Lecture 1

INTRODUCTION TO OBJECT-ORIENTED ANALYSIS AND DESIGN

Course Objectives

- To identify the problem in building large SW system and to give an overview of the process and methods to build such system.
- To present a study of OO systems analysis and domain modelling, using UML diagrams.
- To present a study of OO system design using UML.
- To introduce tools and methods for ensuring usable SW including testing.

Why is this important?

- Lots of project fail.. Or are not as successful as they should be
  - Poor quality
    - Low reliability
    - Low usability
  - High cost
- Complexity
  - It’s a team effort
  - Usually increases as a non- linear function of size
- Uncertainty
  - There are no canned global solutions
  - The context is always changing

LAS : A Lesson For Us All

- London Ambulance Service Automated Despatch Service.
- Abandoned shortly after delivery in 92
  - Cost £ 43 m
  - May have caused up to 30 deaths

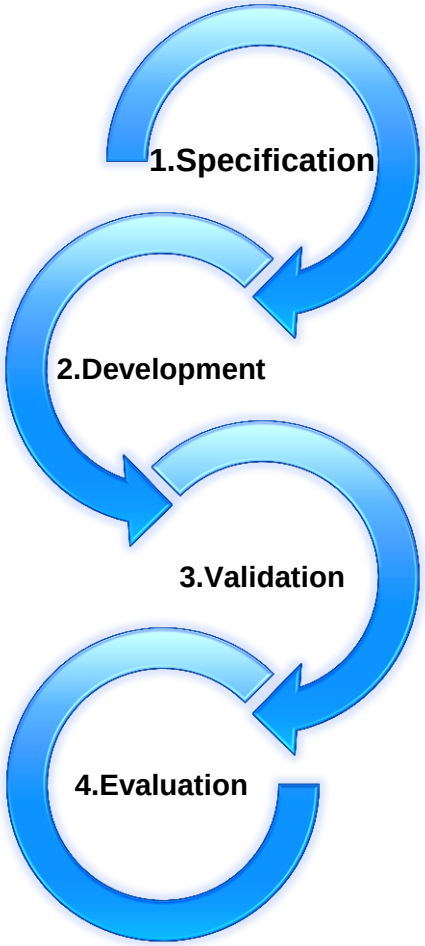
Building a system for a bad reason

Building Wrong System

Building system wrong

What are the problems?

Software Development Process



1. Identifying What's needed
2. Creating a way of satisfying the need
3. Determining if need is satisfied
4. Respond to changes to need or understanding of need

1. Control
2. Abstract
3. Packaging Solution for Re-Use
4. Evidence-based development

What are the solutions?

Control

- Key issues
  - Planning and scheduling
  - Risk Management
  - Change Control
  - Quality Assurance
  - Cost and Quality Estimation

Abstraction

- Modelling
  - **SW model** is a representation of a SW system or its context that captures selected aspects relevant to a development task
  - **Model-based development** uses these representations to capture a SW system and drive the development process.
  - **UML (** Unified Modelling Language) provides a set of models useful for developing OO-SW.

Packaging Solution for Reuse

- OO SW concepts and technology
  - Designed to make it easy to reuse designs or actual code while minimising need to make modifications.
- Software Patterns

Evidence-based Development

- Iterative design
  - Design process is driven by evaluation
  - Emphases need to reflect result of evaluation in changes to requirements and design
  - Design are claims
- User- centred design
  - Evaluation is central
- Test often

(A Simplified) Software Development Process Model

Gathering Specification

Create a Domain Model

Create a Sys. Model

Model Driven Development

- An approach in which the system is developed primarily by creating a set of representations (models)
- Start with a set of **analysis models**
- Use these to create a set of **design models** by adding
  - Detail to analysis models
  - New model elements as needed
- Generate **code**, preferably automatically

Inception

- A “vision” of the project
- The inception phase will generate **a business case** for undertaking the proposed work
  - How much it will cost?
  - What value it will return?
  - How long it will take?
  - What are the risks?
- Purpose of inception is **to get a agreement to proceed**

Requirements gathering and Specification

- Involves finding out what the customer wants and/or needs
- Final result will be
  - **A set of Use Cases**
    - A use case is a description of a typical interaction between a user and system to achieve a user goal
  - **A set of non-functional requirements**
    - other characteristic that the resulting system must possess
    - Constraints and limitation on the project and its outcomes.

The Analysis Phase: Creating a Domain Model

- The domain model is a “ high level” description of the system.
- This is the “big picture” of what the system contains, how the system will operate and the context in which it will operate.
- Differs from use case in that **it describes the information and operations that will provide the functionality needed to realise the use case.**
- In object-oriented development, the domain model consist of a collection of interacting objects.

The Design Phase

- Consists of
  - Architecture
    - Representation of how the complete system will be structured
    - Object may be organized into communicating subsystem
  - System Model
    - Identification of all the characteristics of the subsystems and their basic objects

Analysis Vs Design

- In both analysis and design stages, objects (and classes of objects) will appear in the model produced
- System Analysis:
  - Emphasis on finding & describing objects (or concepts) in the problem domain
  - Investigation of the problem and requirement rather than solution.
- System Design:
  - Emphasis on defining SW objects and how collaborate to fulfill the requirement.
  - A conceptual solution that fulfills the requirements, rather than implementation.

UML's Diagrams

- **Use Case Diagrams**
- Activity Diagram
- **Class Diagram**
- Collaboration Diagram
- **Sequence Diagram**

Analysis

- **Class Diagram & Sequence Diagram (again)**
- Statecharts
- Package Diagrams
- Component Diagram
- Deployment Diagrams

Design

UML is just a collection of ways of representing a system

UML is designed to support object oriented modelling

UML is not a development method or process

Objects

- In an OO approach, a system consist of a collection of interacting objects.
- An object is a computational entity which
  - Provide services with which other entities may interact
    - Typically, the services consumer sends a message (requesting the service) to the provides object
    - Possesses
      - Sate
        - Information that the object holds , called (attributes)
      - Behaviour
        - Operation it can perform , called (method)

Object in UML

- An object is represented by a named rectangle
- The name is a label for a particular object

Services

aLibrary

Provides service for finding books and checking books in and out

aBook

Provides service for ???

Refining Services to operations

aLibrary

In order to provides service for finding books, the library object might have to send book objects message asking for the book title

aBook

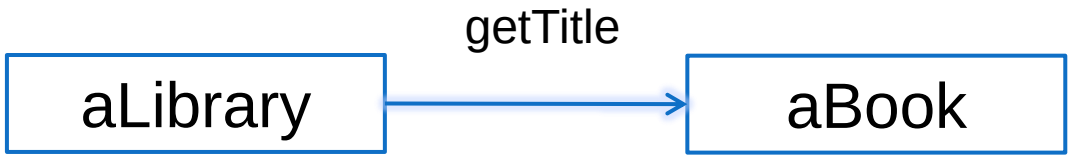
In order to provides service for querying, ???

Messages

- Interaction is achieved by one object sending a message to another
- Message has
  - A sender
  - A receiver
  - Message contents
    - A reference to an operation of the receiver
    - Possible additional information (parameters)
  - Some messages return information to the sender

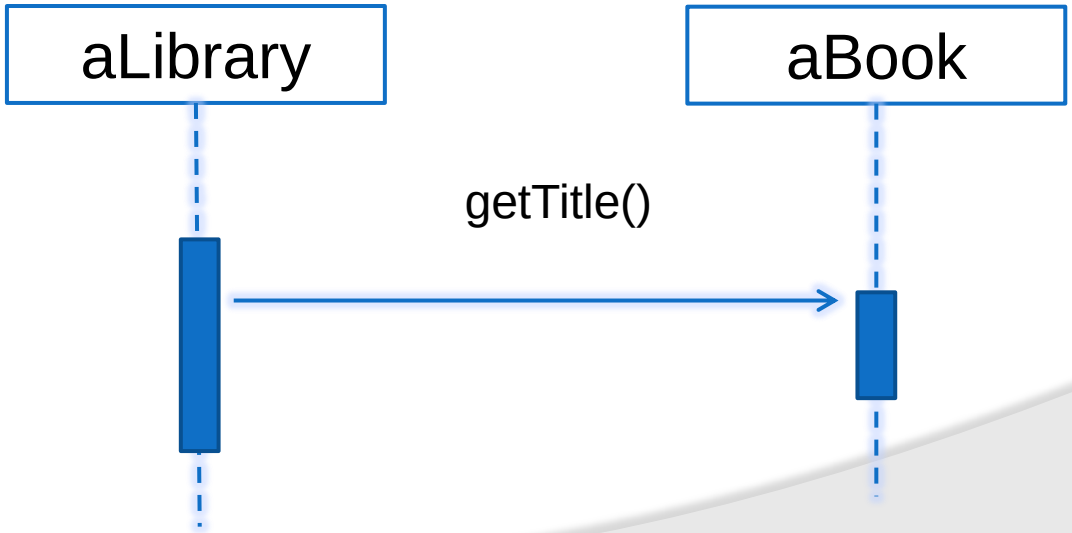
Message in UML

- Messages are shown as labelled arrows between objects



Message in UML

- Or with a representation of the operation (or method) that' invoked



Why Objects?

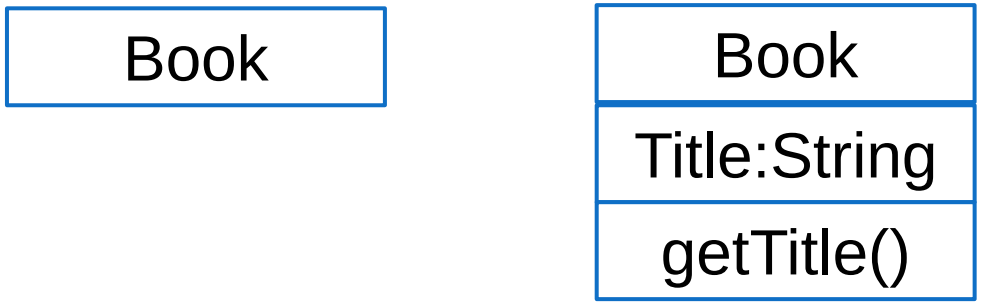
- Encapsulation
  - Helps to organise data & behaviour into meaningful associations.
- Message-based Invocation
  - Helps to make code adaptable and reusable
- Data Hiding
  - Helps to manage complexity, since programmer can control the data that objects are allowed to do manipulate.

Classes

- A Class
  - Describes a set of equivalent objects
    - Hence, operates like a data type of objects
    - Typically, every object in a system belongs to a class
  - Provides a useful way of representing and implementing the shared state and behaviour of the objects that it describes

Class in UML

- A class is represented by rectangle labelled with the class name



Why Class

- Inheritance
  - a form of reuse
- Polymorphism
  - means same operation may behave differently on different classes.

SUMMERY

- System development life cycle (SDLC)
- System analysis.
- System analysis vs. system design
- Unified Modeling Language (UML)

References

- Sommerville,6th edition, Chapter 12, 14, 7
- PSD(M) Lecture Notes, GU, UK