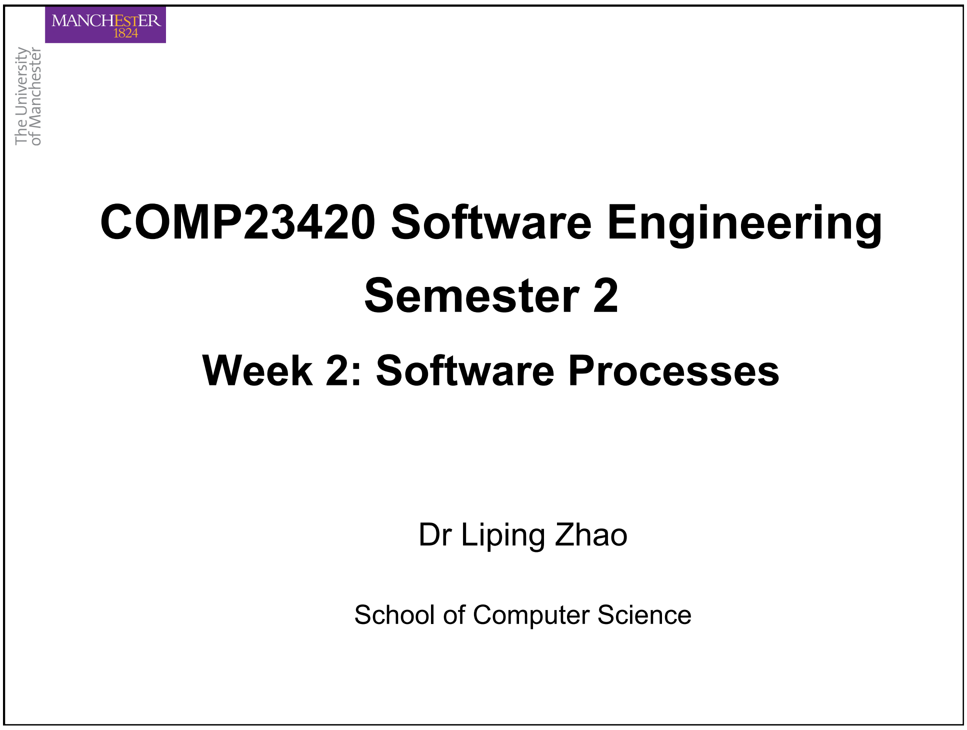
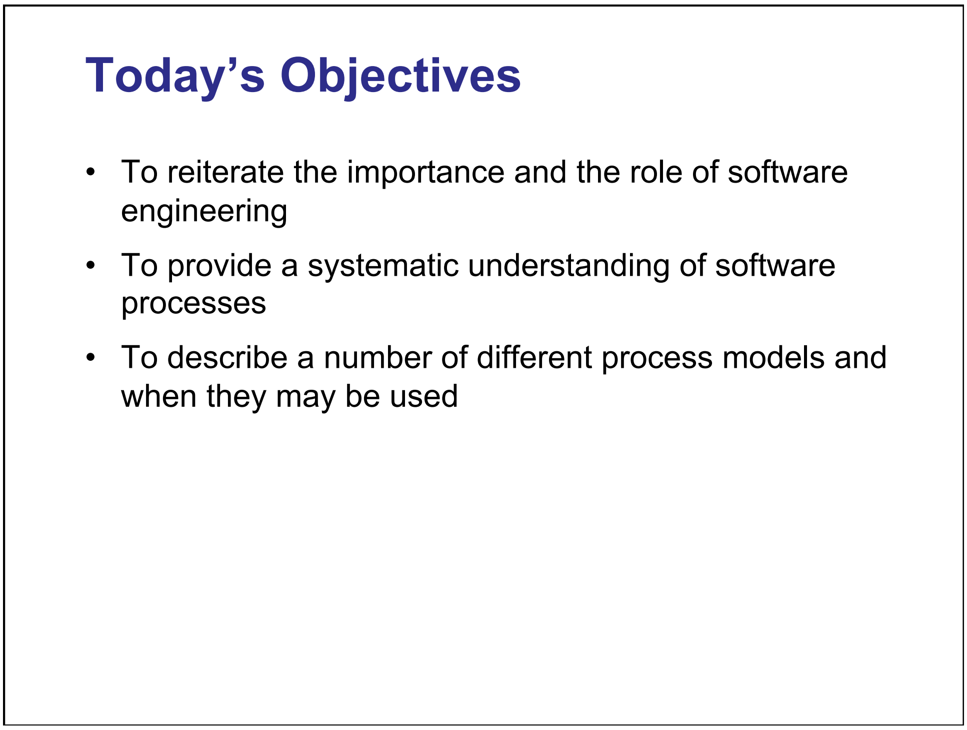


COMP23420 Software Engineering Semester 2

Week 2: Software Processes

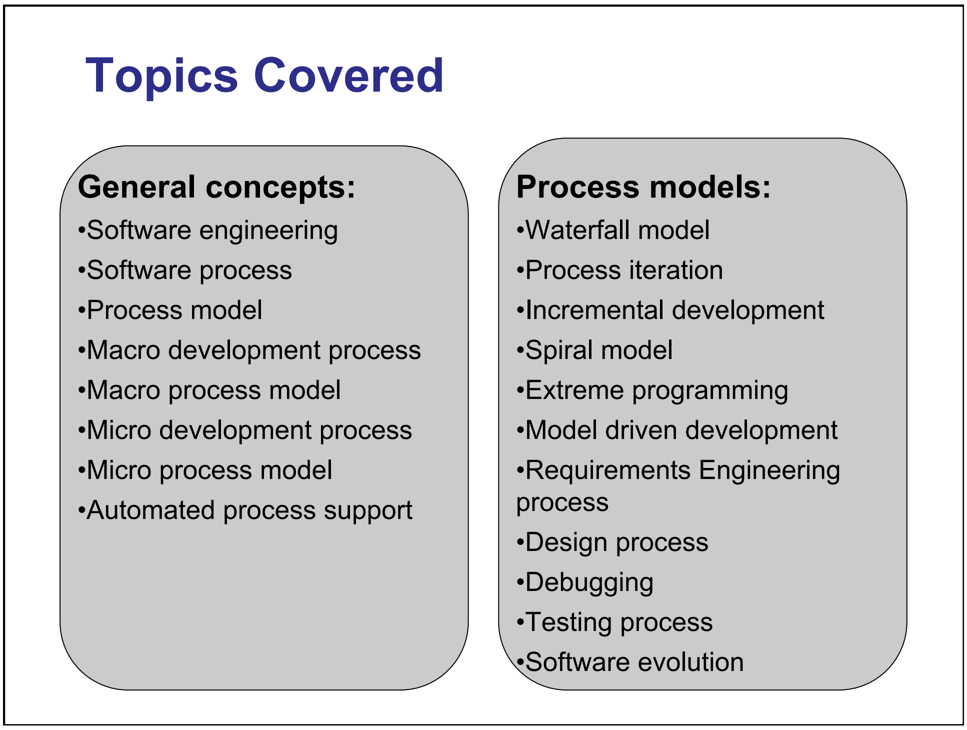
Dr Liping Zhao

School of Computer Science



Today's Objectives

- To reiterate the importance and the role of software engineering
- To provide a systematic understanding of software processes
- To describe a number of different process models and when they may be used



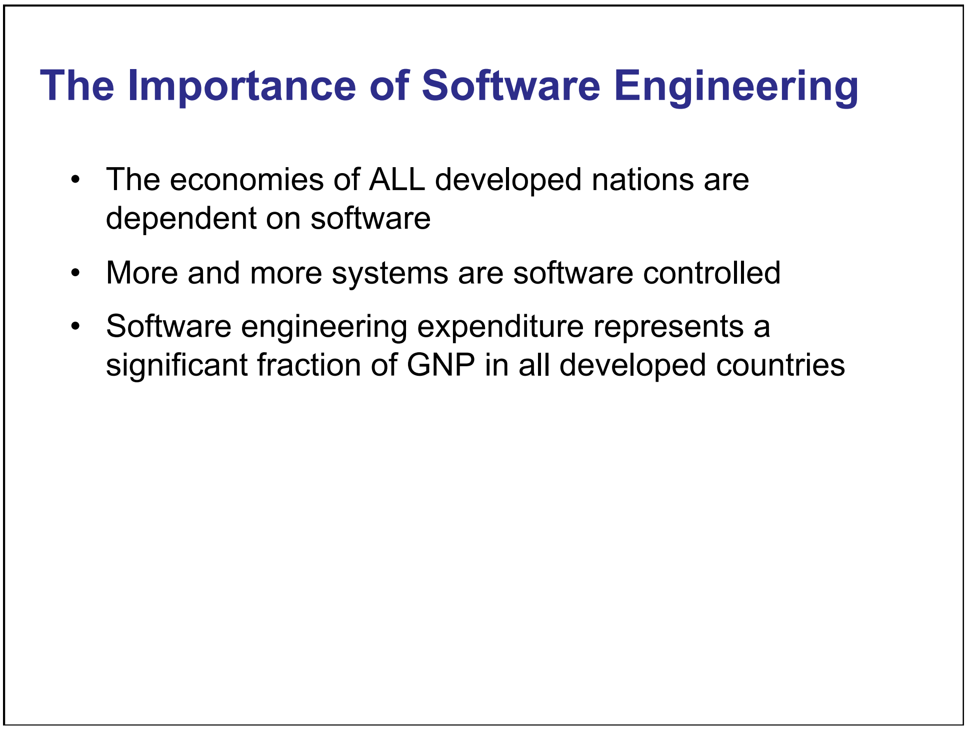
Topics Covered

General concepts:

- Software engineering
- Software process
- Process model
- Macro development process
- Macro process model
- Micro development process
- Micro process model
- Automated process support

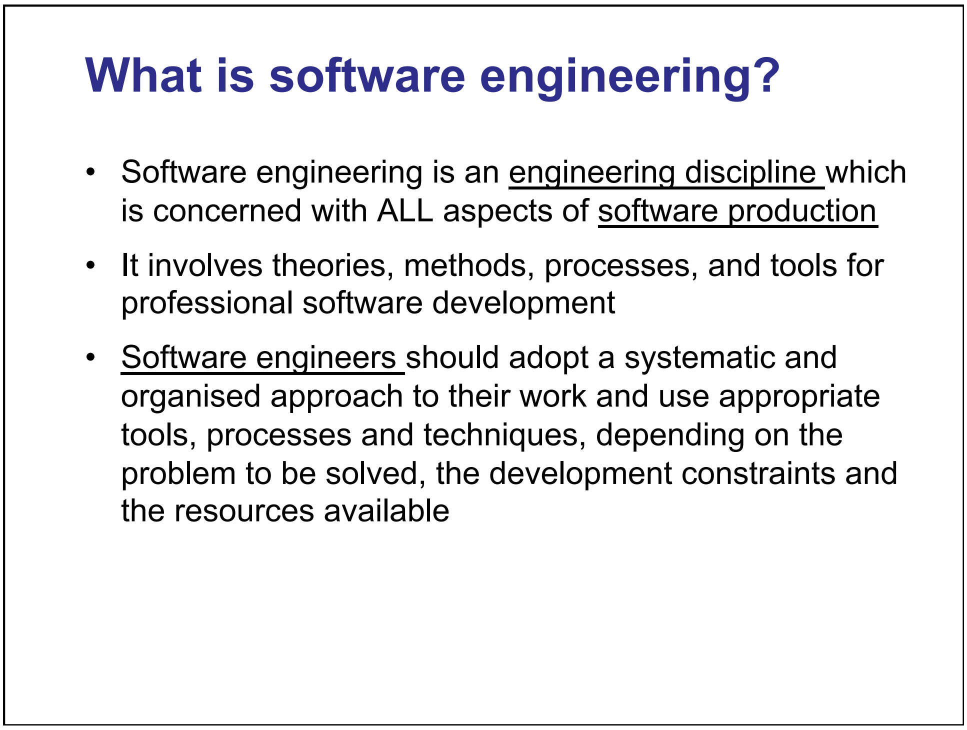
Process models:

- Waterfall model
- Process iteration
- Incremental development
- Spiral model
- Extreme programming
- Model driven development
- Requirements Engineering process
- Design process
- Debugging
- Testing process
- Software evolution



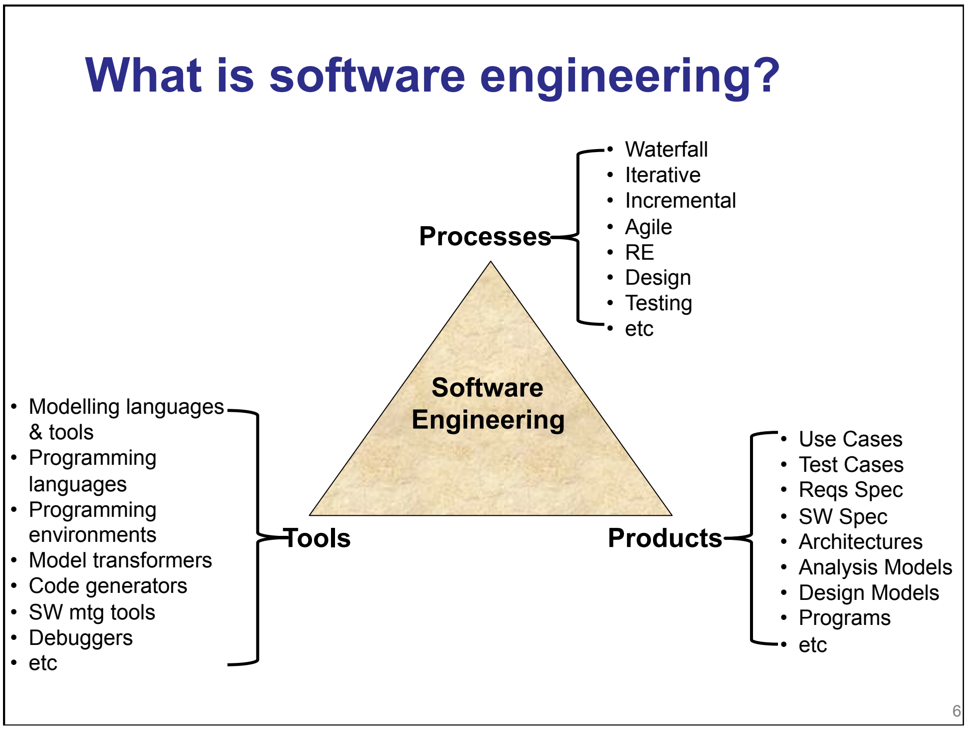
The Importance of Software Engineering

- The economies of ALL developed nations are dependent on software
- More and more systems are software controlled
- Software engineering expenditure represents a significant fraction of GNP in all developed countries



What is software engineering?

- Software engineering is an engineering discipline which is concerned with ALL aspects of software production
- It involves theories, methods, processes, and tools for professional software development
- Software engineers should adopt a systematic and organised approach to their work and use appropriate tools, processes and techniques, depending on the problem to be solved, the development constraints and the resources available



What is software engineering?

Software Engineering

Processes

- Waterfall
- Iterative
- Incremental
- Agile
- RE
- Design
- Testing
- etc

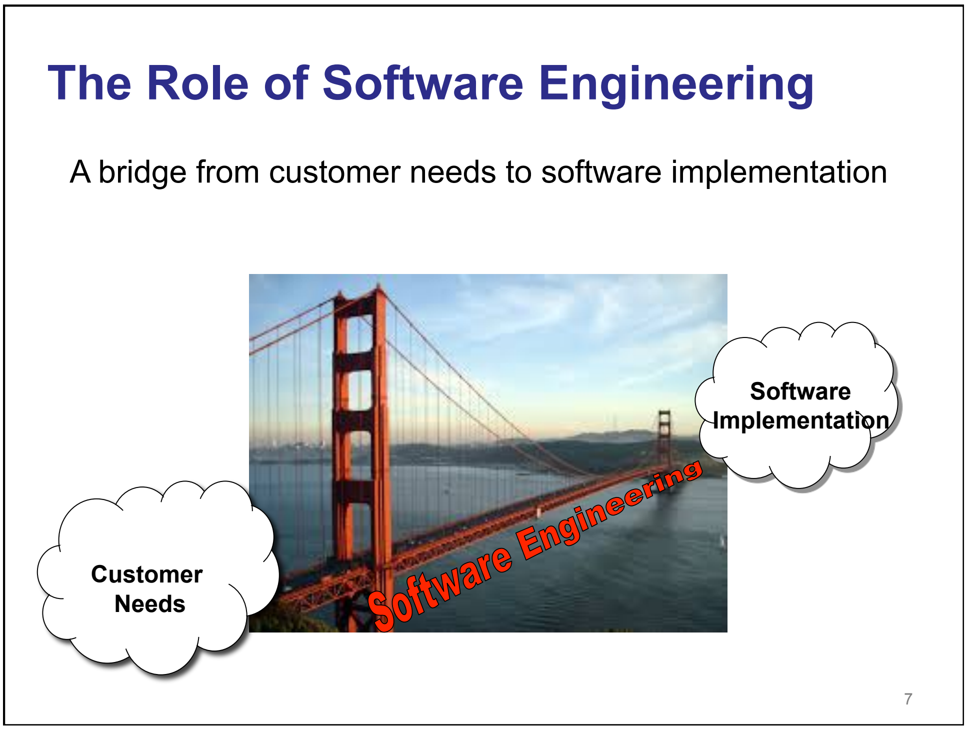
Tools

- Modelling languages & tools
- Programming languages
- Programming environments
- Model transformers
- Code generators
- SW mtg tools
- Debuggers
- etc

Products

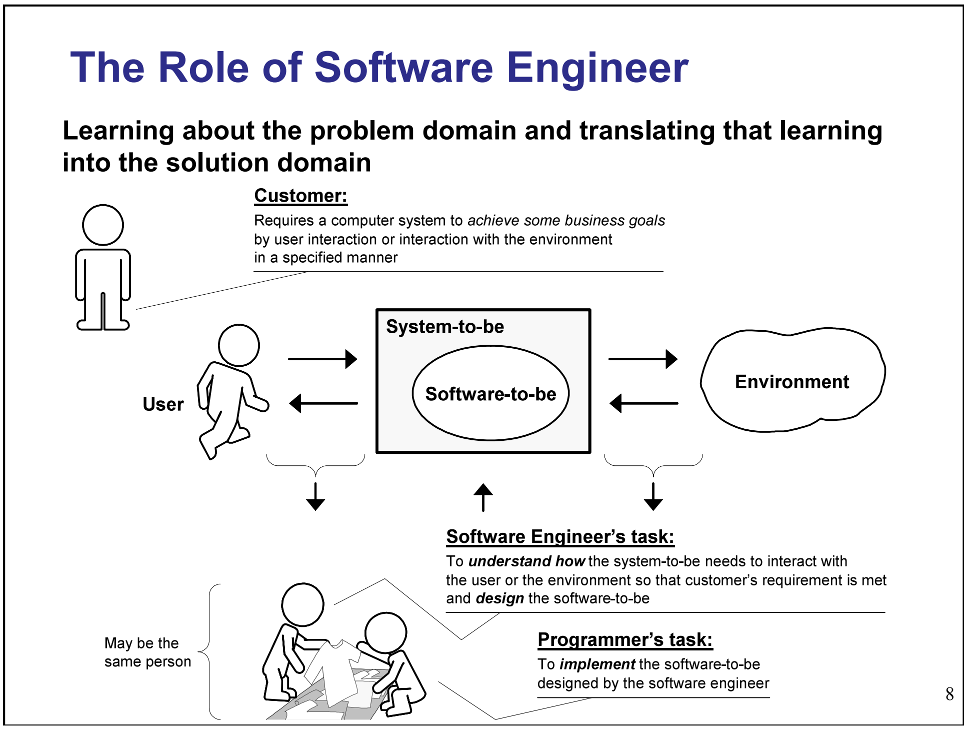
- Use Cases
- Test Cases
- Reqs Spec
- SW Spec
- Architectures
- Analysis Models
- Design Models
- Programs
- etc

6



The Role of Software Engineering

A bridge from customer needs to software implementation



The Role of Software Engineer

Learning about the problem domain and translating that learning into the solution domain

Customer:

Requires a computer system to *achieve some business goals* by user interaction or interaction with the environment in a specified manner

Software Engineer's task:

To ***understand how*** the system-to-be needs to interact with the user or the environment so that customer's requirement is met and ***design*** the software-to-be

Programmer's task:

To ***implement*** the software-to-be designed by the software engineer

What is a software process?

- A set of activities that help software engineers to understand, design and implement the software-to-be
- Generic activities in all software processes are:
  - Specification - what the system should do and its development constraints
  - Development - production of the software system
  - Validation - checking that the software is what the customer wants
  - Evolution - changing the software in response to changing demands

Why do we need software processes?

Traits of successful software engineering projects:

- The existence of a strong architectural vision
- The application of a well-managed, iterative and incremental development process

-- Grady Booch

Fundamental Assumption

- Good software processes lead to good software
- Good processes reduce risk

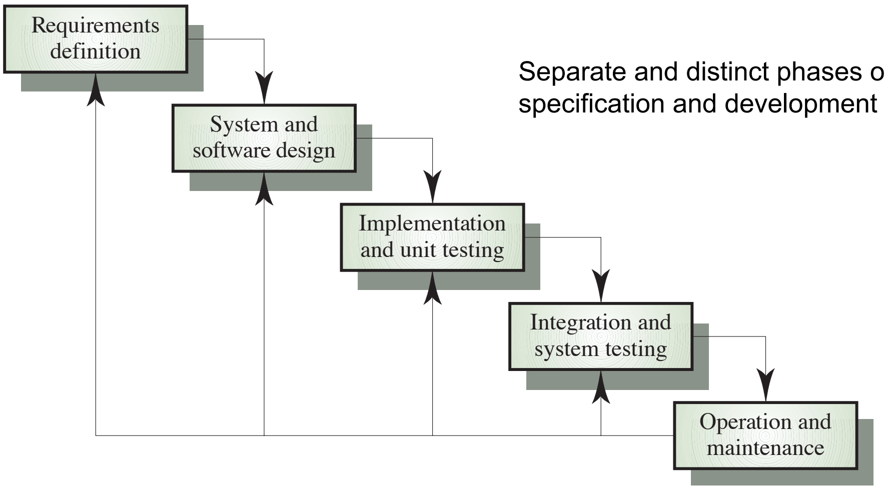
Two Generic Types of Software Process

- The macro process
  - Serves as the controlling framework for the entire software development lifecycle
  - Covers major development activities
  - Measured on the scale of weeks to months of team effort
- The micro process
  - Serves as the controlling framework for individual development activities
  - Measured on the scale of days of individual developers or a small team
  - Controlled by the macro process

Software Process Models

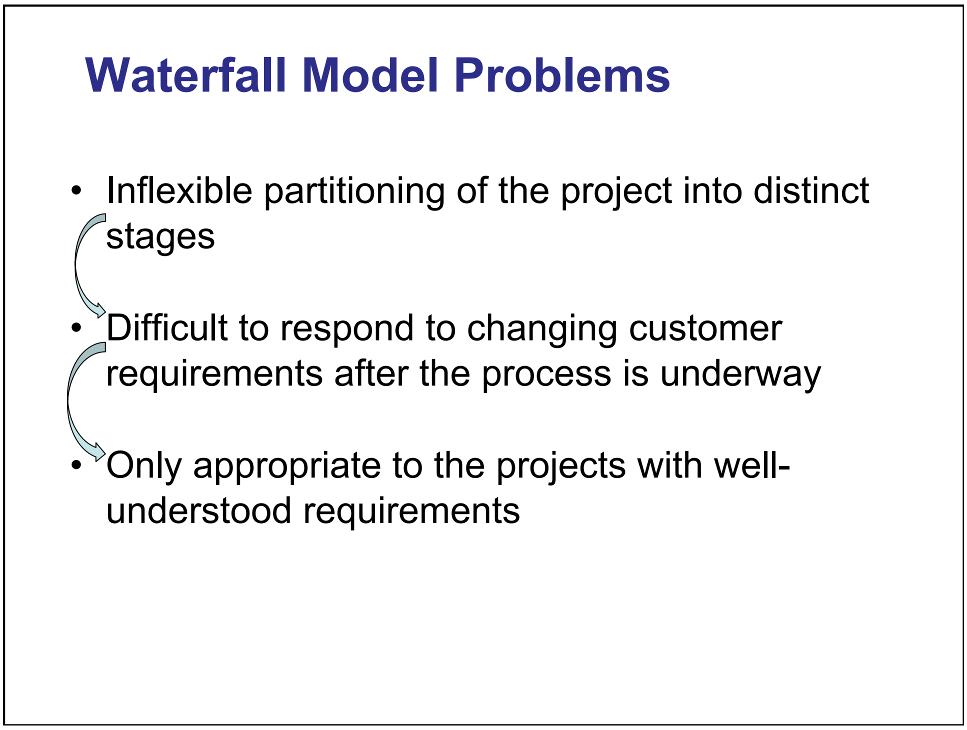
- A software process model is an abstract representation of a software process which is well-defined, predictable and repeatable
- Macro process models
  - Waterfall (traditional)
  - Iterative (Process Iteration)
  - Incremental
  - Spiral
  - Agile
  - Model driven development (MDD)
- Micro process models
  - Requirements engineering process (analysis process)
  - Design process
  - Debugging process
  - Testing process
  - Evaluation process

Waterfall Model


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

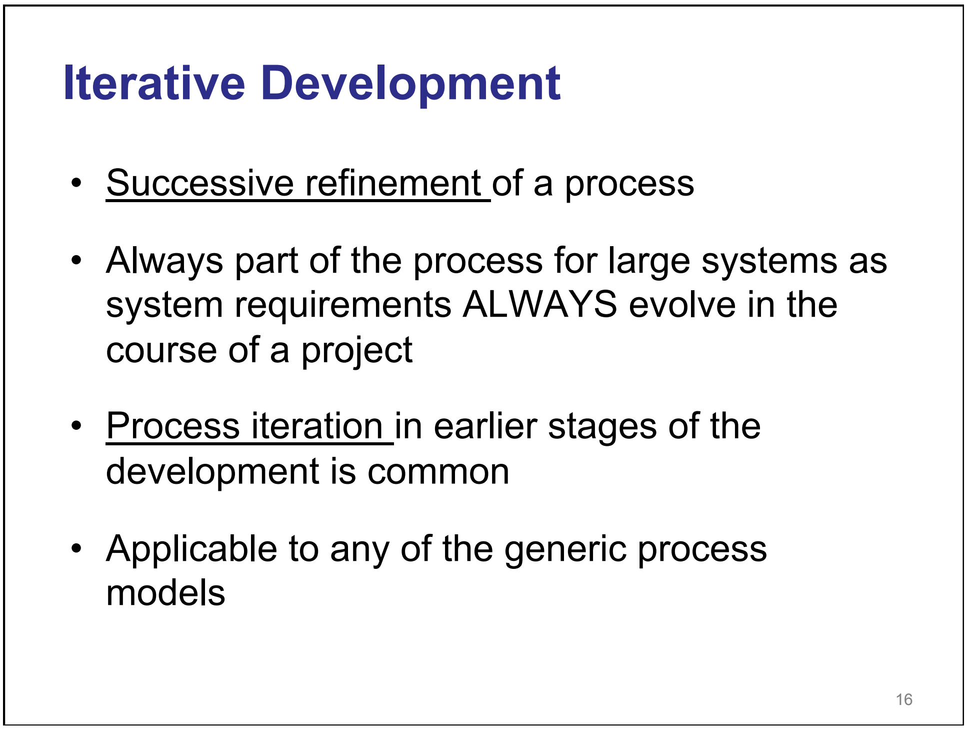
Separate and distinct phases of specification and development

From: Ian Summerville, Software Engineering, 6th Ed. Addison-Wesley, 2000.



Waterfall Model Problems

- Inflexible partitioning of the project into distinct stages
- Difficult to respond to changing customer requirements after the process is underway
- Only appropriate to the projects with well-understood requirements



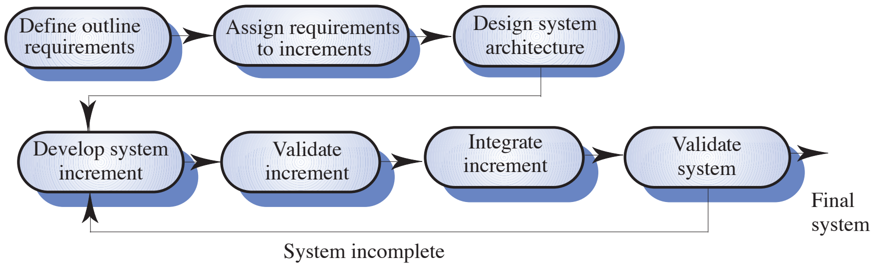
Iterative Development

- Successive refinement of a process
- Always part of the process for large systems as system requirements ALWAYS evolve in the course of a project
- Process iteration in earlier stages of the development is common
- Applicable to any of the generic process models

Incremental Development

- Decompose the development and delivery into increments with each increment delivering part of the required functionality
- Prioritise user requirements and include the highest priority requirements in early increments
- Frozen the requirements once the development of an increment is started
- Continue to evolve the requirements for later increments

Incremental Development Model



From: Ian Summerville, Software Engineering, 6th Ed. Addison-Wesley, 2000.

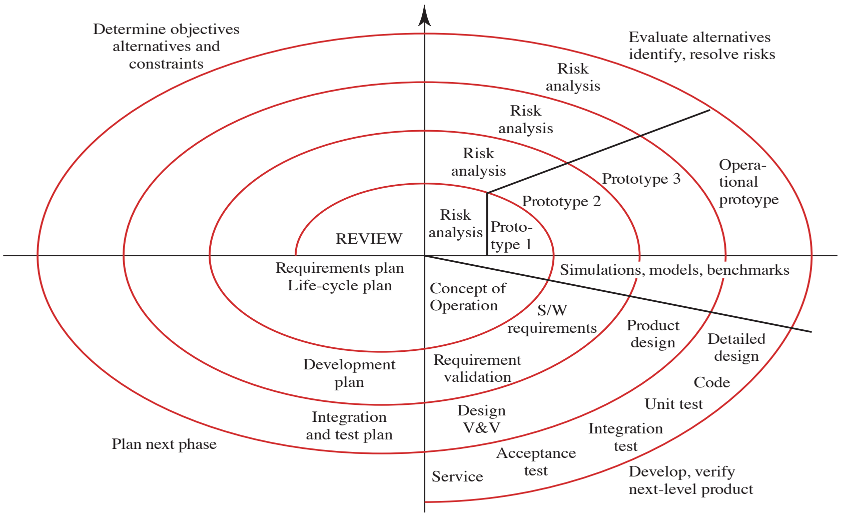
Incremental Development Advantages

- Customer value can be delivered with each increment so system functionality is available earlier
- Early increments act as a prototype to help elicit requirements for later increments
- Lower risk of overall project failure
- The highest priority system services tend to receive the most testing

Spiral Development

- Process represented as a spiral
- Each loop in the spiral represents a phase in the process.
- No fixed phases and loops in the spiral are chosen depending on what is required
- Risks explicitly assessed and resolved throughout the process

Spiral Model



From: Ian Summerville, Software Engineering, 6th Ed. Addison-Wesley, 2000.

Spiral Process Activities

- Objective setting
  - Specific objectives for the phase are identified
- Risk assessment and reduction
  - Risks are assessed and activities put in place to reduce the key risks
- Development and validation
  - A development model for the system is chosen which can be any of the generic models
- Planning
  - The project is reviewed and the next phase of the spiral is planned

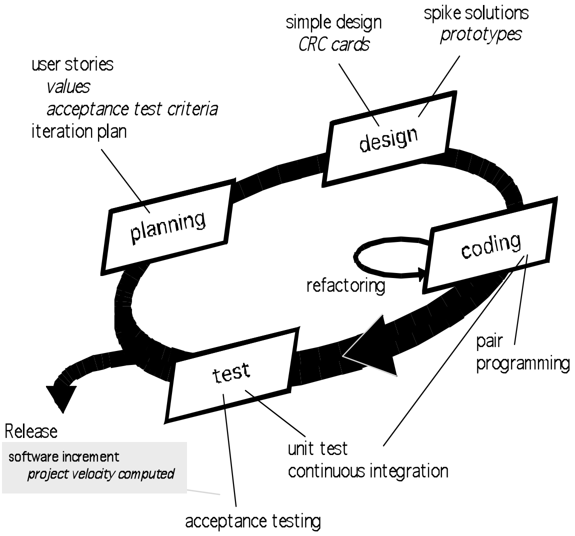
Agile Processes

- Extreme Programming (XP)
  - based on Test-Driven Development (TDD).
- Feature-Driven Development (FDD)
  - integrates ideas from plan-driven processes.
- SCRUM
- Crystal family
- Agile Modelling
- Dynamic Systems Development Method (DSDM)

Extreme Programming (XP)

- Based on the development and delivery of very small increments of functionality
- Relies on constant code improvement, user involvement in the development team and pair programming

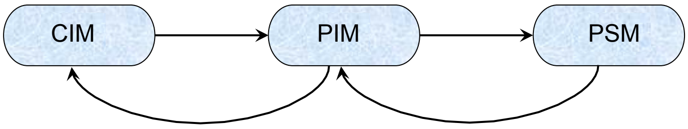
XP Process Model



Model Driven Development (MDD)

- View software development as a model building and model transformation activity
- Separate different concerns in different models:
  - CIM: Computation Independent Model
  - PIM: Platform Independent Model
  - PSM: Platform Specific Model
- Models and meta-models are primary software artefacts
- UML is the main language for model development
- Supported by automated model transformation & code generation technologies

MDD Process Model



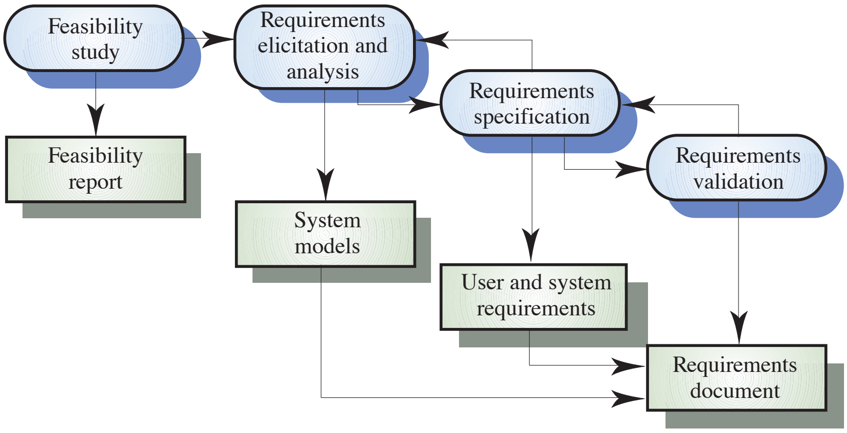
MDD Advantages

- Models are higher-level abstractions than programs and closer to the problem domain, thus easy to understand and share
- Models are independent of programming languages and thus easy to change or move between different implementation platforms

Requirements Engineering Process

- Also called analysis process
- The process establishes what services are required and the constraints on the system's operation and development
- Phases:
  - Feasibility study
  - Requirements elicitation and analysis
  - Requirements specification
  - Requirements validation

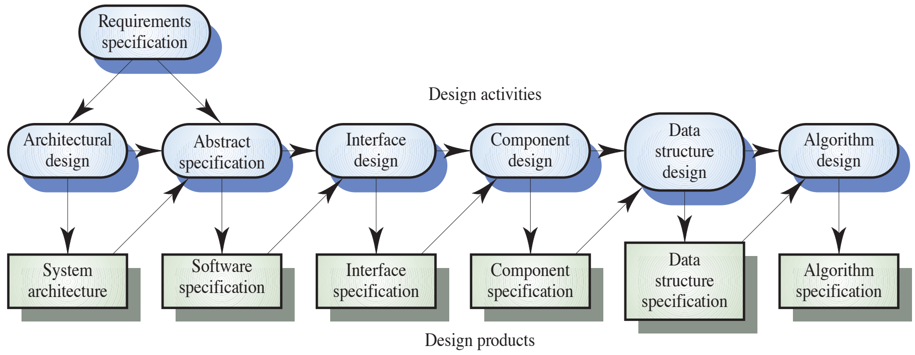
RE Process Model



Software Design Process

- Converts the system specification into an executable system
- The activities of design and implementation are closely related and may be inter-leaved
  - Architectural design
  - Abstract specification
  - Interface design
  - Component design
  - Data structure design
  - Algorithm design

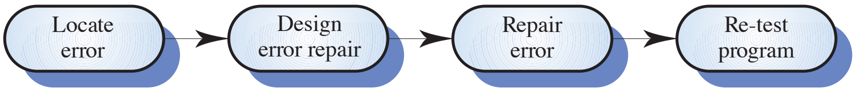
Design Process Model



Programming and Debugging

- Translating a design into a program and removing errors from that program
- Programming is a personal activity - there is no generic programming process
- Programmers carry out some program testing to discover faults in the program and remove these faults in the debugging process

Debugging Process Model



Locate error

Design error repair

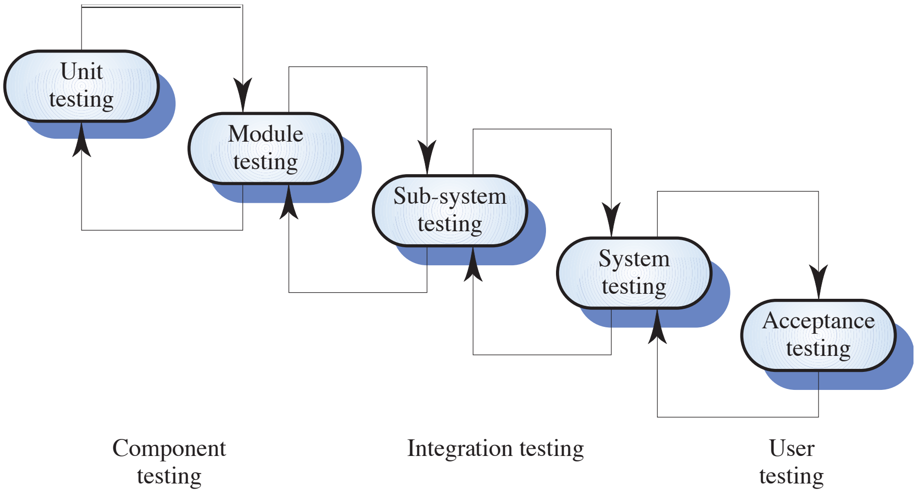
Repair error

Re-test program

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 Stages:
  - **Unit testing** - Individual components are tested
  - Module testing - Related collections of dependent components are tested
  - **Sub-system testing** - Modules are integrated into sub-systems and tested. The focus here should be on interface testing
  - **System testing** - Testing of the system as a whole. Testing of emergent properties
  - **Acceptance testing** -Testing with customer data to check that it is acceptable

Testing Process Model



Unit testing

Module testing

Sub-system testing

System testing

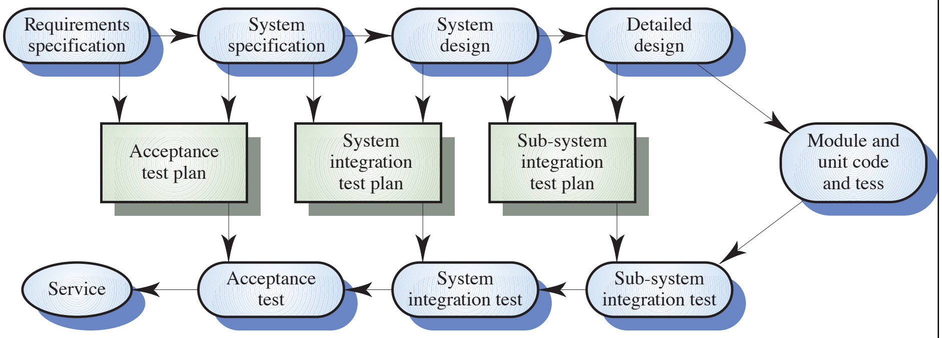
Acceptance testing

Component testing

Integration testing

User testing

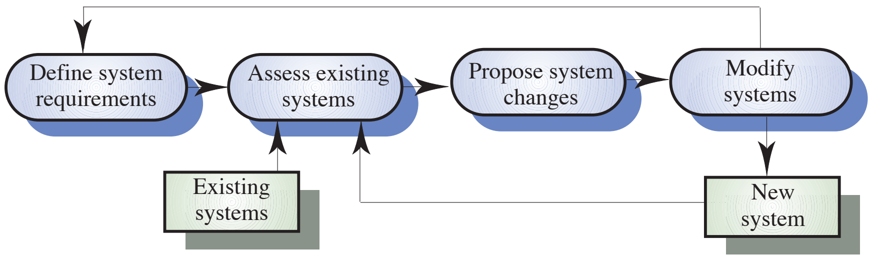
Test-driven development


Requirements specification
System specification
System design
Detailed design
Acceptance test plan
System integration test plan
Sub-system integration test plan
Module and unit code and tess
Service
Acceptance test
System integration test
Sub-system integration test

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

System Evolution Process Model



Define system requirements

Assess existing systems

Propose system changes

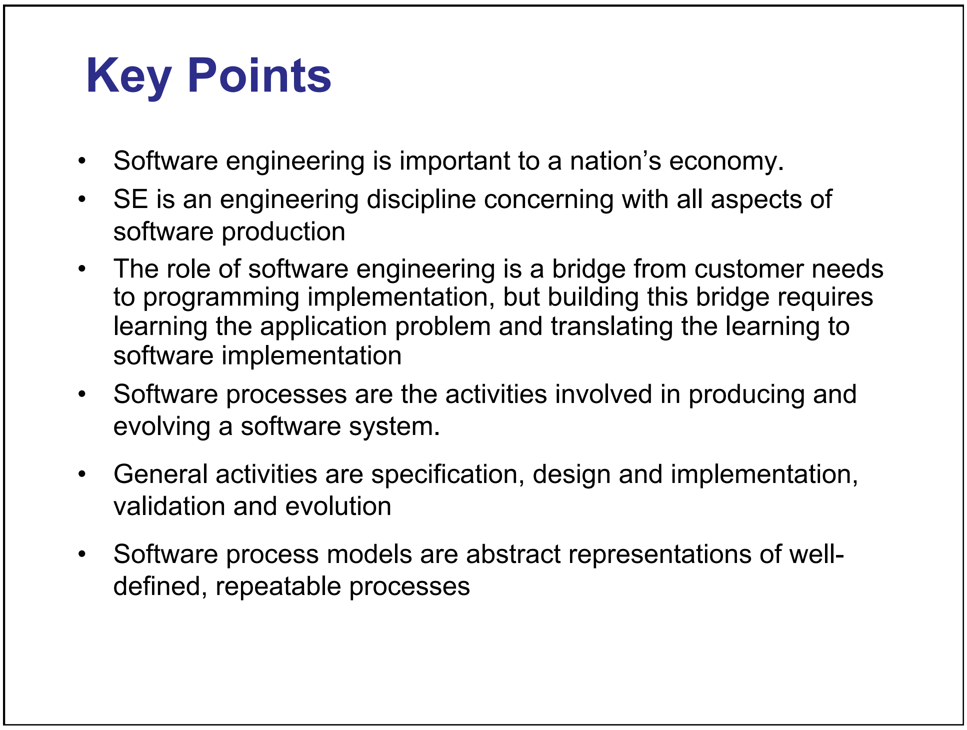
Modify systems

Existing systems

New system

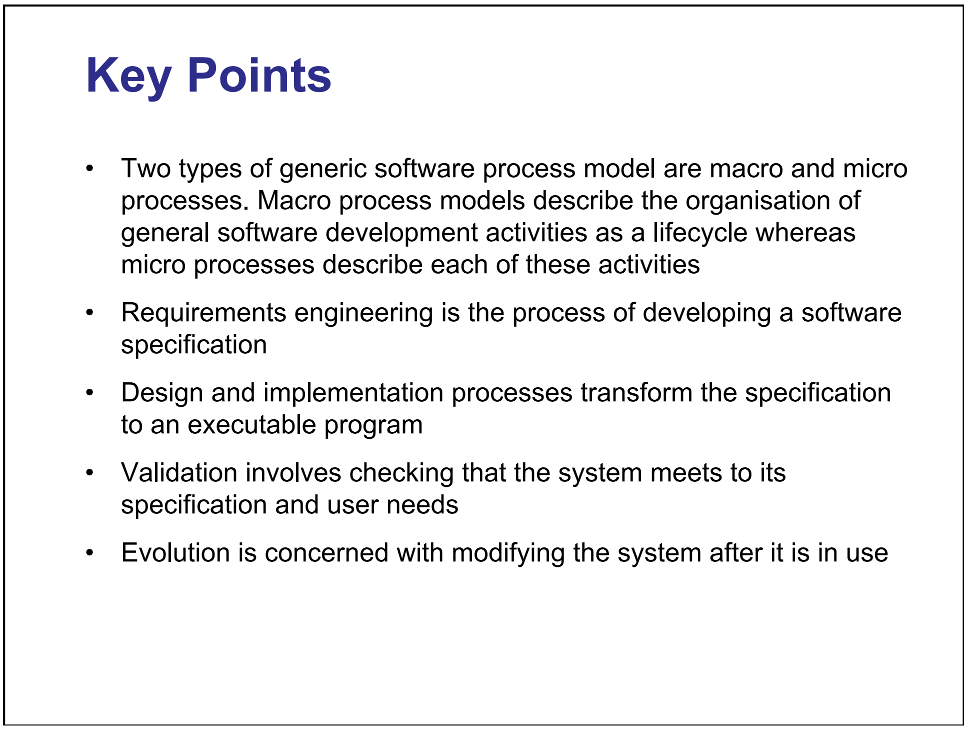
Automated Process Support (CASE)

- Computer-aided software engineering (CASE) is software to support software development and evolution processes
- Activity automation
  - Graphical editors for system model development (UML)
  - Data dictionary to manage design entities
  - Graphical UI builder for user interface construction
  - Debuggers to support program fault finding
  - Automated translators to generate new versions of a program



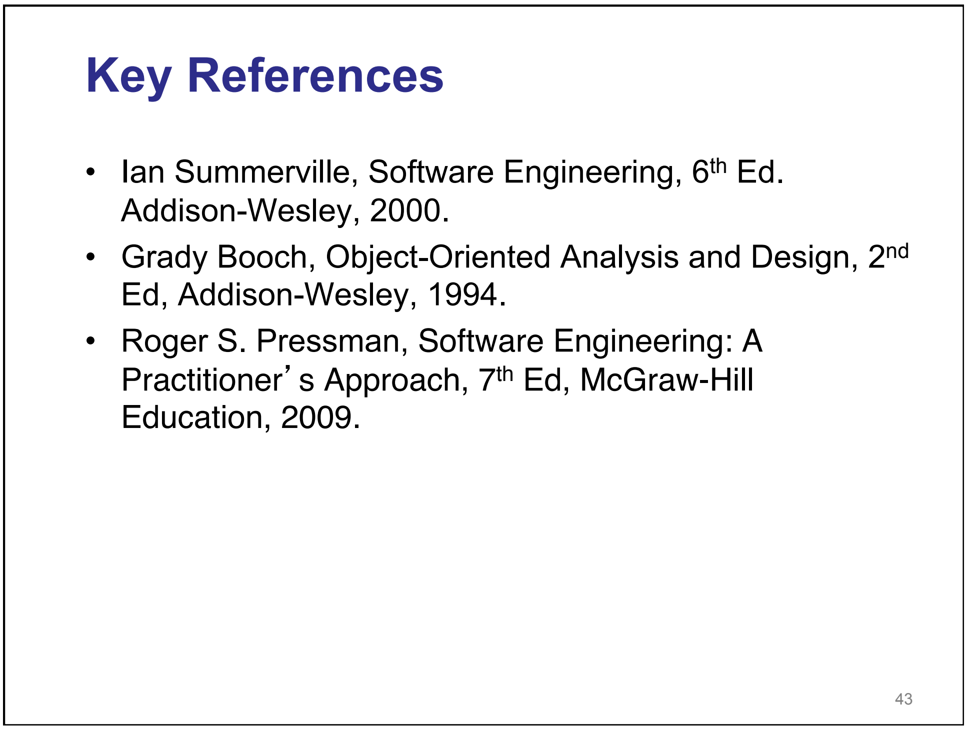
Key Points

- Software engineering is important to a nation’s economy.
- SE is an engineering discipline concerning with all aspects of software production
- The role of software engineering is a bridge from customer needs to programming implementation, but building this bridge requires learning the application problem and translating the learning to software implementation
- Software processes are the activities involved in producing and evolving a software system.
- General activities are specification, design and implementation, validation and evolution
- Software process models are abstract representations of well-defined, repeatable processes



Key Points

- Two types of generic software process model are macro and micro processes. Macro process models describe the organisation of general software development activities as a lifecycle whereas micro processes describe each of these activities
- Requirements engineering is the process of developing a software specification
- Design and implementation processes transform the specification to an executable program
- Validation involves checking that the system meets to its specification and user needs
- Evolution is concerned with modifying the system after it is in use



Key References

- Ian Summerville, Software Engineering, 6th Ed. Addison-Wesley, 2000.
- Grady Booch, Object-Oriented Analysis and Design, 2nd Ed, Addison-Wesley, 1994.
- Roger S. Pressman, Software Engineering: A Practitioner’ s Approach, 7th Ed, McGraw-Hill Education, 2009.