Object-Oriented and Classical Software Engineering

POSTDELIVERY MAINTENANCE

Overview

- Why postdelivery maintenance is necessary
- What is required of postdelivery maintenance programmers?
- Postdelivery maintenance mini case study
- Management of postdelivery maintenance
- Maintenance of object-oriented software
- Postdelivery maintenance skills versus development skills
- Reverse engineering
- Testing during postdelivery maintenance

Overview (contd)

- CASE tools for postdelivery maintenance
- Metrics for postdelivery maintenance
- Postdelivery maintenance: The MSG Foundation case study
- Challenges of postdelivery maintenance

Postdelivery Maintenance

- Postdelivery maintenance
  - *Any* change to *any* component of the product (including documentation) after it has passed the acceptance test
- This is a short chapter
  - But the whole book is essentially on postdelivery maintenance
- In this chapter we explain how to ensure that maintainability is not compromised during postdelivery maintenance

16.1 Why Postdelivery Maintenance Is Necessary

- Corrective maintenance
  - To correct residual faults
    - » Analysis, design, implementation, documentation, or any other type of faults

Why Postdelivery Maint. Is Necessary (contd)

- Perfective maintenance
  - Client requests changes to improve product effectiveness
    - » Add additional functionality
    - » Make product run faster
    - » Improve maintainability

Why Postdelivery Maint. Is Necessary (contd)

- Adaptive maintenance
  - Responses to changes in the environment in which the product operates
    - » The product is ported to a new compiler, operating system, and/or hardware
    - » A change to the tax code
    - » 9-digit ZIP codes

- At least 67% of the total cost of a product accrues during postdelivery maintenance
- Maintenance is a major income source
- Nevertheless, even today many organizations assign maintenance to
  - Unsupervised beginners, and
  - Less competent programmers

What is Required of Postd. Maint. Prog. (contd)?

- Postdelivery maintenance is one of the most difficult aspects of software production because
  - Postdelivery maintenance incorporates aspects of all other workflows

What is Required of Postd. Maint. Prog. (contd)?

- Suppose a defect report is handed to a maintenance programmer
  - Recall that a “defect” is a generic term for a fault, failure, or error
- What is the cause?
  - Nothing may be wrong
  - The user manual may be wrong, not the code
  - Usually, however, there is a fault in the code

Corrective Maintenance

- What tools does the maintenance programmer have to find the fault?
  - The defect report filed by user
  - The source code
  - And often nothing else

Corrective Maintenance (contd)?

- A maintenance programmer must therefore have superb debugging skills
  - The fault could lie anywhere within the product
  - The original cause of the fault might lie in the by now non-existent specifications or design documents

Corrective Maintenance

- Suppose that the maintenance programmer has located the fault
- Problem:
  - How to fix it without introducing a regression fault

Corrective Maintenance (contd)

- How to minimize regression faults
  - Consult the detailed documentation for the product as a whole
  - Consult the detailed documentation for each individual module
- What usually happens
  - There is no documentation at all, or
  - The documentation is incomplete, or
  - The documentation is faulty

Corrective Maintenance (contd)

- The programmer must deduce from the source code itself all the information needed to avoid introducing a regression fault
- The programmer now changes the source code

The Programmer Now Must

- Test that the modification works correctly
  - Using specially constructed test cases
- Check for regression faults
  - Using stored test data
- Add the specially constructed test cases to the stored test data for future regression testing
- Document all changes

Corrective Maintenance (contd)

- Major skills are required for corrective maintenance
  - Superb diagnostic skills
  - Superb testing skills
  - Superb documentation skills

Adaptive and Perfective Maintenance

- The maintenance programmer must go through the
  - Requirements
  - Specifications
  - Design
  - Implementation and integration

  workflows, using the existing product as a starting point

Adaptive and Perfective Maintenance (contd)

- When programs are developed
  - Specifications are produced by analysis experts
  - Designs are produced by design experts
  - Code is produced by programming experts
- But a maintenance programmer must be expert in all three areas, and also in
  - Testing, and
  - Documentation

Conclusion

- No form of maintenance
  - Is a task for an unsupervised beginner, or
  - Should be done by a less skilled computer professional

The Rewards of Maintenance

- Maintenance is a thankless task in every way
  - Maintainers deal with dissatisfied users
  - If the user were happy, the product would not need maintenance
  - The user's problems are often caused by the individuals who developed the product, not the maintainer
  - The code itself may be badly written
  - Postdelivery maintenance is despised by many software developers
  - Unless good maintenance service is provided, the client will take future development business elsewhere
  - Postdelivery maintenance is the most challenging aspect of software production — and the most thankless

The Rewards of Maintenance (contd)

- How can this situation be changed?
- Managers must assign maintenance to their best programmers, and
- Pay them accordingly

16.3 Postdelivery Maintenance Mini Case Study

- The Temperate Fruit Committee orders software to be developed for exactly 7 temperate fruits
  - Apples, apricots, cherries, nectarines, peaches, pears, and plums
- It is extended to include kiwi fruit, with difficulty
- The product now needs to handle 26 additional fruits
- “Just to the same thing 26 times”

Postdelivery Maintenance Case Study (contd)

- Lessons to be learnt from this
  - The problem was caused by the developer, not the maintainer
  - A maintainer is often responsible for fixing other people's mistakes
  - The client frequently does not understand that postdelivery maintenance can be difficult, or all but impossible
  - This is exacerbated when previous apparently similar perfective and adaptive maintenance tasks have been carried out
  - All software development activities must be performed with an eye on future postdelivery maintenance

16.4 Management of Postdelivery Maintenance

- Various issues regarding management of postdelivery maintenance are now considered

16.4 .1 Defect Reports

- We need a mechanism for changing a product
- If the product appears to function incorrectly, the user files a defect report
  - It must include enough information to enable the maintenance programmer to recreate the problem
- Ideally, every defect should be fixed immediately
  - In practice, an immediate preliminary investigation is the best we can do

Management of Postdelivery Mainten. (contd)

- The maintenance programmer should first consult the defect report file
- It contains
  - All reported defects not yet fixed, and
  - Suggestions for working around them

If the Defect Has Been Previously Reported

- Give the information in the defect report file to the user

If it Is a New Defect

- The maintenance programmer should try to find
  - The cause,
  - A way to fix it, and
  - A way to work around the problem
- The new defect is now filed in the defect report file, together with supporting documentation
  - Listings
  - Designs
  - Manuals

If it Is a New Defect (contd)

- The file should also contain the client's requests for perfective and adaptive maintenance
  - The contents of the file must be prioritized by the client
  - The next modification is the one with the highest priority
- Copies of defect reports must be circulated to all
  - Including: An estimate of when the defect can be fixed
- If the same failure occurs at another site, the user can determine
  - If it is possible to work around the defect, and
  - How long until it can be fixed

Management of Postdelivery Mainten. (contd)

- In an ideal world
  - We fix every defect immediately
  - Then we distribute the new version of the product to all the sites
- In the real world
  - We distribute defect reports to all sites
  - We do not have the staff for instant maintenance
  - It is cheaper to make a number of changes at the same time, particularly if there are multiple sites

16.4.2 Authorizing Changes to the Product

- Corrective maintenance
  - Assign a maintenance programmer to determine the fault and its cause, then repair it
  - Test the fix, test the product as a whole (regression testing)
  - Update the documentation to reflect the changes made
  - Update the prologue comments to reflect
    - » What was changed,
    - » Why it was changed,
    - » By whom, and
    - » When

Authorizing Changes to the Product (contd)

- Adaptive and perfective maintenance
  - As with corrective maintenance, except there is no defect report
  - There is a change in requirements instead

Authorizing Changes to the Product (contd)

- What if the programmer has not tested the fix adequately?
  - Before the product is distributed, it must be tested by the SQA group
- Postdelivery maintenance is extremely hard
- Testing is difficult and time consuming
  - Performed by the SQA group

Authorizing Changes to the Product (contd)

- The technique of baselines and private copies must be followed
- The programmer makes changes to private copies of code artifacts, tests them
- The programmer freezes the previous version, and gives the modified version to SQA to test
- SQA performs tests on the current baseline version of all code artifacts

16.4.3 Ensuring Maintainability

- Maintenance is not a one-time effort
- We must plan for maintenance over the entire life cycle
  - Design workflow — use information-hiding techniques
  - Implementation workflow — select variable names meaningful to future maintenance programmers
  - Documentation must be complete and correct, and reflect the current version of every artifact

Ensuring Maintainability (contd)

- During postdelivery maintenance, maintainability must not be compromised
  - Always be conscious of the inevitable further maintenance
- Principles leading to maintainability are equally applicable to postdelivery maintenance itself

16.4.4 The Problem of Repeated Maintenance

- The moving target problem is frustrating to the development team
- Frequent changes have an adverse effect on the maintainability of the product

The Moving Target Problem

- The problem is exacerbated during postdelivery maintenance
- The more changes there are
  - The more the product deviates from its original design
  - The more difficult further changes become
  - Documentation becomes even less reliable than usual
  - Regression testing files are not up to date
  - A total rewrite may be needed for further maintenance

The Moving Target Problem (contd)

- Apparent solution
  - Freeze the specifications once they have been signed off until delivery of the product
  - After each request for perfective maintenance, freeze the specifications for (say) 3 months or 1 year
- In practice
  - The client can order changes the next day
  - If willing to pay the price, the client can order changes on a daily basis
- “He who pays the piper calls the tune”

Warning

- It is no use implementing changes slowly
- The relevant personnel are replaced
- Nothing can be done if the person calling for repeated change has sufficient clout

16.5 Maintenance of Object-Oriented Software

- The object-oriented paradigm apparently promotes maintenance in four ways
  - The product consists of independent units
  - Encapsulation (conceptual independence)
  - Information hiding (physical independence)
  - Message-passing is the sole communication
- The reality is somewhat different

Maintenance of Object-Oriented Software (contd)

- Three obstacles
  - The complete inheritance hierarchy can be large
  - The consequences of polymorphism and dynamic binding
  - The consequences of inheritance

Size of the Inheritance Hierarchy


```
class UndirectedTreeClass
{
   …
   void displayNode (Node a);
   …
}// class UndirectedTreeClass

class DirectedTreeClass : public UndirectedTreeClass
{
   …
}// class DirectedTreeClass

class RootedTreeClass : public DirectedTreeClass
{
   …
   void displayNode (Node a);
   …
}// class RootedTreeClass

class BinaryTreeClass : public RootedTreeClass
{
   …
}// class BinaryTreeClass

class BalancedBinaryTreeClass : public BinaryTreeClass
{
   Node          hhh;
   displayNode (hhh);
}// class BalancedBinaryTreeClass
```

Figure 16.1

Size of Inheritance Hierarchy (contd)

- To find out what `displayNode` does in **`BalancedBinaryTreeClass`**, we must scan the complete tree
  - The inheritance tree may be spread over the entire product
  - A far cry from “independent units”
- Solution
  - A CASE tool can flatten the inheritance tree

Polymorphism and Dynamic Binding

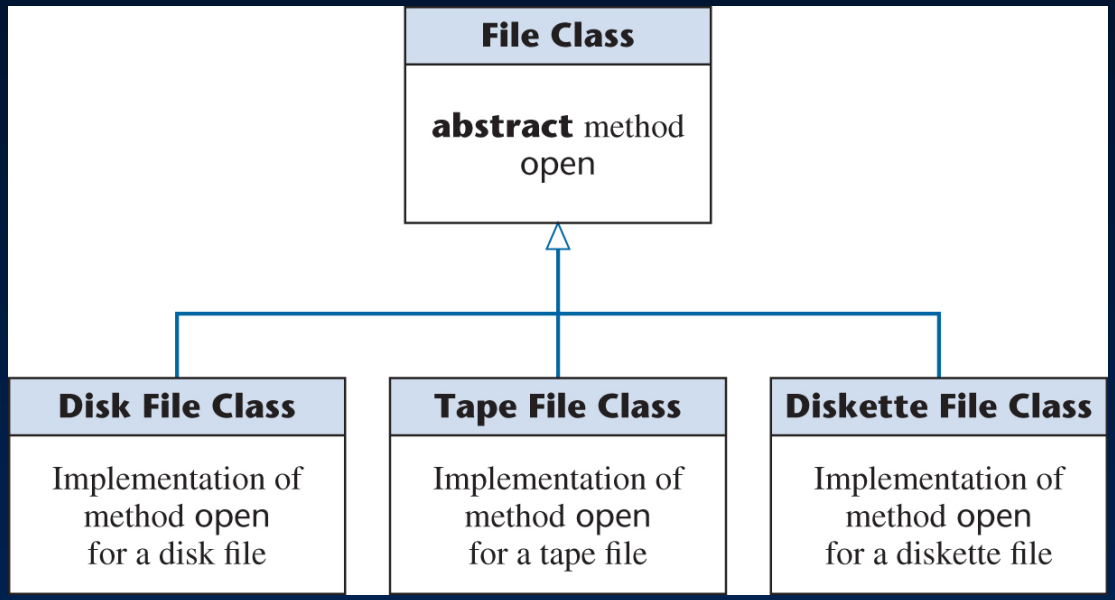

Figure 16.2

- The product fails on the invocation `myFile.open ()`
- Which version of `open` contains the fault?
  - A CASE tool cannot help (static tool)
  - We must trace

Polymorphism and Dynamic Binding (contd)

- Polymorphism and dynamic binding can have
  - A positive effect on development, but
  - A negative effect on maintenance

Consequences of Inheritance

- Create a new subclass via inheritance
- The new subclass
  - Does not affect any superclass, and
  - Does not affect any other subclass
- Modify this new subclass
  - Again, no affect
- Modify a superclass
  - All descendent subclasses are affected
  - “Fragile base class problem”

Consequences of Inheritance (contd)

- Inheritance can have
  - A positive effect on development, but
  - A negative effect on maintenance

16.6 Postdelivery Maintenance versus Development Skills

- The skills needed for maintenance include
  - The ability to determine the cause of failure of a large product
    - » Also needed during integration and product testing
  - The ability to function effectively without adequate documentation
    - » Documentation is rarely complete until delivery
  - Skills in analysis, design, implementation, and testing
    - » All four activities are carried out during development

Postdelivery Maintenance vs. Development Skills (contd)

- The skills needed for postdelivery maintenance are the same as those for the other workflows
- Key Point
  - Maintenance programmers must not merely be skilled in a broad variety of areas, they must be *highly* skilled in *all* those areas
  - Specialization is impossible for the maintenance programmer
- Postdelivery maintenance is the same as development, only more so

16.7 Reverse Engineering

- When the only documentation for postdelivery maintenance is the code itself
  - Start with the code
  - Recreate the design
  - Recreate the specifications (extremely hard)
  - CASE tools can help (flowcharters, other visual aids)

Reverse Engineering (contd)

- Reengineering
  - Reverse engineering, followed by forward engineering
  - Lower to higher to lower levels of abstraction
- Restructuring
  - Improving the product without changing its functionality
  - Examples:
    - » Prettyprinting
    - » Structuring code
    - » Improving maintainability
    - » Restructuring (XP, agile processes)

Reverse Engineering (contd)

- What if we have only the executable code?
  - Treat the product as a black box
  - Deduce the specifications from the behavior of the current product

16.8 Testing during Postdelivery Maintenance

- Maintainers tend to view a product as a set of loosely related components
  - They were not involved in the development of the product
- Regression testing is essential
  - Store test cases and their outcomes, modify as needed

16.9 CASE Tools for Postdelivery Maintenance

- Configuration-control tools are needed
  - Commercial tool
    - » CCC
  - Open-source tools
    - » *cvs*
    - » Subversion
- Reengineering tools
  - Commercial tools
    - » IBM Rational Rose, Together
  - Open-source tool
    - » Doxygen

CASE Tools for Postdelivery Maintenance (contd)

- Defect-tracking tools
  - Commercial tool
    - » IBM Rational ClearQuest
  - Open-source tool
    - » Bugzilla

16.10 Metrics for Postdelivery Maintenance

- The activities of postdelivery maintenance are essentially those of development
  - Metrics for development workflows
- Defect report metrics
  - Defect classifications
  - Defect status

- See Problems 16.16 through 16.21

16.12 Challenges of Postdelivery Maintenance

- The chapter describes numerous challenges
- The hardest challenge to solve
  - Maintenance is harder than development, but
  - Developers tend to look down maintainers, and
  - Are frequently paid more