SE202
Introduction to Software Engineering

Lecture 8-2
Software Maintenance

Today

- Software maintenance
- Maintenance cost
- The four categories of maintenance tasks
- Bug tracking systems
- Task execution

Software maintenance?

Software maintenance is a modification of a software product **after a delivery** in order;

- to correct faults or,
- to improve existing features and adding new ones or,
- to improve performance for product’s environment or maintainability.

Maintenance cost?

- Often very expensive
- 60-80% of a project’s total cost.

Why?

- **Many companies put the most focus on fast development with improper date estimation**
- **Software by its nature is constantly evolving**
    - From customers asking for new and modified features
    - Form a change of system in which the software runs
- **Changing code might end up add more bugs than remove them**

The four categories of maintenance tasks

Improving existing features and adding new ones

Perfective

Adaptive

Modifying the application to meet changes in the application’s environment

Fixing bugs

Corrective

Preventive

Restructuring the code to make it more maintainable

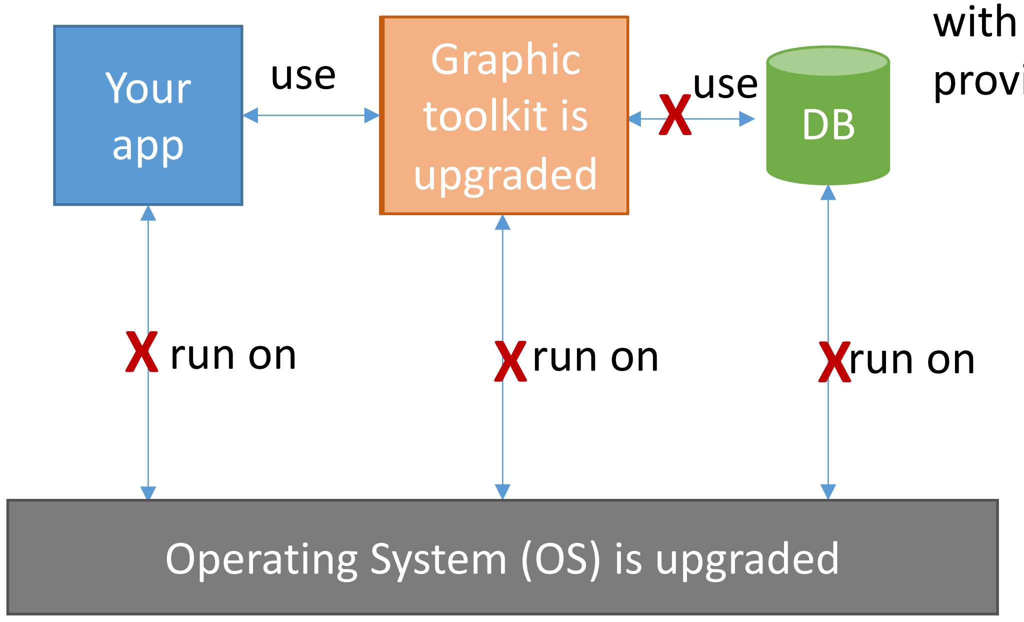
Perfective tasks

- Feature Improvements
  - involve modifying existing code
  - carefully study the existing code so that you’re sure you understand what it does and how it works.
  - E.g. Change the main program to restore the settings in use when the previous session ended.
- New Features
  - Adding new features to an application is a lot like writing code for the initial application
  - Steps: Make TOR (Term of Reference) of new features (requirements) → Sign off → high and low level design → coding → test and test → deployment

Adaptive tasks

- Adaptive tasks help keep the application usable when the things around it change:
  - hardware,
  - operating system (OS),
  - database,
  - other tools (such as spreadsheets or reporting tools),
  - network security
- If pieces of the users’ environment change, it could break your application. → so you have to fix it.

Example


Your app
use
Graphic toolkit is upgraded
X
use
DB
X run on
X run on
X run on
Operating System (OS) is upgraded

No upgrade version of DB that is compatible with the new OS is provided by vendor.

Corrective Tasks

- Corrective tasks are simply bug fixes
  - Bugs are mistakes in the specification, designs, documentation, and other pieces of the program
  - E.g. fix the bug that when the user tried to close the application, it crashes.
- Make sure that you understand the found bug → fix the bug → test → release a new version of the application with the bug fixed.

Preventive Tasks

- Preventive tasks involve restructuring (refactoring) the code to make it easier to debug and maintain in the future.
- E.g. Rewrite a 715-line method to make it smaller.
- Reasons for refactoring code
  - Clarification- If a piece of code is confusing, you should add comments to it explaining how it works.
  - Code Reuse - extract the common code into a new class or method
  - Improved Flexibility – for the future requests/changes
  - Bugs invasion – from e.g. the bad design of the code, several time of code modification
  - Bad Programming Practices – E.g. code is too long and/or duplicated

For a typical large application, Relative effort spent on each of the categories

- Perfective —50 percent
- Adaptive —25 percent
- Corrective —20 percent
- Preventive —5 percent

Bug tracking systems

Status	Meaning
New	The bug has just arrived and has not yet been assigned to anyone.
Assigned	The bug has been assigned to someone to fix.
Reproduced	The bug has been reproduced by a team member
Cannot reproduce	A team member has examined the program and can't make the bug occur.
Pending	A request for more information has been sent to the customer who reported the bug.

Status	Meaning
Fixed	The bug has been fixed but not tested yet.
Tested	The fix has been thoroughly tested and the bug is verified as gone.
Deferred	The bug should not be fixed, or at least not yet.
Closed	The bug has been either fixed, deferred, or otherwise abandoned
Reopened	The bug reappeared after being closed. The bug should probably be treated as if it were a new bug.

TASK EXECUTION

- To perform maintenance tasks successfully, you need to follow the normal software engineering steps:
    - requirement gathering, high-level design, low-level design, development, testing, and deployment.
    - Although you can often skip some of those steps (i.e. high-level design to fix a one-line bug.)
- However, you can reduce maintenance costs by doing a good job when you write the initial code.
    - For example, develop simple but flexible designs, use good programming practices, insert comments to make the code easy to read, and provide documentation so future generations of maintenance programmers can figure out what you were thinking when you wrote the code.

Question?

1. Suppose your programming team writes an application with 10,000 lines of code. During testing, you decide that the team generates roughly 20 bugs per KLOC (kilo-lines of code) for new code (2%). During bug fixing, you discover they generate about twice that many bugs when they modify older code (4%). How many lines of code will the team actually generate including original code, fixes, fixes to fixes, and so forth?
2. After you write the lines of code you predicted in your answer to Exercise 1, are you done with maintenance?