

Unified Modeling Language

Parinya Ekparinya

Parinya.Ek@kmitl.ac.th

Software Architecture and Design

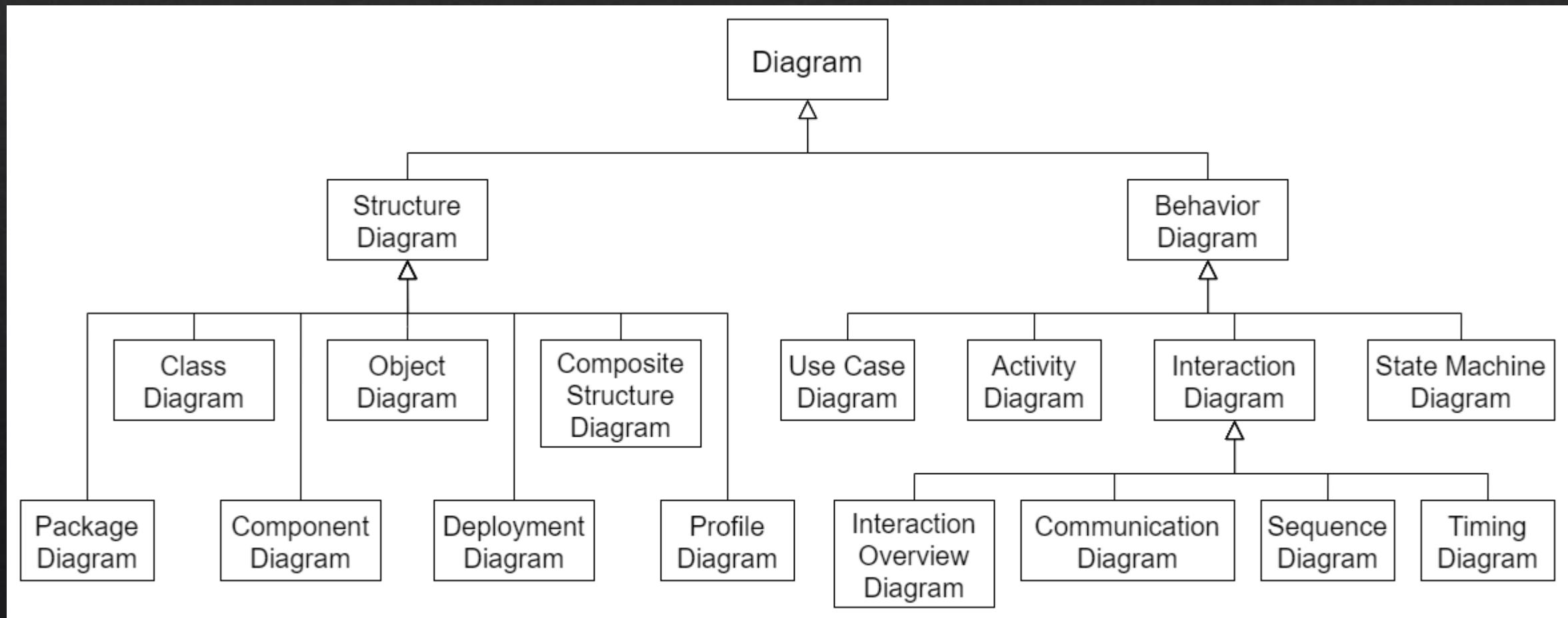
Outline

- ❖ Unified Modeling Language (UML)
- ❖ Class Diagram
- ❖ Activity Diagram
- ❖ Component Diagram
- ❖ Deployment Diagram

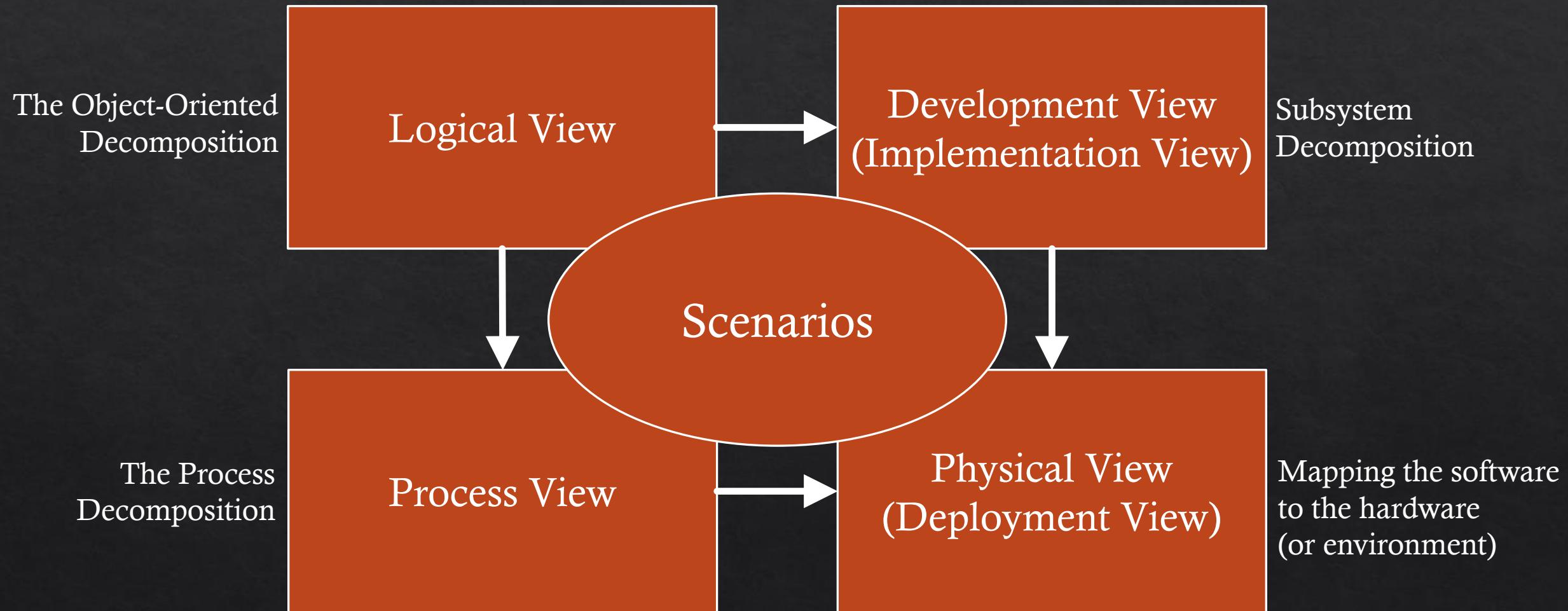
Unified Modeling Language (UML)

- ❖ The UML is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.
- ❖ The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components.
- ❖ The UML was developed at Rational Software in 1994–1995.
- ❖ The UML was also published by the ISO as an approved ISO standard in 2005.
- ❖ The latest version is 2.5, which is published in 2017.
For specification: <https://www.omg.org/spec/UML/2.5/About-UML/>

UML Diagram Types



4+1 View Model of Software Architecture



Kruchten, P. B. (1995). The 4+1 view model of architecture. *IEEE software*, 12(6), 42-50.

Logical View

- ❖ Logical view describes the abstract descriptions of a system's parts.
- ❖ This view is used to model what a system is made up of and how the parts interact with each other.
- ❖ The system is decomposed into a set of key abstractions in the form of objects or object classes. They exploit the principles of abstraction, encapsulation, and inheritance.
- ❖ With reference to UML, logical view typically contains **class diagrams**.

Process View

- ❖ Process view describes the processes within your system.
- ❖ A *process* is a grouping of tasks that form an executable unit.
- ❖ The software is partitioned into a set of independent *tasks*. A task is a separate thread of control, that can be scheduled individually on one processing node.
- ❖ Several styles would fit the process view, *pipes and filters* for example.
- ❖ With reference to UML, process view typically contains **activity diagrams**.

Development View

- ❖ Development view describes how your system's parts are organized into modules and components.
- ❖ Typically, this view focuses on the configuration management of a system; what components depend on what.
- ❖ The development view is also known as **implementation view**.
- ❖ A *layered style* would fit the process view. Each layer has a well-defined responsibility.
- ❖ With reference to UML, development view typically contains **component diagrams**.

Physical View

- ❖ Physical view describes how the system's components, as described in the other views, are arranged in their environments.
- ❖ The diagrams in this view show how the abstract parts map into the final deployed system.
- ❖ Thus, the physical view is also known as **deployment view**.
- ❖ With reference to UML, physical view typically contains **deployment diagrams**.

Scenarios

- ❖ Scenarios describe the functionality of the system being modeled from the perspective of its users and/or the outside world.
- ❖ The scenarios are in some sense an abstraction of the most important requirements.
- ❖ The scenarios are needed to describe what the system is supposed to do.
- ❖ This view is also known as use case view.
- ❖ This view is redundant with the other ones (hence the “+1”).
- ❖ With reference to UML, scenarios are typically expressed through **use case diagrams**.

Learning UML

- ❖ <https://www.omg.org/spec/UML/2.5/About-UML/>
- ❖ <https://www.uml-diagrams.org/>