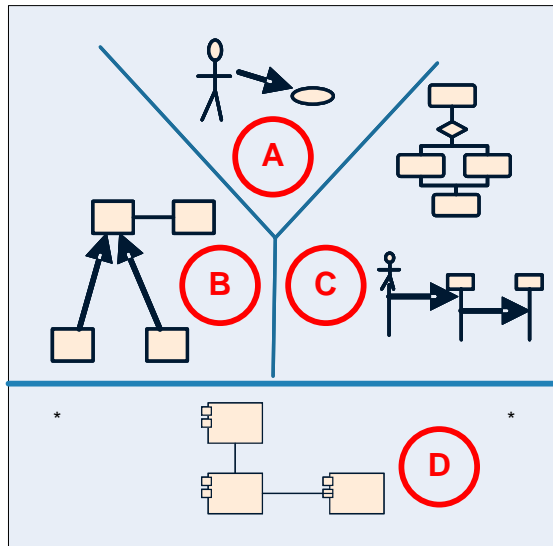


Introduction to UML 2.0 – Quick Reference Guide to Modeling



Where do I Model the Diagrams?

- A** **Use Case View**
Who does what?
 - Package diagrams
 - Use case diagrams
- B** **Logical View**
With what?
 - Package diagrams
 - Class diagrams
 - Object diagrams
 - Composite structure diagrams
- C** **Dynamic View**
How?
 - Package diagrams
 - Analysis diagrams
 - Activity diagrams
 - Sequence diagrams
 - Communication diagrams
 - Interaction overview diagrams
 - State machine diagrams
 - Timing diagrams
- D** **Deployment View**
Implemented by what software components?
 - Package diagrams
 - Component diagrams
 - Deployment diagrams

Rules for Modeling the Diagrams

- A** **Use Case Modeling**
 - Step one.** With all of the information available, identify the actors. Identify the roles (nouns) that are external to the system but uses the system
 - Step two.** Identify the functions internal to the system (verbs and nouns)
 - Step three.** Model the relationships between actors and use cases – associations (the only relationship that may exist between an actor and a use case)
 - Step four.** Model relationships between use cases:
 - Extend – conditional execution of extending use case
 - Include – compulsory execution of included use case
 - Generalization – (specialized use case) Type of (generalized use case). Generalized use case contains common behaviour inherited by the specialized use cases
- B** **Class Modeling**
 - Finding classes:** If the use case diagrams are defined, the classes may be derived from the existing use cases:
 - The noun from the use case – class
 - The verb from the use case – operation
 - The rules of relationships:**
 - Rule 1:** If it is part of, have parts and the parts will still exist if the parent is destroyed - Aggregation
 - Rule 1a:** If the parts cannot exist independently after the parent is destroyed – Composition
 - Rule 2:** If it is a type of another class, or a subtype of another class - Generalization
 - Rule 3:** If not rule 1 or 2 then there is a 98% probability that it is an association
- B** **Object modeling**
 - Definition:** An object is an instance of a class. A class must exist **first**, before an object can be instanced from that class
- C** **Activity Modeling**
 - **End to end process modeling:** Activity = Execution of function. This implies: Activity name = Use case name
 - **Scenario modeling:** 1 activity diagram to explain all scenarios
- C** **Sequence modeling**
 - End to end system process modeling:** Interaction overview diagram
 - Scenario modeling:** 1 sequence diagram per scenario
- C** **Life Cycle Modeling**
 - A life cycle is attached to a class and describes all of the states that class may be in, in its life