جامعة الملك سعود
King Saud University
College of Computer and Information Sciences

Software Evolution

Section 9.1+9.3+9.4 (Somerville)

Fall Semester 2021

1st Semester 1443 H

Topics covered

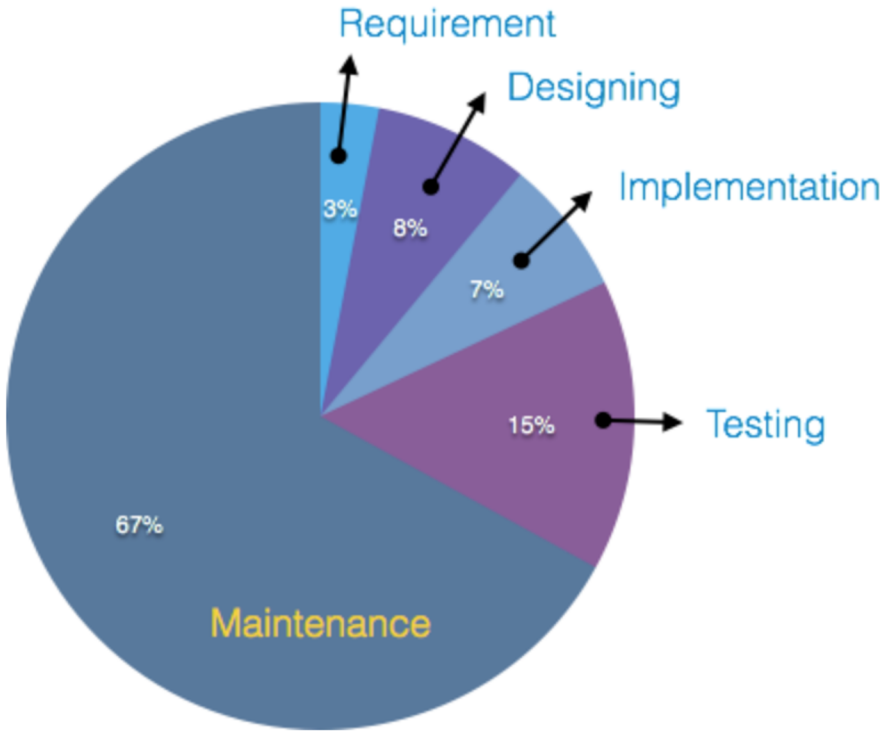
- Evolution processes.
- Legacy systems.
- Software maintenance.

Software change

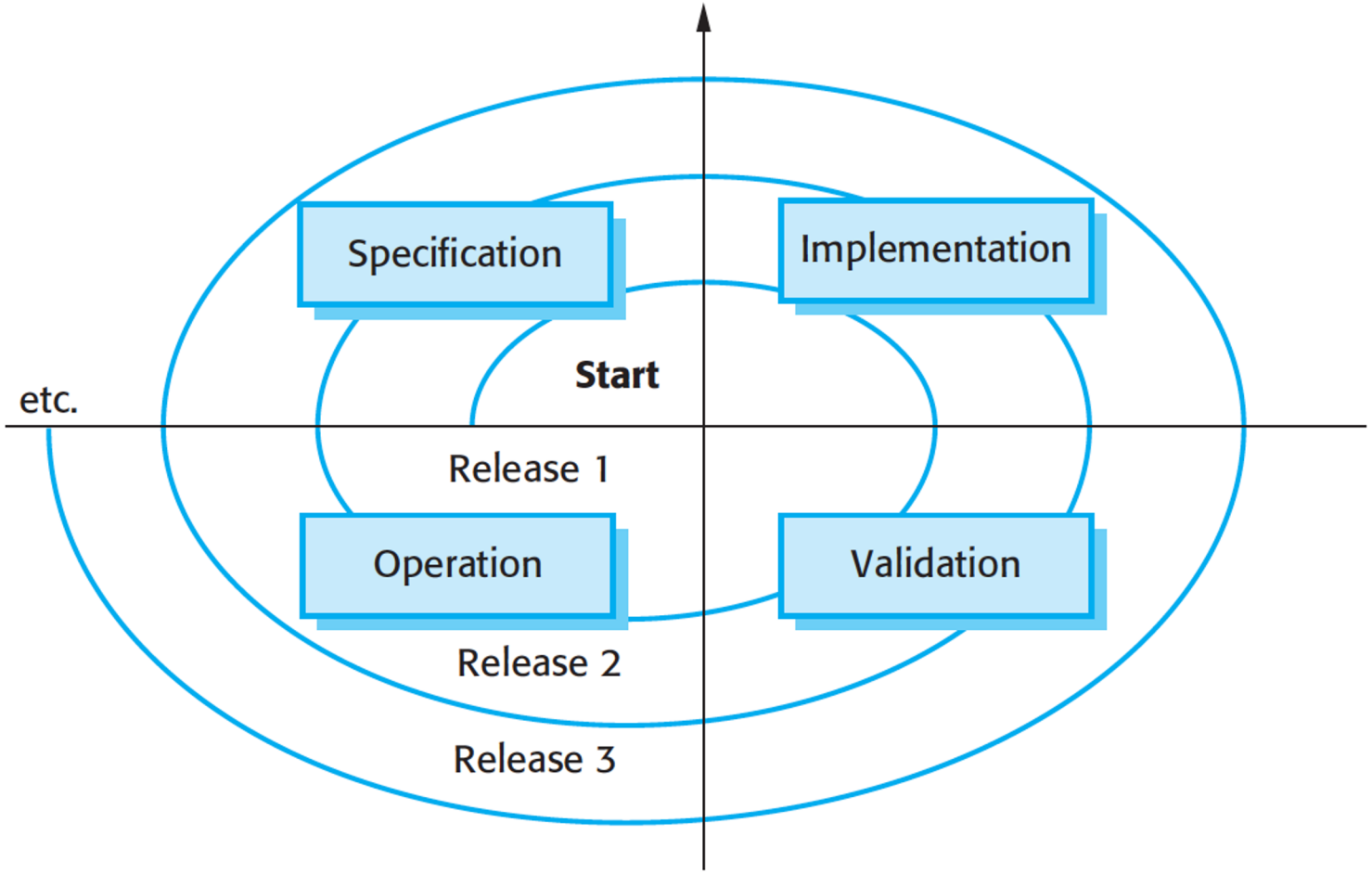
- **Software change is inevitable.**
  - New requirements emerge when the software is used.
  - The business environment changes.
  - Errors must be repaired.
  - New computers and equipment is added to the system.
  - The performance or reliability of the system may have to be improved.
- A **key problem** for all organizations is **implementing and managing change** to their existing software systems.

Importance of evolution

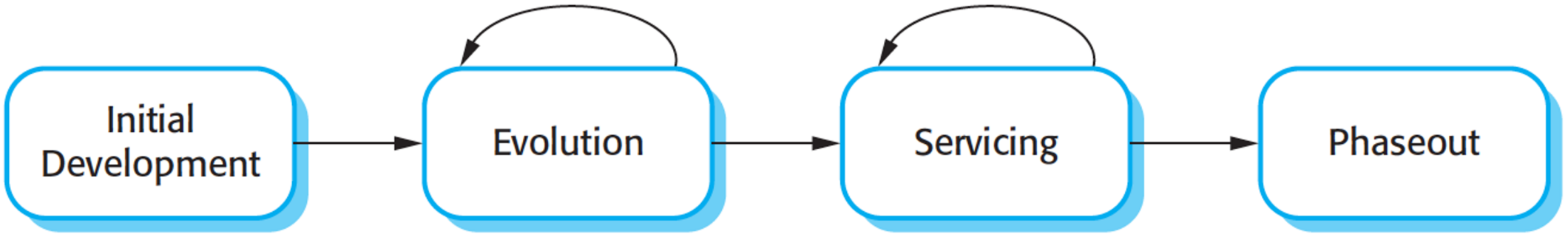
- Organizations have huge investments in their **software systems** - they are **critical business assets**.
- To maintain the value of these assets to the business, they **must be changed and updated**.
- The majority of the software budget in large companies is **devoted to changing and evolving existing software rather than developing new software**.



A spiral model of development and evolution



Evolution and servicing



Evolution and servicing

- **Evolution:** The stage in a software system's life cycle where it is **in operational use and is evolving** as new requirements are proposed and implemented in the system.

- **Servicing:** At this stage, the software remains useful, but the **only changes made are those required to keep it operational** i.e. bug fixes and changes to reflect changes in the software's environment. **No new functionality is added**.

- **Phase-out:** The software may still be used but **no further changes** are made to it.

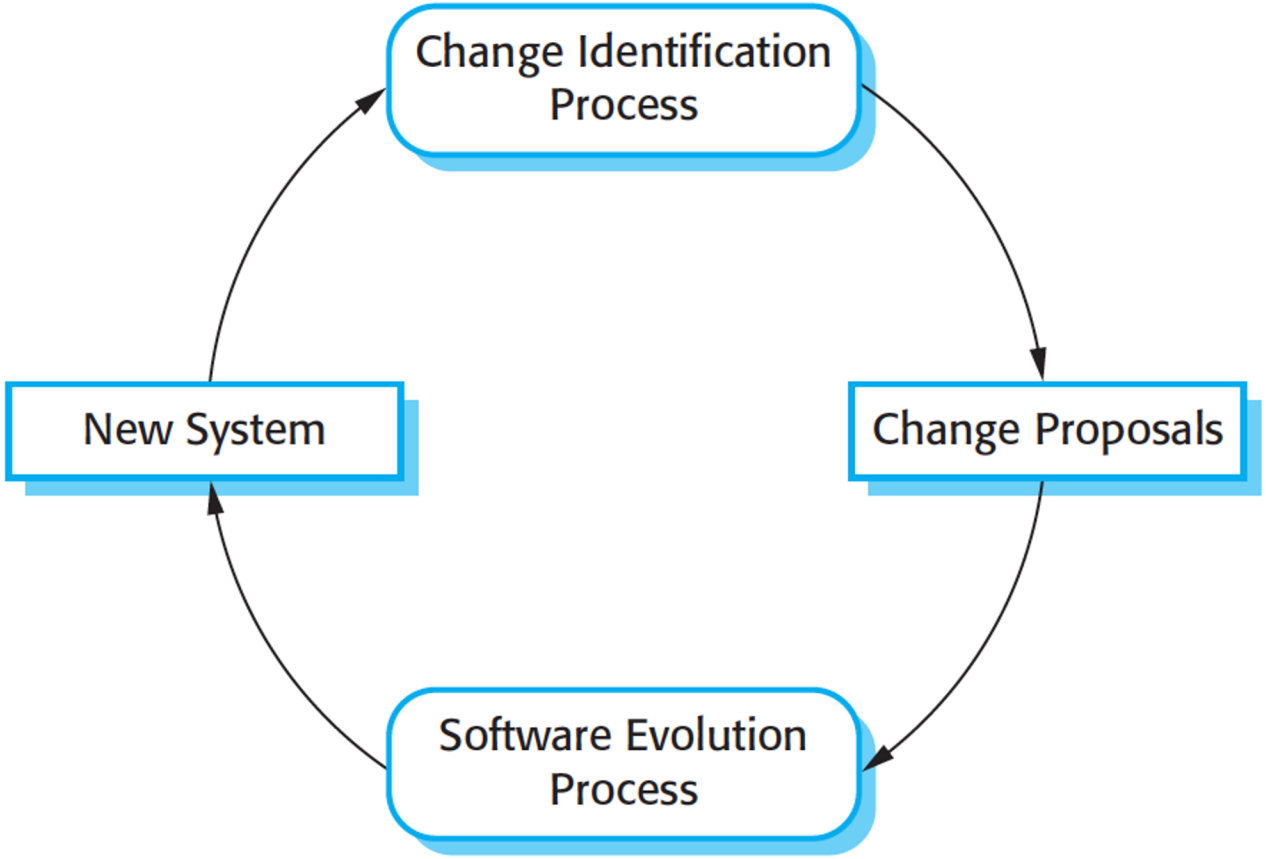
Evolution processes

- Software evolution processes depend on:
  - The **type of software** being maintained.
  - The **development processes** used.
  - The **skills and experience** of the people involved.

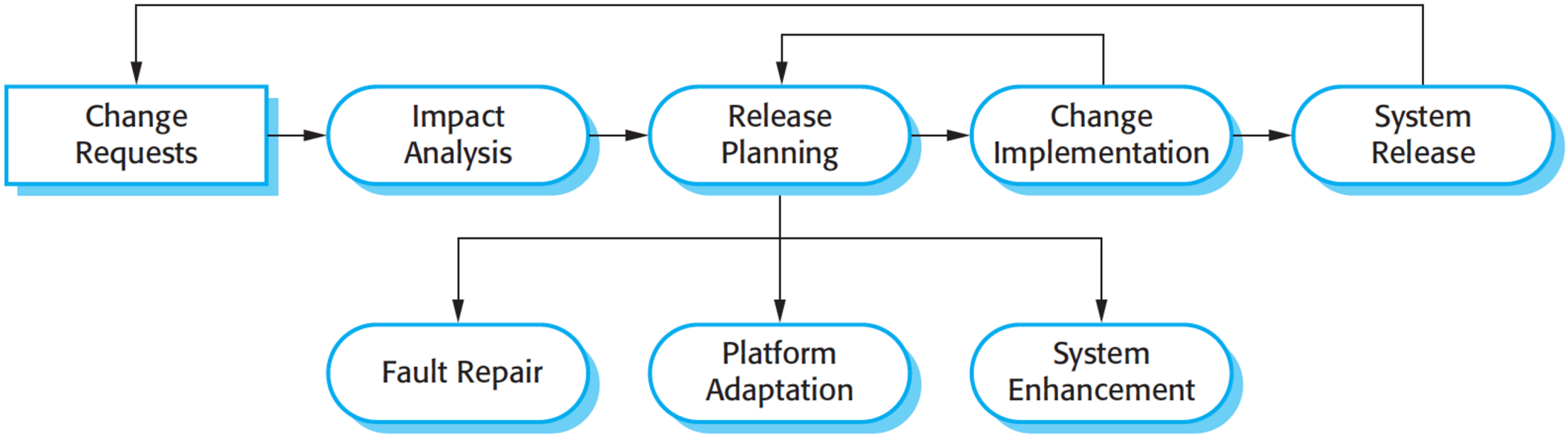
- **Proposals for change are the driver for system evolution**.
  - Should be linked with components that are affected by the change, thus allowing the cost and impact of the change to be estimated.

Change identification and evolution continues throughout the system lifetime.

Change identification and evolution processes



The software evolution process



Agile methods and evolution

- Agile methods are based on incremental development so the transition from development to evolution is a seamless one.
  - Evolution is simply a continuation of the development process based on frequent system releases.

- Automated **regression testing** is particularly valuable when changes are made to a system.

- Changes may be expressed as additional user stories.

Legacy systems

- Legacy systems are **older systems that rely on languages and technology that are no longer used** for new systems development.

- Legacy software may be dependent on older hardware, such as mainframe computers and may have associated legacy processes and procedures.

- Legacy systems are not just software systems but are broader socio-technical systems that include hardware, software, libraries and other supporting software and business processes.

Legacy systems – *change or replace?*

- Legacy system **replacement is risky** and expensive so businesses continue to use these systems. It is risky because of
  - Lack of complete system specification.
  - Tight integration of system and business processes.
  - Undocumented business rules (embedded in the legacy system).
  - New software development may be late and/or over budget.

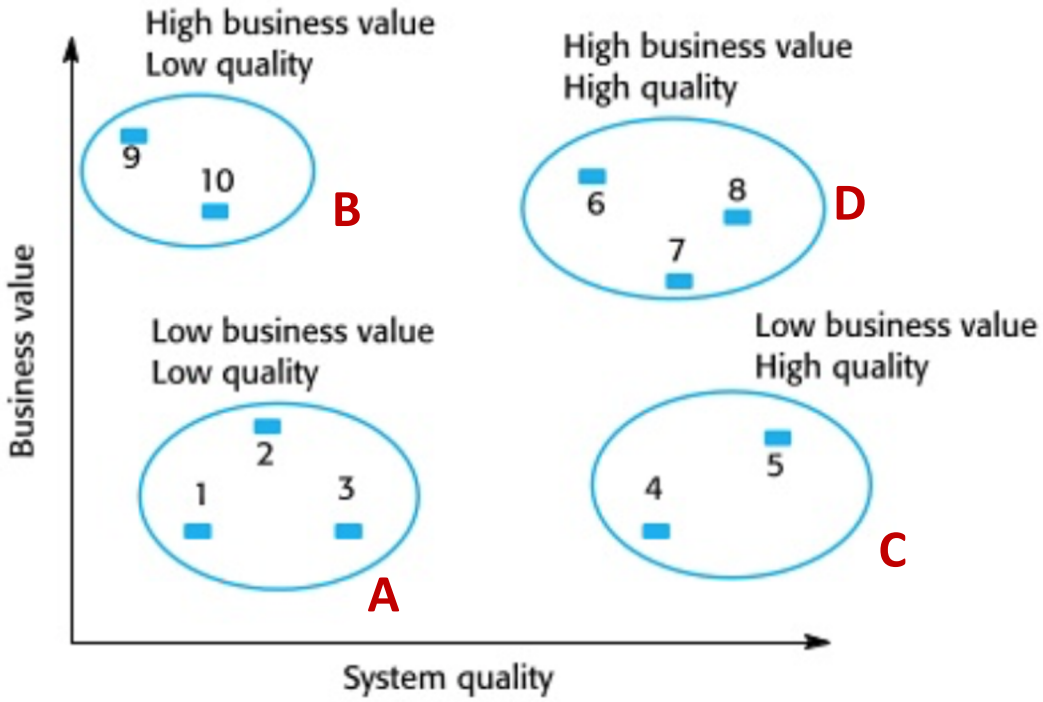
- Legacy systems are **expensive to change** because
  - No consistent programming style.
  - Use of obsolete programming languages with few people available with these language skills.
  - Inadequate system documentation.
  - System structure degradation.
  - Program optimizations may make them hard to understand.
  - Data errors, duplication and inconsistency.

Legacy system management

College of Computer and Information Sciences

- Organizations that rely on legacy systems must choose a strategy for evolving these systems.
  - A. **Scrap** the system completely.
  - B. **Replace** the system or **transform** it by **re-engineering** to improve its maintainability.
  - C. **Replace** with COTS, **scrap** or **maintain.**
  - D. **Continue** in operation using normal **maintenance**.

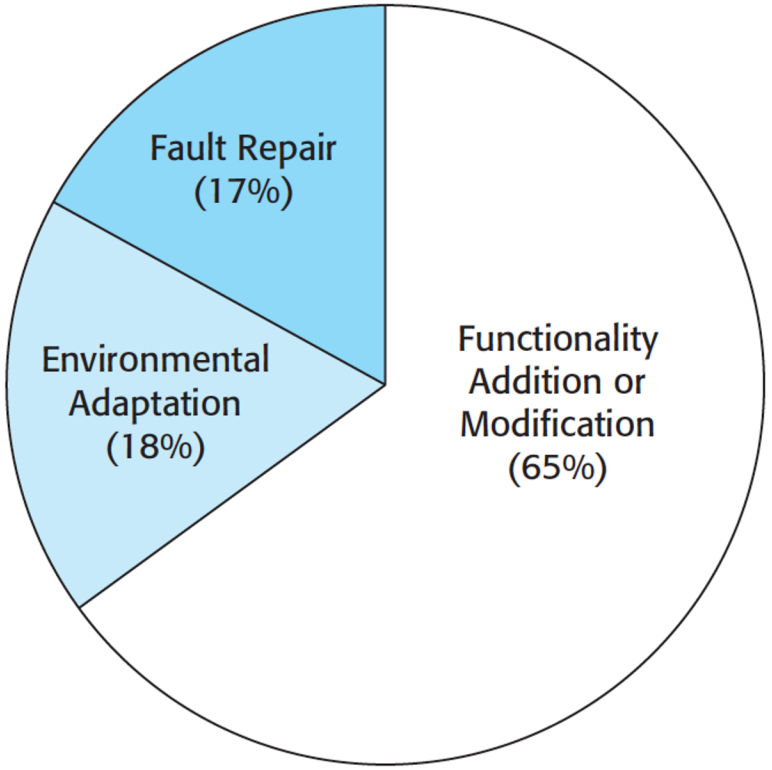
- **The strategy chosen should depend on the system quality and its business value.**



An example of a legacy system assessment

Software Maintenance

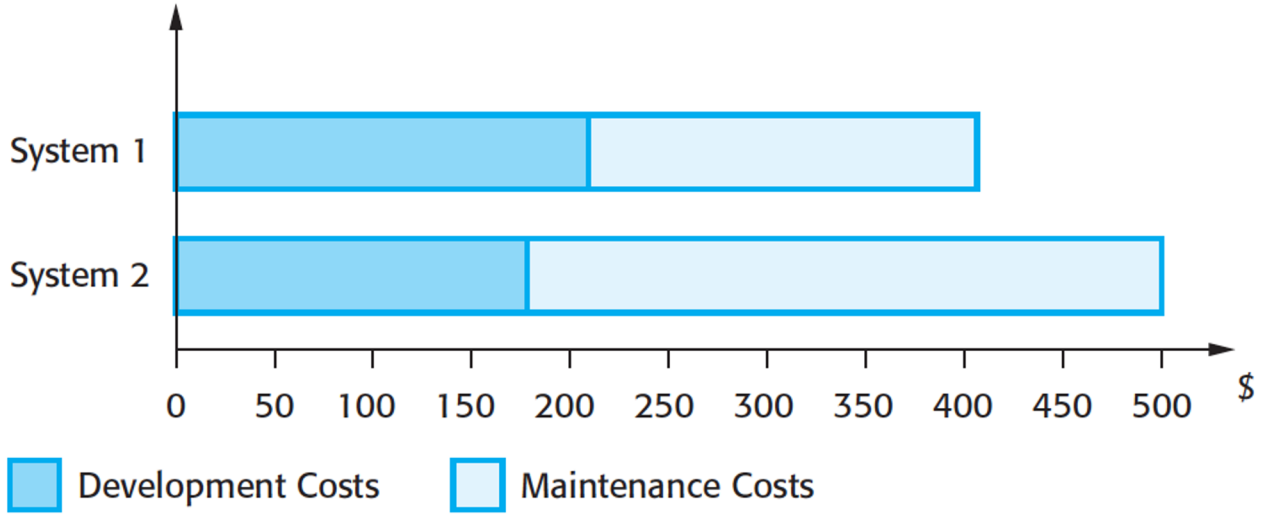
- **Modifying a program after it has been put into use.**
- Maintenance **does not normally involve major changes** to the system's architecture.
- Changes are implemented by **modifying existing components** and adding new components to the system.
- **Three types of maintenance:**
  - Fault repair.
  - Environmental adaptation.
  - Functionality addition.



Maintenance effort distribution

Software Maintenance

- The overall lifetime costs may decrease as more effort is expended during system development to produce a maintainable system.
  - Because of the potential reduction in costs of understanding, analysis, and testing, there is a significant multiplier effect when the system is developed for maintainability.



Development and maintenance costs

Thank you