Software Engineering

Dr. Meisam Nazariani

Email: m_nazariani@aut.ac.ir

February 2022

Outline

1. Introduction
2. The Nature of Software
3. Software Engineering
4. **The Software Process**
5. **Process Models**
6. Agile Development
    1. XP
    2. Scrum
7. DevOps
8. Requirement Engineering
9. Software Modeling
10. Design Concepts
11. Umbrella Activities
12. Case Studies

The Software Process and Process Models

- Software process
- Software process models
- Process activities
- Coping with change
- Process improvement

Objectives

Introduction:

- ✓ concepts of software processes and software process models
- ✓ three general software process models
- ✓ fundamental process activities of software requirements engineering, software development, testing, and evolution
- ✓ notion of software process improvement
- ✓ factors that affect software process quality

Software process

The software process (Quick look)

1.What is it?

When you work to build a product or system, it's important to go through a series of predictable steps and create a timely, high quality result.

The road map that you follow is called a "**software process.**"

2.Who does it?

Software engineers and **their managers** adapt the process to their needs and then follow it.

The software process (Quick look)

3.Why is it important?

Because it provides stability, control, and organization to an activity that can, if left uncontrolled, become quite chaotic. besides, a modern software engineering approach must be **agile**.

4. What is the work product?

From the point of view of a software engineer, the work products are the **programs**, **documents**, and **data** that are produced as a consequence of the activities and **tasks** defined by the process.

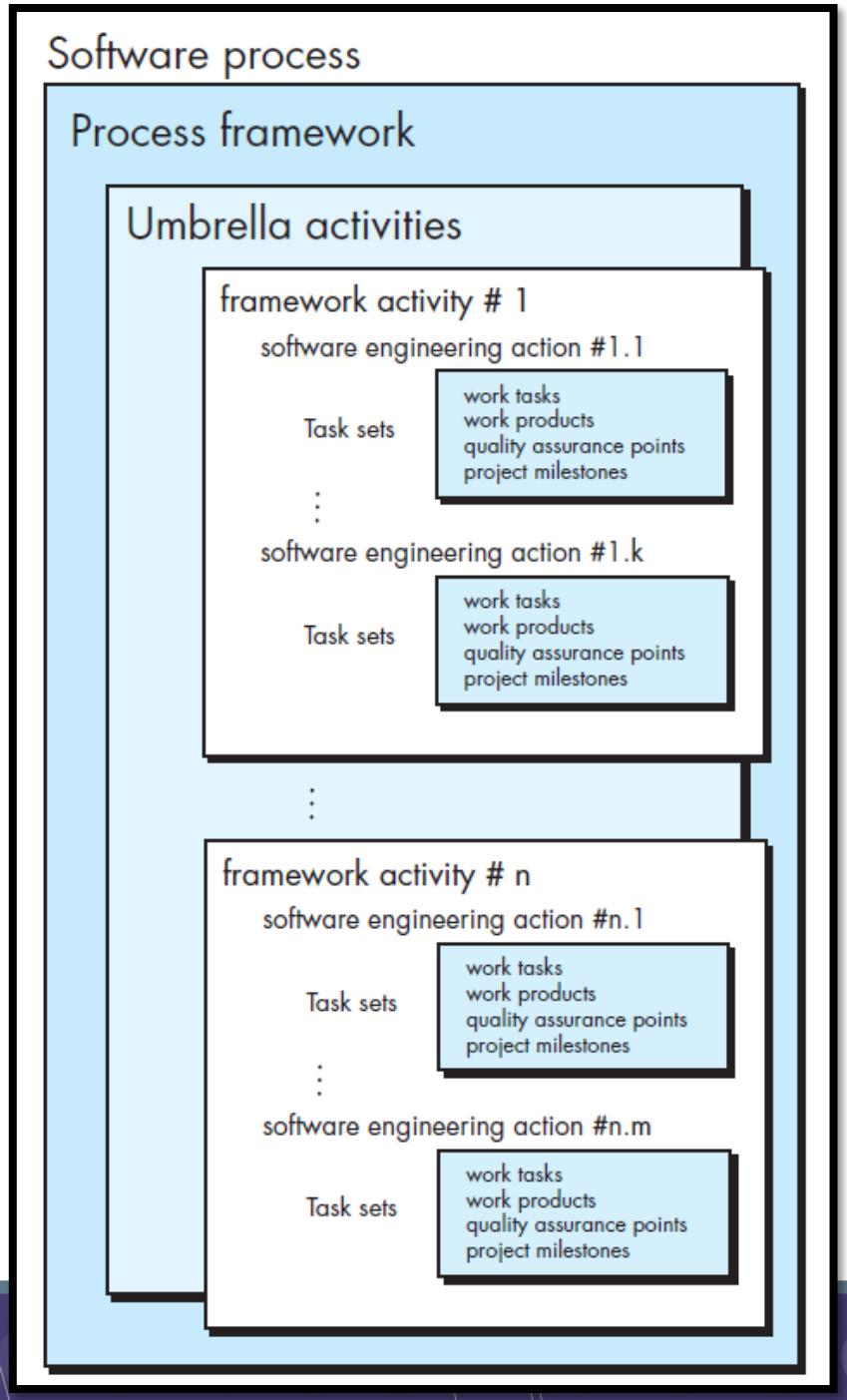
The software process

what exactly is a software process from a technical point of view?

- Software process is a framework for the activities, actions, and tasks that are required to build high quality software.

A software process framework

- Each framework activity is populated by a set of software engineering actions.
- Each software engineering action is defined by a task set that identifies the work tasks.



A generic process model

- A generic process framework for software engineering defines five framework activities:

1. Communication
2. Planning
3. Modeling
4. Construction
5. deployment

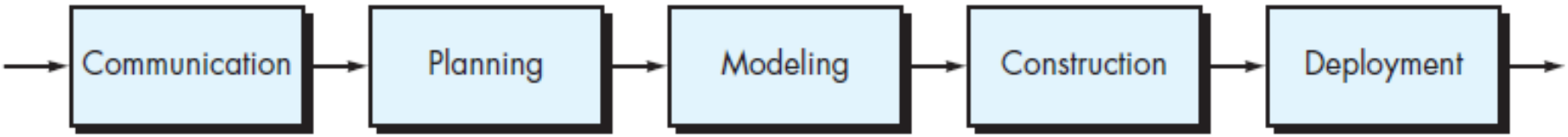
Process flow

❑ Process Flow:

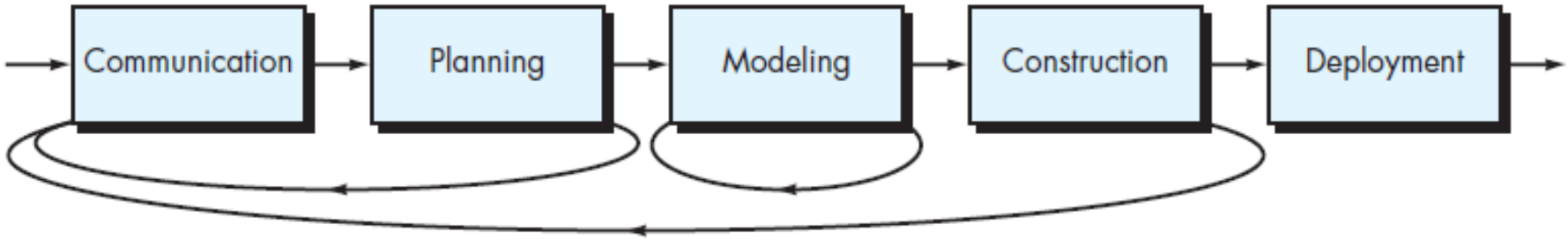
- one important aspect of the software process that describes how the framework activities and the actions and tasks that occur within each framework activity are organized with respect to sequence and time.

Types of process flows

1.Linear process flow

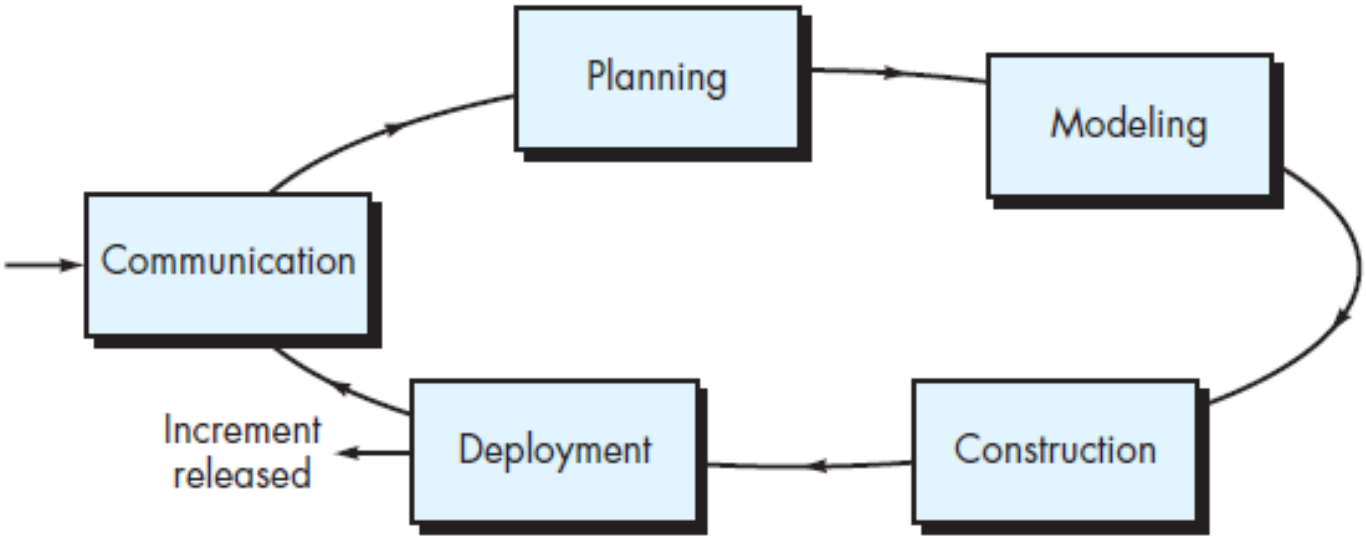


2.Iterative process flow



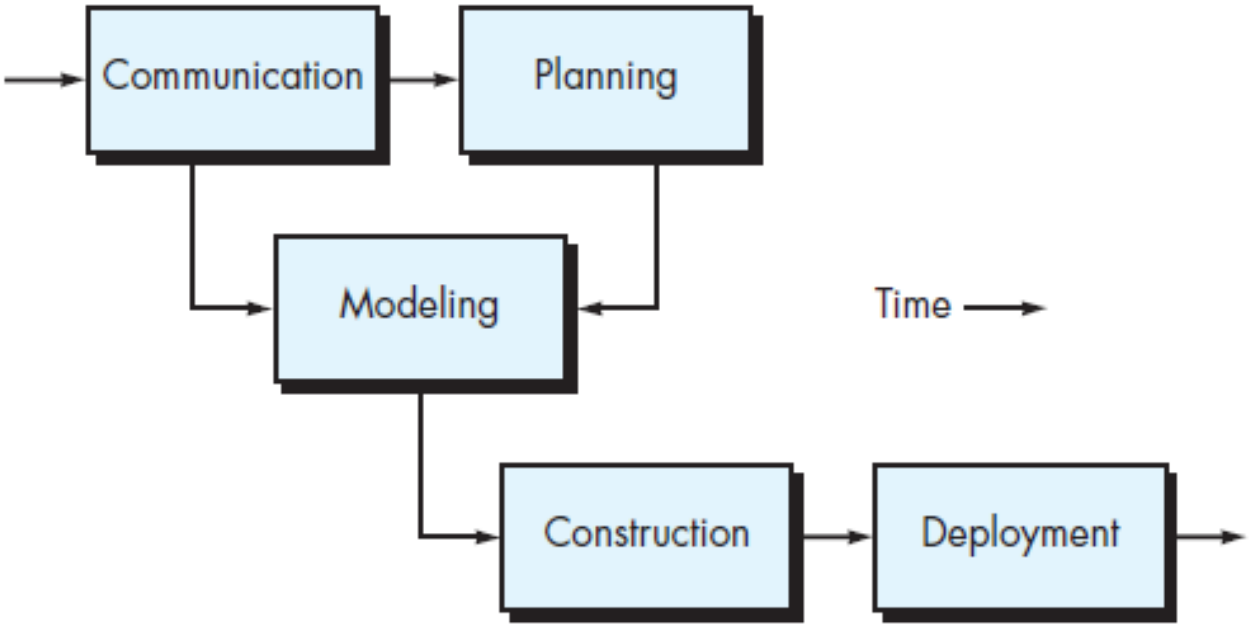
Types of process flows

3. Evolutionary process flow



Types of process flows

4. Parallel process flow



Process Patterns

- a process pattern provides you with a template a consistent method for describing problem solutions within the context of the software process.

- By combining patterns, a software team can solve problems and construct a process that best meets the needs of a project.

- Patterns can be defined at any level of abstraction. In some cases, a pattern might be used to describe a problem and solution associated with a complete process model .In other situations, patterns can be used to describe a problem and solution associated with a framework activity.

An example for process patterns

Ambler has proposed a template for describing a process pattern:

- **Pattern Name**
- **Forces**
- **Type**
  - Stage pattern
  - Task pattern
  - Phase pattern
- **Initial Context**
- **Problem**
- **Solution**
- **Resulting Context**
- **Related Patterns**
- **Known Uses**
- **Examples**

The software process

- **Software process:**

  a structured set of activities required to develop a software system.

- **4 fundamental activities:**

  - ✓ Specification
  - ✓ Design and implementation
  - ✓ Validation
  - ✓ Evolution

Software process descriptions

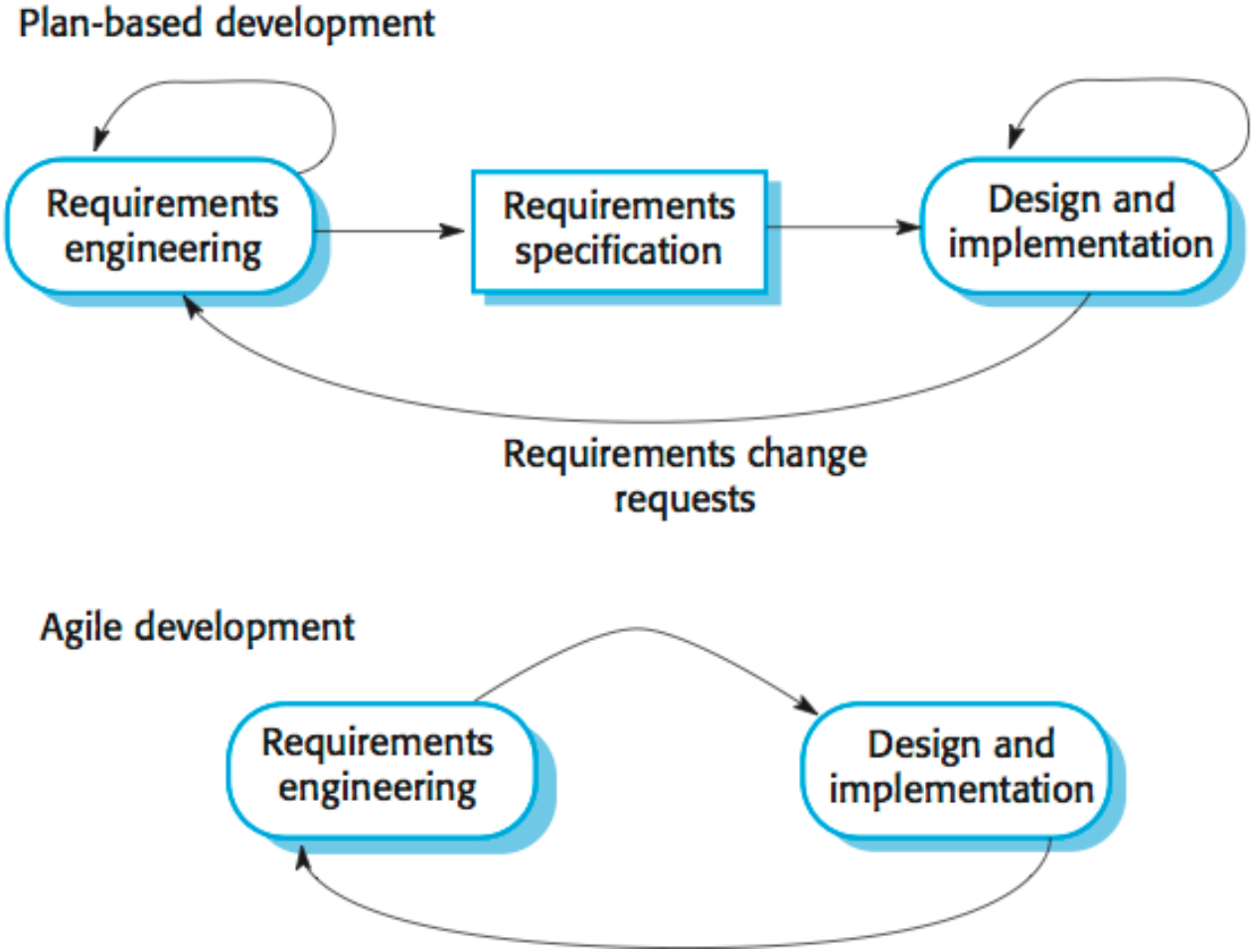
- **Process descriptions include:**
  - ✓ activities of processes
  - ✓ ordering of these activities
  - ✓ Products
  - ✓ Roles
  - ✓ Pre- and post-conditions

Plan-driven and Agile processes

- **Plan-driven processes**: all of the process activities are planned in advance ad progress is measured against this plan

- **Agile processes**: planning is incremental and it is easier to change the process to reflect changing customer requirements.

Plan-driven and Agile processes



Software process models

Software process models(Quick look)

1.What is it?

A process model provides a specific roadmap for software engineering work.

It defines the flow of all activities, actions and tasks, the degree of iteration, the work products, and the organization of the work that must be done.

Software process models

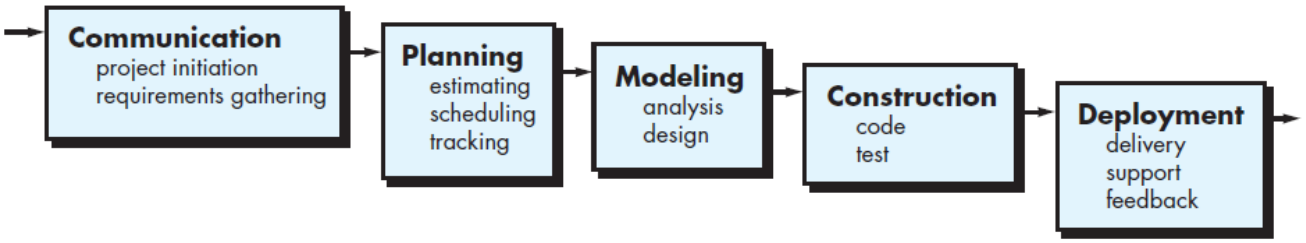
- Waterfall Process models
- Incremental Process Models
- Evolutionary Process Models

The waterfall model

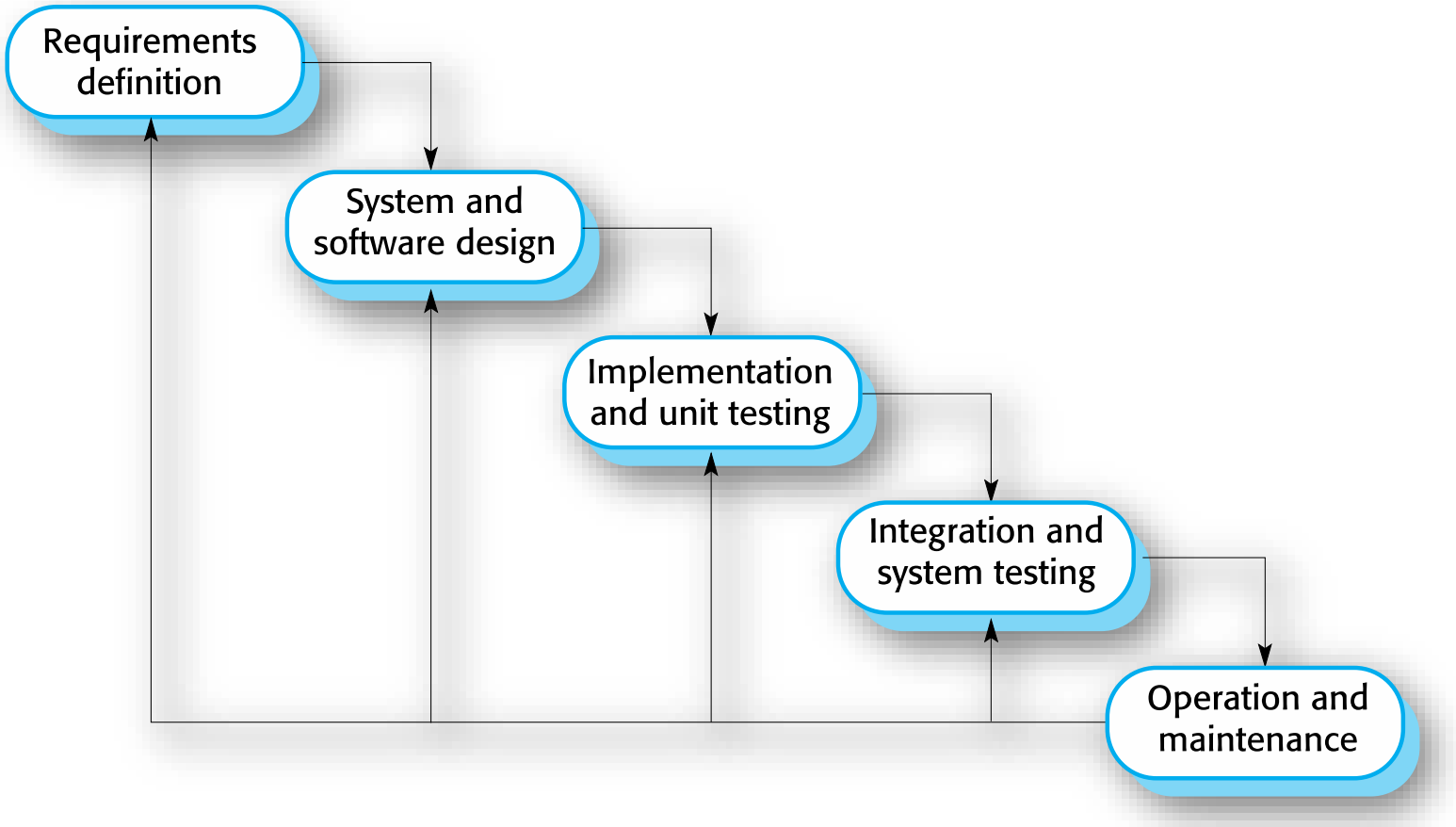
- The waterfall model suggests a systematic, sequential approach to software development that begins with customer specification of requirements and progresses through planning, modeling, construction, and deployment, culminating in ongoing support of the completed software.

- Requirements are very well documented, clear and fixed.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- The project is short.

- the oldest paradigm for software engineering

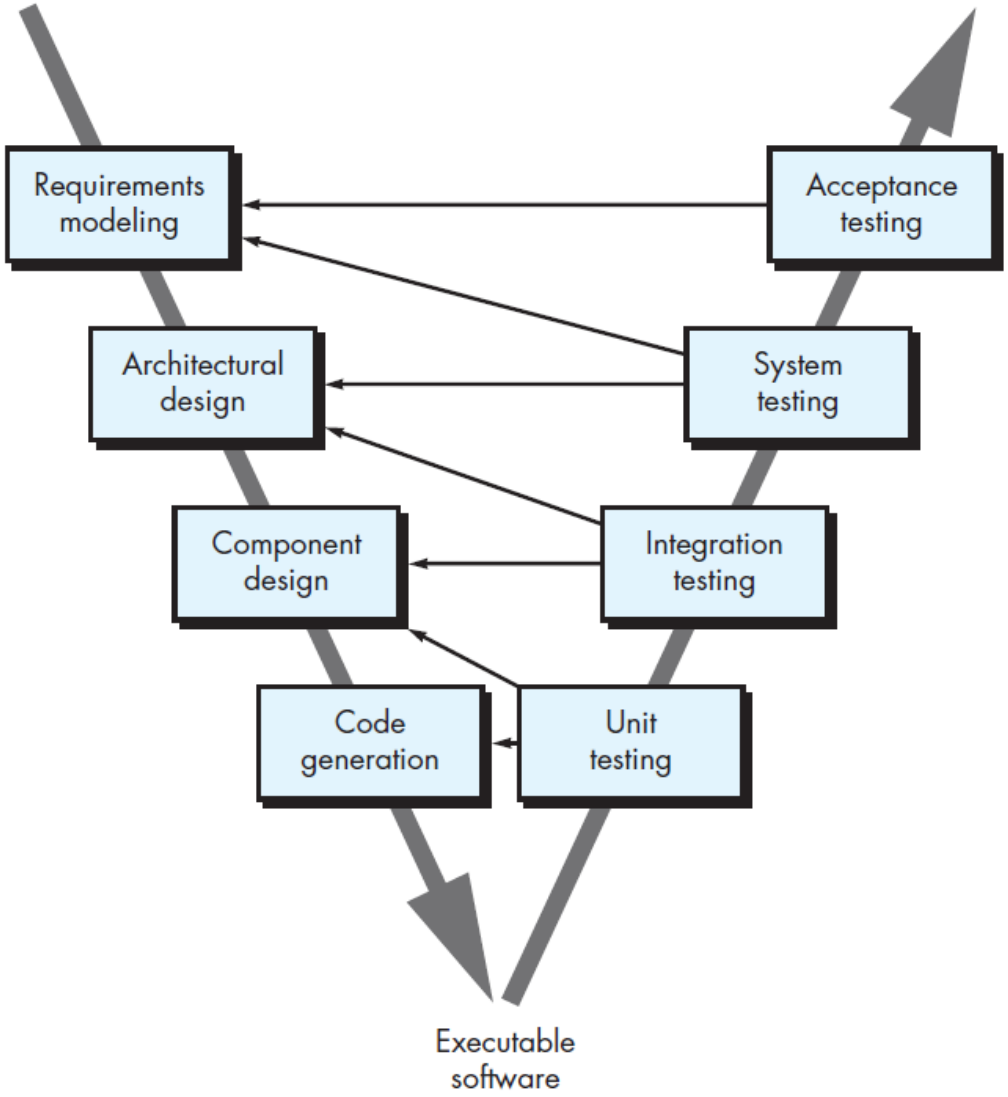


The waterfall model



The waterfall model

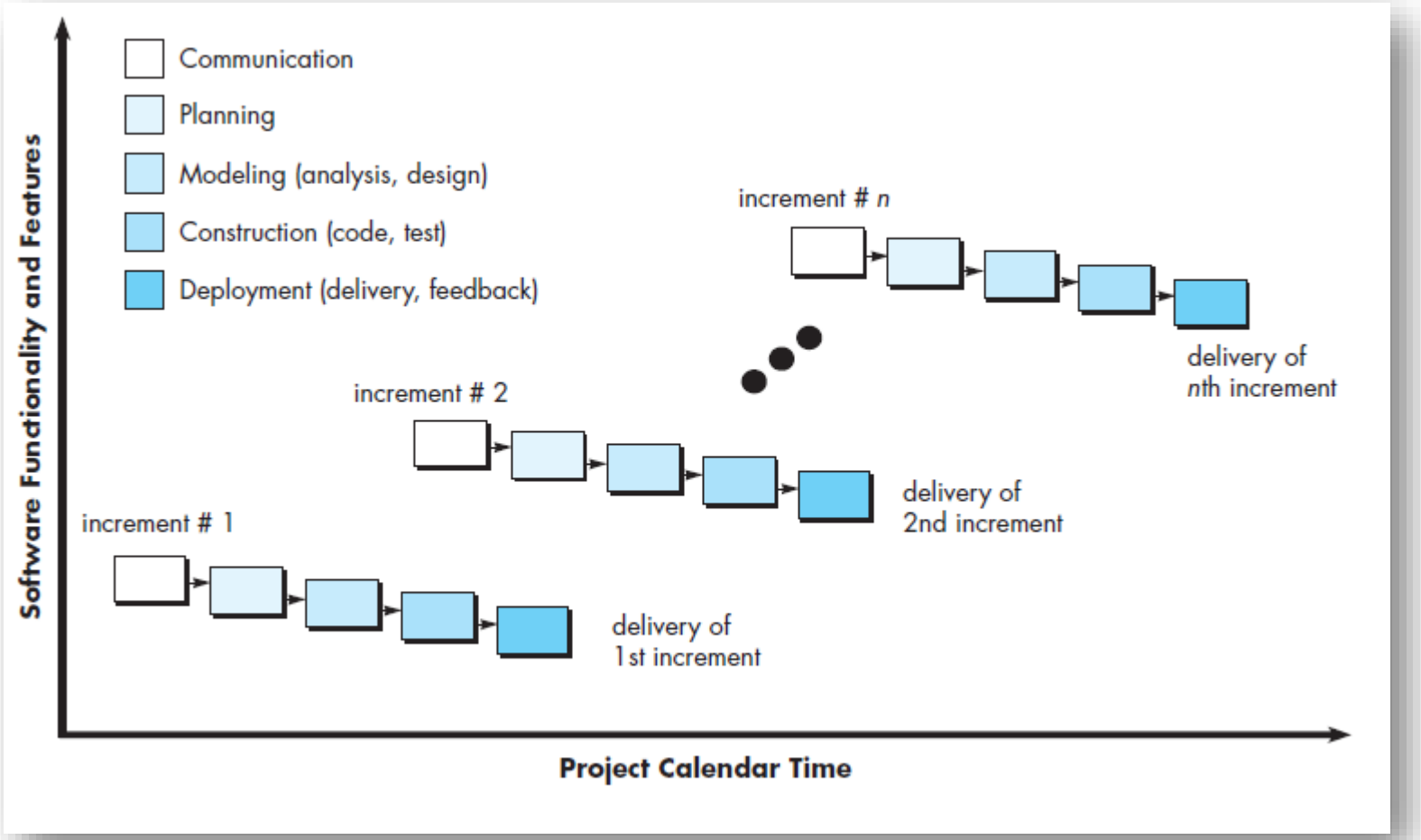
- A variation in the representation of the waterfall model is called the V-model.
- the V-model depicts the relationship of quality assurance actions to the actions associated with communication



Incremental development

- The incremental model combines the elements' linear and parallel process flows. the incremental model applies linear sequences in a staggered fashion as calendar time progresses.
- each linear sequence produces deliverable "increments" of the software.
- This process is repeated following the delivery of each increment, until the complete product is produced.

Incremental Process Models



Evolutionary Process Models

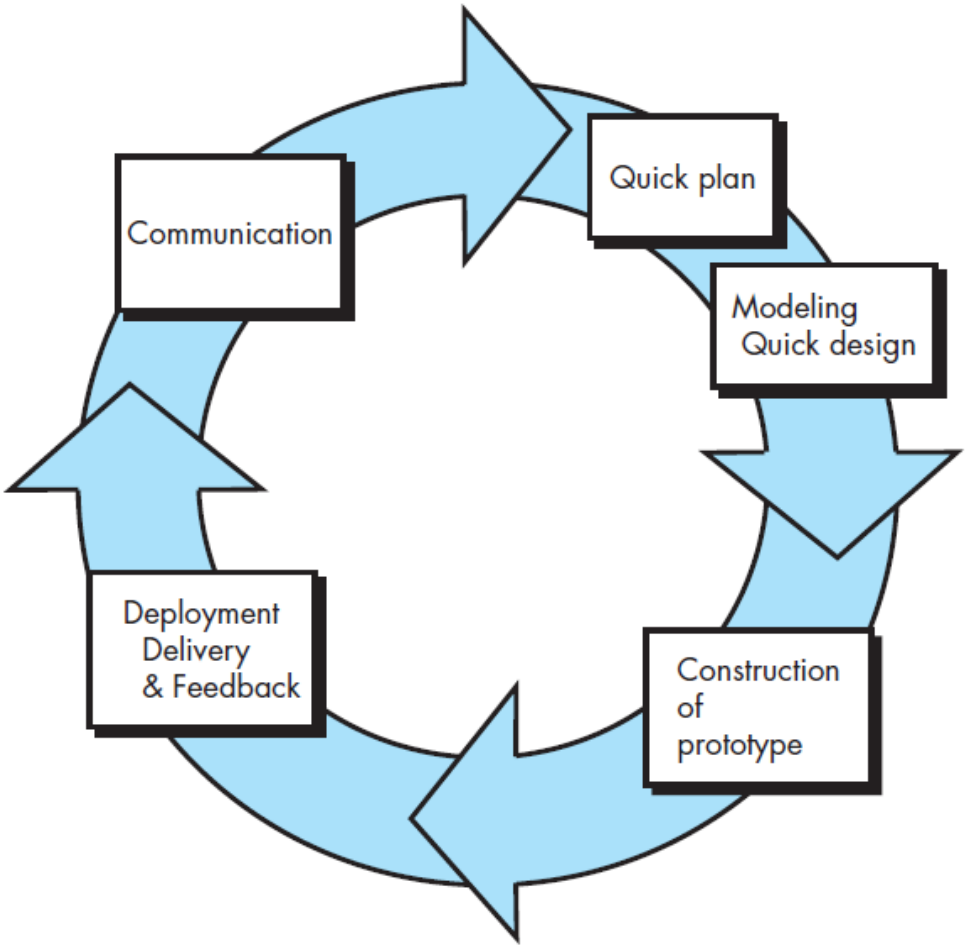
- Software, evolves over a period of time. Business and product requirements often change as development proceeds, making a straight line path to an end product unrealistic, but a limited version must be introduced to meet competitive or business pressure.

- a set of core product or system requirements is well understood, but the details of product or system extensions have yet to be defined. In these situations, you need a process model that has been explicitly designed to accommodate a product that grows and changes.

- Evolutionary models are iterative. They are characterized in a manner that enables you to develop increasingly more complete versions of the software. In the paragraphs that follow, we present two common evolutionary process models.

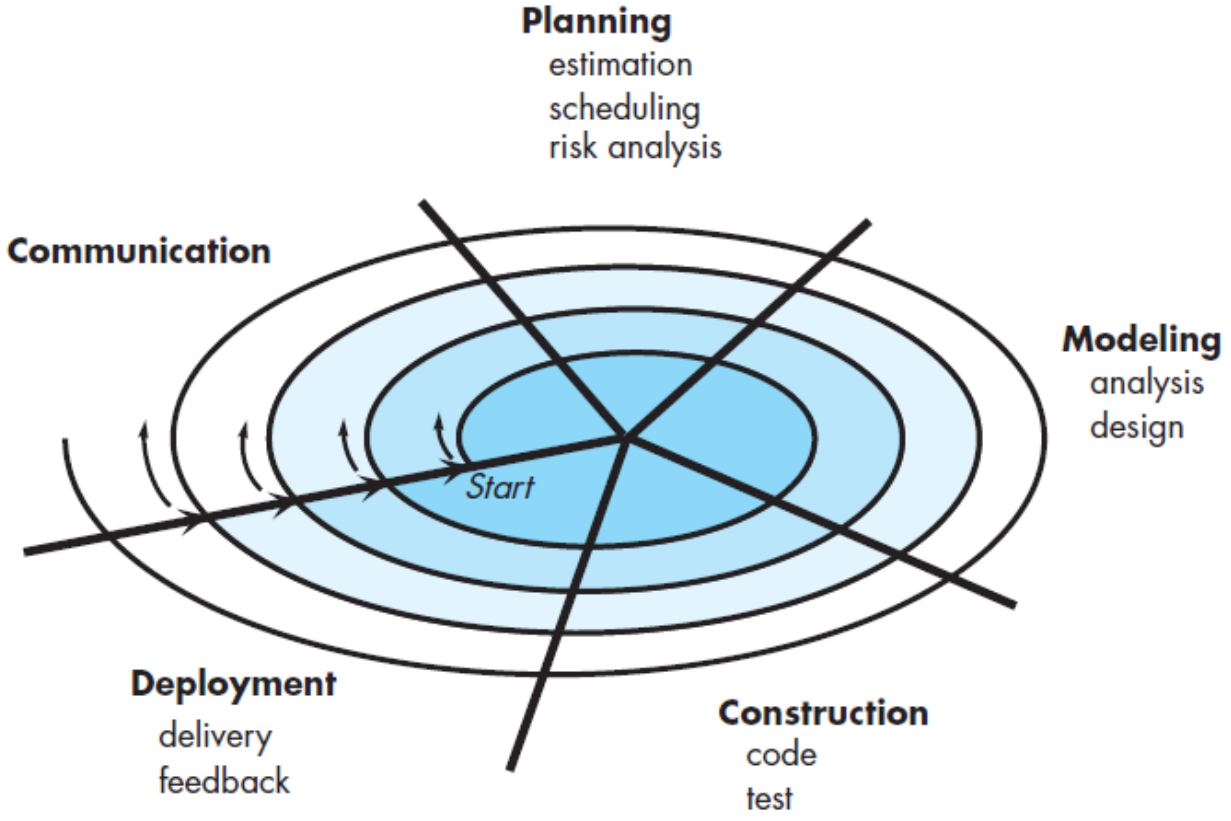
Evolutionary Process Models

1. Prototyping



Evolutionary Process Models

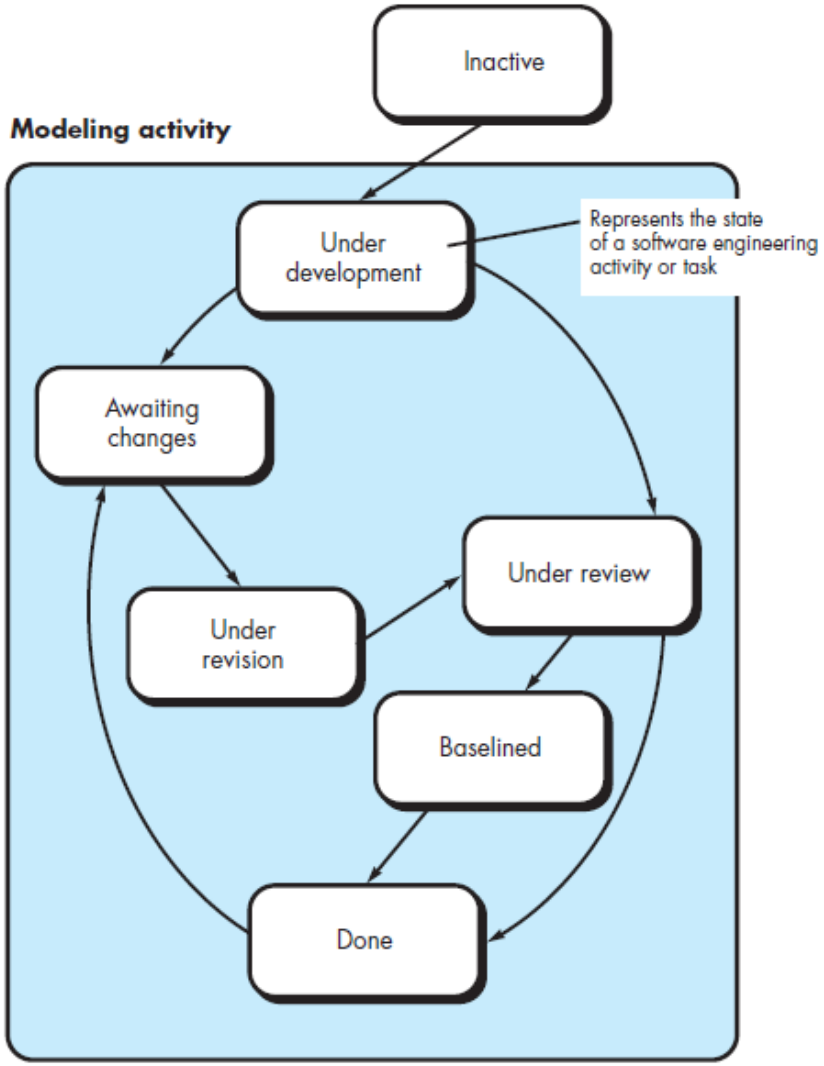
2. The Spiral



Evolutionary Process Models

3. Concurrent

- ✓ In real life the software development activities do not take place in sequence
- ✓ Most activities will be going on concurrently but reside in different states
- ✓ The states will change when some event occurs
- ✓ All the activities are shown along with their states at any point of time
- ✓ As time goes on the states of the activities will change



Process Activities

Process activities

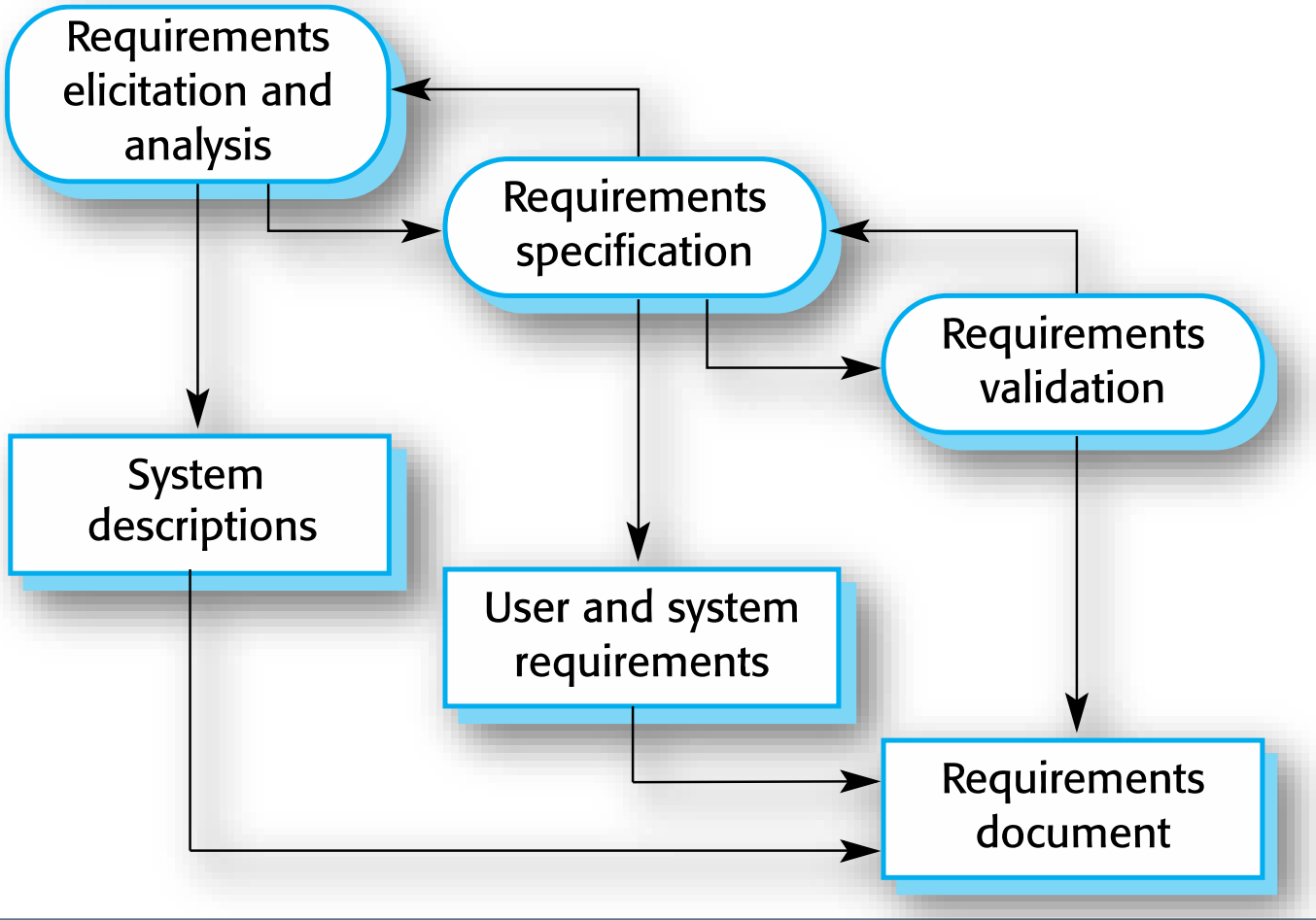
- Inter-leaved sequences of technical, collaborative and managerial activities with the overall goal of specifying, designing, implementing and testing a software system.

- The four basic process activities of specification, development, validation and evolution are organized differently in different development processes.

Requirements engineering process

- Requirements elicitation and analysis
- Requirements specification
- Requirements validation

The requirements engineering process



Software design and implementation

- The process of converting the system specification into an executable system.

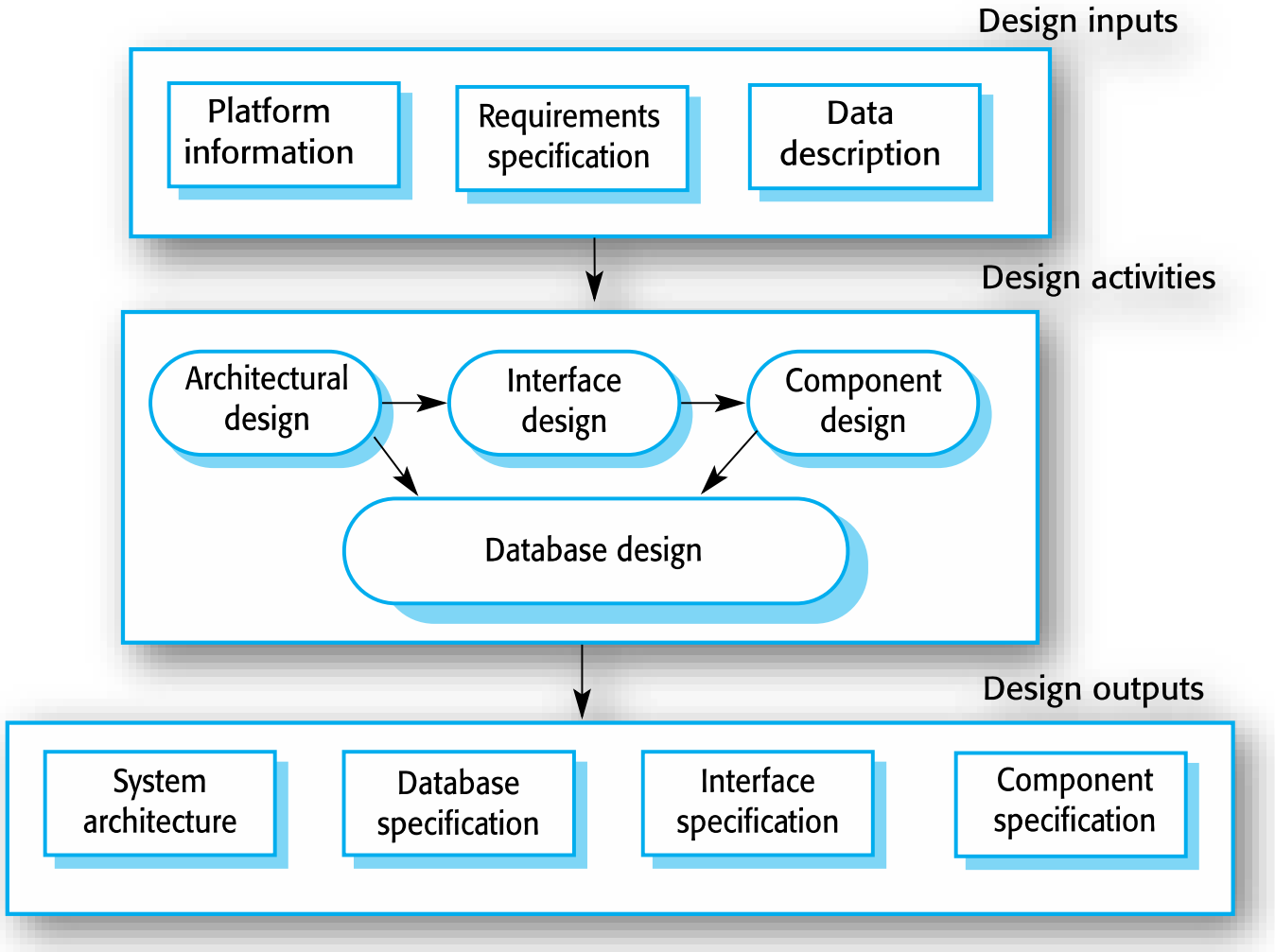
- **Software design:**

  Design a software structure that realises the specification.

- **Implementation:**

  Translate this structure into an executable program.

A general model of the design process



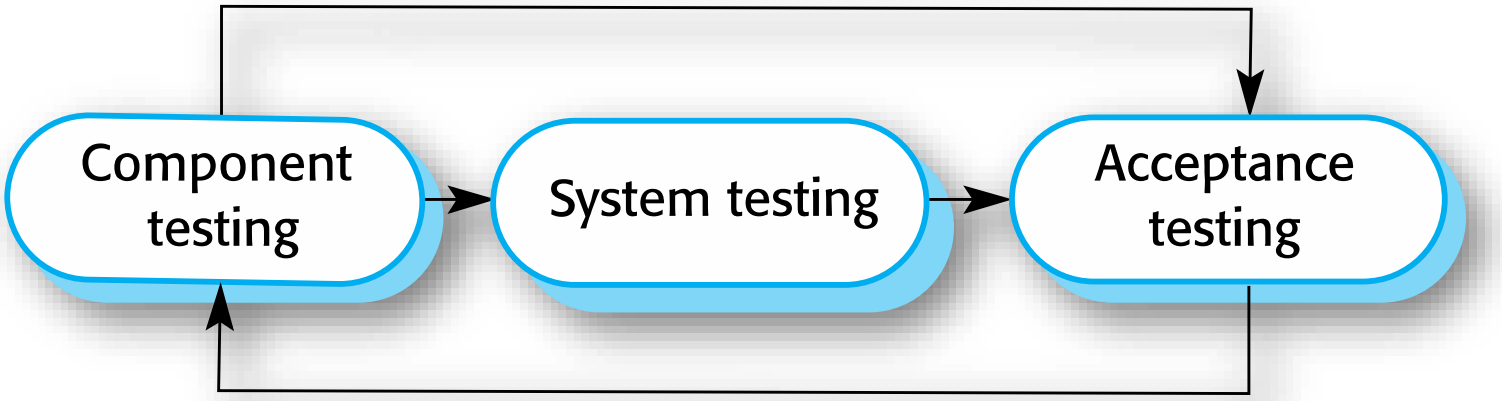
System implementation

- The software is implemented either by developing a program or programs or by configuring an application system.
- Design and implementation are interleaved activities for most types of software system.
- Programming is an individual activity with no standard process.
- Debugging is the activity of finding program faults and correcting these faults.

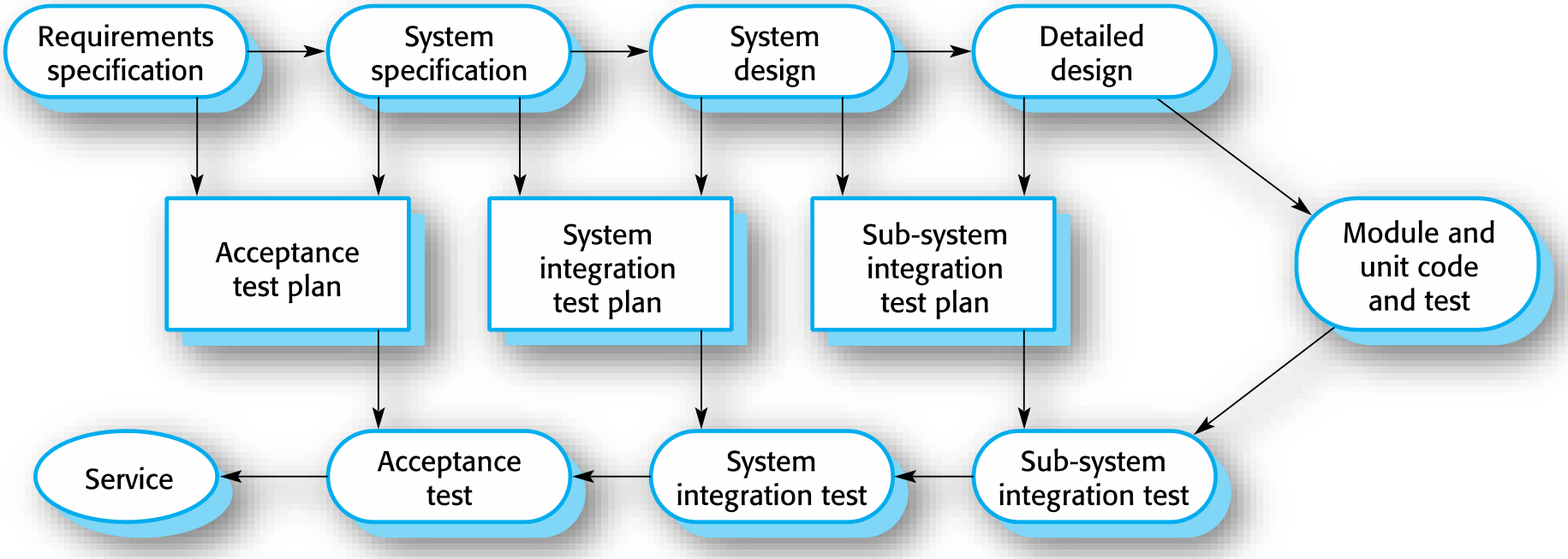
Software validation

- 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 is the most commonly used V & V activity

Stages of testing

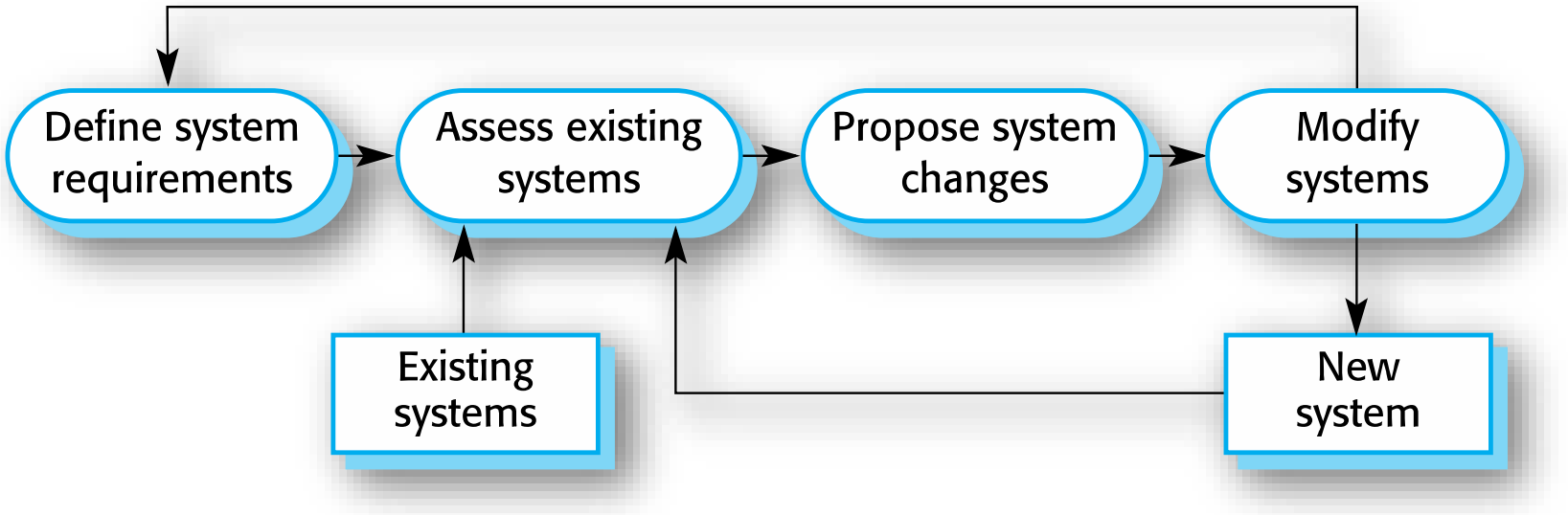


Testing phases in a plan-driven software process (V-model)



System Evolution

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



Coping with change

Coping with change

- **Change is inevitable in all large software projects.**
  - ✓ Business changes lead to new and changed system requirements
  - ✓ New technologies open up new possibilities for improving implementations
  - ✓ Changing platforms require application changes

- **Change leads to rework so the costs of change include:**
  1. Rework
  2. Implementing new functionality

Reducing the costs of rework

- **Change anticipation**
  - ✓ where the software process includes activities that can anticipate or predict possible changes before significant rework is required.
- **Change tolerance**
  - ✓ where the process and software are designed so that changes can be easily made to the system.

Coping with changing requirements

- **System prototyping**
  - ✓ where a version of the system or part of the system is developed quickly to check the customer's requirements and the feasibility of design decisions. This approach supports change anticipation.
- **Incremental delivery**
  - ✓ where system increments are delivered to the customer for comment and experimentation. This supports both change avoidance and change tolerance.

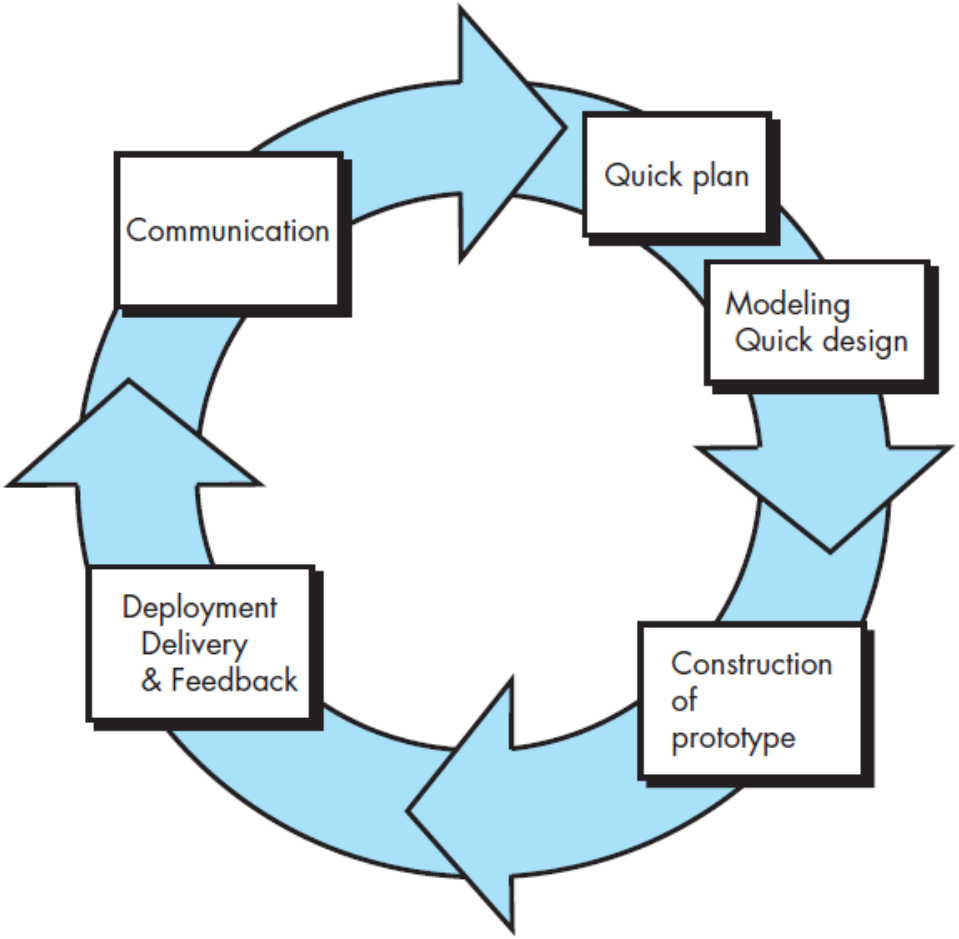
Software prototyping

- **A prototype is an initial version of a system used to demonstrate concepts and try out design options.**

- **A prototype can be used in:**

  1. The requirements engineering process to help with requirements elicitation and validation;
  2. In design processes to explore options and develop a UI design;
  3. In design processes to run back-to-back tests.

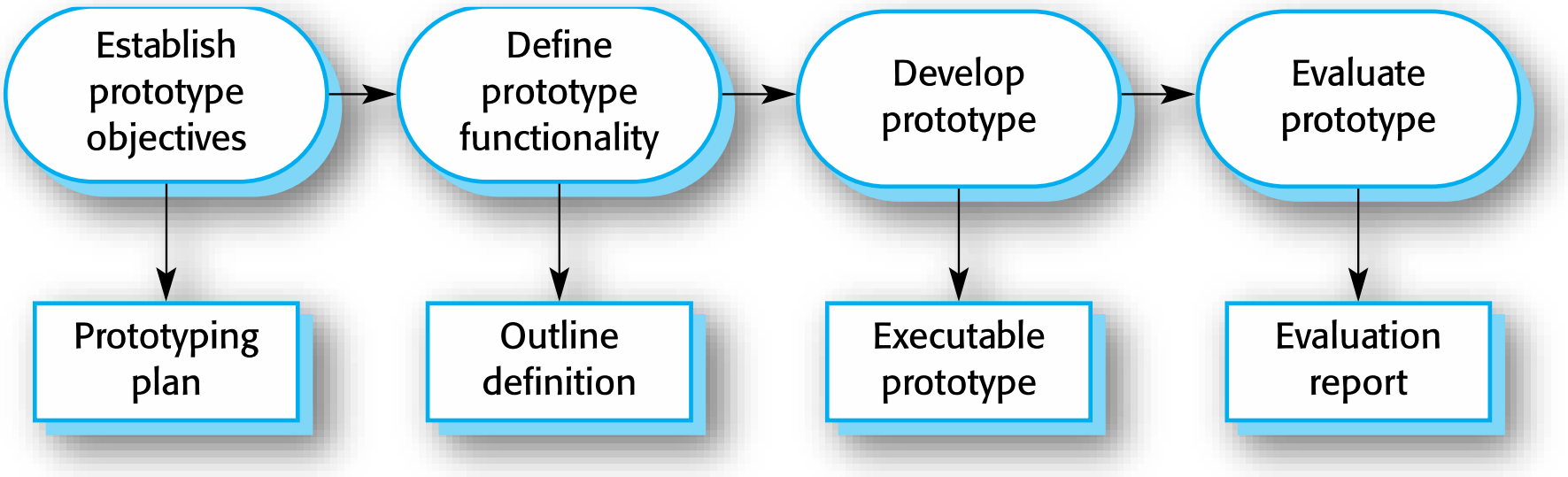
Software prototyping



Benefits of prototyping

- Improved system usability.
- A closer match to users' real needs.
- Improved design quality.
- Improved maintainability.
- Reduced development effort.

The process of prototype development



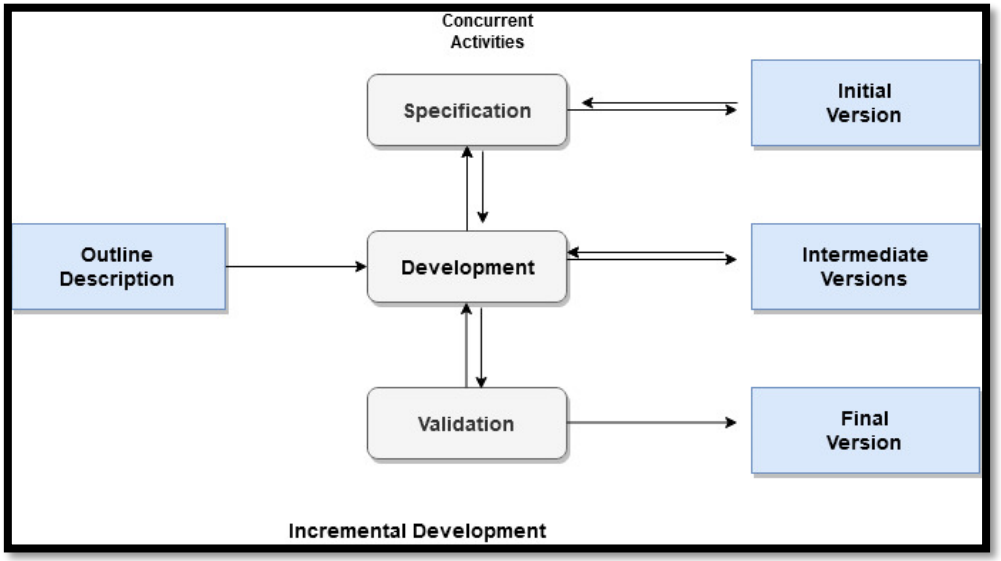
Incremental delivery

- Rather than deliver the system as a single delivery, the development and delivery is broken down into increments with each increment delivering part of the required functionality.

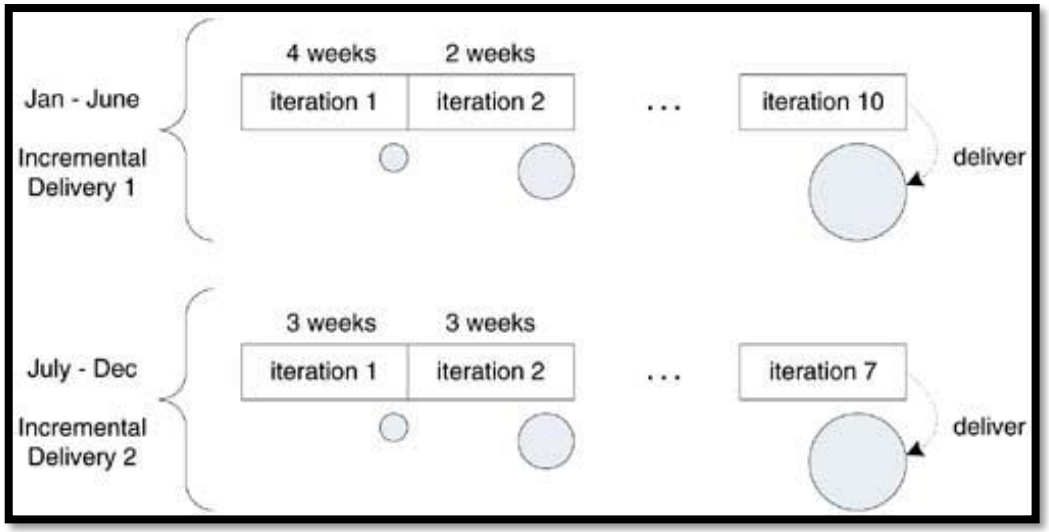
- User requirements are prioritised and the highest priority requirements are included in early increments.

Incremental development and delivery

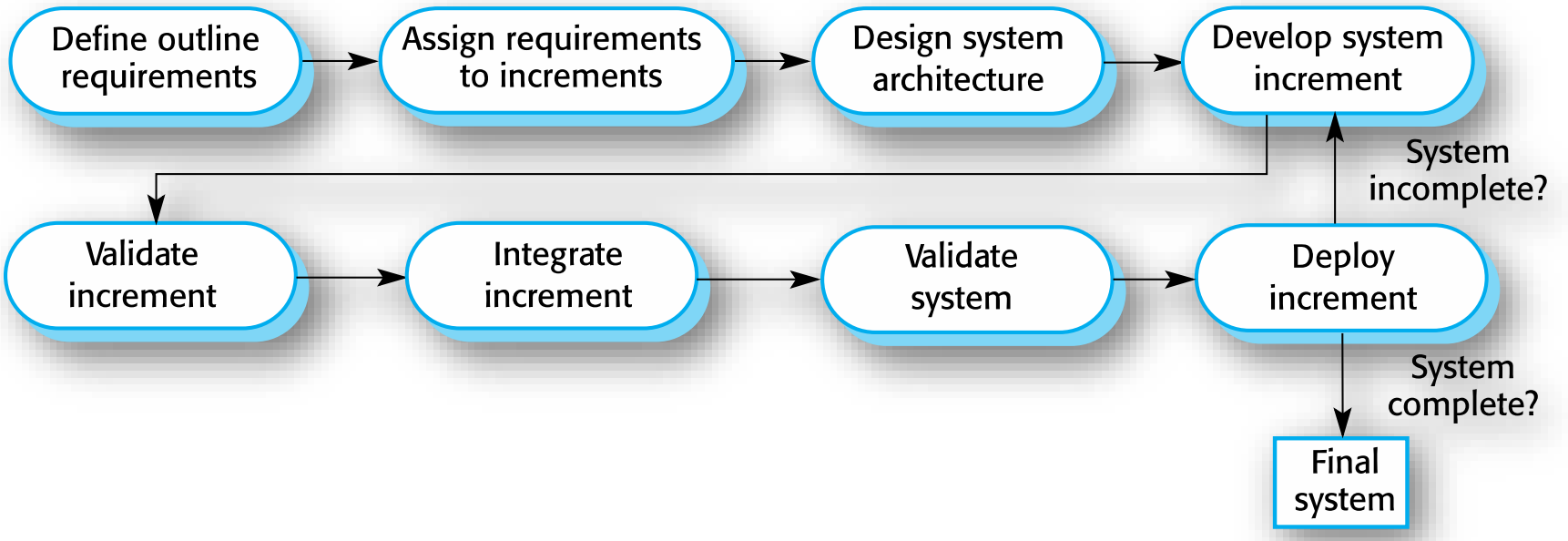
- **Incremental development**



- **Incremental delivery**



Incremental delivery

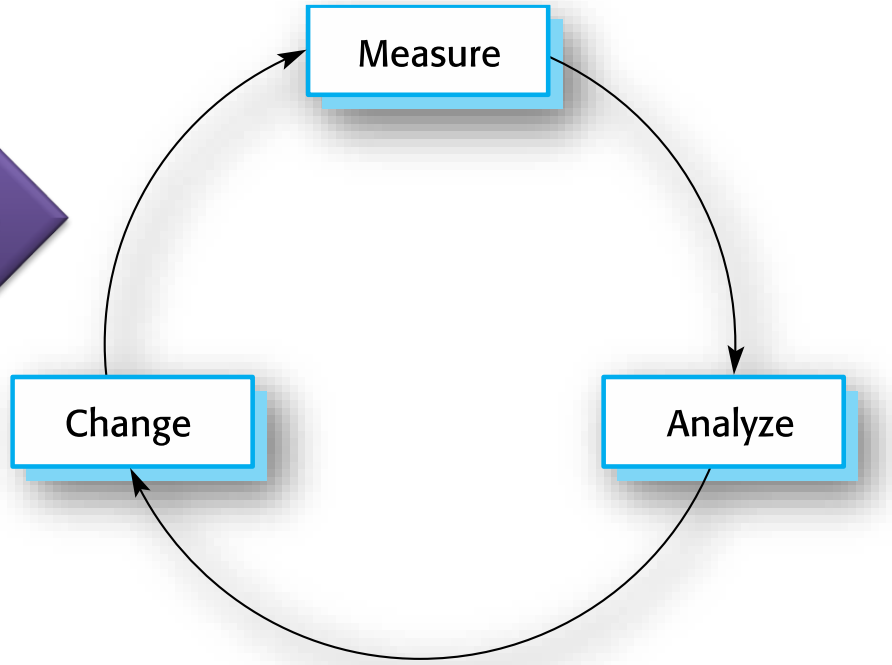


Process improvement

Process improvement

- Process improvement means understanding existing processes and changing these processes to increase product quality and/or reduce costs and development time.

The process improvement cycle



Process assessment and improvement

- the process can be assessed to ensure that it meets a set of basic process criteria

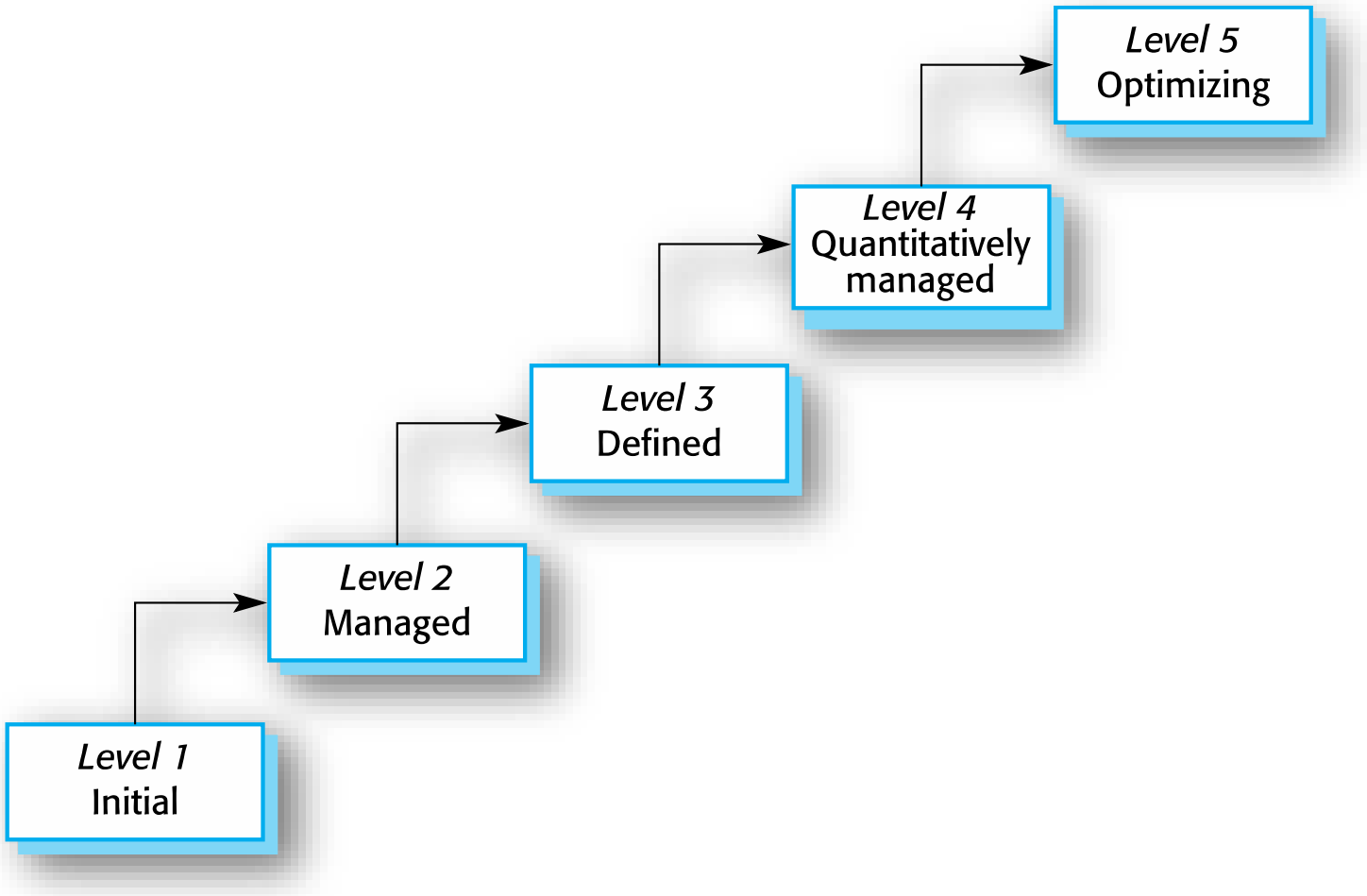
A number of different approaches to software process assessment and improvement have been proposed over the past few decades:

- Standard CMMI Assessment Method for Process Improvement (SCAMPI)
- CMM-Based Appraisal for Internal Process Improvement (CBA IPI)
- SPICE (ISO/IEC15504)
- ISO 9001:2000 for Software

Approaches to improvement

1. The process maturity approach

Process maturity approach



Approaches to improvement

2. The agile approach

Key points

Software processes are the activities involved in producing a software system. Software process models are abstract representations of these processes.

General process models describe the organization of software processes. Examples of these general models include the waterfall model, incremental development, and reusable component configuration and integration.

Requirements engineering is the process of developing a software specification. Specifications are intended to communicate the system needs of the customer to the system developers.

Design and implementation processes are concerned with transforming a requirements specification into an executable software system.

Software validation is the process of checking that the system conforms to its specification and that it meets the real needs of the users of the system.

Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous, and the software must evolve to remain useful.

Processes should include activities to cope with change. This may involve a prototyping phase that helps avoid poor decisions on requirements and design. Processes may be structured for iterative development and delivery so that changes may be made without disrupting the system as a whole.

Process improvement is the process of improving existing software processes to improve software quality, lower development costs, or reduce development time. It is a cyclic process involving process measurement, analysis, and change.