Introduction to Object Oriented Analysis, Design and Unified Modeling Language™ (UML™)

Shanika Karunasekera

What we learnt so far -

- Introduction to Software engineering
- Software life-cycle
- Software design principles
- Object Oriented concepts
- Object Oriented software implementation with JAVA
- *What 's next?*

What's next -

- Object Oriented Analysis
  - Determine what to do
  - Extract Objects (modules)
- Object Oriented Design
  - Determine how to build
- Unified Modeling Language™ (UML™)
  - Is a modeling language for documenting OO Analysis and Design.

Problem Statement

MelbX, a famous Australian University, has committed to a new registration system for the students, employees and academic staff. The university would like to keep records for each type of university member in a centralized filing system. They would like to implement the system using the latest software development technologies. The system should be able to support the following high level requirements.

1. Keep records for all university members.
2. Capability to add new records to the system.
3. Capability to edit specific records.
4. Capability to enrol/unenrol students in subjects.
5. Capability to assign/unassign course to staff.
6. Capability to add/modify/list prerequisites for a course.

Solving the Problem

- Step 1 - Write Requirements for the system
  - Should be clearly written and unambiguous
  - Should be implementable
  - Should be testable
  - Is not a part of this course
- Step 2 – Analysis, Design (Modeling)
  - You will be learning in this section…..
- Step 3 – Implementation
  - You already know…..

Analysis, Design and Implementation

- Proper analysis and design (modeling) prior to implementation results in a high quality product.
- Successful projects spend most time on analysis, less on design and even less time on implementation.

Object Oriented Modeling Processes

- Rational Unified Process (RUP) – also called the Unified Process.
  - Most widely used modeling process
  - Introduced by Ivar Jacobson, Gary Booch and Jim Rumbaugh
- Other Object Oriented Modeling Processes
  - Object Oriented Software Process (OOSP)
  - OPEN Process (www.open.org.au)
  - ICONIX Unified Modeling (www.iconixsw.com)

Phases Rational Unified Process

- Inception
  - Identify the system, what it contains and the business case - Analysis
- Elaboration
  - Perform the detailed design for the system - Design
- Construction
  - Write Software
- Transition
  - Deliver to users.

Introduction to UML

- Is a graphical modeling language that can be used represent the artifacts of analysis and design.
- It is a standard that has international support.
- Was developed by Rational Software –
  - Grady Booch, Jim Rumbaugh and Ivar Jacobson

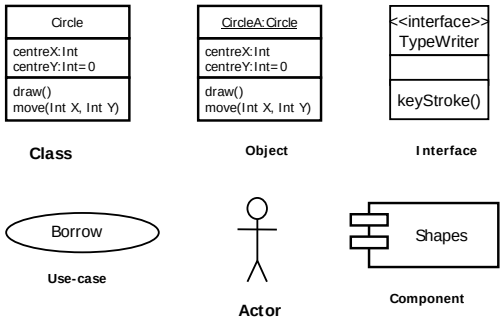
UML™ History

- 1994 – Grady Booch and Jim Rumbaugh started at Rational creating the new notation
  - Grandy Booch – Booch Method
  - Jim Rumbaugh – Object Modeling Technique
- 1995 – Ivar Jacobson Joined the team
  - Object-Oriented Software Engineering
- First Version of UML – 0.8
- 1995 Object Management Group (OMG) agreed to make UML the standard
- 1996 – Additional companies got involved.
- Current version of UML is 2.0

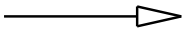
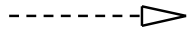
UML™ – Building Blocks

- Elements (Things)
  - *e.g. classes, interfaces etc*
- Relationships
  - *e.g. association, generalization*
- Diagrams
  - *e.g. class diagrams, use case diagrams*

UML™ - Elements



UML™ - Relationships

- Dependency
- Association
- Generalization 
- Realization 

UML™ - Diagrams

- Structural Diagrams
  - Visualize, specify, construct and document the *STATIC* aspects of the system.
- Behavioral Diagrams
  - Visualize, specify, construct and document the *DYNAMIC* aspects of the system.

UML™– Diagrams – cont..

Structural
Class Diagram
Object Diagram
Component Diagram
Deployment Diagram

Behavioral
Use case Diagram
Sequence Diagram
Collaboration Diagram
Statechart Diagram
Activity Diagram

UML™– Diagrams - Notes

- **Class Diagram** shows a set of classes, interfaces and their relationships.
- **Object Diagram** shows a set of objects and their relationships.
- **Component Diagram** shows the organization and dependencies among a set of of components.
- **Deployment Diagram** shows the configuration of run time processing nodes.

UML™– Diagrams - Notes

- **Use Case Diagram** shows use cases, actors and their relationships.
- **Sequence Diagram** shows the interaction between actors and objects and other objects of the system in ordered based on the time they occur.
- **Collaboration Diagram** shows the structural organization of the objects that send and receive messages.
- **Statechart Diagram** shows a state machine consisting of states, transitions an events.
- **Activity Diagram** used to analyze the bahaviour within more complex use-cases.

Life Cycle Phase ⟺ Diagrams

- Requirements
- Object Oriented Analysis
- Object Oriented Design
- Development
- Deployment

Analysis ⟷ Diagrams

Activity	UML Diagram
Understand System Usage	Use-case Diagram
Define Workflows	Activity Diagram
Identify Classes	High Level Class Diagram

Design ⟷ Diagrams

Activity	UML Diagram
Identify Interactions among objects	Sequence and Collaboration Diagram
Analyze State Changes	State Diagram
Refine Class Diagrams	Class Diagram

UML Modeling Tools

- Rational Rose
  - Provides complete UML support
  - As many useful capabilities
  - Provides complete object management
- Visio
- Many other free tool are available on the web –
  - Be careful since some of them don't allow printing.
- Pen and Paper

Reference

- Reference : UML A Beginner's Guide - Jason T. Roff