Research of the Software Engineering Methods Applied in Nuclear Fuel Analysis Software Development

Ying PENG, Hong-yang ZHANG and Song-cheng WANG
19F, Science Building, Shangbu Road, Futian District, Shenzhen, China

Keywords: Software engineering, Nuclear, Fuel analysis.

Abstract. The Nuclear Fuel Analysis software is mostly designed by non-computer professionals; they have some defects in readability, reusability, operation etc. This paper firstly introduced the concept of software engineering. Secondly we expounded it. Finally, we proposed a series of software engineering methods used in the nuclear fuel analysis software development.

Introduction

In December 2, 1942, the world's first nuclear reactor was designed by Enrico Fermi, and in February 15, 1946, the world's first electronic computer "ENIAC" unveiled. Nuclear fuel research of calculation and analysis with the development of computer also began to use "software" form to replace artificial calculus. Nuclear fuel calculation and analysis relating to physical and chemical aspects, and the calculation process is very complex, early, this kind of software is mainly composed of the associated physical personnel to develop, because at that time had not yet emerged the idea of software engineering, so this kind of software today there exists many defects.

Present Situation and Defects

In the domestic and foreign related software research, we found the software mainly has the following shortcomings:

• Bad readability, the programs were developed relatively early, and mostly for the non-computer professionals are prepared, there are many obsolete logic structure, is not conducive to reading.
• Low reusability, the programs were used structured design, similar function code in the program are not universal.
• Poor operation, procedure using the input card and command line mode of operation, requiring specialized learning.
• Worse interaction, data processing is through a text file format, data representation is not intuitive, retrieval difficulties, calculation results in adverse save.

We can see the cost-complexity curve of software development [1] in figure 1, which one is used software engineering methods against the other is not.

![Figure 1. Cost of development with software complexity.](image)

In the basis of summarizing forefathers’ experience, we introduce the modern software engineering ideas to guide the whole process of software development.
Background Analysis

Software Engineering

Software Engineering (SE) is a research project method construct and maintains effective, practical and high quality software. It relates to the programming language, database, software development tools, system platform, standard, design pattern etc. In modern society, various industries and almost all of the computer software application, such as industry, agriculture, banking, airlines, government departments etc. These applications have promoted the development of economy and society, make people work more efficient, while improving the quality of life.

SE’s Goals

The goals of SE is: given the cost, progress under the premise, has developed the modifiability, validity, reliability, understandability, maintainability, reusability, flexibility, portability, traceability and interoperability and meet the needs of users of software products. We can see the cost to correct a defect in development phases [2] in Figure 2.

![Figure 2. Cost to correct a defect in development phases.](image)

In pursuit of these goals is helpful to improve the quality of software products and the development efficiency, reduces the maintenance difficulty. Introduction to be as follows [3]:

**Modifiability.** Allowed to modify the system without increasing system complexity, it supports the software debugging and maintenance; this is one of the more difficult to reach the target.

**Efficiency.** The software system can be used most effectively computer resources of time and space resources. On many occasions, in the pursuit of time effectiveness and availability will conflict, then have to sacrifice time efficiency for space validity or sacrifice efficiency of space for time effectiveness. Time / space compromise is often appeared. Experienced software developers can skillfully use eclectic concept, in the specific physical environment in the realization of the user's needs and their design.

**Reliability.** Prevent system invalid caused by design faultiness of the software with recall caused by improper operation software system failure ability. For a real time embedded computer system, reliability is a very important goal, such as nuclear power plant operation, if reliability cannot be guaranteed, if there are problems with the consequences will be unbearable to contemplate. In software development, coding and testing process must be put in the important position of reliability.

**Understandability.** The system has clear structure, can directly reflect the demand problem. Intelligibility is helpful to control the complexity of software system, and software maintenance, transplantation or reuse.

**Reusability.** Concept or functions relatively independently of one or a group of related modules are defined as a software component. Soft components can on many occasions the application degree called component reusability. Reusable software component should have a clear structure and annotations, should have the correct coding and lower time / space overhead. Reusability is helpful to
improve the quality of software products and efficiency of development, contribute to reducing software development and maintenance costs.

Adaptability. Software in different system conditions, so that the user needs to be satisfied degree of difficulty. Adaptability of software should be used in popular programming language code, in a popular operating system environment, using standard terminology and format documents. The higher adaptable of the software has, the easier to use.

Traceability. According to the software demand for software design, program for positive trace, or according to the program, the software design of software requirements for reverse tracing ability. Software traceability depend on software development at all stages of document and program integrity, consistency and comprehensible. To reduce the complexity of the system will improve the software traceability. Software testing or maintenance processes or procedures during the period of execution when problems arise, should record the program events or related module all or part of the instruction field, in order to analyze the causal relationship, tracking problems.

Interoperability. Interoperability means a plurality of software elements mutual communication and coordination ability to fulfill the task. In order to achieve interoperability, software development usually follows a standard, supported the compromise standard environment for software elements between the interoperable facilitation. Interoperability in distributed computing environment is very important.

Software’s Lifecycle
As with everything else, the software also has a gestation, birth, growth, maturity, decline and fall of the process, this process is the computer software life cycle. This cycle is generally divided into the following six stages [4]:

Plan. To determine the software system development goal, given its function, performance, reliability and interface requirements, by the analyst and user cooperation, research performed in the software task feasibility, discuss the problem and possible solutions, the available resources (computer hardware, software, such as manpower, cost ), can obtain benefit, development progress make estimation, developed to finish development task implementation plan, together with the feasibility study report, submitted to management review.

Analysis. Deal with the development of the software requirements analysis and gives a detailed definition. Software developers and users of common to discuss and decide what needs can be satisfied, and for its exact description. Then write a software requirements specification or system function description and preliminary system user manual, and submit a review.

Design. Software engineering design is the core technology of. In the design phase, design personnel to have identified the needs into the corresponding system structure. The structures of each component are well-defined modules, each module and some relative to the requirements, the so-called outline design. And for each module to complete the work were described in detail, as the source program to lay a foundation, the so-called detailed design. All design considerations should be with the design specifications of the form to be described, for subsequent work, and submit a review.

Coding. The software design into a program design language "source code", this step is called encoding. The code should be well structured, clear and readable, and consistent with the design.

Testing. Testing is an important means to ensure software quality, the main way is in the design of test case based on the test software components. Is the first of the unit test, for each module in the structure and function of the existing problems and to be corrected; followed by an integration test, the tested module sequentially assembled; finally according to the provisions of the demand, item by item recognition test, decided software has been developed which is qualified or not, whether delivered users.

Operating. The software has been delivered to put to use, will enter the stage of operation. This stage may last several years or even decades. Software running in May due to various reasons, it needs to be revised. The causes may include: operation were found in the software error correction; in order to adapt to changes in the work environment and make the appropriate changes to the software; in order to enhance the software function to do change.
Solutions

According to the practical situation of the nuclear fuel analysis software development, we implemented the following methods:

**Use a Phased Lifecycle Plan to Manage Project**

Statistic makes clear, more than half of the project failed because of bad planning by. In software development and maintenance of long life cycle, needs to do many different jobs. This means that, should the software life cycle is divided into several stages, and accordingly make feasible plan, and in strict accordance with the plan of software development and maintenance management. In the software life cycle should be specified and strictly enforce the 6 plans: project summary plan, program, project control plan milepost, control plan, verification plan, operation and maintenance plan.

**Insist on Phase Review**

In the software development process, most of the errors are due to wrong coding. Discovered too late, correcting it will cost 2 to 3 orders of magnitude. Therefore, software quality assurance work cannot wait until after the coding end, should adhere to the strict phase review, so as to find the error [5].

**Implement Strict Control of Product**

Good demand analysis of software development is the guarantee of success, but the practice proves, demand changes are often inevitable. So we should adopt scientific control technology products to comply with this requirement, it is to adopt change control, also called the baseline configuration management. When the needs change, the documents and codes other stages of the other phase also needs change corresponding, in order to ensure the consistency of the software.

**Use the Modern Software Design Technology**

From structured software development technology in the sixties and seventies to the nearest object oriented technology, the development of computer technology change rapidly. Use an advanced technology can improve the efficiency of software development, but also can reduce the cost of software maintenance.

**The Results Should Be Able to Clearly Review**

Software is a kind of invisible, not a logical product. Software development group working progress circumstance visibility is poor, difficult to evaluation and management. For better management, should according to the software development goals and deadlines, as far as possible to clearly define the development team's responsibility and product standards, so that the resulting standard can clearly examine.

**Continuous Improvement**

Not only should actively adopt new technology of software development, we must pay attention to sum up experience ceaselessly, collect progress and consumption data, do error types and problems of statistics reports. These data can be used to assess the effect of new software technology, also can be used to specify must focus on the issues and should be given priority for research tools and techniques.

**Acknowledgement**

This research was financially supported by the China Nuclear Power Technology Research Institute.

**References**