Practical 4

Unified Modelling Language (**UML**) is a standardized (**ISO/IEC 19501:2005**), general-purpose modelling language in the field of software engineering. The Unified Modelling Language includes a set of graphic notation techniques to create visual models of object oriented software-intensive systems.

The Unified Modelling Language was developed by Grady Booch, Ivar Jacobson and James Rambaug at Rational Software in the 1990s.It was adopted by the Object Management Group (OMG) in 1997, and has been managed by this organization ever since. In 2000 the Unified Modelling Language was accepted by the International Organization for Standardization (ISO) as industry standard for modelling software-intensive systems.

List of UML Diagram Types

There are many different types of UML diagrams. Some of them are listed below.

- Class Diagram
- Component Diagram
- Deployment Diagram
- Object Diagram
- Package Diagram
- Profile Diagram
- Composite Structure Diagram
- Use Case Diagram
- Activity Diagram
- State Machine Diagram
- Sequence Diagram
- Communication Diagram
- Interaction Overview Diagram
- Timing Diagram

Class Diagram

Class diagrams are arguably the most used UML diagram type. It is the main building block of any object oriented solution. It shows the classes in a system, attributes and operations of each class and the relationship between each class. In most modelling tools a class has three parts, name at the top, attributes in the middle and operations or methods at the bottom. In large systems with many classes related classes are grouped together to create class diagrams. Different relationships between diagrams are show by different types of Arrows.

Component Diagram

A component diagram displays the structural relationship of components of a software system. These are mostly used when working with complex systems that have many components. Components communicate with each other using interfaces are linked using connectors.

Deployment Diagram

A deployment diagrams shows the hardware of your system and the software in those hardware. Deployment diagrams are useful when your software solution is deployed across multiple machines with each having a unique configuration.

Object Diagram

Object Diagrams, sometimes referred as Instance diagrams are very similar to class diagrams. As class diagrams they also show the relationship between objects but they use real world examples. They are used to show how a system will look like at a given time. Because there is data available in the objects they are often used to explain complex relationships between objects.

Package Diagram

As the name suggests a package diagrams shows the dependencies between different packages in a system. z

Profile Diagram

Profile diagram is a new diagram type introduced in UML 2. This is a diagram type that is very rarely used in any specification.

Composite Structure Diagram

Composite structure diagrams are used to show the internal structure of a class. For a detailed explanation of composite structure diagrams

Use Case Diagram

Most known diagram type of the behavioural UML diagrams, Use case diagrams gives a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions are interacted. It’s a great starting point for any project discussion because you can easily identify the main actors involved and the main processes of the system.

Activity Diagram

Activity diagrams represent workflows in an graphical way. They can be used to describe business workflow or the operational workflow of any component in a system. Sometimes activity diagrams are used as an alternative to State machine diagrams.

State Machine Diagram

State machine diagrams are similar to activity diagrams although notations and usage changes a bit. They are sometime known as state diagrams or start chart diagrams as well. These are very useful to describe the behaviour of objects that act different according to the state they are at the moment. Below State machine diagram show the basic states and actions.

Sequence Diagram

Sequence diagrams in UML shows how object interact with each other and the order those interactions occur. It’s important to note that they show the interactions for a particular scenario. The processes are represented vertically and interactions are show as arrows.

Communication Diagram

Communication diagram was called collaboration diagram in UML 1. It is similar to sequence diagrams but the focus is on messages passed between objects. The same information can be represented using a sequence diagram and different objects.

Interaction Overview Diagram

Interaction overview diagrams are very similar to activity diagrams. While activity diagrams shows a sequence of processes Interaction overview diagrams shows a sequence of interaction diagrams. In simple term they can be called a collection of interaction diagrams and the order they happen. As mentioned before there are seven types of interaction diagrams, so any one of them can be a node in an interaction overview diagram.

Timing Diagram

Timing diagrams are very similar to sequence diagrams. They represent the behavior of objects in a given time frame. If its only one object the diagram is straight forward but if more then one objects are involved they can be used to show interactions of objects during that time frame as well.

Types of Uml tools

UML Tools are:

- Rational Rose
- MS Visio
- Enterprise Architecture
- UMLet
- DIA
- Umbrello

Rational Rose

Rational Rose is an object oriented Unified Modelling Language software design tool intended for visual modelling and component construction of enterprise-level software applications. In much the same way a theatrical director blocks out a play, a software designer uses Rational Rose to visually create (model) the framework for an application by blocking out classes with actors (stick figures), use case elements (ovals), objects (rectangles) and messages/relationships (arrows) in a sequence diagram using drag-and-drop symbols. Rational Rose documents the diagram as it is being constructed and then generates code in the designer's choice of C++, Java, Oracle8,Cobra or Data Definition Language.

Rational Rose is extensible, with downloadable add-ins and third-party partner applications. It supports COM/DCOM, JavaBeans, and Cobra component standards.

M S Visio

A drawing and diagramming program for Windows from Microsoft that includes a variety of pre-drawn shapes and picture elements that can be dragged and dropped onto the illustration. Users can define their own elements and place them onto the Visio palette. The Visio package is part of the Microsoft Office brand. Standard and professional editions are available.

Visio was introduced in 1992 by Seattle-based Shapeware, which coined its name from the pre- drawn shapes. It changed its name to Visio Corporation in 1995 when it went public and was acquired by Microsoft in 1999.

Enterprise architecture

Enterprise architecture (EA) is a conceptual blueprint that defines the structure and operation of an organization. The intent of an enterprise architecture is to determine how an organization can most effectively achieve its current and future objectives.

Microsoft's Michael Platt offers a view of enterprise architecture as containing four points-of- view, called the business perspective, the application perspective, the information perspective, and the technology perspective. The business perspective defines the processes and standards by which the business operates on a day-to-day basis. The application perspective defines the interactions among the processes and standards used by the organization. The information perspective defines and classifies the raw data (such as document files, databases, images, presentations, and spread sheets) that the organization requires in order to efficiently operate. The technology perspective defines the hardware, operating systems, programming, and networking solutions used by the organization.

Purported advantages of having an enterprise architecture include improved decision making, improved adaptability to changing demands or market conditions, elimination of inefficient and redundant processes, optimization of the use of organizational assets, and minimization of employee turnover.

UMLet:

UMLet, a small, very simple, and easy-to-use drawing tool (as opposed to a fully-fledged modelling tool) from the Technical University of Vienna. It's free for educational and non- commercial use. While it's pure Java, the jar file is so small (v1.0 is less than 60K) that it should run well for most people. This tool would work very well for classroom use since students can download their own copy and it's so simple (and customization), there's no real learning curve. This tool will not scale up to heavyweight

DIA:

DIA, a freeware/GPL drawing tool based on GTK; supports UML and other kinds of diagrams; this is intended to be a freeware Visio-like drawing tool rather than a modelling tool; I have a few UI quibbles, but overall this looks like a very nice drawing tool; DIA now comes standard with many Linux distributions.