REQUIREMENTS ENGINEERING

SOFTWARE ENGINEERING 1

SOONPHEI TIN

OBJECTIVES

- The objective of this chapter is to introduce software requirements and to discuss the processes involved in discovering and documenting these requirements. When you have read the chapter, you will:
- understand the concepts of user and system requirements and why these requirements should be written in different ways;
- understand the differences between functional and nonfunctional software requirements;
- understand how requirements may be organized in a software requirements document;
- understand the principal requirements engineering activities of elicitation, analysis and validation, and the relationships between these activities;

SOFTWARE PROCESS

BACKGROUND

- The requirements for a system are the descriptions of what the system should do
  - reflect the needs of customers for a system
- The process of finding out, analyzing, documenting and checking these needs and constraints is called requirements engineering (RE)
- The requirements can be described in a high-level, abstract statement of a service that a system should provide or a constraint on a system. At the other extreme, it is a detailed, formal definition of a system function

USER REQUIREMENTS AND SYSTEM REQUIREMENTS

- High-level description or detail description?
  - As part of a contract for a large software development project, it must define its needs in a sufficiently abstract way
  - Once a contract has been awarded, the contractor must write a system definition for the client in more detail so that the client understands and can validate what the software will do

USER REQUIREMENTS AND SYSTEM REQUIREMENTS

- Different level of details serve different purpose. User requirements and system requirements
  - User requirements - Statements in a natural language plus diagrams to describe the services and constraint of a system
  - System requirements - more detailed descriptions of the software system's functions, services, and operational constraints

USER REQUIREMENTS AND SYSTEM REQUIREMENTS

User Requirement Definition

1. The MHC-PMS shall generate monthly management reports showing the cost of drugs prescribed by each clinic during that month.

System Requirements Specification

1.1 On the last working day of each month, a summary of the drugs prescribed, their cost, and the prescribing clinics shall be generated.

1.2 The system shall automatically generate the report for printing after 17.30 on the last working day of the month.

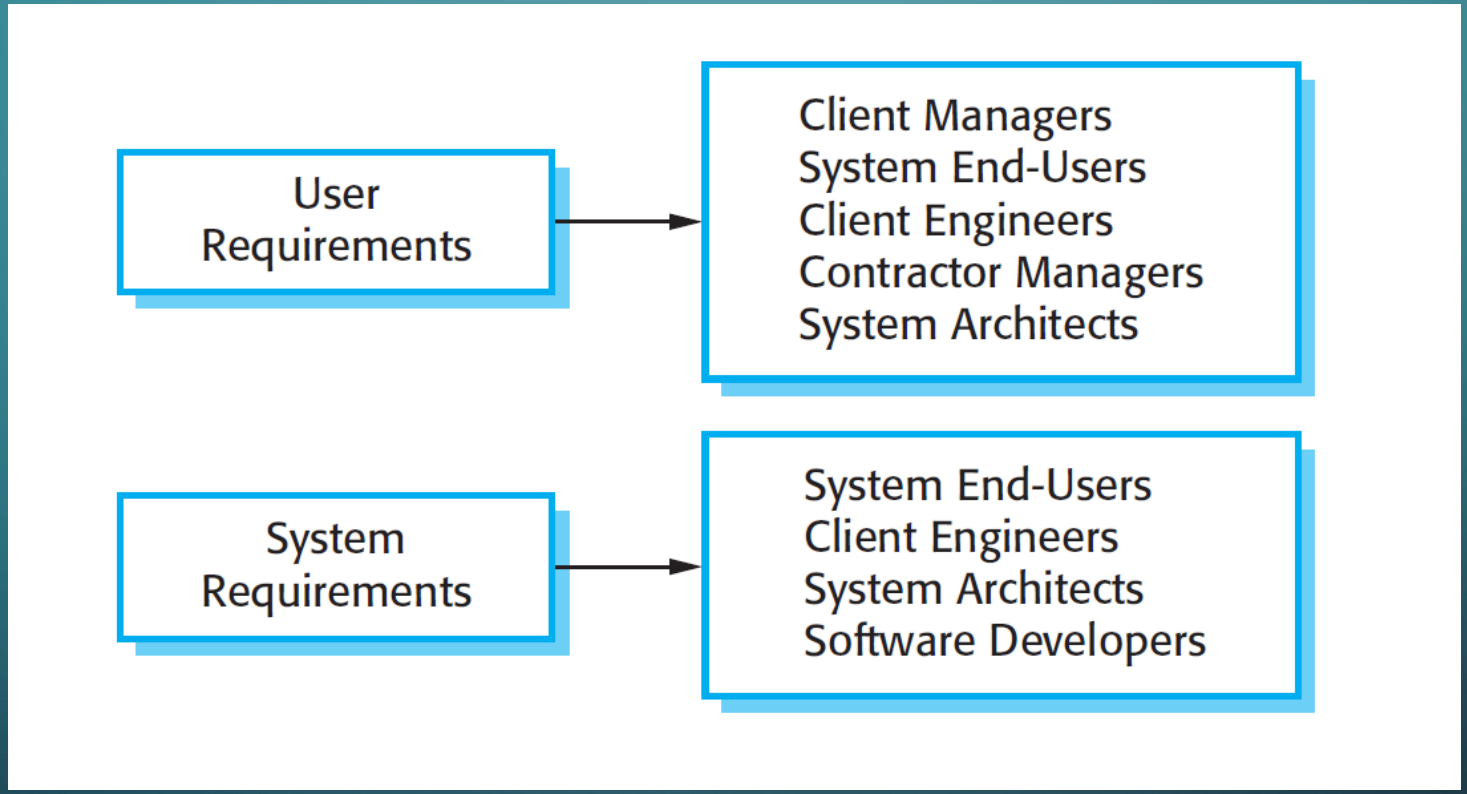
1.3 A report shall be created for each clinic and shall list the individual drug names, the total number of prescriptions, the number of doses prescribed, and the total cost of the prescribed drugs.

1.4 If drugs are available in different dose units (e.g., 10 mg, 20 mg) separate reports shall be created for each dose unit.

1.5 Access to all cost reports shall be restricted to authorized users listed on a management access control list.

Requirements should state ***what*** the system should do and the design should describe ***how*** it does this

USER REQUIREMENTS AND SYSTEM REQUIREMENTS



Why are there different levels of details for different types of roles?

FUNCTIONAL & NON-FUNCTIONAL REQUIREMENTS

- Software system requirements are often classified as functional requirements or nonfunctional requirements:
  - Functional requirements These are statements of services the system should provide. How the services should react and behave in certain condition. In some cases, the functional requirements may also explicitly state what the system should not do.
  - Non-functional requirements These are constraints on the services or functions offered by the system. Non-functional requirements often apply to the system as a whole, rather than individual system features or services.

The distinction between different types of requirement is not as clear-cut as these simple definitions suggest

FUNCTIONAL REQUIREMENTS

- When expressed as *user requirements*, functional requirements are usually described in an abstract way that can be understood by system users.
- More specific *functional system requirements* describe the system functions, its inputs and outputs, exceptions, etc., in detail.

FUNCTIONAL REQUIREMENTS

- Sample requirement for MHC-PMS system:
  1. A user shall be able to search the appointments lists for all clinics.
  2. The system shall generate each day, for each clinic, a list of patients who are expected to attend appointments that day.
  3. Each staff member using the system shall be uniquely identified by his or her eight-digit employee number.

Imprecision in the requirements specification

FUNCTIONAL REQUIREMENTS

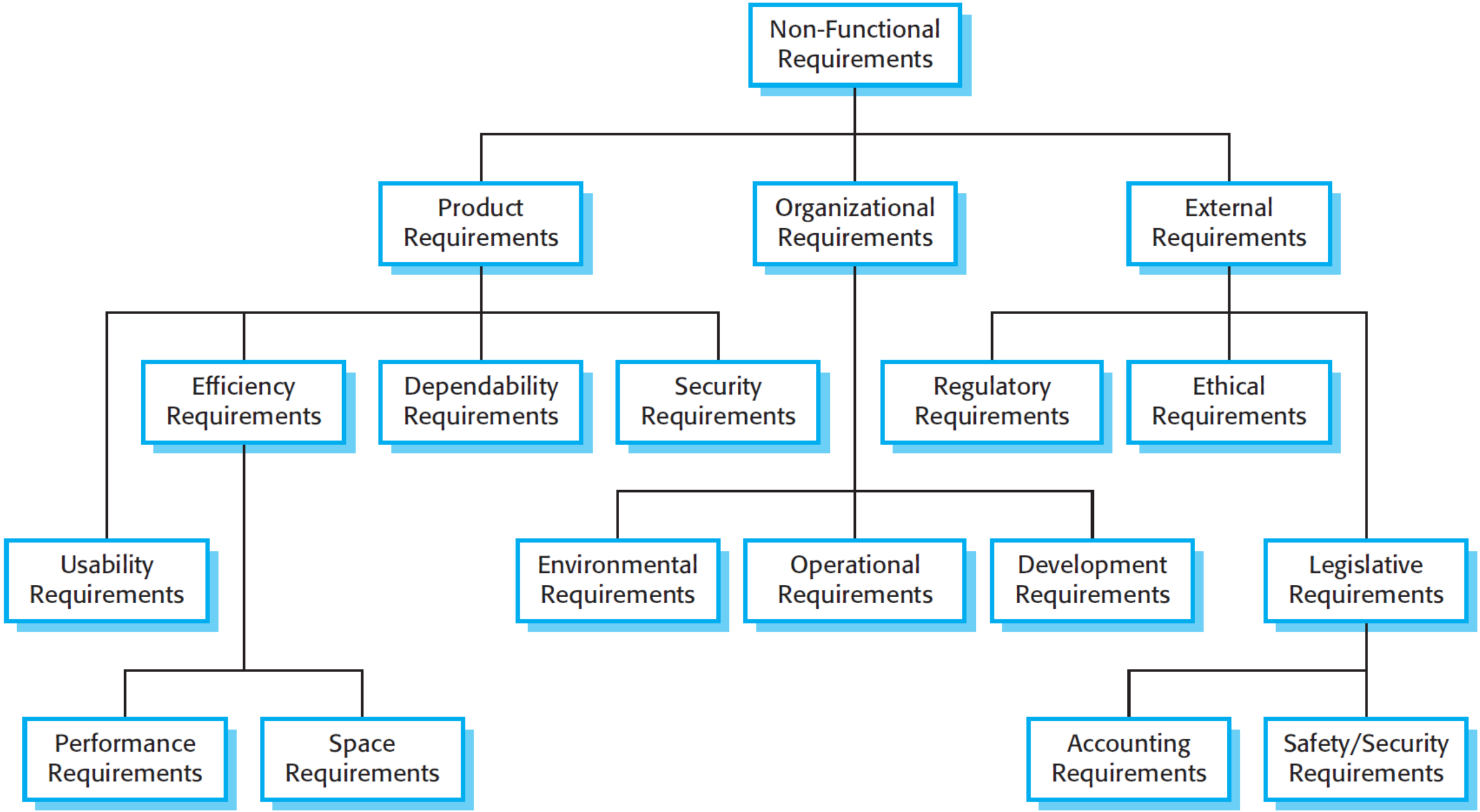
- The functional requirements specification of a system should be both complete and consistent.
  - Completeness means that all services required by the user should be defined.
  - Consistency means that requirements should not have contradictory definitions.
- In practice, for large, complex systems, it is practically impossible to achieve requirements consistency and completeness.
  - it is easy to make mistakes and omissions when writing specifications for complex systems
  - there are many stakeholders in a large system. Stakeholders have different and often inconsistent needs.

NON-FUNCTIONAL REQUIREMENTS

- Non-functional requirements, as the name suggests, are requirements that are not directly concerned with the specific services delivered by the system to its users.
- System properties such as reliability, response time, and store occupancy.
- Constraints on the system implementation such as performance, security, or availability.
- Usually specify constrain characteristics of the system as a whole
- Often more critical than individual functional requirements

NON-FUNCTIONAL REQUIREMENTS

- Often more critical than individual functional requirements
- failing to meet a non-functional requirement can mean that the whole system is unusable
- The implementation of non-functional requirements may be intricately dispersed throughout the system.
  - They may affect the overall architecture of a system rather than the individual components.
  - A single non-functional requirement may generate a number of related functional requirements.


Non-Functional Requirements
Product Requirements
Organizational Requirements
External Requirements
Efficiency Requirements
Dependability Requirements
Security Requirements
Regulatory Requirements
Ethical Requirements
Usability Requirements
Environmental Requirements
Operational Requirements
Development Requirements
Legislative Requirements
Performance Requirements
Space Requirements
Accounting Requirements
Safety/Security Requirements

CLASSIFICATION OF NON-FUNCTIONAL REQUIREMENTS

- *Product requirements* - These requirements specify or constrain the behavior of the software.
- *Organizational requirements* - These requirements are broad system requirements derived from policies and procedures in the customer's and developer's organization.
- *External requirements* - This broad heading covers all requirements that are derived from factors external to the system and its development process

CLASSIFICATION OF NON-FUNCTIONAL REQUIREMENTS

PRODUCT REQUIREMENT
The MHC-PMS shall be available to all clinics during normal working hours (Mon–Fri, 08.30–17.30). Downtime within normal working hours shall not exceed five seconds in any one day.

ORGANIZATIONAL REQUIREMENT
Users of the MHC-PMS system shall authenticate themselves using their health authority identity card.

EXTERNAL REQUIREMENT
The system shall implement patient privacy provisions as set out in HStan-03-2006-priv.

NON-FUNCTIONAL REQUIREMENTS - TESTABILITY

- A common problem with non-functional requirements is that users or customers often propose these requirements as general goals, such as …

The system should be easy to use by medical staff and should be organized in such a way that user errors are minimized.

VS

Medical staff shall be able to use all the system functions after four hours of training. After this training, the average number of errors made by experienced users shall not exceed two per hour of system use.

NON-FUNCTIONAL REQUIREMENTS - TESTABILITY

Property	Measure
Speed	Processed transactions/second User/event response time Screen refresh time
Size	Mbytes Number of ROM chips
Ease of use	Training time Number of help frames
Reliability	Mean time to failure Probability of unavailability Rate of failure occurrence Availability
Robustness	Time to restart after failure Percentage of events causing failure Probability of data corruption on failure
Portability	Percentage of target dependent statements Number of target systems

NON-FUNCTIONAL REQUIREMENTS

- Non-functional requirements often conflict and interact with other functional or non-functional requirements
- It is difficult to separate functional and non-functional requirements in the requirements document.
  - you should explicitly highlight requirements that are clearly related to emergent system properties, such as performance or reliability

THE SOFTWARE REQUIREMENTS DOCUMENT

- The software requirements document (sometimes called the software requirements specification or SRS) is an official statement of what the system developers should implement.
- It should include both the user requirements for a system and a detailed specification of the system requirements

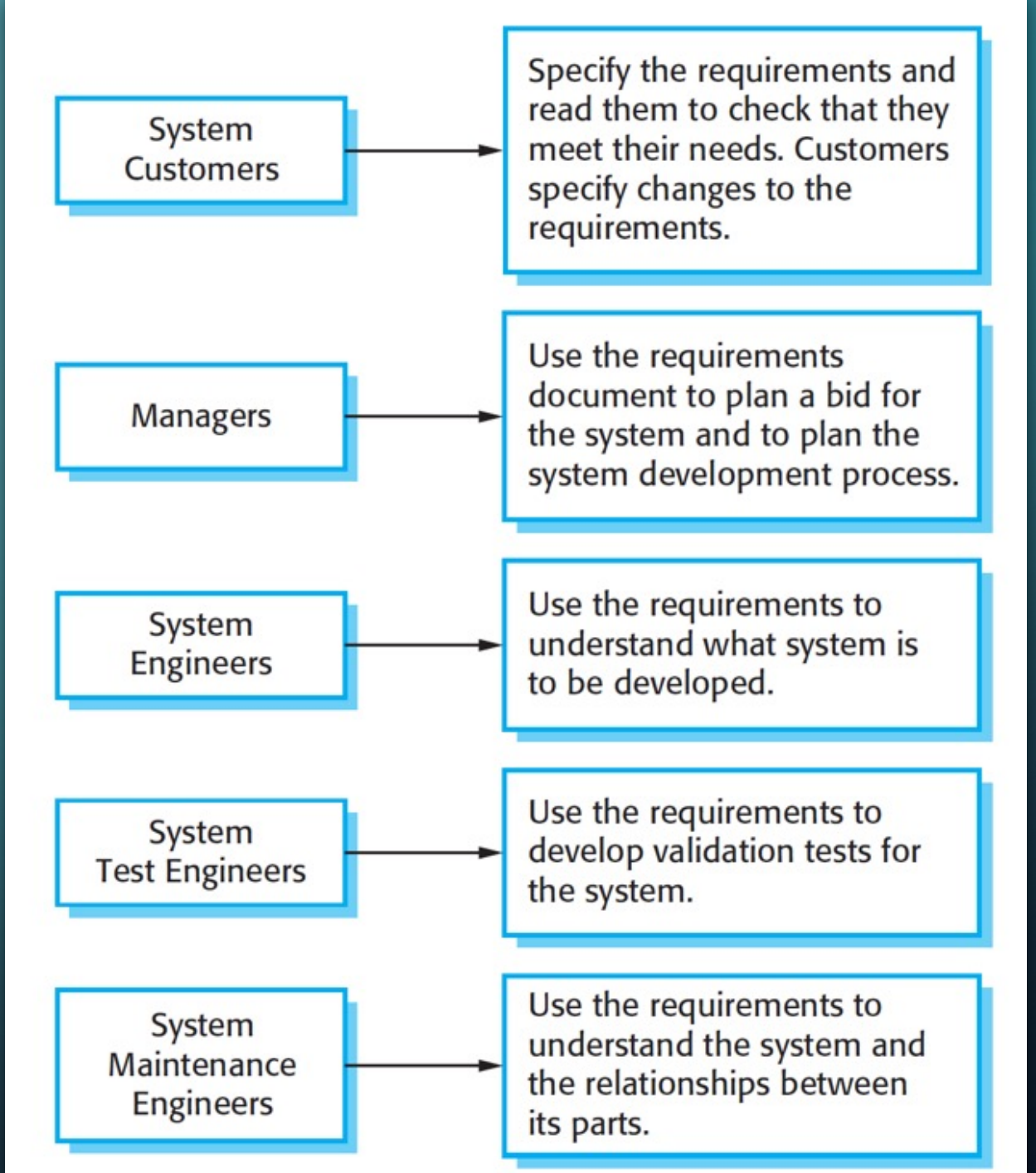
http://www.SoftwareEngineering-9.com/Web/Requirements/IEEE-standard.html

Table of Contents

THE SOFTWARE REQUIREMENTS DOCUMENT

- Requirements documents are essential when an outside contractor is developing the software system
- Agile development methods argue that requirements change so rapidly that a requirements document is out of date as soon as it is written.
  - Suitable for business systems where requirements are unstable;
  - it is still useful to write a short supporting document that defines the business and dependability requirements for the system;

THE SOFTWARE REQUIREMENTS DOCUMENT



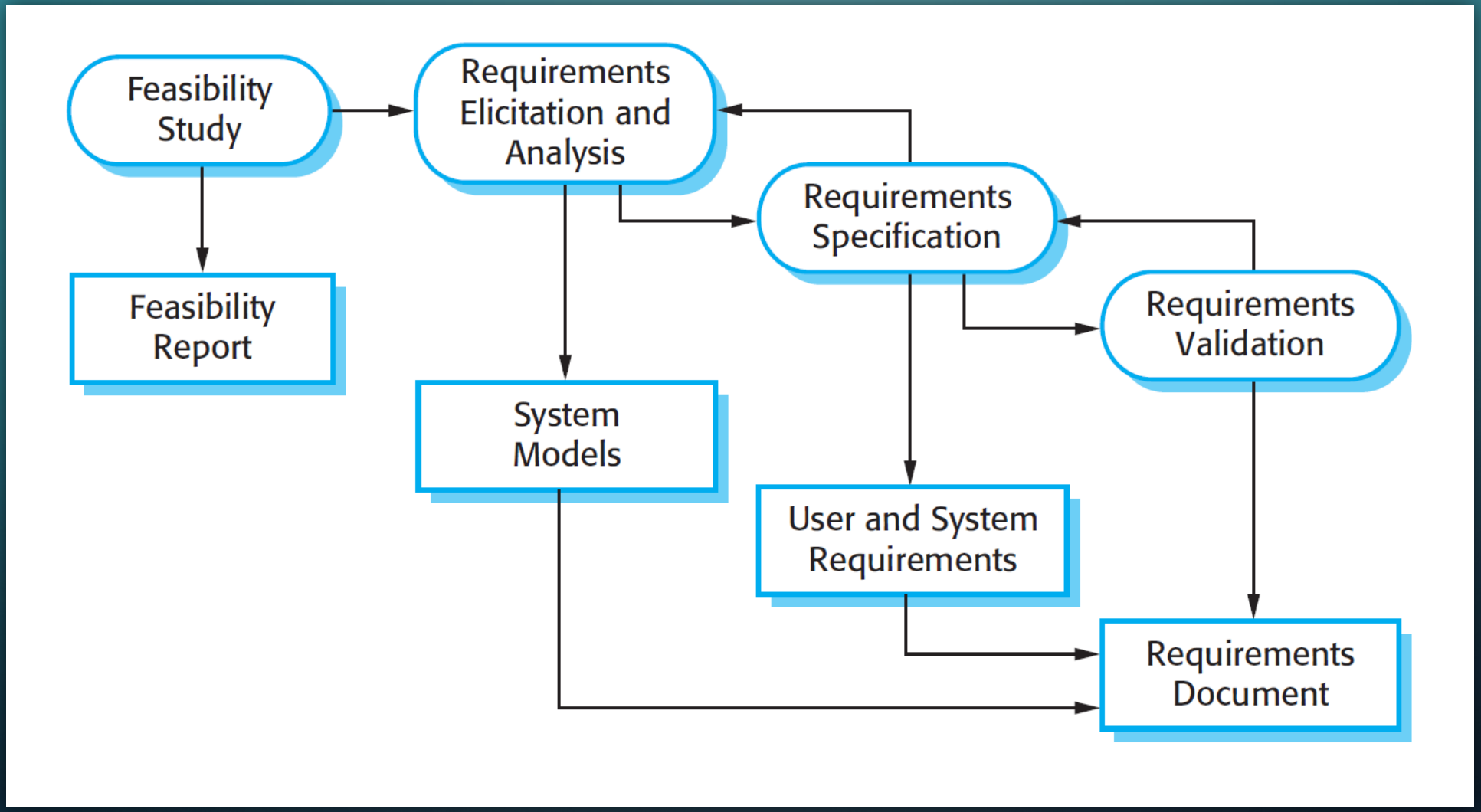
THE SOFTWARE REQUIREMENTS DOCUMENT
SRS QUALITY CHARACTERISTIC

- • Correct
  - Complete
  - Unambiguous
  - Verifiable
  - Consistent
  - Ranked for importance and/or stability
  - Modifiable
  - Traceable

THE SRS – LEVEL OF DETAILS

- The level of detail that you should include in a requirements document depends on the type of system that is being developed and the development process used
- Detailed requirements
  - Critical systems
  - system is to be developed by a separate company
- Less detailed requirements
  - inhouse
  - iterative development process

REQUIREMENTS ENGINEERING PROCESSES



REQUIREMENTS ENGINEERING PROCESSES

- include four high-level activities
  - assessing if the system is useful to the business (feasibility study)
  - discovering requirements (elicitation and analysis)
  - converting these requirements into some standard form (specification)
  - checking that the requirements actually define the system that the customer wants (validation).

REQUIREMENTS ENGINEERING PROCESSES

iterative process in which the activities are interleaved.

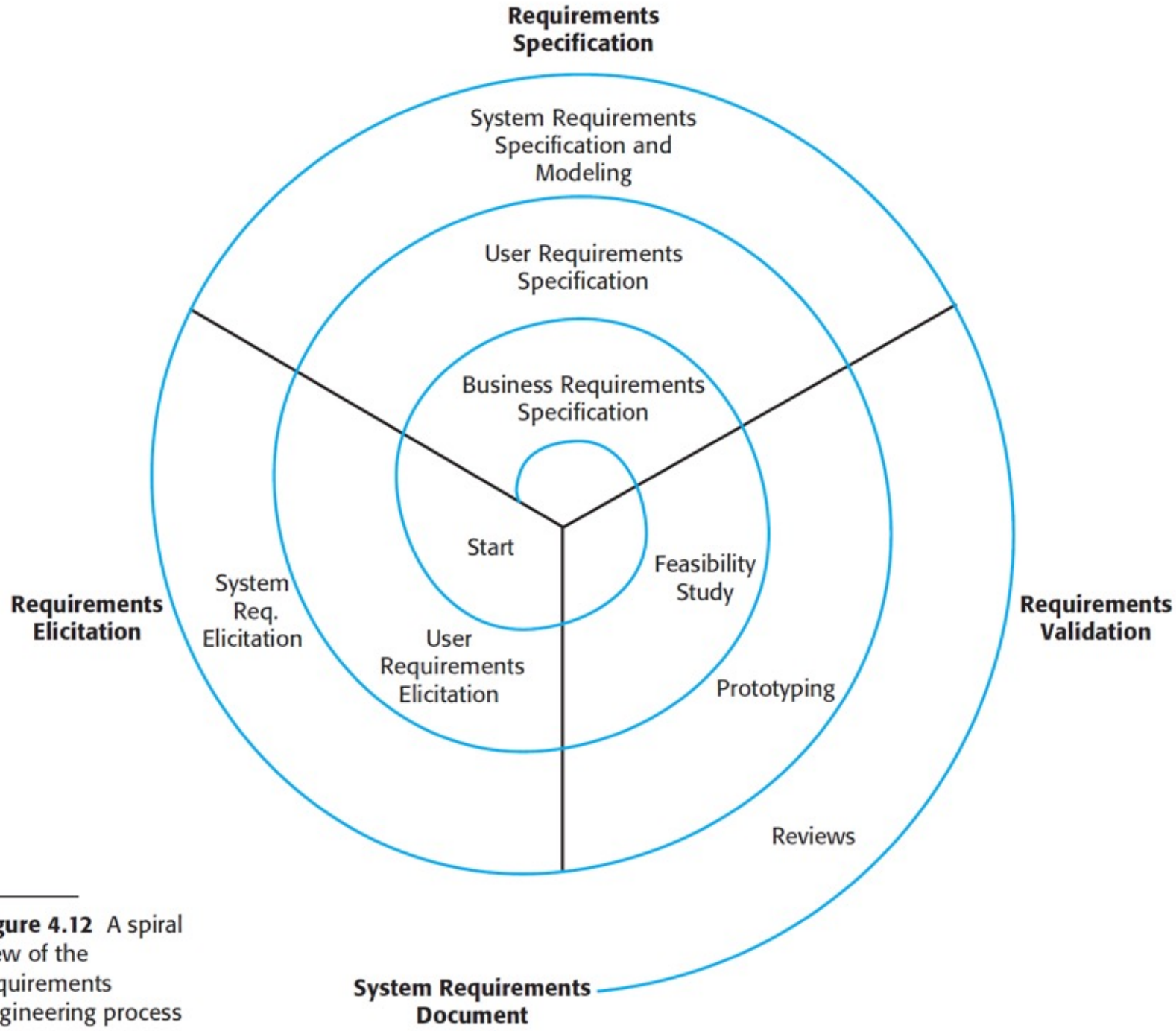


Figure 4.12 A spiral view of the requirements engineering process

REQUIREMENTS ELICITATION AND ANALYSIS

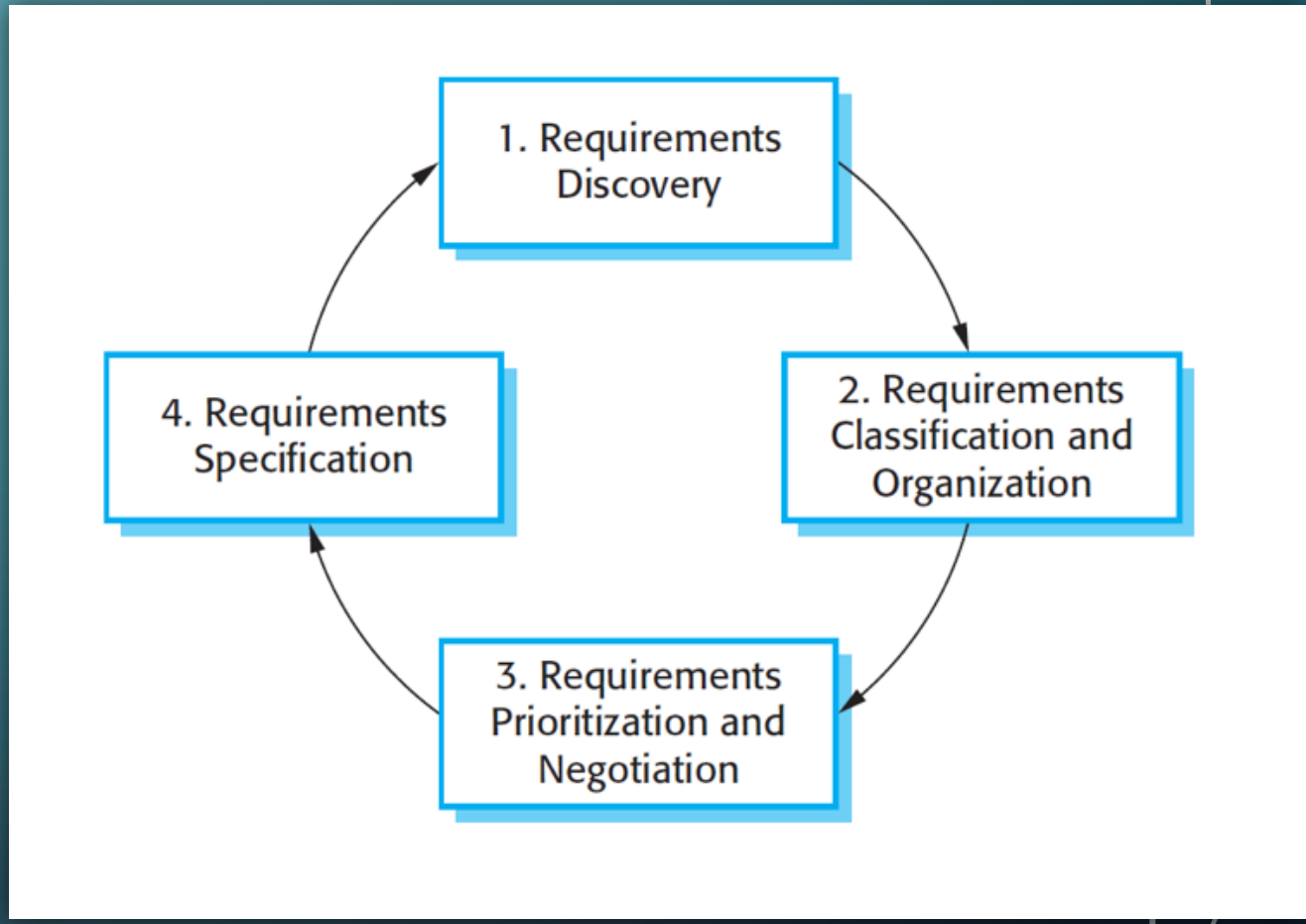
- software engineers work with customers and system end-users to find out about
  - the application domain
  - what services the system should provide
  - the required performance of the system
  - hardware constraints, and so on.

REQUIREMENTS ELICITATION AND ANALYSIS

- may involve a variety of different kinds of people in an organization include: -
  - end users who will interact with the system
  - anyone else in an organization who will be affected by it
  - engineers who are developing or maintaining other related systems
  - business managers
  - domain experts
  - trade union representatives.

REQUIREMENTS ELICITATION AND ANALYSIS

- Each organization will have its own version or instantiation of this general model depending on local factors such as the expertise of the staff, the type of system being developed, the standards used, etc.
- Requirements elicitation and analysis is an iterative process with continual feedback from each activity to other activities



REQUIREMENTS ELICITATION AND ANALYSIS

- Requirements discovery - This is the process of interacting with stakeholders of the system to discover their requirements
- Requirements classification and organization - This activity takes the unstructured collection of requirements, groups related requirements, and organizes them into coherent clusters.
  - The most common way of grouping requirements is to use a model of the system architecture to identify sub-systems and to associate requirements with each sub-system

REQUIREMENTS ELICITATION AND ANALYSIS

- Requirements prioritization and negotiation - When multiple stakeholders are involved, requirements will conflict. This activity is concerned with prioritizing requirements and finding and resolving requirements conflicts through negotiation.
- Requirements specification - The requirements are documented and input into the next round of the spiral.

REQUIREMENTS ELICITATION AND ANALYSIS CHALLENGES

- Eliciting and understanding requirements from system stakeholders is a difficult process for several reasons:
  - Stakeholders often don't know what they want from a computer system except in the most general terms; they may find it difficult to articulate the requirements; they may make unrealistic demands because they don't know what is and isn't feasible.
  - Stakeholders in a system express requirements in their own terms and with implicit knowledge of their own work.

REQUIREMENTS ELICITATION AND ANALYSIS CHALLENGES

- Eliciting and understanding requirements from system stakeholders is a difficult process for several reasons:
  - Different stakeholders have different requirements and priorities, and they may express these in different ways, some of these requirements have commonalities and conflict.
  - Political factors may influence the requirements of a system.
  - The economic and business environment in which the analysis takes place is dynamic. It changes during the analysis process.

REQUIREMENTS ELICITATION AND ANALYSIS
REQUIREMENTS DISCOVERY

- Requirements discovery (sometime called requirements elicitation) is the process of gathering information about the required system and existing systems, and distilling the user and system requirements from this information.
- Sources of information during the requirements discovery phase include documentation, system stakeholders, and specifications of similar systems.
  - interviews
  - observation
  - scenarios
  - prototypes

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY

- Stakeholders range from end-users of a system through managers to external stakeholders such as regulators
- Stakeholders for the mental healthcare patient information system include:
  - Patients
  - Doctors
  - Nurses
  - Medical receptionists
  - IT staff
  - Medical ethics manager
  - Healthcare managers
  - Medical records staff

requirements may also come from the application domain and from other systems that interact with the system

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - INTERVIEW

- Formal or informal interviews with system stakeholders
- The requirements engineering team puts questions to stakeholders about the system that they currently use and the system to be developed.
- Requirements are derived from the answers to these questions.
- Interviews may be of two types:
  - Closed interviews - where the stakeholder answers a pre-defined set of questions.
  - Open interviews - no pre-defined agenda. The requirements engineering team explores a range of issues with system stakeholders

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - INTERVIEW

- Interviews are good for: -
  - getting an overall understanding of what stakeholders do
  - how they might interact with the new system
  - the difficulties that they face with current systems

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - INTERVIEW

- It can be difficult to elicit domain knowledge through interviews for two reasons:
  - All application specialists use terminology and jargon that are specific to a domain. They normally use terminology in a precise and subtle way that is easy for requirements engineers to misunderstand.
  - Some domain knowledge is so familiar to stakeholders that they either find it difficult to explain or they think it is so fundamental that it isn't worth mentioning

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - INTERVIEW

- Not an effective technique for eliciting knowledge about organizational requirements and constraints because there are subtle power relationships between the different people in the organization.
  - most people are generally reluctant to discuss political and organizational issues that may affect the requirements

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - INTERVIEW

- Effective interviewers have two characteristics:
- They are open-minded, avoid pre-conceived ideas about the requirements, and are willing to listen to stakeholders.
- They prompt the interviewee to get discussions going using a springboard question, a requirements proposal, or by working together on a prototype system.

Information from interviews supplements other information about the system from documentation describing business processes or existing systems, user observations, etc.

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - SCENARIOS

- System user can understand and criticize a scenario of how they might interact with a software system.
- Use the information gained from the discussion based on a scenario to formulate the actual system requirements.
- A scenario is the descriptions of example interaction sessions. Each scenario usually covers one or a small number of possible interactions
- A scenario starts with an outline of the interaction. During the elicitation process, details are added to this to create a complete description of that interaction.

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – SCENARIOS

COLLECTING MEDICAL HISTORY

INITIAL ASSUMPTION:
The patient has seen a medical receptionist who has created a record in the system and collected the patient's personal information (name, address, age, etc.). A nurse is logged on to the system and is collecting medical history.

NORMAL:
The nurse searches for the patient by family name. If there is more than one patient with the same surname, the given name (first name in English) and date of birth are used to identify the patient.

The nurse chooses the menu option to add medical history.

The nurse then follows a series of prompts from the system to enter information about consultations elsewhere on mental health problems (free text input), existing medical conditions (nurse selects conditions from menu), medication currently taken (selected from menu), allergies (free text), and home life (form).

WHAT CAN GO WRONG:
The patient's record does not exist or cannot be found. The nurse should create a new record and record personal information.

Patient conditions or medication are not entered in the menu. The nurse should choose the 'other' option and enter free text describing the condition/medication.

Patient cannot/will not provide information on medical history. The nurse should enter free text recording the patient's inability/unwillingness to provide information. The system should print the standard exclusion form stating that the lack of information may mean that treatment will be limited or delayed. This should be signed and handed to the patient.

OTHER ACTIVITIES:
Record may be consulted but not edited by other staff while information is being entered.

SYSTEM STATE ON COMPLETION:
User is logged on. The patient record including medical history is entered in the database, a record is added to the system log showing the start and end time of the session and the nurse involved.

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY - SCENARIOS

- Scenario-based elicitation involves working with stakeholders to identify scenarios and to capture details to be included in these scenarios.
- Scenarios may be written as text, supplemented by diagrams, screen shots, etc.
- Alternatively, a more structured approach such as event scenarios or use cases may be used.

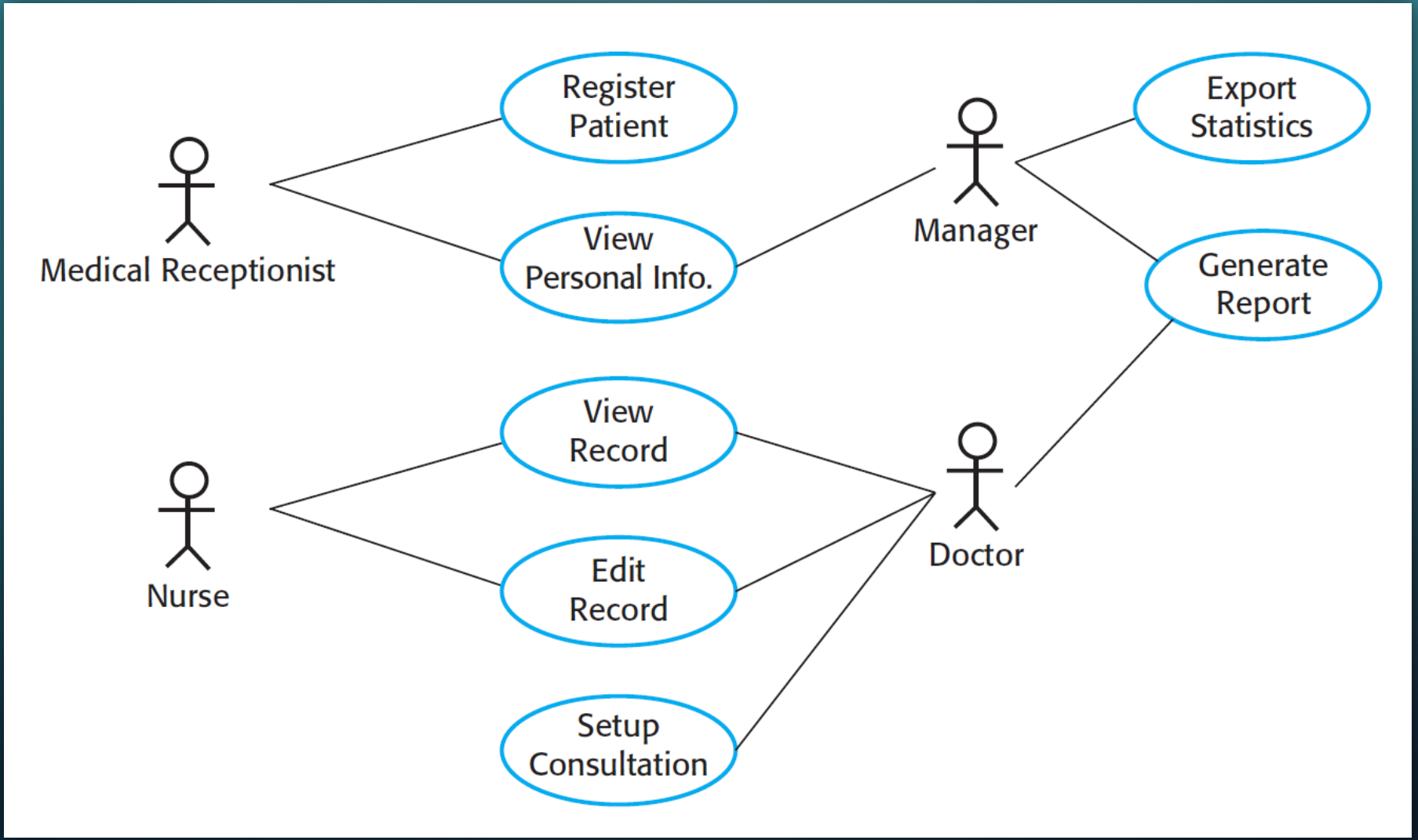
REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – USE CASES

- Use cases are a requirements discovery technique. In their simplest form, a use case identifies the actors involved in an interaction and names the type of interaction
- It is supplemented by additional information describing the interaction with the system.
  - Textual description
  - one or more graphical models
- Use cases are documented using a high-level use case diagram. The set of use cases represents all of the possible interactions that will be described in the system requirements.
  - each use case is a single scenario;
  - encapsulate a set of scenarios in a single use case.

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – USE CASES



REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – USE CASES

- Use cases identify the individual interactions between the system and its users or other systems.
- Each use case should be documented with a textual description.
- A use case can then be linked to other models in the UML that will develop the scenario in more detail

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – USE CASES

Setup Consultation Use Case

Setup consultation allows two or more doctors, working in different offices, to view the same record at the same time. One doctor initiates the consultation by choosing the people involved from a drop-down menu of doctors who are online. The patient record is then displayed on their screens but only the initiating doctor can edit the record. In addition, a text chat window is created to help coordinate actions. It is assumed that a phone conference for voice communication will be separately set up.

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – USE CASES

- Use case focus on interactions with the system, they are not as effective for eliciting: -
  - constraints
  - high-level business
  - nonfunctional requirements
  - domain requirements.

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – ETHNOGRAPHY

- Systems are used in a social and organizational context and software system requirements may be derived or constrained by that context
- Satisfying these social and organizational requirements is often critical for the success of the system.
- Ethnography is an observational technique that can be used to understand operational processes and help derive support requirements for these processes.
- An analyst immerses himself or herself in the working environment to observe the actual tasks in which participants are involved.
- Helps discover implicit system requirements that reflect the actual ways that people work

REQUIREMENTS ELICITATION AND ANALYSIS

REQUIREMENTS DISCOVERY – ETHNOGRAPHY

- Ethnography is particularly effective for discovering two types of requirements: -
  - Requirements that are derived from the way in which people actually work, rather than the way in which process definitions say they ought to work
  - Requirements that are derived from cooperation and awareness of other people's activities.
- Ethnography focuses on the end-user, this approach: -
  - is not always appropriate for discovering organizational or domain requirements
  - cannot always identify new features that should be added to a system.

REQUIREMENTS SPECIFICATION

- The process of writing down the user and system requirements in a requirements document.
- The user and system requirements should be clear, unambiguous, easy to understand, complete, and consistent.
- Stakeholders interpret the requirements in different ways and there are often inherent conflicts and inconsistencies in the requirements.
- The user requirements for a system should describe the functional and nonfunctional requirements in a way they are understandable by system users who don't have detailed technical knowledge

REQUIREMENTS SPECIFICATION

- Specify only the external behavior of the system.
- Should not include details of the system architecture or design
- Write user requirements in natural language, with simple tables, forms, and intuitive diagrams.
- System requirements are expanded versions of the user requirements
- They add detail and explain how the user requirements should be provided by the system

REQUIREMENTS SPECIFICATION

- They may be used as part of the contract for the implementation of the system
- describe the external behavior of the system and its operational constraints. They should not be concerned with how the system should be designed or implemented

REQUIREMENTS SPECIFICATION

- it is practically impossible to exclude all design information. There are several reasons for this:
  - You may have to design an initial architecture of the system to help structure the requirements specification. The system requirements are organized according to the different sub-systems that make up the system.
  - Systems may interoperate with existing systems, which constrain the design and impose requirements on the new system.
  - The use of a specific architecture to satisfy non-functional requirements may be necessary.

REQUIREMENTS SPECIFICATION

- System requirements may be written in natural language but other notations based on forms, graphical system models, or mathematical system models can also be used.
- Graphical models are most useful when you need to show how a state changes or when you need to describe a sequence of actions.

REQUIREMENTS SPECIFICATION
NATURAL LANGUAGE SPECIFICATION

- It is expressive, intuitive, and universal.
- It is also potentially vague, ambiguous, and its meaning depends on the background of the reader

REQUIREMENTS SPECIFICATION
NATURAL LANGUAGE SPECIFICATION

- To minimize misunderstandings when writing natural language requirements:
  - Invent a standard format and ensure that all requirement definitions adhere to that format. Standardizing the format makes omissions less likely and requirements easier to check
  - Use language consistently to distinguish between mandatory and desirable requirements
  - Use text highlighting to pick out key parts of the requirement.
  - Do not assume that readers understand technical software engineering language.
  - Associate a rationale with each user requirement. The rationale should explain why the requirement has been included

REQUIREMENTS SPECIFICATION
STRUCTURED SPECIFICATION

- Structured natural language is a way of writing system requirements in a predefined structure or template using natural language
- user requirements can be written on cards, one requirement per card.
  - a number of fields on each card, such as:
  - requirements rationale
  - dependencies on other requirements
  - the source of the requirements
  - supporting materials, and so on

REQUIREMENTS SPECIFICATION
STRUCTURED SPECIFICATION - SAMPLE

Insulin Pump/Control Software/SRS/3.3.2

Function	Compute insulin dose: Safe sugar level.
Description	Computes the dose of insulin to be delivered when the current measured sugar level is in the safe zone between 3 and 7 units.
Inputs	Current sugar reading (r2), the previous two readings (r0 and r1).
Source	Current sugar reading from sensor. Other readings from memory.
Outputs	CompDose—the dose in insulin to be delivered.
Destination	Main control loop.
Action	CompDose is zero if the sugar level is stable or falling or if the level is increasing but the rate of increase is decreasing. If the level is increasing and the rate of increase is increasing, then CompDose is computed by dividing the difference between the current sugar level and the previous level by 4 and rounding the result. If the result, is rounded to zero then CompDose is set to the minimum dose that can be delivered.
Requirements	Two previous readings so that the rate of change of sugar level can be computed.
Pre-condition	The insulin reservoir contains at least the maximum allowed single dose of insulin.
Post-condition	r0 is replaced by r1 then r1 is replaced by r2.
Side effects	None.

REQUIREMENTS SPECIFICATION
STRUCTURED SPECIFICATION

- Overcome the limitation of natural language using tables or graphical models to show computations, system state changes, system interaction, execution sequences
- Tables are particularly useful when there are a number of possible alternative situations

Condition	Action
Sugar level falling (r2 < r1)	CompDose = 0
Sugar level stable (r2 = r1)	CompDose = 0
Sugar level increasing and rate of increase decreasing ((r2 − r1) < (r1 − r0))	CompDose = 0
Sugar level increasing and rate of increase stable or increasing ((r2 − r1) ≥ (r1 − r0))	CompDose = round ((r2 − r1)/4) If rounded result = 0 then CompDose = MinimumDose

REQUIREMENTS VALIDATION

- Requirements validation is the process of checking that requirements actually define the system that the customer really wants
- It is important because errors in a requirements document can lead to extensive rework costs when these problems are discovered during development or after the system is in service.

REQUIREMENTS VALIDATION

- During the requirements validation process, different types of checks should be carried out on the requirements in the requirements document. These checks include:
  - *Validity checks* - The functions proposed by stakeholders should be aligned with what the system needs to perform. You may find later that there are additional or different functions are required instead.
  - *Consistency checks* - Requirements in the document should not conflict. That is, there should not be contradictory constraints or different descriptions of the same system function.

REQUIREMENTS VALIDATION

- During the requirements validation process, different types of checks should be carried out on the requirements in the requirements document. These checks include:
  - *Completeness checks* - The requirements document should include requirements that define all functions and the constraints intended by the system user.
  - *Realism checks* - Using knowledge of existing technology, the requirements should be checked to ensure that they can actually be implemented.
  - *Verifiability* - To reduce the potential for dispute between customer and contractor, system requirements should always be written so that they are verifiable.

REQUIREMENTS VALIDATION

- There are a number of requirements validation techniques that can be used individually or in conjunction with one another:
  - *Requirements reviews* - The requirements are analyzed systematically by a team of reviewers who check for errors and inconsistencies.
  - *Prototyping* - In this approach to validation, an executable model of the system in question is demonstrated to end-users and customers
  - *Test-case generation* - Requirements should be testable. If the tests for the requirements are devised as part of the validation process, this often reveals requirements problems.