

CS2212

Introduction to Software Engineering

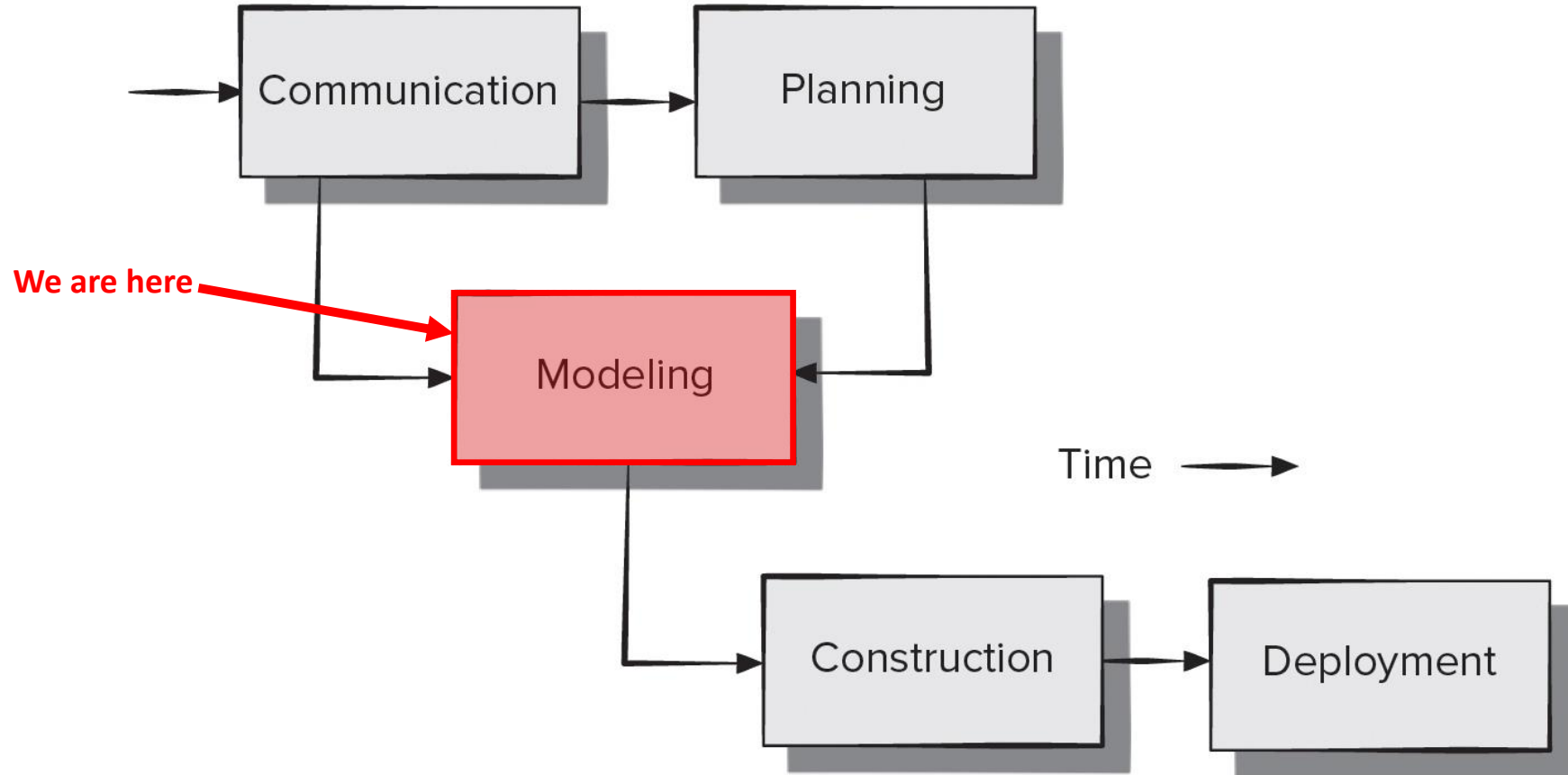
Requirements Modeling



Requirements Analysis

- **A Requirements Analysis:**
 - **Specifies** the software's **operational characteristics**.
 - **Indicates** the software's **interface with other system elements**.
 - **Establishes constraints** that the software must meet.
- **Requirements Analysis** allows a software engineer to:
 - **Elaborate on** basic **requirements** established during earlier requirement engineering tasks
 - **Build models** that depict **user scenarios**, **functional activities**, **problem classes**, and their **relationships**, system and class **behaviour**, and the **flow of data** as it is transformed

Requirements Analysis



Requirements Models

- A **requirements model** must achieve **three primary objectives**:
 1. To describe what the customer requires.
 2. To establish a basis for the creation of a software design.
 3. To define a set of requirements that can be validated once the software is built.
- A **requirements model** bridges the gap between a **system-level description** and a **software design**.

As A Bridge

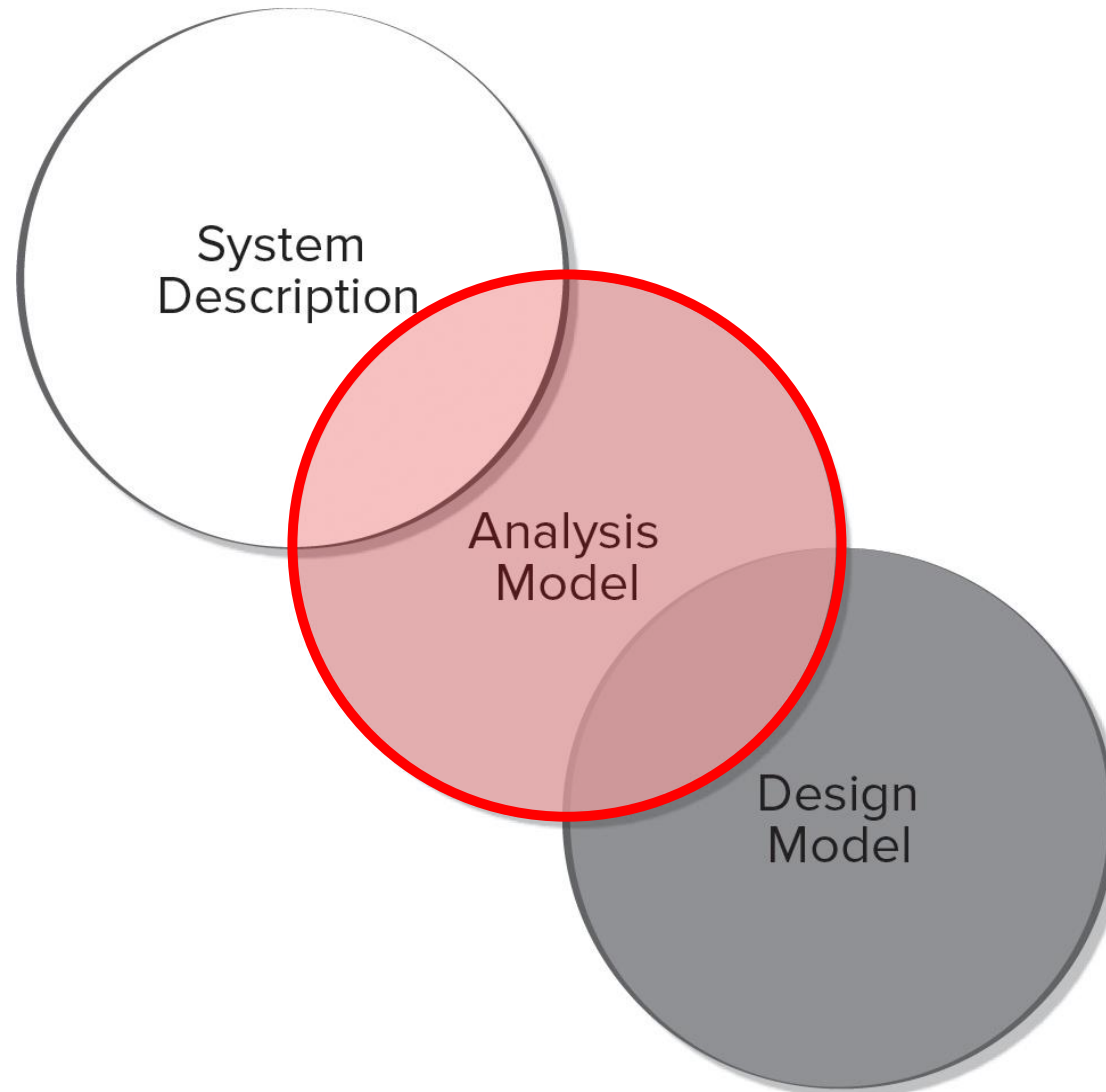


Figure 8.1 from your textbook.

Requirements Models

Scenario-Based Models: depict requirements from the point of view of various system “actors.”

Class-Oriented Models: represent object-oriented classes (*attributes and operations*) and how classes collaborate to achieve system requirements.

Behavioural Models: depict how the software reacts to internal or external “events.”

Data Models: depict the information domain for the problem.

Flow-Oriented Models: represent functional elements of the system and how they transform data in the system.

Requirements Models

Scenario-Based Models: depict requirements from the point of view of various system “actors.”

Class-Oriented Models: represent object-oriented classes (*attributes and operations*) and how classes collaborate to achieve system requirements.

Behavioural Models: depict how the software reacts to internal or external “events.”

Data Models: depict the information domain for the problem.

Flow-Oriented Models: represent functional elements of the system and how they transform data in the system.

Requirements Models

Scenario-Based Models: depict requirements from the point of view of various system “actors.”

Class-Oriented Models: represent object-oriented classes (*attributes and operations*) and how classes collaborate to achieve system requirements.

Behavioural Models: depict how the software reacts to internal or external “events.”

Data Models: depict the information domain for the problem.

Flow-Oriented Models: represent functional elements of the system and how they transform data in the system.

Requirements Models

Scenario-Based Models: depict requirements from the point of view of various system “actors.”

Class-Oriented Models: represent object-oriented classes (*attributes and operations*) and how classes collaborate to achieve system requirements.

Behavioural Models: depict how the software reacts to internal or external “events.”

Data Models: depict the information domain for the problem.

Flow-Oriented Models: represent functional elements of the system and how they transform data in the system.

Requirements Models

Scenario-Based Models: depict requirements from the point of view of various system “actors.”

Class-Oriented Models: represent object-oriented classes (*attributes and operations*) and how classes collaborate to achieve system requirements.

Behavioural Models: depict how the software reacts to internal or external “events.”

Data Models: depict the information domain for the problem.

Flow-Oriented Models: represent functional elements of the system and how they transform data in the system.

Requirements Models

Scenario-Based Models: depict requirements from the point of view of various system “actors.”

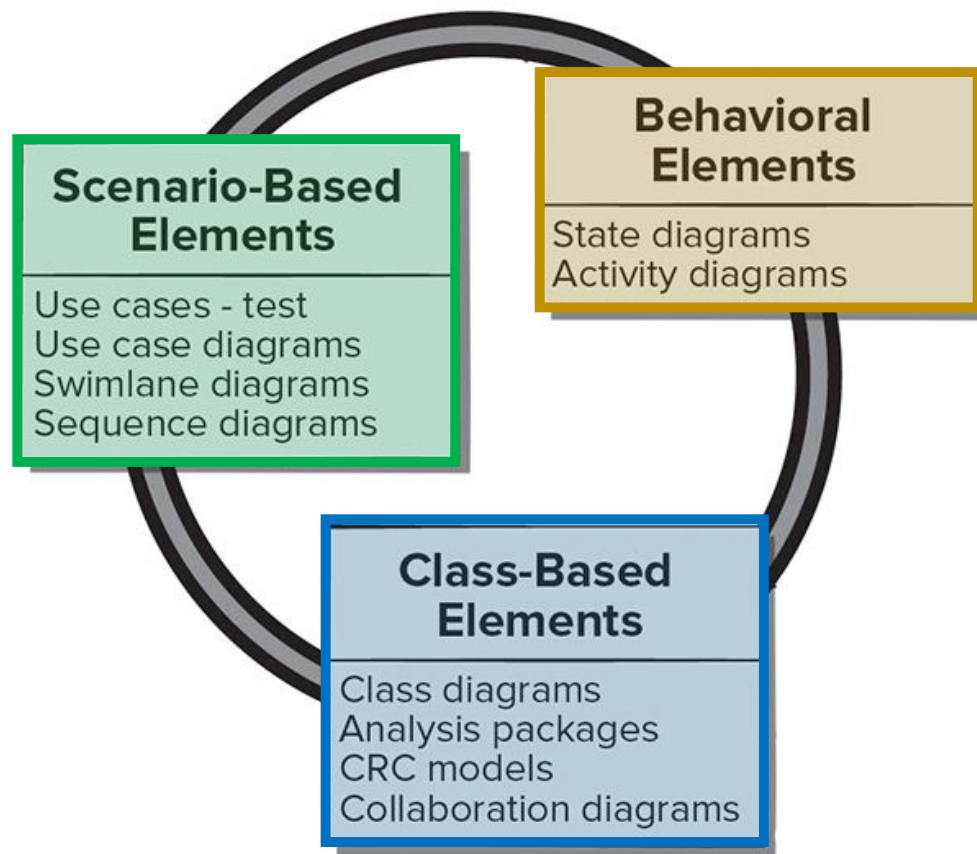
Class-Oriented Models: represent object-oriented classes (*attributes and operations*) and how classes collaborate to achieve system requirements.

Behavioural Models: depict how the software reacts to internal or external “events.”

Data Models: depict the information domain for the problem.

Flow-Oriented Models: represent functional elements of the system and how they transform data in the system.

Requirements Modeling Principles



Analysis Model

Rules of Thumb

- The **level of abstraction** should be **relatively high** - focus on requirements visible in problem or business domains.
- **Analysis model** should provide insight into **information domain**, **function**, and **behaviour** of the software.
- **Delay consideration of infrastructure and other non-functional models** until later in the modeling activity.
- The **analysis model** provides **value to all stakeholders**, keep the model as **simple as it can be**.

Requirements Modeling Principles

- **Principle 1:** The information domain of a problem must be represented and understood.
- **Principle 2:** The functions that the software performs must be defined.
- **Principle 3:** The behaviour of the software (as a consequence of external events) must be represented.
- **Principle 4:** The models that depict information, function, and behaviour must be partitioned in a manner that uncovers detail in a layered (or hierarchical) fashion.
- **Principle 5:** The analysis task should move from essential information toward implementation detail.