IF3141
SISTEM INFORMASI
System Modelling Technique

Semester I 2023/2024

1

System

- A model is a representation of a subject. Typically that subject is a real-world object or system that either already exists or may exist in the future.

2

What is System Modelling?

- A model is a representation of a subject. Typically that subject is a real-world object or system that either already exists or may exist in the future.
- System characteristics:
  - A system has a boundary outside of which are external entities (people or other systems) that interact with the system in focus.
  - Within this system are components (systems, people) that implement that system's behaviour.
  - These interactions and internal system behaviours involve data being exchanged, transformed and stored.
  - Any system exists within a broader context that needs to be understood and the system should be defined in terms of how it delivers functionality and services within this context, usually in the form of requirements.
- Modelling is the act of producing appropriate models of systems through the fundamental act of abstraction that allows all these issues to be examined, explored and refined prior to development of the system elements

3

Abstraction

- In philosophical terminology, abstraction is the thought process wherein ideas are distanced from objects. In system modelling, abstraction is the process of removing anything unnecessary or irrelevant from a system model (or the view of that model) depending on its purpose.
- Different system models tend to focus on specific facets of the system, for example its functionality or implementation. A single model is therefore an incomplete abstraction of the real thing but a combination of models can between them provide a more complete description, providing they are consistent with each other.

- 'The model is not the reality, but the best models are the ones that stick very close to reality.' (Booch, et al., 2005)

4

All models are wrong

- Any model, as an incomplete abstraction, is therefore imperfect.
- Remember that all models are wrong; the practical question is – how wrong do they have to be to not be useful?
  (Box and Draper, 1987)
- Usefulness is the key, although intangible, measure of the value of any model. Usefulness can be increased by
  - abstracting out details irrelevant to the facet being focused on,
  - providing that what remains in that model is effective in communicating or describing that facet to the appropriate level.
    - What is appropriate and effective usually depends on the nature of the intended audience. A representation that works effectively for a technical developer is unlikely to be equally useful for a business stakeholder and vice versa.

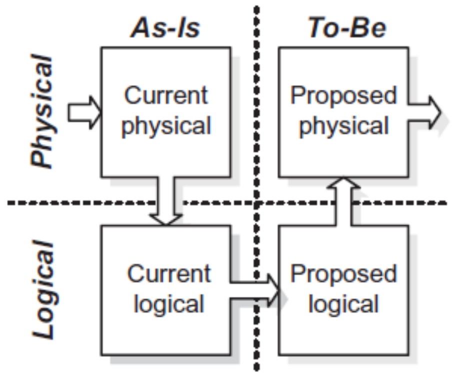
5

Structure over words

- Modelling does not need to be visual, although as the saying goes ‘a picture paints a thousand words’.
- System modelling typically involves a combination of visual representations (diagrams) and structured textual descriptions of:
  - model elements; for example, tasks in a process model, classes or entities in a data model;
  - links and dependencies between elements; for example, arrows that show the sequence of tasks or lines that represent some semantic dependency between classes or entities.

6

The U curve



7

Rationale for modelling

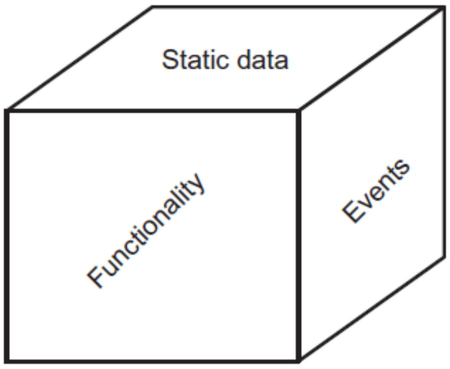
- Modelling can be employed within solution development with the ultimate aim of developing a solution which meets key stakeholder requirements; in other words a quality solution.

- What does modelling facilitate?
  - Communication and understanding
  - Experimentation
  - Validation
  - Gap analysis

- The value of modelling
  - Value from the act of modelling
  - Value of the delivered model

8

The three-view model of a system



9

Pre-UML Modelling Techniques

- Functionality (Functional Facet)
  - data flow diagram (DFD);
  - flow chart (including variants such as swimlane diagrams);
  - Jackson structure chart.
- Static data (Structural Facet)
  - entity relationship diagram (ERD), also known as a Logical data model;
  - hierarchical tree diagram;
  - bubble diagram.
- Events (Behavioral Facet)
  - Entity Life History (ELH), for example using Jackson structures;
  - Statechart, for example those promoted by Harel pre-UML (see Figure 7.5).

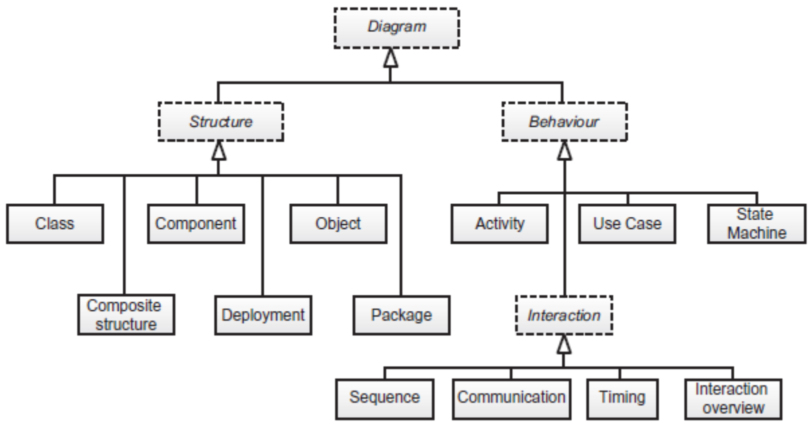
10

UML

- Functionality (Functional Facet)
  - use case diagram;
  - activity diagram;
  - interaction diagrams:
    - sequence diagram
    - communication diagram
    - timing diagram
    - interaction overview diagram.
- Static data (Structural Facet)
  - Class diagram
- Events (Behavioral Facet)
  - State machine

11

UML



12

Abstraction, Levelling and Scope

- Classification
- Generalisation
- Composition
- Idealisation
  - Contextual model
    - an unstructured representation that helps to outline the scope and purpose of an element or system. It provides the background for the other levels
  - Conceptual model
    - a structured description and high-level detail about the element without regard to implementation, for example a business process or data structure. Conceptual models are often incomplete or not structurally robust.
  - Logical model
    - Logical models should be more complete and structurally robust while still ignoring implementation issues
  - Physical model
    - Physical model considers implementation issues, such as technology-specific details, physical location details and so forth.

13

Opaqueness of Model Elements

- Black-box elements
  - where we are not concerned about its inner workings, but only concerned with it as a whole and its links and dependencies on other elementsflow chart (including variants such as swimlane diagrams);
  - Example:
    - Context diagram
    - Use case diagram
- White-box views
  - where we are interested in its internal workings

14

Cross-referencing Models

- Producing a number of models is all well and good, but in order to be effective they need to be consistent with each other in terms of:
  - validating that model elements can be traced back to underlying business requirements;
  - cross-referencing elements in one view or model to related elements in other models where there is some dependency; for example, certain events can be mapped to use cases, which in turn are mapped to the static data they manipulate;
  - ensuring different views are modelling consistent system scopes;
  - checking that models and views conform to the same semantics; i.e. calculations and conditions in Physical models correctly implement business rules defined in Conceptual and Logical models.

15

Traceability

- Traceability from business requirements to deliverables via intervening models can reveal the following:
  - which requirements have been met or not;
  - how to plan incremental delivery based on requirement priority
  - which deliverables, and intervening models, are effected by a proposed requirements change request; the first dimension of impact analysis

- Tools:
  - Matrices

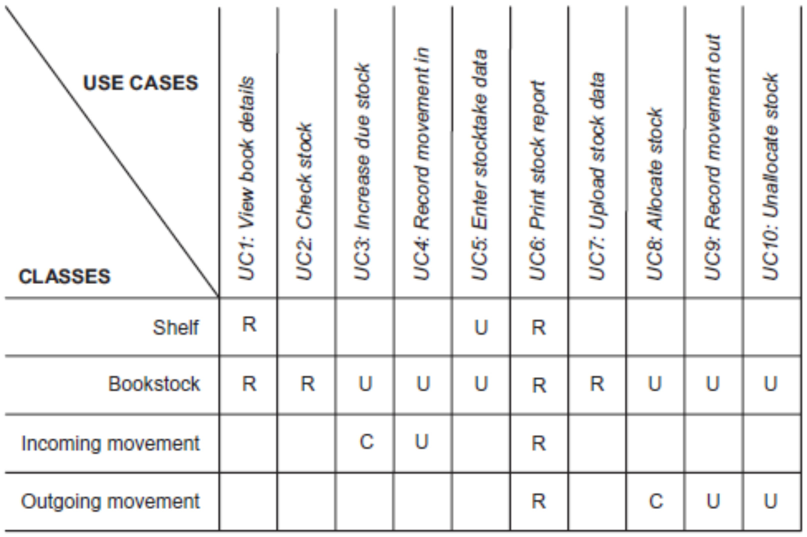
16

Matrices Examples

- Mapping functional business requirements to system use case: this reveals which requirements have and, more importantly, have not been incorporated into the system functional model.
- Mapping entities in a Conceptual data model to those in the Logical data model or on to physical database tables: at the first level, this is done to ensure that all the data entities have been incorporated; then to indicate where the data semantics (meaning and rules) need to be checked for consistency.
- Mapping external entities of a system (actors) to the services they require from the system.
- Mapping events to the functional elements that are triggered.
- Mapping functional elements to static data elements at any level of idealisation: a commonly used example of this is referred to as the CRUD matrix.
- Mapping events to the data elements that are likely to be effected can also be shown in a CRUD matrix.

17

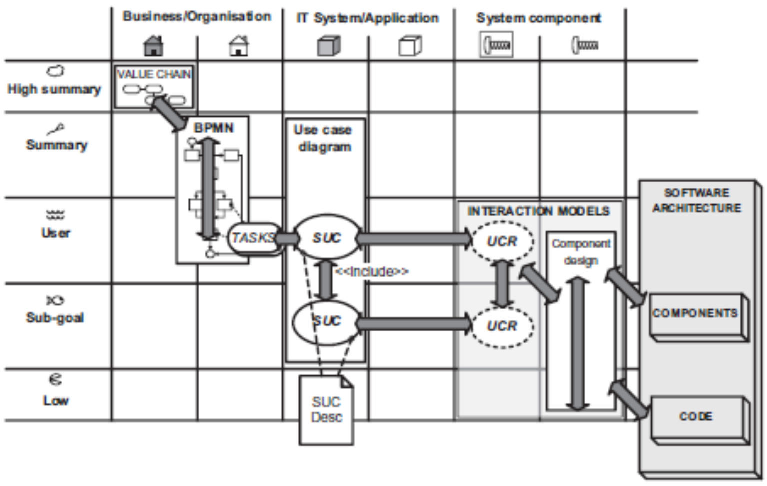
CRUD Matrix



USE CASES / CLASSES	UC1: View book details	UC2: Check stock	UC3: Increase due stock	UC4: Record movement in	UC5: Enter stocktake data	UC6: Print stock report	UC7: Upload stock data	UC8: Allocate stock	UC9: Record movement out	UC10: Unallocate stock
Shelf	R				U	R				
Bookstock	R	R	U	U	U	R	R	U	U	U
Incoming movement			C	U		R				
Outgoing movement						R		C	U	U

18

Functional Model Map



overlaid with traceability paths from requirements to code

19

Documentation

- Waterfall lifecycle documentation
  - 1. business case and feasibility study;
  - 2. requirements document;
  - 3. system specification document;
  - 4. system design document;
  - 5. test documentation;
  - 6. implemented system documentation:
    - a. operations manual
    - b. user documentation
    - c. training materials.
- Agile documentation
  - a business case defining scope and providing the project justification with which to control the project;
  - evidence of testing and quality control;
  - documentation for the released product (ops manual, user documentation and training materials).

20

Which model?

- A significant factor in these choices will be practical ones such as:
  - what modelling skills team members possess;
  - what tools, whether pen and paper, diagramming software or modelling software are available to the team;
  - what standards are adopted within the organisation or even industry.

- Most important of all, before spending time and effort modelling each artefact:
  - Is it or will it be useful?

21