# Software Maintenance Life Cycle

SADAF FARHAN

# Software Maintenance

Process of changing, modifying, and updating software to keep up with customer needs.

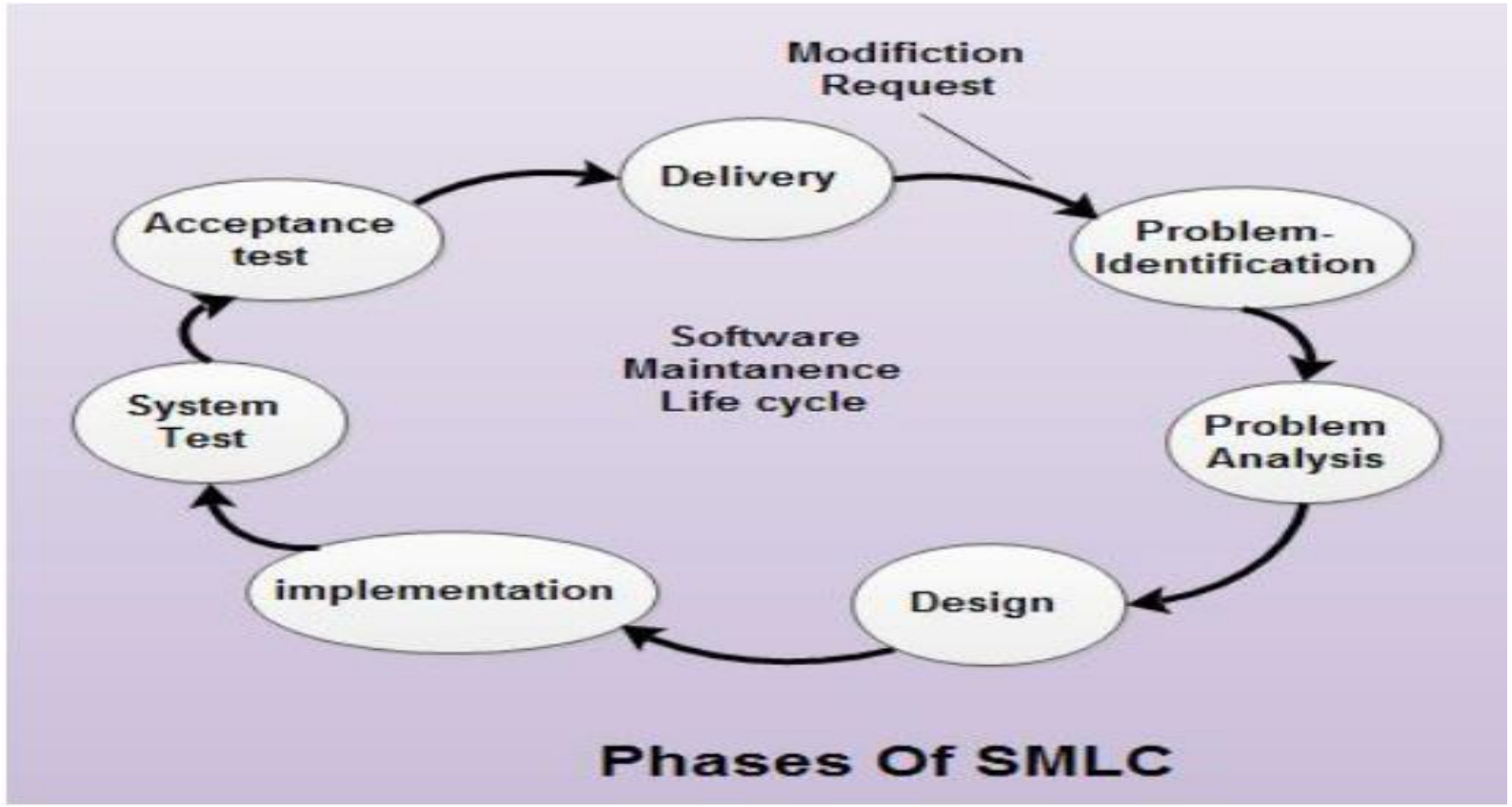
Done after the product has launched for several reasons including:

- improving the software overall,
- correcting issues or bugs,
- to boost performance,
- and more.

# Software Maintenance Life Cycle

Process of implementing the change required.

# SMLC


Modifiction
Request
Delivery
Problem-
Identification
Problem
Analysis
Design
implementation
System
Test
Acceptance
test
Software
Maintanence
Life cycle
Phases Of SMLC

# SMLC

All of the phases of SMLC evolve through four attributes:

Input.

Process.

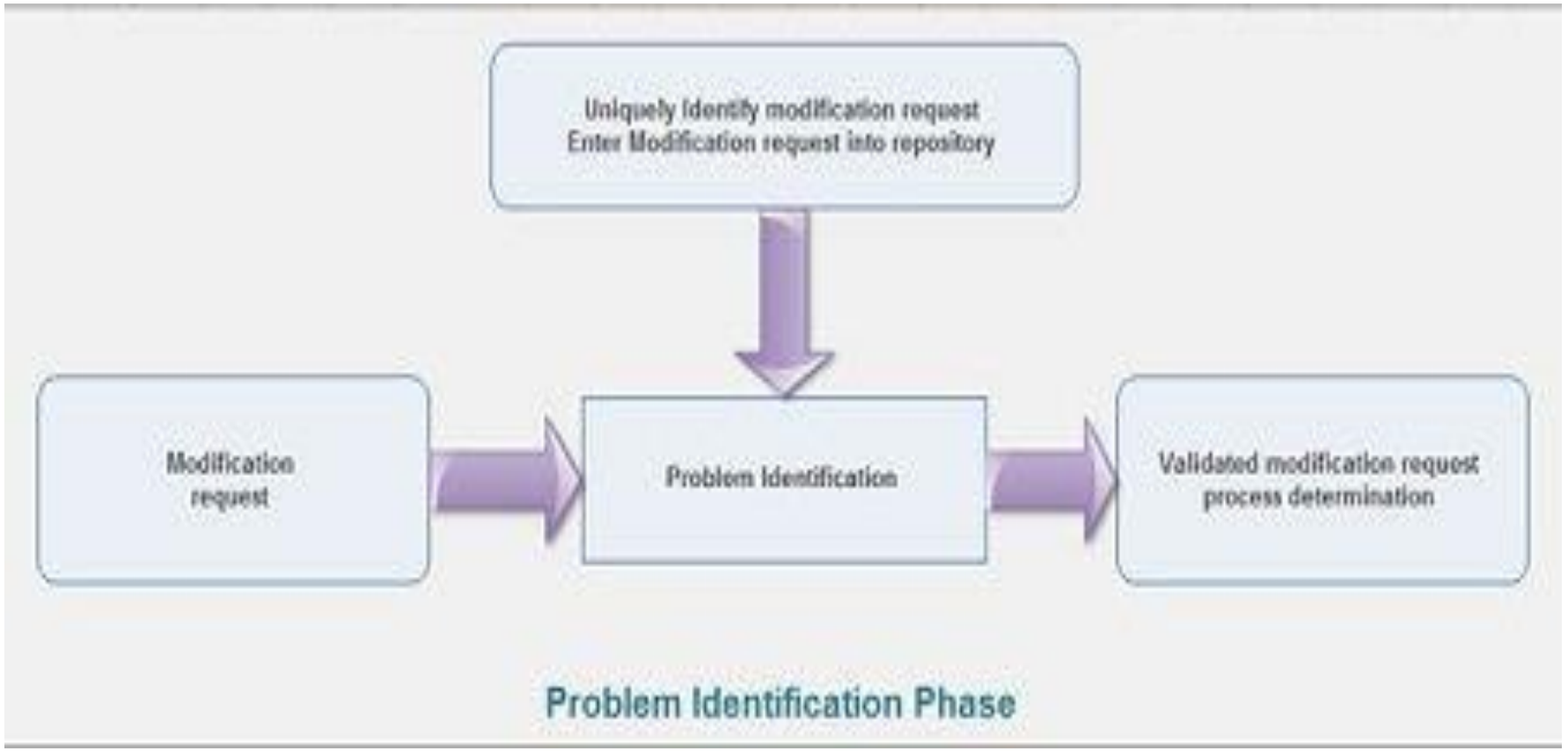
Control.

Output.

# Problem Identification Phase

| Input | Process | Control | Output |
|---|---|---|---|
| •Modification request (MR) | •Assign identification number<br>•Classify MR into appropriate category<br>•Corrective.<br>•Adaptive.<br>•Perfective.<br>•Preventive.<br>•Accept or reject change<br>•Assign a priority | •Uniquely identified modification request<br>•Enter processed modification request in repository | •Validated modification request |

# Problem Identification Phase



# Repository Components

Statement of problem or modification request,

Requirement evaluation,

Type of software maintenance,

Initial priority,

Estimate of resources required in software maintenance.

# Problem Analysis Phase

| | | | | |
|---|---|---|---|---|
| | •Project document<br>•Repository information<br>•Validated modification request | •Feasibility study<br>•Detailed analysis | •Conduct technical review<br>•Verify test strategy<br>•Verify whether documentation is updated or not<br>•Identify security issues | •Feasibility report<br>•Updated requirements<br>•Test strategy<br>•Detailed analysis report<br>•Preliminary modification list |

# Problem Analysis Phase

**Feasibility analysis** outcomes:

1. Impact of the changes
2. Alternative solutions including prototyping
3. Safety and security implications
4. Human factors
5. Short-term and long-term costs.

# Problem Analysis Phase

**Detailed analysis** results include:

1. Defining firm requirements for modification
2. Determining the elements to be modified
3. Determining safety and security issues
4. Devising an implementation plan
5. Preparing a test strategy

# Design Phase

| | | | | |
|---|---|---|---|---|
| | •Project document<br>•Source code<br>•Databases<br>•Output of analysis phase | •Revise requirements<br>•Revise implementation plan<br>•Develop test cases | •Software inspections /reviews<br>•Verify design | •Refined modification list<br>•Refined detailed analysis<br>•Modified test plans |

# Design Phase

The **process attribute** for design comprises the following steps.

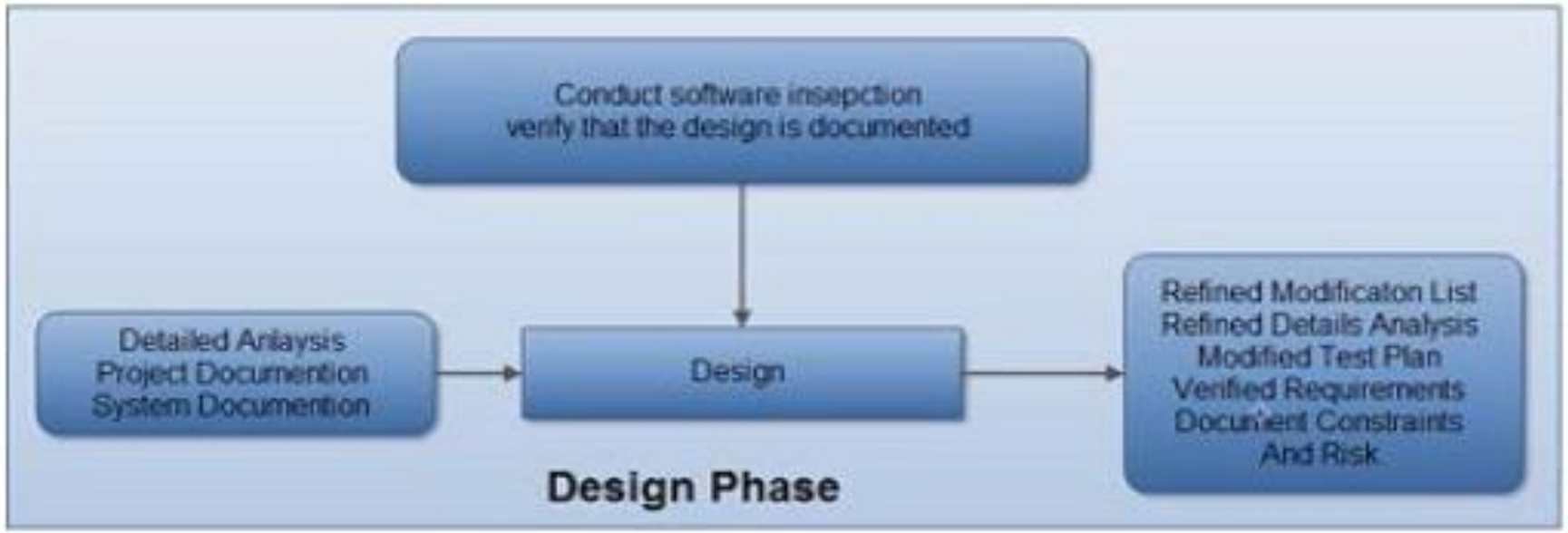
1. Identifying the affected software modules
2. Modifying software module documentation (like data-flow diagrams)
3. Developing test cases for the new design including safety and security issues.
4. Documenting the updated requirements
5. Revising the list of modifications.

# Design Phase

The **control attribute** for design comprises the following steps.

1. Conducting software inspection of the design.
2. Verifying that the new design/requirement has been documented
3. Verifying that the new design along with safety and security issues has been included
4. Verifying that the test documentation has been modified.

# Design Phase



# Implementation Phase

| | | | | |
|---|---|---|---|---|
| | •Source code<br>•Project documentation<br>•Output of design phase | •Software code<br>•Unit test | •Software inspections/review | Updated software<br>Updated design documents<br>Updated test documents<br>Updated user documents |

# Implementation Phase

The **process attribute** for implementation comprises the following steps.

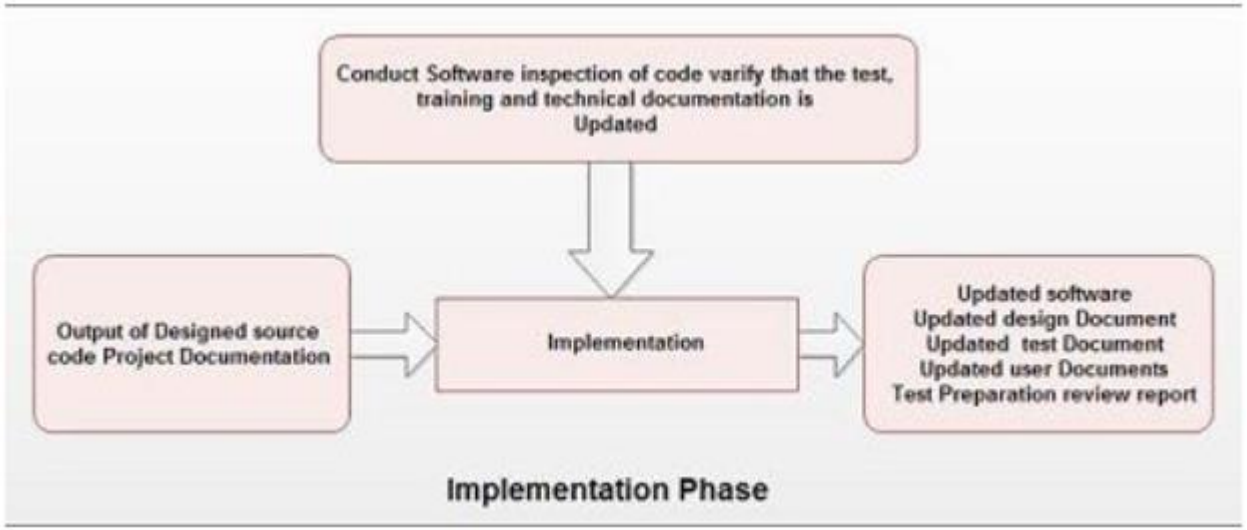
1. Coding and unit testing.
2. Integration testing.
3. Risk analysis and reviews.

# Implementation Phase

The **control attribute** for implementation comprises the following steps.

1. Conducting software inspections.
2. Ensuring that the unit and integration testing have been performed and documented.
3. Ensuring that test documentation such as test plan and test cases are either updated or created.
4. Verifying that the changes in training and technical documentation have been made.

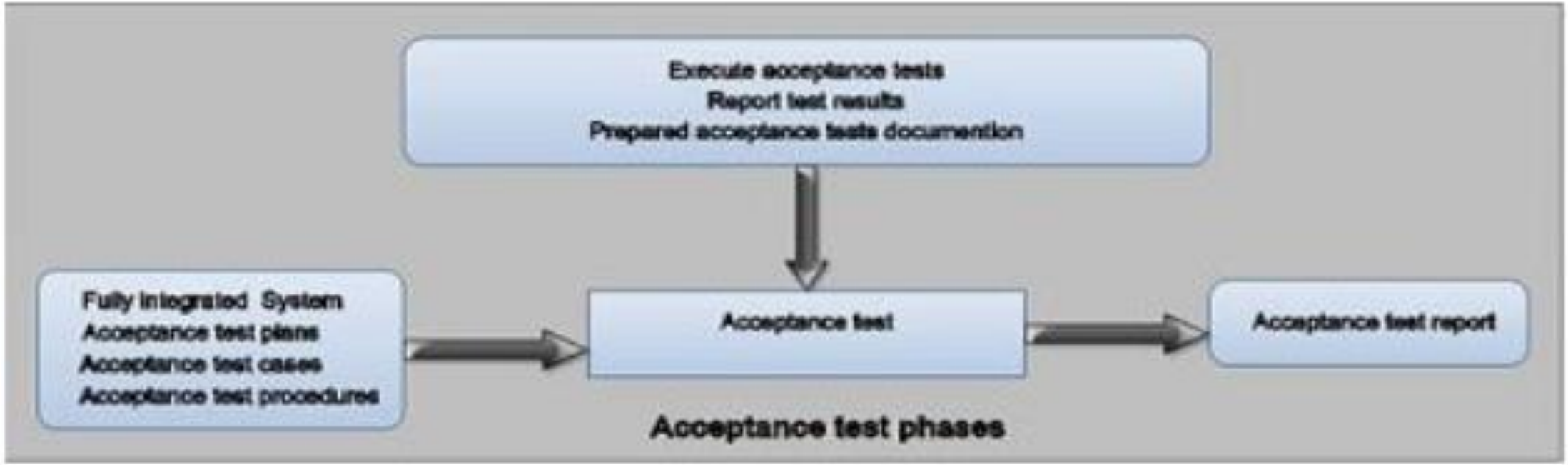
# Implementation Phase



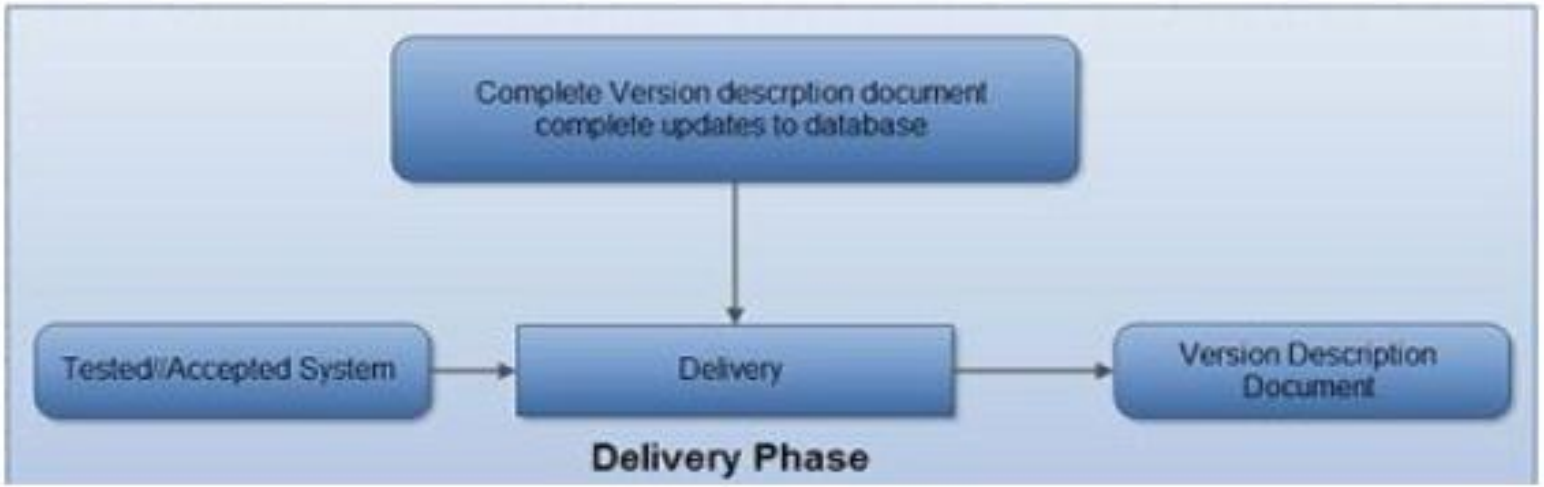
# System Testing Phase

| | | | | |
|---|---|---|---|---|
| | •Updated software documentation<br>•Test preparation review report<br>•Updated system | •Functional test<br>•Interface testing | •Verify test documentation | •Tested and integrated system<br>•Test report |

# Acceptance Testing Phase



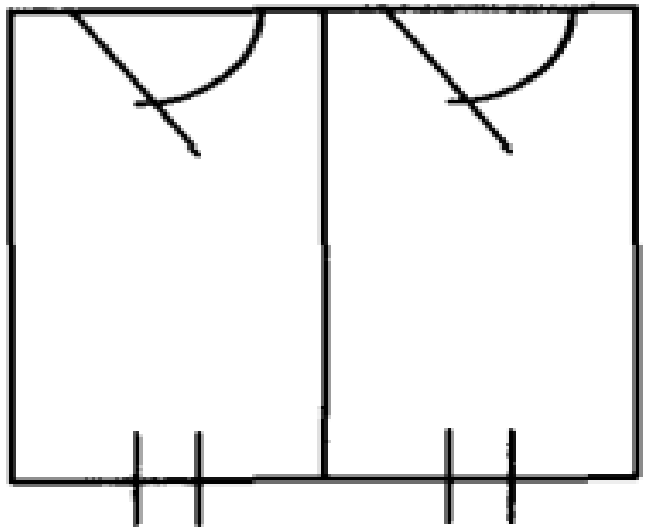
# Delivery Phase



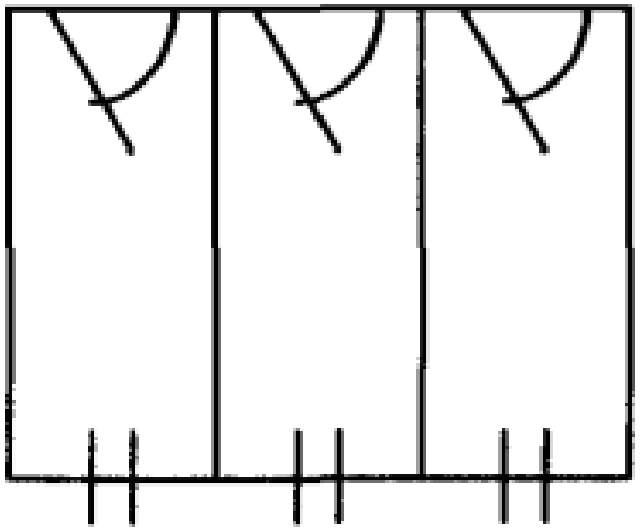
Easy to maintain?
Or

Easy to develop at first time?

The original



The later requirement



# Points to Consider

The wall between rooms A and B must be knocked down.

Software interfaces between different components may have to be altered

# Points to Consider

This is a building in use - the problem of creating and removing a pile of rubble must be addressed.

There is far less leeway to allow for the introduction of errors and ripple effects in a piece of software which must be released quickly to a large customer base.

# Points to Consider

Adding the third room may well require people and materials to travel through, and thus affect, parts of the building they would not have had to access originally.

A modification to a large and complex software system has the potential to affect parts of the software from which it could have been kept completely separate had it been added originally

# Points to Consider

Is the wall between A and B a load-bearing wall? If so, there will be a need for a supporting joist.

The software will have to be modified to cater for the addition of the new functionality. Suppose that the new functionality calls for data to be held in memory in a large table. It may be that the existing system does not allow the creation of such a structure because of memory constraints.

# Points to Consider

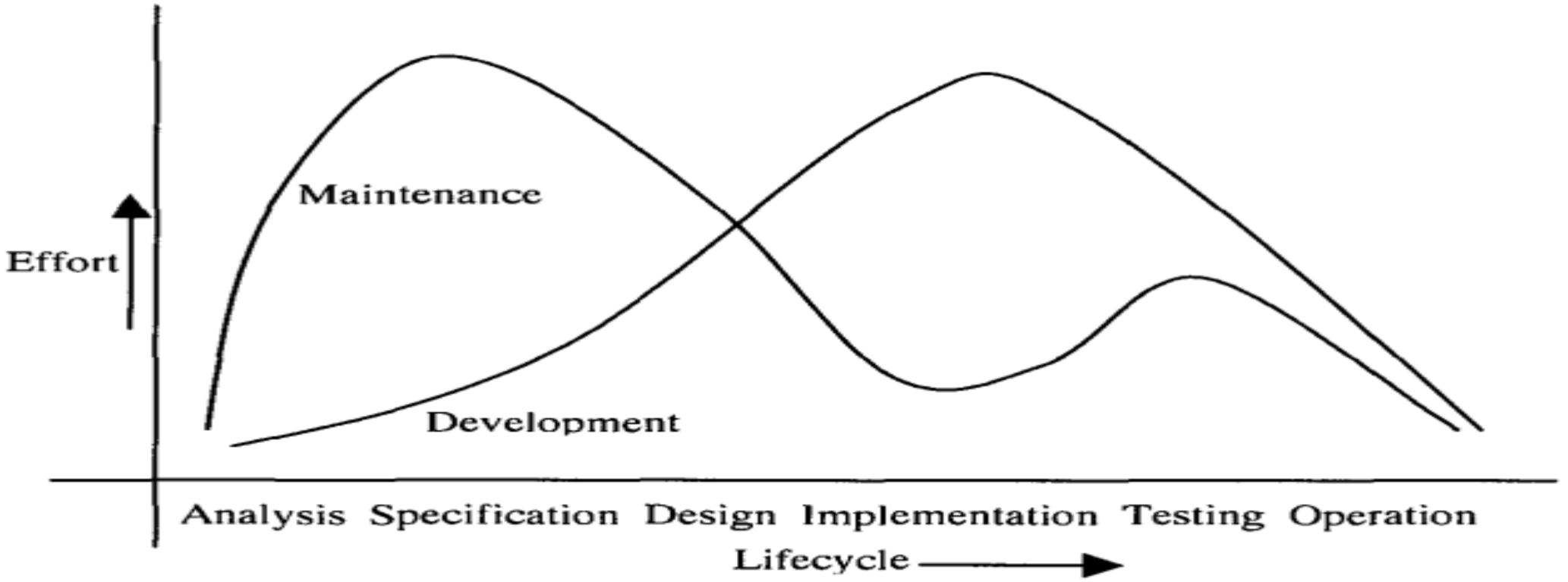
Does the wall contain central heating pipes, wiring ducts, network cables or anything else which may have to be taken into account prior to its being demolished?

Likewise, are there hidden dependencies within the software modules which are to be modified?

# Similarities

The same skills and expertise that are required to build the new wall whether constructing it in the new building or adding it later.

The stages of SDLC and SMLC are almost same,


Maintenance
Effort
Development
Analysis Specification Design Implementation Testing Operation
Lifecycle

# The End