OntarioTech
UNIVERSITY

Software Design and Analysis
CSCI 2040U

Week 1.1 – Course and Topic Introduction

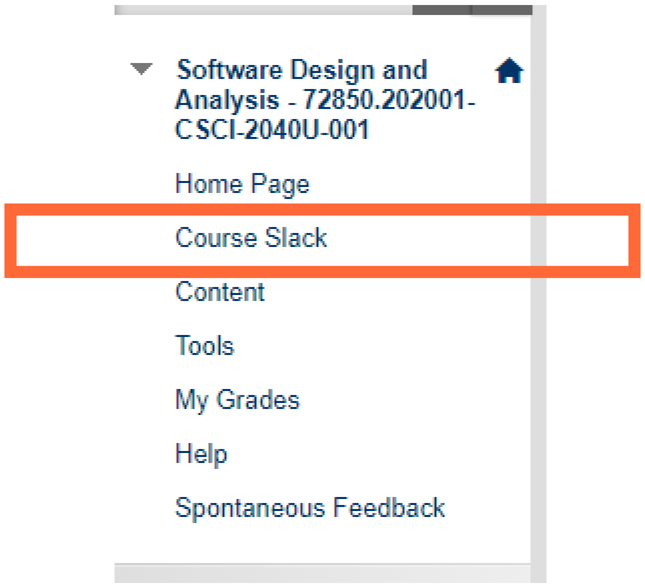
Mariana Shimabukuro

Acknowledgement: Parts of this course materials are based on material prepared by Jarek Szlichta, and Randy Fortier.

Before we start

- Course Slack

ontariotech-csci2040.slack.com


Software Design and Analysis - 72850.202001-CSCI-2040U-001

Home Page

Course Slack

Content

Tools

My Grades

Help

Spontaneous Feedback

Today we will…

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

The course

Software Design and Analysis: what does it mean?

Software Design and Analysis (big picture)

- Software and systems engineering
  - User, client and system requirements analysis
  - Solution design / system architecture
    - Principles and guidelines
      - Industry standards
      - Language and framework specific
    - Software pieces: scripts, algorithms, integration of technologies and services
      - Examples integrated technologies: e-mail server (G-Mail), cloud application and services (AWS, G-Suite), internet provider, databases, data warehouses, deployment and building tools (Gradle, docker, Git, Kubernetes, …), etc.
  - Testing and software quality assurance (SQA)
  - Deployment and maintenance

Software Design and Analysis (our focus)

- Object-Oriented Design and Analysis (OOD/A)
  - Object-Oriented Programming (OOP) technologies and languages.
    - Classic examples: Java or C#
  - Key skill: assignment of responsibilities to software objects

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

– Larman (p. 6).

Software Design and Analysis

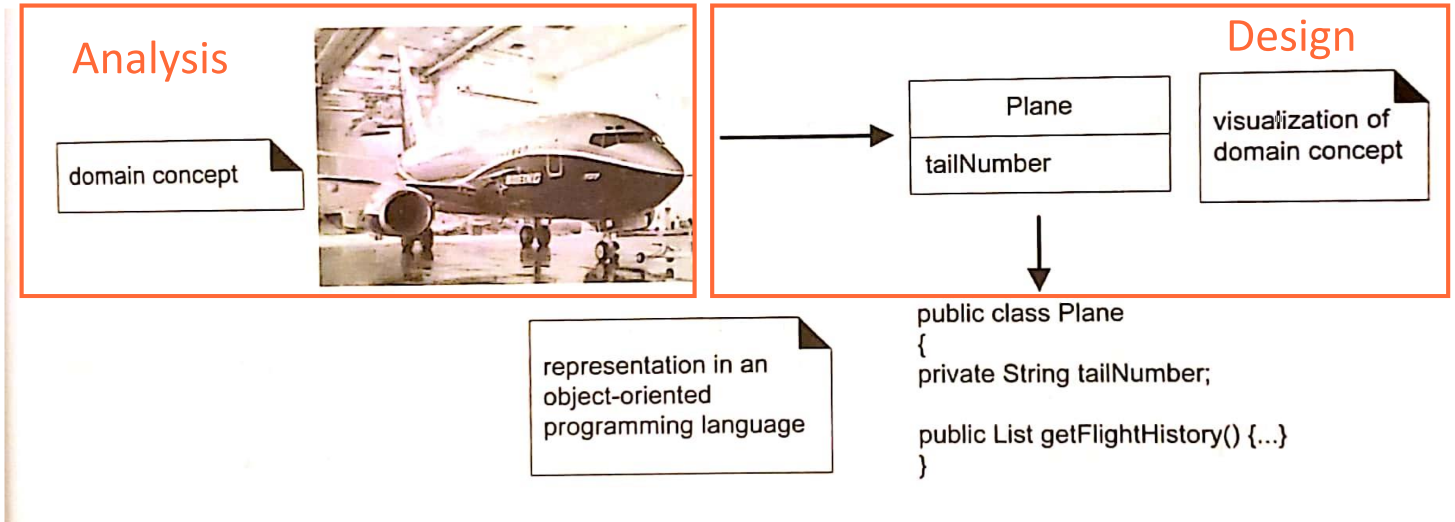
- **Analysis** emphasizes in the investigation of the problem.
  - *Requirement analysis*: investigation of what the problem requires the solution to have.
  - *Object-oriented analysis*: investigation of how many and what objects are needed to represent the domain.
- **Design** emphasizes in abstracting the conceptual solution that fulfills the requirements, rather than its implementation. However, designs can be implemented (such as code).

"…do the right thing (analysis), and do the thing right (design)."

– Larman (p. 7).

Object-oriented analysis and design

- Object-orientation emphasizes representation of objects
- During implementation or object-oriented programming, design objects are implemented, such as a *Plane* class in Java.


Analysis
domain concept
Design
Plane
tailNumber
visualization of domain concept
representation in an object-oriented programming language

```
public class Plane
{
private String tailNumber;

public List getFlightHistory() {...}
}
```

Figure 1.2 Object-orientation emphasizes representation of objects.

Thinking in Objects

What you need to learn

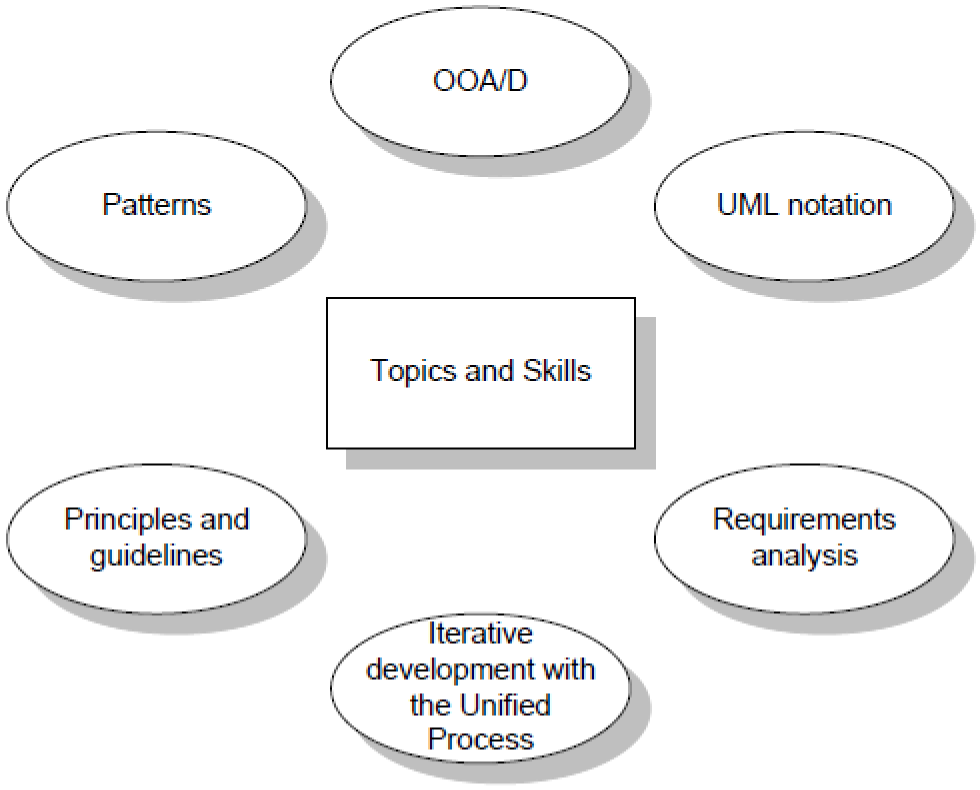
Knowing how to "think in objects" is critical

- This course 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
- Along the semester the course 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 concepts 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 to evaluate and improve an existing one.
  - Yet, we need a language for OOA/D and “software blueprints”, as a form of communication.

Topics and Skills


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

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 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..

Course overview

How we are going to learn

About the instructor

Shimabukuro, Mariana Akemi

- I work with data visualization and educational applications for language learning; more specifically English as a second language.

- Ontario Tech University, Ph.D. candidate (currently)
- Ontario Tech University, M.Sc. (2017)
- Universidade Estadual Paulista - Brazil, B. of CS (2015)

- First industry position in 2009 working with Delphi for a retail system.

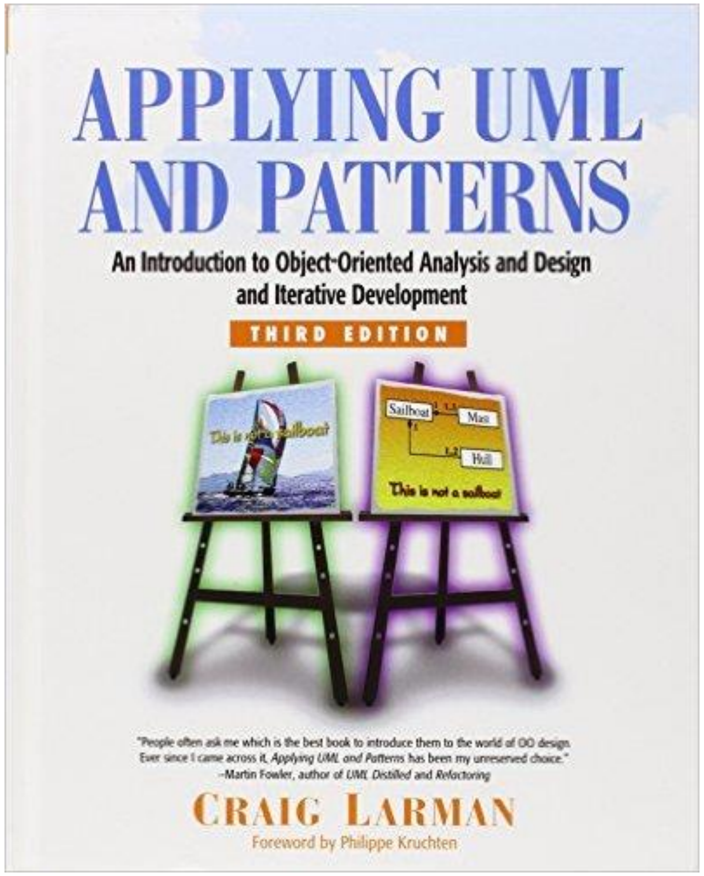
More about me and my work at: makemis.com

Learning outcomes

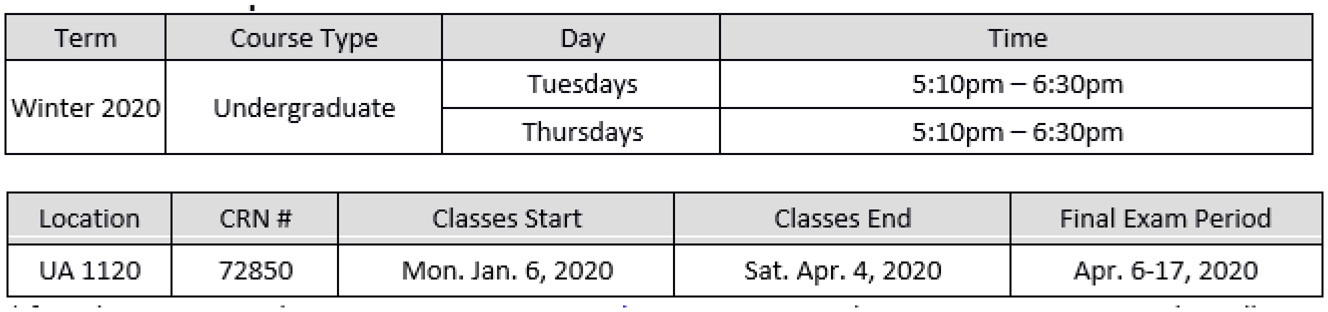
- Use a software process commonly used in industry
- Create UML diagrams to describe requirements and software design
- Understand common software architecture and design patterns
- Create requirements documentation for case study applications
- Design software for case study applications

Course textbook

- *Applying UML and Patterns*, Craig Larman

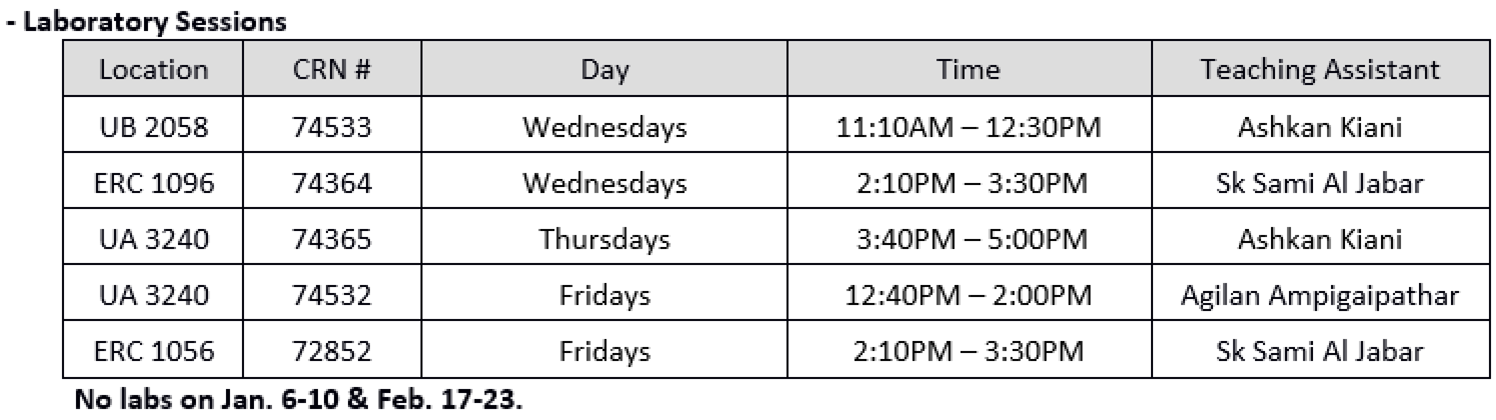

APPLYING UML AND PATTERNS
An Introduction to Object-Oriented Analysis and Design and Iterative Development
THIRD EDITION
This is not a sailboat
Sailboat
Mast
Hull
This is not a sailboat
"People often ask me which is the best book to introduce them to the world of OO design. Ever since I came across it, Applying UML and Patterns has been my unreserved choice."
–Martin Fowler, author of UML Distilled and Refactoring
CRAIG LARMAN
Foreword by Philippe Kruchten

Lectures and Labs



Term	Course Type	Day	Time
Winter 2020	Undergraduate	Tuesdays	5:10pm – 6:30pm
		Thursdays	5:10pm – 6:30pm

Location	CRN #	Classes Start	Classes End	Final Exam Period
UA 1120	72850	Mon. Jan. 6, 2020	Sat. Apr. 4, 2020	Apr. 6-17, 2020



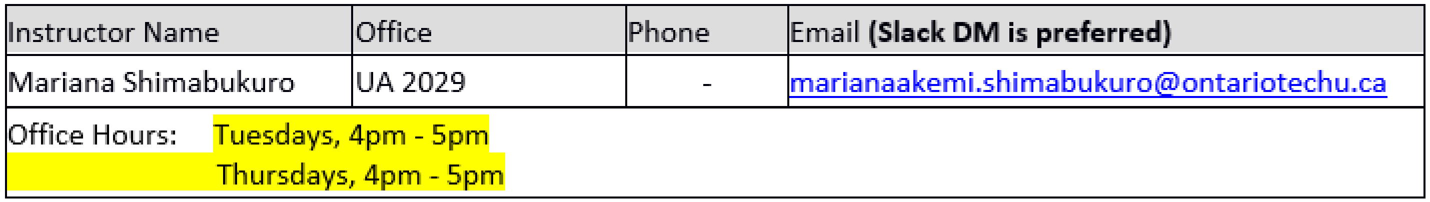
- Laboratory Sessions

Location	CRN #	Day	Time	Teaching Assistant
UB 2058	74533	Wednesdays	11:10AM – 12:30PM	Ashkan Kiani
ERC 1096	74364	Wednesdays	2:10PM – 3:30PM	Sk Sami Al Jabar
UA 3240	74365	Thursdays	3:40PM – 5:00PM	Ashkan Kiani
UA 3240	74532	Fridays	12:40PM – 2:00PM	Agilan Ampigaipathar
ERC 1056	72852	Fridays	2:10PM – 3:30PM	Sk Sami Al Jabar

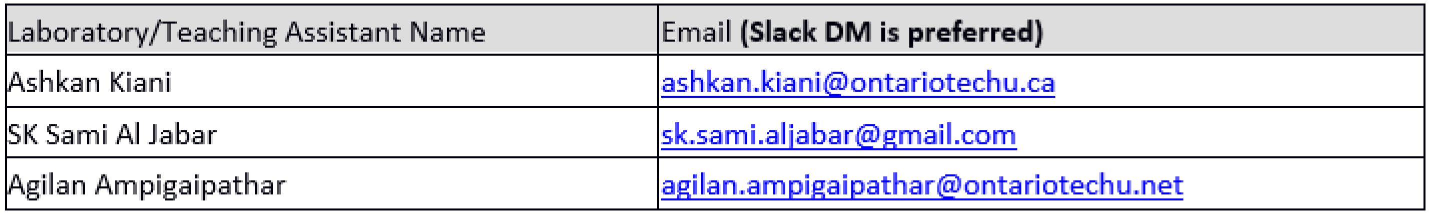
No labs on Jan. 6-10 & Feb. 17-23.

Contact and office hours

- **Slack workspace:** https://join.slack.com/t/ontariotechcs-idy6536/signup
  - Please use SLACK for faster response.
  - If you have any questions about Slack, or you are unfamiliar with the application, please, contact the Tas during your lab session about it. It will be important for this course.
  - TA office hours will be announced via Slack and Blackboard announcements.



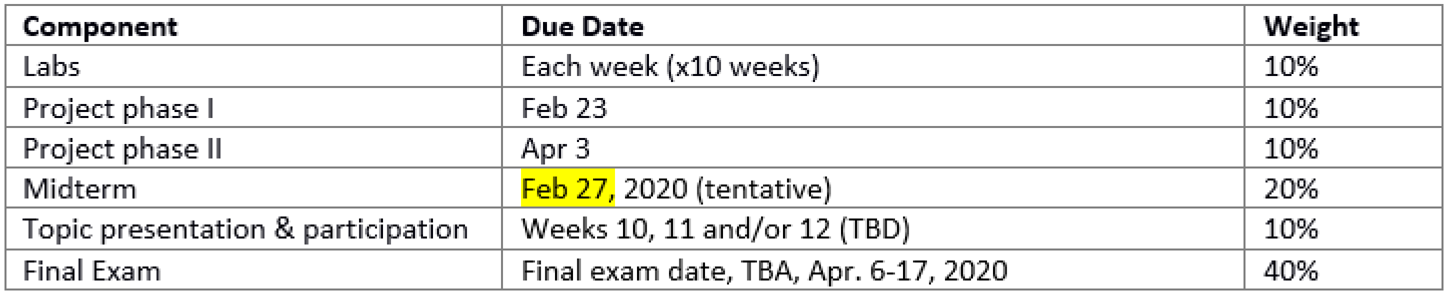
Instructor Name	Office	Phone	Email **(Slack DM is preferred)**
Mariana Shimabukuro	UA 2029	-	marianaakemi.shimabukuro@ontariotechu.ca
Office Hours: Tuesdays, 4pm - 5pm Thursdays, 4pm - 5pm			



Laboratory/Teaching Assistant Name	Email **(Slack DM is preferred)**
Ashkan Kiani	ashkan.kiani@ontariotechu.ca
SK Sami Al Jabar	sk.sami.aljabar@gmail.com
Agilan Ampigaipathar	agilan.ampigaipathar@ontariotechu.net

Course components

- Labs start in the week of January 13th.
- The tests will be written exams.
- The final is accumulative.



Component	**Due Date**	**Weight**
Labs	Each week (x10 weeks)	10%
Project phase I	Feb 23	10%
Project phase II	Apr 3	10%
Midterm	Feb 27, 2020 (tentative)	20%
Topic presentation & participation	Weeks 10, 11 and/or 12 (TBD)	10%
Final Exam	Final exam date, TBA, Apr. 6-17, 2020	40%

Labs (10%)

- **Topic:** Designing Online Food Ordering System
- **Start date:** week of January 13th
- **Labs description:** Each lab will contain a short system modelling component related to the course project.
  - You will use any software for diagramming, suggested MS Visio.
  - Lab assignments will be graded as part of the project deliverables.
  - Attendance will be taken as participation mark of each session 10x1%.
  - Lab assignments will be made available via Blackboard weekly.

Course project (20%)

- **Topic:** Designing Online Food Ordering System

- **System Description:** In this project you should design an online shopping system. The system should *resemble* the Uber Eats shopping system available at <ubereats.com>. The system should scale well to many users and provide web-based interface.

- Submission via Blackboard **(PDF only)**
  - **Part I (10%):** a report encapsulating the assignments of labs 1 to 5.
    - Due *February 23rd* (end of week 6)
  - **Part II (10%):** a report encapsulating the assignments of labs 6 to 10.
    - Due *April 3rd* (end of week 12)

Topic presentation & participation (10%)

- **Groups:** 7 or 8 students
  - To form your group any member will create a Slack group chat including the instructor on slack informing their decision.
  - Group formation **deadline Feb 8th**
    - You SHOULD use the slack channel #assignments to look for students or groups to join.
    - After deadline smaller groups or solo students will be randomly assigned to groups
    - *If you are having issues finding a group, contact the instructor ASAP to get assistance*
- **Topic:** Chosen from a list of topics (*covered in lecture or not*) related to the course
  - Every group will submit via SLACK their top 5 choices of topics.
  - Groups formed first will have preference on the topic of their liking.

Refer to topic presentation document under Content on Blackboard for more details.

Tests (60%) – Midterm (20%) + Final (40%)

- **Style:** written exam offline
  - Multiple choice questions + software design/UML questions
- **Midterm:** Feb 27th (Thursday after reading week)
- **Final:** final exam schedule TBA
  - Accumulative content

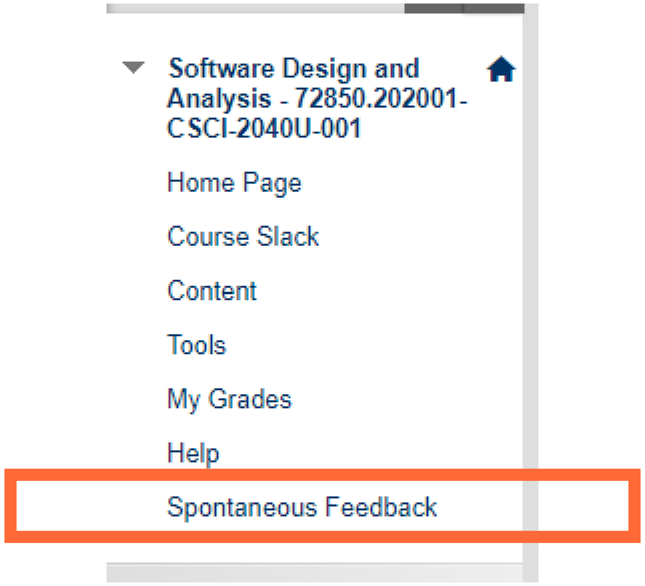
Spontaneous Feedback form

I want to hear from you

Please, with the link below or on Blackboard, you can leave suggestions for the lectures, labs, assignments, complaints about our learning environment or classroom, things you would like see more often, things you did not find helpful, any problem you might encounter with the materials, and **feedback about me either positive or negative**.

It is anonymous!! The form will require you to sign in with your university email for authentication, but it will not store it.

https://docs.google.com/forms/d/e/1FAIpQLScySGTq5F5G0Aj_EU20yHtho0jjQjHfoF7GZGsvPz2qf009gQ/viewform?usp=sf_link



Example

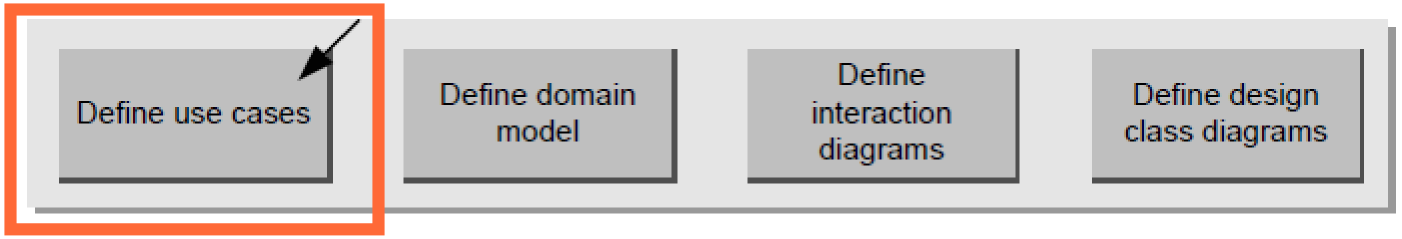
Object-Oriented Design and Analysis

Short Example of OOD/A

- 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.

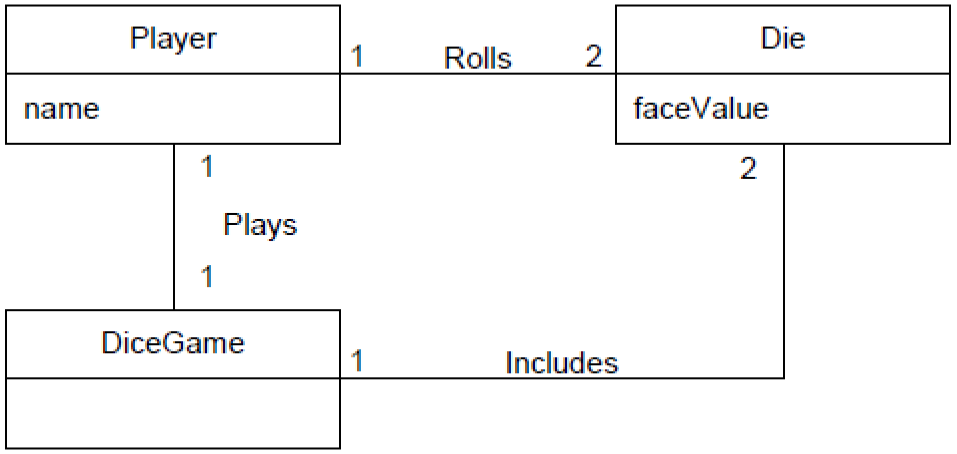
Define Use Cases

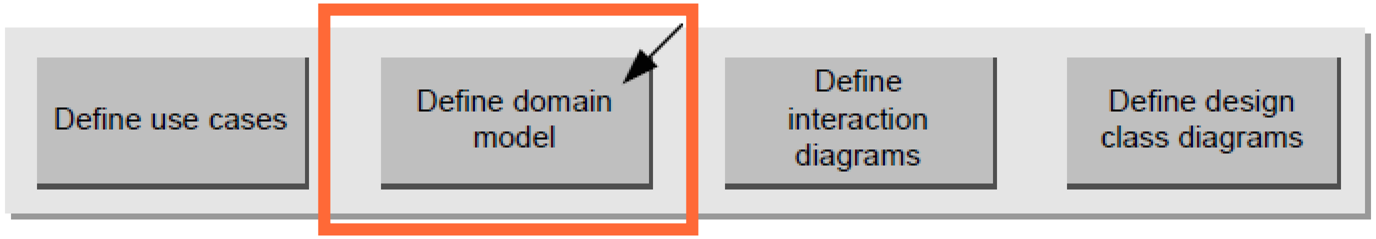
- Use cases are simply written stories.
- For example, here is a brief version of the *Play a Dice Game* use case:
  - **Play a Dice Game:** A player picks up and rolls the dice.
  - If the dice face value total seven, they win; otherwise, they lose.



Define a Domain Model

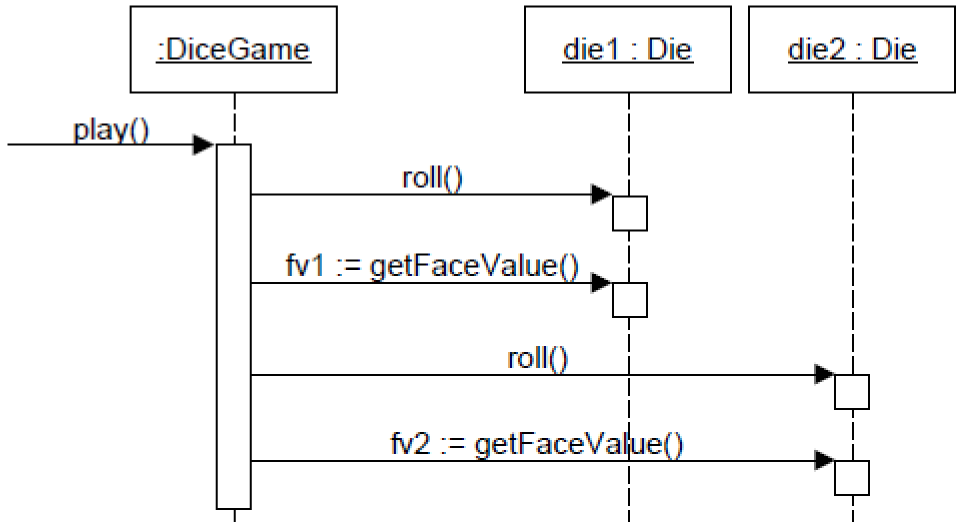
- Defining a domain model involves an identification of the *concepts*, *attributes*, and *associations* that are considered noteworthy.

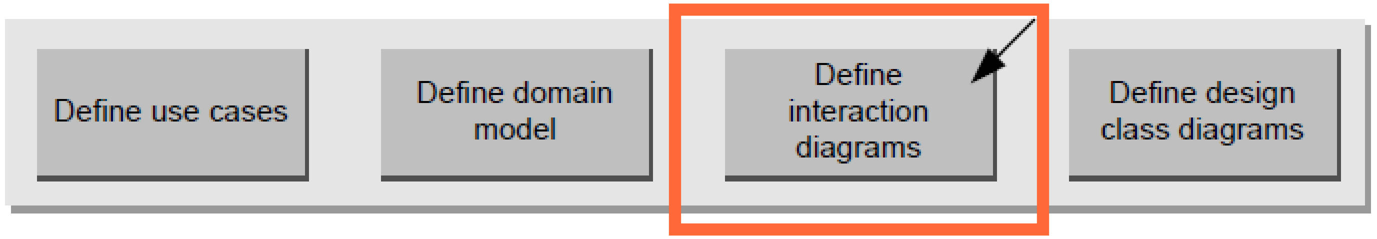




Define Interaction Diagrams

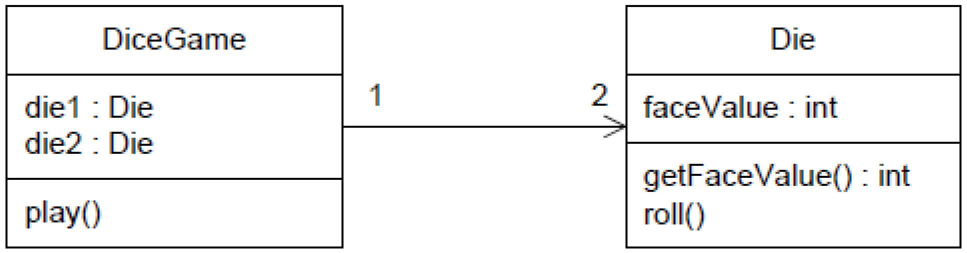
- Interaction Diagrams show the flow of messages between software objects, and thus the *invocation of methods*.

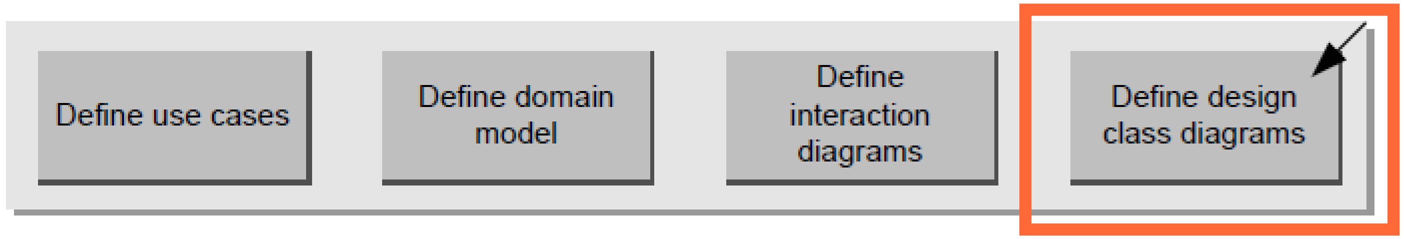




Define Design Class Diagrams

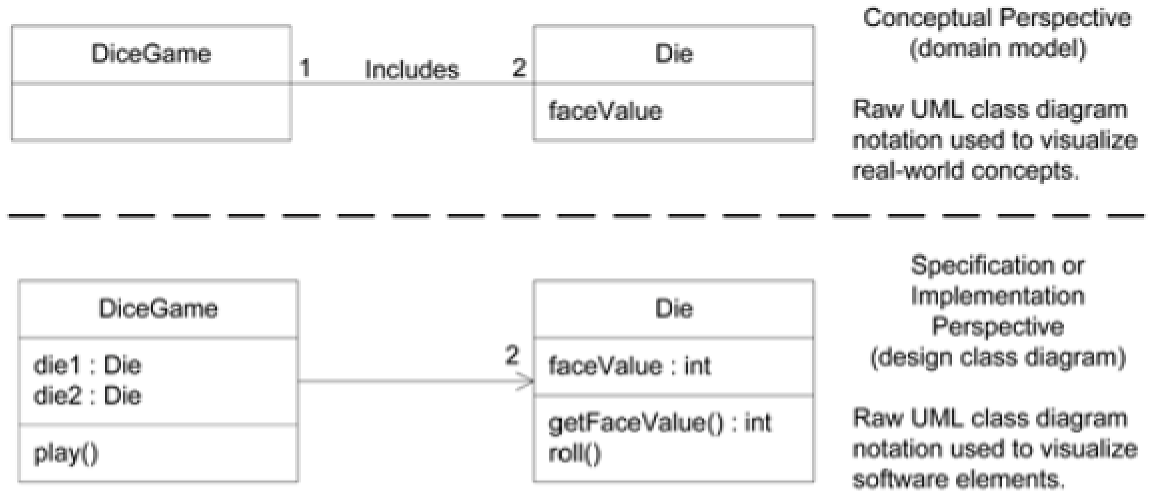
- In addition to a *dynamic* view of collaborating objects shown in interaction diagrams,
  - it is useful to create a *static* view of the class definitions with a design class diagram.





Different Perspectives with UML

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


DiceGame
1
Includes
2
Die
faceValue
Conceptual Perspective (domain model)
Raw UML class diagram notation used to visualize real-world concepts.
DiceGame
die1 : Die
die2 : Die
play()
2
Die
faceValue : int
getFaceValue() : int
roll()
Specification or Implementation Perspective (design class diagram)
Raw UML class diagram notation used to visualize software elements.

Review

What have we learned today, where to next?

Today we…

- Defined the course topic and scope
- Reviewed the course outline
- Learned more about OOD/A with an example

Next lecture we will learn about…

- Introduction to Unified Process and Agile
- Defining use cases
- Inception *versus* requirements

Your actions

- Read Chapter 1
  - *Applying UML and Patterns*, Craig Larman

- Review the **topic presentation** document on Blackboard and start forming your group.

- Install Microsoft Visio
  - MSDN DreamSpark Premium
    - https://uoitsci.onthehub.com/WebStore/Security/SignIn.aspx
  - If you have problems, ask the TA during your 1st lab session.

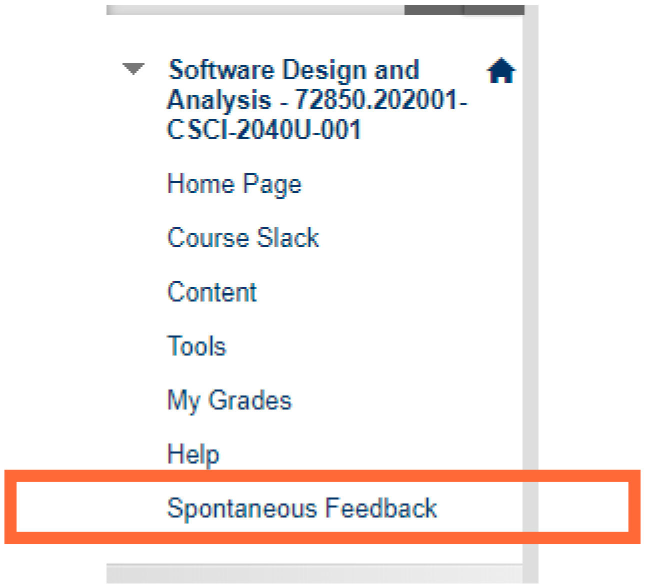
Spontaneous Feedback form

I want to hear from you

Please, with the link below or on Blackboard, you can leave suggestions for the lectures, labs, assignments, complaints about our learning environment or classroom, things you would like see more often, things you did not find helpful, any problem you might encounter with the materials, and **feedback about me either positive or negative**.

It is anonymous!! The form will require you to sign in with your university email for authentication, but it will not store it.

https://docs.google.com/forms/d/e/1FAIpQLScySGTq5F5G0Aj_EU20yHtho0jjQjHfoF7GZGsvPz2qf009gQ/viewform?usp=sf_link


Software Design and Analysis - 72850.202001-CSCI-2040U-001
Home Page
Course Slack
Content
Tools
My Grades
Help
Spontaneous Feedback