Introduction to Object-Oriented Analysis and Design

Curt Clifton

Rose-Hulman Institute of Technology

Q1,2

Book Organization

- Inception Phase
- Iteration 1
  - OO Analysis
  - OO Design
  - Translating Designs to Code
- Iteration 2
- Iteration 3

Focused on requirements analysis, so we'll mostly skip

Introduces design patterns

Architectural analysis, framework design, etc.

Owning a hammer doesn’t make one an architect

Knowing how to “think in objects” is critical

Q3

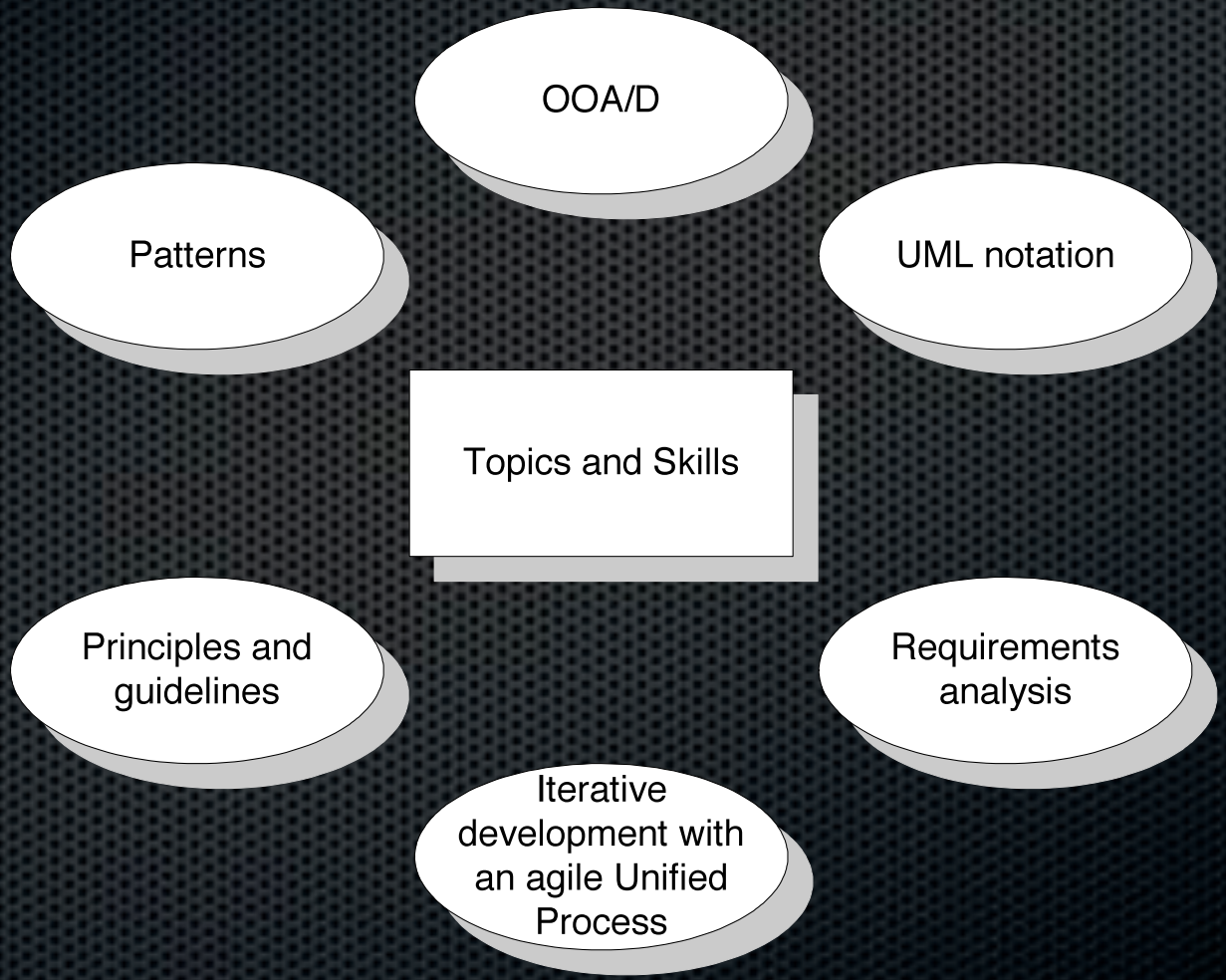
Key Questions to Help Us Think in Objects

Responsibility-driven Design

- How should responsibilities be allocated to classes?
- How should objects collaborate?
- What classes should do what?

Guided by **patterns**

Book Topics


OOA/D
Patterns
UML notation
Topics and Skills
Principles and guidelines
Requirements analysis
Iterative development with an agile Unified Process

A critical ability in OO development is to skillfully assign responsibilities to software objects.

Q6

Analysis vs. Design

- Analysis:
  - Investigation of the **problem** and requirements, rather than a solution
- Design:
  - A **conceptual solution**, rather than its implementation

Excludes low-level details

Q7

Cartoon of the Day!

http://xkcd.com/516/

You didn't run a chemical **analysis** against the Shroud of Turin? Man, all that work for NOTHING.

Example: Flight Information System

- Analysis:
  - What are some concepts in the problem domain?
- Design:
  - What are some attributes and methods that a Plane object might have?

Example: Dice Game

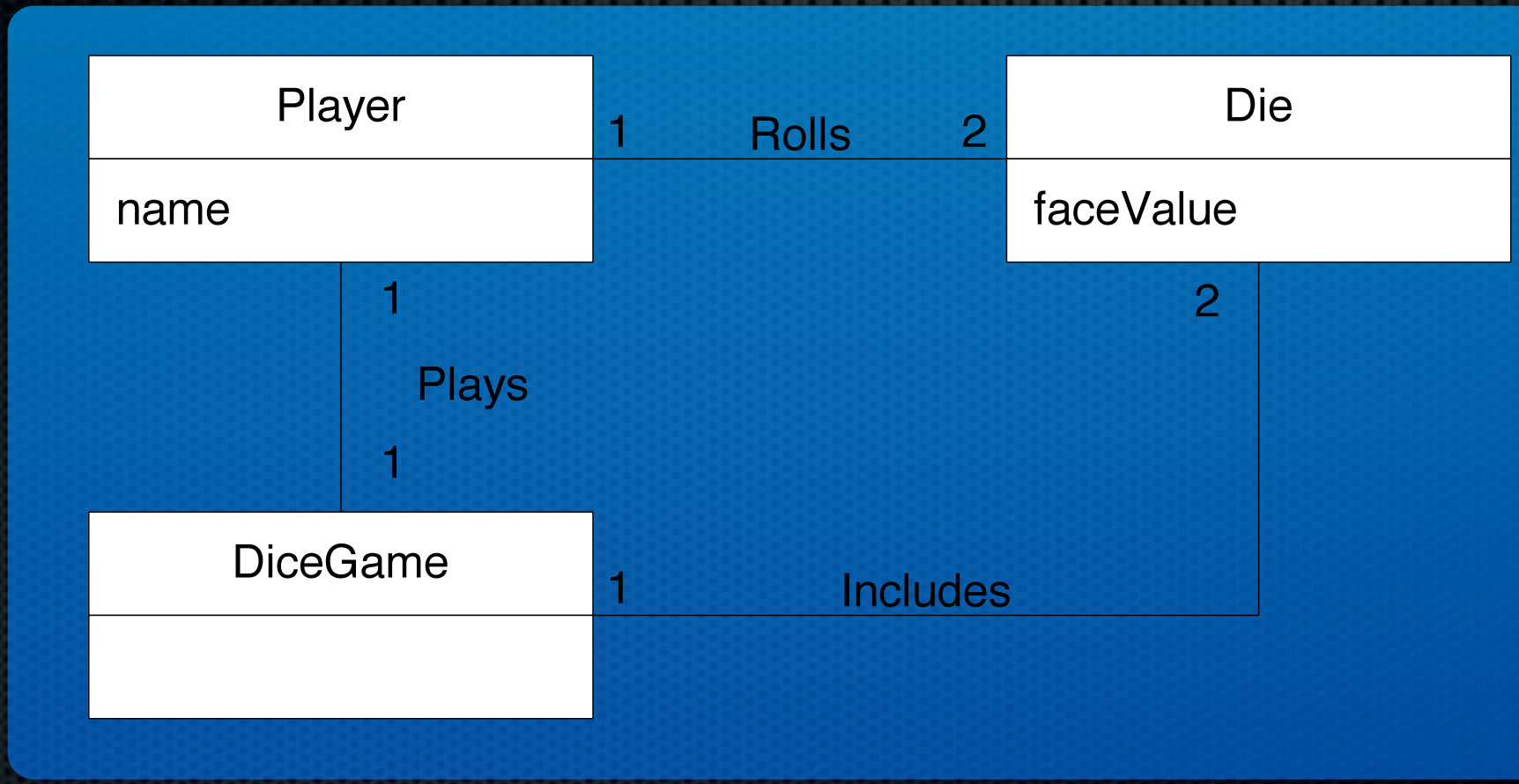
- Define Use Cases
- Define a Domain Model
- Assign Object Responsibilities, Draw Interaction Diagrams
- Define Design Class Diagrams

Dice Game: Define Use Cases

- Play a dice game: Players requests to roll the dice. System presents results: If the dice face value totals seven, player wins; otherwise player loses

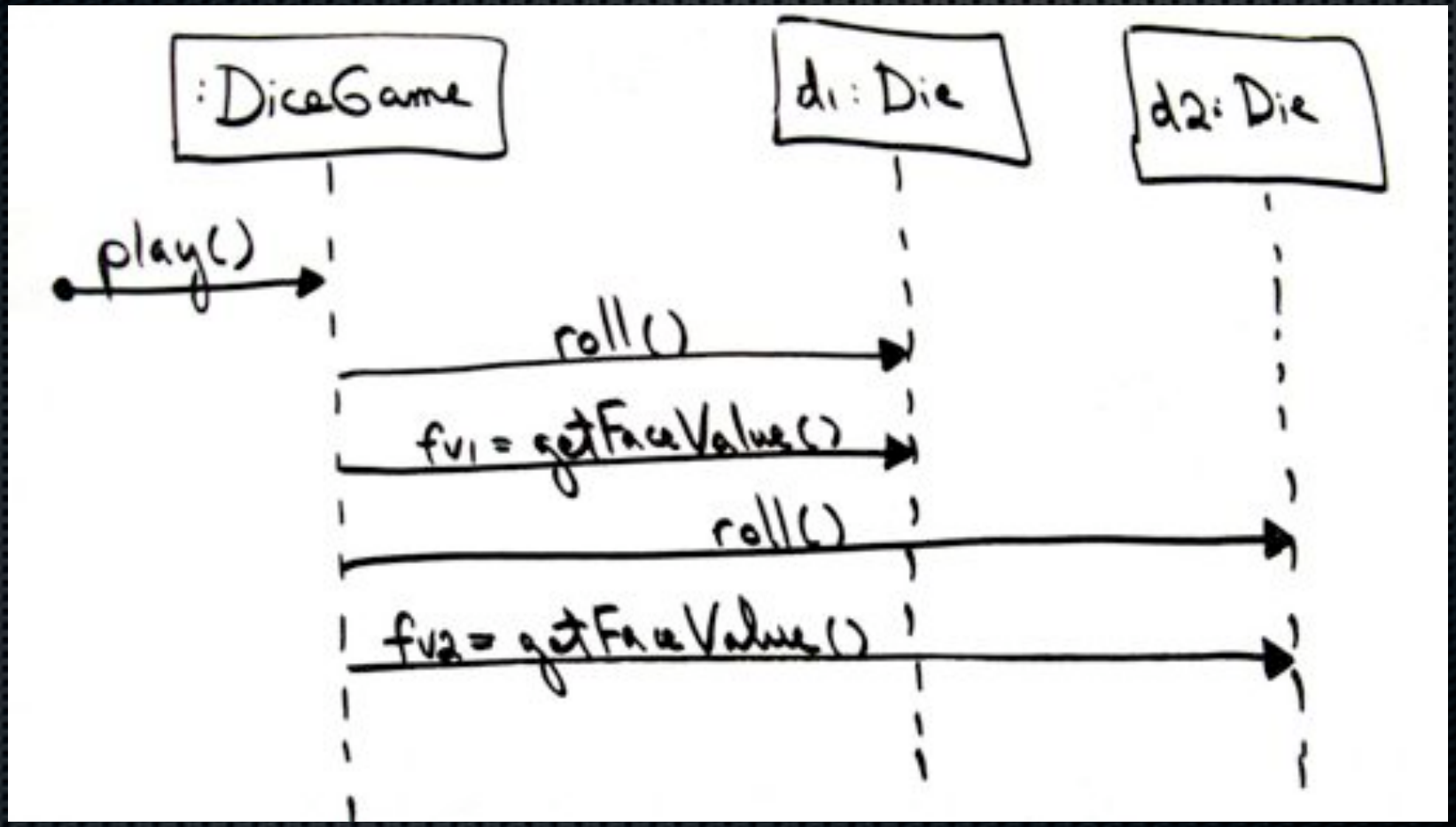
Dice Game: Define a Domain Model

Shows **noteworthy** domain concepts or objects

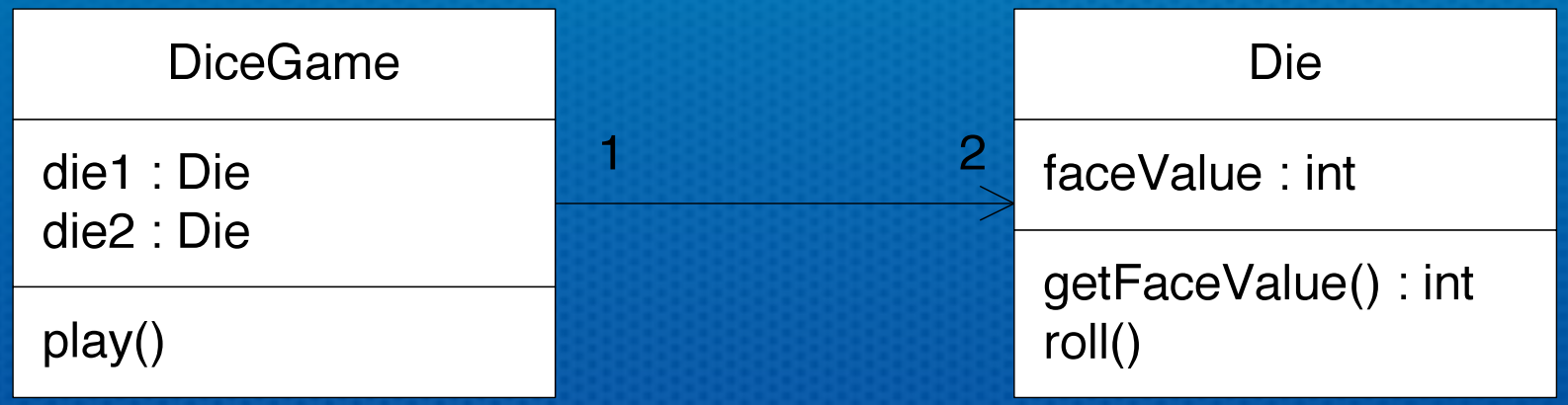


Dice Game: Define Interaction Diagrams

Moving into **design** now!



Dice Game: Define Design Class Diagrams



Note similarities to domain model. OO designs and languages **lower** the **representation gap**.

Exercise

- Read the problem scope and use case on the handout
- Answer the quiz questions

Q8,9

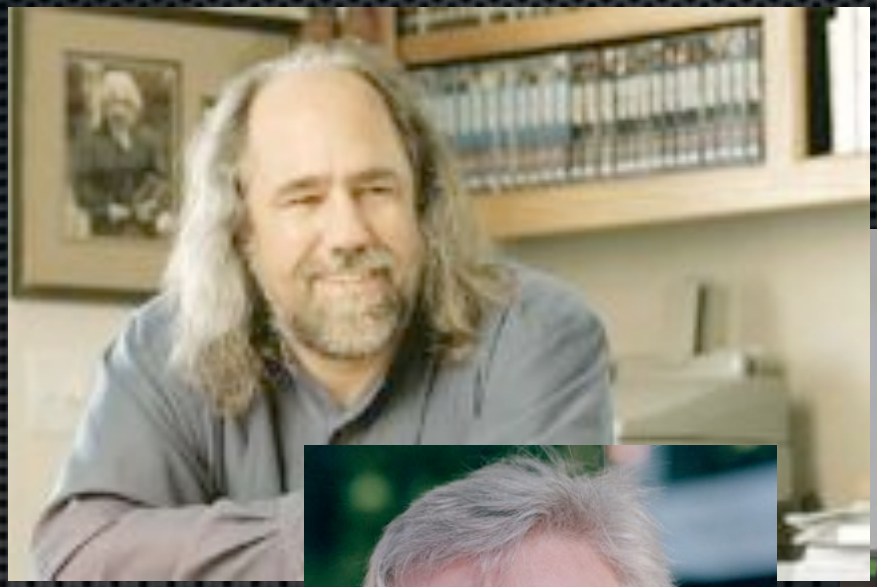
Unified Modeling Language

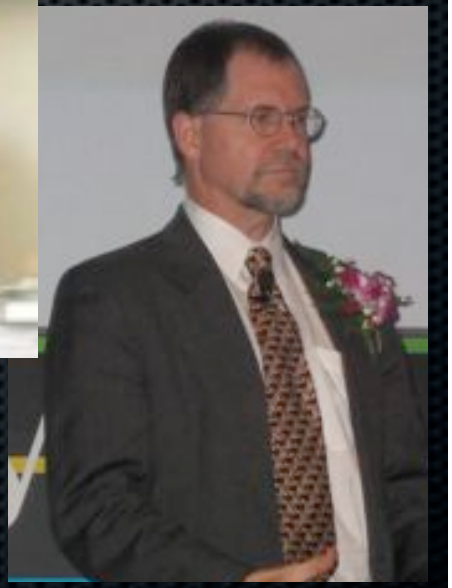
The Three Amigos

- Grady Booch
- Jim Rumbaugh
- Ivar Jacobson

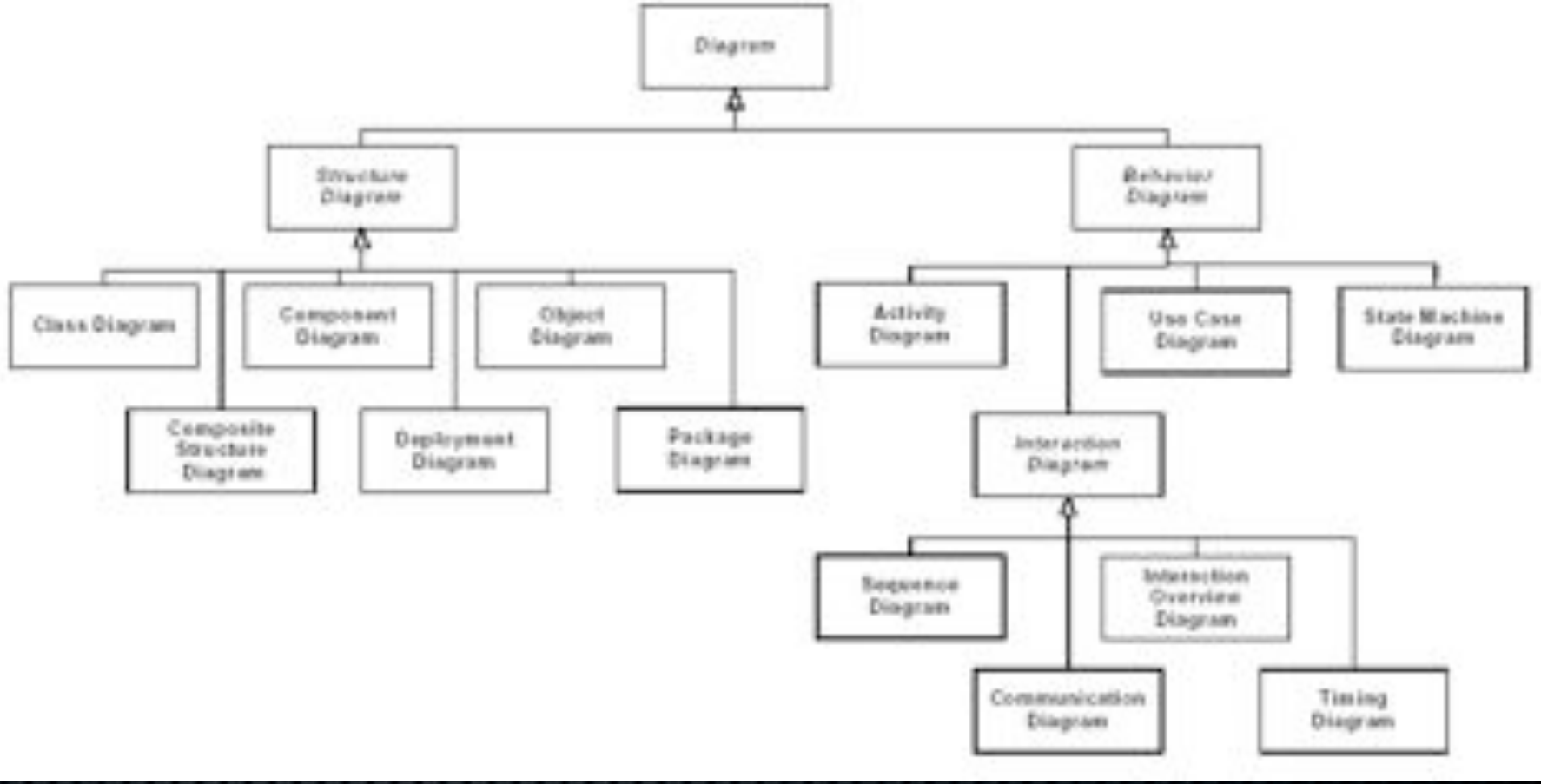
The Three Amigos

- Grady Booch
- Jim Rumbaugh
- Ivar Jacobson





A Union of Notations



Three Ways to Apply UML

- As sketch
- As blueprint
  - Reverse engineering
  - Forward engineering
- As programming language

Three Perspectives for Applying UML

- **Conceptual perspective**: real-world concepts in the problem domain
- **Software specification perspective**: not committed to a particular implementation language
- **Software implementation perspective**: technology specific (e.g., Java access modifiers, etc.)

The Meaning of “Class”

- Conceptual class
- Software class
- Implementation class

Q10

Don’t Forget

- Homework 1 due today at 5 p.m.
- Milestone 1 due Friday at 11:59 p.m.
- Reading for next time (see schedule)
- Team meetings as scheduled