Introduction to Software Maintenance

There is no such thing as a 'finished' computer program

- Lehman

Software Maintenance

The process of modifying a software system or its components after delivery to correct faults, improve performances or other attributes, or adapt to a changed environment

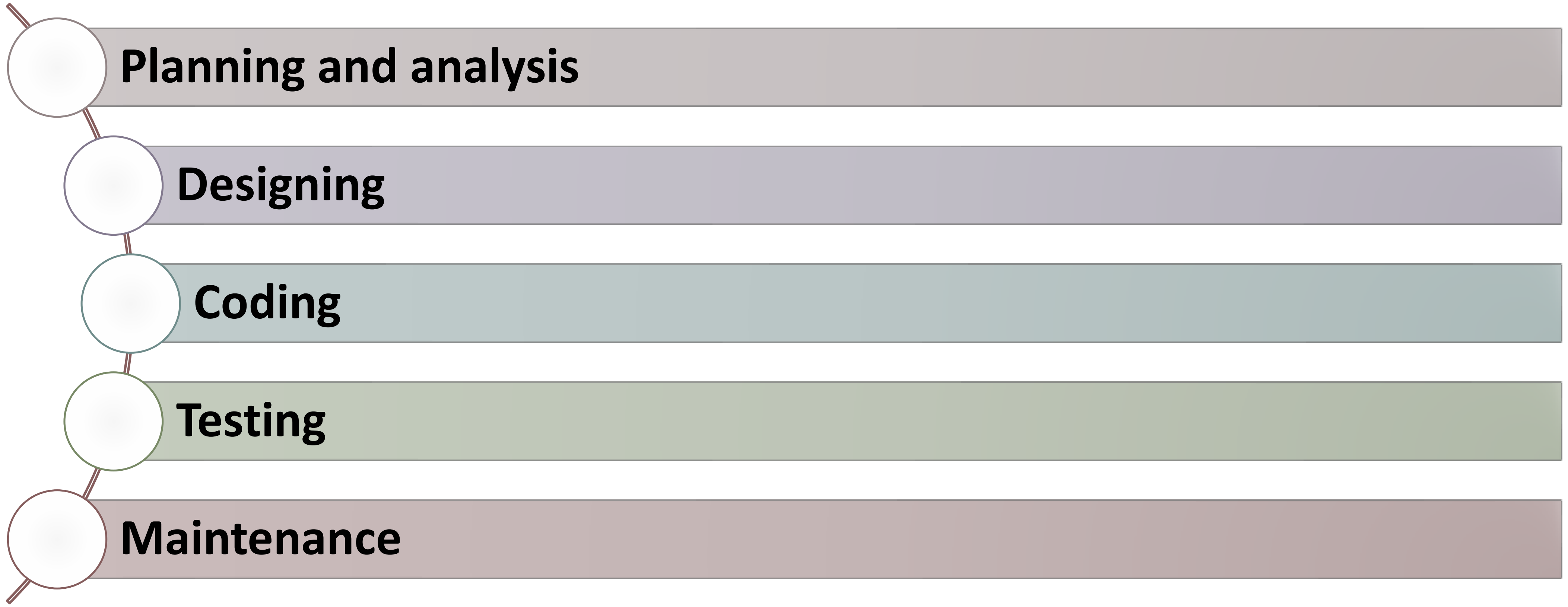
Importance of Software Maintenance

- **To provide continuity of service:** Maintenance activities help to keep a system operational by fixing bug, recovering from failure, and accommodating changes in the operating system and hardware.
- **To support mandatory upgrades:** This type of change would be necessary because of such things as amendments to government Regulations. Besides, the need to maintain a competitive edge over rival products will trigger this kind of change.

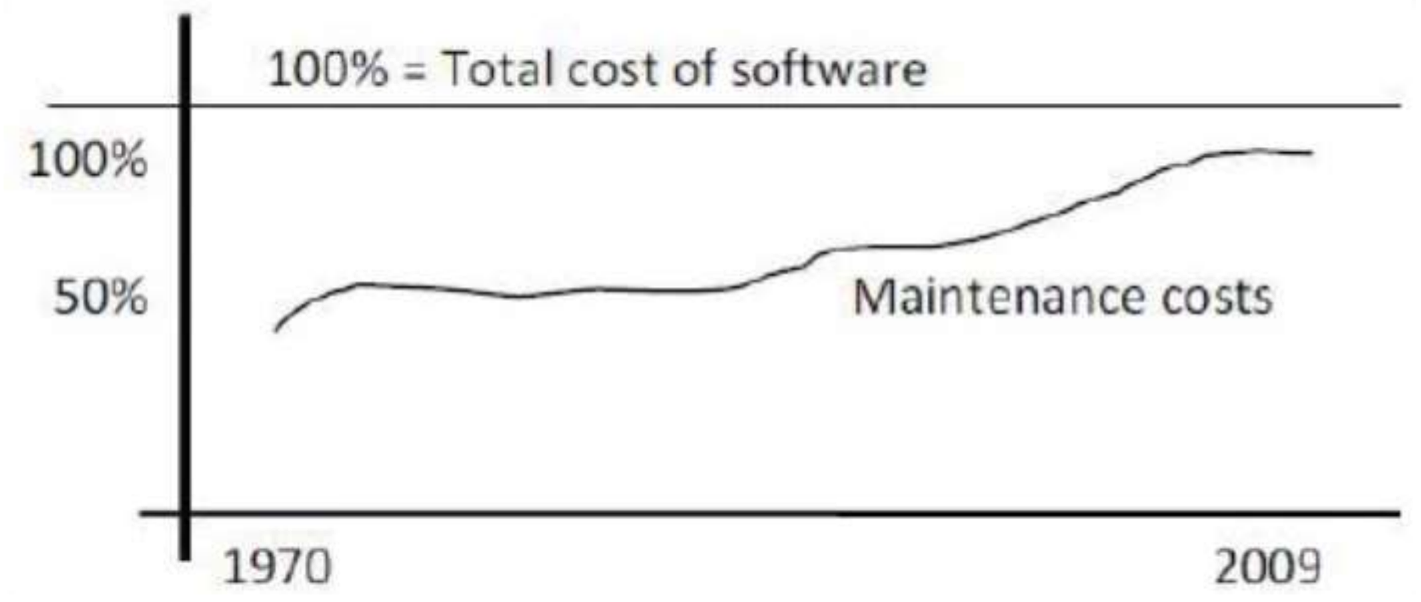
Importance of Software Maintenance

- **To support user requests for improvements:** On the whole, the better a system is, the more it will be used and the more the users will request enhancements in functionality. There may also be requirements for better performance and customization to local working patterns.
- **To facilitate future maintenance work:** Shortcuts at the software development stage are very costly in the long run. It is often financially and commercially justifiable to initiate change solely to make future maintenance easier. This would involve such things as code and database restructuring and updating of documentation.

Software Development Life Cycle



Software Maintenance Cost



Types of Software Maintenance

- Based on the Standard for Software Engineering–Software Maintenance, ISO/IEC 14764, there are four categories of maintenance activities:

1. Corrective
2. Adaptive
3. Perfective
4. Preventive

Corrective Maintenance

- Modification initiated by defects in the software
- A reactive process (performed after detecting defects with the system)
- Example: fixing a program that produces wrong output

Adaptive Maintenance

- Driven by the need to accommodate modifications in the environment of a software
- Example: changing a system to support new hardware configuration

Perfective Maintenance

- Changes undertaken to expand the existing requirements of a system to make a variety of improvements, namely, user experience, processing efficiency
- Example: Making a program output more readable for better user experience

Preventive Maintenance

- Prevent problems from occurring by modifying software products
- Example: restructuring a program to achieve good styles to make later program comprehension easier

Practice Examples

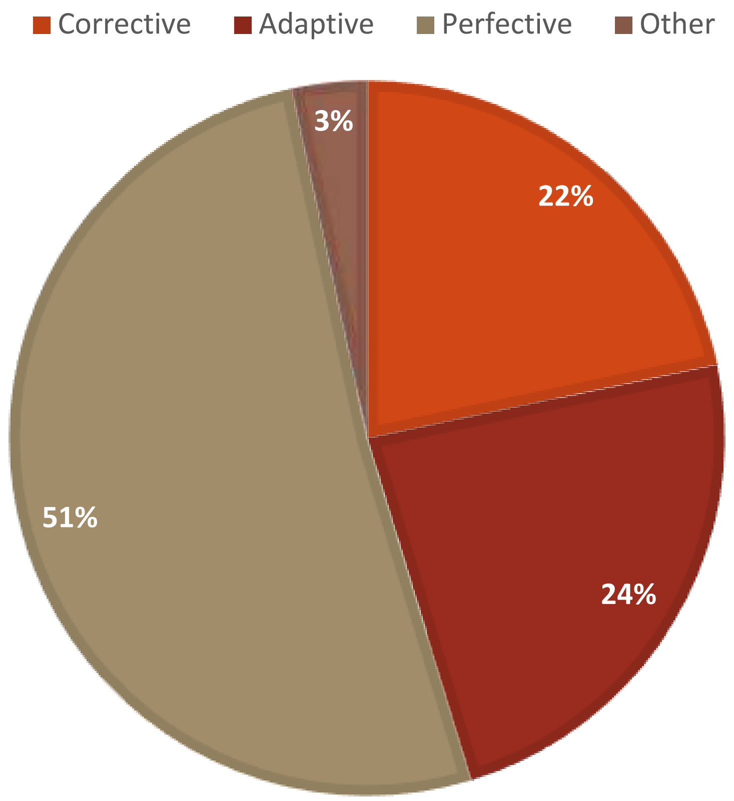
- Changing a system to make it compatible with other applications.
- Cleaning up the system internal state to prevent the occurrence of crash in the future.
- Modifying a software to make it faster.

Maintenance Effort

- A study by Lientz and Swanso
- Sample: 487 Organizations
- Published in 1980

Maintenance Effort

- **Corrective Maintenance: 21.7%**
  - Emergency fixes: 12.4%
  - Routine debugging: 9.3%
- **Adaptive Maintenance: 23.6%**
  - Accommodating changes to data inputs and files: 17.4%
  - Accommodating changes to hardware and system software: 6.2%
- **Perfective Maintenance: 51.3%**
  - Customer enhancements: 41.8%
  - Improvements to documentation: 5.5%
  - Optimization: 4.0%
- **Other: 3.4%**



Software Maintenance Challenges

- **Intellectual activity:** One needs the creativity to identify the code to be changed and to make an appropriate change without destroying the integrity of the software system as a whole.
- **Maintainability of software:** Many computer programs were written without having maintainability in mind. This is mostly because of two contributing factors. First, most of the software is developed under very tight budget and time constraints and there is virtually no time to ponder upon the maintainability issue. Second, programmers are quite often not the users of software products and it is difficult to anticipate what course the future will take.

Software Maintenance Challenges

- **Lack of documentation:** The lack of time during software development leads to producing poor documentation, especially when the software is behind schedule. Poor documentation means that others have to rely on the programmers.
- **Mobility of software personnel:** The computer industry is a fast growing industry. The demand for software specialists far exceeds the supply. To gain quick promotion or better financial reward, software specialists frequently change jobs. When a software specialist quits, the corporation loses the expertise gained over a long period of time. The long-term effect of high turnover is that the software will gradually degrade to a state whereby its maintenance is too costly or impossible.

Software Maintenance Challenges

- **Management attitude:** Management often regards software maintenance as a second-rate job. Consequently, the financial reward paid to the software maintenance personnel lower than that paid to the software development personnel. Besides, ill-qualified people are assigned to maintain the software. Hence, competent programmers have chosen to avoid software-maintenance jobs.
- **The complexity of understanding computer programs:** In reading computer programs, one has to understand the code and relate it to the operating environment to make sense out of it. Thus, in maintaining software, one has to suffer from the learning curve effect.