UMT MOOC
Massive Open Online Course

MINISTRY OF HIGHER EDUCATION

UNIVERSITI MALAYSIA TERENGGANU

UMT MOOC
Massive Open Online Course

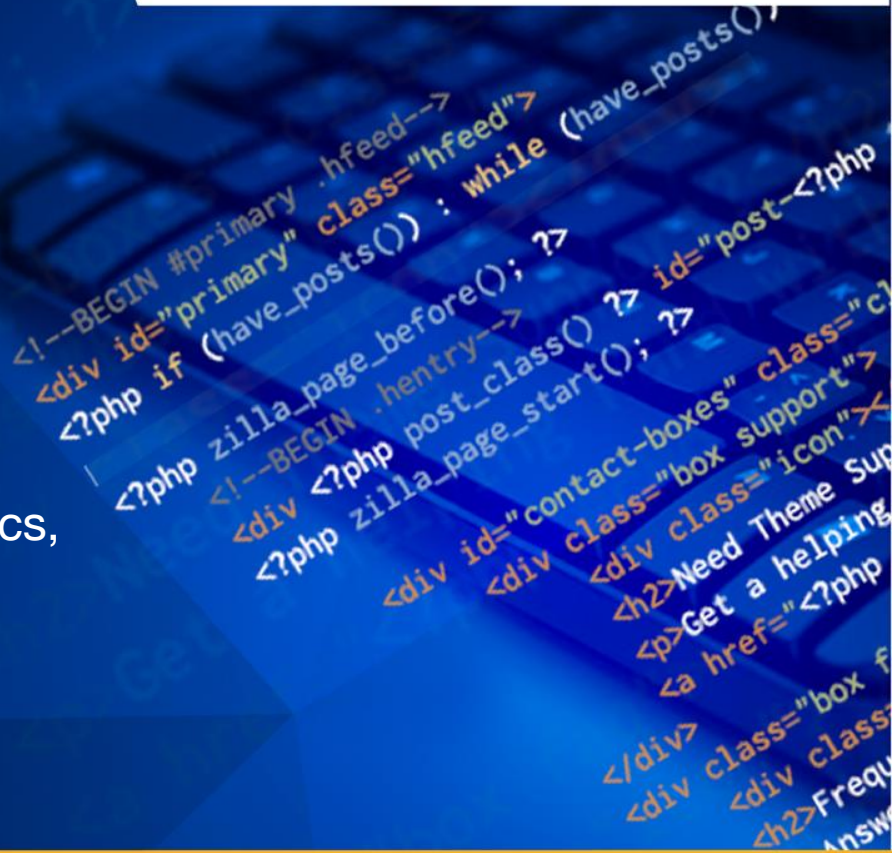
SOFTWARE ENGINEERING

UNIVERSITI MALAYSIA TERENGGANU

SOFTWARE PROCESS

Presented By :
Dr. Abdul Aziz K. Abdul Hamid
School of Informatics and Applied Mathematics,
Universiti Malaysia Terengganu
MALAYSIA

Email: abdulazizkah@umt.edu.my



Objectives

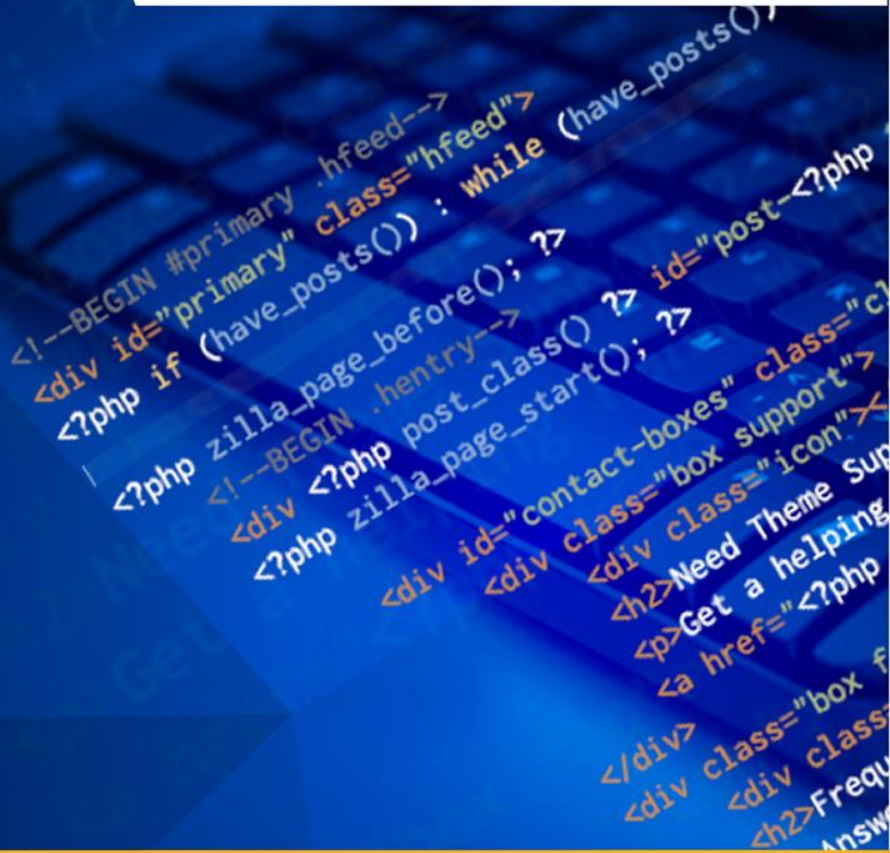
- To introduce software process models
- To describe a number of different process models and when they may be used
- To describe outline process models for requirements engineering, software development, testing and evolution
- To explain the Rational Unified Process model

Learning Outcomes

To explain the software process models.

To explain methods in software development phaseds based on methodology used.

To relate engineering design in software development.



Contents

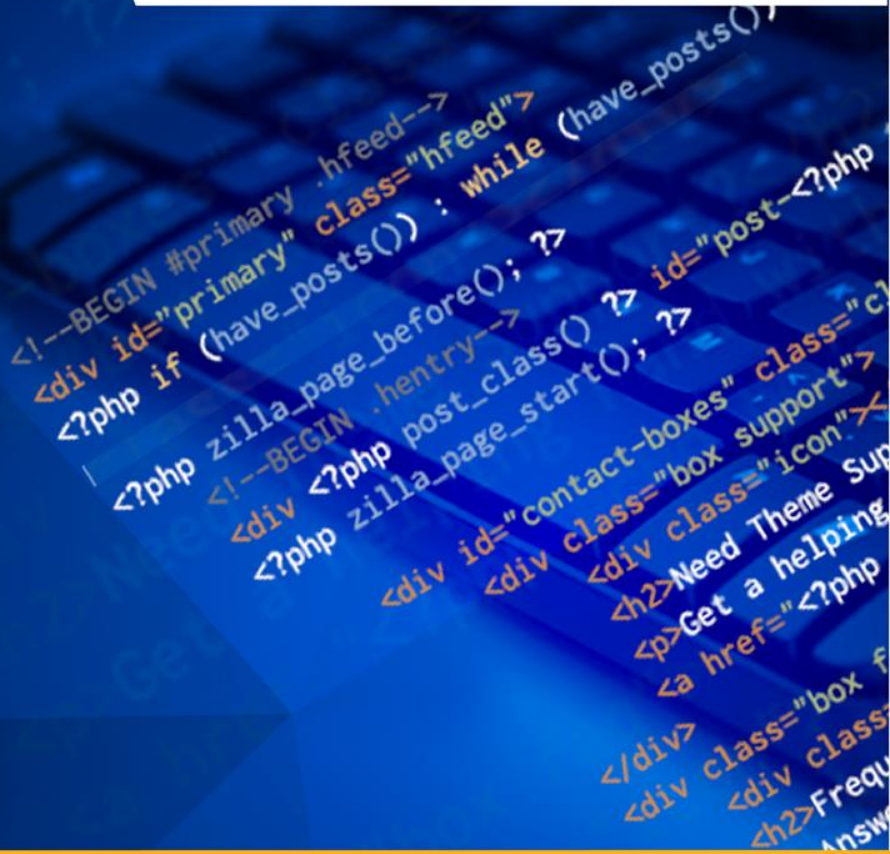
Software Process Model (SPM)

- What is software process?
- Why is software process model?
- Types of SPM
- Strength and Weakness of SPM

Process Activities

- Specification
- Design
- Implementation
- Validation
- Evolution

Rational Unified Process Modelling



What is software process?

Logical and structured sets of activities for specifying, designing, implementing, testing and evolution required to develop software systems

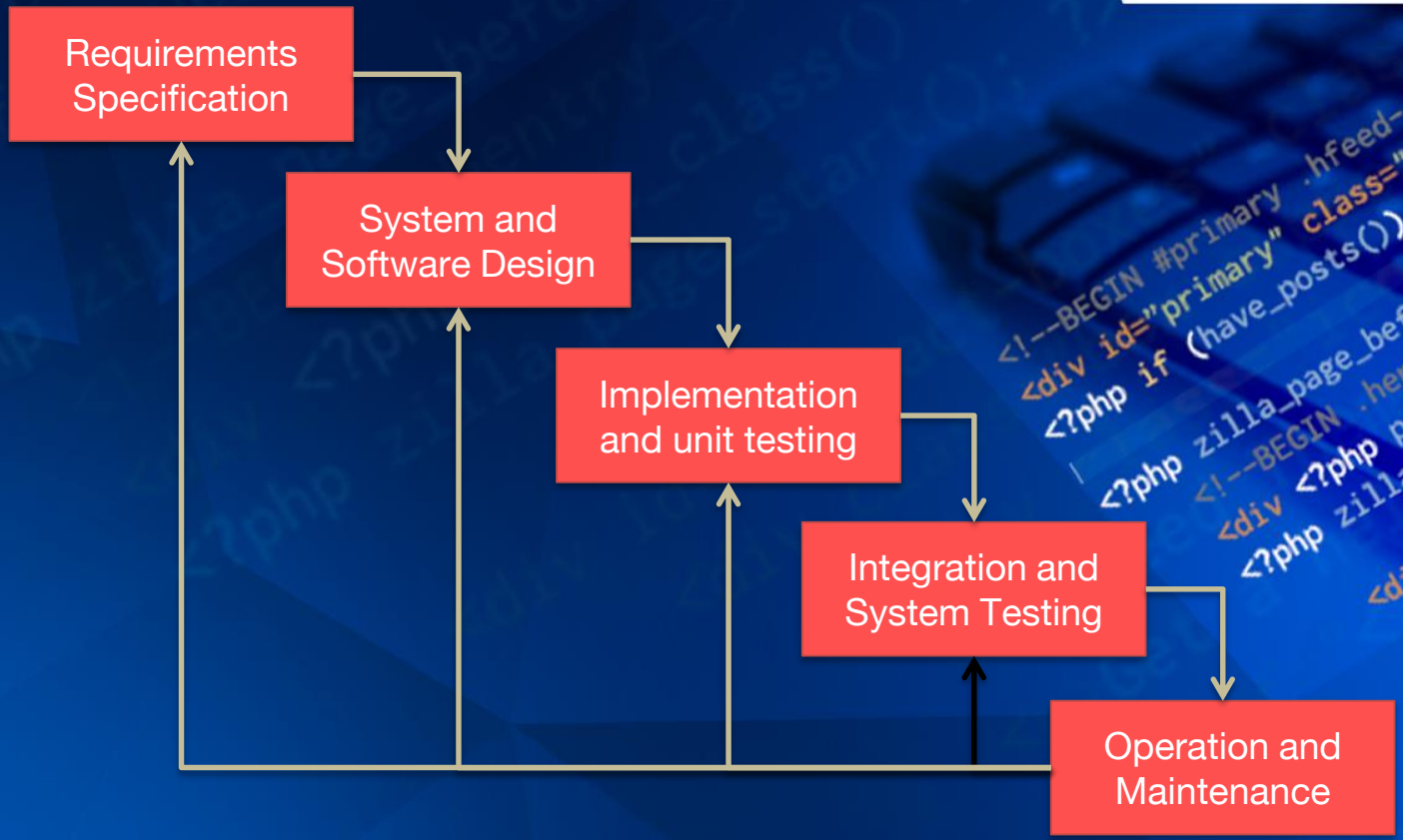
What is software process model?

An abstract representation of a process. It presents a description of a process from some particular perspective

Types of software process model?

- The Waterfall model
- Separate and distinct phases of specification and development
- Evolutionary/Incremental development
- Specification and development are interleaved
- Reuse-based development
- The system is assembled from existing components

Waterfall Model

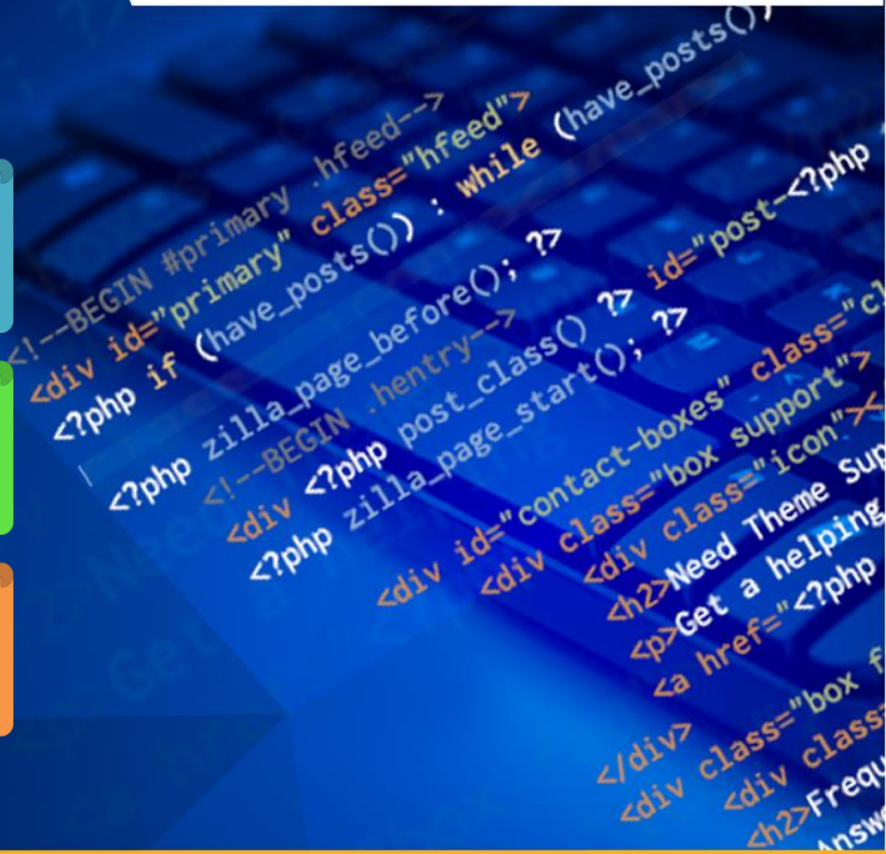


Drawbacks of Waterfall Model

Difficult to accommodate with user requirements changes after the process is running

Inflexible partitioning of the project into different stages makes it difficult to respond to changing customer requirements

Appropriate only when the requirements are well-understood



Evolutionary/Incremental Devel

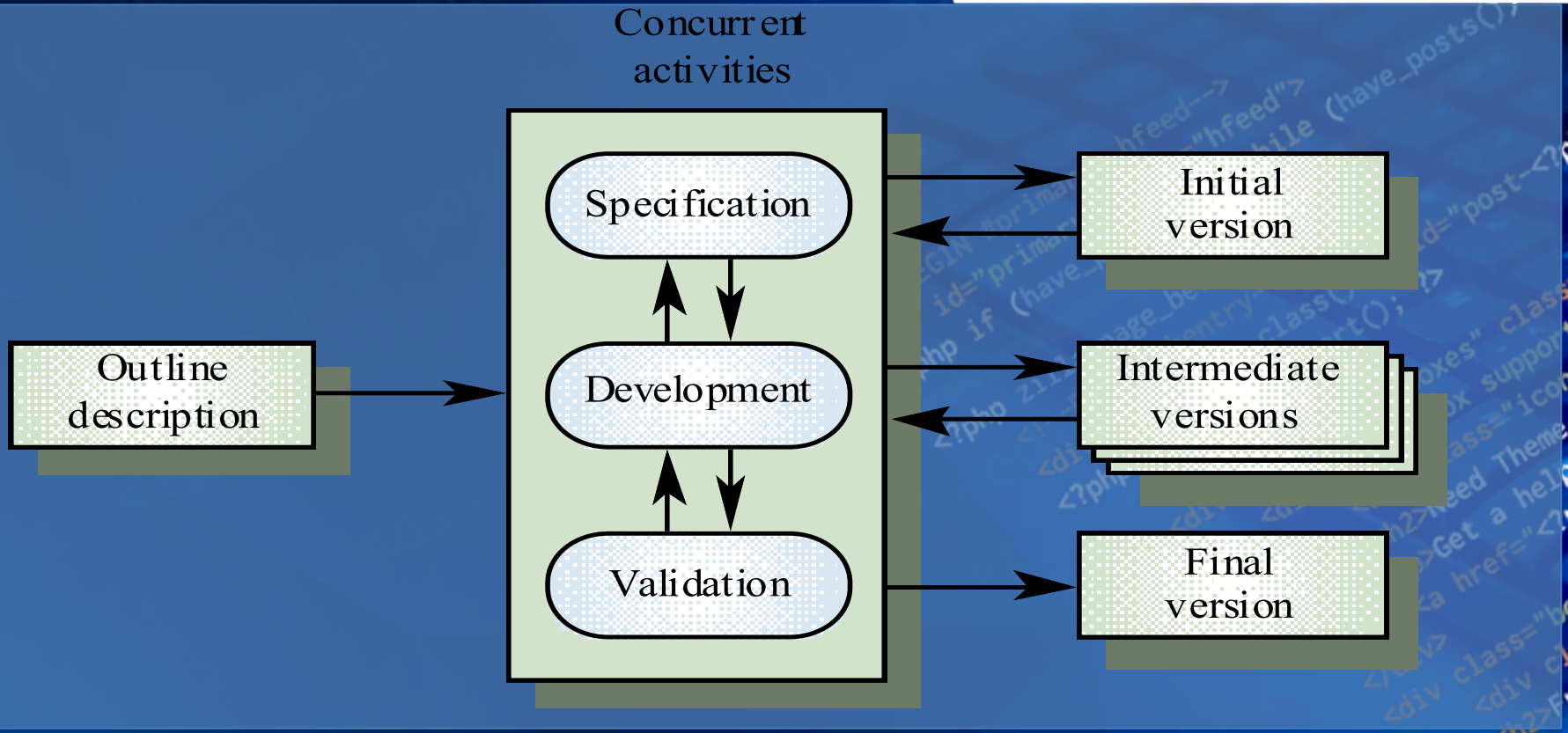
Exploratory development

Objective is to work with customers and to evolve a final system from an initial outline specification. Should start with well-understood requirements and add new features as proposed by the customer.

Throw-away prototyping

Objective is to understand the system requirements. Should start with poorly understood requirements to clarify what is really needed

Evolutionary/Incremental Devel



Evolutionary/Incremental Development

Benefits

- The cost of accommodating changing customer requirements is reduced.
- It is easier to get customer feedback on the development work that has been done.
- More rapid delivery and deployment of useful software to the customer is possible.

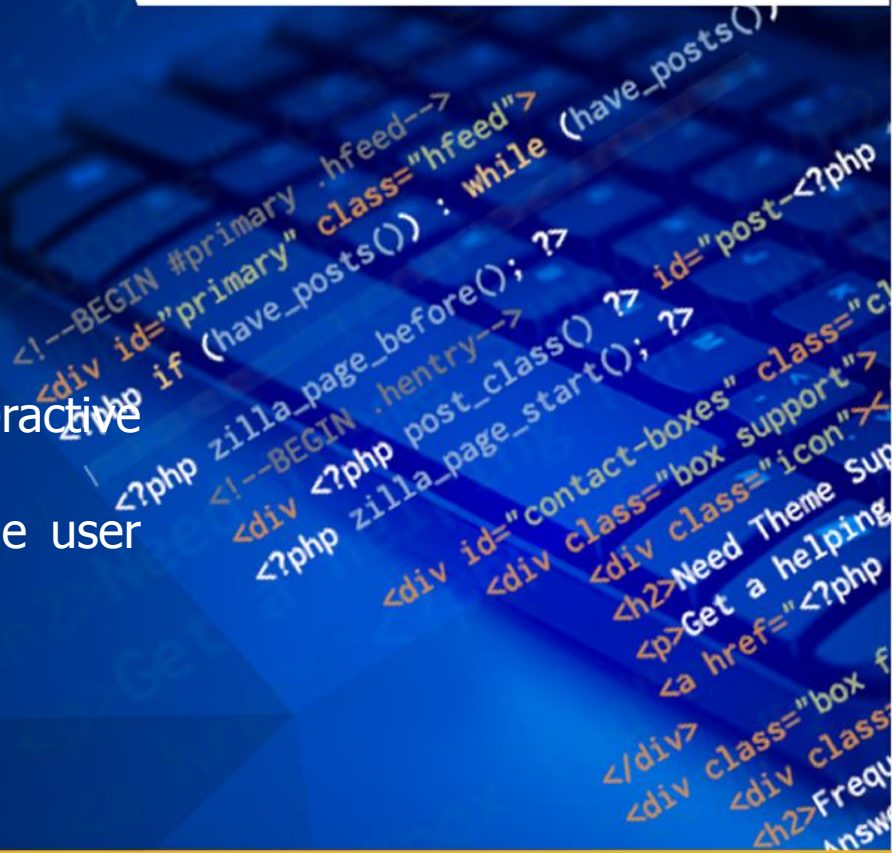
Evolutionary/Incremental Development

Problems

- Lack of process visibility
- Systems are often poorly structured

Applicability

- For small or medium-size interactive systems
- For parts of large systems (e.g. the user interface)
- For short-lifetime systems



Reuse-based software engineeri

MINISTRY OF HIGHER EDUCATION

Universiti Malaysia Terengganu

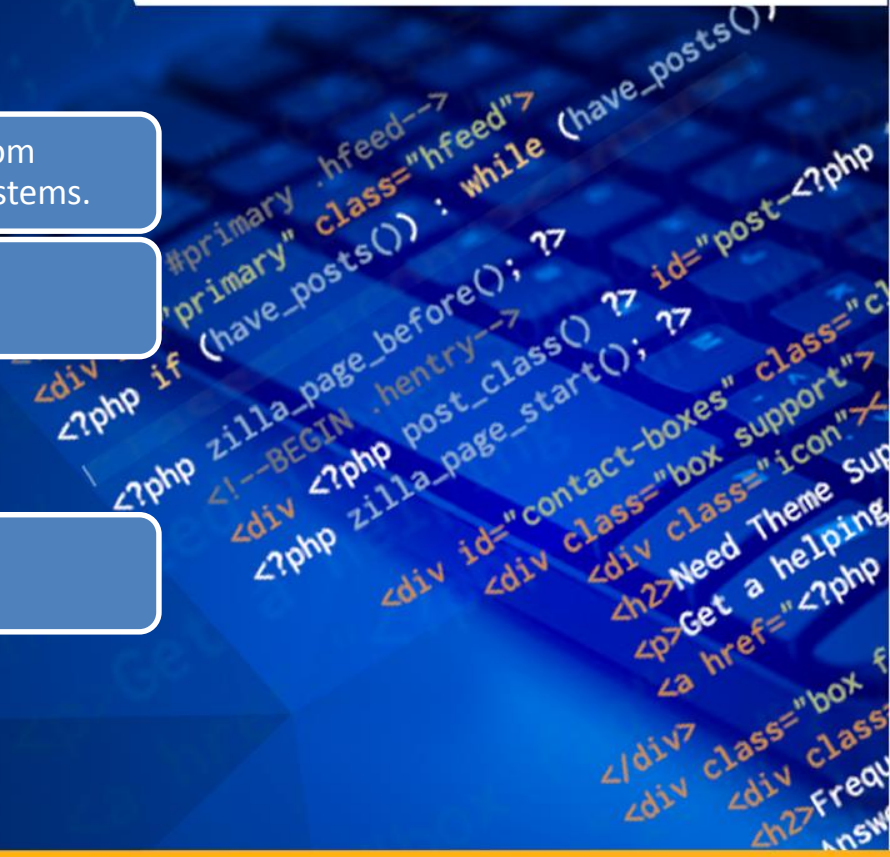
UMT MOOC
Massive Open Online Course

Based on systematic reuse where systems are integrated from existing components or COTS (Commercial-off-the-shelf) systems.

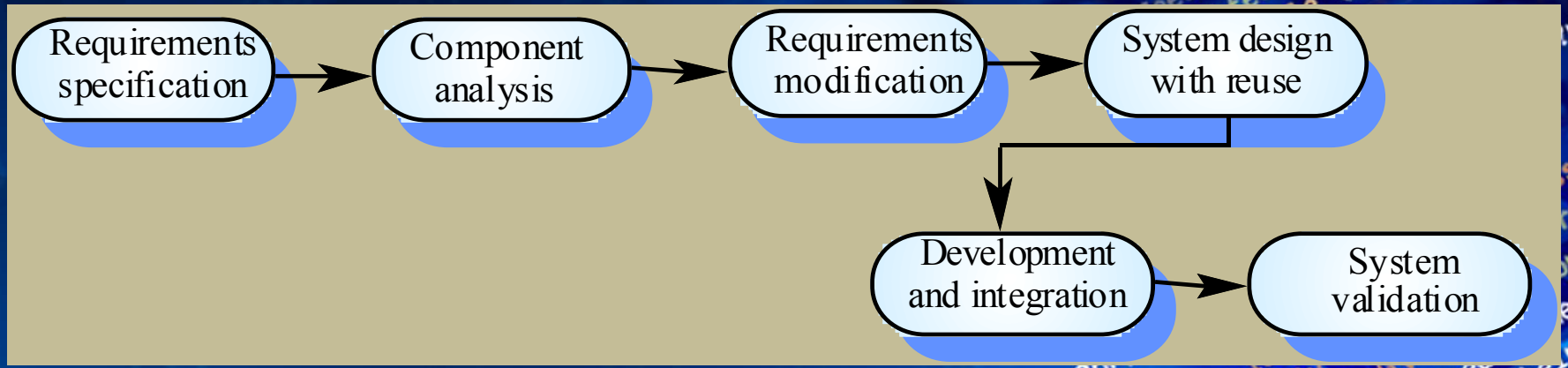
Process stages

- Component analysis;
- Requirements modification;
- System design with reuse;
- Development and integration.

This approach is becoming increasingly used as component standards have emerged.

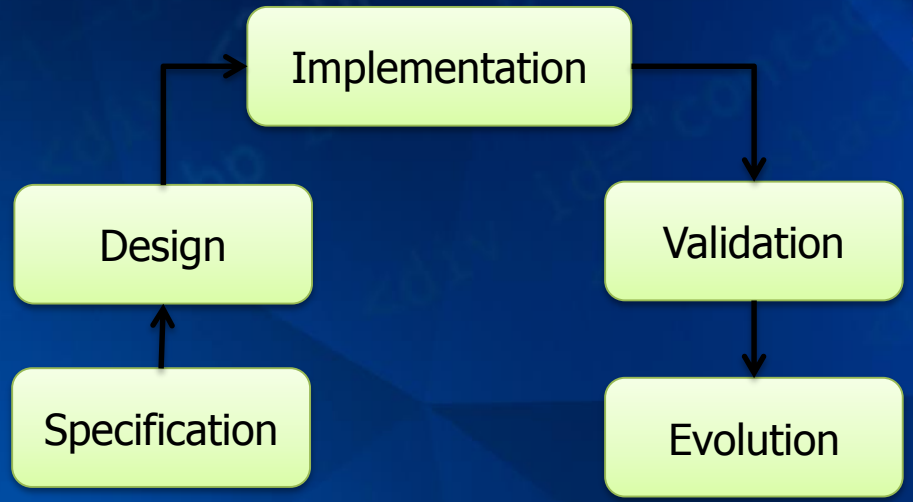


Component-based software engineering



What is Process Activities?

Real software processes are inter-leaved sequences of technical, collaborative and managerial activities with the overall goal of specifying, designing, implementing and testing a software system



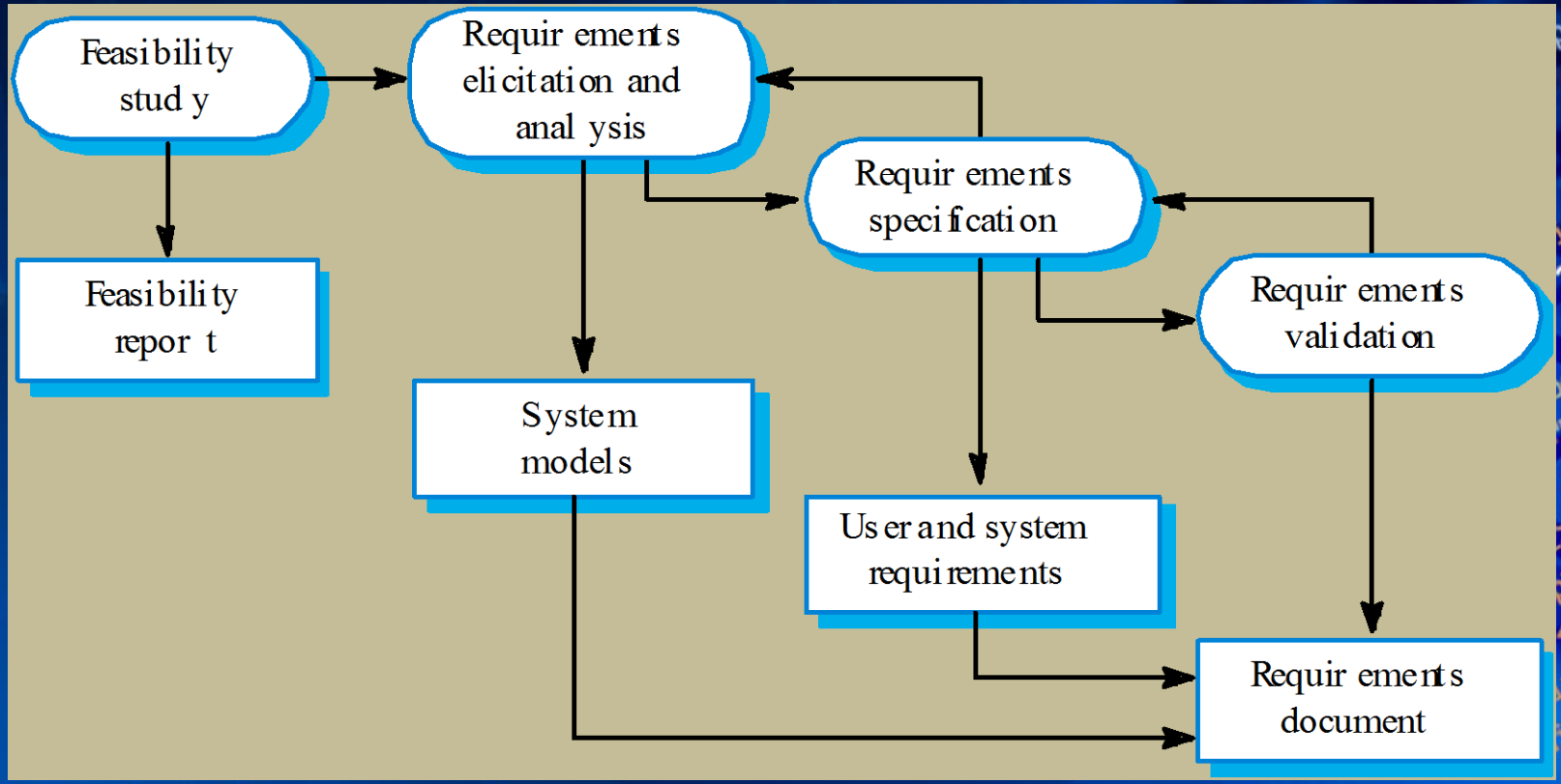
Software Specification

The process of establishing what services are required and the constraints on the system's operation and development.

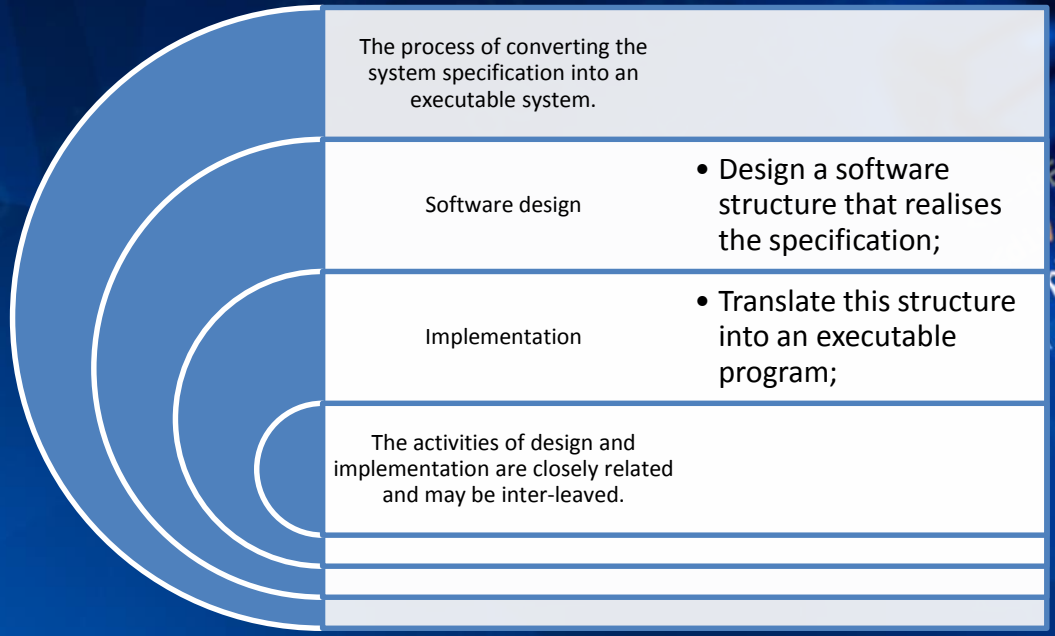
Requirements engineering process

- Feasibility study
- Requirements elicitation and analysis
- Requirements specification
- Requirements validation

Requirement Engineering Process



Software Design & Implementation



Design Process Activities

Architectural design	• identify the overall structure of the system, the principal components, their relationships and how they are distributed
Interface design	• define the interfaces between system components
Component design	• take each system component and design how it will operate
Database design	• design the system data structures and how these are to be represented in a database.

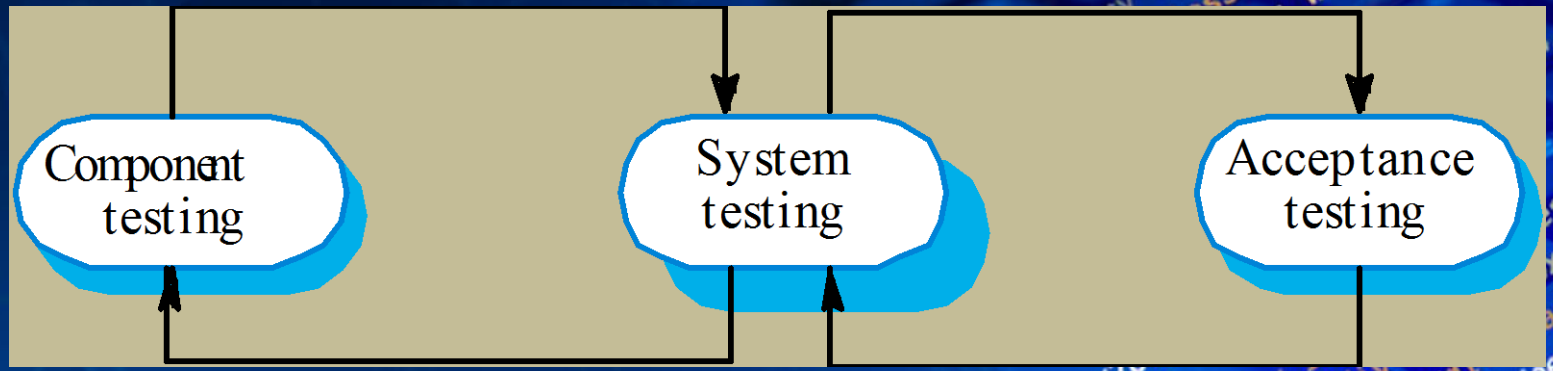
Software Validation

Verification and validation (V & V) is intended to show that a system conforms to its specification and meets the requirements of the system customer.

Involves checking and review processes and system testing.

System testing involves executing the system with test cases that are derived from the specification of the real data to be processed by the system.

Testing Process



Testing Stages

Component or unit testing

- Individual components are tested independently;
- Components may be functions or objects or coherent groupings of these entities.

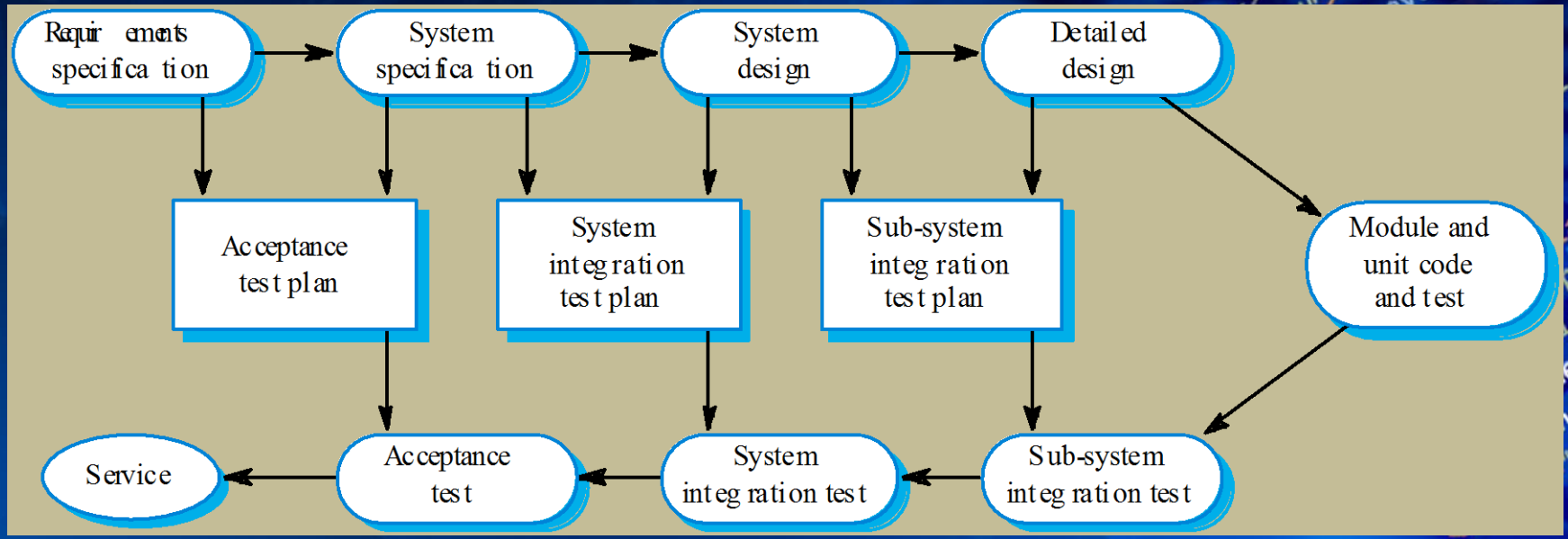
System testing

- Testing of the system as a whole. Testing of emergent properties is particularly important.

Acceptance testing

- Testing with customer data to check that the system meets the customer's needs.

Testing Phases



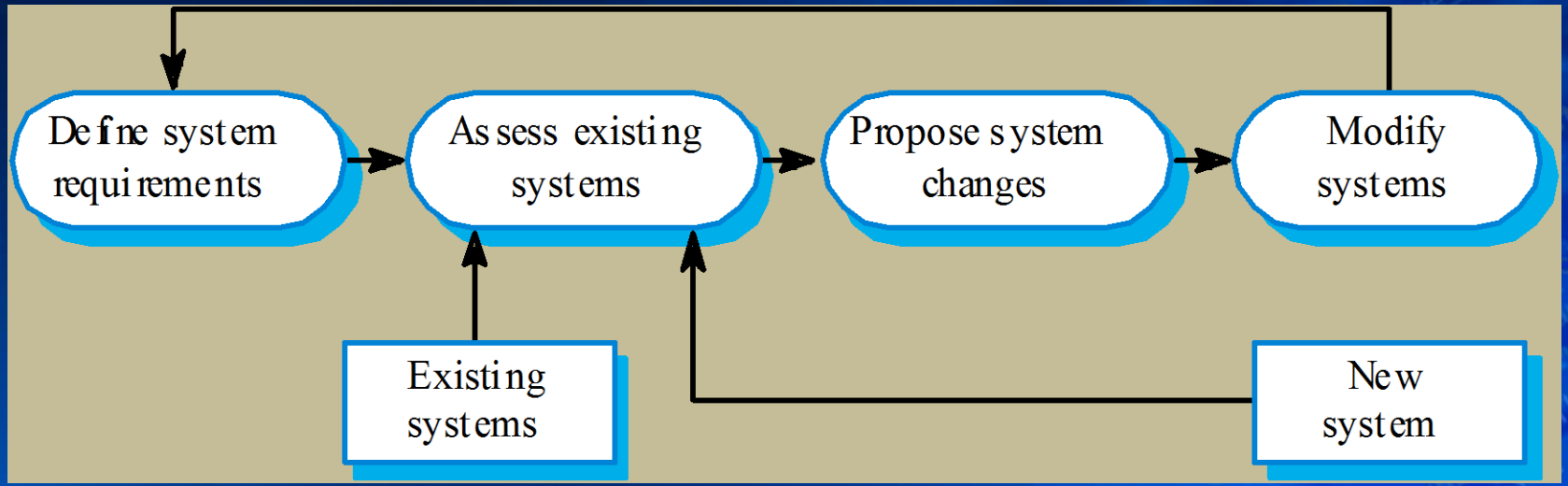
Software Evolution

Software is inherently flexible and can change.

As requirements change through changing business circumstances, the software that supports the business must also evolve and change.

Although there has been a demarcation between development and evolution (maintenance) this is increasingly irrelevant as fewer and fewer systems are completely new.

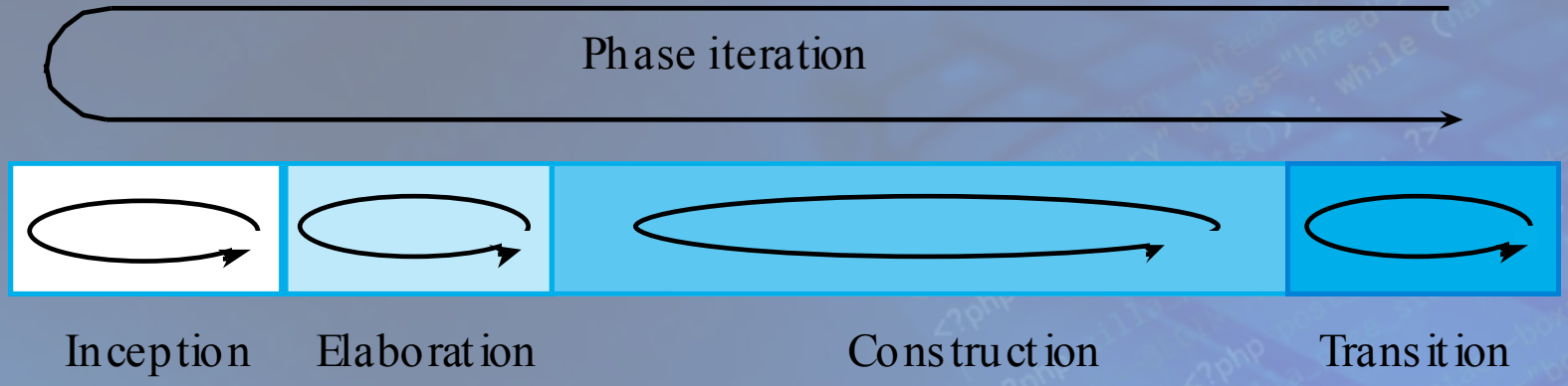
Software Evolution



Rational Unified Process

A modern process model derived from the work on the UML and associated process.	
Normally described from 3 perspectives	• A dynamic perspective that shows phases over time; • A static perspective that shows process activities; • A practice perspective that suggests good practice.

RUP Model


Phase iteration
Inception
Elaboration
Construction
Transition

RUP Phases

Inception

- Establish the business case for the system

Elaboration

- Develop an understanding of the problem domain and the system architecture.

Construction

- System design, programming and testing.

Transition

- Deploy the system in its operating environment.

RUP Phases

- Develop software iteratively
- Manage requirements
- Use component-based architectures
- Visually model software
- Verify software quality
- Control changes to software

MINISTRY OF HIGHER EDUCATION

UNIVERSITI MALAYSIA TERENGGANU

UMT MOOC
Massive Open Online Course

THANK YOU