



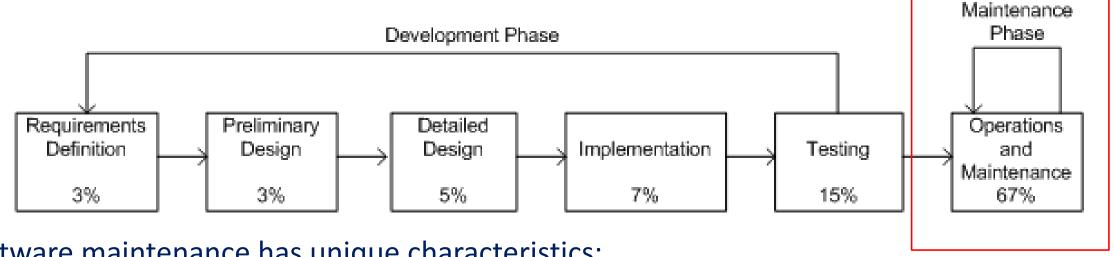
Dr. Lamia Abo Zaid

د. لمياء أبوزيد

Software Evolution: TOC

- 1. Introduction to Software Evolution
- 2. Taxonomy of Software Maintenance and Evolution
- Evolution and Maintenance Models
- 4. Reuse and Domain Engineering
- 5. Program Comprehension
- 6. Impact Analysis
- 7. Refactoring
- 8. Reengineering
- 9. Legacy Information Systems

Maintenance as part of the SDLC



Software maintenance has unique characteristics:

- Constraints of an existing system
- ■Shorter time frame
- ■Available test data

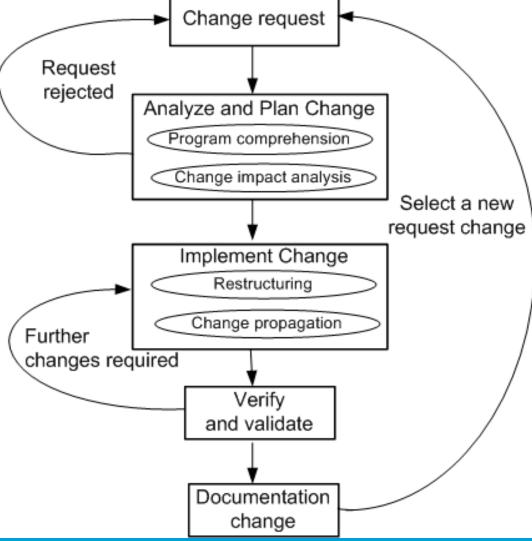
Software Maintenance Life Cycle (SMLC)

Why Do We Need Models to Manage & Execute Software Maintenance Activities?

- ☐ Change, Evolution, and system configuration complicate maintenance activities.
- The software product released to a customer is in the form of executable code, whereas the corresponding "product" within the supplier organization is source code. Thus, strict control must be kept, otherwise exact source code representation of a particular executable version may not exist.
- ☐ Three maintenance models will be explained:
 - 1. Reuse → old
 - 2. Simple Staged → relatively new
 - 3. Change Mini-cycle → still in research

Change Mini-Cycle Model

Introduce new requirements or alter the software system if requirements are not correctly implemented. Yau et al (1978) introduced the change mini-cycle evolutionary model to capture this and revisited by other researchers, namely, Bennet et al. (2000) and Mens (2008).



Change Mini-Cycle Model

mini-cycle model consists of five major phases:

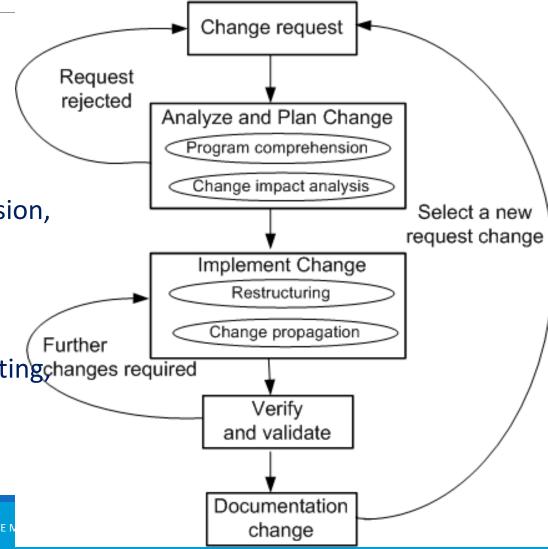
1. Change Request (CR): defect report or enhancement request

Analyze and plan change: Program comprehension, Impact analysis and ripple effect

3. Implement change: change propagation

4. Verify and validate: code review, regression testing thanges required and execution of new tests

5. Documentation change.



Software Maintenance Standard

- ☐ IEEE and ISO have both addressed s/w maintenance processes.
- □ IEEE/EIA 1219 considers maintenance as a fundamental life cycle process, it organizes the maintenance process in seven activities:
 - problem identification, analysis, design, implementation, system test, acceptance test and delivery.
- □ ISO/IEC 14764 describes s/w maintenance as an iterative process for managing and executing software maintenance activities, which are :
 - process implementation, problem and modification analysis, modification implementation, maintenance review/acceptance, migration and retirement.

The standard focuses on a seven-phases

Each of the seven activities has five associated attributes as follows:

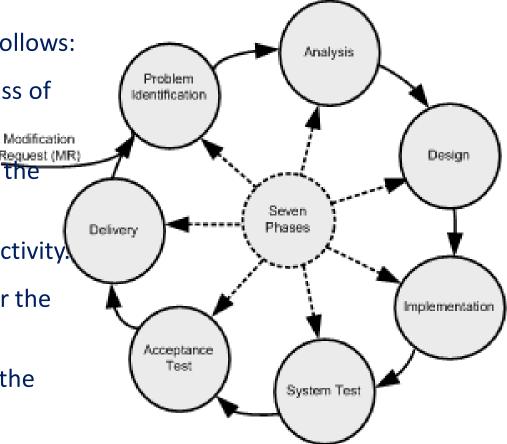
Activity definition: This refers to the implementation process of the activity.

□Input: This refers to the items that are required as input to the activity.

Output: This refers to the items that are produced by the activity.

□ Control: This refers to those items that provide control over the activity.

■ Metrics: This refers to the items that are measured during the execution of the activity.



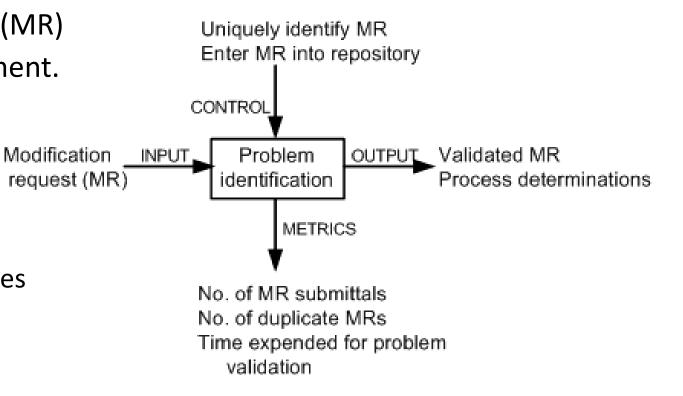
•The change request (CR) is submitted in the form of a modification request (MR) for a correction or for an enhancement. MR & CR are used interchangeably.

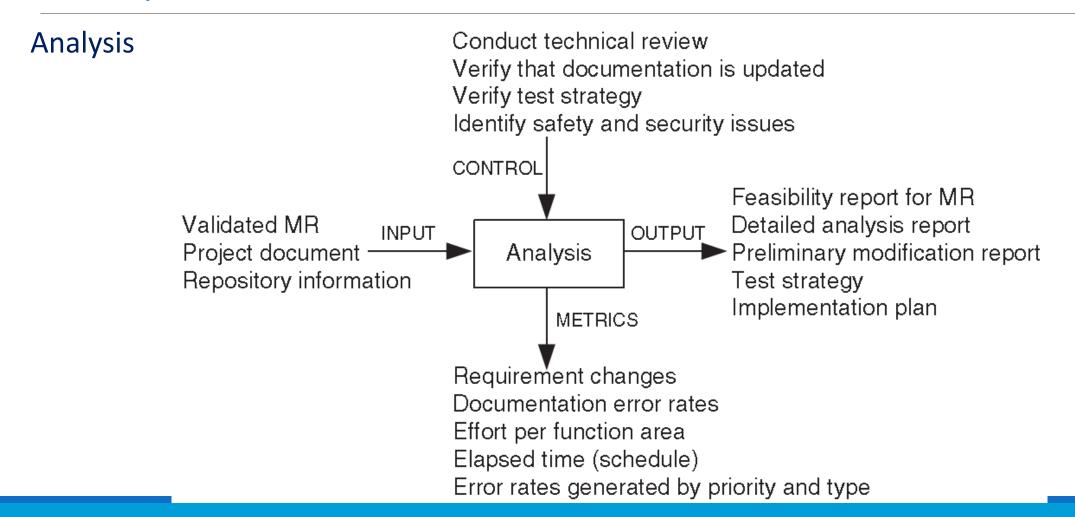
Activities include:

reject or accept the MR,

identify and estimate the resources needed to change the system

3. put the MR in a batch of changes scheduled for implementation.





Conduct software inspection Design Verify that design is documented Complete traceability of requirements to design CONTROL Revised modification list Project document Updated design baseline OUTPUT INPUT. Analysis phase output-Design Updated test plan Source code database Revised detail analysis Verified requirements **METRICS** Revised implementation plan Documented constraints and risks Software complexity Design changes Effort per function area Elapsed time (schedule) Test plan and procedure changes Error rates generated by priority and type Number of lines of code added, deleted,

modified, and tested

Implementation

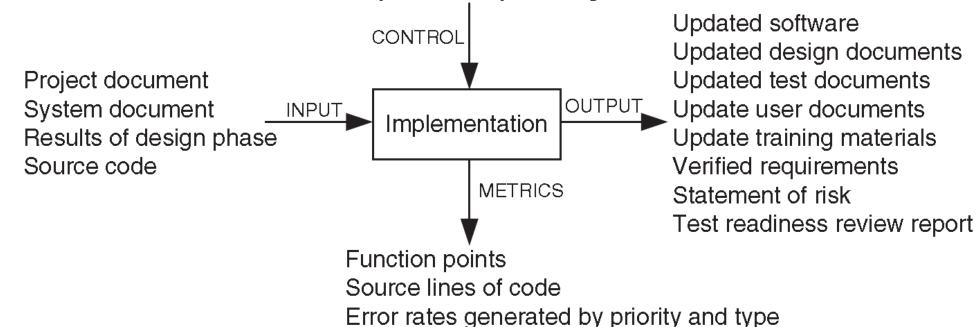
Conduct software inspections

Ensure that unit and integration testing are performed

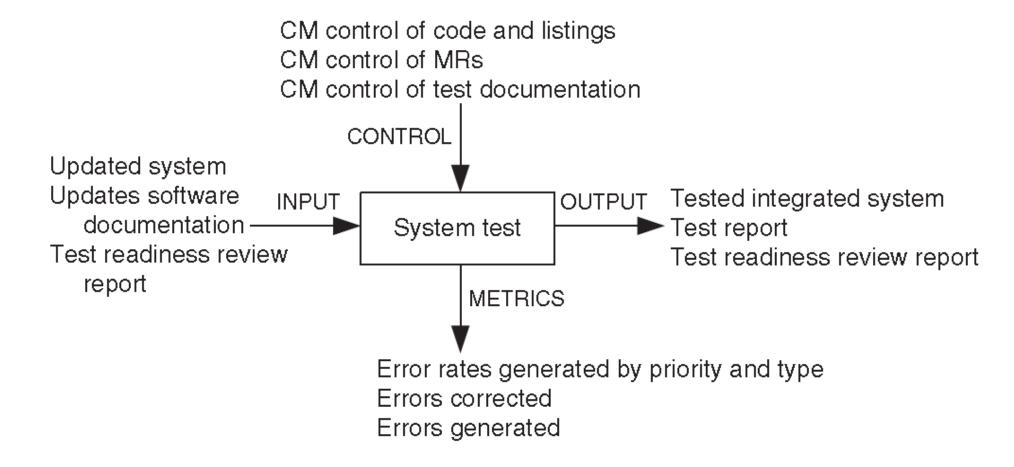
Ensure software placed under CM control

Ensure training and documentation have been updated

Verify traceability of design to code



System test



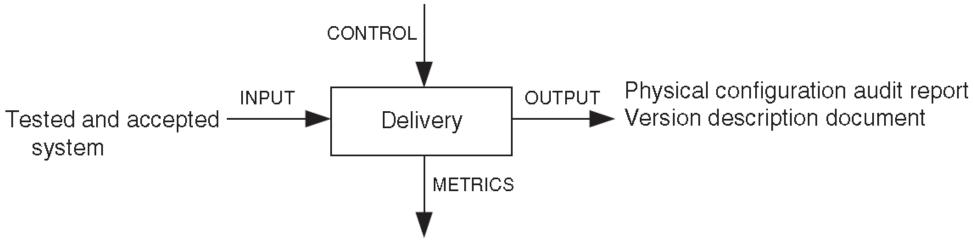
Execute acceptance tests Acceptance test Report test results Conduct functional audit Establish new baseline Place acceptance test documentation under CM CONTROL Test readiness review report New system baseline Fully integrated system **INPUT** OUTPUT Acceptance Functional configuration audit Acceptance test plan test report Acceptance test cases Acceptance test report **METRICS** Acceptance test procedures Error rates generated by priority and type

Errors corrected

Errors generated

Delivery

Arrange physical configuration audit Complete version description document Complete updates to status accounting database



Documentation changes

- training manuals
- operation guidelines
- version description document

Questions



Readings

Chapter 3: 3.4, 3.5, 3.6