Faculty of ECMS / School of Computer Science

Software Engineering & Project
Software Process Models

Software Process Models

Lecture 5

Chapters 4 (2 in Edition 9)
in the course text book

Outline

- Fundamental software processes and activities
- Generic models of software processes
- Software process models in practice

What is a Software Process?

- A software process is a structured set of activities that produce or maintain a software product

- Fundamental activities in software development:
  - Specification – defining what the system should do
  - Design and implementation – defining the organization of the system and implementing the system
  - Validation – checking that it does what the customer wants
  - Evolution – changing the system in response to new customer needs.

Software Process Models

- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective
- Useful as a roadmap to guide software teams

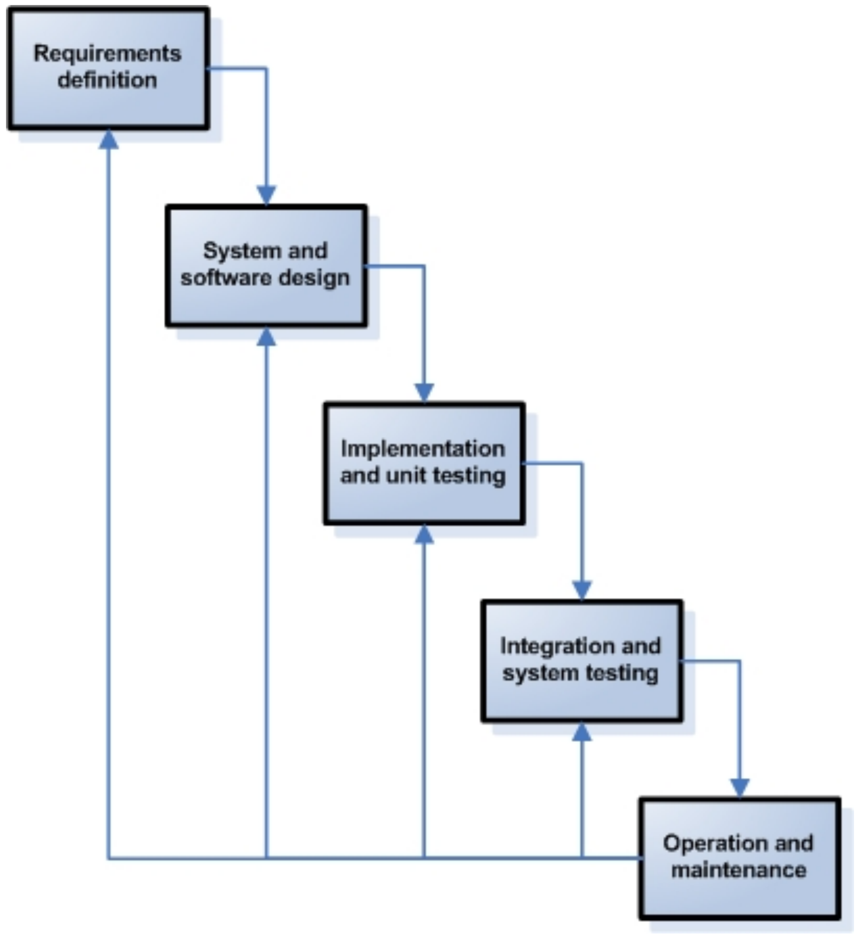
Generic Software Process Models

- Most software process models are based on one of the following four generic models or paradigms of software development
    - The Waterfall Model
    - Evolutionary Development
    - Component-based Development (Reuse oriented)
    - Agile Methods
- Each model has strengths and weaknesses. No model can be one-fit-all
- In practice, most large systems are developed using a process that incorporates elements from all of these models

The Waterfall Model

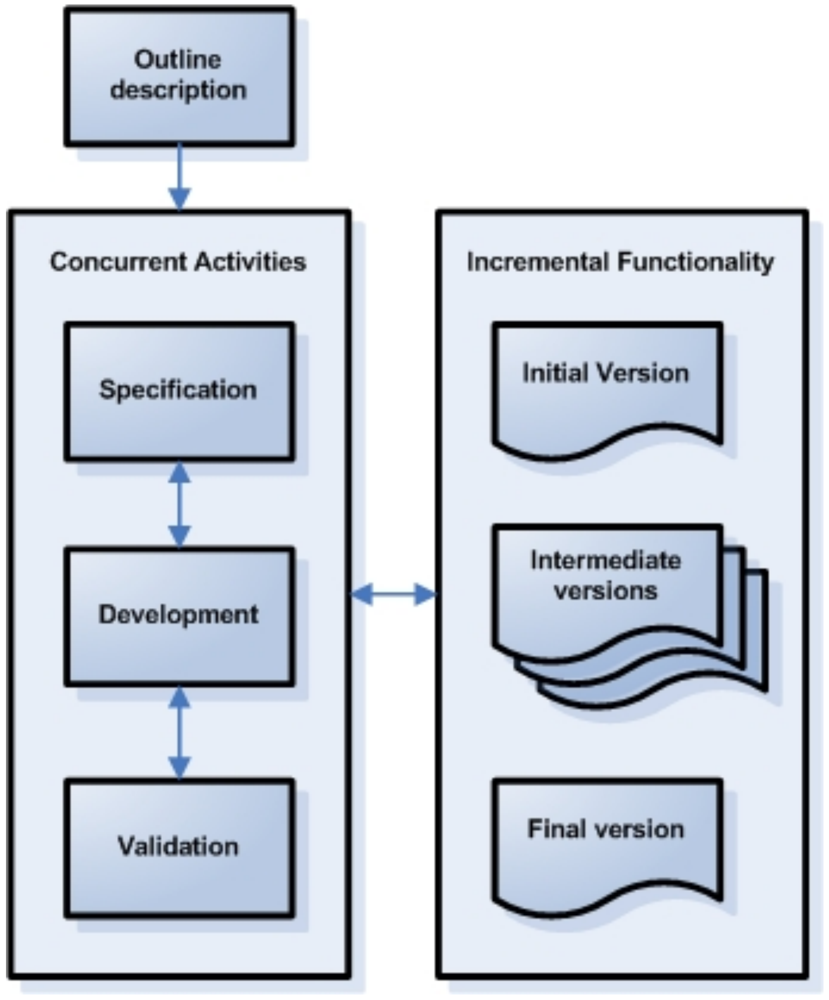
- Strengths
  - Aligns to the systems engineering process
  - Complete set of documentation
- Weaknesses
  - Inflexible to changing requirements
  - Late discovery of technical problems

- This model is suitable when the requirements are well-understood and changes will be fairly limited during the design process.


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

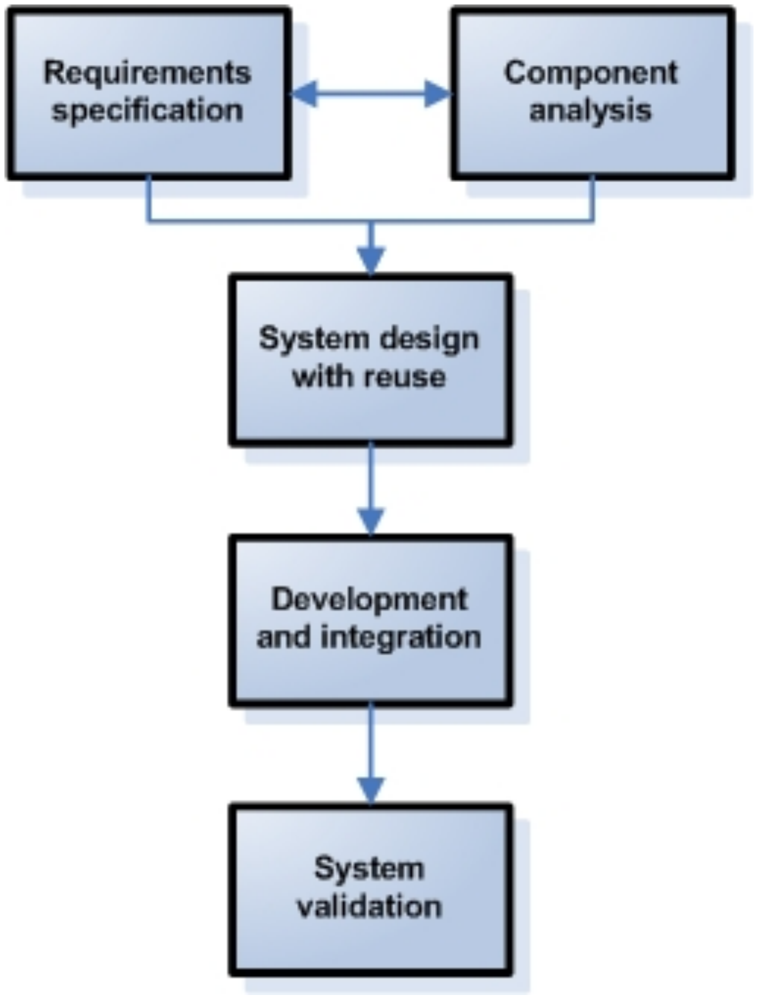
Evolutionary Development

- Strengths
  - Effectively manages evolving requirements
  - Rapid delivery of useful software to clients
  - Identifies and resolves technical risk early
- Weaknesses
  - Reduced visibility and control of activities, hard for managers to measure the progress
  - May lead to poorly structured software due to regular changes



Component-based Development

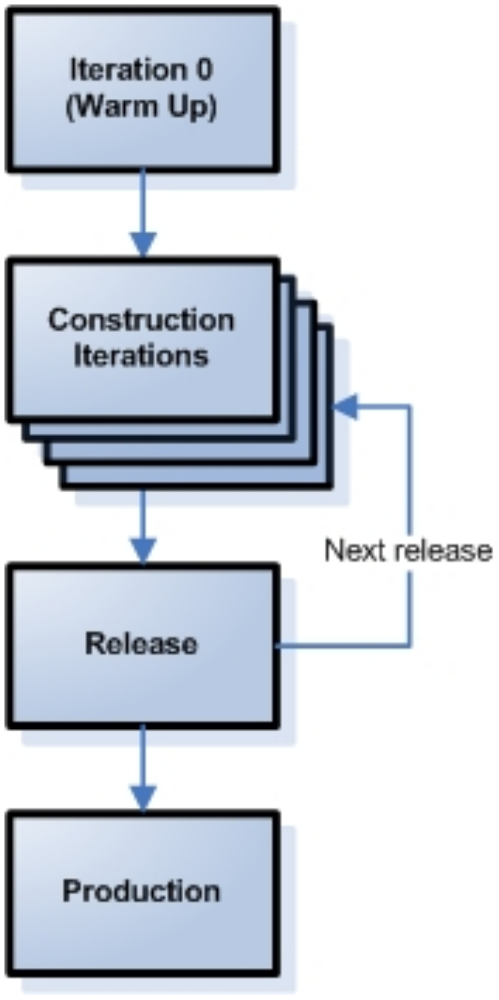
- Based on systematic reuse where systems are integrated from existing components or COTS (Commercial-off-the-shelf) systems.
- Reuse is now the standard approach for building many types of business system
- E.g., Web services, .Net, J2EE



Agile Methods

- Strengths
  - Extremely responsive to change
  - Working software produced very early
- Weaknesses
  - Highly dependent on customer involvement
  - Requires a high performing team

- Daily Scrum
  - About 15 mins
  - What have you done since yesterday?
  - What are you planning to do today?
  - Any impediments?



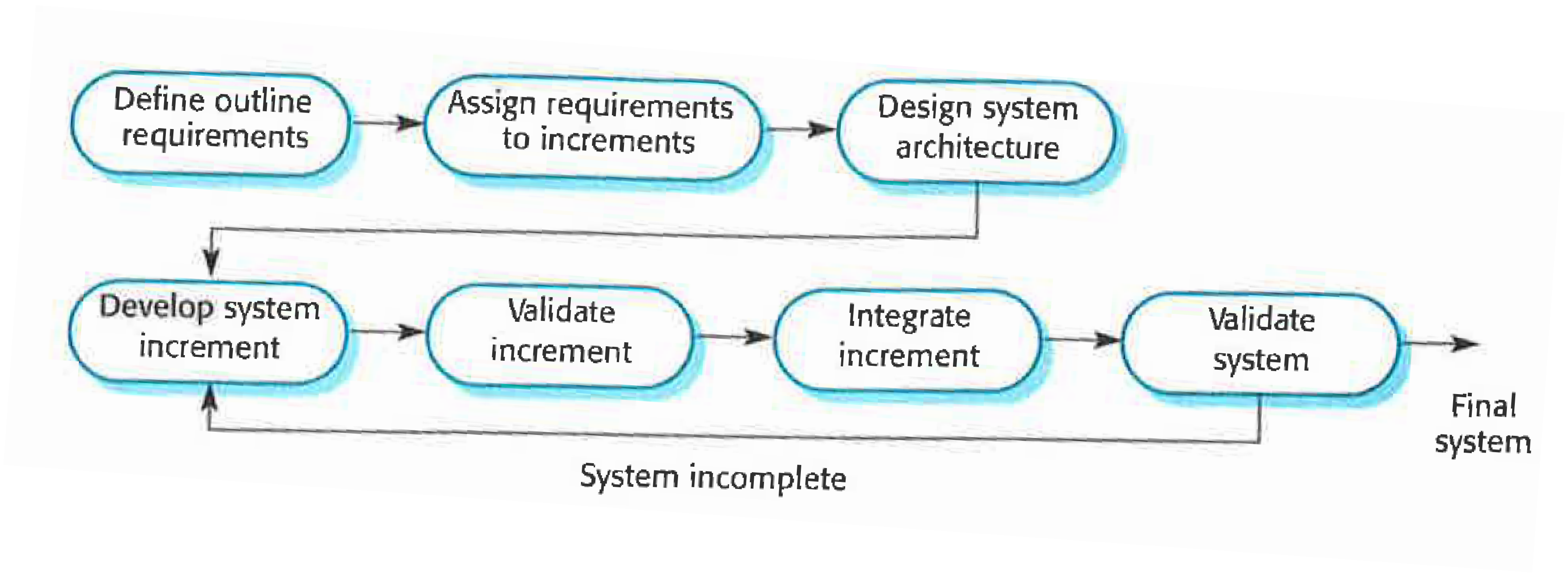
Selecting a Process Model

	Waterfall Model	Evolutionary Development	Component Development	Agile Methods
Requirements Volatility	Low	Medium	Low	High
Project Size	Large	Medium	Any	Small
Customer Involvement	Low	Medium	Low	High
Technical Risk	Low	High	Medium	High
Release Schedule	Long	Medium	Short	Very Short

Software Process Models in Practice

- In practice, software organisations may combine aspects of generic process models to meet the specific needs of their projects
- In many cases the choice of process model is also constrained by a range of other factors:
  - Contractual or regulatory requirements
  - Experience and familiarity with the process model
  - Process models used by related projects
  - Team experience and abilities

Incremental Delivery



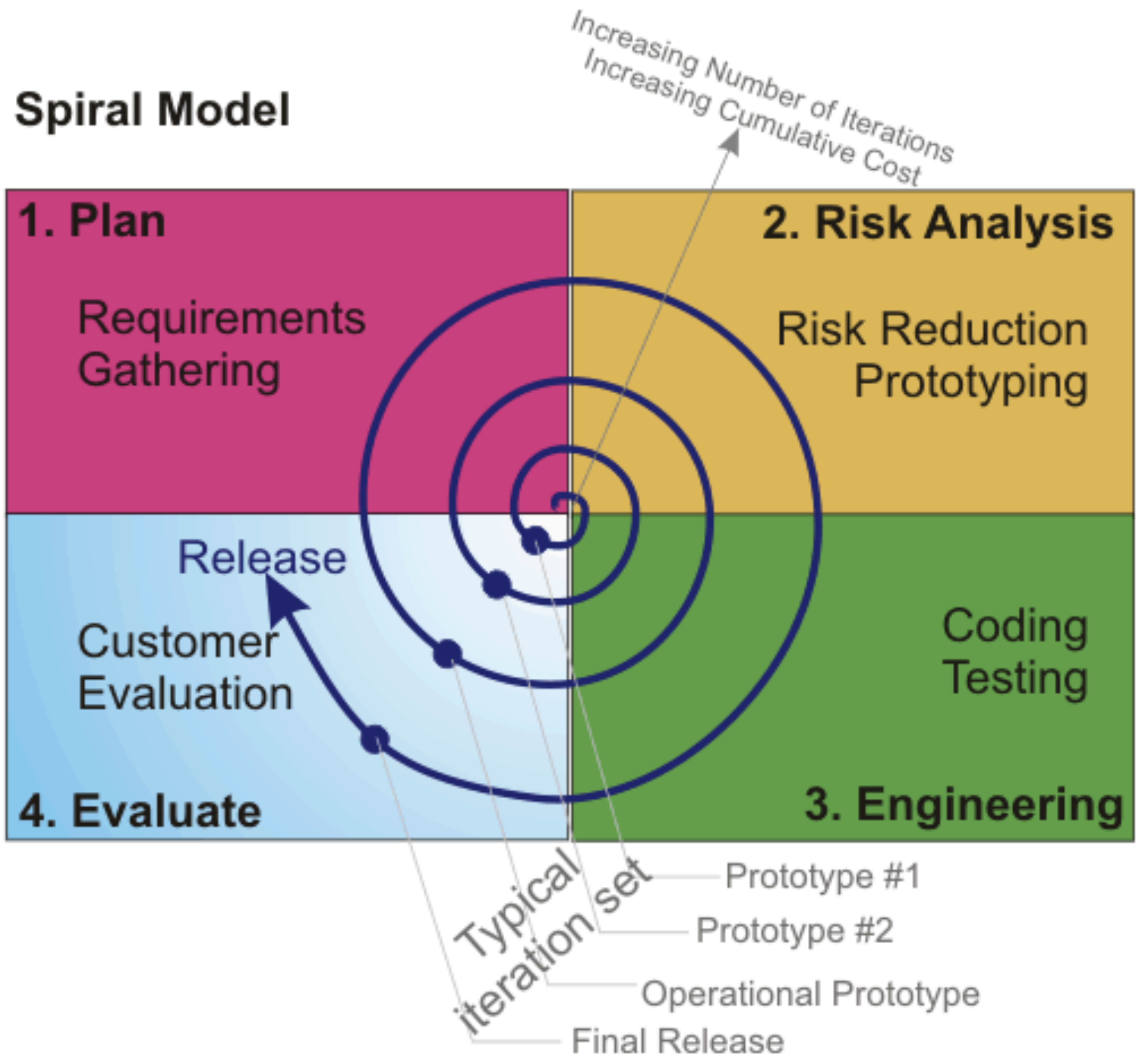
Incremental Delivery Advantages

- 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

The Spiral Model

- Process is represented as a spiral rather than as a sequence of activities with backtracking
- Each loop in the spiral represents a phase in the process
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required
- Risks are explicitly assessed and resolved throughout the process

The Spiral Model (Cont.)



The Spiral Model (Cont.)

- Spiral model has been very influential in helping people think about iteration in software processes and introducing the risk-driven approach to development

- It is typically used for large mission critical projects – similar to the waterfall process model

Summary of Key Points

- A software process is the set of activities involved in producing and maintaining software
- All software processes include the fundamental activities of software specification, design, implementation, validation and evolution
- Software process models are abstract representations that describe the organisation of fundamental software process activities
- The choice of which process model(s) to use is dependent on the project objectives and context