Modelling

- This is the step of synthesising all the requirements you collected into key requirements
- You start making diagrams that represent how requirements relate, using diagrams
- Then you can produce a comprehensive set of requirements

What are models?

Models are selective **representations** to portray **some aspect** that is considered to be **important**. A model is an **abstraction** of the system being studied rather than an **alternative representation** of that system … An abstraction **deliberately** simplifies and picks out the most **salient characteristics**.

The purpose of models

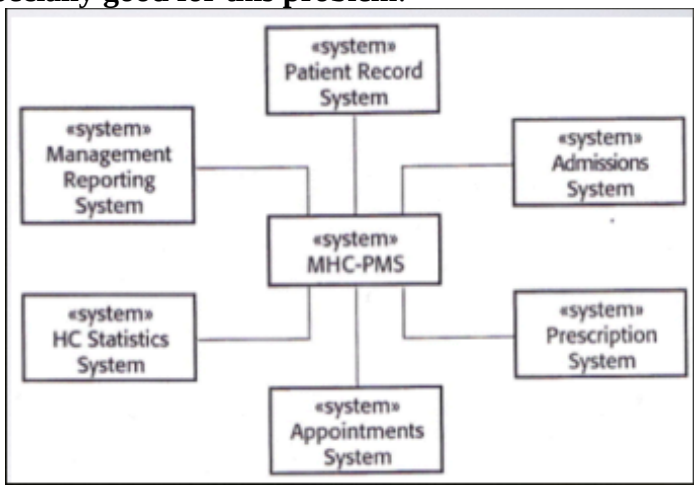
- Derive the requirements for a system
- During design process describe the system to engineers implementing the system
- After implementation to document the systems structure and operations

Stage 1

After figuring out all the systems that software has to work with, you need to model it. User case diagrams **can do it a bit**, though Context Models are **especially good for this problem**.

Context models

- Allow you to represent the context of new software
- Related systems
- Important constraints
- Defines **boundaries** of the system
  - Represents **key** services that need to be developed, and for who
  - The services represent the relationship to the other components
  - Also what **not** to develop



Stage 2

Need a good way of documenting all the subtasks necessary to do that task.

Task analysis

- A complete breakdown of what has to be done
- Not how a system is used
- Define goals
- Identify and list tasks
- Decompose the components of each task
- Identify dependencies and sequences
- Validate the process with stakeholders

Benefits

- Detailed task-focused requirements for a system
- Scope content/use of a system
- Structure the interface of a system
- Define procedures for a system
- Help build better scenarios and personas

Stage 3

It’s fine modelling subtasksand subtasks of a task. But that does not give you a picture of how different tasks work together. You want to be able to explain :

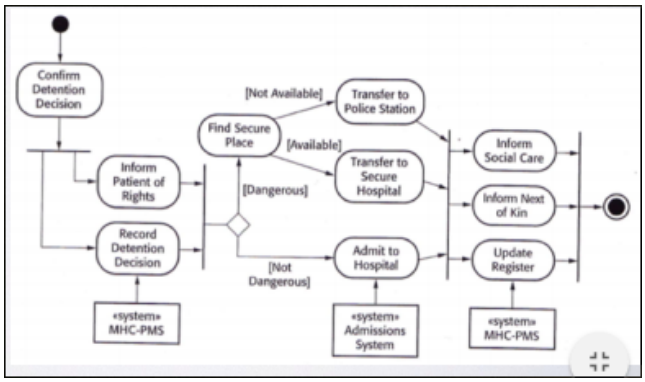
- The different stages of the process
- How will different people contribute

Unified modelling language (UML)

- Use case diagrams – who does what in the system
- Activity Diagrams – activities in the process
- Sequence Diagrams – interaction between user and system
- State Diagrams – possible states of “things” (objects)

UML Activity diagrams

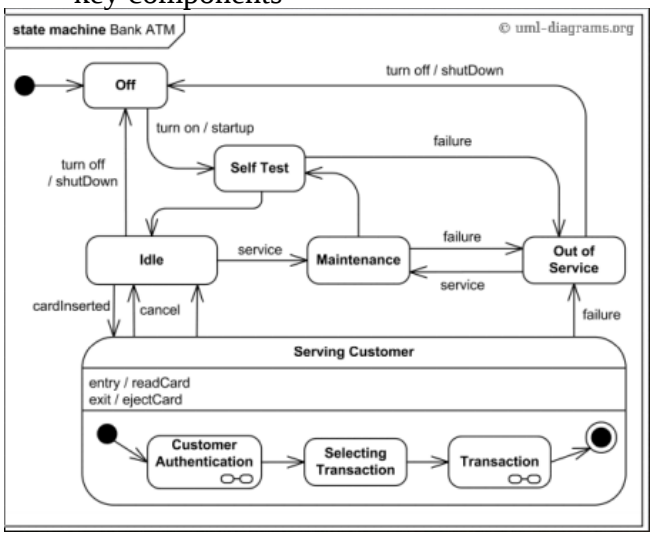
used to elaborate workflows for key activities – especially if they involve decisions.

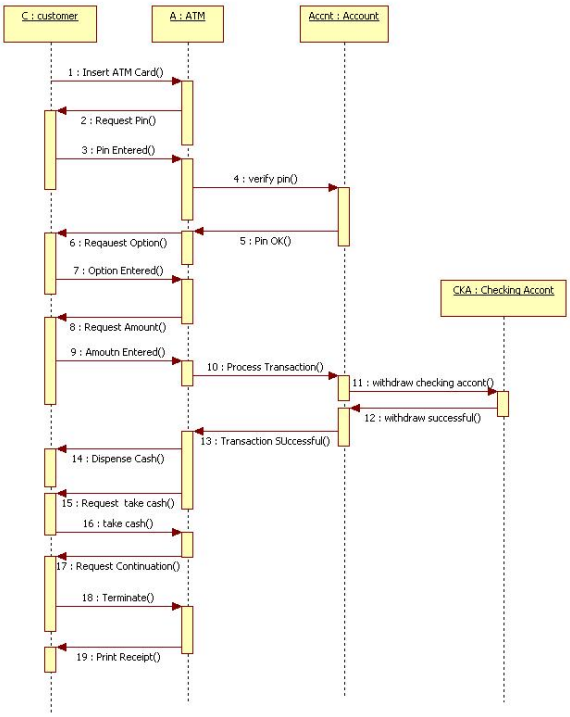


- Explains the process, decision points, wait points and parallel work
- May tie together several Use Cases for one of the main Activities.
- Or define one Use Case in more detail
- Rounded rectangles represent actions
- Diamonds represent decisions
- Bars represent the start (split) or end (join) of concurrent activities
- The black circle represent the start
- The Encircled black circle represent the end

Sequence diagrams

- Good for complex sharing of information between people and the system
- Could be seen as a series of messages between key components





State diagrams

- All the states that object can be in
- What state in can change to

Stage 4

You have a lot of information visualisation like diagrams, personas, etc. but you don’t quite have a good picture of what its like. Need a way to summarise what its like to go through them

Scenarios

Envision and document typical and significant user activities.

- Adding context/detail to models
- Not just a story but a structured description of the process.
- Muse **define a setting** or context
  - Might come from a technology tour
  - The environment, the things in it, etc
- Must define one or more **actors or users** (perhaps a persona)
- Must define **goals or objectives** (perhaps from a task analysis or user story)
- Must **describe a plot** as a sequence of events (perhaps from task analysis)
  - The plot describes how a user, in a context, achieves a goal
- Scenarios are written text descriptions describing current events OR describing ideal or possible events in design(Up to 5 persona if its helpful)

Summary of models

- Identify something that needs modelling
- Decide what type of diagram it should be
- Explain what the diagram is going to model
- Cross reference it with other diagrams.

Requirements/Specifications document

SRS = Software Requirements Specification (Usually a big set of nested lists)

Should now be able to produce a specific list of requirements, at least **categorised** by importance, if not by other things. Where the models/diagrams/personas/user stories etc. - can be cross referenced with the list

Chapter	Description
Preface	This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version.
Introduction	This should describe the need for the system. It should briefly describe the system's functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software.
Glossary	This should define the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader.
User requirements definition	Here, you describe the services provided for the user. The nonfunctional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified.
System architecture	This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted.
System requirements specification	This should describe the functional and nonfunctional requirements in more detail. If necessary, further detail may also be added to the nonfunctional requirements. Interfaces to other systems may be defined.
System models	This might include graphical system models showing the relationships between the system components and the system and its environment. Examples of possible models are object models, data-flow models, or semantic data models.
System evolution	This should describe the fundamental assumptions on which the system is based, and any anticipated changes due to hardware evolution, changing user needs, and so on. This section is useful for system designers as it may help them avoid design decisions that would constrain likely future changes to the system.
Appendices	These should provide detailed, specific information that is related to the application being developed; for example, hardware and database descriptions. Hardware requirements define the minimal and optimal configurations for the system. Database requirements define the logical organization of the data used by the system and the relationships between data.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on.