Software Engineering

Lecture 7

Mr. Noor Ul Arfeen

Motivation for Requirement Engineering

In September 1999, ***NASA lost its $125-million Mars Climate Orbiter*** when it tried to enter the orbit, just 100 kilometers too close to Mars. The mission failed due to poor requirements management: it was not discussed earlier in the stage whether the ‘navigation software’ required metric units or imperial units.

Software Requirements

A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document

The software requirements are description of features and functionalities of target system

Requirements convey the expectations of users from the software product

The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view

Requirements Engineering

What is it?

Before you begin any technical work, it's a good idea to create a set of requirements for any engineering tasks. These tasks lead to an understanding of what the business impact of the software will be, what the customer wants, and how end users will interact with the software.

Who does it?

Software engineers (sometimes referred to as system engineers or "analysts" in the IT world) and other project stakeholders (managers, customers, and end users) all participate in requirements engineering.

Requirements Engineering (Conti.)

Why is it important?

Designing and building an elegant computer program that solves the wrong problem serves no one's needs. That's why it's important to understand what the customer wants before you begin to design and build a computer-based system.

What are the steps?

Requirements engineering begins with inception (a task that defines the scope and nature of the problem to be solved). It moves onward to elicitation (a task that helps stakeholders define what is required), and then elaboration (where basic requirements are refined and modified). As stakeholders define the problem, negotiation occurs (what are the priorities, what is essential, when is it required?) Finally, the problem is specified in some manner and then reviewed or validated to ensure that your understanding of the problem and the stakeholders' understanding of the problem coincide.

Requirements Engineering (Conti.)

What is the work product?

The intent of requirements engineering is to provide all parties with a written understanding of the problem. This can be achieved though a number of work products: usage scenarios, functions and features lists, requirements models, or a specification.

How do I ensure that I've done it right?

Requirements engineering work products are reviewed with stakeholders to ensure that what you have learned is what they really meant. A word of warning: Even after all parties agree, things will change, and they will continue to change throughout the project.

Requirements Engineering (Conti.)

The system requirements are the descriptions of the system services and constraints that are generated during the requirements engineering process

The process of finding out, analyzing, documenting and checking these services and constraints is called requirements engineering (RE)

Requirements engineering encompasses seven distinct tasks: inception, elicitation, elaboration, negotiation, specification, validation, and management. It is important to note that some of these tasks occur in parallel and all are adapted to the needs of the project.

Requirements Engineering (Conti.)

1. Inception.

How does a software project get started? Is there a single event that becomes the catalyst for a new computer-based system or product, or does the need evolve over time? There are no definitive answers to these questions. In some cases, a casual conversation is all that is needed to precipitate a major software engineering effort. But in general, most projects begin when a business need is identified or a potential new market or service is discovered. Stakeholders from the business community (e.g., business managers, marketing people, product managers) define a business case for the idea, try to identify the breadth and depth of the market, do a rough feasibility analysis, and identify a working description of the project's scope.

Requirements Engineering (Conti.)

2. Elicitation.

It certainly seems simple enough—ask the customer, the users, and others what the objectives for the system or product are, what is to be accomplished, how the system or product fits into the needs of the business, and finally, how the system or product is to be used on a day-to-day basis. But it isn't simple— it's very hard.

3. Elaboration

Elaboration is driven by the creation and refinement of user scenarios that describe how the end user (and other actors) will interact with the system. Each user scenario is parsed to extract analysis classes—business domain entities that are visible to the end user.

Requirements Engineering (Conti.)

4. Negotiation.

It isn't unusual for customers and users to ask for more than can be achieved, given limited business resources. It's also relatively common for different customers or users to propose conflicting requirements, arguing that their version is "essential for our special needs."

5. Specification.

In the context of computer-based systems (and software), the term specification means different things to different people. A specification can be a written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype, or any combination of these.

Requirements Engineering (Conti.)

6. Validation.

The work products produced as a consequence of requirements engineering are assessed for quality during a validation step. Requirements validation examines the specification5 to ensure that all software requirements have been stated unambiguously; that inconsistencies, omissions, and errors have been detected and corrected; and that the work products conform to the standards established for the process, the project, and the product.

7. Requirements management.

Requirements for computer-based systems change, and the desire to change requirements persists throughout the life of the system. Requirements management is a set of activities that help the project team identify, control, and track requirements and changes to requirements at any time as the project proceeds.

What is a Requirement?

It may range from a high-level abstract statement of a service or of a system constraint to a detailed mathematical functional specification

This is inevitable as requirements may serve a dual function

- May be the basis for a bid for a contract - therefore must be open to interpretation
- May be the basis for the contract itself - therefore must be defined in detail
- Both these statements may be called requirements

Requirements Abstraction

“If a company wishes to let a contract for a large software development project, it must define its needs in a sufficiently abstract way that a solution is not pre-defined. The requirements must be written so that several contractors can bid for the contract, offering, perhaps, different ways of meeting the client organization’s needs. 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. Both of these documents may be called the requirements document for the system.”

Functional Requirements

Functional requirements:

These are statements of services the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do.

The functional requirements for a system describe what the system should do. These requirements depend on the type of software being developed, the expected users of the software, and the general approach taken by the organization when writing requirements. When expressed as user requirements, functional requirements are usually described in an abstract way that can be understood by system users. However, more specific functional system requirements describe the system functions, its inputs and outputs, exceptions, etc., in detail.

Non-Functional Requirements

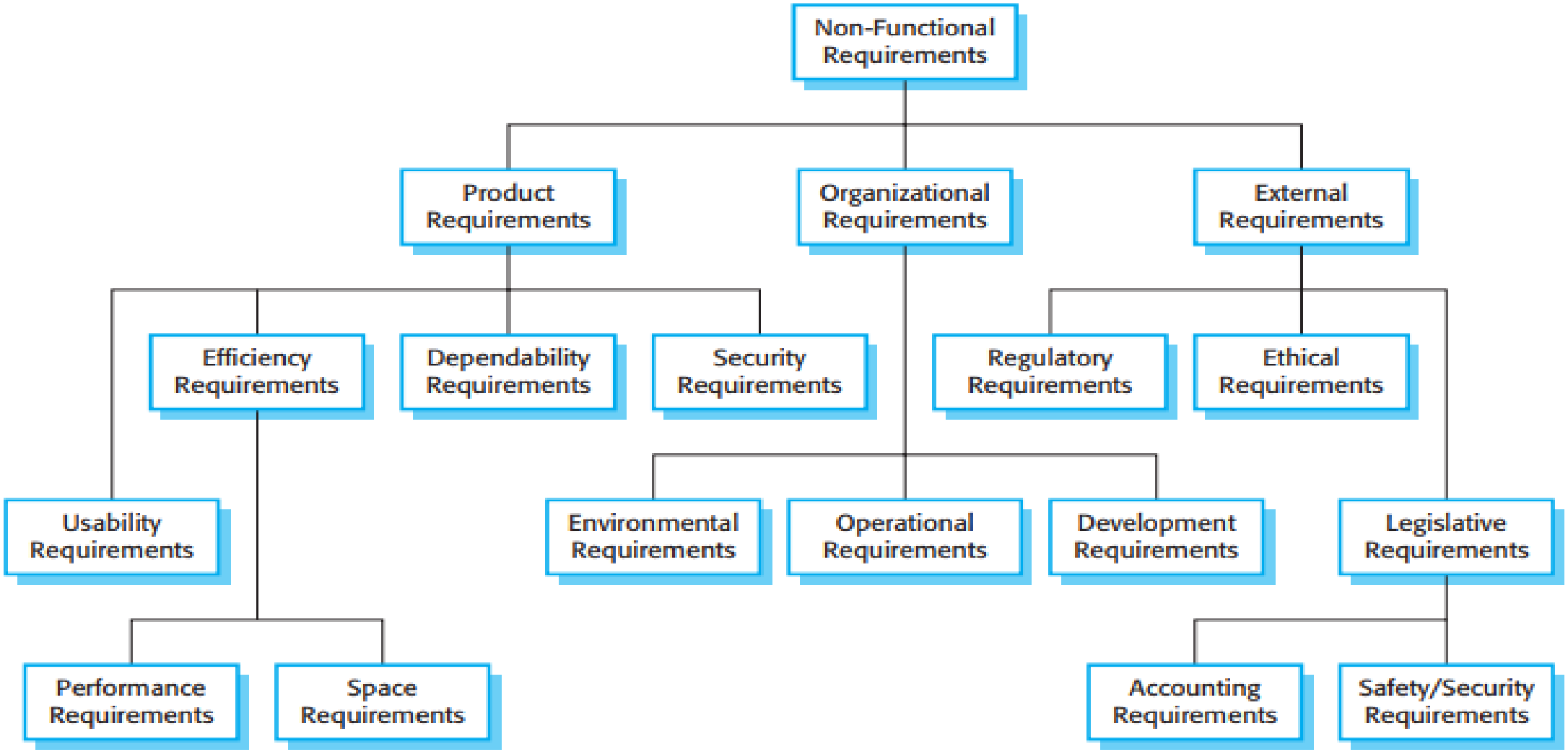
Non-functional requirements:

These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards. Non-functional requirements often apply to the system as a whole, rather than individual system features or services.

Non-functional requirements may affect the overall architecture of a system rather than the individual components. For example, to ensure that performance requirements are met, you may have to organize the system to minimize communications between components.

A single non-functional requirement, such as a security requirement, may generate a number of related functional requirements that define new system services that are required. In addition, it may also generate requirements that restrict existing requirements.

Non-Functional Requirements (Conti.)



Non-Functional Requirements (Conti.)

Non-functional requirements arise through user needs, because of budget constraints, organizational policies, the need for interoperability with other software or hardware systems, or external factors such as safety regulations or privacy legislation. Figure is a classification of non-functional requirements. You can see from this diagram that the non-functional requirements may come from required characteristics of the software (product requirements), the organization developing the software (organizational requirements), or from external sources:

1. *Product requirements*

These requirements specify or constrain the behavior of the software. Examples include performance requirements on how fast the system must execute and how much memory it requires, reliability requirements that set out the acceptable failure rate, security requirements, and usability requirements.

Non-Functional Requirements (Conti.)

2. *Organizational requirements*

These requirements are broad system requirements derived from policies and procedures in the customer's and developer's organization. Examples include operational process requirements that define how the system will be used, development process requirements that specify the programming language, the development environment or process standards to be used, and environmental requirements that specify the operating environment of the system.

3. *External requirements*

This broad heading covers all requirements that are derived from factors external to the system and its development process. These may include regulatory requirements that set out what must be done for the system to be approved for use by a regulator, such as a central bank; legislative requirements that must be followed to ensure that the system operates within the law; and ethical requirements that ensure that the system will be acceptable to its users and the general public.

Types of Requirement

User requirements

- Statements in natural language plus diagrams of the services the system provides and its operational constraints. Written for customers

System requirements

- A structured document setting out detailed descriptions of the system's functions, services and operational constraints. Defines what should be implemented so may be part of a contract between client and contractor

User 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.

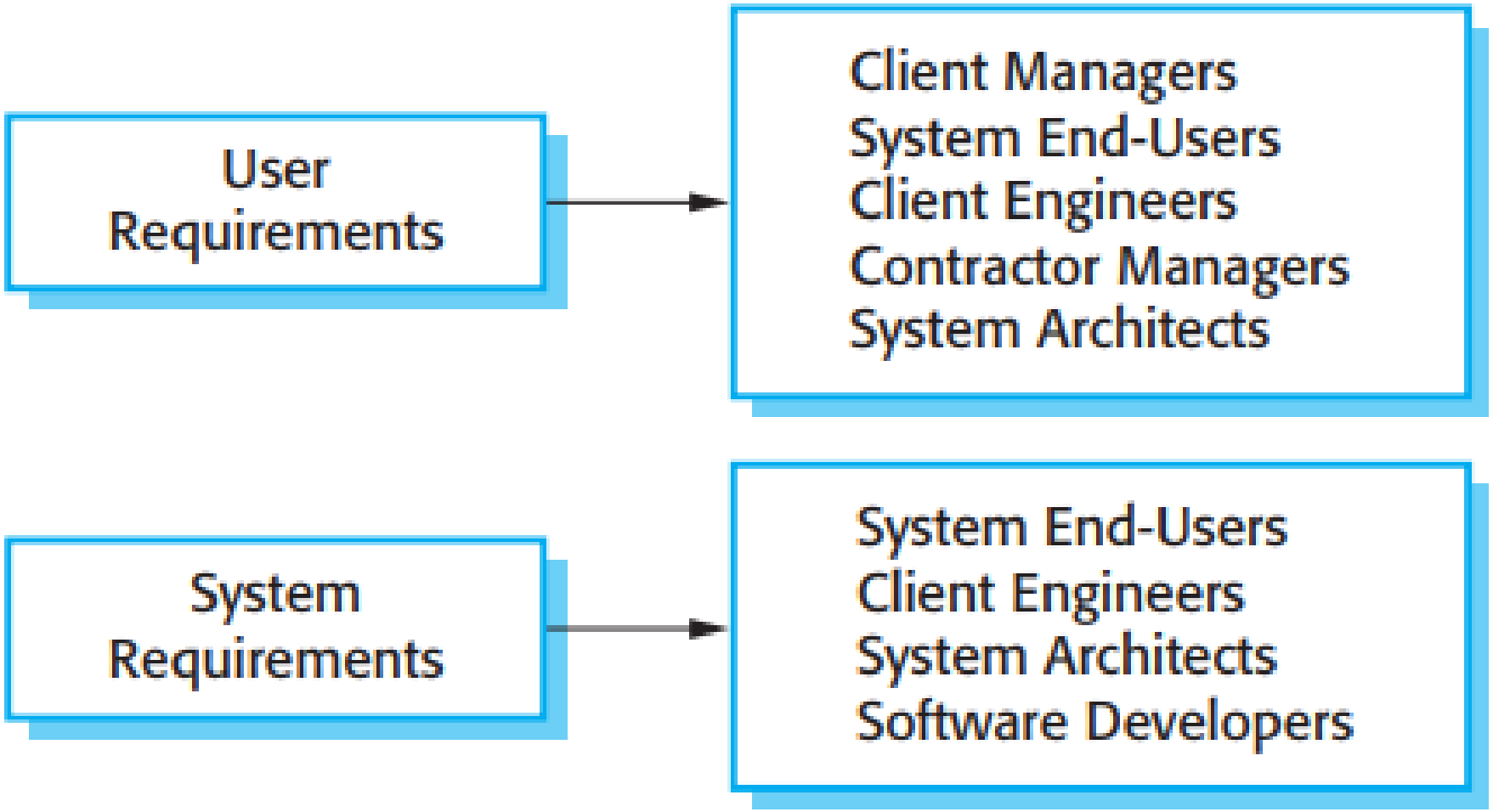
1.2 The system shall automatically generate the report for printing after 17.30 on the last working day of the month.

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.

Readers of different types of requirements specification



System Stakeholders

Any person or organization who is affected by the system in some way and so who has a legitimate interest

Stakeholder types

- End users
- System managers
- System owners
- External stakeholders

Stakeholders in Mental-Care System

Patients whose information is recorded in the system

Doctors who are responsible for assessing and treating patients

Nurses who coordinate the consultations with doctors and administer some treatments

Medical receptionists who manage patients' appointments

IT staff who are responsible for installing and maintaining the system

Medical records staff who are responsible for ensuring that system information can be maintained and preserved, and that record keeping procedures have been properly implemented

Requirements Engineering Processes

The processes used for RE vary widely depending on the application domain, the people involved and the organization developing the requirements

However, there are a number of generic activities common to all processes

- Feasibility Study
- Requirements Gathering
- Requirements Specification
- Requirements Validation

Feasibility Study

When the client approaches the organization for getting the desired product developed, it comes up with rough idea about what all functions the software must perform and which all features are expected from the software

Referencing to this information, the analysts does a detailed study about whether the desired system and its functionality are feasible to develop

The output of this phase is a feasibility study report that should contain adequate comments and recommendations for management about whether or not project should be undertaken

Requirements Gathering

If the feasibility report is positive towards undertaking the project, next phase starts with gathering requirements from the user

Analysts and engineers communicate with the client and end-users to know their ideas on what the software should provide, and which features they want the software to include

Requirements Specification

SRS is a document created by system analyst after requirements are collected from various stakeholders

SRS defines how the intended software will interact with hardware, external interfaces, speed of operation, response time of system, portability of software across various platforms, maintainability, speed of recovery after crashing, security, quality, limitations etc

Example of SRS Document

https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database

Ways of writing system 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.
Design description languages	This approach uses a language like a programming language, but with more abstract features to specify the requirements by defining an operational model of the system. This approach is now rarely used although it can be useful for interface specifications.
Graphical notations	Graphical models, supplemented by text annotations, are used to define 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

Structured Specification

Definition of the function or entity

Description of inputs and where they come from

Description of outputs and where they go to

Information about the information needed for the computation and other entities used

Description of the action to be taken

Pre and post conditions (if appropriate)

The side effects (if any) of the function

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.

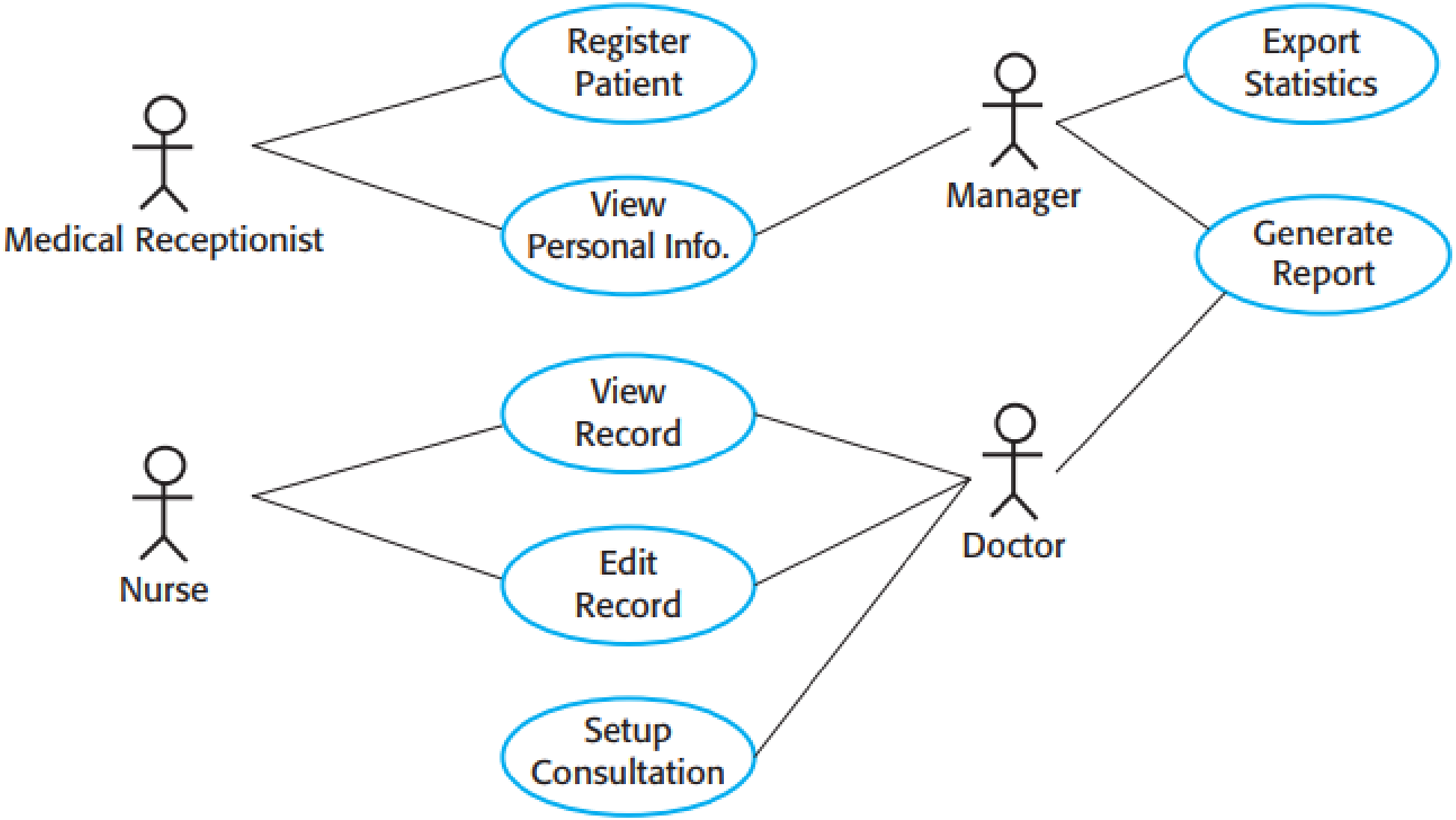
Pre-condition

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.

Use cases for the Mental-Care System



Requirements Validation

After requirement specifications are developed, the requirements mentioned in this document are validated. This results in huge increase in cost if not nipped in the bud Requirements can be checked against following conditions

Validity: Does the system provide the functions which best support the customer's needs?

Consistency: Are there any requirements conflicts?

Completeness: Are all functions required by the customer included?

Realism: Can the requirements be implemented given available budget and technology

Verifiability: Can the requirements be checked?

Requirements Elicitation Techniques

Requirements elicitation is the process to find out the requirements for an intended software system by communicating with client, system users and others who have a stake in the software system development

There are various ways to discover requirements

- Interviews
- Surveys
- Questionnaires
- Domain Analysis
- Brainstorming
- Prototyping

Exercise

Discover ambiguities or omissions in the following statement of requirements for part of a ticket-issuing system:

An automated ticket-issuing system sells rail tickets. Users select their destination and input a credit card and a personal identification number. The rail ticket is issued and their credit card account charged. When the user presses the start button, a menu display of potential destinations is activated, along with a message to the user to select a destination. Once a destination has been selected, users are requested to input their credit card. Its validity is checked and the user is then requested to input a personal identifier. When the credit transaction has been validated, the ticket is issued

Exercise

Discover ambiguities or omissions in the following statement of requirements for part of a ticket-issuing system:

Can a user buy more than one ticket for the same destination at the same time, or should the user buy them one by one?

Is it possible in the system that a user can cancel a ticket after making the transaction for the same?

What are the system behavior and output in the case of wrong card insertion?

What is the process if the user is required to buy another ticket for a different destination? Do they have to follow the same process from the beginning?