# IT3102: OBJECT ORIENTED SYSTEMS DEVELOPMENT

*(Compulsory)*

## 1. OUTLINE OF THE SYLLABUS

<table>
<thead>
<tr>
<th>Topic</th>
<th>Minimum number of hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Object Oriented Concepts</td>
<td>02</td>
</tr>
<tr>
<td>• Visual Modelling using Unified Modelling Language (UML)</td>
<td>03*</td>
</tr>
<tr>
<td>• Introduction to Unified Software Development Process</td>
<td>03</td>
</tr>
<tr>
<td>• Creating Use Case Diagrams</td>
<td>04*</td>
</tr>
<tr>
<td>• Identifying Classes, Packages and drawing a Class diagram</td>
<td>04*</td>
</tr>
<tr>
<td>• Discovering Object Interaction</td>
<td>04*</td>
</tr>
<tr>
<td>• Specifying Relationships</td>
<td>03*</td>
</tr>
<tr>
<td>• Adding Behaviour and Structure</td>
<td>03*</td>
</tr>
<tr>
<td>• Analysing Object Behaviour</td>
<td>03*</td>
</tr>
<tr>
<td>• Checking the Model</td>
<td>03*</td>
</tr>
<tr>
<td>• Designing the System Architecture</td>
<td>03*</td>
</tr>
<tr>
<td>• Building the Iterations</td>
<td>03*</td>
</tr>
</tbody>
</table>
Learning Objectives

After successfully completing the module student should be able to:
- To have a firm foundation on object oriented systems development environment, concepts, visual modeling techniques and the UML (Unified Modeling Language)
- To gain knowledge sufficient to undertake a moderately complex systems development project using an Object Oriented CASE tool.

2. DETAILED SYLLABUS

1. Object Oriented Concepts (2 hrs)

   **Instructional Objectives**
   - Be able to distinguish the differences between traditional methodologies and Object Oriented methodologies.
   - Be familiar with Object Oriented concepts.

   1.1. Introduction [Ref : Slides]
   1.2. Classes, Objects [Ref 2: pg. 32-33]
   1.3. Inheritance [Ref 2: pg. 35-36]
   1.4. Encapsulation [Ref 2: pg. 37-38]
1.5. Association [Ref 2: pg. 40-41]
1.6. Aggregation [Ref 2: pg. 42-43]
1.7. Polymorphism [Ref 2: pg. 36-37]

2. Visual Modelling using Unified Modelling Language (UML) (3 hrs)

**Instructional Objectives**
- Be able to understand the benefits of Visual Modeling.
- Be familiar with Rational Rose interface.
- Be familiar with the history of UML.

2.1. What is Visual Modeling? [Ref 1: pg. 3-4]
   2.1.1. What is a Model?
   2.1.2. Importance of Modeling

2.2. Introduction to Unified Modelling Language (UML) [Ref 1: pg. 5-6]

2.3. Introduction to Software Development Process [Ref 1: pg. 8-10]

2.4. Introduction to Modelling tools [Ref 1: pg. 11]

3. Introduction to Unified Software Development Process (3 hrs) [Ref: Slides]

**Instructional Objectives**
- Be able to understand the benefits of Rational Unified Process.
- Be familiar with the stages of the Rational Unified Process (RUP).

3.1. Introduction
3.2. Benefits
3.3. Phases and Iterations
   3.3.1. Inception Stage
   3.3.2. Elaboration Stage
   3.3.3. Construction Stage
   3.3.4. Transition Stage

4. Creating Use Case Diagrams (4 hrs)

**Instructional Objectives**
- Be competent with the techniques used to examine system behaviour from a Use Case approach.
- Be familiar with creating Use Case diagrams in Rational Rose

4.1. Actors and Use Cases [Ref 1: pg. 21-28]
   4.1.1. Actors
   4.1.2. Use Cases

4.2. Documenting Use Cases [Ref 1: pg. 29-34]
   4.2.1. Flow of Events Document
4.3. Use Case Relationships [Ref 1: pg. 35-36]
   4.3.1. Stereotypes, Communicate, Extend, Include

4.4. Use Case Diagrams [Ref 1: pg. 37-39]
   4.4.1. Creating Main Use Case Diagram in Rational Rose
   4.4.2. Creating Relationships in Rational Rose
   4.4.3. Creating Additional Use Case Diagrams in Rational Rose

4.5. Activity Diagrams [Ref 1: pg. 40-49]
   4.5.1. Activities, Transitions, Decision Points, Synchronization Bars, Swimlanes

5. Identifying Classes, Packages and Drawing a Class Diagram (4 hrs)

   **Instructional Objectives**
   - Be able to understand the concepts and notations used for finding Objects and Classes.
   - Be familiar with UML concepts of Stereotypes and Packages
   - Be familiar with creating Package diagrams and Class diagrams in Rational Rose

5.1. State, Behaviour, and Identity of Objects [Ref 1: pg. 53-55]
5.2. Stereotypes and Classes [Ref 1: pg. 56-60]
   5.2.1. Identifying Classes, Entity classes, Boundary classes, Control classes
5.3. Packages [Ref 1: pg. 61-65]
5.4. Class Diagram [Ref 1: pg. 65-68]

6. Discovering Object Interactions (4 hrs)

   **Instructional Objectives**
   - Be able to add scenarios to the system to describe how Use Cases are realized as interactions among societies of objects.
   - Also be competent with Sequence diagrams and Collaboration diagrams that are used to capture scenarios.
   - Be familiar with creating Sequence diagrams and Collaboration diagrams in Rational Rose

6.1. Use Case Realizations [Ref 1: pg. 73-76]
6.2. Sequence Diagrams [Ref 1: pg. 77-81]
6.3. Collaboration Diagrams [Ref 1: pg. 82-85]

7. Specifying Relationships (3 hrs)

   **Instructional Objectives**
   - Be able to illustrate the definition of relationships between classes in the system. Specifically, the concepts of Association and Aggregation.
   - Be familiar with reflexive and package relationships.
   - Be able to illustrate the application of generalization and specialization principles to discover super class/subclass relationships.
7.1. The Need of Defining Relationships [Ref 1: pg. 89]
7.2. Association and Aggregation Relationships [Ref 1: pg. 89-92]
7.3. Naming Relationships [Ref 1: pg. 92-93]
7.4. Role Names [Ref 1: pg. 93-94]
7.5.Multiplicity Indicators [Ref 1: pg. 95-96]
7.6. Reflexive Relationships [Ref 1: pg. 96-97]
7.7. Finding Relationships [Ref 1: pg. 98]
7.8. Package Relationships [Ref 1: pg. 98-100]

8. Adding Behaviour and Structure (3 hrs)

**Instructional Objectives**
- Be able to understand how the needed structure and behaviour of classes are added to the model under development.

8.1. Representing Behaviour and Structure [Ref 1: pg. 105]
8.2. Creating Attributes and documenting them [Ref 1: pg. 110-112]
8.3. Creating Operations and Documenting them [Ref 1: pg. 106-110]
8.4. Display Attributes and Operations [Ref 1: pg. 112-114]
8.5. Association Classes [Ref 1: pg. 115]

9. Analysing Object Behaviour (3 hrs)

**Instructional Objectives**
- Be familiar with state transition diagrams that provide additional analysis techniques for classes with significant dynamic behaviour.
- Be familiar with creating State Transition diagrams in Rational Rose

9.2. States [Ref 1: pg. 132-133]
9.3. State Transitions [Ref 1: pg. 133-135]
9.4. Special States (Start and Stop) [Ref 1: pg. 135-137]
9.5. State Transition Details [Ref 1: pg. 137-138]
9.6. State Details [Ref 1: pg. 138-140]

10. Checking the Model (3 hrs)

**Instructional Objectives**
- Be familiar with the techniques used to blend and check models for consistency

10.1. Making the Model Homogeneous [Ref 1: pg. 145]
10.2. Combining Classes [Ref 1: pg. 145-146]
10.3. Splitting Classes [Ref 1: pg. 146]
10.4. Eliminating Classes [Ref 1: pg. 147]
10.5. Consistency Checking [Ref 1: pg. 147]

11. Designing the System Architecture (3 hrs)

**Instructional Objectives**
- Know the concepts and notation needed to specify, visualize, and document the system architecture.
- Be familiar with multiple views of architecture.
- Be familiar with the UML notations used to represent architectural decisions made for a system.

11.1. The need for Architecture [Ref 1: pg. 153-154]
11.2. The “4+1” view of Architecture [Ref 1: pg. 154]
11.3. The Logical view [Ref 1: pg. 154-157]
11.4. The Implementation View [Ref 1: pg. 157-162]
11.5. The Process View [Ref 1: pg. 163-164]
11.6. The Deployment View [Ref 1: pg. 164-166]
11.7. The Use Case View [Ref 1: pg. 166]

12. Building the Iterations (3 hrs)

**Instructional Objectives**
- Be familiar with the iteration planning process.
- Be familiar with the notations used to specify and document the design decisions that occur during the implementation of an iteration.

12.1. The Iteration Planning Process [Ref 1: pg. 171-173]
12.2. Design the User Interface [Ref 1: pg. 173-174]
12.3. Adding Design Classes [Ref 1: pg. 174-175]
12.4. The Emergence of Patterns [Ref 1: pg. 175-176]
12.5. Designing Relationships [Ref 1: pg. 176-181]
12.6. Designing Attributes and Operations [Ref 1: pg. 182-183]
12.7. Designing for Inheritance [Ref 1: pg. 184]
12.8. Coding, Testing, and Documenting the Iteration [Ref 1: pg. 184-185]

13. Object Oriented Programming (09 hrs) [Ref: Slides]

**Instructional Objectives**
- Be familiar with object-oriented concepts and C++ programming language.

13.1. Introduction to C++
13.2. Input/Output, Variables, Constants, Data types, Operators
13.3. Identifiers, Declarations and Functions
13.4. Control Structures
13.5. Arrays and Structures
13.6. Pointers
13.7. Implementing Object Oriented Programming Concepts using C++
   13.7.1. Classes
   13.7.2. Constructors, Destructors, Copy Constructors
   13.7.3. new and delete operators
13.8. Inheritance, Multiple Inheritance using C++
13.9. Polymorphism using C++
13.10. Aggregation using C++

14. Code Generation and Reverse Engineering with C++ and Rational Rose (2 hrs)
   **Instructional Objectives**
   - Be familiar with the code Generation and Reverse Engineering with C++ and Rational Rose

14.2. Reverse Engineering using the C++ Analyzer [Ref 1: pg. 198-207]

15. Introduction to UML 2.0 (1 hr)
   **Instructional Objectives**
   - Be familiar with the new features introduced in UML 2.0

15.1. New Concepts and Diagrams in UML 2.0
   15.1.1. Composite Structure Diagram [Ref 1: pg. 22]
   15.1.2. Interaction Overview Diagram [Ref 1: pg. 23]
   15.1.3. Timing Diagram [Ref 1: pg. 24]
   15.1.4. State Diagram [Ref 1: pg. 130-131]
   15.1.5. Sequence Diagram [Ref 1: pg. 149-153]
   15.1.6. Activity Diagram [Ref 1: pg. 181-191]
   15.1.7. Component Diagram [Ref 1: pg. 200-209]

16. A Case Study Using an Object Oriented CASE Tool (10 hrs)
   **Instructional Objectives**
   - Be able to undertake a moderately complex systems development project using Rational Rose and C++ language.

3. BOOKS RECOMMENDED FOR READING AND REFERENCE
Main Reading


Ref 2: Sams Teach Yourself “UML” In 24 Hours By Joseph Schmuller, 3rd edition, 2004

Supplementary Reading


4. HARDWARE AND SOFTWARE REQUIREMENTS

Hardware

Any standard PC (Pentium)

Software

- Windows 95/98/2000/NT/XP
- Rational Rose 2000/2002
- C++ Compiler

5. EXAMINATION PROCEDURE

Examination Paper will consist of two parts.

- Part 1: 1 1/2 Hour paper consisting of Multiple Choice Questions
- Part 2: 1 1/2 Hour Paper consisting of Structured Questions

Note: Students will be given two assignments/case studies during the semester. There will be multiple choice and/or structured questions based on these assignments/case studies.