Software Engineering (CSM 31212)

Lecture 11:Software Maintenance

MM. Mohamed Mufassirin

Lecturer in Computer Science

Dept. of Computer Science

FAS/SEUSL

Reasons for changes

- Errors in the existing system
- Changes in requirements
- Technological advances
- Legislation and other changes

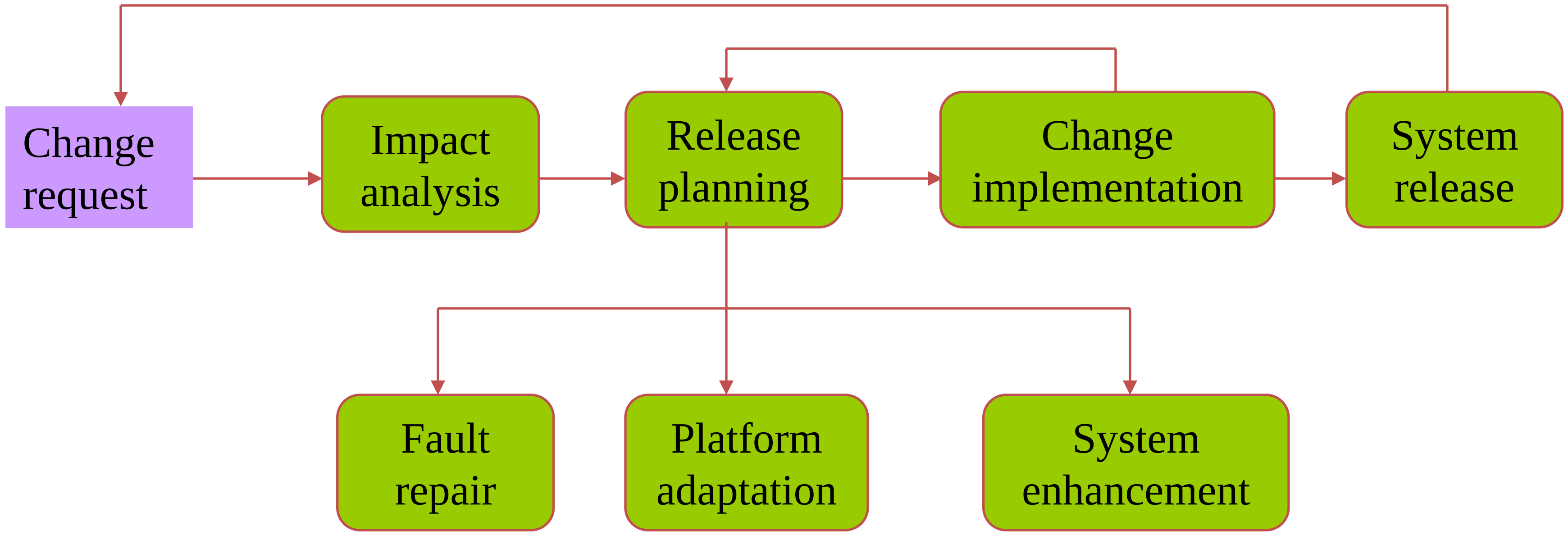
Reasons for High Maintenance Cost

- ***Program age and structure -*** The design and the programming practices used in developing old systems may not be flexible, hence modifying existing systems may be difficult.
- ***Team Stability -*** After a system has been delivered, it is normal for the development team to be broken up and people work on new projects. The new team responsible for the maintenance of the system may not understand the design decisions and hence lot of effort during the maintenance process is taken up with understanding the existing system.
- ***Contractual Responsibility –*** The maintenance contract may be given to a difference company rather than the original system developer. This factor along with the lack of team stability, means that there is no motivation for a development team to write the software so that it is not easy to change.
- ***Staff Skills -*** Maintenance is seen as a less skilled process than system development and is often allocated to junior staff members. Programming languages and techniques used in old systems may be outdated, hence new staff members may not like to learn these languages and techniques.

Types of Maintenance

- **Corrective Maintenance**
  - to repair software faults(fixing bugs in the code)
- **Adaptive Maintenance (adapting the software to new environments)**
  - Maintenance to add to or modify the system's functionality
  - Modifying the system to satisfy new requirements
  - Modifying the system to suit new operation environment
- **Perfective Maintenance**
  - Improving programs performance, structure, reliability etc. Making changes to avoid future problems or prepare for future changes

An overview of the maintenance process

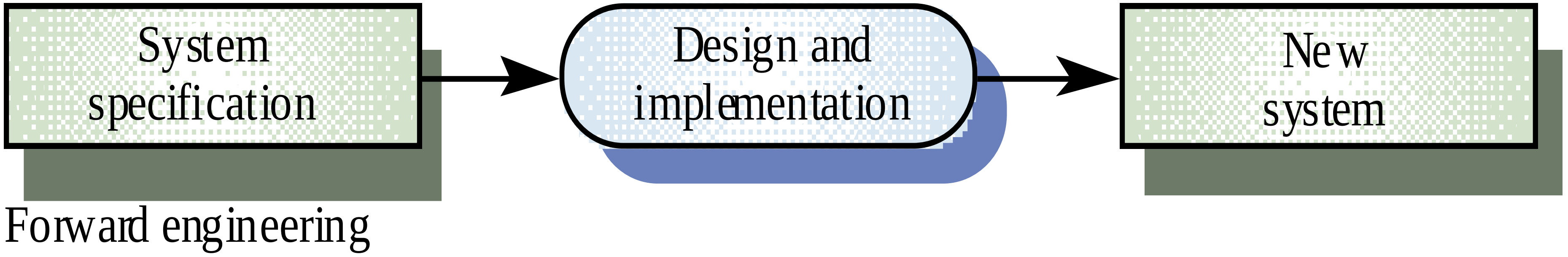


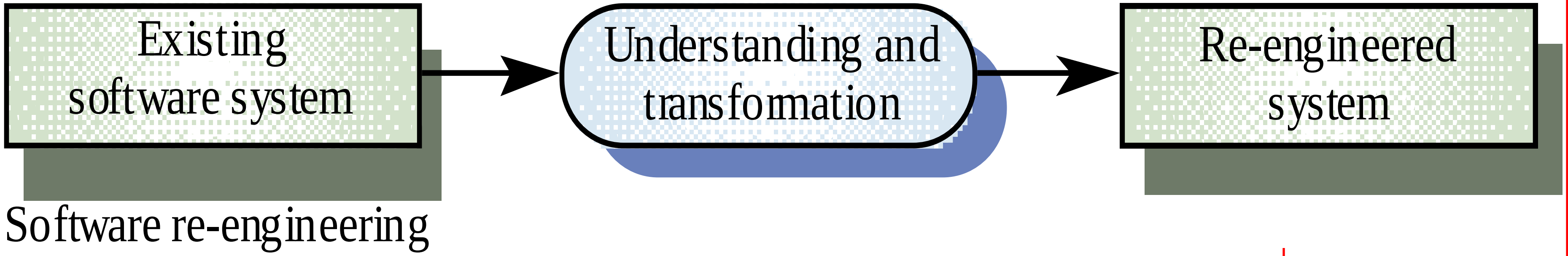
Software Re-engineering

- **What? -** Re-structuring or re-writing part or all of an existing system without changing its functionality
- **When?**
  - When some but not all sub-systems of a larger system require frequent maintenance
  - When hardware or software support becomes obsolete(out of date)
- **How?**
  - The system may be re-structured and re-documented to make it easier to maintain
- **Why?**
  - Reduced risk
    - New software development carries high risk.
  - Reduced cost

The cost of re-engineering is often significantly less than the costs of developing new software

Forward Engineering and Re-engineering





Automated program conversion
Automated program restructuring
Manual Program and Data Restructuring
Restructuring plus Architectural Changes

$\Big\downarrow$ cost

Re-engineering activities

Source code translation- The program is converted from an old programming language to a more modern version of the same language or to different language.

Reverse engineering - The program is analysed and information extracted from it which help to document its organisation and functionality.

Program structure improvement – The control structure of the program is analysed and modified to make it easier to read and understand.

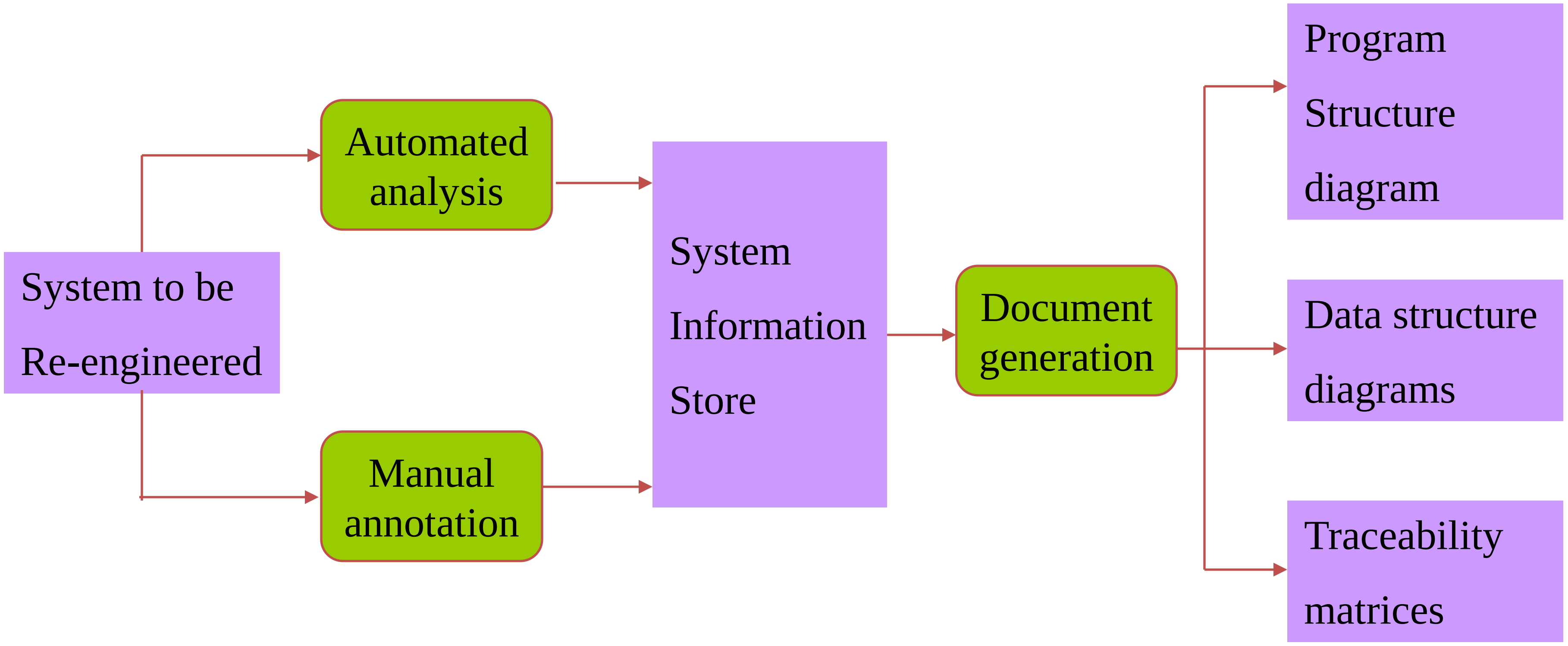
Program modularisation – Related parts of the program are grouped together and, where appropriate, redundancy is removed. In some cases this stage may involve architectural transformation.

Data re-engineering – The data processed by the program is changed to reflect program changes.

Reverse Engineering

- Analysing software to gain an understanding of its design and specification
- May be part of a re-engineering process **or** to re-specify a system for re-implementation
- Builds a program data base and generates information from this
- Program understanding tools (browsers, CASE tools, cross-reference generators, etc.) may be used in this process
- Why?
  - Original code may have been written under limitations which no longer apply e.g. memory needs, performance etc
  - Maintenance tends to corrupt the structure of a program. It becomes harder and harder to understand
  - The program may be automatically restructured to remove unconditional branches or complex conditional statements

The reverse engineering process



Configuration Management(CM)

- CM is the development and application of standards and procedures for managing an evolving software product.
- You need to manage evolving systems because, as they evolve, many different versions of the software are created.
- These versions incorporate proposals for change, corrections of faults and adaptations for different hardware and operating systems.
- There may be several versions under development and in use at the same time. You need to keep track of these changes that have been implemented and how these changes have been included in the software.

Configuration Management

- All products of the software process may have to be managed
  - Specifications
  - Designs
  - Programs
  - Test data
  - User manuals
- Thousands of separate documents are generated for a large software system
- CM Plan

  A CM plan described the standards and procedures which should be used for configuration management. The starting point for developing the plan should be a set of general, company-wide CM standards and these should be adapted as necessary for each specific project

Versions/variants/releases

- ***Version*** - An instance of a system which is functionally distinct in some way from other system instances
- ***Variant*** - An instance of a system which is functionally identical but non-functionally distinct from other instances of a system
- ***Release*** - An instance of a system which is distributed to users outside of the development team
- Version Numbering
  - Simple naming scheme uses a linear derivation e.g. V1, V1.1, V1.2, V2.1, V2.2 etc.