost of reproducing software forces software solutions to many late-discovered system problems.
- In other engineering fields, release to manufacturing provides a natural discipline that is not present with software. When some independent party must sign off on production costs and schedules, the design definition is either complete or all work stops until it is.
- The software discipline is not grounded in natural science. As an artifact of human ingenuity, software does not rest on a stable foundation of physical principles.

Except above three items, the following two items can also be found.

- Software designers are not specialized with domain knowledge. In other engineering fields, it is not the case. One bridge designer will not be asked to design a building, though the two project share some similarities, such as concrete and reinforce bars are used, and architectural mechanics principles are same.
- For software development, the efforts of analysis, design and implementation are generally made by same group of engineers.

While comparing the software development process and the design for a new product in other engineering area, though there are some differences, they do have the similar scenario:

- The design based on the requirement of customer. In this aspect, all the requirement management and technology are applicable, such as QFD, system design and function design etc.
- The design team should be organized and motivated in order to achieve superior design quality and performance. Management must establish challenging quality goals and strive to meet them.
- The superior design is resulted from the knowledge of designers. After an extensive survey, Curtis claimed that success programs were always designed by people who understand the application.
- All the processes were influenced by system variables and random variables, where the system variables determine the process capability while the random variables cause the process variation. In design process, the engineers, management methods, design tools and technologies comprise of the system variables; the human error, environment changes are random variables.

3. The Quality for the Development of New Products

The quality of newly produced product is from two aspects. One is the designed quality

for the product, which is positive quality. The second is the producing quality, which is negative. In the analysis and design phase, designers will composite their knowledge, creativities and advanced technologies to obtain a good solution to meet customer requirements and other implicated requirements, such as reliability, usability, maintainability, readability, portability etc. The design quality for the product should be a positive number. In this phase, if designers do not fully understand the customer requirements, or designers are lack of knowledge in design activities, the design quality might be lower. During implementing or manufacturing phase, since the human activities are error-prone [2], some human error might inject into the internal product. The quality problem caused by these factors is negative for the product quality. In the best cases, by means of most effective inspection and testing, the number of the human errors is zero, i.e., the producing quality is zero. The product quality equals to the designed quality.

For the design quality, it is determined by three factors: the talent of designers, design methods and supporting tool, and design process model. There are tremendous studies and results about the way to improve design quality, such as value-added designs, requirement engineering, QFD, object oriented analysis, concurrent engineering, prototype and evolution design etc.

For the producing quality for software, since the specification and variation for most quality attributes are difficult to obtain, it is impossible to measure and monitor the producing (detailed design, coding) quality by statistical process control. But, the inspection/review can do the same mission. For the reasons list bellow, a defect driven process control and improvement is needed.

- Any error injected by human activities is harmful. In a program with millions LOC (lines of code), even one error line may cause big quality problems.
- The inspection activities can find the errors at the early stage of production. So it is more cost efficient than testing.
- Most errors injected by software engineers are difficult to be found by automatic testing. While effective way to pick out the errors is through peer review, as the proverb goes the truth is easily seen by another pair of eyes.
- The inspection/review is the objective measurement to measure the producing quality. It has proven that the code modules with more inspected errors have more defects when they are tested.

Through consulting with some senior engineers of manufacturing industries, we found that the inspection/review method is extensively used by engineers when they design a new specific product. In the project described bellow, the design engineers spent as much the same time in inspection as in design. Unfortunately, in software engineering, more time and effort

have been spent in testing instead of inspection to assure the quality for software. Even in the most advanced software companies, it is the same case. Table 1 shows the personnel ratio in two projects conducted by Microsoft [5]. Test Engineers/Development Engineers ratios are 2.5 and 1.9 for Exchange 2000 and Windows 2000 respectively.

Table 1. The personnel distribution of two projects conducted by Microsoft [5]

	Exchange 2000	**Windows 2000**
Project Manager	25	250
Development Engineers	140	1700
Test Engineers	250	3200
Test Engineers/Development Engineers	2.5	1.9

4. The Design Projects

In order to compare the inspection procedure and efficiency, we selected two design projects. One is the design and manufacturing of a rotary press machine. The other is the development of statistical quality control (SQC) software. These two projects are described bellow briefly.

4.1 The Rotary Press Machine

The rotary press machine is a specific machine, which is designed to manufacture automobile parts. The market price of such kind machine is more than one million Yuan (Chinese Yuan). After investigating and consulting with some senior engineers, the top manager of a factory decided to design one for the purpose of saving money. The machine consists of mechanical component, cooling system, lubrication system, PLD control and touch screen. The design team has five engineers. After two months design, 261 A4 size drawings had finished. In order to assure the design quality, the chief designer inspected the designed drawings and then guided his workers to design the machine.

4.2 The SQC V2.0 Software

In 1994, we developed a SQC program (version 1.0) with C language. This release of SQC software is based on DOS operating system. As the quick development of computer technology, DOS operating system was substituted by Windows gradually. At the same time, some new methods and application of SQC have been developed. From 2000 to 2001, we redesigned the SQC software (SQC V2.0). The SQC V2.0 was implemented by Java, which

is a platform free and an Internet programming tool. The software consists of data analysis charts, traditional attribute and variable control charts, short run data transformation and control charts, small variation control charts, measurement system analysis and control etc.

In the SQC v2.0 software development project, there are six people participated in the design and implemented the project. Since most of them worked in part time and some time was spent on studying for new statistical methods, the whole project takes 18 months. The SQC software consists of 10000 lines of code except some images. The printed codes are about 300 A4 pages.

5. Inspection Task and Data Collection

5.1 Inspection Task

The inspection task was conducted by the chief designer for each project. The main purposes of the two inspection procedures are shown in Table 2.

Table 2. Inspection tasks for the two projects

	Rotary press machine	**SQC V2.0**
Inspector	Chief designer	Chief designer (project manager)
When	After detailed design	After coding
Duration	1.5 months	1 month
Inspection material	Drawings	Code
Main focus and task	• Design errors and drawing errors • New design, new technology • Easy to assemble • Easy to operate	• Coding errors: variable initialization, incomplete condition, and algorithm error. • Class simplifying • Up stream design errors • Up stream design misunderstanding

5.2 Data Collection

The inspection task of these two projects is a little bit different. In the rotary press machine, the inspection task focuses on design drawing errors. Except inspection on drawing errors, the chief designer also adopted some new technologies to substitute old ones and he also concerned the down stream manufacturing and assembling. While for the SQC V2.0, the inspection mainly focus on coding errors, up stream detailed design errors and misunderstanding are also collected. The collected data and classification are shown in Table 3 and Table 4.

Table 3. The inspection results and modification on the design drawings of rotary press machine

Error catalog	Error number	Description
Design error	10	Errors found in all the drawings
Illogic structure	28	The original designs are based on traditional principles, but for the specific machine, they are illogical.
New structure	5	The original designs have been changed because new structure is better while the cost is also affordable. Such as an integrated component of oil pump and electromotor instead of a separated pump and a separated electromotor.
Technological changes	3	Some changes for easy assembling and adjusting
Other	4	Minor adjustments.
Total error inspected	50	All the inspection errors and modifications

Table 4. The inspection results and modification on the codes of SQC V2.0

Error catalog	Error number	Description
Design error	2	2 classes are defined incorrect.
Algorithm	12	Algorithm function is not realized.
Condition	22	Divided by 0; square root for a negative expression; data exception, incomplete condition.
Variable initialization	5	Forgetting or wrong initialization
Design Misunderstanding	1	The programmer misunderstood the design, so the whole codes of a module are wrong.
Design modifications	4	Classes simplicity, classes merging.
Total error inspected	47	All the inspection errors and modifications

6. Result Comparison and Conclusion

Both of the two projects have resulted in high quality products. For the rotary press machine, through 1 and a half-month inspection, 10 design errors have been found and some design modifications have been made in view of new technology, assembling and manufacturing. The rotary press machine was built with very high quality. While for the SQC V2.0 software, after the inspection, it had gone unit test, an integrated test and trial run. Up to now, only a few minor (label and text field for example) bugs have been reported. The reason is that inspection processes of software project have paid less attention on the text in strings.

Though software development process and software products have many unique characteristics, it is still can use the experience learned in hardware design and the knowledge of other engineering area. The peer inspection method, which is widely practiced in other engineering areas, is also an efficient approach in detecting and preventing human injected errors. As the experiments shown, the chief designer can employ his update knowledge of technology and domains to optimize the product design and to enhance software quality. Though some successful process improving methods such as experiment design and statistical quality control can not easily used in software development process, some traditional practice still can be effectively applied in software quality improvement.

References

1. Jan Rickard Westerberg(1998), Experience Reuse in Ericsson Software Technology AB. CIV, ISSN 1402-1617.
2. Niclas Ohlsson(1999), Software Quality Engineering by Early Identification of Fault-Prone Methods. ISSN 0280-7971.
3. Paul E. Plsek(2000), "Creative Thinking for Surprising Quality," *Quality Progress*, pp. 67-73, May.
4. Watts S. Humphrey(1989), Managing the Software Process, Addison-Wesley.
5. Chen Honggang, etc.(2002), The Science and Art of Software Development, Publishing House of Electronics Industry.
6. Sami Zahran(2003), Software Process Improvement-Practical Guidelines for Business Success, China Machine Press.