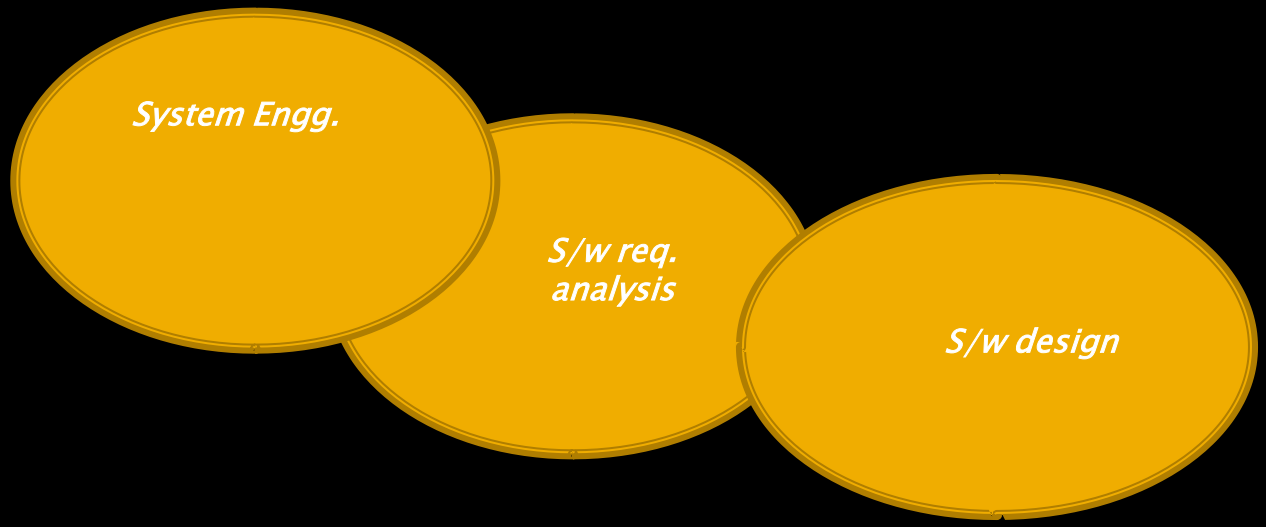
Lecture-15

Requirements Analysis Concepts & Principles

Dronacharya College of Engineering

Analysis Concepts and Principles

Requirement analysis is a software engineering task that bridges the gap between system engineering and software design.



Participated by both the customer and developer

Requirements Analysis Concepts and Principles

- *Requirements Analysis*
- *Communication Techniques*
    - *Initiating the Process*
    - *Facilitated Application Specification Techniques*
- *Analysis Principles*
    - *Information Domain*
    - *Modeling*
    - *Partitioning*
- *Software Prototyping*
    - *Selecting the Prototyping Approach*
    - *Prototyping Methods and Tools*
- *The Software Requirements Specification*
    - *Specification Principles*
    - *Representation*

1.Requirements Analysis

Requirements analysis

A process of discovery, refinement, modeling, and specification. During the process, both the developers and customers take an active role.

Focus on: ***"what" instead of "how"***

Input of the requirements analysis process:

- ***Software Project Plan***
- ***System specification (if one exists)***

Output:

Software requirements specification document

- provides the software engineer with models that can be translated in to data, architectural, interface, and procedure design.

- customer and developer can check the quality of the software and provide the feedback.

Who perform requirements analysis:

system analysts

Requirements Analysis

Major tasks and efforts:

- Problem recognition (or system understanding)
  - Discover and understand the system requirements
  - Refine the requirements
- Evaluation and synthesis:
  - what are the alternative solutions
  - focus on what solution should be selected or used instead of how to implement a solution.
- Modeling: to represent the various aspects of the system
  - the required data
  - information and control flow
  - operation behavior
- Specification of:
  - software functions, and performance
  - interfaces between system elements
  - system constraints

Requirements Analysis Process

- Domain understanding:

- Understanding of application domain.

- Requirements collections:

- The process of interacting with customers, users to discover the requirements for the system.

- Requirements classification:

- Group and classify the gathered requirements.

- Conflict resolution:

- Resolve the conflict requirements.

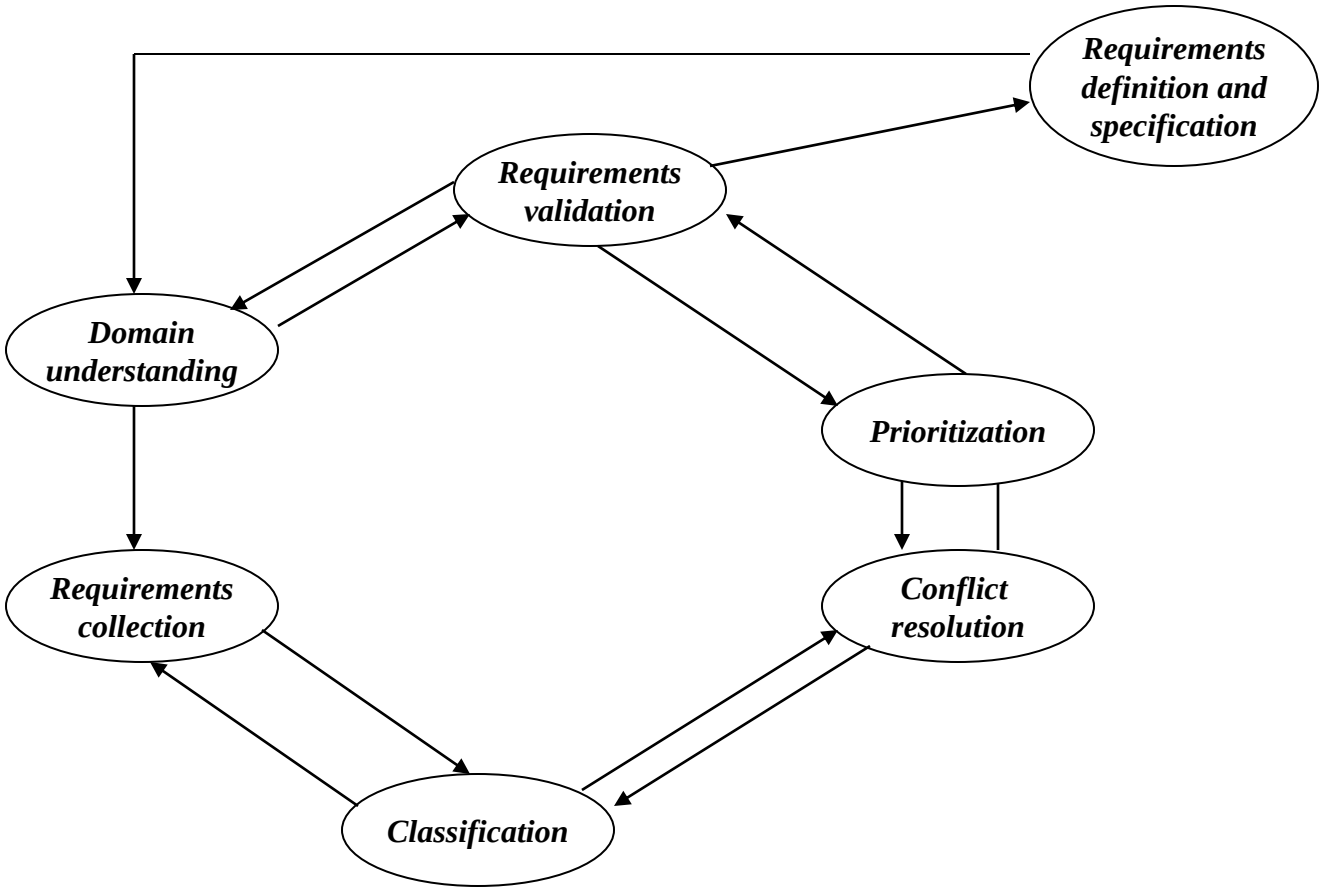
- Prioritization:

- Identify and list requirements according to their importance

- Requirements validation:

- Check and validate the gathered requirements to see if they are complete, correct, and sound.

Requirements Analysis Process



2.Communication Techniques

Initiating the Process:

Q1 set: Context free questions to lead the basic understanding of the problem

Who is behind the solution?
Who will use the solution?
…..

Q2 set: Questions to gain a better understanding of the problem and the customer's perceptions about a solution.

How would you characterize "good" output that would be generated by a successful solution?

What problems will this solution address?

Q3 set: Meta-questions focus on the effectiveness of the meeting.

Are you the right person to answer these questions? Are your answers "official"?

Facilitated Application Specification Techniques (FAST)

Customers and software engineers often have an unconscious "us and them" mind set.

This may cause: misunderstandings, miss important information,….

To solve the problem, FAST approach is proposed.

FAST encourages the creation of a joint team of customers and developers.

They work together

- ***to identify the problem and proposed and***
- ***to negotiate the different elements of solutions and approaches***

The basic guidelines of FAST:

- ***hold a meeting at a neutral site***
- ***establish rules for preparation and participation***
- ***have a formal meeting agenda***
- ***control the meeting by a "facilitator"***
- ***use a "definition mechanisms"***
- ***have a common goal to***
  - ***identify the problem***
  - ***propose elements of solutions and requirements***
  - ***negotiate different approaches***

Quality Function Deployment

Quality function deployment (QFD) is quality management technique

- Translate the needs of the customer into technical requirements for software.

QFD identifies three types of requirements:

- Normal requirements:

Objectives and goals:

examples: types of graphic displays, specific system functions

- Expected requirements:

implicit requirements:

examples: ease of human-machine interaction
ease of software installation

- Exciting requirements:

Features go beyond the customer's expectations

Use Cases

- *Scenarios that describe how the product will be used in specific situations.*
- *Written narratives that describe the role of an actor (user or device) as it interacts with the system.*
- *Use-cases designed from the actor's point of view.*
- *Not all actors can be identified during the first iteration of requirements elicitation, but it is important to identify the primary actors before developing the use-cases.*

User Profile - Example

- *Full Control (Administrator)*
- *Read/Write/Modify All (Manager)*
- *Read/Write/Modify Own (Inspector)*
- *Read Only (General Public)*

Use Case Example - 1

- **Read Only Users**
  - *The read-only users will only read the database and cannot insert, delete or modify any records.*

- **Read/Write/Modify Own Users**
  - *This level of users will be able to insert new inspection details, facility information and generate letters. They will be also able to modify the entries they made in the past.*

Use Case Example - 2

- **Read/Write/Modify All Users**
  - *This level of users will be able to do all the record maintenance tasks. They will be able to modify any records created by any users.*

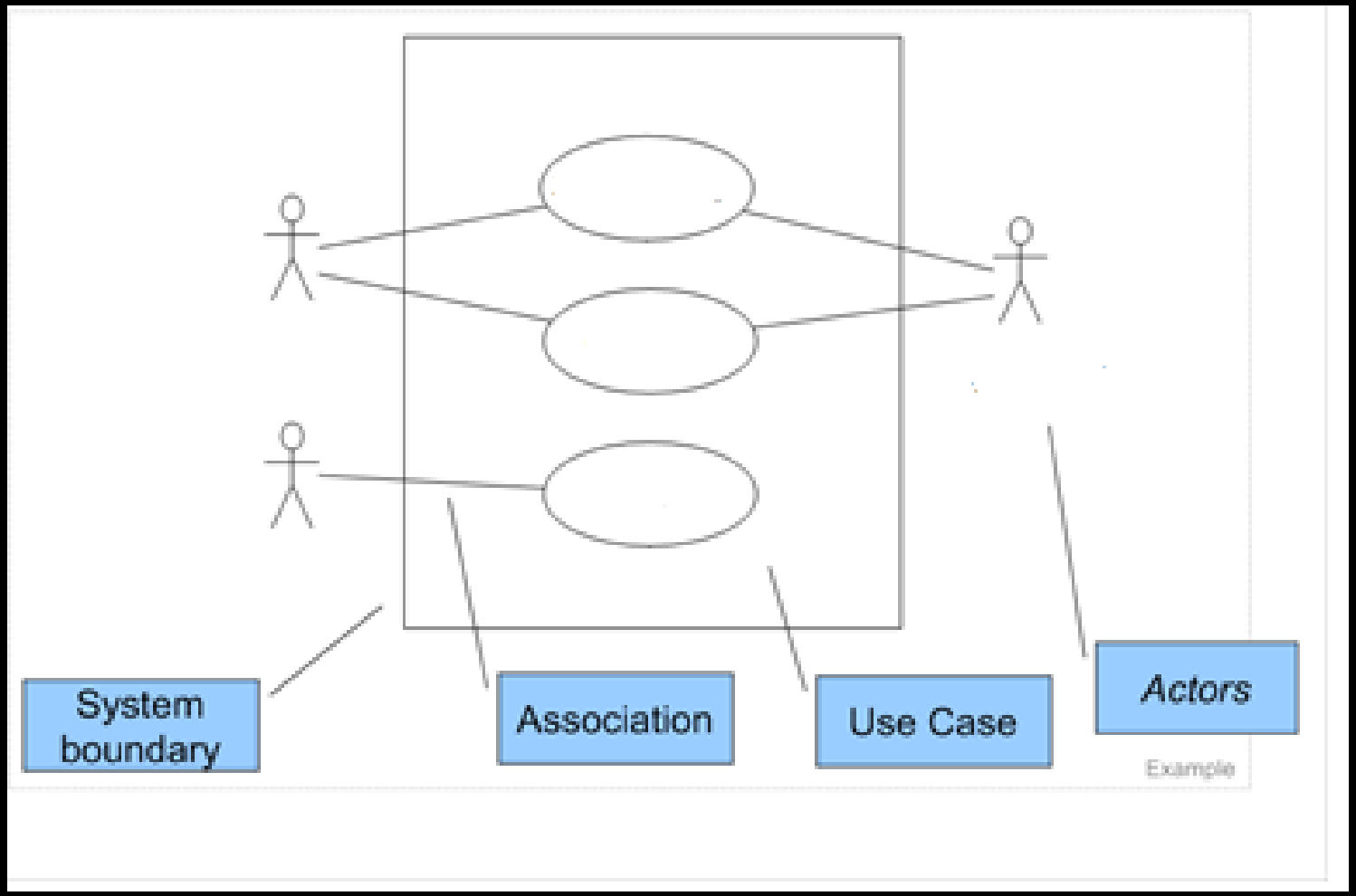
- **Full Control Users**
  - *This is the system administrative level which will be able to change any application settings, as well as maintaining user profiles.*

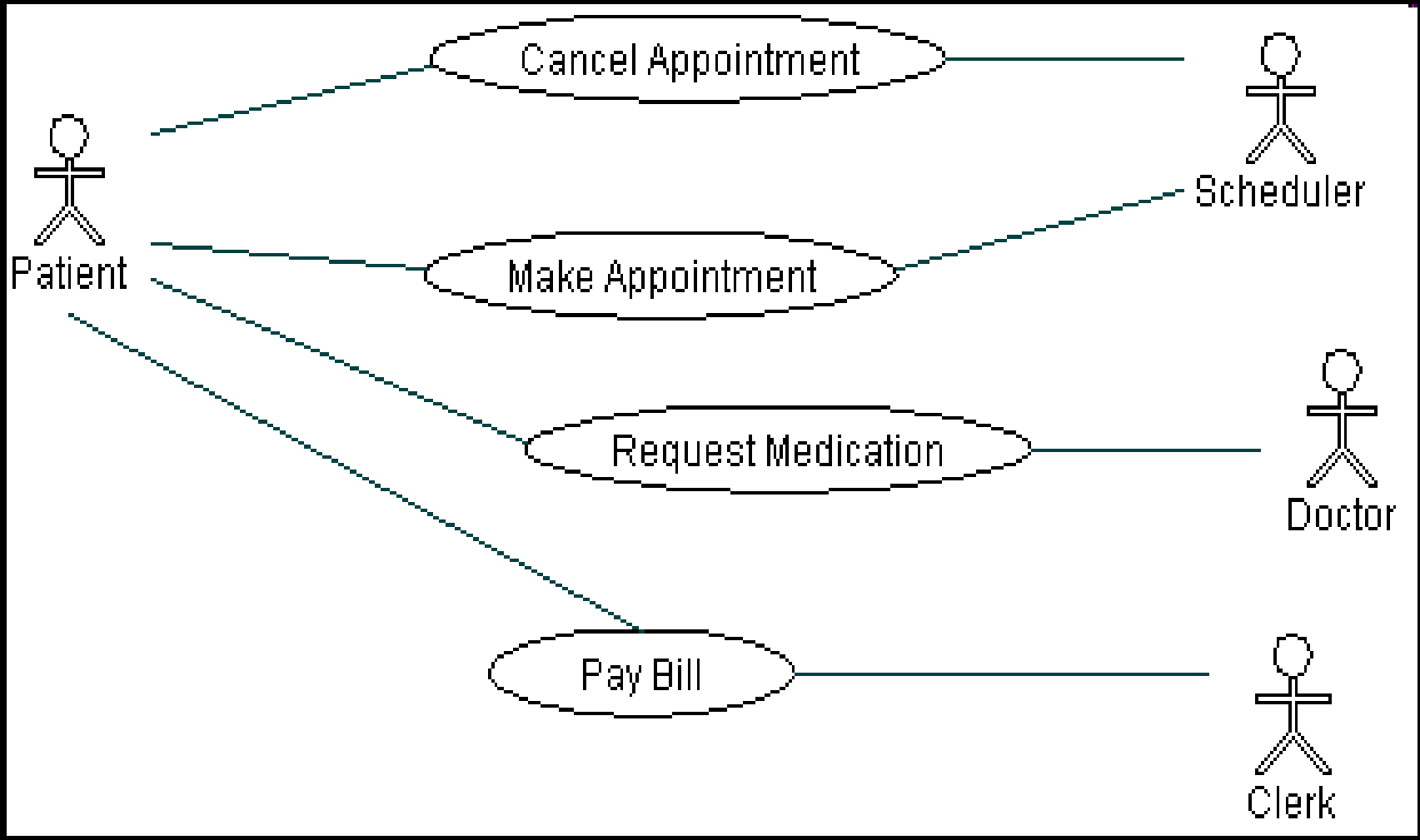
Use Case

It describes the sequence of interactions between actors and the system necessary to deliver the service that satisfies the goal.

How to create a use case ?

- **Identify the different users (actors) of the system.**
- **Create use cases which captures who (actor) does what (interaction) without dealing with the system internals.**

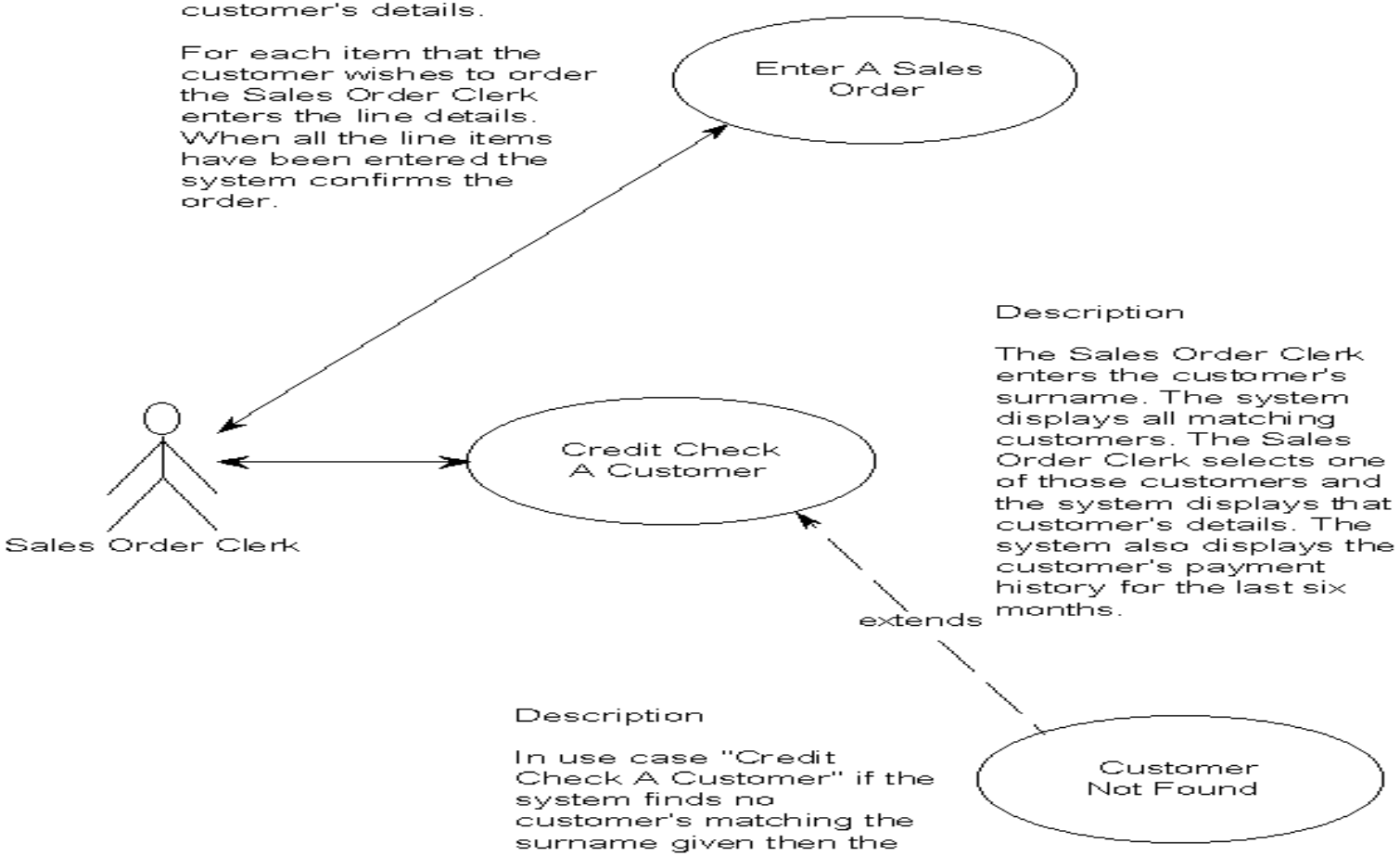

System boundary
Association
Use Case
Actors
Example


Cancel Appointment
Scheduler
Patient
Make Appointment
Request Medication
Doctor
Pay Bill
Clerk

Description

The Sales Order Clerk enters the customer's surname. The system displays all matching customers. The Sales Order Clerk selects one of those customers and the system displays that customer's details.

For each item that the customer wishes to order the Sales Order Clerk enters the line details. When all the line items have been entered the system confirms the order.

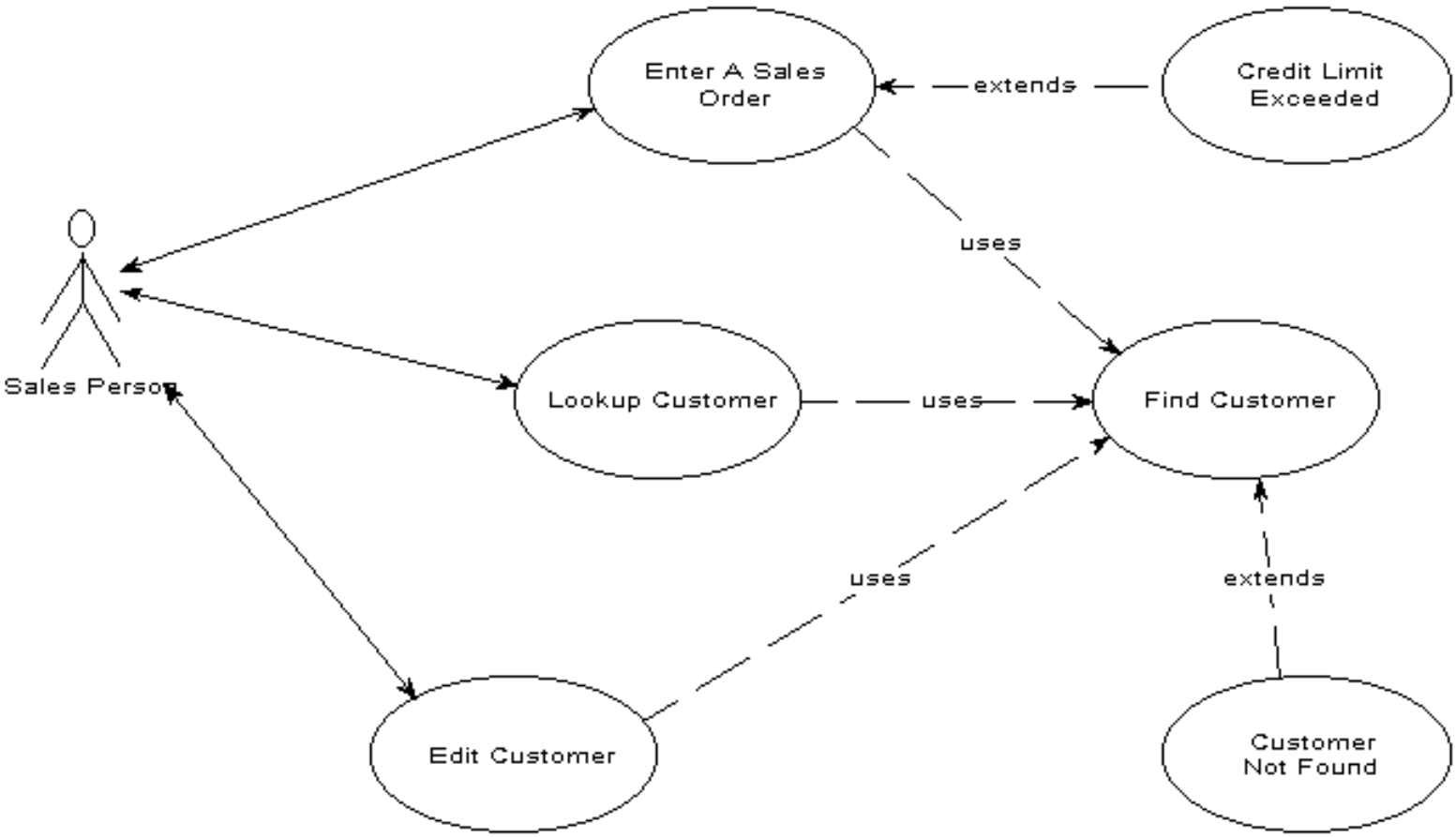


Description

The Sales Order Clerk enters the customer's surname. The system displays all matching customers. The Sales Order Clerk selects one of those customers and the system displays that customer's details. The system also displays the customer's payment history for the last six months.

Description

In use case "Credit Check A Customer" if the system finds no customer's matching the surname given then the system displays an error message and allows the user to enter a different surname to search against.


Enter A Sales Order
extends
Credit Limit Exceeded
uses
Sales Person
Lookup Customer
uses
Find Customer
uses
extends
Edit Customer
Customer Not Found