Module Two

BASIC TECHNIQUES OF MODELING COMPUTER SYSTEMS

1. Introduction
2. What is a model?
   - 2.1. What is the Importance of Modelling? (or why do we model?)
3. Structured approach
   - 3.1. Structured Analysis
     - 3.1.1. Structured Analysis Tools
   - 3.2. Structured Design
4. Object Oriented Approach
     - 4.1.1. Phases in Object-Oriented Software Development
   - 4.2. Types of Object-Oriented Methodologies
   - 4.3. Objectives of Object-Oriented Methodologies
   - 4.4. Benefits of Object-Oriented Methodologies
   - 4.5. Structured Approach Vs. Object-Oriented Approach

Module Two
BASIC TECHNIQUES OF MODELING COMPUTER SYSTEMS

1. Introduction

At the beginning of the creation of a new product the problem is often ill-defined and only some ideas exist about potential solutions. The architecting effort must change this situation in the course of the project into a well-articulated and structured understanding of both the problem and its potential solutions. Modelling and analysis support the architecting in several ways during the project life cycle. Early modelling and analysis efforts help to understand the problem and solution space. When the project gets more tangible the purpose of modelling shifts to exploration of specification and design alternatives. For some problems it is rewarding to optimize the solution by means of models. When the realization gets up and running, then model and realization can be compared for verification purposes

2. What is a model?

A model is an abstraction or a simplified representation of something, or a real system for the purpose of understanding it better before building it. In other words, a model is a simplification of reality which takes the theoretical abstractions and puts it into a form that we can manipulate. Another definition of a model is an abstraction of a system, aimed at understanding, communicating, explaining, or designing aspects of interest of that system. Because, real systems that we want to study are generally very complex. In order to understand the real system, we have to simplify the system. So a model is an abstraction that hides the non-essential characteristics of a system and highlights those characteristics, which are pertinent to understand it.

2.1.What is the Importance of Modelling? (or why do we model?)

The purpose of modelling is to support the architecting effort, or to support the project to get the right specification, design and to take the right decisions to achieve the project goals. Right specification and design is assessed in terms of customer satisfaction, risk level, meeting project

constraints (cost, effort, time), and business viability (profit margin). In order to get to this point, we need information, modelling results, with sufficient accuracy, working range, and credibility. Modelling achieves four aims:

- Helps you to visualize a system as you want it to be.
- Permits you to specify the structure or behaviour of a system.
- Gives you a template that guides you in constructing a system.
- Documents the decisions you have made.

You build models of complex systems because you cannot comprehend such a system in its entirety. You build models to better understand the system you are developing. Several reasons can justify the importance of using a model:

- For reduction of complexity in order to understand it.
- To detect errors and omissions early in the lifecycle
- To test a physical entity before actually building it.
- To communicate with stakeholders
- To understand requirements
- To drive implementation
- To understand the impact of change
- To ensure that resources are deployed efficient
- To ease eventual updates and upgrades of the system
- To set the stage for communication between customers and developers.
- For visualization i.e. for finding alternative representations.

Several approaches are used for the realisation of computer systems. These approaches have different models and processes that characterise them:

3. Structured approach

The structured approach has two major project phases: The Structured Analysis and the Structured Design. It was the most widespread and popular approach to information system development. The method is mainly based on the use of diagrammatic tools. The structured analysis and design method was developed during the 1970s, before workstations and PCs were commonplace. The diagrammatic tools were designed for paper and pencil environments. It is a simple method, easy to understand, and it captures most of the relevant features of conventional, transaction-oriented data processing. The analysis phase is often seen as developing so-called functional requirements specifications. The design phase is seen as developing software and database solutions that satisfy the functional requirements, as well as non-functional requirements concerning, for example, performance and security. Structured analysis and design is a process-oriented method, because the starting point of the information system development is in the analysis of the information processes of the system. These processes are described by the use of so-called Data Flow Diagrams (DFDs). From this starting point the other major systems components of data, rules, and program objects are discovered

and designed. There are different description tools to be used for each of these major systems components.

3.1. Structured Analysis

Structured Analysis is a development method that allows the analyst to understand the system and its activities in a logical way. A functional specification is intended to describe everything the user sees of the software and the interaction between the software and its environment. The functional description must be given in a terminology that is familiar to the user. That is, the system's properties must be described by using the terminology of the particular computer application area being investigated. If the application area is banking, the functional specification must be phrased in bankers' terminology; if it is widget making then that particular widget-community's terminology must be used. The functional description should also refrain from describing the implementation details that the user does not see. Structured Analysis is one particular approach to abstract away implementation details and concentrate on describing those features of an information system that are particularly important to the user. The functional specification that results from applying this approach must also form an effective basis for the subsequent design and implementation of the software. So the specification documents must make sense both to the users and to the software designers. It is a systematic approach, which uses graphical tools that analyse and refine the objectives of an existing system and develop a new system specification which can be easily understandable by user. Experience shows that this is a very difficult task that in the current state of the art is not performed satisfactorily.

3.1.1. Structured Analysis Tools

During Structured Analysis, various tools and techniques are used for system development. They
are:

- Data Flow Diagrams
- Data Dictionary
- Decision Trees
- Decision Tables
- Structured English
- Pseudocode

These tools provide the systems analyst with the means to carry out a disciplined analysis, and provide the end-product of analysis - the functional specification. The objective of using these tools is the preparation of a functional specification that is well understood by the users, sets out the functional requirements of the system without dictating its implementation, expresses users' preferences and system trade-offs.

3.2.Structured Design

The next design step is to create software structures which satisfy the functional requirements

as they are expressed in the dataflow diagrams. Structured design is a data-flow based methodology that helps in identifying the input and output of the developing system. The main objective of structured design is to minimize the complexity and increase the modularity of a program. The central issue is: How does a systems designer transform an information system model in a dataflow diagram into structures of programs and subprograms. Structured design also helps in describing the functional aspects of the system. In structured designing, the system specifications act as a basis for graphically representing the flow of data and sequence of processes involved in a software development with the help of DFDs. After developing the DFDs for the software system, the next step is to develop the structure chart. Subsequent steps may include design changes to satisfy performance requirements and other non-functional requirements, for example, security and reliability requirements.

4. Object Oriented Approach

In the object-oriented approach, the focus is on capturing the structure and behaviour of information systems into small modules that combines both data and process. The main aim of Object-Oriented Design (OOD) is to improve the quality and productivity of system analysis and design by making it more usable. In analysis phase, OO models are used to fill the gap between problem and solution. It performs well in situation where systems are undergoing continuous design, adaption, and maintenance. It identifies the objects in problem domain, classifying them in terms of data and behaviour.

The OO model is beneficial in the following ways:

- It facilitates changes in the system at low cost.
- It promotes the reuse of components.
- It simplifies the problem of integrating components to configure large system.
- It simplifies the design of distributed systems.

4.1.1. Phases in Object-Oriented Software Development

The major phases of software development using object–oriented methodology is: object-oriented analysis, object-oriented design, and object-oriented implementation.

a) Object–Oriented Analysis

In this stage, the problem is formulated, user requirements are identified, and then a model is built based upon real–world objects. The analysis produces models on how the desired system

should function and how it must be developed. The models do not include any implementation details so that it can be understood and examined by any non–technical application expert.

b) Object–Oriented Design

Object-oriented design includes two main stages, namely, system design and object design.

System Design

In this stage, the complete architecture of the desired system is designed. The system is conceived as a set of interacting subsystems that in turn is composed of a hierarchy of interacting objects, grouped into classes. System design is done according to both the system analysis model and the proposed system architecture. Here, the emphasis is on the objects comprising the system rather than the processes in the system.

Object Design

In this phase, a design model is developed based on both the models developed in the system analysis phase and the architecture designed in the system design phase. All the classes required are identified. The designer decides whether:

- new classes are to be created from scratch,
- any existing classes can be used in their original form, or
- new classes should be inherited from the existing classes.

The associations between the identified classes are established and the hierarchies of classes are identified. Besides, the developer designs the internal details of the classes and their associations, i.e., the data structure for each attribute and the algorithms for the operations.

c) Object–Oriented Implementation and Testing

In this stage, the design model developed in the object design is translated into code in an appropriate programming language or software tool. The databases are created and the specific hardware requirements are ascertained. Once the code is in shape, it is tested using specialized techniques to identify and remove the errors in the code.

4.2.Types of Object-Oriented Methodologies

There are three types of Object-Oriented Methodologies

1. Object Modelling Techniques (OMT)
2. Object Process Methodology (OPM)
3. Rational Unified Process (RUP)

4.3.Objectives of Object-Oriented Methodologies

a) Object Modelling Techniques (OMT)

It was one of the first object-oriented methodologies and was introduced by Rumbaugh in 1991. OMT uses three different models that are combined in a way that is analogous to the older structured methodologies.

i. Analysis

The main goal of the analysis is to build models of the world. The requirements of the users, developers and managers provide the information needed to develop the initial problem statement.

ii. OMT Models

1) Object Model

- Represents the static, structural, 'data' aspects of a system.
- It depicts the object classes and their relationships as a class diagram, which represents the static structure of the system.
- It observes all the objects as static and does not pay any attention to their dynamic nature.

2) Dynamic Model

- Represents the temporal, behavioural, 'control' aspects of a system.
- It captures the behaviour of the system over time and the flow control and events in the Event-Trace Diagrams and State Transition Diagrams.
- It portrays the changes occurring in the states of various objects with the events that might occur in the system.

3) Functional Model

- Represents the transformational, 'functional' aspects of a System.
- It describes the data transformations of the system.
- It describes the flow of data and the changes that occur to the data throughout the system.

iii. Design

- It specifies all of the details needed to describe how the system will be implemented.
- In this phase, the details of the system analysis and system design are implemented.
- The objects identified in the system design phase are designed.

b) Object Process Methodology (OPM)

It is also called as second-generation methodology. It was first introduced in 1995. It has only one diagram that is the Object Process Diagram (OPD) which is used for modelling the structure, function and behaviour of the system. It has a strong emphasis on modelling but has a weaker emphasis on process. It consists of three main processes:

i. Initiating: It determines high level requirements, the scope of the system and the resources that will be required.
ii. Developing: It involves the detailed analysis, design and implementation of the system.
iii. Deploying: It introduces the system to the user and subsequent maintenance of the system.

c) Rational Unified Process (RUP)

It was developed in Rational Corporation in 1998. It consists of four phases which can be broken down into iterations.

1) Inception
2) Elaboration
3) Construction
4) Transition

Each iteration consists of nine work areas called disciplines. A discipline depends on the phase in which the iteration is taking place. For each discipline, RUP defines a set of artefacts (work products), activities (work undertaken on the artefacts) and roles (the responsibilities of the members of the development team).

4.4. Benefits of Object-Oriented Methodologies

1) It represents the problem domain, because it is easier to produce and understand designs.
2) It allows changes more easily.
3) It provides nice structures for thinking, abstracting and leads to modular design.
4) Simplicity:
    - The software object's model complexity is reduced and the program structure is very clear.
5) Reusability:
    - It is a desired goal of all development process.
    - It contains both data and functions which act on data.
    - It makes easy to reuse the code in a new system.
    - Messages provide a predefined interface to an object's data and functionality.
6) Increased Quality:
    - This feature increases in quality is largely a by-product of this program reuse.
7) Maintainable:
    - The OOP method makes code more maintainable.
    - The objects can be maintained separately, making locating and fixing problems easier.

4.5. Structured Approach Vs. Object-Oriented Approach

Structured Approach	**Object oriented approach**
It works with Top-down approach.	It works with Bottom-up approach
Program is divided into number of submodules or functions	Program is organized by having number of classes and objects.
Function call is used.	Message passing is used.
Software reuse is not possible.	Reusability is possible