5 Requirements Modelling

Requirements Analysis:

- Specifies software's operational characteristics.
- Indicate software's interface with other system elements.
- Establishes constraints that software must meet.

Requirements analysis allows the software engineer to:

- Elaborate basic requirements
- Build models that describe user's needs from several perspectives

Requirements Models:

- Scenario-based models: point of view of 'actors'
- Class oriented models: how classes collaborate to achieve system requirements
- Behavioral models: depict how the software reacts to internal or external events
- Data models: Depict the information domain for the problem
- Flow oriented models: Represent functional elements of the system and how they transform data in the system.

Analysis model provide insight into information domain, function, and behaviour of the software
& provides value to all stakeholders keep the model as simple as it can be.

Scenario based modelling: Actors and profiles
UML actor models interact with a system object: actor role is played by human stakeholders
UML profile provides a way of extending an existing model to other domains or platforms, allow you to revise the model of a web based system

Use Cases:

- Use cases are simply an aid to defining what exists outside the system (actors) and what should be performed by the system

Primary scenarios: beginning of inception and elicitation for information you need to write use cases.

Secondary scenarios: can the actor take some action? & is it possible actor will encounter some error? - To answer these questions, you will need alternative use cased behaviour.

Exception: a situation failure condition or an alternative chosen by the actor that causes the system to behave differently.

Class Based Modelling:

- Class based modelling represents:
  - Objects that the system manipulate
  - Operations that will be applied to the objects to effect the manipulation
  - Relationships between the objects.
- Class based model include: classes and objects, attributes, operations, CRC models, UML class diagrams.

Identifying Analysis Classes:

- Examining the usage scenarios developed as part of the requirements model and perform a “grammatical parse”.

Potential Analysis Classes:

- Part of information domain for the problem.

Analysis Class Selection:

- Retained information: Potential class will be useful during analysis only if information about it must be remembered.

Defining Attributes: Attributes describe a class that has been selected for inclusion in the analysis model.

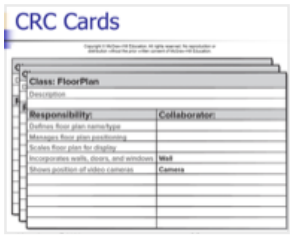
- It is the attributes that define the class - that clarify what is meant by the class in the context of the problem space.
- Data items define in the class

Operations: define the behaviour of an object.

CRC Modeling:
Class Responsibility Collaborator modelling provides a simple means for identifying and organizing the classes that are relevant to system or product requirements.

- CRC model is really a collection of standard index cards that represent classes.
- The cards are divided into three sections:
  - Top: name of the class
  - List of class responsibilities on the left
  - List of collaborators on the right

e.g



All stakeholders in the review are given a subset of the CRC model index cards.

Functional Modelling: Addresses two application processing elements

- User observable functionality that is delivered by the app to end users.
- Operations contained within analysis classes that implement behaviours associated with the class.

UML activity diagrams can be used to represent processing details.
UML activity diagram supplements a use case by providing a graphical representation of the flow of interaction within a specific scenario.

Sequence Diagram:

- The UML sequence diagram can be used for behavioral modelling.
- Sequence diagrams can also be used to show how events cause transitions from object to object.
- Sequence diagram is a shorthand version of a use case.

Behavioral modelling:

- Indicates how software will respond to internal or external events
- This information is useful in the creation of an effective design for the system to be built
- UML activity diagrams can be used to model how system elements respond to internal/external events.

To create behavioural models: evaluate all use cases to fully understand, identify events that drive the interaction sequence and understand how these events relate to specific objects, create sequence diagram for each use case, built a state diagram for the system

Identifying Events:
A use case represents a sequence of activities that involves actors and the system
Events are used to trigger state transitions.

Swimlane Diagrams:

- UML swim lane diagram is a useful variation of the activity diagram that allows you to represent the flow of activities described by the use case
- It indicates which actor (multiple actors may involved) or analysis class

has responsibility for the action described by an activity rectangle
- Responsibilities are represented as parallel segments