Requirements Engineering

Software Engineering 2

(3103313-1)

Amirkabir University of Technology

Fall 1399-1400

Requirements Engineering

- The process of understanding and defining what services are required from the system and identifying the constraints on the system's operation and development

- Converting high level business requirements (from the system request) into detailed requirements that can be used as inputs for the following steps (e.g., creating models).

- The single(!) most critical step of the entire software development
- Changes can be made easily in the requirements
- Most (>50%) system failures are due to problems with requirements

The importance of Requirements Discovery

- One of the primary challenges is the ability to elicit the correct and necessary system requirements from the *stakeholders* and specify them in a manner understandable to them so those requirements can be verified and validated.

> *The hardest single part of building a software system is **deciding precisely what to build**. No other part of the conceptual work is a difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.*
>
> --Fred Brooks--

Standish Group Report - CHAOS

Project Impaired Factors	% of Responses
1. Incomplete Requirements	13.1%
2. Lack of User Involvement	12.4%
3. Lack of Resources	10.6%

Project Challenged Factors	% of Responses
1. Lack of User Input	12.8%
2. Incomplete Requirements & Specifications	12.3%
3. Changing Requirements & Specifications	11.8%

Project Success Factors	% of Responses
1. User Involvement	15.9%
2. Executive Management Support	13.9%
3. Clear Statement of Requirements	13.0%

Results of Incorrect Requirements

- The system may cost more than projected.
- The system may be delivered later than promised.
- The system may not meet the users' expectations and they may not to use it.
- Once in production, costs of maintaining and enhancing system may be excessively high.
- The system may be unreliable and prone to errors and downtime.
- Reputation of IT staff is tarnished as failure will be perceived as a mistake by the team.

Relative Cost to Fix an Error

Phase in Which Error Discovered	Cost Ratio
Requirements	1
Design	3–6
Coding	10
Development Testing	15–40
Acceptance Testing	30–70
Operation	40–1000

Requirements

- Requirement: A statement of what the system must do or what characteristic it must have

- Initially, requirements are written from the perspective of the business persons.
  - Will later evolve into a technical description of how the system will be implemented.

- Conventionally, two kinds of requirements
  1. Functional: relates to a process or data
  2. Nonfunctional: relates to performance or usability

Requirements

Categories

- Business
- User
- System
- Implementation

- Functional
- NonFuntional

- And?

- FURPS
  - Functionality, Usability, Reliability, Performance, and Scalability

- FURPS+
  - FURPS with missing categories
  - Design constraints
  - Implementation requirements
  - Interface requirements
  - Physical

Requirements Engineering

Activities

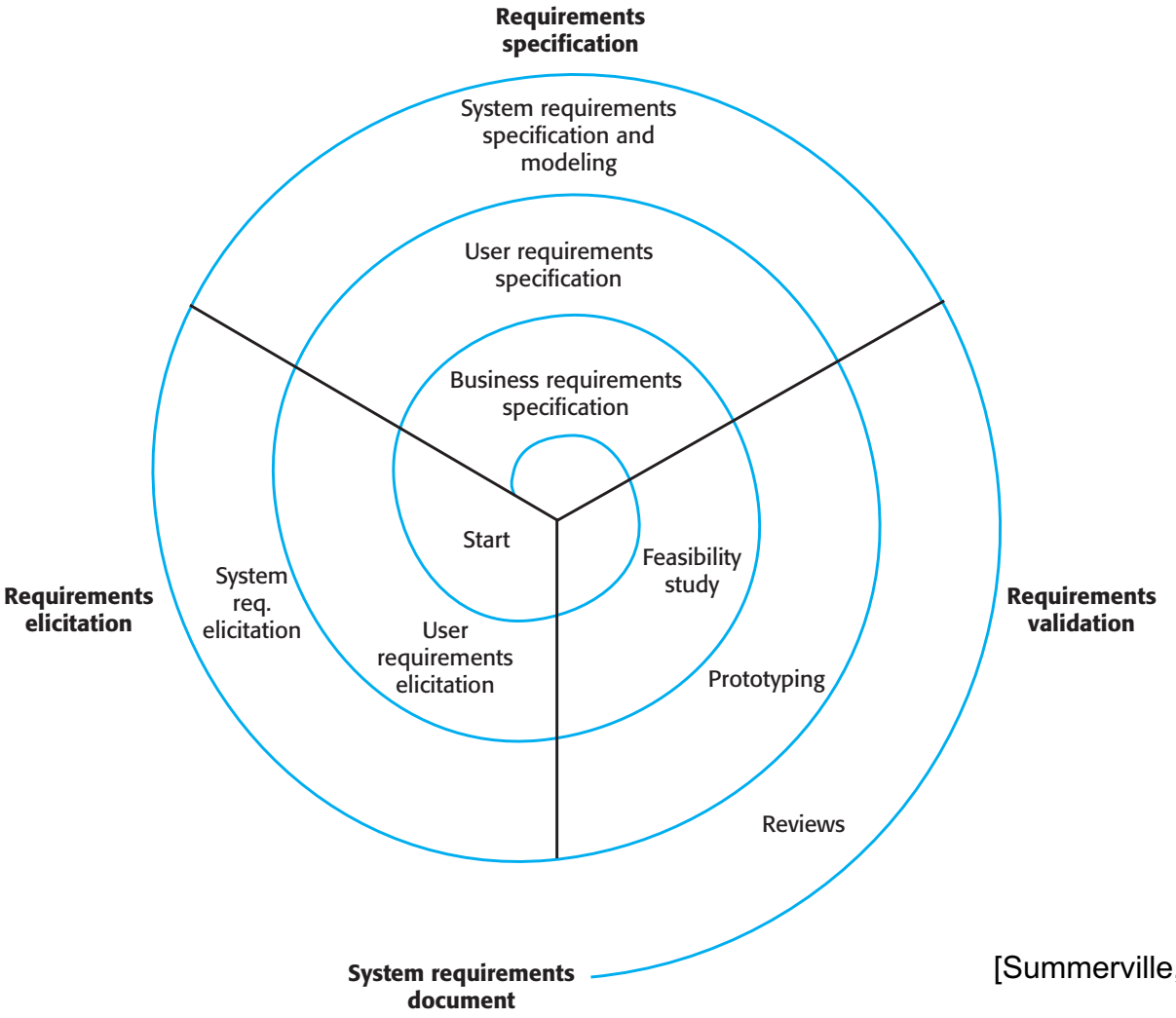
- Inception
- Elicitation
- Elaboration
- Negotiation
- Specification
- Validation
- Management

- Elicitation and Analysis
- Specification
- Validation
- & …!

- Scope Definition
- Problem Analysis
- Requirements Analysis
- Logical Design
- Decision Analysis

Requirements Engineering

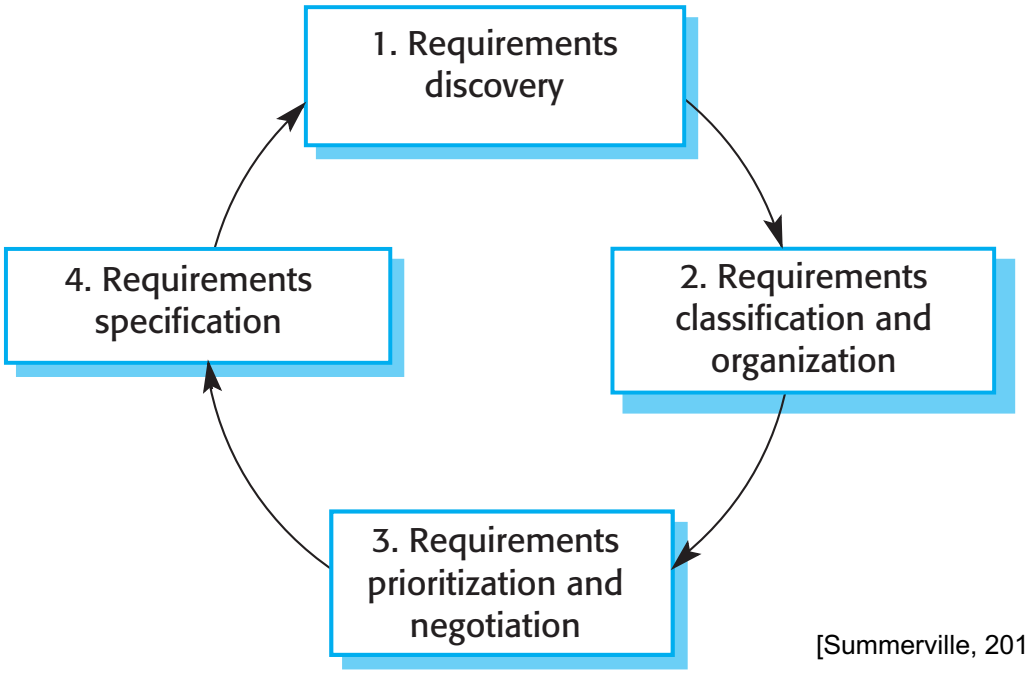
Activities



[Summerville, 2016]

Requirements Elicitation

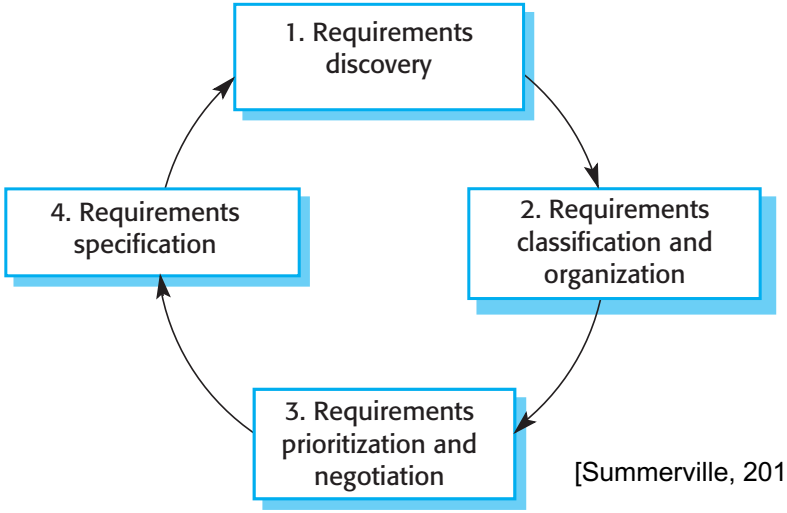
- Working with a range of system stakeholders to find out about
  - the application domain, the services that the system should provide, the required system performance, hardware constraints, other systems, etc.



[Summerville, 2016]

Requirements Elicitation

- Techniques
  - Interview
  - Observation or Ethnography
  - Brainstorming
  - ?



[Summerville, 2016]

- Methods
  - Scenario-driven
    - description of how the system can be used for some particular task.
  - Five Ws (and One H)
    - Ws (who, what, when, where, and why) and one H (how)
  - ?

Requirements Elicitation

Appropriate Technique

	Interview	JAD	Questionnaires	Document Analysis	Observation
Type of information	As-is, improves, to-be	As-is, improves, to-be	As-is, improves	As-is	As-is
Depth of info	High	High	Medium	Low	Low
Breadth of info	Low	Medium	High	High	Low
Info integration	Low	High	Low	Low	Low
User involvement	Medium	High	Low	Low	Low
Cost	Medium	Low-medium	Low	Low	Low-medium

- A combination of techniques may be used
- Document analysis & observation require little training; JAD sessions can be very challenging

Requirements Specification

Notation	Description
Natural language	The requirements are written using numbered sentences in natural language. Each sentence should express one requirement.
Structured natural language	The requirements are written in natural language on a standard form or template. Each field provides information about an aspect of the requirement.
Graphical notations	Graphical models, supplemented by text annotations, are used to define the functional requirements for the system; UML use case and sequence diagrams are commonly used.
Mathematical specifications	These notations are based on mathematical concepts such as finite-state machines or sets. Although these unambiguous specifications can reduce the ambiguity in a requirements document, most customers don't understand a formal specification. They cannot check that it represents what they want and are reluctant to accept it as a system contract

[Summerville, 2016]

Sample of Requirements Definition

Nonfunctional Requirements

1. Operational Requirements
1.1. The system will operate in Windows environment.
1.2. The system should be able to connect to printers wirelessly.
1.3. The system should automatically back up at the end of each day.

2. Performance Requirements
2.1. The system will store a new appointment in 2 seconds or less.
2.2. The system will retrieve the daily appointment schedule in 2 seconds or less.

3. Security Requirements
3.1. Only doctors can set their availability.
3.2. Only a manager can produce a schedule.

4. Cultural and Political Requirements
4.1. No special cultural and political requirements are anticipated.

Functional Requirements

1. Manage Appointments
1.1. Patient makes new appointment.
1.2. Patient changes appointment.
1.3. Patient cancels appointment.

2. Produce Schedule
2.1. Office Manager checks daily schedule.
2.2. Office Manager prints daily schedule.

3. Record Doctor Availability

Requirements Validation

- Checking that requirements define the system that the customer really wants.

- **Consistent** – not conflicting or ambiguous.
- **Complete** – describe all possible system inputs and responses.
- **Feasible** – can be satisfied based on the available resources and constraints.
- **Required** – truly needed and fulfill the purpose of the system.
- **Accurate** – stated correctly.
- **Traceable** – directly map to functions and features of system.
- **Verifiable** – defined so can be demonstrated during testing.

A Bad Requirement

Initial Specification: Software will not be loaded from unknown sources onto the system without first having the software tested and approved.

Critique:

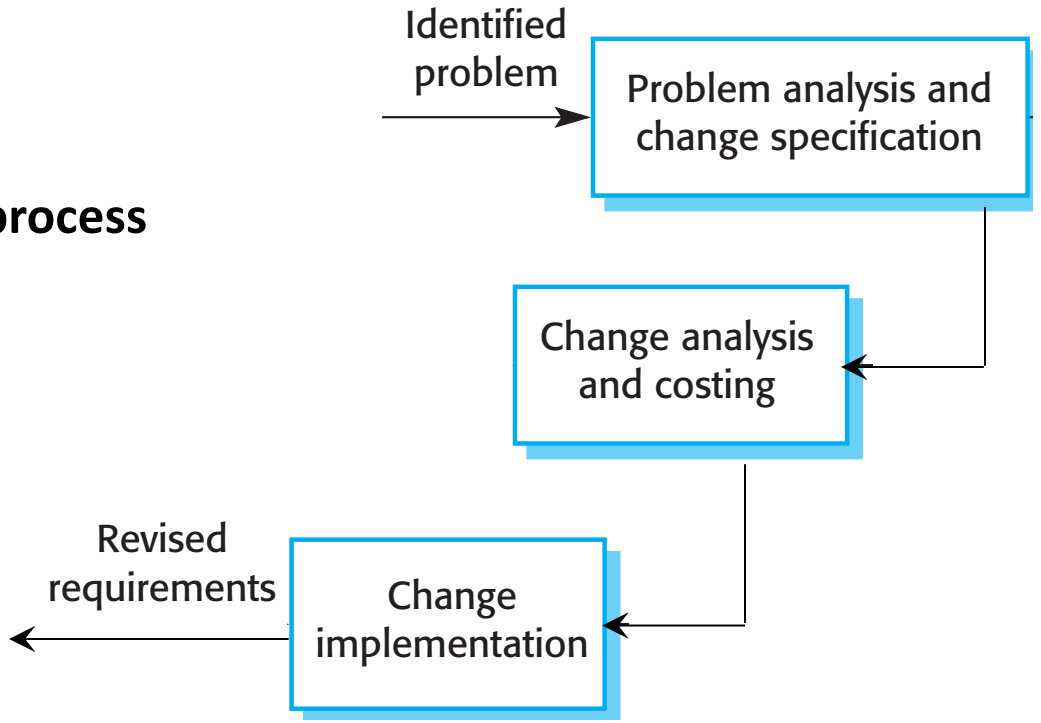
- Ambiguous – if the software is tested and approved, can it be loaded from unknown sources?
- (not) Testable – it is stated as a negative requirement making it difficult to verify.
- (not) Traceable – a unique identifier is missing.

Re-specification: 3.4.5.2 Software shall be loaded onto the operational system only after it has been tested and approved.

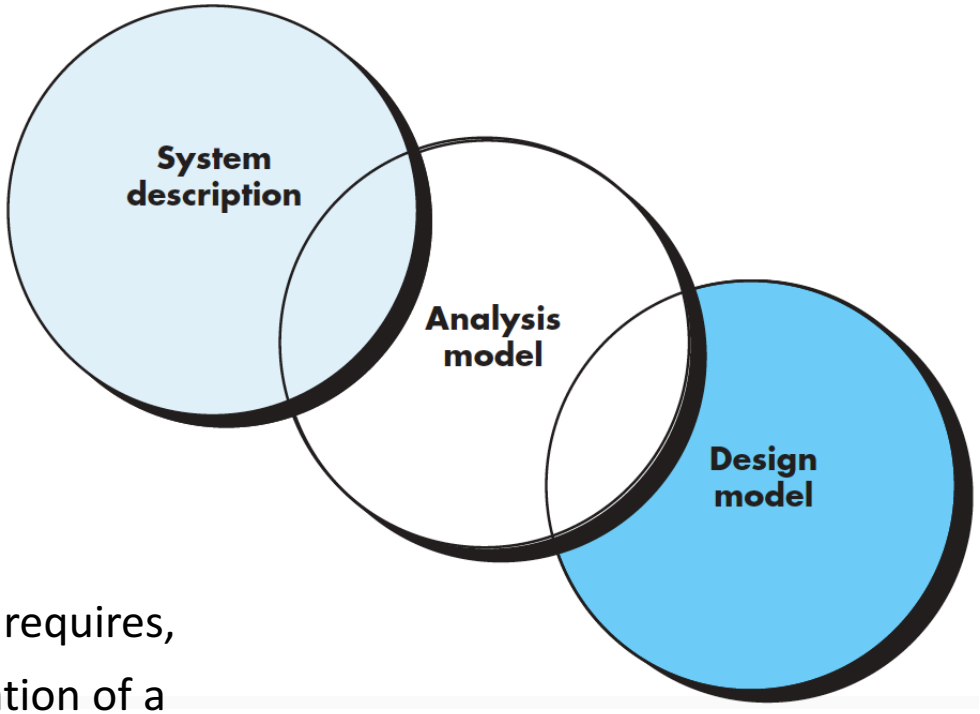
Requirements Management

- Process of managing changing requirements during the requirements engineering process and system development.

- **A change management process**
- Traceability policies
- Tool support



Requirements Modelling

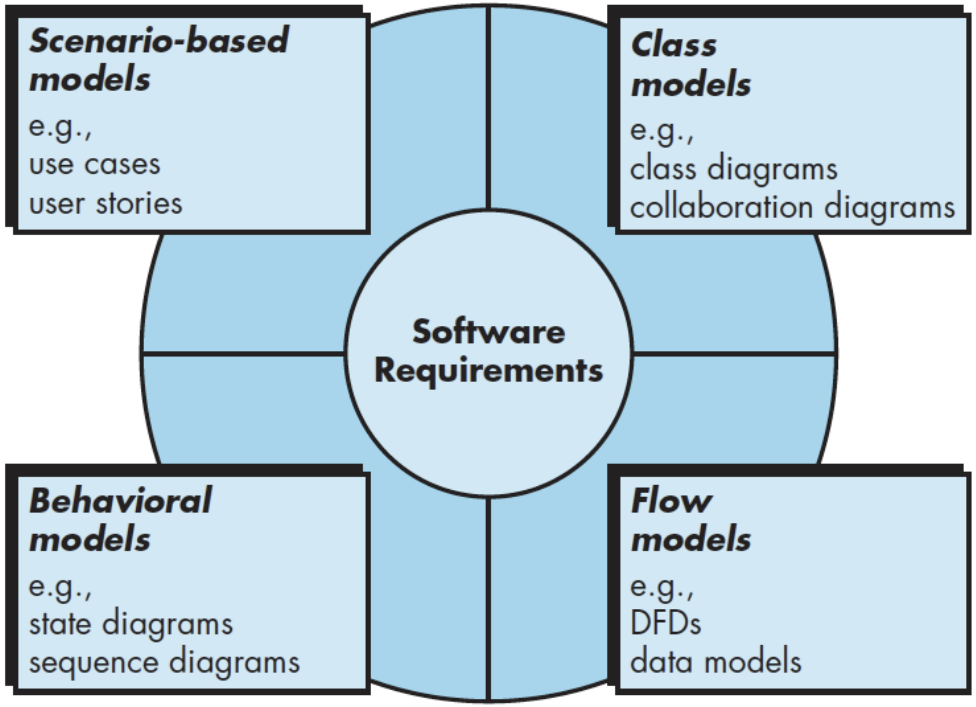


1. Describe what the customer requires,
2. Establish a basis for the creation of a software design, and
3. Define a set of requirements that can be validated once the software is built.

[Pressman, 2015]

Requirements Modelling

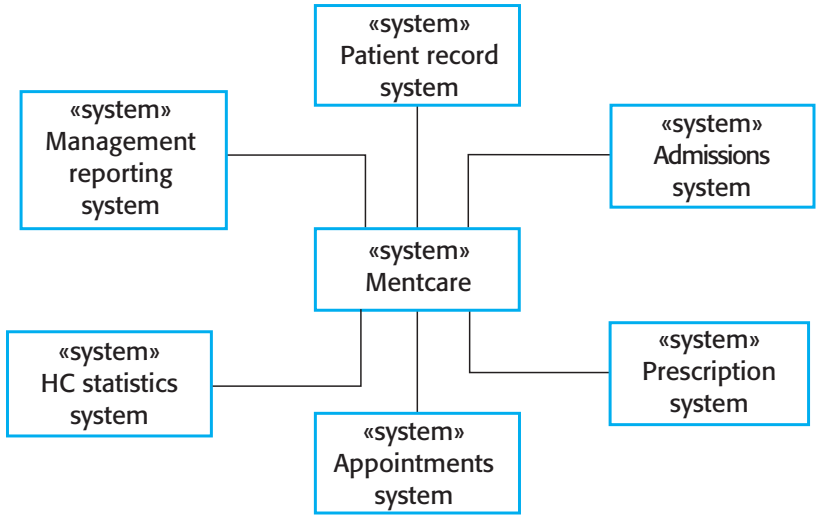
- Context models
- Interaction models
- Structural models
- Behavioral models



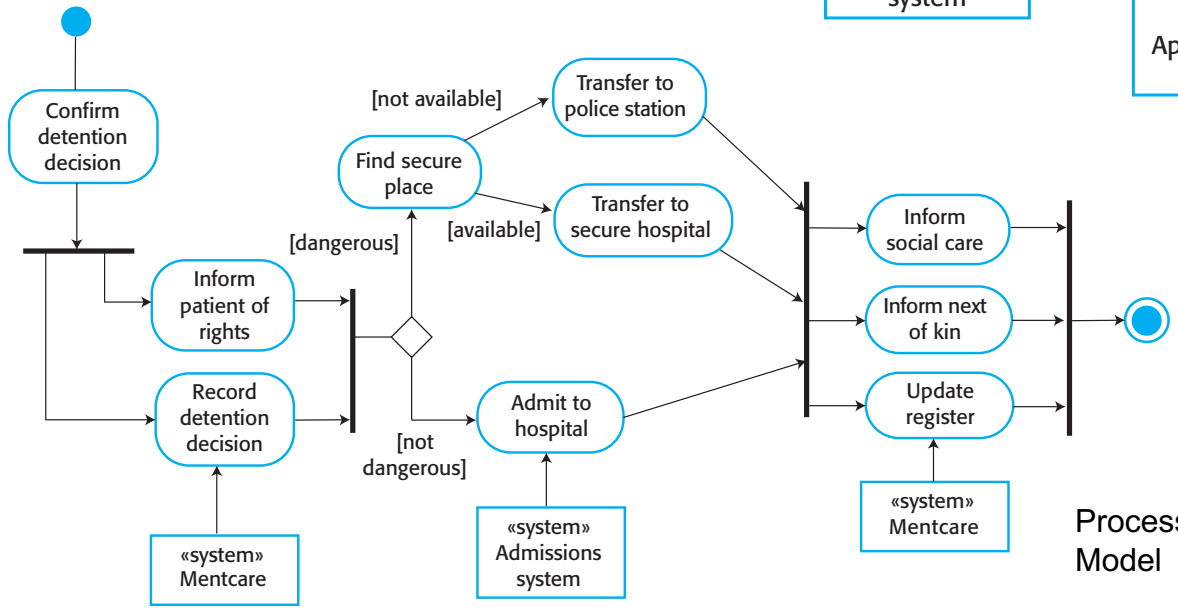
[Pressman, 2015]

Requirements Modelling

- **Context models**
- Interaction models
- Structural models
- Behavioral models



Context of the System

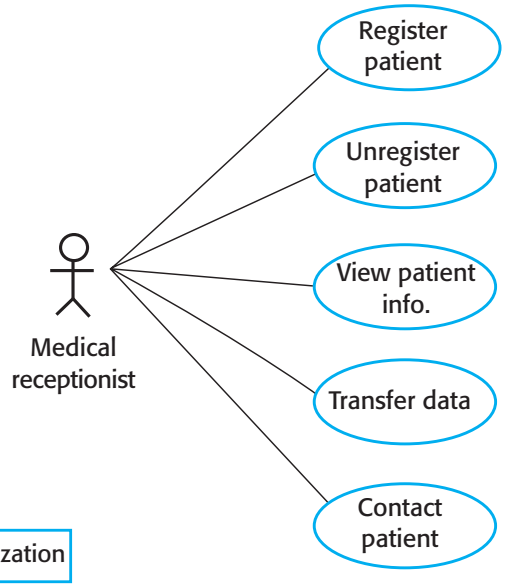


Process Model

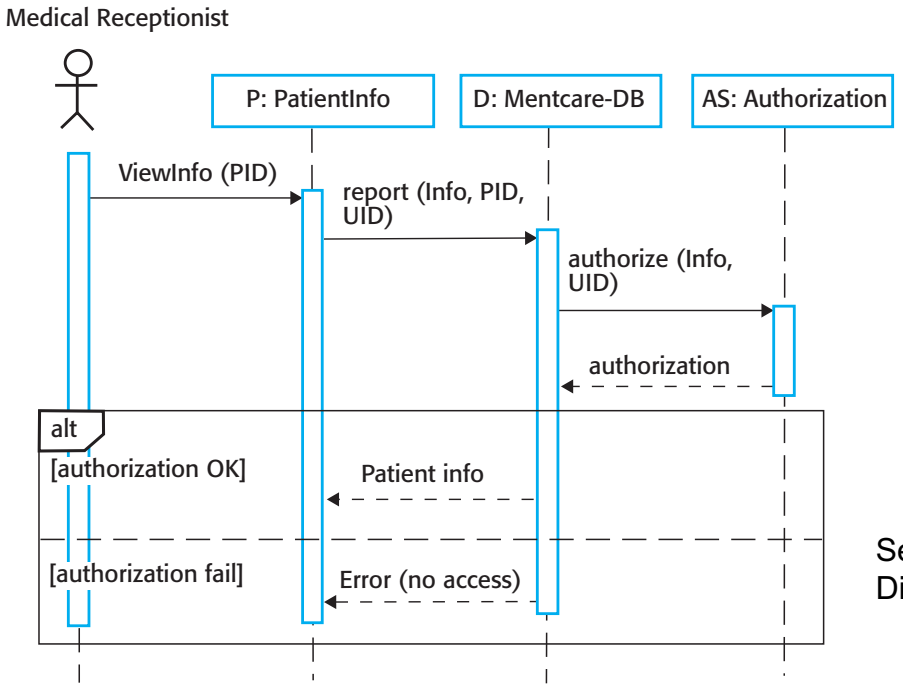
[Summerville, 2016]

Requirements Modelling

- Context models
- **Interaction models**
- Structural models
- Behavioral models



Use Case Model



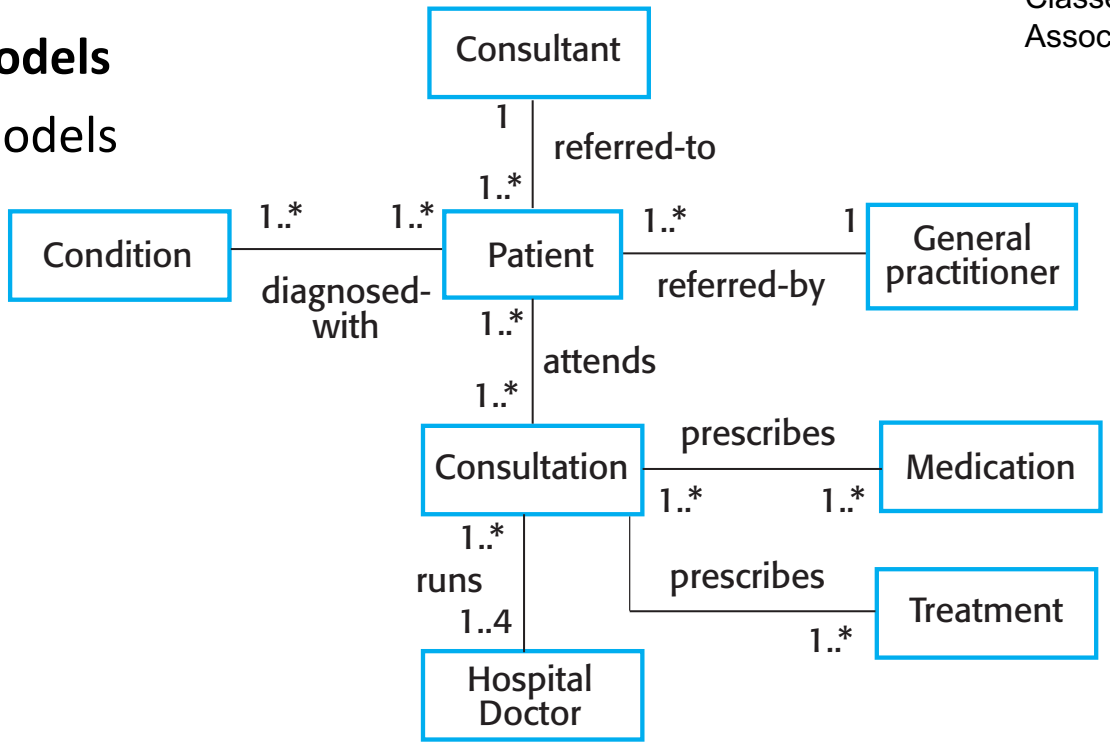
Sequence Diagram

[Summerville, 2016]

Requirements Modelling

- Context models
- Interaction models
- **Structural models**
- Behavioral models

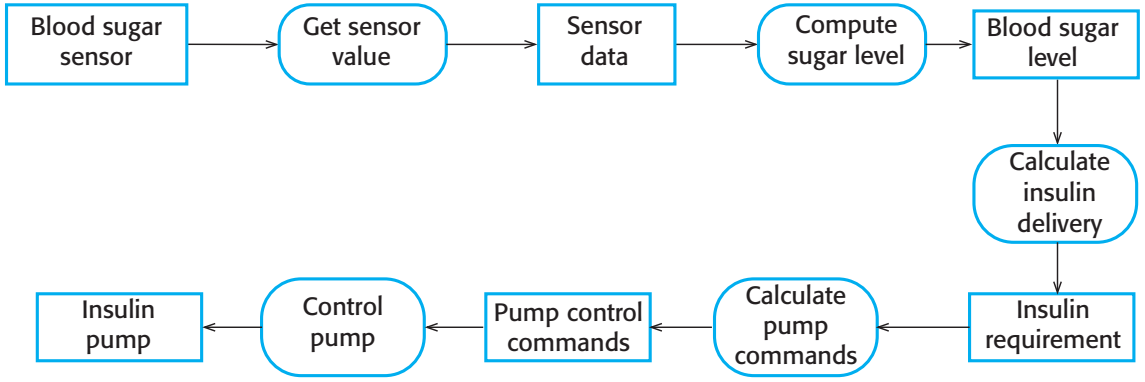
Classes and Associations



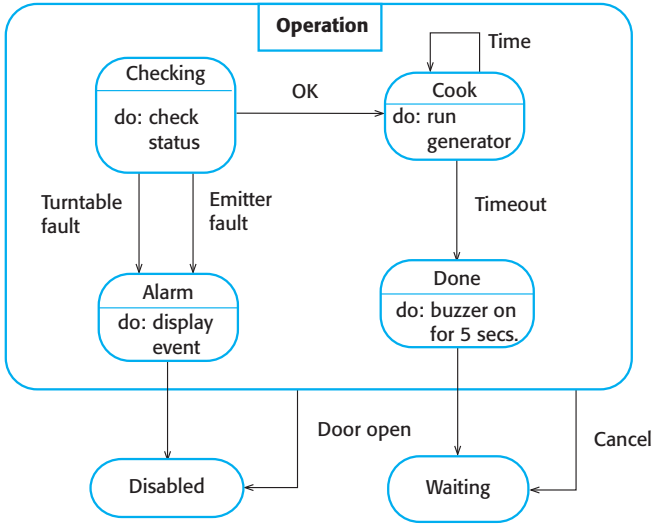
[Summerville, 2016]

Requirements Modelling

- Context models
- Interaction models
- Structural models
- **Behavioral models**
  - **Data-driven**
  - **Event-driven**



Activity Model



State Model

[Summerville, 2016]