Tim Pengajar IF2150

IF2150 – Rekayasa Perangkat Lunak
Model Proses

SEMESTER I TAHUN AJARAN 2024/2025

KNOWLEDGE & SOFTWARE ENGINEERING

Generic Software Process Framework

- **Communication**
  - System analyst vs User
  - System analyst vs Programmer
- **Planning**
  - Cost, Time, human resources
- **Modeling**
  - Structured approach
  - Object oriented approach
- **Construction**
  - Coding and Testing
- **Deployment**
  - Software delivery to customer

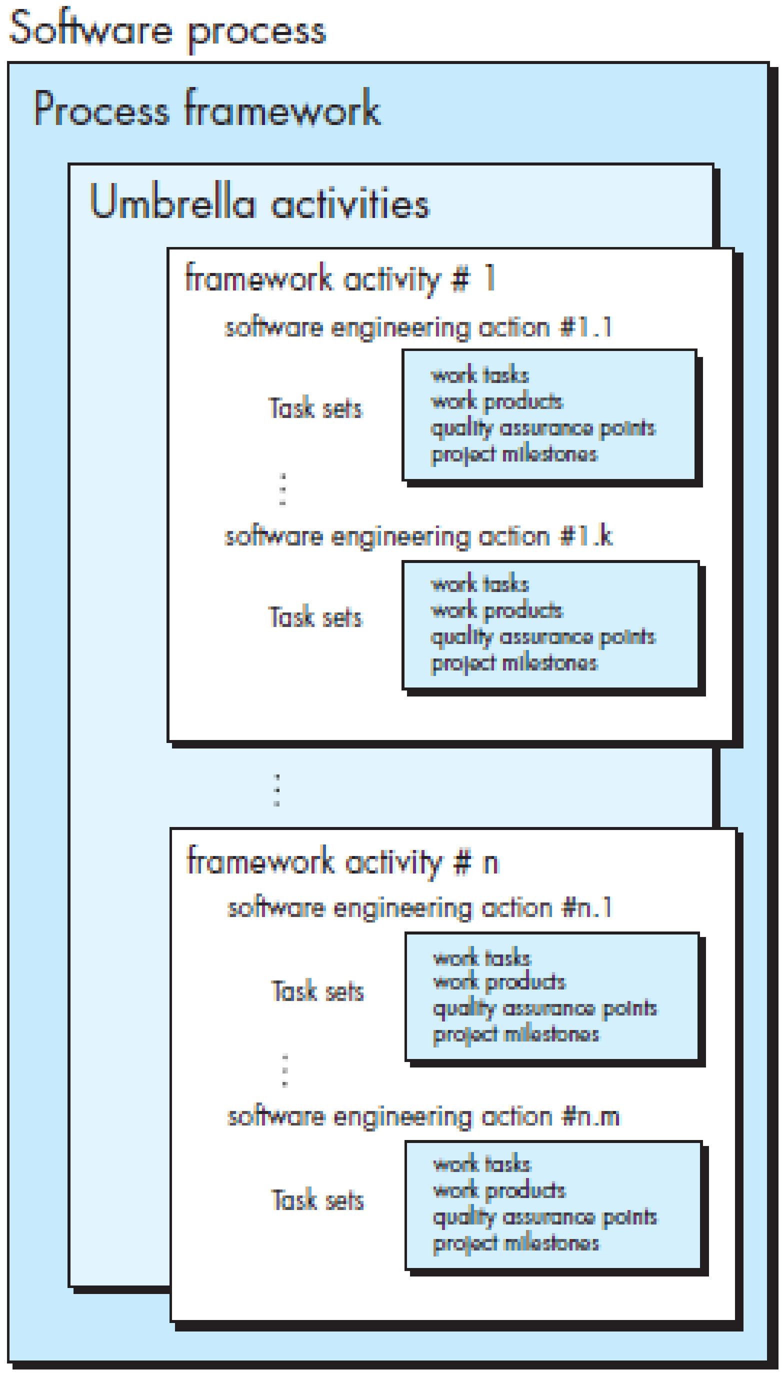
Umbrella Activities

- **Software project tracking and control**
  - allows the software team to **assess progress** against the project plan and **take** any necessary **action** to maintain the schedule.
- **Risk management**
  - **assesses risks** that may affect the **outcome** of the project or the **quality** of the product.
- **Software quality assurance**
  - defines and conducts the activities required to **ensure** software **quality**.
- **Technical reviews**
  - assesses software engineering work products in an effort to **uncover** and **remove errors** before they are propagated to the next activity.

Umbrella Activities

- **Measurement**
  - defines and collects **process**, **project**, and **product measures** that assist the team in delivering software that meets **stakeholders' needs**; can be used in conjunction with all other framework and umbrella activities.
- **Software configuration management**
  - manages the **effects** of **change** throughout the software process.
- **Reusability management**
  - defines **criteria** for work product **reuse** (including software components) and establishes mechanisms to **achieve reusable** components.
- **Work product preparation and production**
  - encompasses the activities required to create work products such as **models**, **documents**, **logs**, **forms**, and **lists**.

A Software Process Framework



KNOWLEDGE & SOFTWARE ENGINEERING

Process Adaptation

- The software engineering process should be agile and adaptable
  - to the problem,
  - to the project,
  - to the team, and
  - to the organizational culture
- A process adopted for one project might be significantly different than a process adopted for another project.

KNOWLEDGE & SOFTWARE ENGINEERING

The essence of software engineering practice

1. Understand the problem (communication and analysis).
2. Plan a solution (modeling and software design).
3. Carry out the plan (code generation).
4. Examine the result for accuracy (testing and quality assurance).

KNOWLEDGE & SOFTWARE ENGINEERING

Software Practice Core Principles

- **The reason it all exist**
  - Software exists **to provide value** to its users
- **Keep it simple stupid (KISS)**
  - Keep the design as **simple as possible**, but not simpler
- **Maintain the vision**
  - **Clear vision** is essential to the success of any software project
- **We produce, others will consume**
  - Always specify, design, and implement knowing that **someone** else **will have to understand** what you **have done** to **carry out** his or her tasks
- **Open to the future**
  - Be **open to future changes**, don't code yourself into a corner
- **Plan for Reuse!**
  - Planning ahead for **reuse** reduces the cost and increases the value of both the reusable components and the systems that require them
- **Think First!**
  - Placing **clear** complete **thought** before any action almost always produces better results

KNOWLEDGE & SOFTWARE ENGINEERING

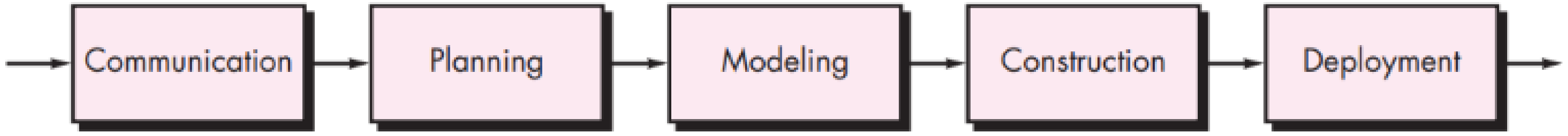
One additional aspect of the software process: Process flow

- Process framework:
    - communication,
    - planning,
    - modeling,
    - construction,
    - deployment ,
    - umbrella activities + process flow
- Organized with respect to sequence and time
    - *linear process flow*
    - *iterative process flow*
    - *evolutionary process flow*
    - *parallel process flow*

KNOWLEDGE & SOFTWARE ENGINEERING

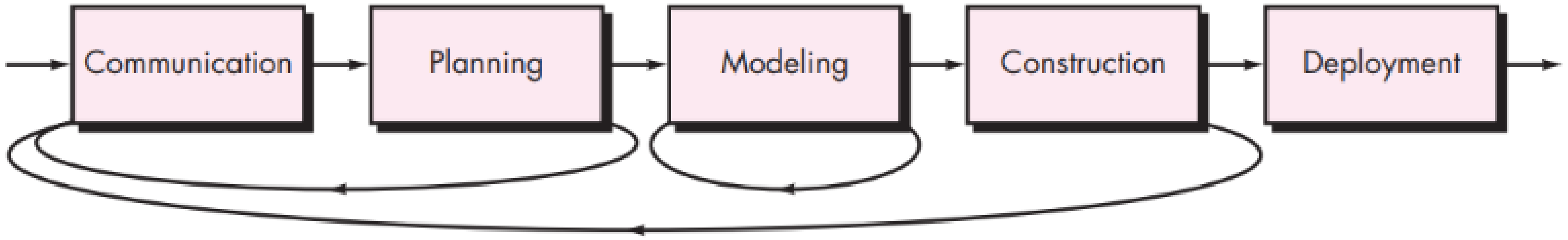
Process Flow (I)

Linear Process Flow



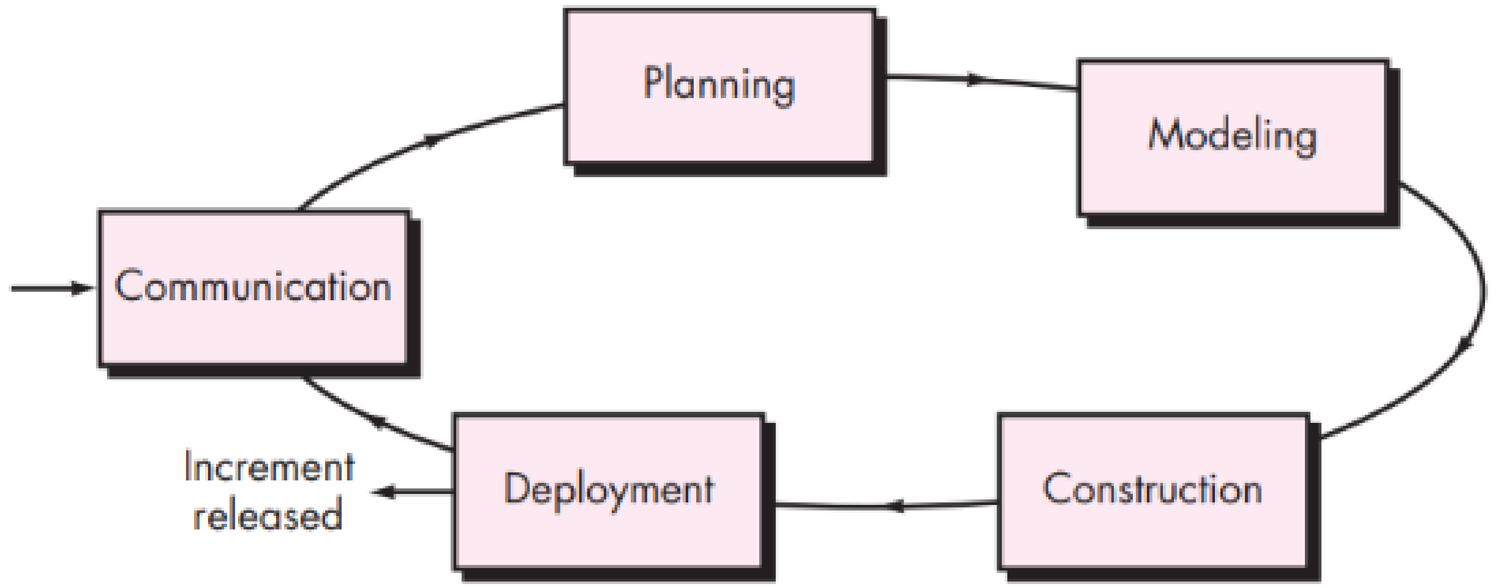
Process Flow (2)

Iterative Process Flow



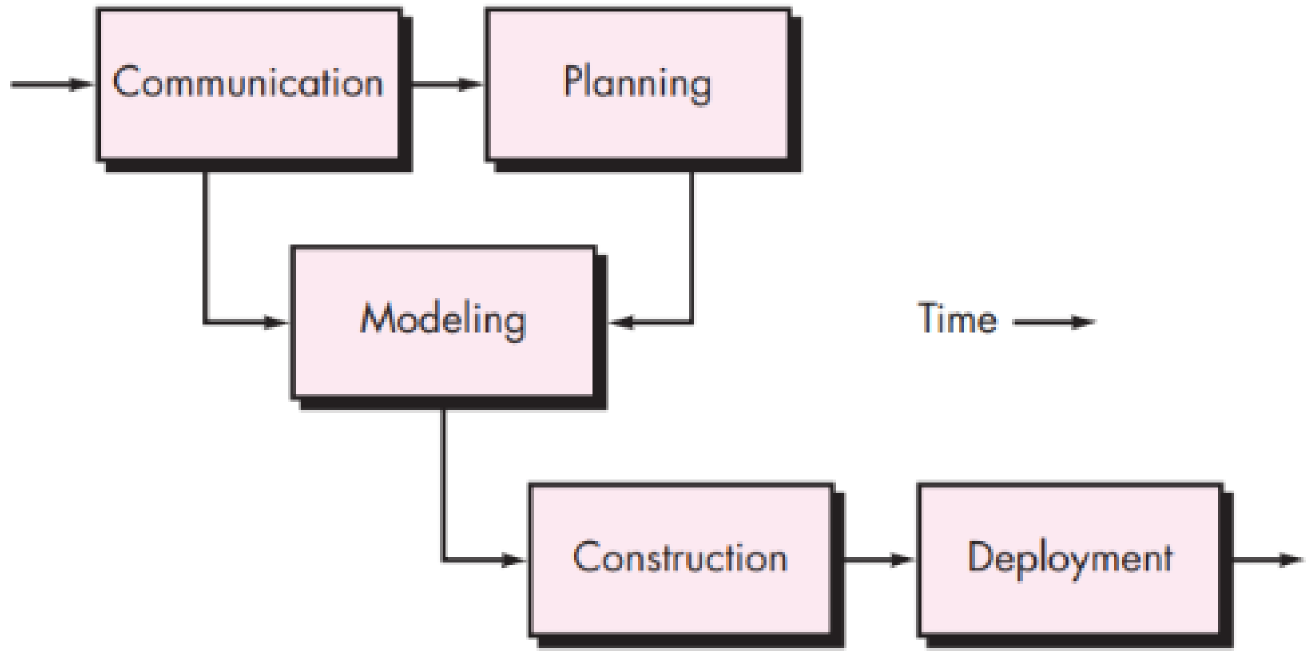
Process Flow (3)

Evolutionary process flow



Process Flow (4)

Parallel process flow

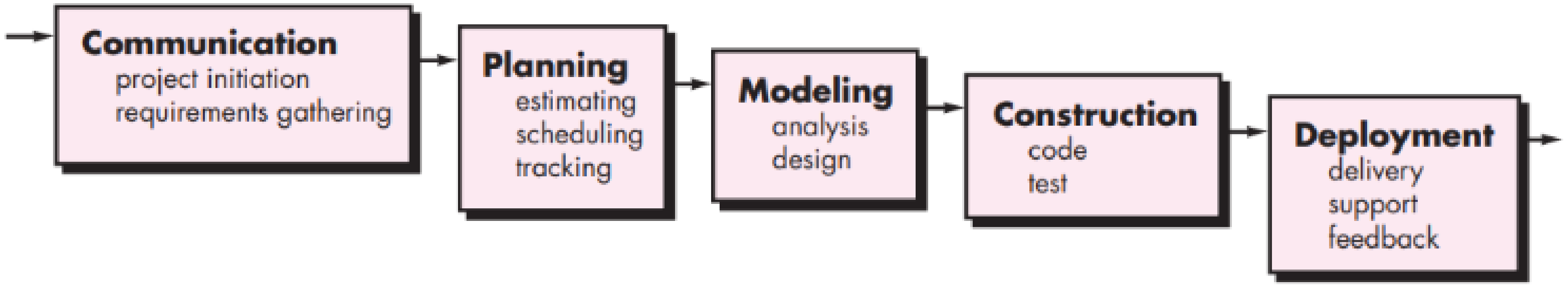


Process Models

- PRESCRIPTIVE PROCESS MODELS
- SPECIALIZED PROCESS MODELS
- UNIFIED PROCESS

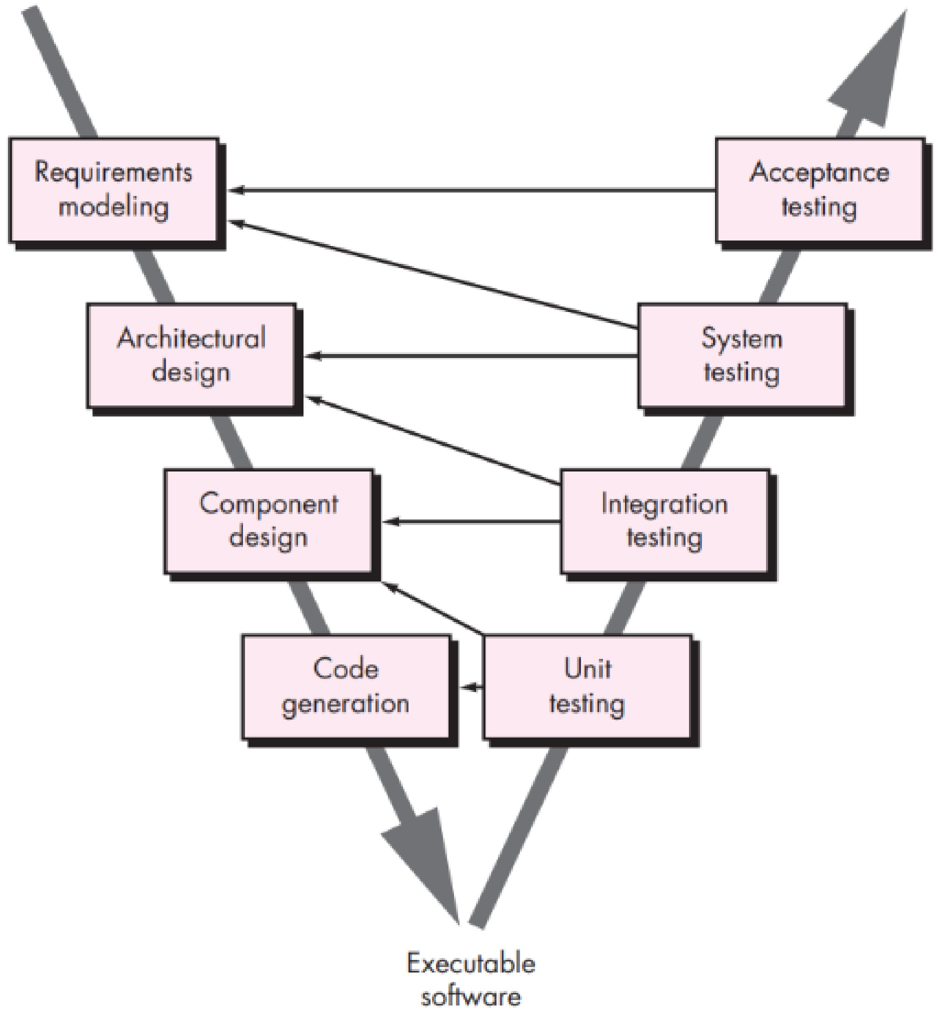
Prescriptive Process Models

- **The Waterfall Model -** *classic life cycle*



Prescriptive Process Models (2)

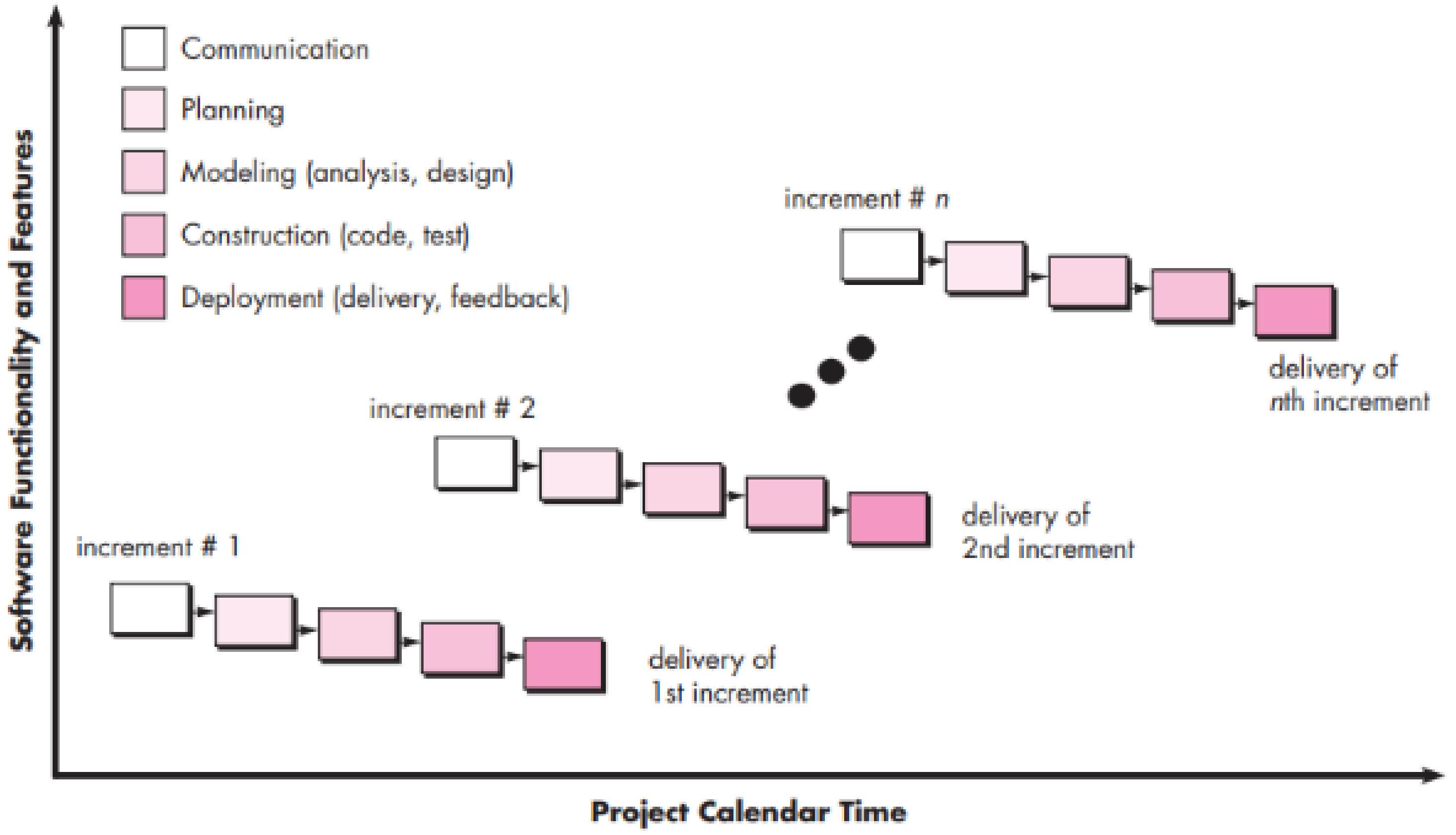
- The V model



KNOWLEDGE & SOFTWARE ENGINEERING

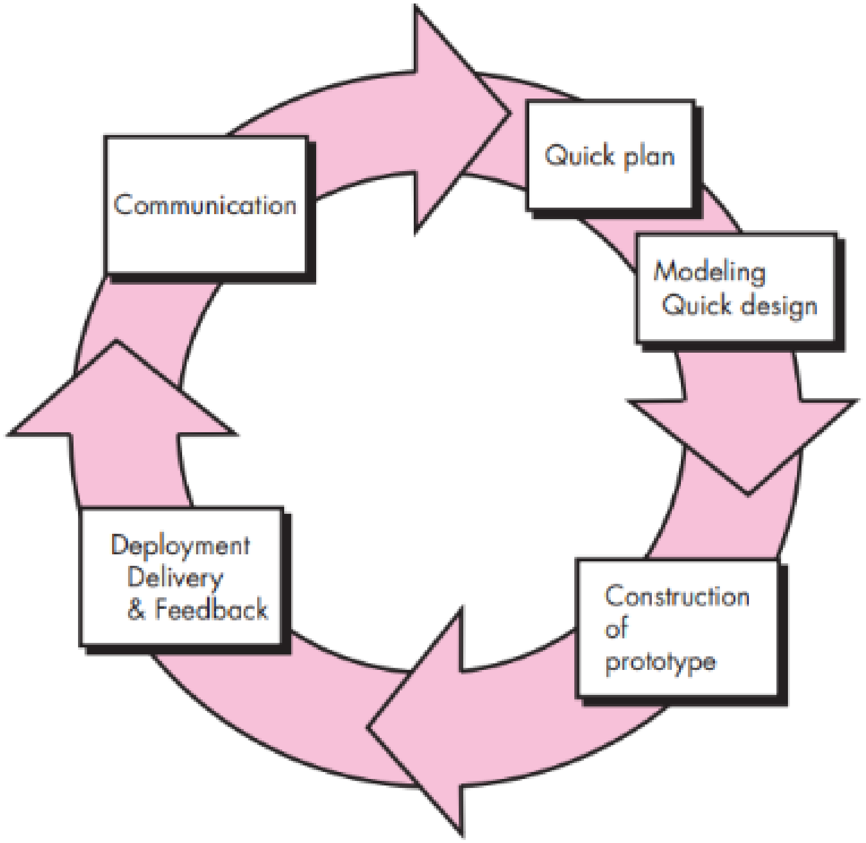
Prescriptive Process Models (3)

- Incremental Process Models


Communication
Planning
Modeling (analysis, design)
Construction (code, test)
Deployment (delivery, feedback)
increment # 1
delivery of 1st increment
increment # 2
delivery of 2nd increment
increment # n
delivery of nth increment
Software Functionality and Features
Project Calendar Time

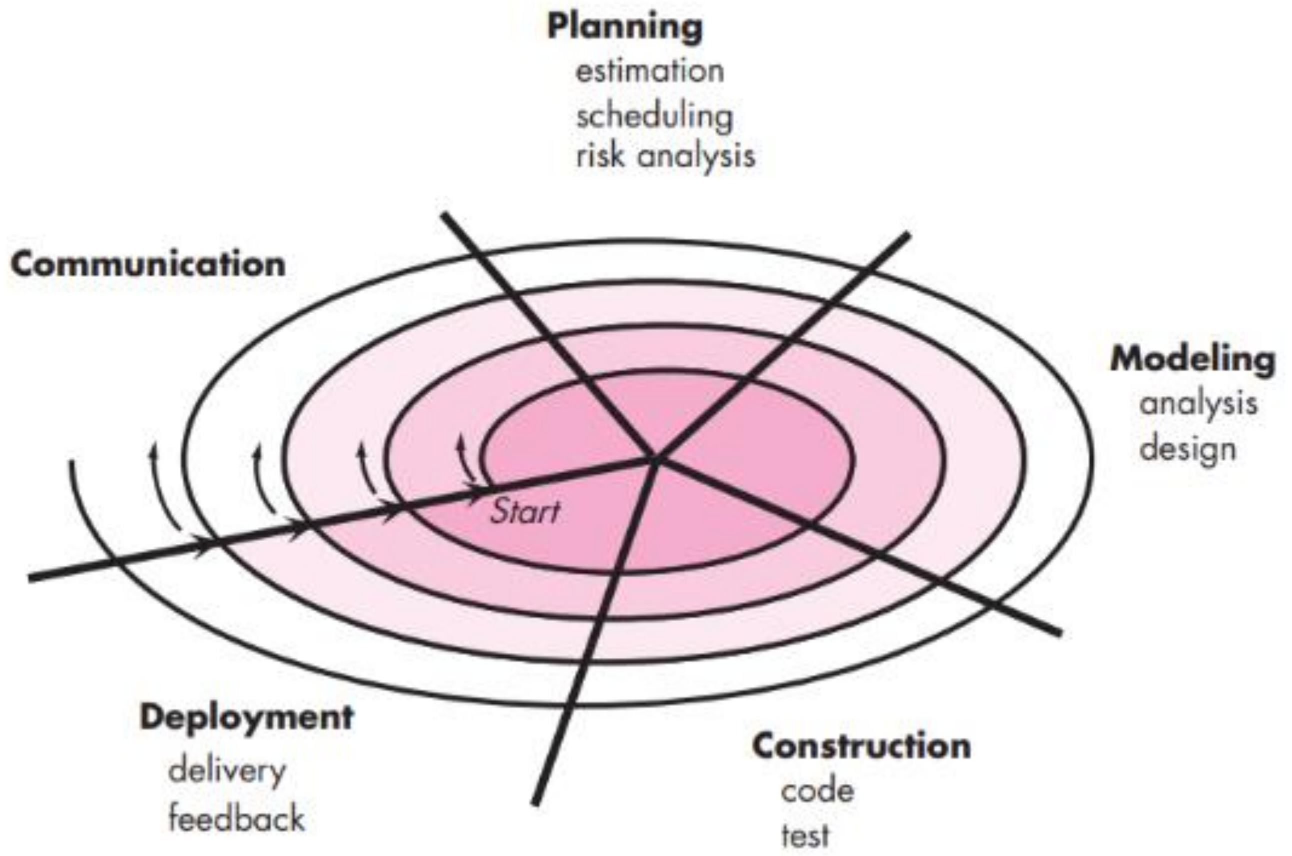
Prescriptive Process Models (4)

- Evolutionary Process Models – prototyping paradigm



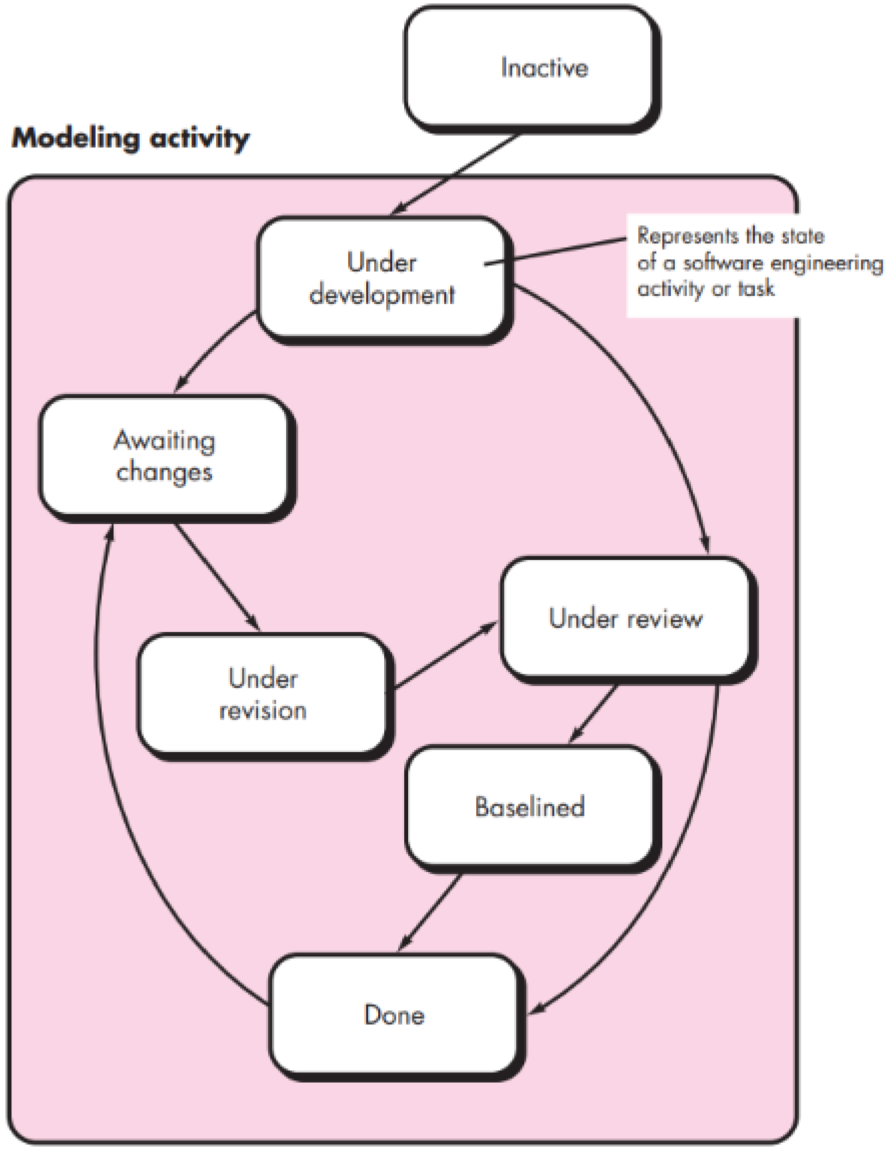
Prescriptive Process Models (5)

- Evolutionary Proces Models – the Spiral Model



Prescriptive Process Models (6)

- Concurrent Models



Specialized process models

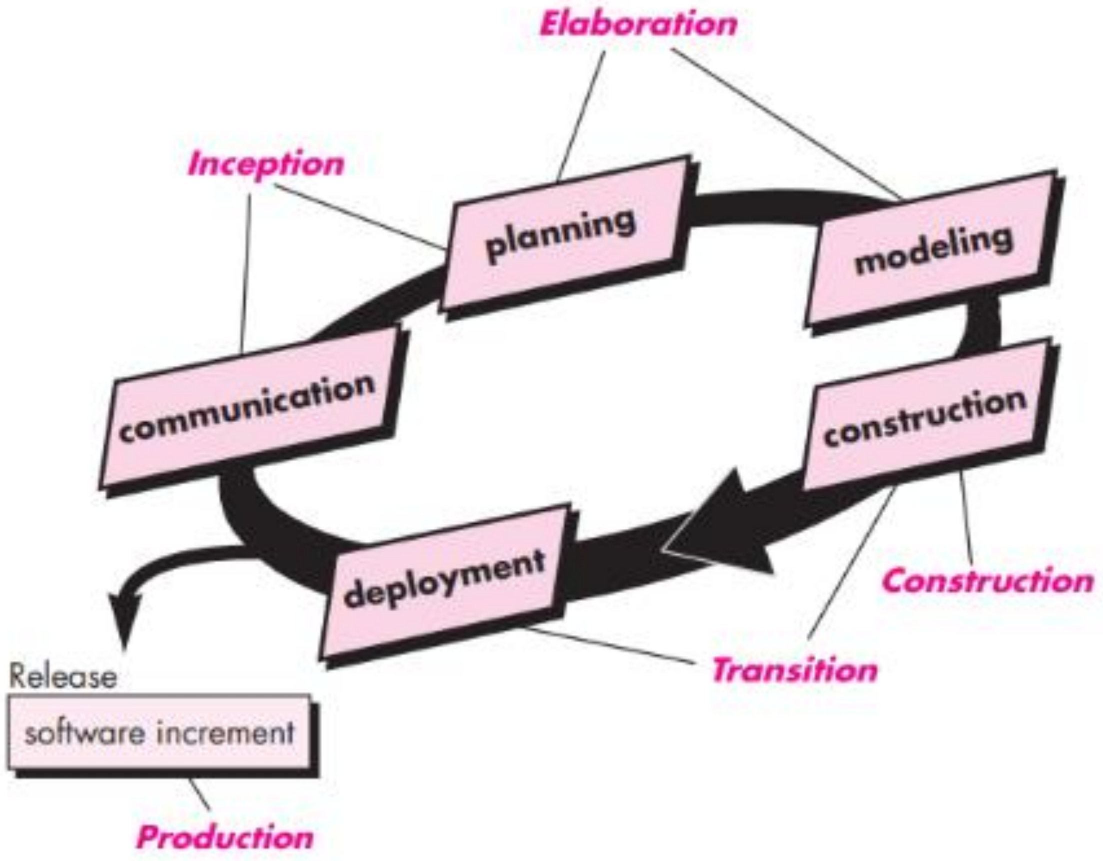
- **Component-Based Development -** comprises applications from prepackaged software components.
- **The Formal Methods Model**
    - encompasses a set of activities that leads to formal mathematical specification of computer software, enable you to specify, develop, and verify a computer-based system by applying a rigorous, mathematical notation.
    - the formal methods approach has gained adherents among software developers who must build safety-critical software (e.g., developers of aircraft avionics and medical devices) and among developers that would suffer severe economic hardship should software errors occur
- **Aspect-Oriented Software Development**
    - often referred to as *aspect-oriented programming (AOP) or aspect-oriented component engineering*
    - a relatively new software engineering paradigm that provides a process and methodological approach for defining, specifying, designing, and constructing *aspects* —"*mechanisms beyond subroutines and inheritance for* localizing the expression of a crosscutting concern"

KNOWLEDGE & SOFTWARE ENGINEERING

Unified Process

- a "use case driven, architecture-centric, iterative and incremental"
- The result was UML—a *unified modeling*
- *language that contains a robust notation for the modeling and development of* object- oriented systems.

Unified Process (2)



System Engineering vs Software Engineering

KNOWLEDGE & SOFTWARE ENGINEERING

System – Definition

Webster's Dictionary

- A set or arrangement of things so related as to form a unity or organic whole
- A set of facts, principles, rules, etc., classified and arranged in an orderly form so as to show a logical plan linking the various parts
- A method or plan of classification or arrangement
- An established way of doing something; method; procedure….
- …..
- ….

KNOWLEDGE & SOFTWARE ENGINEERING

Computer-Based Systems [PRE2007]

- *A set or arrangement of elements that are organized to accomplish some predefined goal by **processing information***
- The goal:

  To support some business function or to develop a product that can be sold to **generate business revenue**
- To accomplish the goal, a computer-based system makes use of a variety of system elements

Computer-Based System Elements

- Software
- Hardware
- People
- Data
- Documentation
- Procedures

**** SEPA 6th ed, Roger S. Pressman***

System Engineering Hierarchy

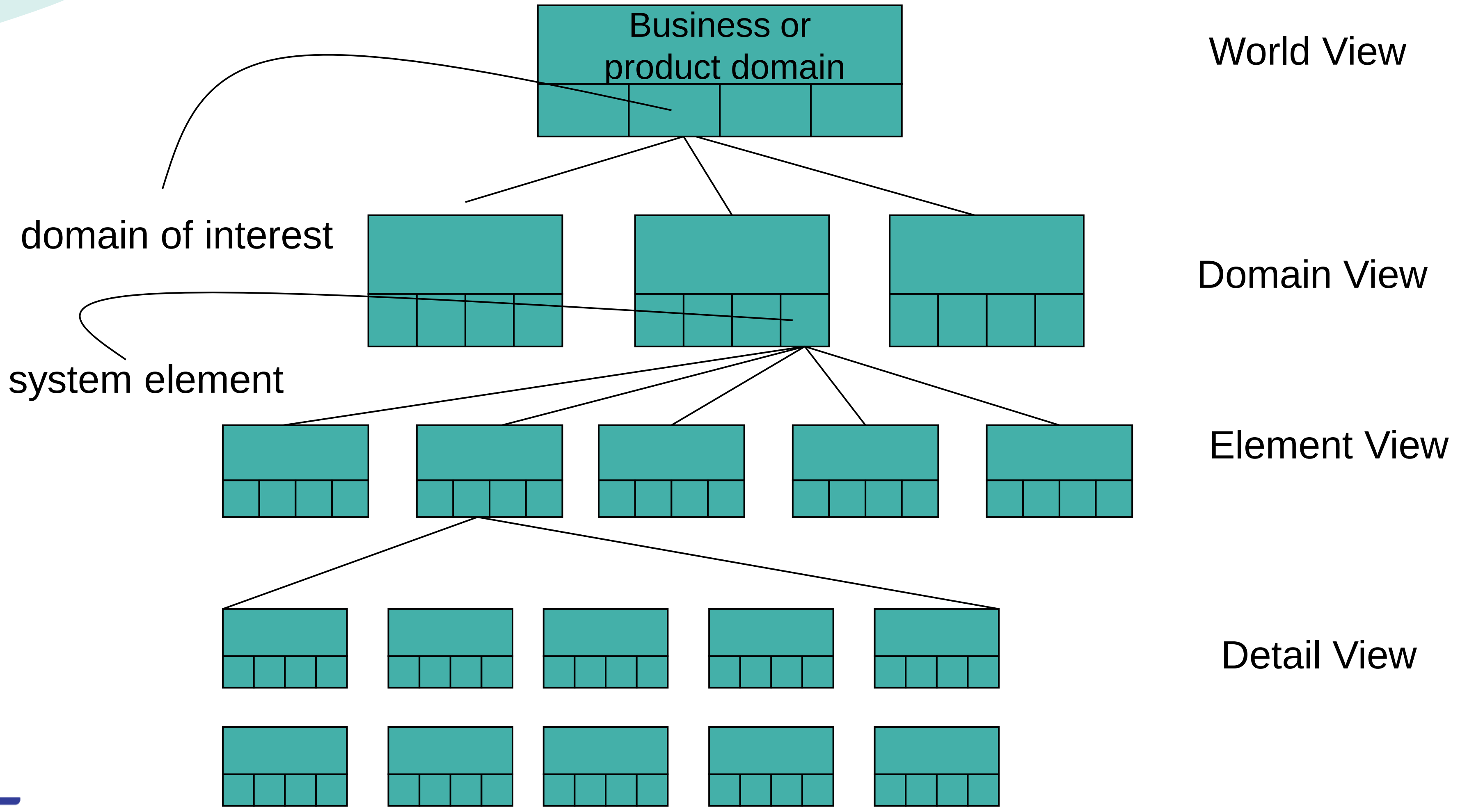
- World view → WV = $\{D_1, D_2, D_3, \ldots, D_n\}$
  - Composed of a set of domains (D_i) which can be each be a system or system of systems
- Domain view → DV = $\{E_1, E_2, E_3, \ldots, E_m\}$
  - Composed of specific elements (E_j) each of which serves some role in accomplishing the objective and goals fo the domain or component
- Element view → EV = $\{C_1, C_2, C_3, \ldots, C_k\}$
  - Each element is implemented by specifying the technical component (C_k) that achieve the necessary function for an element
- Detail view

***** SEPA 6th ed, Roger S. Pressman***

INSTITUT TEKNOLOGI BANDUNG 1920

KNOWLEDGE & SOFTWARE ENGINEERING

System Engineering Hierarchy

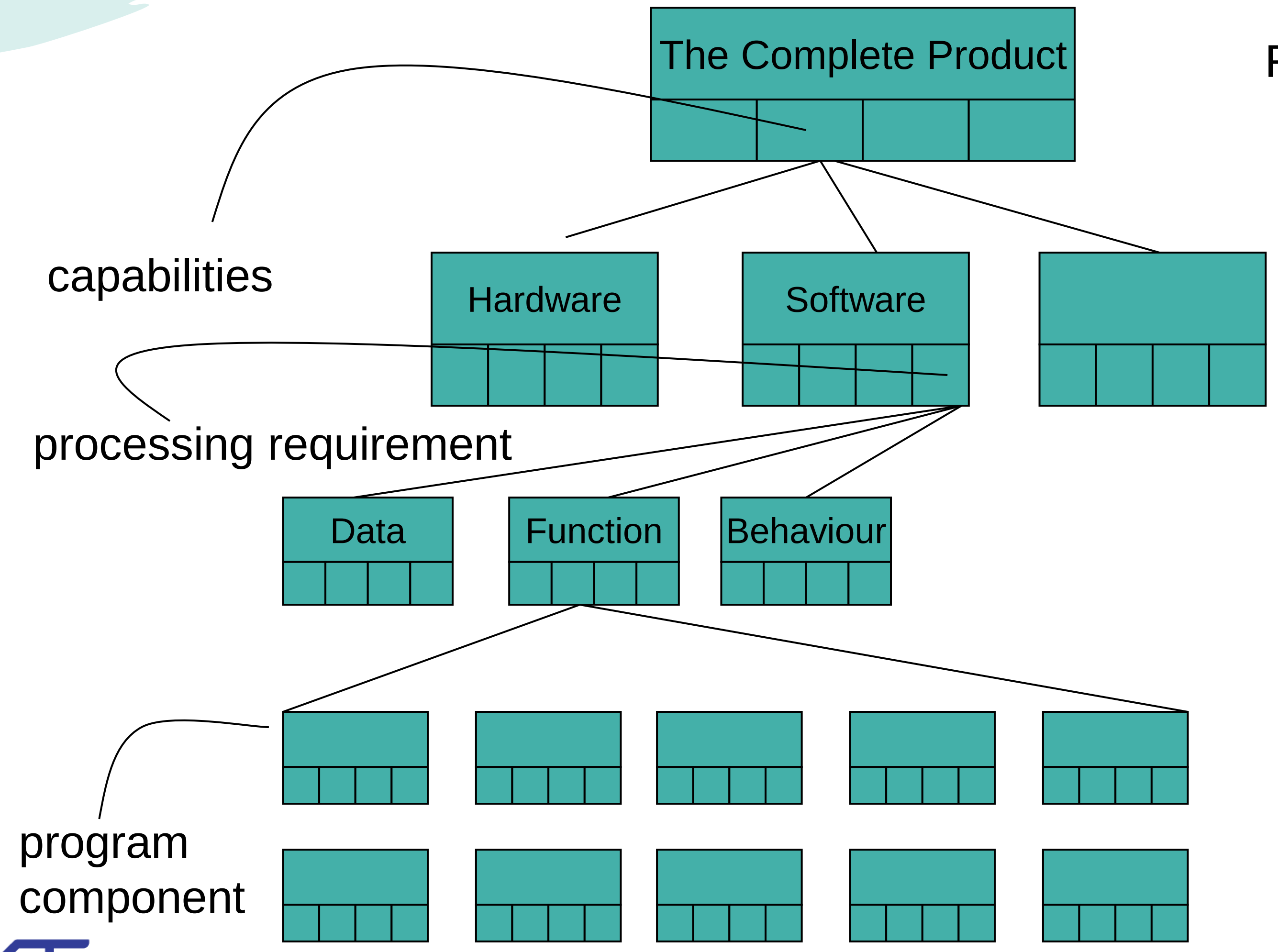


Product Engineering

- Goal
  - to translate the customer’s desire for a set of defined capabilities into a working product
- Hirarchy
  - Requirements engineering (world view)
  - Component engineering (domain view)
  - Analysis and Design modeling (element view - software engineers)
  - Construction and Integration (detailed view - software engineers)

**** SEPA 6th ed, Roger S. Pressman***

The Product Engineering Hierarchy



Requirement Engineering
(World View)

Component Engineering
(Domain View)

Analysis and Design
modeling
(Element View)

Construction &
Integration
(Detail View)