Design and Analysis Intro

Software Design and Analysis

CSCI 2040

About me..and course..

- http://data.science.uoit.ca

  -> Home

  -> Teaching -> Software Design and Analysis

Objectives

- Describe the goals.
- Define object-oriented analysis and design (OOA/D).
- Illustrate a brief OOA/D example.
- Overview UML and visual agile modeling.
- Course structure.

What Will You Learn? Is it Useful?

- Learn core skills in object-oriented analysis and design (OOA/D).
- These skills are essential for the creation of
  - well-designed, robust, and maintainable software using OO technologies and languages such as Java or C#.
  - The proverb "owning a hammer does not make one an architect" is especially true with respect to object technology..
  - Knowing an object-oriented language (such as Java) is a necessary but insufficient first step to create object systems.

Think in Objects

- Knowing how to "think in objects" is critical!
  - This is an introduction to OOA/D while applying the Unified Modeling Language (UML) and patterns.
  - And, to iterative development, using an agile approach to the Unified Process as an example iterative process.
  - The material progresses to some framework design and architectural analysis.

UML vs. Thinking in Objects

- The course is not just about UML.
  - The **UML** is a standard diagramming notation.
- Common notation is useful, but there are more important OO things to learn especially, how to think in objects.
  - It is useless to learn UML and a UML CASE tool, but not really know how to create an excellent OO design, or evaluate and improve an existing one.
  - Yet, we need a language for OOA/D and "software blueprints," as a form of communication.

Responsibilities

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

These are critical questions in the design of a system...

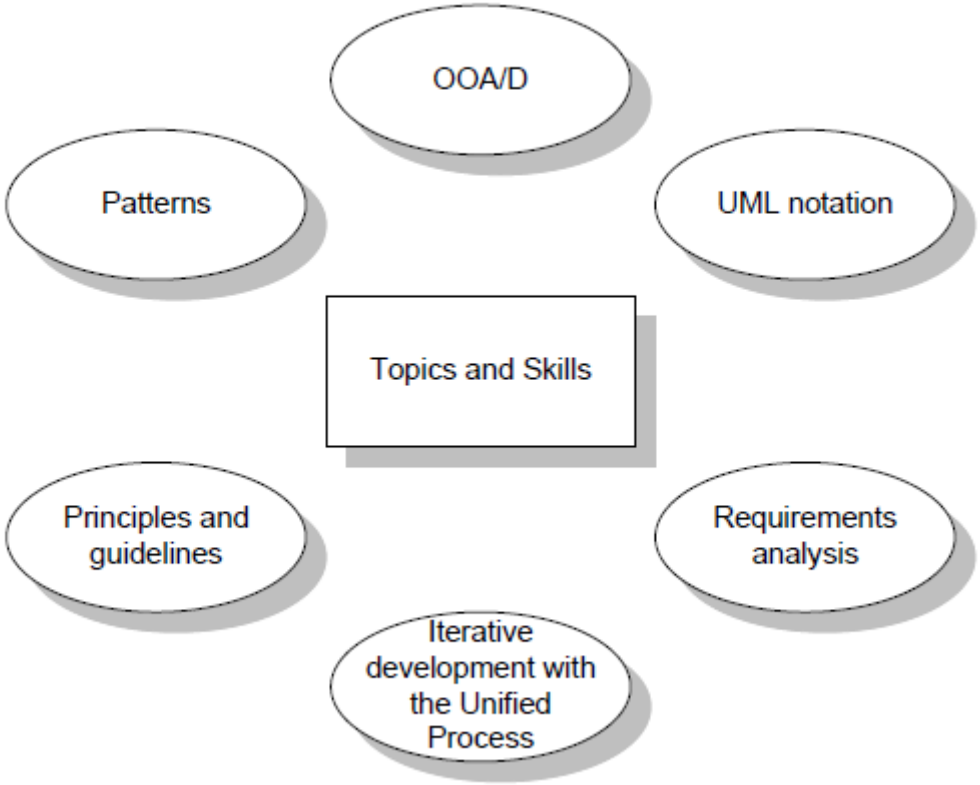
OOD: Principles and Patterns

- Certain tried-and true solutions to design problems can be (and have been) expressed as best-practice principles, heuristics, or **patterns**
  - named problem-solution formulas that codify exemplary design principles.

Iterative Development, Agile Modeling and Agile UP

- An **agile** (light, flexible) approach to the well-known **Unified Process** (UP) is used as the *sample* **iterative development process** within which these topics are introduced.
- Learning them in the context of an agile UP does not invalidate their applicability to other methods, such as Scrum, Feature-Driven Development, Lean Development, Crystal Methods, and so on..

Topics and Skills Covered



The Most Important Learning Goal ("desert island" skill..)

- A critical ability in OO development is to skillfully assign responsibilities to software objects.
  - Why? Because it is one activity that must be performed either while drawing a UML diagram or programming..
- Nine fundamental principles in object design are presented and applied.
  - They are organized in a learning aid called **GRASP**

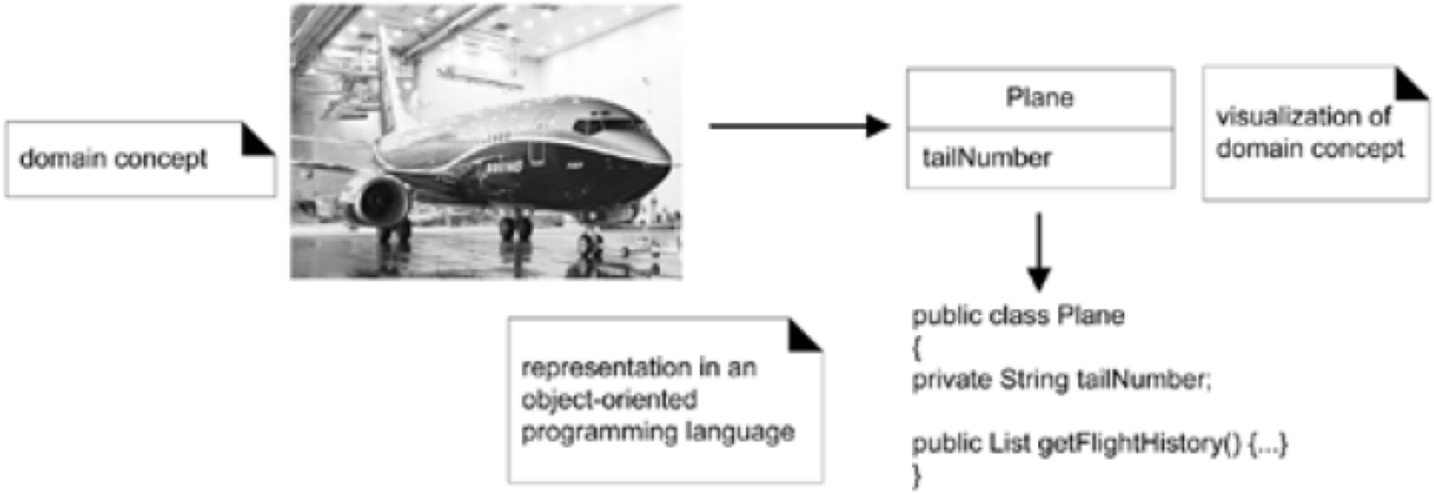
What is Analysis and Design?

- **Analysis** emphasizes an *investigation* of the problem and requirements, rather than a solution.
  - For example, if a new online trading system is desired, how will it be used? What are its functions?
- **Design** emphasizes a *conceptual solution* (in software and hardware) that fulfills the requirements.
  - For example, a description of a database schema and software objects.

What is Object-Oriented Analysis and Design?

- During **object-oriented analysis** there is an emphasis on finding and describing the objects or concepts in the problem domain.
  - For example, in the case of the flight information system, some of the concepts include *Plane, Flight,* and *Pilot*.
- During **object-oriented design** there is an emphasis on defining software objects and how they collaborate to fulfill the requirements.
  - For example, a *Plane* software object may have a *tailNumber* attribute and a *getFlightHistory* method

Object-orientation emphasizes representation of objects



- Finally, during implementation or object-oriented programming, design objects are implemented, such as a *Plane* class in Java.

Short Example

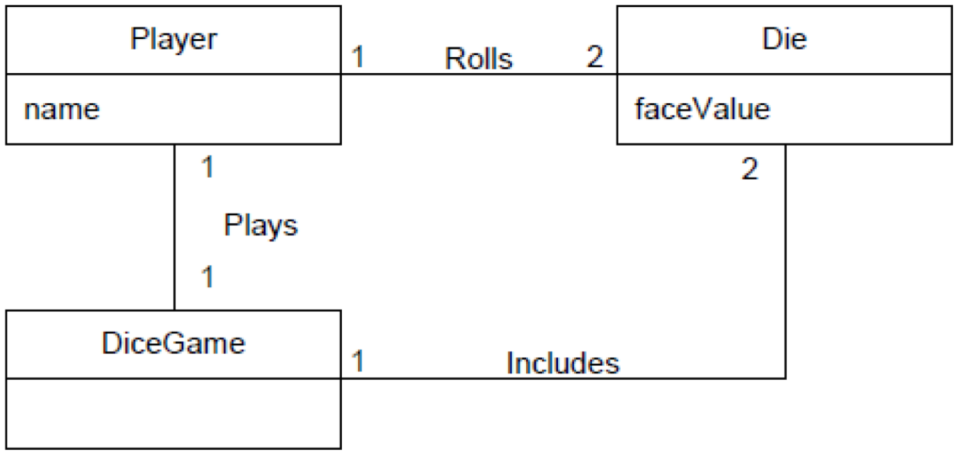
- Bird's-eye view of a few key steps and diagrams, using a simple example a "dice game" in which software simulates a player rolling two dice.
  - If the total is seven, they win; otherwise, they lose.

Define Use Cases

- Use cases are simply written stories.
- For example, here is a brief version of the *Play a Dice Game* use case:
  - **Play a Dice Game:** A player picks up and rolls the dice. If the dice face value total seven, they win; otherwise, they lose.

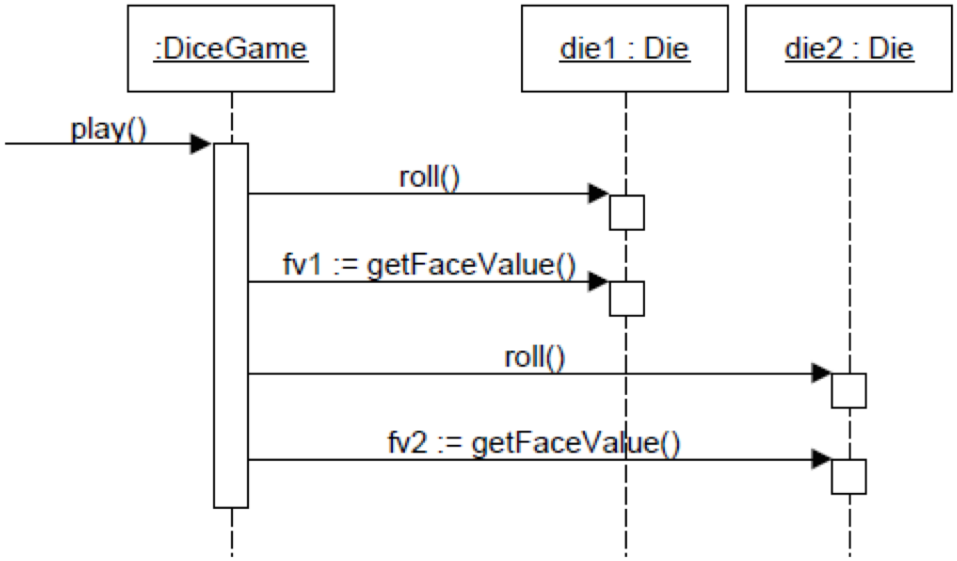
Define a Domain Model

- Defining a domain model involves an identification of the concepts, attributes, and associations that are considered noteworthy.

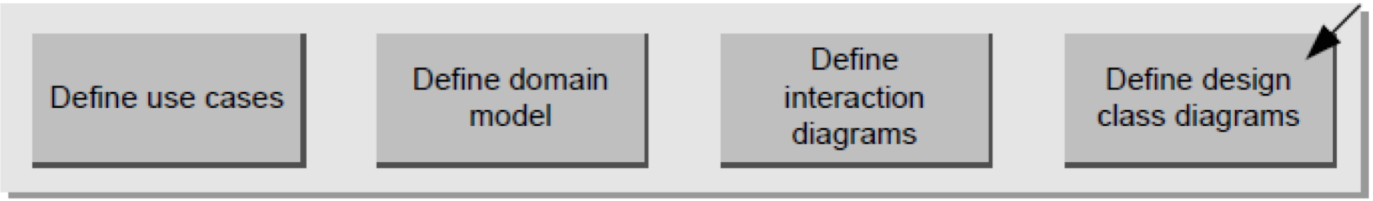


Define Interaction Diagrams

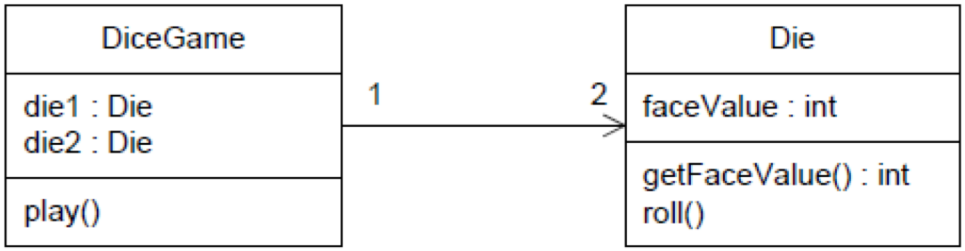
- Interaction Diagrams show the flow of messages between software objects, and thus the invocation of methods.



Define Design Class Diagrams

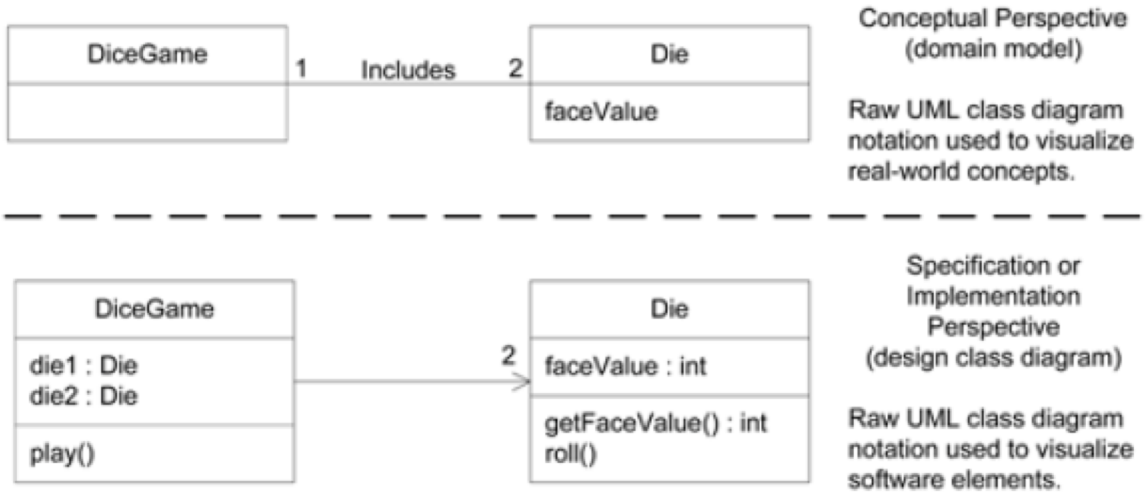


- In addition to a *dynamic* view of collaborating objects shown in interaction diagrams,
  - it is useful to create a *static* view of the class definitions with a **design class diagram.**



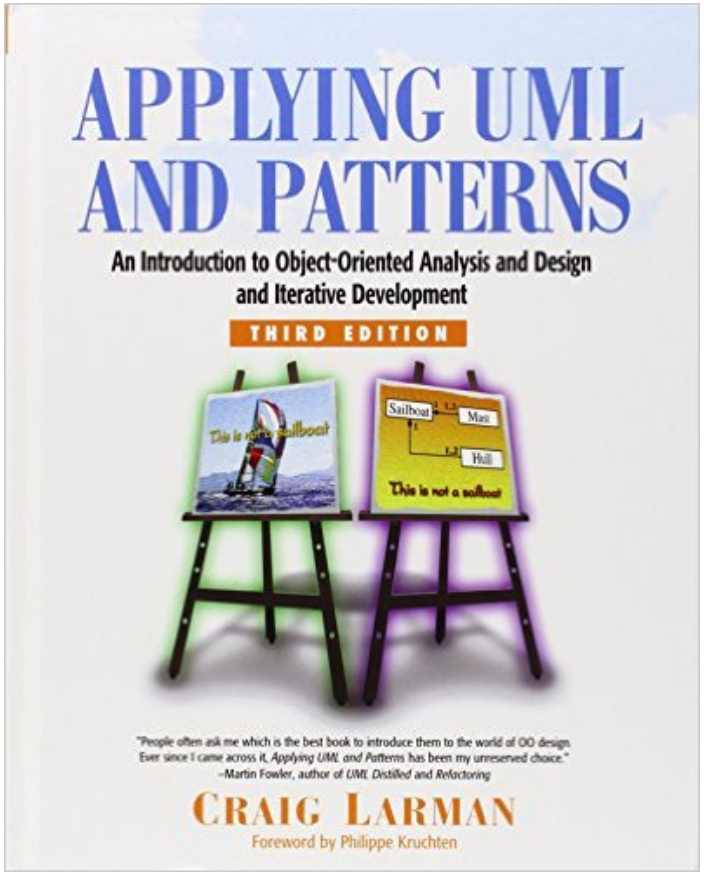
Different Perspectives with UML

- The Unified Modeling Language is a visual language for specifying, constructing and documenting the artifacts of systems..

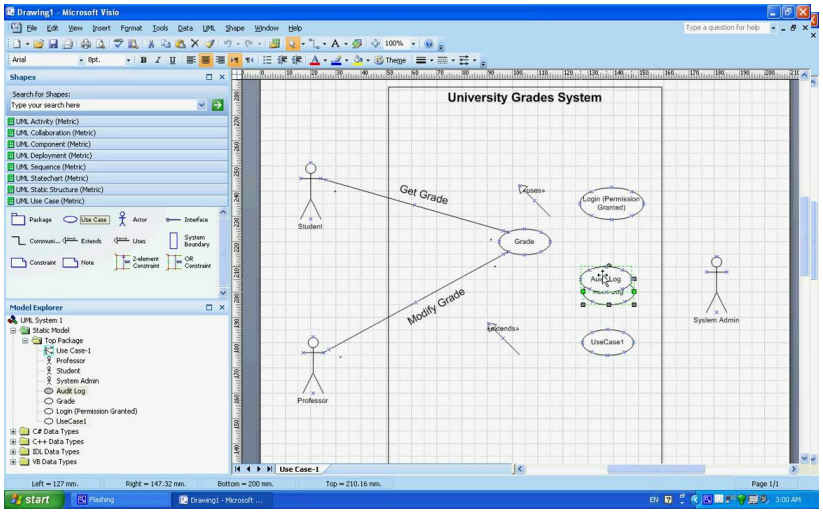


Textbook

- *Applying UML and Patterns*, Craig Larman



Equipment


Drawing1 - Microsoft Visio
University Grades System
Get Grade
Modify Grade
Student
Professor
Grade
Login (Permission Granted)
Audit Log
System Admin
UseCase1

Structure of the Course

- Lectures:
  - Twice a week
- Labs:
  - Once a week (**First lab in the week of 21st of January**)
  - 10 Labs
- Midterm I (in the middle of the term):
- Final Midterm (in the end of the term)
- Marking
  - Labs and Project: 30% (10%: 10 x 1% each lab + 20%: 2 x 10% project reports)
  - Midterm: 20%
  - Final Midterm: 40%
  - Participation and Presentation 10%

Communication

- Blackboard for posting Labs/Midterms/Exam/Marking
- Take time when composing a message - think of it as a professional message to a co-worker.
  - There will not be space for SMS-speak in your work life.
- Slack for discussions?
- Use e-mail/Slack for correspondence

jaroslaw.szlichta@uoit.ca

Actions

- Read Chapter 1
  - *Applying UML and Patterns*, Craig Larman
- Install Microsoft Visio unless you already have it
  - MSDN DreamSpark Premium
    - https://uoitsci.onthehub.com/WebStore/Security/SignIn.aspx
    - Ask TA: Spencer Bryson spencer.bryson@uoit.net in case of any troubles with the installation (or other TAs)