




SOFTWARE MAINTENANCE: A TUTORIAL

BY KEITH H. BENNETT



2008.10.13
소프트웨어
200310612
조보경

Software Engineering Field

- Main problem of software engineering
 - Scale and Complexity of the software
- Goal of software maintenance
 - Software system should be
 - Delivered on time
 - Fully meet requirements
 - Applicable in safety critical systems

Software Maintenance

- Definition:
Process of modifying software system or component *after delivery* to correct faults, improve performance or other attributes, or adapt to a change in environment
- 40~90% of total life cycle costs according to surveys
- “Applications backlog” occurs when unable to undertake maintenance quickly, safely and cheaply required by business needs/marketing

Types of Software Maintenance

1. *Perfective maintenance* – changes required as a result of user requests.
2. *Adaptive maintenance* – changes needed due to change of OS, hardware or DBMS
3. *Corrective maintenance* – the identification and removal of faults in software
4. *Preventative maintenance* – changes made to software to make it more maintainable
 - Majority of maintenance is perfective maintenance

Requirements of maintenance

- Quickly accomplished
- Cost Effective
- Reliability should not be degraded
- Maintainability should not be degraded – future change will become more expensive
 - “Laws of evolution” by Lehman
 - Ultimately maintenance will be almost infeasible or software becomes “legacy system”

Problems of software maintenance

- Domino or ripple effect – change in source code may cause substantial changes to documentation, design and test suites.
- 3 categories of maintenance problems
 - Alignment with organizational objectives
 - Process issues
 - Technical issues

IEEE Standard for software maintenance (1994)

- Iterative approach, differentiate development process and maintenance.
- Four key stages:
 - **Help Desk:** preliminary analysis of problem received to determine sensibility
 - **Analysis:** choose solution after managerial and technical analysis
 - **Implementation:** implement chosen solution
 - **Release:** change is released to customer

Overview of IEEE standard for software maintenance

- 7 stage activity model
 - Problem Identification
 - Analysis
 - Design
 - Implementation
 - System Test
 - Acceptance Test
 - Delivery
- Each of the seven activities has 5 associated attributes
 - Input life cycle products
 - Output life cycle products
 - Activity definition
 - Control
 - Metrics

Technical aspects of software maintenance

- Required technology is similar to initial development with minor changes
- Impact analysis is needed for maintenance
 - Translation of user expressed problems into software terms to decide viability of problem
 - Identify primary components to be altered
 - Investigate alternate solutions
- Determine best solution based on result of impact analysis

Technical problems

- Ripple effect
 - changes made to a software component have a tendency to be felt in other components
- Static Analysis and dynamic analysis are used
- Impact Analysis identifies further objects impacted by changes until no further objects can be identified.

Traceability

- Provide semantic links that can be used to perform impact analysis
- Some links are hard to determine
- Most impact analysis is done at code level
- Documentation is modeled using a ripple propagation graph for analysis
 - Allows analyze assess costs without reference to code
- Research that allows transformation of specifications to executable code and vice versa are underway (1995)

Legacy Systems

- No formal definitions
 - Very old system that has been heavily modified
 - Usually large scale
 - written in obsolete language
 - no documentation
 - large quantities of live data are utilized by system
 - still functional
 - hard to meet growing needs
 - expensive to replace
- Most software today will end up as legacy software in 20 years.

Reverse Engineering

- Definition
 - The process of analyzing a subject system to identify the system's components and their inter-relationships, and to create representations of the system in another form or at higher levels of abstraction
- Passive:
 - does not change the system nor produce new system
 - Provide help in program comprehension
- Two types:
 - Re-documentation
 - Design recovery

Methods of reverse engineering

- Slicing
 - Static analysis technique
 - Only the source code that affect a nominated variable are examined
- Tools to attach notes the the source code
 - Avoid re-documentation of whole system
 - Stable part of code are not studied
 - Cost saving

Criteria for reverse engineering

- List of criteria
 - Management criteria
 - Quality criteria
 - Technical criteria
- Analysis should start with business rule contained in legacy systems
 - legacy system represents accumulated experience

Reverse Engineering Techniques

- Use of control flow and data flow graphs
 - Restructure of control flow
 - Commercial tools are available to support maintenance
- Program plan or cliché approach
 - Majority of program uses generic design ideas
 - Matching generic design patterns in source code from library
- Source language independent approach
 - Design of intermediate languages (UNIFORM, WSL)
 - Different languages are handled by adding front ends
- Discovery of abstract data types in existing code using tools