COMP11113

Information Systems Analysis and Design

Unit 2

An Introduction to UML (Unified Modelling Language)

Aims

This unit provides an introduction to the Unified Modelling Language for object-oriented analysis and design. It explains the advantages of using object-orientation in system development and how UML can support object-oriented analysis and design. It outlines the diagrams included in UML. It discusses how an object-oriented system can be produced using UML in a use case-driven development process. This unit also explains the analysis model and the views that the model will address in modelling an object-oriented system.

Objectives

When you have completed this unit, you should be able to:

- Understand the history of object-oriented technology.
- Explain the advantages of object-orientation.
- Explain fundamental object-oriented principles and concepts.
- Understand what UML can be used for object-oriented system development.
- Describe the use case-driven process for object-oriented system development.
- Understand the analysis model and the views used in building the model.

Overview

This unit introduces you to an industry standard, the Unified Modelling Language (UML) that is a programmatic notation. The system analyst and designer can use this unified modelling language to describe user requirements, and specify and design an object-oriented system during system development.

In this module the topics are broken down into the two sections:

Section 1 provides the history of object-oriented technology and discusses the necessity and importance of UML in object-oriented analysis and design. It explains fundamental object-oriented principles and concepts.

Section 2 outlines all diagrams included in UML. It explains a use case-driven process for system development. It then describes the analysis model in detail.

Reading of the textbooks:

John W. Satzinger, Robert B. Jackson and Stephen D. Burd, *Object-Oriented Analysis and Design with the Unified Process,* Thomson, 2005.

Alan Dennis, Barbara Wixom, and David Tegarden, Systems Analysis and Design: An Object-Oriented Approach with UML (6th edition), Wiley, 2021.

Table of Contents

1 INTRODUCTION TOOBJECT-ORIENTED TECHNOLOGY

1.1 Objectives

After completing this section you should be able to:

- explain history of object-oriented technology.
- understand advantages of object-orientation.
- Indicate the benefits of object-orientation.

1.2 History of Object-Oriented Technology

The following shows the history of object-oriented technology:

1967	Simula-67 (the first object-oriented programming language)
1980	Smalltalk-80 (another new object-oriented programming language)
1984	Object Pascal, Objective-C (object-oriented programming languages based on structural programming languages Pascal and C)
1985	C++ (a new object-oriented programming language based on C)
1986	Object-oriented design techniques and methods
1991	Object-oriented analysis techniques and methods
1997	UML, industry standard technique for object-oriented software development

1.3 Why Object-Orientation

- Things in the real world become the components (i.e., objects) in software system.
- Data and functions are encapsulated into one component.
- A seamless process of developing software systems becomes true.

1.4 Benefits of Object-Orientation

It makes software

- more maintainable
- more understandable and testable
- more reusable

Object-oriented technology can help coping with large and complex systems.

1.5 Self Assessment Questions

1. What does object-orientation address?

2. What are benefits of object-orientation?

1.6 Learning Outcomes

When you have completed this unit, you should be able to:

- Understand the history of object-oriented technology.
- Explain the advantages of object-orientation.
- Understand what UML can be used for system development.
- Describe the use case-driven process for object-oriented system development.

2 OVERVIEW OF UML

2.1 Objectives

After completing this section you should be able to:

- explain what is UML.
- Understand what UML consists of.

2.2 What is UML?

The Unified Modelling Language (UML) is a graphical modelling language, i.e. the notation used to express systems. UML mostly unifies and extends the methods of Booch, Rumbaugh (OMT), and Jacobson. It is used as OMG (the Object Management Group) standard currently for object-oriented modelling and design.

The UML is called a modelling language rather than a method, since it provides only the notation that is used to represent analysis and design. A method must provide both of a modelling language and a development process. A modelling language is a language whose vocabulary and rules focus on the conceptual and physical representation of a system.

- **The UML is a Language for Visualizing**
  Writing explicit models in the UML can facilitate communication between software developers and between the user and developer.
- **The UML is a Language for Specifying**
  The UML builds models that are precise, unambiguous, and complete. In particular, the UML addresses the specification of all the important analysis, design, and implementation decisions that must be made in developing and deploying a software-intensive system.
- **The UML is a Language for Constructing**
  You may be able to map from a model in the UML to a programming language such as Java, C++, or Visual Basic, or even to tables in a relational database or the persistent store of an object-oriented database.
- **The UML is a Language for Documenting**
  The UML addresses the documentation of a system's architecture and all of its details.

2.3 Diagrams in UML

A *diagram* is the graphical presentation of a set of elements of models. The UML mainly consists of five different groups of the UML diagrams that offer five different ways of looking at the same problem in analysis and designing a solution to the problem in design. By creating the diagrams in each group, you can analyze a problem from different perspectives. The UML diagrams are grouped as follows:

- ***Use Case Diagrams (view the functional parts of a system)***
- ***Static Structure Diagrams (view the static parts of a system)***
  - Class diagram
  - Object diagram
- ***Interaction Diagrams (view the dynamic parts of a system)***
  - Sequence diagrams
  - Collaboration diagrams

- ***State Diagrams (view the dynamic parts of a system)***
  - State diagrams
  - Activity diagrams
- ***Implementation Diagrams (view the static parts of a system)***
  - Package diagrams
  - Component diagrams
  - Deployment diagrams

2.4 A Use Case-Driven Process

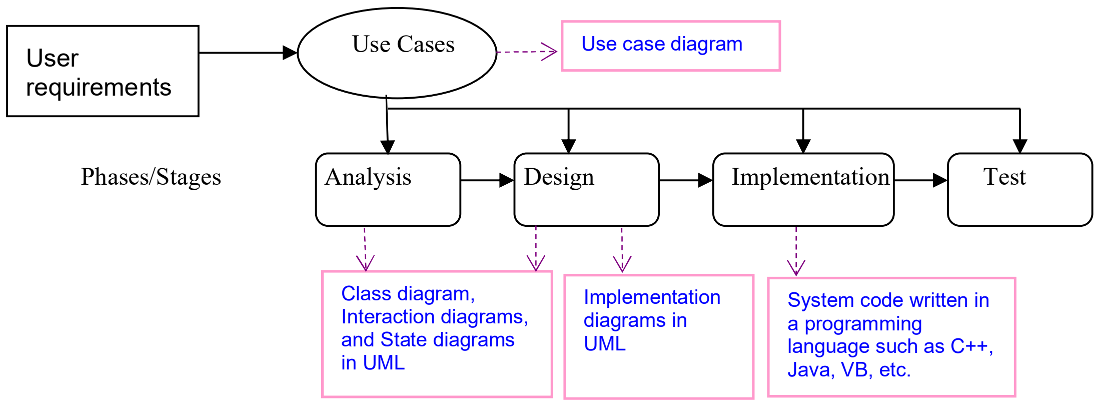
UML use cases capture the functional requirements of the system.

Thus use cases

- are used to ensure that all functionality is realized in the system, and to verify and test the system.
- affect all phases and all views because the use cases contain the descriptions of the functions.
- during analysis, are used to capture the required functionality and to validate this functionality with the users.
- during design and implementation, the use cases must be realized in the construction.
- during testing, the use cases verify the system; they become the basis for the test cases.

UML use cases capture the functional requirements of the system. Thus use cases are used to ensure that all functionality is realized in the system, and to verify and test the system. Because the use cases contain the descriptions of the functions, they affect all phases and all views, as shown in Figure 1. During analysis, they are used to capture the required functionality and to validate this functionality with the users. During design and implementation, the use cases must be realized in the construction. And finally, during testing, the use cases verify the system; they become the basis for the test cases.

Figure 1. Use-case driven process


User requirements
Use Cases
Use case diagram
Phases/Stages
Analysis
Design
Implementation
Test
Class diagram, Interaction diagrams, and State diagrams in UML
Implementation diagrams in UML
System code written in a programming language such as C++, Java, VB, etc.

2.5 Analysis Models

Analysis models are built for determining and specifying the user requirements. They must be

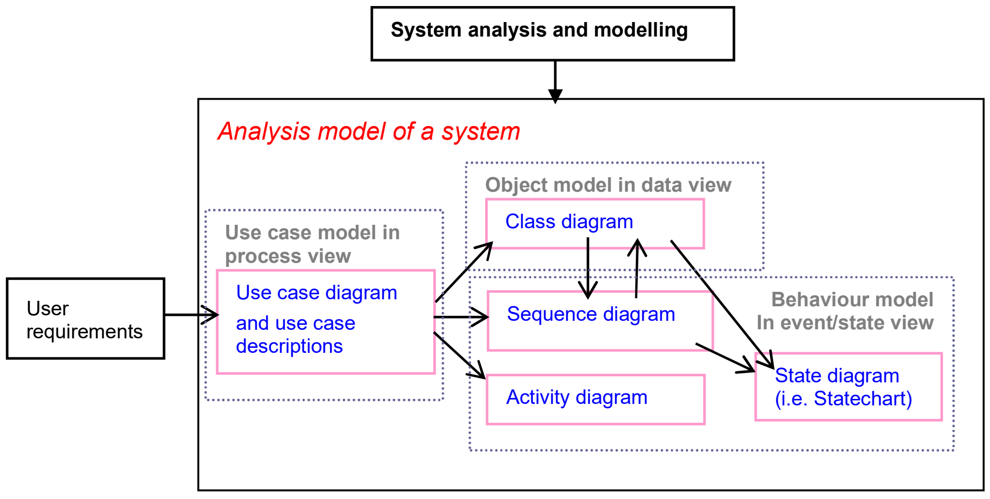
- contain an overall description of functions.
- represent any people, physical things and concepts that are important in a problem situation or the problem domain.
- show structures and interactions among the people, things and concepts.
- show the problem situation in enough detail to evaluate possible designs.

Analysis models are produced in different perspectives of a system and described by one kind of diagrams, each of which represents one view of the system.

- The ***use case model*** is built by requirement capturing in the functional view. It is represented by use case diagrams in a process view.
- The ***object model*** is built by requirements analysis in the data view. It is represented by class diagrams in a data view.
- The ***behavioural model*** is built by requirements analysis in the event/state view. It is represented by sequence diagrams, collaboration diagrams, activity diagrams, and state diagrams in an event/state view.

Figure 2 shows the relationships among the diagrams in the analysis model that you need to learn and understand in this module.

Figure 2. Analysis model and UML diagrams


System analysis and modelling
Analysis model of a system
Object model in data view
Class diagram
Use case model in process view
Use case diagram and use case descriptions
User requirements
Sequence diagram
Behaviour model In event/state view
State diagram (i.e. Statechart)
Activity diagram

2.6 The Case Study: A Solent Hotel System

This case study will be used thought out the following units as an example of using UML in object-oriented analysis and design. Figure 3 shows the case study information for the system:

Figure 3 The case study: a Solent Hotel system

A computer system is required by the Solent Hotel to assist the management of the hotel using computer.

The Solent Hotel has 5 function rooms, 20 single bedrooms and 40 double bedrooms. The room are distinguished by their room numbers. The function rooms have different sizes, but have same tariff. The hotel accepts the booking requests for bedroom or function room only at present.

When customers arrive at the hotel, the receptionist books them into the first available room of the required type. If none of rooms of the required type is available, the receptionist has to reject the room booking request. The customer's name is recorded, along with the payee (i.e. who is paying for the room.) This is recorded as 'private' if the customer is the payee, or the name of a company or organization may be entered. The tariffs are £40 for a single bedroom, £55 for a double bedroom and £200 for a function room. There is a set of presentation equipment in the hotel which may be moved between function rooms if necessary. The equipment are known with their names and descriptions. The hotel also ensures that a room is made available for further bookings as soon as it is vacated.

In order to improve the business, the hotel requires an information system that supports the room booking using the computer and to store information for later access.

2.7 Self Assessment Questions

1. What is UML?

2. Why isn't UML a method?

3. What is UML used for in system development?

4. What diagrams does UML include?

5. What is a use case-driven process?

6. What is an analysis model?

2.8 Learning Outcomes

Having completed this section, you should be able to:

- Describe the process of system development.
- Discuss the relationship among software quality, development process, development methods and techniques, and CASE tools in system development.
- Define the software lifecycle models.

- Explain how CASE tools support system development.
- Explain the factors that affect system quality.